



UNIVERSITAT POLITÈCNICA  
DE CATALUNYA

# TRABAJO DE FIN DE CARRERA

**TITULO DEL TFC:** Creación de Sistemas Android Personalizados

**TITULACION:** Ingeniería Técnica de Telecomunicación, especialidad en  
Telemática

**AUTOR:** Miguel Francisco Núñez Burguera

**DIRECTOR:** Juan López Rubio

**FECHA:** 20 de enero de 2011



**Titulo:** Creación de sistemas Android personalizados

**Autor:** Miguel Francisco Núñez Burguera

**Director:** Juan López Rubio

**Fecha:** 20 de enero de 2011

## **Resumen**

En este proyecto se pretende obtener un conocimiento más detallado sobre la plataforma Android. Para ello se ha propuesto la creación de una versión modificada del sistema operativo (ROM) para instalar en un terminal GOOGLE NEXUS ONE que pueda tener utilidad para los usuarios de la EETAC.

La primera fase consiste en la preparación de todas las herramientas y software necesario para la programación de una ROM en Android. En esta fase se desea obtener unos conocimientos básicos de las herramientas para poder empezar la compilación y modificación del código fuente de Android.

La segunda fase trata de programar en el entorno Eclipse con los añadidos de Android, para crear un widget de la EETAC y traducir diversas aplicaciones para la ROM.

Finalmente, como última y tercera fase, es necesaria una revisión final de la ROM pensando en futuras ampliaciones y/o modificaciones que permitan mejorarla para el usuario final así como también comparar la ROM creada con otras existentes en el mercado.

**Title:** Creación de Sistemas Android Personalizados

**Author:** Miguel Francisco Núñez Burguera

**Director:** Juan López Rubio

**Date:** January, 20th 2011

## Overview

In this project we intend to obtain a more detailed understanding of the Android platform. For this it has been proposed to create a modified version of the operative system (ROM) to install in a Google Nexus One terminal that can be utilised by the users from EETAC.

The first phase consists in the preparation of all the tools and software necessary for the ROMs programming in Android. In this phase we wish to obtain some basic knowledge of the tools to there fore be able to start the compilation and modification of the source code of Android.

The second phase is based on programming the Eclipse platform with the add-ons from Android, to create an EETAC widget and translate differents applications for the ROM.

Finally, as our third and last phase, it s' necessary to carry out a final revision of the ROM thinking in future modifications and/or upgrades and compare the ROM with other existent ones on the market.

Dedico este trabajo final de carrera a mi madre, porque sin su esfuerzo para que estudiara no habría llegado hasta aquí, a todos mis amigos y compañeros que tantos días y noches nos hemos quedado estudiando y a todos los profesores que he tenido que me han transmitido parte de su saber,  
Gracias a todos vosotros que me habéis enseñado que la vida es un aprendizaje constante y esto es solo el comienzo.



# INDICE

<b>INTRODUCCIÓN .....</b>	<b>1</b>
<b>1. INTRODUCCION A ANDROID.....</b>	<b>4</b>
1.1.    El sistema operativo Android.....	4
1.2.    El crecimiento de Android.....	4
1.3.    Características principales.....	6
1.4.    Arquitectura interna .....	7
1.4.1.    Aplicaciones.....	7
1.4.2.    Armazón de aplicaciones.....	7
1.4.3.    Librerías .....	8
1.4.4.    Runtime de Android .....	8
1.5.    Android en el teléfono .....	8
1.6.    Las interfaces usadas por los fabricantes.....	9
1.6.1.    Interfaz en el Nexus One .....	9
1.6.2.    Interfaz Sense.....	11
1.6.3.    Interfaz Motoblur .....	12
1.6.4.    Interfaz Touchwiz.....	13
1.6.5.    Interfaz Rachel.....	13
1.6.6.    Interfaz MIUI .....	14
1.7.    Conclusiones:.....	15
<b>2. LAS ROMS EN ANDROID .....</b>	<b>16</b>
2.1.    El concepto de ROM .....	16
2.2.    Estructura general de una ROM.....	16
2.3.    Herramientas para la creación y gestión de una ROM .....	18
2.3.1.    Android Debug Bridge (ADB).....	18
2.3.2.    Monkey .....	19
2.3.3.    Repo y Git.....	19
2.3.4.    Fastboot.....	19
2.4.    Descargar el código fuente de Android .....	20
2.5.    Instalar una ROM.....	22
2.5.1.    Instalar una ROM con la herramienta fastboot .....	22
2.5.2.    instalación de una ROM en modo recovery.....	24
2.5.3.    Los archivos update-script y updater-script .....	28
2.6.    Conclusiones.....	29
<b>3. LA ROM MODIFICADA.....</b>	<b>30</b>
3.1.    Radio FM en el móvil HTC Nexus One .....	30
3.2.    El consumo de energía del terminal.....	31

3.2.1. Diferencia de tiempo en la duración de la batería .....	32
<b>3.3. Modificaciones generales de la ROM .....</b>	<b>33</b>
3.3.1. Cambiar los tonos por defecto en la ROM.....	33
3.3.2. Cambiar la animación de inicio en la ROM.....	34
3.3.2.1. Componentes del bootanimation.zip.....	34
3.3.3. Cambiar el fondo de pantalla por defecto en la ROM.....	34
3.3.4. Cambiar al apariencia de las aplicaciones.....	35
<b>3.4. Cargar la configuración .....</b>	<b>38</b>
3.4.1. Los archivos de configuración de las aplicaciones .....	39
3.4.2. Los archivos de configuración del sistema .....	39
<b>3.5. Gestión de las aplicaciones instaladas en la ROM .....</b>	<b>40</b>
3.5.1. Aplicaciones situadas en /system/app .....	40
3.5.2. Aplicaciones situadas en /data/app .....	41
3.5.3. Aplicaciones incluidas en la ROM.....	41
<b>3.6. Conclusiones.....</b>	<b>45</b>
<b>4. PROGRAMACIÓN EN ANDROID.....</b>	<b>46</b>
<b>4.1. Estructura de una aplicación .....</b>	<b>46</b>
4.1.1. Activity.....	46
4.1.2. Intent Receiver.....	46
4.1.3. Service .....	46
4.1.4. Content Provider.....	47
4.1.5. AndroidManifest.xml .....	47
<b>4.2. Programación de widgets para la EETAC.....</b>	<b>47</b>
4.2.1. Funcionamiento de los Widgets .....	48
<b>4.3. Gestión de una aplicación en diferentes idiomas.....</b>	<b>51</b>
<b>4.4. Generar un .apk .....</b>	<b>53</b>
<b>4.5. Conclusiones.....</b>	<b>57</b>
<b>5. CONCLUSIONES FINALES.....</b>	<b>58</b>
<b>5.1. Conocimientos adquiridos .....</b>	<b>58</b>
<b>5.2. Comparativa de la ROM creada con otras existentes en el mercado .....</b>	<b>58</b>
<b>5.3. Visión de futuro del sistema operativo Android .....</b>	<b>59</b>
<b>5.4. Estudio medioambiental.....</b>	<b>59</b>
<b>BIBLIOGRAFÍA .....</b>	<b>60</b>
<b>ANEXOS .....</b>	<b>62</b>
<b>I. Versiones de Android.....</b>	<b>62</b>
a.     Android 1.5 (Cupcake).....	62
b.     Android 1.6 (Donut).....	62
c.     Android 2.0 / 2.1 (Éclair) .....	63
d.     Android 2.2 (Froyo) .....	63

e.	Android 2.3 (Gingerbread) .....	64
f.	Android 3.0 (Honeycomb) .....	65
<b>II. Dalvik vs Java Virtual Machine.....</b>	<b>65</b>	
<b>III. Uso de herramientas para la creación de una ROM.....</b>	<b>66</b>	
a.	ADB.....	66
b.	Monkey .....	68
c.	Fastboot.....	69
<b>IV. Características de las ROMS de Cyanogen.....</b>	<b>70</b>	
<b>V. Habilitar los privilegios de administrador en el terminal.....</b>	<b>71</b>	
<b>VI. Ciclo de vida de una aplicación Android.....</b>	<b>73</b>	
a.	Foreground process (proceso de primer plano).....	74
b.	Visible process (proceso visible).....	74
c.	Service process (proceso de servicio).....	74
d.	Background process (proceso de fondo).....	74
e.	Empty process (proceso vacío) .....	74



## INTRODUCCIÓN

El sistema operativo Android esta cobrando cada día mayor importancia en el mercado de la telefonía móvil, debido a que cada día más fabricantes lo usan en sus teléfonos. Además es muy probable que se empiece a usar Android en dispositivos en los que hasta ahora no se utilizaba este sistema operativo porque Android es libre y gratuito, dando a los fabricantes la posibilidad de modificarlo y adaptarlo a las características de sus dispositivos con un coste muy bajo y con unos usuarios que se ven atraídos por la fama de este sistema operativo.

Con este panorama cada vez aparecen en el mercado más adaptaciones del sistema operativo hechas por los diferentes fabricantes, diferenciando así sus productos de los de la competencia. El ejemplo más conocido de esto seguramente sea la interfaz Sense que usa HTC en la mayoría de sus terminales aunque también existen otras menos conocidas como son Rachel que es la que usa Sony en sus teléfonos, la interfaz Touchwiz de Samsung o incluso las creadas por desarrolladores, también denominados “cocineros”, haciendo sus propias adaptaciones del código de Google.

Aprovechando que cada vez resulta más importante para los fabricantes poder personalizar y adaptar el código a sus productos, que el código sea libre y que es una área poco investigada en estos momentos, con este TFC se tratará de estudiar las diferentes posibilidades de personalización y las diferentes maneras para generar una versión adaptada del sistema operativo Android, así como los métodos de instalación en un teléfono HTC Google Nexus One. Para todo ello la memoria queda dividida en los siguientes capítulos:

En el primer capítulo se describe y estudia el sistema operativo Android. Para ello se muestran sus características, estructura y funcionalidades se hace una comparativa entre las Interfaces de los fabricantes y se compara con la que tiene el Nexus One de serie.

En el segundo capítulo se describe el concepto de ROM, se muestra la estructura general de una ROM así como las herramientas para la descarga y compilación del código fuente también se estudian los métodos de instalación de una ROM para el móvil Nexus One.

El tercer capítulo se centra en las características particulares de la ROM que se ha creado en este trabajo final de carrera, en especial las modificaciones realizadas a las aplicaciones que vienen instaladas, y los archivos de configuración de la ROM.

El cuarto capítulo explica la estructura de las aplicación en Android así como su programación en el entorno eclipse, y se explica el funcionamiento de los widgets creados para la ROM.

En un quinto y último capítulo se hacen unas conclusiones, se comparan características de la versión creada con otros productos existentes en el mercado y se da la visión de futuro de Android.



## 1. INTRODUCCION A ANDROID

En esta capítulo se introducirá al sistema operativo Android, se repasara brevemente su historia, se mostrara el gran crecimiento que esta experimentando este sistema, también se explicara como esta formado y las diferentes partes que lo componen, una vez hecho esto se compararan las interfaces de los diferentes fabricantes tratando de mostrar sus características y la experiencia que tiene el usuario al trabajar con cada una.

### 1.1. *El sistema operativo Android*

Android es un sistema operativo para dispositivos móviles. Está basado en Linux 2.6e inicialmente fue desarrollado por Android Inc., compañía que fue comprada después por Google, y en la actualidad lo desarrollan los miembros de la Open Handset Alliance (liderada por Google). Esta plataforma es de código abierto, lo que permite el desarrollo de aplicaciones por terceros bajo una licencia Apache, versión 2. Se gestiona mediante el lenguaje de programación Java y los dispositivos móviles se controlan por medio de bibliotecas creadas y/o adaptadas por Google.

La presentación de la plataforma Android se realizó el 5 de noviembre de 2007 junto con la fundación Open Handset Alliance, un consorcio de 48 compañías de hardware, software y telecomunicaciones comprometidas con la promoción de estándares abiertos para dispositivos móviles.

### 1.2. *El crecimiento de Android*

En este apartado se estudia el aumento en el uso de dispositivos Android, de este modo se intenta justificar y dar respuesta a la pregunta, de en que versiones del sistema operativo resulta más rentable desarrollar dichas aplicaciones.

Si se desea consultar con detalle las diferencias entre cada una de las versiones del sistema operativo se puede consultar en el [apartado de 1 de anexos](#) en donde se detallan en profundidad sus características.

El mercado de la telefonía móvil es hoy por hoy uno de los mercados más grandes del mundo, dentro de este el que tiene más crecimiento es el de los teléfonos llamados Smartphone, una unión entre una PDA y un móvil. Como ejemplo de lo grande que resulta este mercado las cifra de ventas en 2009 fueron de 172 millones de smartphones en todo el mundo.

Dentro de los Smartphones los que más crecimiento a tenido en estos últimos años han sido IOS y Android siendo este último el que a registrado más ventas en el año 2010 según la empresa analista Canalys tal y como se puede ver en la figura 1.1.

OS vendor	Q4 2010		Q4 2009		Growth Q4'10/Q4'09
	shipments (millions)	% share	shipments (millions)	% share	
Total	101.2	100.0%	53.7	100.0%	88.6%
Google	33.3	32.9%	4.7	8.7%	615.1%
Nokia	31.0	30.6%	23.9	44.4%	30.0%
Apple	16.2	16.0%	8.7	16.3%	85.9%
RIM	14.6	14.4%	10.7	20.0%	36.0%
Microsoft	3.1	3.1%	3.9	7.2%	-20.3%
Others	3.0	2.9%	1.8	3.4%	64.8%

Source: Canalys estimates, © Canalys 2011

Fig. 1.1 Ventas de Smartphones a nivel mundial

En el gráfico de la figura 1.2 se puede observar el aumento del uso del sistema operativo Android en el mercado de Smartphones en Estados Unidos, siendo el mercado estadounidense un claro indicador a nivel mundial:

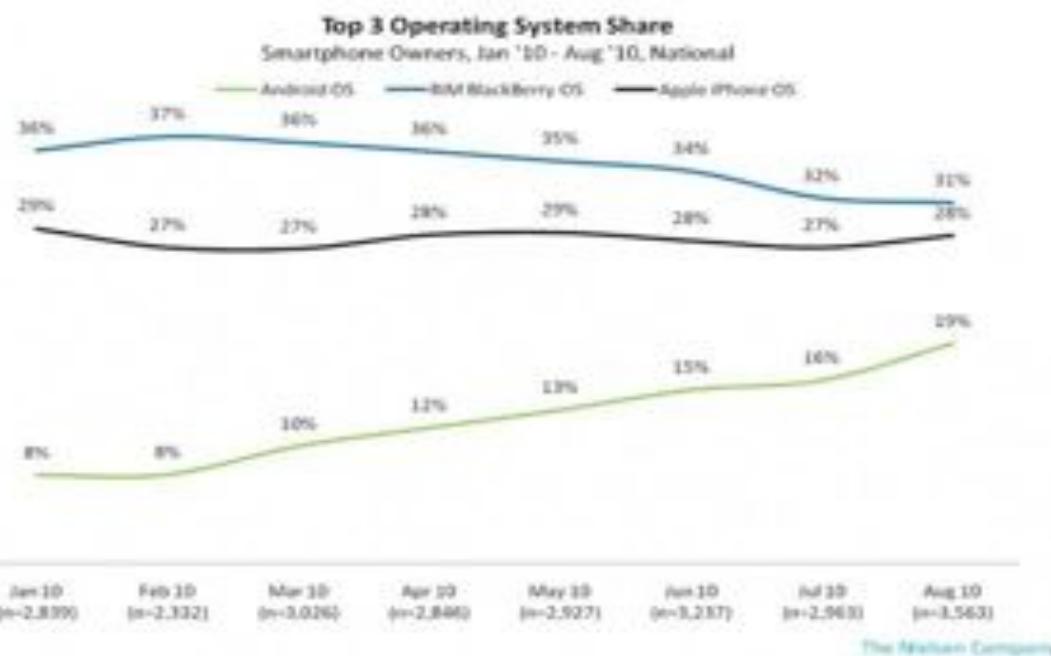


Fig. 1.2 Evolución del mercado de Smartphones en EEUU

Android es un sistema operativo que dispone de muchas versiones pues ha sido actualizado cada poco tiempo. Esto que en principio podría ser un inconveniente para el desarrollador; ya que lo que puede ser compatible con una versión a lo mejor no lo es para otra. Se a comprobado que en la práctica

esta segmentación casi no afecta, pues las diferencias a nivel de desarrollo son muy pocas, además casi todo el mercado suele estar dominado por una o dos versiones distintas tal y como se puede observar en el gráfico de la figura 1.3 y es a estas versiones en las que los desarrolladores deben dirigir sus aplicaciones para tener un mayor número de usuarios.

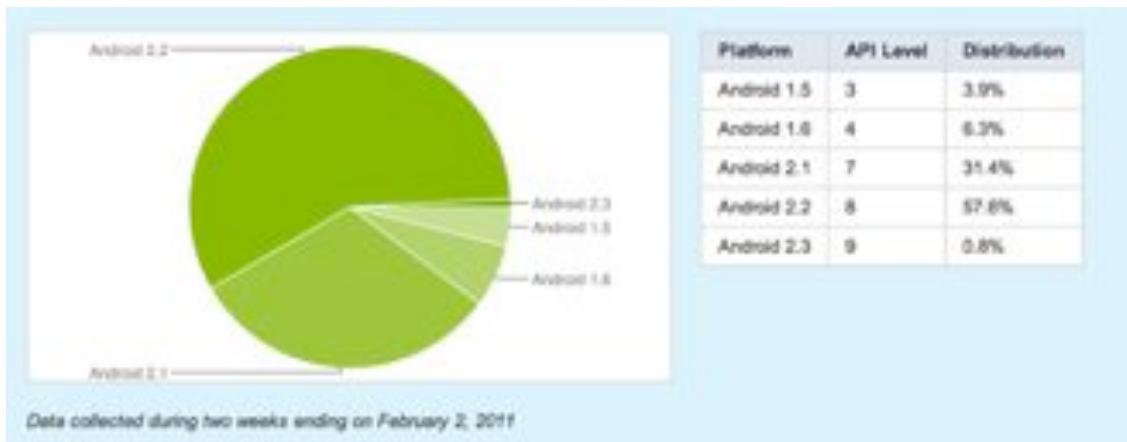


Fig. 1.3 Distribución de las diferentes versiones de Android

### 1.3. *Características principales*

Una de sus principales características de este sistema operativo es que está pensado para optimizar recursos, un punto a tener en cuenta ya que al ser utilizado en terminales móviles es primordial no consumir demasiada batería.

Para ejecutar las aplicaciones Java no se utiliza la Java Virtual Machine (JVM) ya que está bajo licencia de Sun Microsystems y tiene restricciones. La máquina virtual que utiliza Android es creación propia de Dan Bornstein y otros ingenieros de Google, su nombre es Dalvik y está pensada y diseñada para que utilice poca memoria y pueda ejecutar varias instancias simultáneamente. En el [apartado 2 de los anexos](#) se explica con más detalle la Dalvik Virtual Machine.

Android incluye una base de datos propia llamada SQLite, que permite almacenar datos de forma local. Tratándose de una base de datos local, está limitada a información guardada en el propio móvil pero aporta una facilidad y sencillez a la hora de ser utilizada e integrada en las aplicaciones.

Otra de las características, es que al ser un sistema operativo orientado a móviles con múltiples aplicaciones multimedia, soporta diferentes formatos de audio, vídeo e imágenes como MPEG4, MP3, 3GP, JPG, GIF.

## 1.4. Arquitectura interna

Como cualquier sistema operativo, Android está formado por diferentes capas, esto se puede observar en la figura 1.4, partiendo del kernel basado en Linux, pasando por las librerías y el Runtime, el armazón de las aplicaciones y finalmente las aplicaciones.

A continuación se explica detalladamente la función de cada una de ellas.

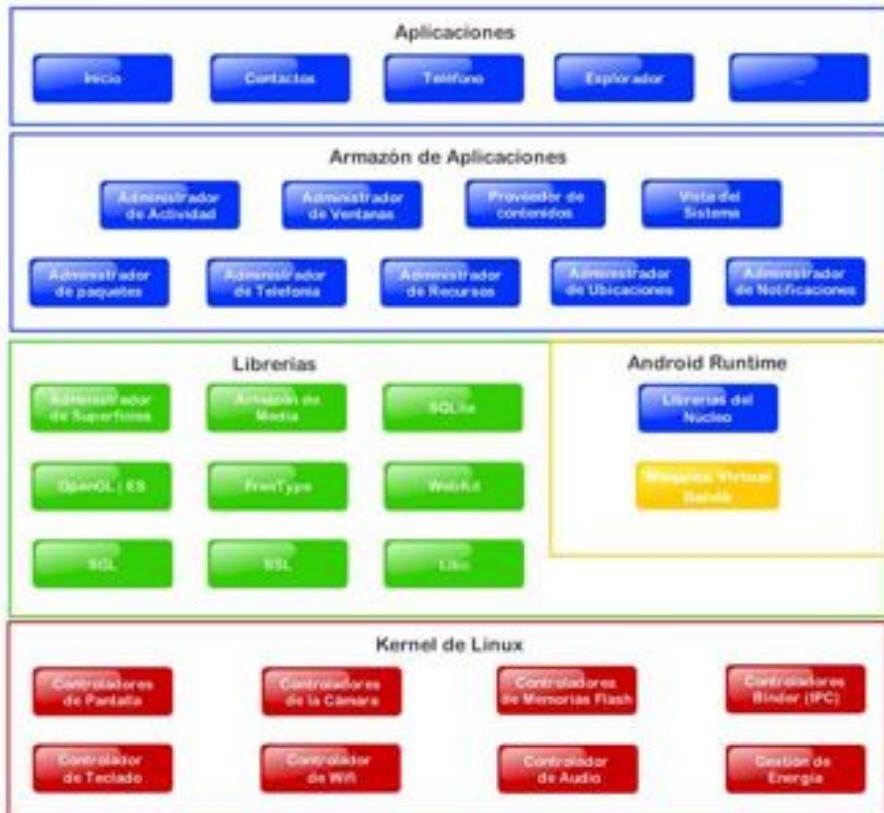


Fig. 1.4 Arquitectura del sistema operativo Android

### 1.4.1. Aplicaciones

Todas las aplicaciones creadas con la plataforma Android, incluirán como base un cliente de email (correo electrónico), calendario, programa de SMS, mapas, navegador, contactos, y algunos otros servicios mínimos. Todas ellas escritas en el lenguaje de programación Java.

### 1.4.2. Armazón de aplicaciones

Todos los desarrolladores de aplicaciones Android, tienen acceso total al código fuente usado en las aplicaciones base. Esto ha sido diseñado de esta forma para que no se generen cientos de componentes de aplicaciones distintas (que respondan a la misma acción) dando la posibilidad de que los programas sean modificados o reemplazados por cualquier usuario sin tener que empezar a programar sus aplicaciones desde el principio.

#### **1.4.3. Librerías**

Android incluye en su sistema un set de librerías C/C++, que son expuestas a todos los desarrolladores a través del Framework de las aplicaciones Android, librería C del sistema, librerías de medios, librerías de gráficos, 3D, SQLite, etc.

#### **1.4.4. Runtime de Android**

Android incorpora un set de librerías que aportan la mayor parte de las funcionalidades disponibles en las librerías base del lenguaje de programación Java. La Máquina Virtual está basada en registros, y ejecuta clases compiladas por el compilador de Java que anteriormente han sido transformadas al formato .dex (Dalvik Executable) por la herramienta "dx".

### **1.5. *Android en el teléfono***

El sistema operativo Android en el teléfono esta dividido en particiones, de la memoria NAND, esto queda reflejado en la figura 1.5, en ella se puede ver las diferentes particiones que tiene el móvil y las relación que tienen la partición System con el directorio /system del móvil y la partición Userdata con el directorio /data del terminal. Más adelante se expondrá como se pueden modificar estas particiones mediante la ROM y el efecto que tendrán estos cambios en el teléfono.

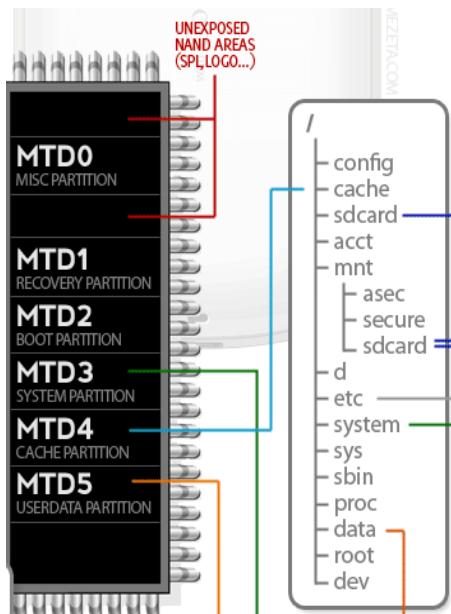


Fig. 1.5 Particiones

Las particiones más importantes son:

- la partición de Recovery, se encuentra el sistema de recuperación del móvil.
- la partición Boot, están el Kernel y el sistema de arranque del terminal.
- la partición System, es la parición que contiene el sistema de Android.
- la partición Userdata, contiene la configuración del usuario y de las aplicaciones

## 1.6. *Las interfaces usadas por los fabricantes*

En este apartado se van a detallar las características de las principales interfaces, que dispositivos las usan, sus características principales, las aplicaciones que incluyen.

### 1.6.1. Interfaz en el Nexus One

La Interfaz usada en el teléfono móvil Nexus One es considerada la interfaz de “serie”, es decir sin modificar, por parte de Google. Se considera que esto es así porque el teléfono móvil HTC Nexus One fue fabricado por HTC pero bajo las especificaciones de Google y sin incluir la interfaz Sense que es la que hasta esa fecha usaban casi todos los teléfonos de HTC.

En este móvil la animación de inicio se basa en los colores de Google, además de incluir un vistoso fondo animado también con los mismos colores. En el teclado tiene un diseño muy parecido al de Sense, con predicción de palabras.

El sistema de marcación del teléfono y gestión de contactos no es demasiado bueno ya que no dispone de predicción de contactos, aunque si que muestra las caras de los contactos si se han introducido en la agenda.

Para desbloquear el terminal aparecen dos barras que desplazándolas horizontalmente permiten desbloquear el teléfono o bien silenciarlo.

El nivel de widgets que lleva incorporados es muy básico. Los widgets que esta interfaz incluye son:

- Búsqueda de Google, que permite buscar dentro del terminal o bien directamente en Google.
- Calendar, en el que se muestran la fecha actual así como la cita más próxima.
- Consejos iniciales, un widget que sirve como tutorial para dar los primeros pasos en con el sistema operativo
- Control de energía, un widget muy útil que permite gestionar desde el escritorio las diferentes interficies de comunicación (Wifi, Bluethooth, GPS, luminosidad de la pantalla...)
- Noticias y tiempo, que es widget de en el que se pueden consultar las noticias y el tiempo o solo una de estas dos a la vez.
- Música, que proporciona el control de la música del móvil
- Reloj, es un reloj analógico que muestra la hora y si se pulsa da acceso a las alarmas del móvil.
- Youtube, permite buscar videos en el youtube, recomienda videos para ver, así como también puede subir videos directamente al youtube.
- Picture frame, un widget que permite poner en el escritorio una foto de la galería.

La cámara es muy básica, permite pocos ajustes y no deja enfocar pulsando ni tiene reconocimiento de caras.

El launcher (es una aplicación encargada de la gestión de los escritorios y organizar y listar todas las aplicaciones) resulta bastante bueno, dando acceso en todo momento al navegador y a la aplicación de teléfono.

El uso general de las aplicaciones resulta muy bueno aunque poco depurado visualmente si es comparado con otras interfaces. Se puede ver una muestra de esto en la fotografía 1.6.



Fig. 1.6 interfaz del Nexus One

### 1.6.2. Interfaz Sense

La interfaz Sense es una interfaz creada por HTC, resulta muy depurada tanto visualmente como a nivel de personalización de las aplicaciones, tiene una gran integración con las redes sociales como son Facebook y Twiter pues cuenta con widgets muy útiles a la par que vistosos en los que se pueden ver todas las actualizaciones de los contactos. La gestión del calendario queda mejorada con el uso de un widget que muestra el mes entero si hay citas en un dia en concreto y además permite introducir citas directamente sin tener que abrir la aplicación de calendario.

La gestión de los SMS y correos electrónicos ha sido modificada visualmente dándole un aspecto o bien de sobres o bien de lista. La cámara dispone de enfoque tocando la pantalla. La aplicación de contactos sincroniza las fotos de los contactos del Facebook con la cuenta de Google automáticamente, también incluye los cumpleaños de estos en la información del contacto. La aplicación del teléfono facilita la búsqueda de contactos haciendo búsquedas predictivas cuando se marca en el teclado.

Como contrapunto de esta interfaz es el mayor consumo de memoria en el terminal, a consecuencia de esto HTC se ve obligada a dotar de mayor memoria RAM a sus dispositivos, aunque este incremento de coste de fabricación se ve compensado con los buenos niveles de ventas de sus dispositivos, se puede ver un ejemplos de los diferentes escritorios de Sense en la figura 1.7.



Fig. 1.7 interfaz Sense

### 1.6.3. Interfaz Motoblur

Interfaz hecha por Motorola, basada en redes sociales, dispone de widgets para ver las actualizaciones en tiempo real integrándolas en un mismo lugar. Las redes sociales a las que tiene acceso son MySpace, Gmail, Twiter, Facebook, Yahoo y Last.fm. Incorpora la socorrida combinación de poner los accesos directos del teléfono y el navegador junto al lanzador de aplicaciones, se muestra en la figura 1.8.



Fig. 1.8 Interfaz Motoblur

### 1.6.4. Interfaz Touchwiz

Interfaz usada por Samsung en su versión 3.0 mejora sensiblemente, poniéndose a la altura del Sense de HTC o incluso superándolo. Presenta un aspecto visual muy cuidado, integración con las principales redes sociales, aplicaciones mejoradas visualmente, widgets propios, posibilidad de cambiar los accesos directos en el dock. Control de música mediante la barra de notificaciones. A diferencia de Sense no se sabe si consume mucha cantidad de memoria RAM pues es una interfaz muy nueva, su aspecto se puede ver en la figura 1.9.



Fig. 1.9 Interfaz Touchwiz

### 1.6.5. Interfaz Rachel

Interfaz creada por Sony, aunque visualmente es atractiva y muy vistosa en la práctica resulta engorrosa y lenta para trabajar con ella, principalmente dispone de dos áreas muy diferenciadas una llamada timescape, que agrupa SMS, correo, Twiter y Facebook en un mismo sitio, presentando en forma de cascada visual con la foto de cada contacto, esto hace que al final sea complicado buscar la información que interesa en un determinado momento.

La segunda área es la denominada Mediascape en ella se agrupa toda la información multimedia del teléfono como son fotos, videos, música y además ofrece acceso a servicios en línea como Picasa o YouTube, se muestran en la figura 1.10



Fig. 1.10 Interfaz Rachel

### 1.6.6. Interfaz MIUI

La interfaz creada en las ROMs MIUI es una interfaz creada en china, está desarrollada desde cero y basada en el código de AOSP, estéticamente es muy parecida al IPhone pues no diferencia el espacio en el que se ponen los widgets con el que se listan las aplicaciones si no que se muestra todo junto, la cámara ha sido retocada y dispone de enfoque pulsando sobre la pantalla, además también tiene incorporadas aplicaciones de gran utilidad desarrolladas especialmente para esta interfaz, como son el reproductor musical, la gestión de las aplicaciones que se les permite conectarse a internet y otra aplicación que permite compartir archivos vía WIFI.

El problema más grande con esta interfaz se ha encontrado que no todas las aplicaciones son compatibles con ella, haciendo que den fallos o directamente no se puedan instalar en el terminal. Se muestra un ejemplo de esta interfaz en la figura 1.11.



Fig. 1.11 Interfaz Miui

### 1.7. **Conclusiones:**

En este capítulo se ha estudiado el sistema operativo Android, presentando sus características principales. Se ha demostrado que es un sistema operativo que está teniendo un gran crecimiento en el mercado de la telefonía móvil, porque presenta un coste muy bajo para los fabricantes, ya que es gratuito y solo tienen que invertir en adaptarlo. Por último se ha presentado las diferentes adaptaciones que hacen los fabricantes comparándolas entre sí, también se ha podido ver como la interfaz de serie de Android que es bastante básica y como algunos de los fabricantes se han centrado en mejorarla usando widgets y integrándola con las principales redes sociales, todo esto servirá para tener unas nociones de qué es Android y poder comparar las interfaces de los fabricantes con la ROM creada.

## 2. LAS ROMS EN ANDROID

Una vez visto el contexto general de Android así como su funcionamiento interno y las diferencias que hay entre los diferentes fabricantes.

Se pasará a detallar en que consiste una ROM, la estructura que tienen, las herramientas que son necesarias para generarlas y los diferentes métodos de instalación que hay.

Todo ello servirá para alcanzar los conocimientos previos para crear una versión modificada del sistema operativo Android.

### 2.1. *El concepto de ROM*

Las ROMs son imágenes del sistema operativo Android que se instalan en los dispositivos. Estas versiones suelen estar modificadas por programadores o los mismos fabricantes con el objetivo de aportar mejoras al sistema original, bien sea modificando las aplicaciones nativas de Android o introduciendo de nuevas.

Una ROM sería lo equivalente en un ordenador, a una de las muchas distribuciones de Linux que existen, pues en ellas, al igual que en las ROM, se adapta el código fuente de Linux, dando como resultado diferentes interpretaciones y usos del sistema operativo como por ejemplo:

- Suse
- Redhat
- Debian
- Wifislax

Una ROM a diferencia de una distribución de Linux, esta adaptada para un único terminal, pues incorpora los controladores especiales para el uso de sus componentes internos.

### 2.2. *Estructura general de una ROM*

La estructura general de una ROM viene a estar compuesta por las carpetas, data, META-INF, system y el archivo boot.img tal y como se puede ver en la figura 2.1:



Fig. 2.1 Arquitectura del sistema operativo Android

Dentro de boot.img está el núcleo (kernel) y ramdisk de arranque (Un ramdisk es básicamente un pequeño sistema de archivos del núcleo necesarios para iniciar el sistema).

Dentro de META-INF se encuentran las firmas de todos los ficheros que instala la ROM y el scripts que controla la instalación de los archivos, esta carpeta tiene la siguiente estructura:

```
META-INF/
|-- CERT.RSA
|-- CERT.SF
|-- com
|   |-- Android
|   |   '-- metadata
|   '-- google
|       '-- Android
|           '-- update-binary
|               '-- updater-script
`-- MANIFEST.MF
```

Dentro de la carpeta "system" están todos los ficheros a ser copiados al directorio "/system" del teléfono móvil Android, tal como se ve en la figura 2.2.



Fig. 2.2 Carpetas dentro del directorio system

Dentro de la carpeta "data" están todos los ficheros a ser copiados al directorio "/data" del teléfono móvil Android, en esta carpeta está la configuración de las diferentes aplicaciones del sistema, aunque conviene indicar que esta carpeta no es obligatoria y casi nunca se encuentra en las ROMs.

### 2.3. *Herramientas para la creación y gestión de una ROM*

En este apartado se va a hacer una introducción general de las herramientas más importantes para la creación y gestión de ROMs en Android. Como el uso de algunas de estas puede ser muy complejo pues disponen de muchas opciones en el [apartado 3 de los anexos](#) se puede encontrar información acerca del uso de estas.

#### 2.3.1. **Android Debug Bridge (ADB)**

ADB es una herramienta incluida en el SDK de Google que permite administrar emuladores o dispositivos Android. Tiene una estructura de cliente-servidor que está formado por tres componentes:

- Un cliente, que se ejecuta en el equipo de desarrollo.
- Un servidor, que se ejecuta como un proceso en segundo plano en su equipo de desarrollo
- Un demonio, que se ejecuta como un proceso en segundo plano en cada emulador o dispositivo.

Cuando se inicia un cliente ADB, el cliente comprueba primero si hay un proceso de servidor ADB que ya se está ejecutando. Si no hay ninguno, se inicia el proceso del servidor. Cuando el servidor se inicia, se une al puerto local TCP 5037 y empieza a escuchar los comandos enviados desde el clientes ADB.

Los emuladores o dispositivos son localizados mediante el escaneo de los puertos impares en el rango de 5555 a 5585.

Cuando el servidor encuentra un demonio ADB, en ella se establece una conexión a ese puerto.

Si se instalado el plugin ADT de Android en Eclipse no es necesario acceder a ADB desde la línea de comandos ya que el plugin ADT ofrece una integración transparente de ADB en el IDE de Eclipse.

Aunque puede resultar muy útil para la depuración de errores o gestión del dispositivo Android hacerlo por línea de comandos pues se tiene mayor control e incluso se puede acceder internamente al móvil por medio de una Shell.

### **2.3.2. Monkey**

Monkey es una herramienta en línea de comando que se ejecuta en el emulador o dispositivo y genera corrientes pseudo-aleatorio de eventos de usuario, como clicks, tocar o gestos, así como una serie de eventos de nivel de sistema. Se puede utilizar Monkey para probar las aplicaciones que se están desarrollando, en una forma aleatoria pero repetible. Las opciones, pero se dividen en cuatro categorías principales:

- Opciones básicas de configuración, tales como la configuración del número de repeticiones.
- limitaciones operacionales, como la restricción de la prueba para un solo paquete.
- Los tipos de eventos y las frecuencias.
- Opciones de depuración.

Esta herramienta se a usado para probar la estabilidad de las aplicaciones en especial del widget y poder ver si fallan o son estables.

### **2.3.3. Repo y Git**

Para trabajar con el código fuente de Android, se usan tanto Git como Repo. Git es una herramienta VCS (Version Control System) de código abierto diseñada para manejar grandes proyectos que están en múltiples repositorios. Los proyectos de Android se almacenan en repositorios Git.

Se ha desarrollado también una herramienta llamada REPO para facilitar el trabajo con repositorios basados en GIT.

### **2.3.4. Fastboot**

Fastboot es una herramienta en línea de comandos que nos permite comunicarnos con el teléfono cuando este esta en modo bootloader, esta herramienta nos es de gran utilidad porque nos permite hacer acciones tales como actualizar el teléfono, borrar la información, obtener su número de serie...

Para poder trabajar con ella es necesario poner el móvil en este modo para ello hay que apagar el móvil y encenderlo pulsando el trackball y el botón de encendido simultáneamente.

## 2.4. *Descargar el código fuente de Android*

Para poder crear una ROM el primer paso será el de obtener el código fuente de Android, para ello es necesario descargarse una aplicación llamada REPO con la que podremos descargar los distintos repositorios del sistema operativo Android.

Para hacer esto se tiene que hacer.

```
curl http://Android.git.kernel.org/repo > /bin/repo  
chmod a+x /bin/repo
```

Se crea una carpeta para el proyecto.

```
mkdir /ruta_de_instalacion/system
```

En este caso se decide descargar el de CyanogenMod, existen otras como puede ser la original de Google que se encuentra en el repositorio de AOSP pero por facilidad de trabajo se usa el de CyanogenMod en el *apartado 4 de los anexos* existe información más detallada acerca de las ROMs de Caonogen.

Para empezar la descarga se hace

```
cd /<ruta>/system  
repo init -u git://github.com/CyanogenMod/Android.git -b ginger  
repo sync
```

El parámetro -b indica la versión de un proyecto en concreto, cupcake, froyo, donut,...

Una vez descargado el código fuente tenemos en la carpeta system los siguientes directorios:

```
bionic  
bootable  
build  
dalvik  
development
```

```
device  
external  
frameworks  
hardware  
kernel-msm  
Makefile  
ndk  
packages  
prebuilt  
sdk  
system  
vendor
```

Con esto ya se tiene lo necesario para compilar el source de Android para el Nexus ONE pero nos hacen falta algunas librerías.

para extraerlas tenemos que conectar el teléfono por USB al ordenador ya que vamos a tener que sacar las librerías de este una vez conectado haremos lo siguiente.

```
cd vendor/htc/passion/  
.extract-file.sh
```

Este script nos descargara del teléfono las librerías necesarias mediante adb. En el caso que se usase otro terminal distinto al Nexus One se tendría que cambiar la ruta por la del dispositivo adecuado dentro de la carpeta vendor..

Se vuelve a la raíz del proyecto y se hace lo siguiente.

```
cp build/buildspec.mk.default buildspec.mk
```

Una ves hecho este proceso ya se puede empezar a compilar el código fuente para generar la ROM.

```
make otapackage
```

o si el procesador es de doble núcleo.

```
make -j2 otapackage
```

Cuando termine el proceso de compilación en la ruta out/target/product/passion/ estarán las imágenes generadas, system.img,

recovery.img, boot.img además de los binarios, aplicaciones y todo el contenido de las imágenes.

## 2.5. *Instalar una ROM*

Para instalar una ROM en un terminal principalmente existen 3 métodos, vía OTA, con la herramienta Fastboot y por medio el modo recovery.

El modo de instalación vía OTA (Over The Air) es controlado por los fabricantes y las operadoras, es distribuido de forma inalámbrica a los usuarios y salta automáticamente cuando existe una actualización disponible.

El modo de instalación con la herramienta fastboot es muy poco usado aunque algunas actualizaciones oficiales se pueden instalar por esta vía de actualización.

La instalación usando el modo recovery es la que se utiliza normalmente por los usuarios, exceptuando las OTAs automáticas, ya que pueden actualizar sus dispositivos incluso antes que el fabricante presente sus versiones oficiales. Para poder instalar por medio de recovery la ROM tiene que contener un archivo updater-script o update-script de otra forma es imposible la instalación de la ROM con este método, pues este es el que contiene las órdenes de instalación.

Para poder instalar ROM modificadas es muy recomendable tener el terminal con privilegios de administrador, este proceso se explica en el [apartado 5 de los anexos](#).

### 2.5.1. Instalar una ROM con la herramienta fastboot

Para poder instalar una ROM con la herramienta fastboot primero se debe de poner el teléfono en modo bootloader.

Para ello se debe de conectar el terminal al PC con el cable de datos.

Acceder a línea de comandos

Activar depuración USB desde “Ajustes > Aplicaciones > Desarrollo”

Apagar el teléfono

Mantener apretado el TrackBall y botón de encendido, esto pondrá el teléfono en modo bootloader.

Confirmar que hay conexión fastboot escribiendo en la línea de comandos:

```
fastboot devices
```

Borrar datos de usuario tecleando en línea de comandos:

```
fastboot erase userdata
```

Borrar cache con:

```
fastboot erase cache
```

A partir de este punto se puede instalar un update.zip para instalar la ROM entera o únicamente instalar alguna de sus partes es decir el boot.img, el system.img, el recovery.img o radio.img.

Para instalar la ROM por medio de un update.zip se tiene que situar dentro de la carpeta que contiene el archivo a instalar:

```
cd /ruta de la carpeta con el archivo .zip/
```

finalmente se inserta:

```
fastboot update <nombre del archivo.zip>
```

Si lo que interesa es instalar únicamente una parte de la ROM entonces, se tiene que descomprimir el archivo update.zip para obtener los archivos .img.

Para ir a la carpeta con los archivos .img introduciremos en línea de comandos:

```
cd /ruta de la carpeta con los archivos .img/
```

Si se quiere instalar el boot.img se pulsara:

```
fastboot flash boot boot.img
```

Si se desea instalar el system.img se introducirá:

```
fastboot flash system system.img
```

Si se quiere instalar el recovery.img se tiene que introducir:

```
fastboot flash recovery recovery.img
```

Hay que tener en cuenta que con un recovery distinto al original (de Google) ya no se podrá actualizar el terminal vía OTA

Para instalar una radio.img se debe teclear:

```
fastboot flash radio <nombre del archivo radio>.img
```

Para salir de este modo y reiniciar el terminal se debe de pulsar:

```
fastboot reboot
```

### 2.5.2. instalación de una ROM en modo recovery

El método de instalación de ROMs en modo recovery es el más usado, para instalar ROMs diferentes de la versión original. Para poder usar este método es muy recomendable tener cambiada la versión del recovery del terminal, para este paso hay un ejemplo en el [apartado 2 de los anexos](#).

Las versiones más modernas de recovery han sido modificadas para no requerir la comprobación de firmas, pues en principio toda ROM tiene que estar firmada, esto que en un principio da mayor seguridad dificulta mucho la tarea de hacer pruebas.

El primer paso para instalar las ROMs por medio del recovery es apagar el terminal y volverlo a encender en modo fastboot, tal y como se ha explicado anteriormente en la memoria.

Una vez dentro del modo fastboot se debe de seleccionar BOOTLOADER con las teclas de subir y bajar volumen y presionar la tecla de encender el teléfono para confirmar tal y como muestra la figura 2.3.

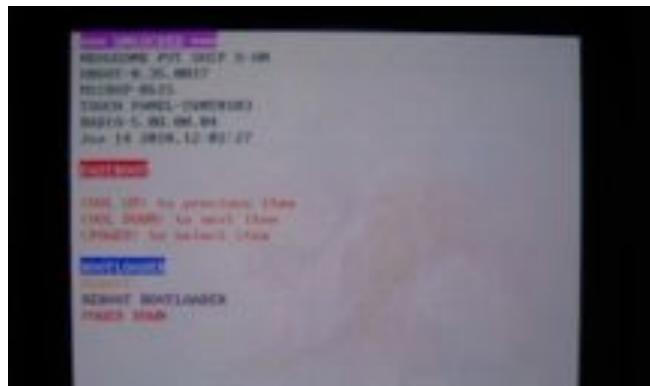


Fig. 2.3 Modo fastboot

Seguidamente se debe de seleccionar RECOVERY y presionar de nuevo el botón de encendido para entrar en el modo recovery como se puede ver en la figura 2.4.

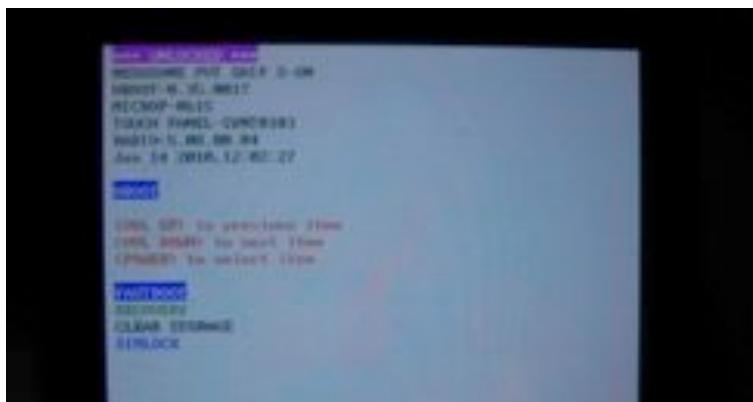


Fig. 2.4 Seleccionar recovery

Una vez dentro del modo recovery, aparecerá un menú muy similar al de la figura 2.5. para desplazarse por el menú se puede hacer bien con los botones del volumen o con el trackball, además también sirve para confirmar pulsándolo, para regresar al menú anterior se utilizará el botón de encendido.

Suponiendo que el archivo que contiene la ROM que deseamos instalar aún no está en la tarjeta SD del terminal seleccionaremos la opción mounts and storage.

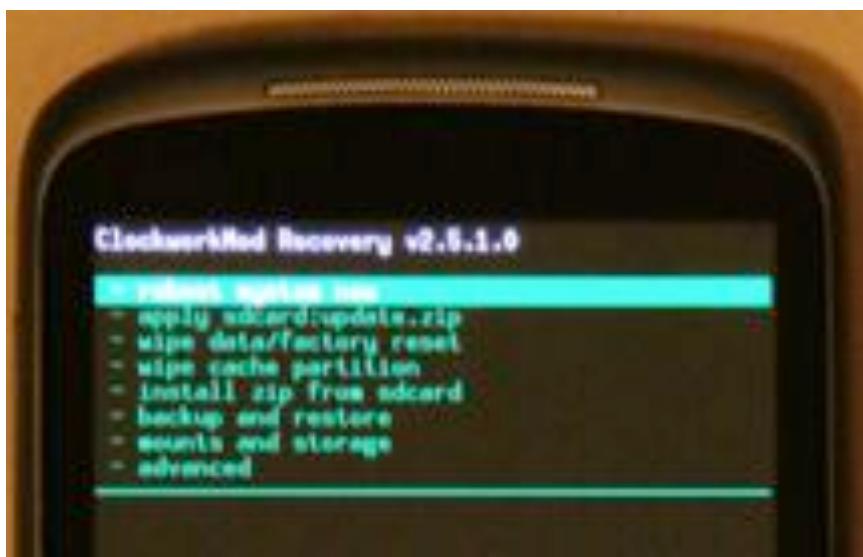


Fig. 2.5 Modo recovery

Dentro de este menú se tiene que seleccionar la opción mount USB storage tal y como muestra la figura 2.6.



Fig. 2.6 Menú mount and storage

Esto permite conectar el terminal al ordenador por medio del cable USB, y este se vera en el ordenador como si de una memoria USB se tratara. De manera que se podrá pasar a la tarjeta SD del terminal el archivo de la ROM que se desea instalar. Una vez transferido el archivo se da a UNMOUNT esto desmontara la unidad y regresara al menú anterior como se puede ver en la figura 2.7.

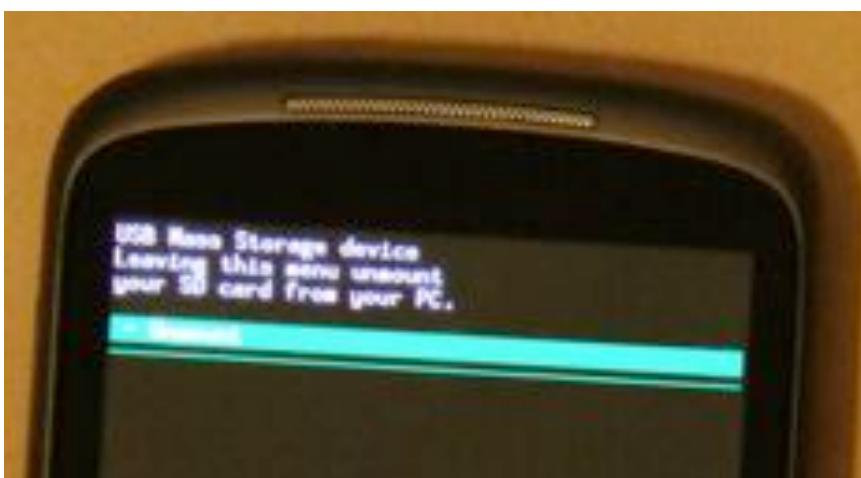
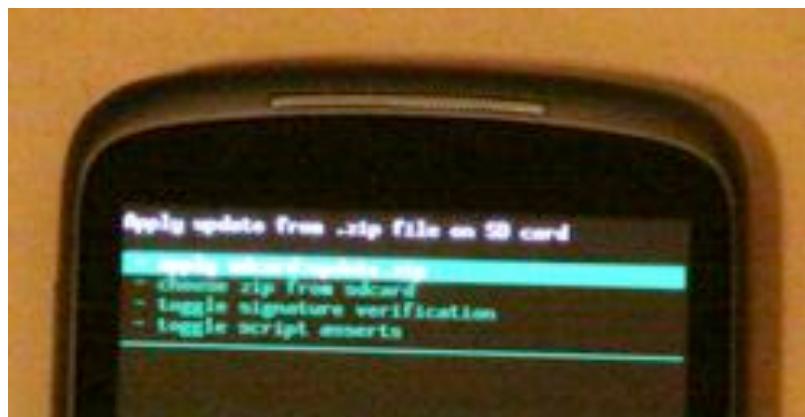


Fig. 2.7 Menú unmount

En este momento se debe regresar al menú principal del modo recovery figura 2.5 y seleccionar *wipe data/factory reset*, esto aunque no es obligatorio, borrara toda la configuración de Android así como sus aplicaciones y permitirá instalar la ROM desde cero previniendo posibles problemas de configuración.

Seguidamente se debe de seleccionar *install zip from sdcard* desde el menú principal figura 2.5 para proceder a la instalación de la ROM.

Una vez dentro aparecerá el menú de la figura 2.8 en el se debe de elegir la opción *choose zip from sdcard* para seleccionar la ROM. Es en este menú también donde permite seleccionar si se desea que tenga en cuenta las firmas en la ROM desde la opción de *toggle signature verification*, que por defecto no se comprobaran y con la opción de *toggle script asserts* se desactiva la comprobación de en el script de instalación de la ROM.



**Fig. 2.8 Menu install zip from sdcard**

Finalmente aparece un menú como el de la figura 2.9 con todos los archivos de la tarjeta SD en este menú se tiene que seleccionar el archivo que de la ROM a instalar.



### Fig. 2.9 archivos en la tarjeta SD

#### 2.5.3. Los archivos update-script y updater-script

Las ROMs que se instalan por medio del modo recovery (el método usado para generar la ROM modificada) necesitan incorporar el archivo update-script o bien el archivo updater-script para poder ser instaladas correctamente, ya que estos archivos se encargan de formatear, copiar las carpetas al terminal, establecer los permisos de las carpetas y los archivos incluso ejecutar programas externos.

El archivo update-script esta escrito en formato AMEND, este formato esta siendo cada vez menos usado ya que tiene algunas limitaciones. Los archivos updater-script están escritos en formato EDIFY, A diferencia de los update-script, los updater-script necesitan un binario llamado update-binary para poder ejecutarse. Éste se obtiene compilando desde las fuentes.

Aunque utilizan la misma lógica al menos en los comandos más básicos y más comunes, la sintaxis difiere entre AMEND y EDIFY, para generar la ROM modificada se ha optado por escribir un archivo updater-script por tanto esta en formato EDIFY, a continuación se van a exponer partes del archivo que se consideran relevantes.

```
ui_print("INICIO DE LA INSTALACION UPC");
```

Muestra por pantalla el mensaje de “INICIO DE LA INSTALACION UPC” informando al usuario durante el proceso de instalación

```
show_progress(0.500000, 0);
```

Controla el desplazamiento de la barra de progreso durante la instalación de la ROM

```
format("yaffs2", "MTD", "system");
```

Formatea el directorio system dentro del teléfono

```
mount("yaffs2", "MTD", "system", "/system");
```

Monta el directorio system en del archivo .zip en el teléfono.

```
package_extract_dir("system", "/system");
```

Copia todo el directorio system de dentro del archivo .zip al teléfono incluyendo todos los archivos y subcarpetas.

```
unmount("/system");
```

Desmonta el directorio system del teléfono

```
set_perm_recursive(0, 0, 0755, 0555, "/system/etc/ppp");
```

Se encarga de poner los permisos a las carpetas, subcarpetas y archivos que contenga el directorio /system/etc/ppp el primer cero indica al creador y el segundo al grupo al que pertenece 0755 son los permisos que se otorgaran a las carpetas 0555 indica los permisos que se otorgaran a los archivos

```
set_perm(0, 0, 0755, "/system/etc/init.d");
```

Funciona de la misma forma que el anterior pero solo otorga permisos a un único archivo.

```
run_program("/tmp/script.sh", "argumento1", "argumento2"...);
```

Ejecuta script externo.

```
symlink("origen", "destino");
```

Crea enlaces simbólicos (accesos directos) entre las dos rutas

## 2.6. **Conclusiones**

En este capítulo se han presentado el método de obtención del código fuente de Android así como los principales modos de instalación de las ROM con esto se pretende poder tener los conocimientos a partir de los cuales se pueda crear una ROM modificada. Se tiene especial mención en el modo de instalación de la ROM por medio del modo recovery, pues es el que se usara para crear la ROM de la UPC.

### 3. LA ROM MODIFICADA

En este apartado de la memoria se explicara los diferentes cambios y modificaciones que se han realizado en la ROM, para generar una versión propia del sistema operativo Android.

Una de las principales motivaciones para desarrollar este trabajo final de carrera ha sido la de dotar al teléfono móvil HTC Nexus One de radio FM, pues es un teléfono que comparte gran parte de los componentes con el HTC Desire pero al que a diferencia de este último HTC decido no dotar al móvil de radio FM para tener más ventas con el terminal que incorpora la interfaz Sense creada por ellos.

Otro de los motivos para desarrollar este trabajo final de carrera ha sido el de intentar mejorar el consumo de energía del terminal pues la batería se consume muy rápido. Esto se a hecho mejorando el uso que hace de la CPU bajando la frecuencia cuando no es necesaria y incluso llegando a subirla por encima del limite de 1GHz cuando es necesario el uso intensivo.

Se ha llevado acabo modificaciones gráficas de muchas de las aplicaciones que incorpora el móvil dándoles un aspecto azulado y con detalles de la UPC, además de cambiar los tonos y alarmas, el fondo de pantalla y la animación de inicio del móvil.

Las aplicaciones que se han considerado de poca utilidad o innecesarias han sido borradas o sustituidas por otras diferentes de más utilidad, además de incorporar aplicaciones no incluidas de serie pero que pueden tener una utilidad para los usuarios de la UPC.

Finalmente se ha hecho que las aplicaciones tengan la configuración que se desea en el momento de la instalación, consiguiendo de esta forma una especie de imagen de disco del teléfono.

#### 3.1. *Radio FM en el móvil HTC Nexus One*

El teléfono móvil Nexus One fue fabricado por HTC pero bajo las especificaciones de Google, porque Google quería un móvil para desarrolladores y que no estuviera alterado, es decir que fuera un Android original. HTC aceptó fabricar el móvil pero paralelamente diseño otro terminal con las mismas especificaciones del de Google pero con:

- Trackpad óptico en lugar de trackball (bolita)
- Radio FM
- Más memoria RAM para poder gestionar con fluidez la interfaz Sense

Esto hizo que las ventas del terminal de Google no fueran todo lo buenas que se esperaban ya que la gente prefería el HTC Desire antes que el Nexus One, pero como habían compartido diseño el chip broadcom BCM4329 que es el

encargado de controlar el Wifi, el Bluetooth y la radio FM se encuentra en los dos teléfonos móviles.

Para proporcionar radio FM al móvil Nexus One se tuvo que liberar los controladores del chip broadcom BCM4329 extrayéndolos del HTC Desire. Este proceso fue logrado por un usuario apodado intersectRaven del foro de XDA, una vez extraídos los controladores fueron incorporados al kernel de Android.

Con estas modificaciones en el Kernel es posible usar la aplicación de la Radio FM, para crear la ROM personalizada he hecho servir una portada de Miui, modificándola gráficamente.

Para que la aplicación de radio FM funcione se ha tenido que compilar la versión del kernel de CyanogenMod que ya incorpora los controladores del chip broadcom, compilar el código fuente de la aplicación radioFM.apk y incorporar las librerías necesarias para el uso de la aplicación en el terminal. Con todo esto se consigue dotar de radio FM al Nexus One.

### 3.2. ***El consumo de energía del terminal***

El consumo de la batería del móvil es bastante alto, con un uso normal (usándolo para llamar y recibir llamadas y navegar unos 10 minutos al día), la duración de la batería apenas sobrepasa el día de duración.

Así pues para intentar solucionar esta carencia del móvil y lograr que la batería tenga mayor duración se ha modificado la frecuencia de la CPU porque a mayor frecuencia de la CPU más consumo de energía y por lo tanto es menor la duración de la batería.

Los archivos que controlan la frecuencia de la CPU se encuentran en la ruta del terminal:

```
/sys/devices/system/cpu/cpu0/cpufreq/
```

En estos archivos se puede seleccionar la frecuencia máxima, frecuencia mínima, el rango de frecuencias aceptado, y los modos de trabajo de la CPU

```
interactive  
conservative  
ondemand  
powersave  
userspace  
performance
```

El kernel que usa la ROM soporta hasta una frecuencia máxima de 1113600 Hz y una frecuencia mínima de 245000 Hz. Para gestionar la frecuencia de la CPU

en lugar de modificar directamente estos archivos he usado un programa llamado CPU Tuner que adapta la frecuencia e la CPU en función de la batería además de gestionar los servicios activos para un mayor ahorro de energía.

### 3.2.1. Diferencia de tiempo en la duración de la batería

Los resultados conseguidos haciendo la gestión de frecuencia de CPU para aumentar la energía han conseguido unos resultado mejor de lo esperado en un primer momento.

A continuación se adjuntan unas gráficas que muestran el nivel de batería respecto del tiempo:

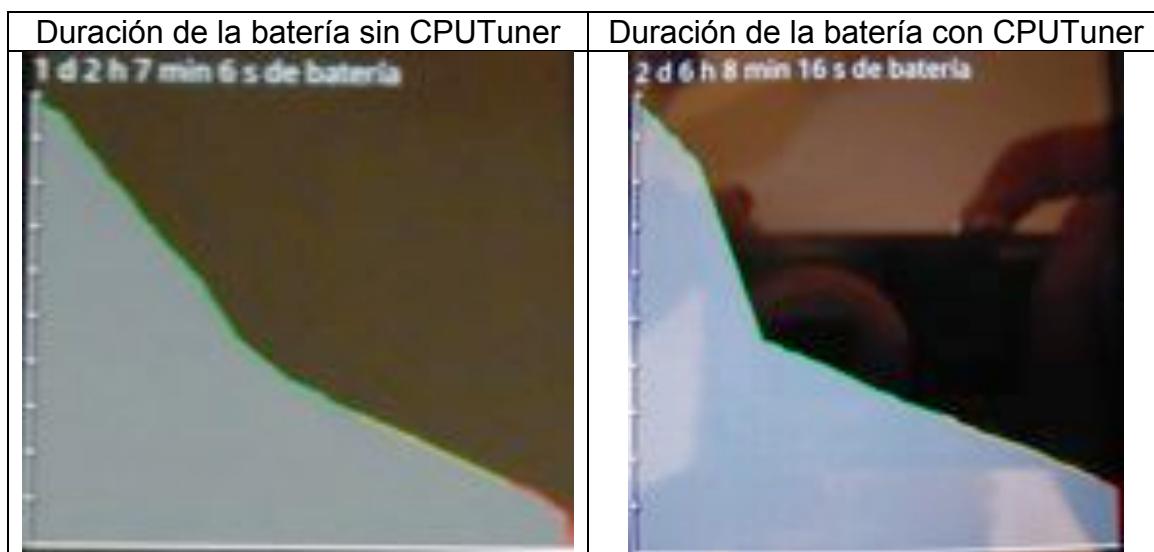


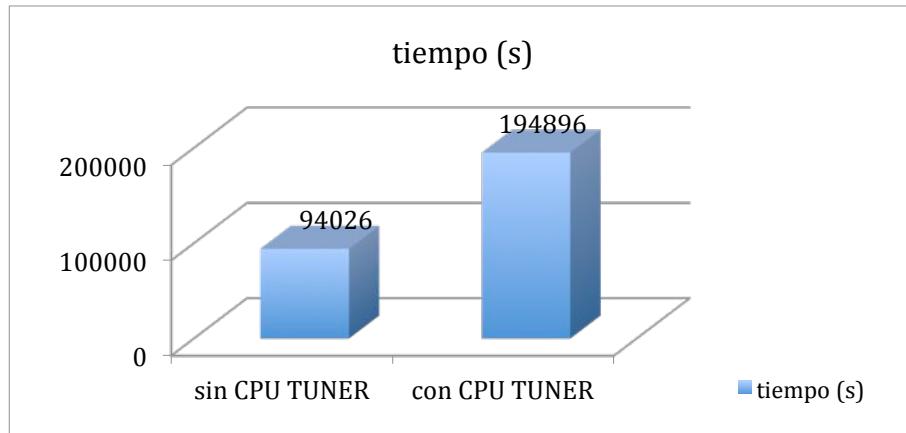
Fig. 3.1 Gráficos de duración de batería

Como se puede ver que la gestión de la frecuencia de la CPU alarga mucho el tiempo que el terminal puede estar activo, concretamente se puede ver una comparación entre el tiempo en la que se usa dicha gestión respecto del que no se hace.

La duración de la batería gestionando la frecuencia de la CPU es de un 207,27% respecto de no usarla, tal como queda reflejado en la figura 3.2.

Uso sin cputuner = 1 dia 2 horas 7 min 6 seg = 94026 segundos

Uso con cputuner = 2 dias 6 horas 8 min 16 seg = 194896 segundos



**Fig. 3.2 Diferencia de tiempos en la duración de la batería**

Se experimenta tanta mejoría porque en los tiempos en que la pantalla esta apagada el móvil funciona a una frecuencia de 245Mhz, y cuando la batería se encuentra en niveles bajos la CPU también baja hasta esos 245Mhz y se desactivan todos los servicios excepto la red móvil, con lo que se consigue reducir el consumo de energía de manera drástica. Como contrapartida el móvil cuando esta bajo de batería se queda sin internet.

### 3.3. **Modificaciones generales de la ROM**

En este apartado se va a explicar de forma detallada como se han hecho las modificaciones gráficas de las diferentes aplicaciones y cambios multimedia en el sistema

#### 3.3.1. Cambiar los tonos por defecto en la ROM

El sistema operativo Android tiene diferentes tipos de tono multimedia, ya que pueden ser muy distintos un tono de llamada a un tono de notificación de un correo por ejemplo. Los diferentes tipos de tono estas divididos en cuatro carpetas, están son:

- Ringtones, son los tonos que suenan al recibir una llamada
- Notifications, son los sonidos de suenan cuando llega un SMS, correo, actualizaciones de aplicación en el Market...
- Alarms, son los sonidos de alarma que tiene el despertador del sistema
- Ui, son los sonidos del sistema como por ejemplo desbloquear la pantalla

Todas estas carpetas contienen archivos del tipo .ogg, estas carpetas se encuentran en la ruta:

/system/media/audio/

### 3.3.2. Cambiar la animación de inicio en la ROM

La animación de inicio de Android es controlado por un archivo llamado bootanimation.zip, este archivo Zip se encuentra en la ruta

```
/system/media/bootanimation.zip
```

#### 3.3.2.1. Componentes del bootanimation.zip

Dentro del archivo bootanimation.zip se encuentran un archivo llamado Desc.txt y una o más carpetas con el nombre de part seguido de un número como puede ser “part0” o “part1” dentro de esta carpeta nos encontramos con una serie de imágenes estas imágenes son .png y su nombre es un número para ser ordenadas cronológicamente, no es necesario que el número sea el 1 solo que estén ordenadas numéricamente. El tamaño de las imágenes se adapta automáticamente a la pantalla pero conviene poner imágenes de 320\*480 para un tamaño con una relación igual para evitar posibles distorsiones.

El archivo Desc.txt es un archivo de texto, contiene la información del nombre de la carpeta que contiene la serie de imágenes, el ancho de las imágenes, el alto de las imágenes, los FPS y también el número de veces a reproducir la secuencia o si se hace en forma de bucle. Un ejemplo de archivo es:

```
320 480 30
p 1 10 part0
```

Conviene remarcar que hay que incluir el enter del final del archivo.

320 = Ancho de la animación

480 = Altura de la animación

30 = imágenes por segundo

p = puntero (esto no se debe de cambiar)

1 = # de veces para reproducir, 0 = bucle

10 = Congela el último fotograma, en FPS

part0 = La carpeta que contiene imágenes de animación

### 3.3.3. Cambiar el fondo de pantalla por defecto en la ROM

El fondo de pantalla ha sido cambiado directamente en la aplicación framework-res poniendo un logo de la UPC, en un archivo llamado default\_wallpaper.jpg. Este es un archivo del tipo .jpg con una resolución de 960 x 960 pixels para que tenga un tamaño adecuado y se puedan ver las letras de UPC en el escritorio por defecto del móvil.

Este cambio se puede realizar también sin modificar la aplicación pues el archivo wallpaper situado en:

```
/data/data/com.android.settings/files/wallpaper
```

Este archivo es el mismo default\_wallpaper.jpg de la aplicación framework-res pero renombrado a wallpaper y sin extensión. El archivo wallpaper se genera cada vez que cambia el fondo de pantalla y es la forma que tiene el sistema de guardar dicho fondo.

### 3.3.4. Cambiar al apariencia de las aplicaciones

Para cambiar el aspecto de las aplicaciones instaladas se han usado un conjunto de iconos ,de aspecto azulado en lugar de verde o anaranjado, estos han sido sustituidos por los iconos originales de las aplicaciones, con todo ello se ha intentado escoger iconos que quedaran bien con la estética de la UPC y a la vez no desentonaran con la estética de Android.

Las principales modificaciones visuales han sido:



Fig. 3.3 Lockscreen

La pantalla de bloqueo a sido modificada, ver figura 3.3, ahora se desliza hacia abajo y muestra la hora y la fecha directamente en la barra de desplazamiento, en lugar de desplazarse horizontalmente. Esto se a conseguido modificando el archivo de preferencias com.cyanogenmod.cmparts\_preferences.xml.

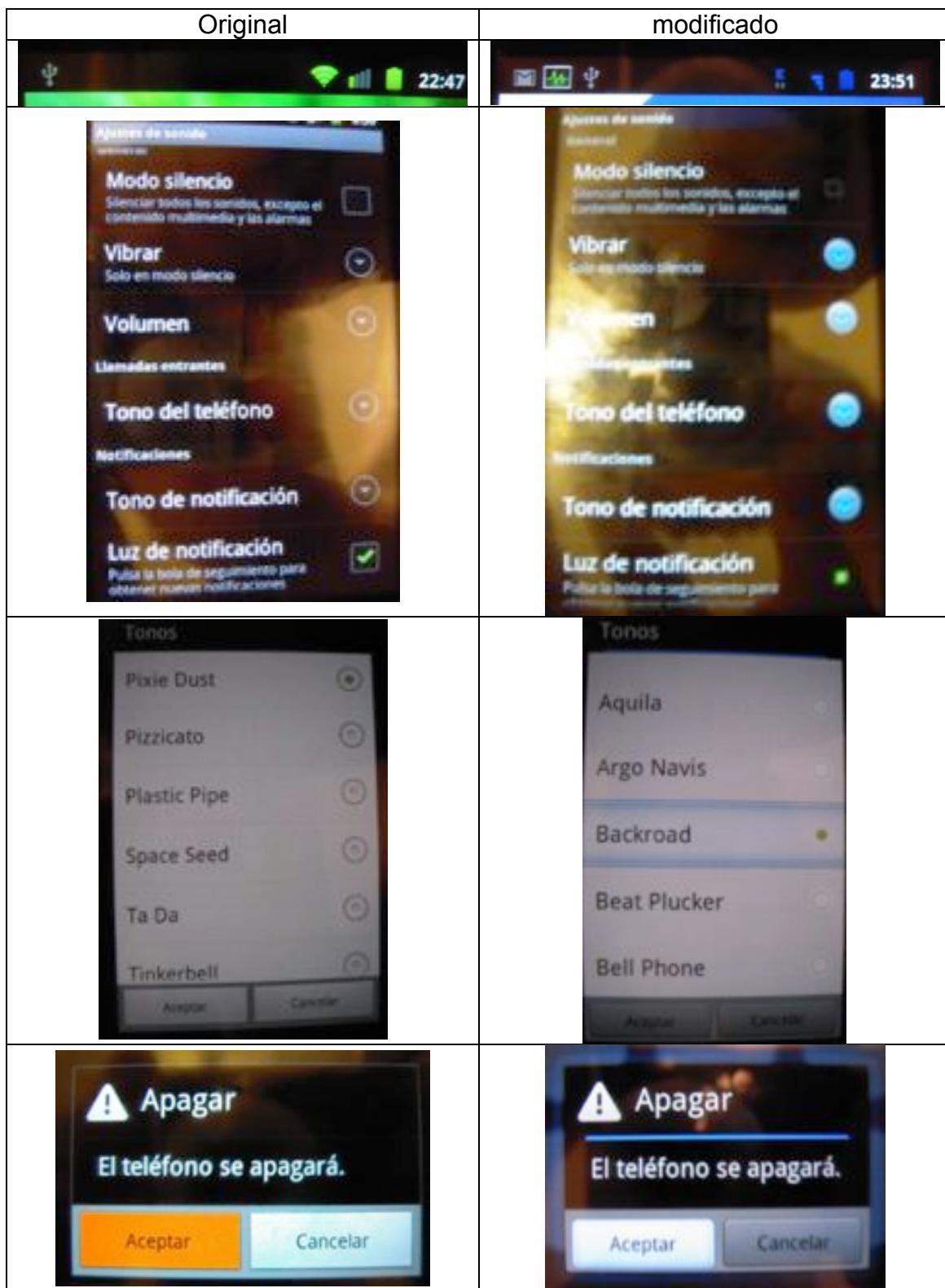


Fig. 3.4 **Modificaciones en los menús**

Los menús del sistema se han modificado cambiando los iconos de las aplicaciones framework-res.apk que controla los botones de los menús del

sistema así como las notificaciones emergentes. También se ha modificado SystemUi.apk que controla la barra de notificaciones, tal como puede observarse en la figura 3.4



Fig. 3.5 Modificaciones en Contacts y Phone

Se han modificado las aplicaciones Contacts.apk y Phone.apk para que cuando se crea o llama un contacto que no dispone de la foto del contacto en lugar del logo de Android por defecto aparezca el logo de la UPC.

También se ha modificado el aspecto de las aplicaciones Phone.apk y Contacts.apk para que los botones al ser presionados sean azulados, además el aspecto de los menús ahora es azul en lugar de verde, otra modificación gráfica que se ha hecho es que en cuanto se marca un número de teléfono tanto las teclas como la pantalla donde aparece el número se vuelva azul. Esto se puede ver en la figura 3.5

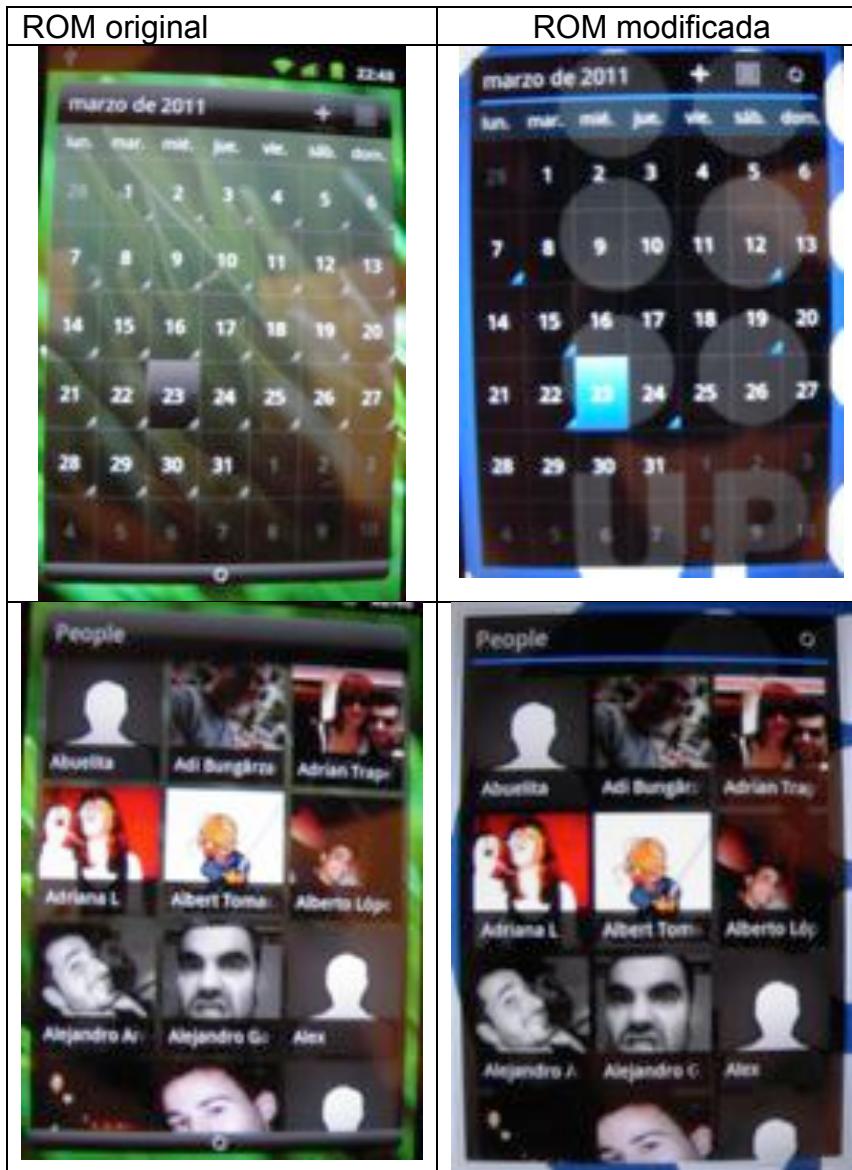


Fig. 3.6 Widgets de Launcher Pro Plus

La apariencia de los widgets del Launcher Pro han sido cambiados para que tengan la misma estética que el resto de menús, ver figura 3.6, como los widgets solo son para la versión de pago se a retocado el archivo de shared\_preferences para que permita ponerlos, además también se han modificado el numero de escritorios por defecto indicándole que son cinco en lugar de tres con esto se consigue que el logo de la UPC quede centrado.

### 3.4. Cargar la configuración

Para restaurar la configuración de las aplicaciones y de los archivos de configuración del sistema se ha creado un script de post-instalacion. Este script esta escrito en sh, debe de ejecutarse justo después de instalar la ROM la primera vez que se enciende el móvil. Carga la configuración por medio de comandos de la herramienta adb finalmente reinicia el terminal.

### 3.4.1. Los archivos de configuración de las aplicaciones

Los ficheros de configuración de todas las aplicaciones de Android están guardados en la carpeta:

```
/data/data/nombre_del_paquete
```

Esta información suele estar guardada en ficheros del tipo .xml, suelen encontrarse en la carpeta shared\_prefs, o bien en bases de datos sql. Aunque la carpeta de la aplicación también puede contener librerías de uso de la aplicación y ficheros necesarios para su uso.

### 3.4.2. Los archivos de configuración del sistema

Adicionalmente la configuración del lenguaje, país, zona horaria esta en la carpeta

```
/data/property
```

La configuración de todas las redes wifi conocidas esta en el archivo:

```
/data/misc/wifi/wpa_supplicant.conf
```

En este archivo hay un fallo de seguridad ya que la contraseña de las redes conocidas no esta encriptada y se puede capturar para usos malintencionados tal y como se muestra en la figura 3.7.

```

ctrl_interface=etc/wpa_supplicant
update_config=1

network={
    ssid="wifich"
    psk="6d4[REDACTED]"
    key_mgmt=WPA-PSK
    priority=1
}

network{
    ssid="eduroam-web"
    key_mgmt=NONE
    priority=6
}

network{
    ssid="P.I.J._FISCINA"
    key_mgmt=NONE
}

```

Fig. 3.7 Fallo de Seguridad en el archivo `wpa_supplicant.conf`

### 3.5. Gestión de las aplicaciones instaladas en la ROM

Para gestionar que aplicaciones que incorporara la ROM existen dos carpetas importantes en las que una están las aplicaciones dentro de una ROM en Android. Estas carpetas son:

```
/data/app
/system/app
```

Para que una ROM tenga las aplicaciones que interesan y no tenga las que no se consideran importantes en el momento de la instalación solo se tendrá que poner o quitar las aplicaciones .apk de estas dos rutas.

#### 3.5.1. Aplicaciones situadas en /system/app

Las aplicaciones que están dentro de la carpeta /system/app son aplicaciones de sistema. Estas aplicaciones son las consideradas importantes y controlan funciones como la cámara, el navegador, el teléfono, la agenda y por lo tanto no pueden ser desinstaladas por un usuario normal, solo pueden ser desinstalada por un usuario con privilegios de administrador (ROOT).

En la figura 3.8 se muestran todas las aplicaciones contenidas dentro de esta carpeta.

```
localhost:system # cd app
localhost:app # ls
ADMLauncher.apk           LatinIME.apk
AccountAndSyncSettings.apk LatinIMETutorial.apk
AndroidDex.apk             LiveWallpapers.apk
ApplicationsProvider.apk   LiveWallpapersPicker.apk
Bluetooth.apk              MagicSmokeWallpapers.apk
Browser.apk                Max.apk
CMParts.apk                MarketUpdater.apk
CMPartsHelper.apk          MediaProvider.apk
CmStats.apk                MediaUploader.apk
CMUpdateNotify.apk         Mes.apk
CMWallpapers.apk           Music.apk
Calculator.apk             NetworkLocation.apk
Calendar.apk               OneTimeInitializer.apk
CalendarProvider.apk       PackageInstaller.apk
Cameras.apk                PassionQuickOffice.apk
CarroceGoogle.apk          Phone.apk
CarroceLauncher.apk        PicoTts.apk
CertInstaller.apk          Protips.apk
Contacts.apk               RoamManager.apk
ContactsProvider.apk       Settings.apk
DfManager.apk              SettingsProvider.apk
DefaultContainerService.apk SetupWizard.apk
DeskClock.apk              SoundRecorder.apk
Development.apk            SpareParts.apk
DownloadProvider.apk       Stk.apk
DraftProvider.apk          Street.apk
Email.apk                 SuperUser.apk
FM.apk                    Talk.apk
Facebook.apk              TelephonyProvider.apk
FileManager.apk            Torch.apk
Galleria3D.apk             TtsService.apk
GenreWidget.apk            Twitter.apk
Gmail.apk                 UserDictionaryProvider.apk
GoogleBackupTransport.apk Vending.apk
GoogleCalendarSyncAdapter.apk VisualizationWallpapers.apk
GoogleContactsSyncAdapter.apk VoiceDialer.apk
GoogleFeedback.apk          VoiceSearch.apk
GooglePartnerSetup.apk     VpnServices.apk
GoogleQuickSearchDox.apk   YouTube.apk
GoogleServicesFramework.apk googleService.apk
HTMLViewer.apk              Kickback.apk
HttpCopyright.apk          soundback.apk
HttpWaitPolicy.apk          talkback.apk
HttpSettings.apk           Localhost.apk
```

Fig. 3.8 Aplicaciones del /system/app

### 3.5.2. Aplicaciones situadas en /data/app

Las aplicaciones que estén en la carpeta /data/app si que podrán ser borradas sin ninguna clase de problema por el usuario pues no se consideran aplicaciones del sistema. En esta carpeta tendrán que ir las aplicaciones de menor importancia para el sistema como pueden ser juegos, aplicaciones de salvapantallas, lectores de pdf etc.

No todas las aplicaciones pueden ser instaladas en /data/app pues las aplicaciones que dependan de librerías específicas pueden presentar problemas si se instalan aquí.

### 3.5.3. Aplicaciones incluidas en la ROM

Una de las principales diferencias incluidas en la ROM ha sido la de cambiar el Launcher por defecto del sistema por otro Launcher Pro Plus, este launcher incorpora unos widgets similares a los de la interfaz Sense también permite configurar varios docks con los iconos que se quieran en cada uno de ellos, además también permite configurar gestos para el dock. En la figura 3.3 se muestran el dock original y los diferentes docks configurados en el Launcher Pro Plus.

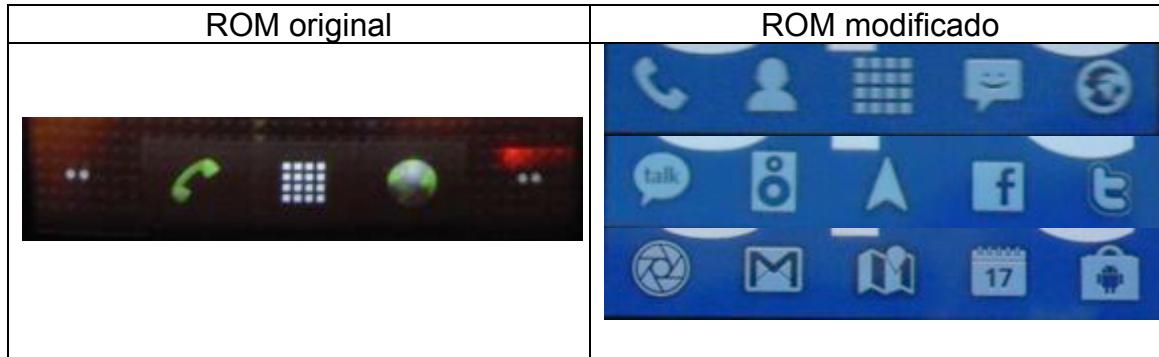


Fig. 3.9 Diferencias entre docks

Como se ha comentado anteriormente se ha incluido la aplicación de radio FM, esta aplicación está explicada en el punto 3.1 de la memoria, también se ha sustituido la aplicación de la cámara por otra que incorpora la posibilidad de enfocar la imagen en un punto de la pantalla solo tocando dicho punto. La comparación entre las dos aplicaciones de la cámara se pueden observar en la figura 3.10.

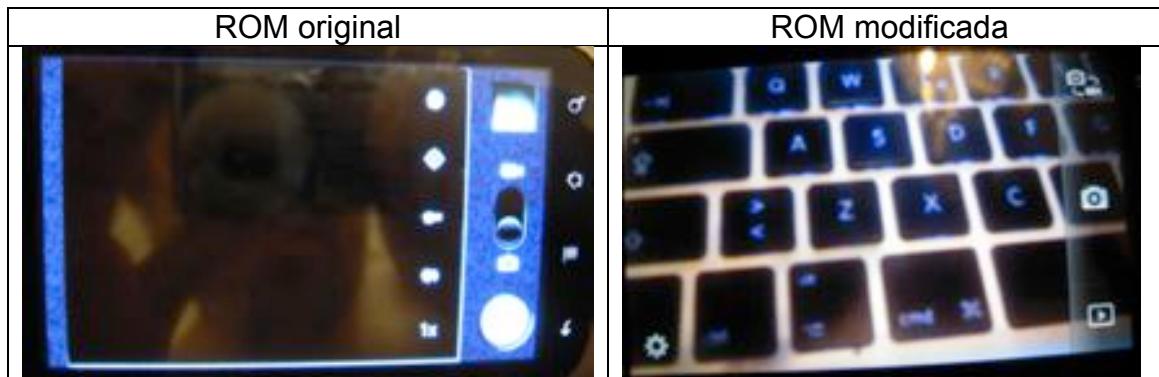


Fig. 3.10 Aplicaciones de cámaras

La aplicación de la calculadora ha sido sustituida por otra más completa como es Realcalc, pues la que incorpora Android tiene muy pocas operaciones disponibles y es insuficiente para un estudiante de ingeniería, por el contrario Realcalc tiene mas opciones y su apariencia es muy parecida a las típicas calculadoras científicas Casio, esto se puede apreciar en la figura 3.11.

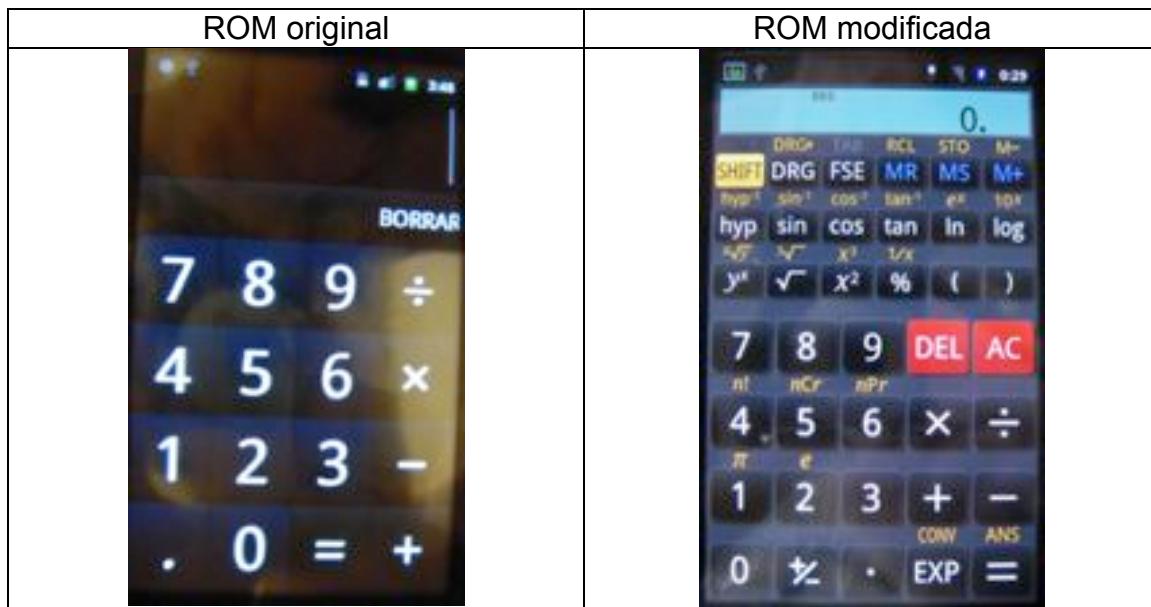


Fig. 3.11 Aplicaciones de calculadoras

La aplicación del teclado LatinIME.apk a sido cambiada por SwiftKey, ver figura 3.12, porque aunque las dos tienen teclado predictivo el algoritmo que usa esta ultima es mejor, además permite adaptarse al vocabulario que normalmente se usa en el teléfono basándose en los SMS que se han escrito esto permite que se adapte más rápido al usuario.



Fig. 3.12 Aplicaciones de teclado

La aplicación terminal emulator, permite poder insertar comandos vía terminal directamente en el teléfono

Se ha decidido incluir la aplicación adbWireless pues permite poder trabajar con la herramienta adb sin usar el cable microUSB y hacerlo por medio de WIFI, esta aplicación ha sido traducida al castellano.

Cputuner, es la aplicación que se encarga de gestionar la frecuencia de la cpu y las interficies activas para ahorrar energía.

Superuser, una aplicación encargada de gestionar los permisos de las aplicaciones que reclaman privilegios de administrador

Dropbox, para guardar datos en la nube se utiliza esta aplicación así los datos podrán ser compartidos desde el ordenador y el móvil o cualquier otra ubicación donde exista conexión a internet.

Escáner de redes, es una aplicación que se encarga de detectar las redes existentes así como su nivel de señal, como la ROM esta hecha para una carrera de telecomunicaciones se considera que puede ser útil poder observar estos datos, esta aplicación ha sido traducida al castellano.

La aplicación Torch hace la función de encender el LED del flash de la cámara para que actúe de linterna, puede ser accesible desde el escritorio pues dispone de un pequeño widget

El reproductor de música Music.apk se a cambiado por Winamp pues es un reproductor mejor, dispone de widgets, control por gestos, se puede descargar información del artista de internet, buscar videos por YouTube y se puede sincronizar con el Winamp del ordenador, tiene la posibilidad de descargar música gratuita. La comparación entre una aplicación y otra se puede ver en la figura 3.13.



Fig. 3.13 Reproductores musicales

Como las redes sociales tienen cada día mas demanda entre los usuarios, todos los fabricantes las integran en sus dispositivos, por ese motivo se ha decidido añadir el máximo de redes sociales posibles contando que los usuarios de la EETAC utilizan habitualmente muchas de estas redes sociales. Las redes sociales incluidas son:

- Twiter
- Facebook
- MySpace
- Tuenti
- Picasa
- Foursquare
- Ebuddy Messenger (da la posibilidad de conectarse al messenger)

### 3.6. **Conclusiones**

En este capítulo se a expuesto las diferentes modificaciones, cambios gráficos y nuevas características que han sido integradas en la ROM. Solo se a dejado para el próximo capítulo el diseño de los widgets creados para personalizar más la ROM diseñada para los usuarios de la EETAC

## 4. PROGRAMACIÓN EN ANDROID

Este apartado trata de la programación en Android, explicando la estructura básica de las aplicaciones, además se detallaran el funcionamiento de los widgets desarrollados para los usuarios de la EETAC.

También de explicara la forma que tiene Android de gestionar diferentes lenguajes y como tienen que estar estructurados en la aplicación.

Finalmente se explicara como portar las aplicaciones a la ROM en forma de aplicaciones .apk.

### 4.1. *Estructura de una aplicación*

En este apartado se definen y explican los 4 componentes o bloques básicos que puede contener una aplicación, si se desea saber más sobre el ciclo de vida de las aplicaciones se puede consultar el [apartado 6 de los anexos](#).

No todas las aplicaciones requieren de los 4 tipos, pero si una combinación entre algunos de ellos.

Cada vez que queramos usar uno de estos componentes se debe de especificar en un archivo llamado AndroidManifest.xml, del cual se hablará más adelante.

#### 4.1.1. Activity

Un Activity es el componente inicial de una aplicación, y por ello el más importante. Se podría definir como una simple pantalla de la aplicación. Cada actividad se define en una clase aparte y se identifica por extender de la clase Activity. Sus funciones son mostrar una interfaz con vistas (Views) y responde a eventos concretos. Para pasar de una Activity a otra, se utiliza la clase Intent, en la que se debe indicar la activity origen y la activity destino.

#### 4.1.2. Intent Receiver

Los Intent Receiver se utilizan para códigos que se quieren ejecutar cuando sucede un evento externo, por ejemplo cuando llaman al teléfono o se ha localizado la posición GPS. A diferencia de los activities, estos no muestran una interfaz, en vez de eso muestran una notificación avisando al usuario que algo ha ocurrido.

No es necesario definir los intents receivers en el AndroidManifest.xml, ya que pueden llamarse usando `Context.registerReceiver()`.

#### 4.1.3. Service

Service o servicio es un código que define una acción que se realizará en segundo plano, como puede ser localizar la posición GPS del usuario, mientras el usuario está consultando el tiempo que hará hoy. Por ello, se trata de un código no visible, suele utilizarse mientras se está en un activity. Se ejecuta usando `Context.startService()` y este terminará cuando finalice su función.

#### 4.1.4. Content Provider

Cuando queremos almacenar datos en bases de datos, como por ejemplo con SQLite, no podemos compartir esos datos con otras aplicaciones. Mediante un Content Provider se consigue compartir esa información. Se trata de una clase que implementa métodos para dejar que otras aplicaciones puedan guardar y obtener esos datos mediante el gestor contentprovider.

#### 4.1.5. AndroidManifest.xml

El AndroidManifest.xml es un fichero necesario para cualquier aplicación Android. Se encuentra en la carpeta raíz por defecto y describe valores globales de la aplicación. Incluye la descripción de las actividades y servicios, el tipo de información de cada actividad y que pueden soportar y como serán ejecutadas. También se definen los permisos, librerías y actividades que se usarán en la aplicación.

A continuación se mencionan las opciones masimportantes que se pueden definir en el manifest.

Package: situación de los ficheros que se ejecutan.

Uses-permission: permisos que se le otorgan a la aplicación que por defecto no tiene. Ej: Acceso a Internet.

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

Uses-library: librerías de Google que se van a usar en el programa. Ej: Mapas de Google.

```
<uses-library android:name="com.google.android.maps" />
```

Activity: permite a una actividad iniciarse. Todas las activities deben estar especificadas en el AndroidManifest.xml. El siguiente código debe incluirse en la actividad inicial para que sea la primera en ejecutarse.

```
<category android:name="android.intent.category.LAUNCHER" />
```

### 4.2. Programación de widgets para la EETAC

Uno de los objetivos de este trabajo final de carrera es la creación de varios widgets que permitan a los estudiantes de la EETAC acceder directamente a

las páginas más consultadas de la universidad. En época de exámenes las páginas más consultadas son:

- NETAREA para la consulta de notas de los estudiantes.
- EETAC para información general de la universidad
- ATENEA para la consulta de información y trabajos de las diferentes asignaturas

El primer widget consta de tres botones cada uno de ellos permite el acceso a uno de estos sitios de forma automática sin tener que abrir el navegador y teclear la dirección de la página manualmente.

El segundo widget consiste en un logo de la UPC, que cuando se presiona muestra un layout en el que permite seleccionar en forma de menú las siguientes páginas:

- NETAREA
- EETAC
- ATENEA
- E-SECRETARIA
- UPC
- AMICS UPC
- BIBLIOTECNIA

El aspecto de los dos widgets se puede apreciar en la figura 4.1.



Fig. 4.1 Widgets

#### 4.2.1. Funcionamiento de los Widgets

Los dos widgets constan de una clase llamada ButtonWidget que actúa como clase principal. Esta clase es del tipo AppWidgetProvider, esto indica al sistema que el tipo de programa será un widget, y lo reconocerá en el menú de Widgets, esto se puede ver en la figura 4.2



Fig. 4.2 Menú de widget

El funcionamiento de esta clase ButtonWidget básicamente consiste en, enlazar la parte gráfica definida en el layout de nombre main.xml, y hacer un intent en el que cargue un activity.

En el caso del primer widget la activity que se cargará, será la encargada de abrir el navegador pasándole la dirección de la pagina web de la siguiente manera.

```
Intent intent1 = null;
intent1 = new Intent("android.intent.action.VIEW", Uri.parse("https://atenea.upc.edu/moodle/login/index.php"));
startActivity(intent1);
```

Además de abrir el navegador, crea un método Toast que consiste en una vista que contiene un pequeño mensaje que se muestra brevemente al usuario. Cuando la vista aparece, se muestra en una vista flotante sobre la aplicación, nunca recibe el foco de acción, intentando no estorbar en exceso al usuario, simplemente advirtiéndolo de algún evento, tal y como se puede ver en la figura 4.3



cargando web NETAREA

Fig. 4.3 Ejemplo de Toast

Una vez finalizado esto se termina la activity con el método finish(); para que no aparezca de nuevo si el usuario apretara el botón atrás.

Para el caso del segundo widget la clase ButtonWidget en lugar de acceder directamente a las actividades que se encargan de abrir el navegador se enlaza el botón con una clase llamada menú. Esta clase es del tipo ListActivity y se encarga de crear una lista con los sitios a los que se puede ir, ver figura 4.4.

Una vez listados los sitios, se espera a que el usuario haga click a alguno de los elementos de la lista, cuando el usuario presiona el elemento se carga la misma activity que en el primer widget que se encarga de abrir la pagina web seleccionada.



Fig. 4.4 Lista de sitios Web

A continuación se adjuntan los diagramas UML de los dos widgets en las figuras 4.5 y 4.6.

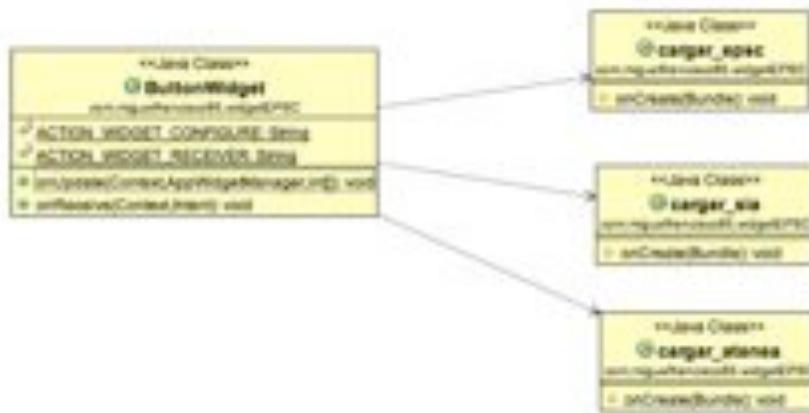


Fig. 4.5 Diagrama UML primer widget

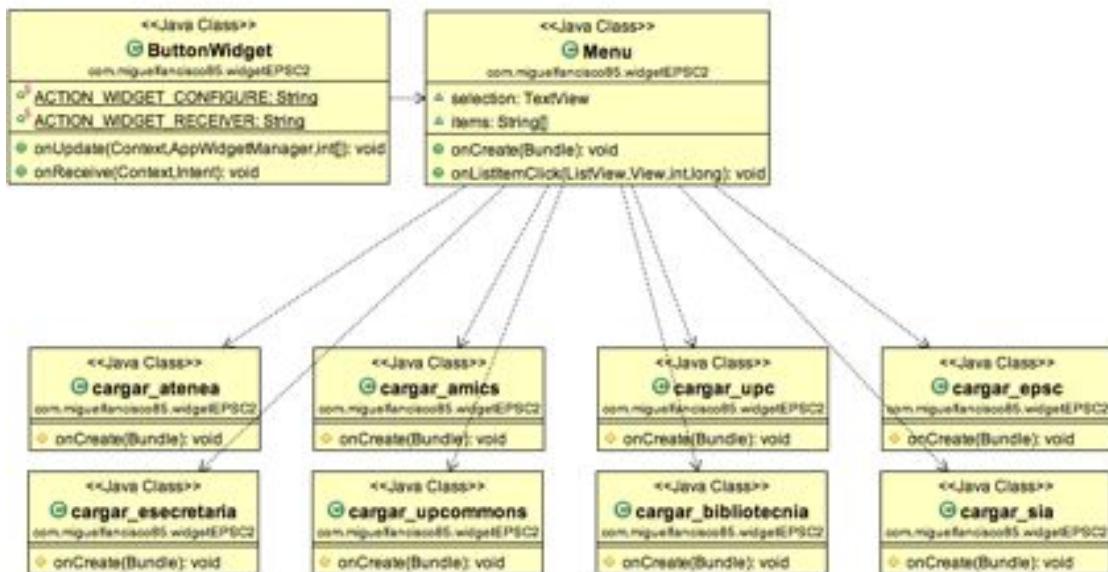


Fig. 4.6 Diagrama UML segundo widget

### 4.3. Gestión de una aplicación en diferentes idiomas

Como parte de este trabajo final de carrera se ha traducido varias aplicaciones, estas aplicaciones proceden del repositorio github y su código fuente es libre, así pues se ha modificado su código para estudiar la gestión de una aplicación en diferentes idiomas bajo la plataforma Android.

La programación en Android permite poder tener la aplicación en varios idiomas, entre todos ellos el propio sistema escogerá el correcto, si lo tiene, en caso contrario usara el idioma predefinido de la aplicación.

En Android muchos de los archivos y configuraciones están puestos en archivos del tipo .xml los archivos que contienen las frases de un programa no son ninguna excepción, para poder gestionar los diferentes idiomas se tiene que tener el archivo strings.xml en diferentes carpetas siguiendo la esta nomenclatura:

values-idioma-región

Cada una de estas carpetas tienen que estar dentro de la carpeta res tal y como se ve en la figura 4.7 de esta forma el sistema puede utilizar los diferentes idiomas o usar el predeterminado. Por ejemplo si el usuario hubiera configurado su móvil para trabajar en castellano, Android utilizaría automáticamente el archivo strings.xml de la carpeta values-es pero si por el contrario el usuario a definido como idioma catalán entonces Android usara el archivo strings.xml dentro de la carpeta values porque no tiene la carpeta correspondiente al idioma catalán.

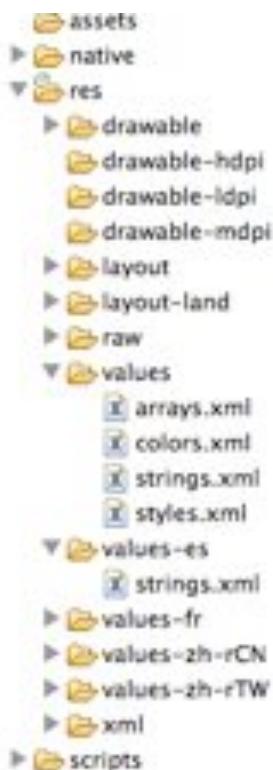


Fig. 4.7 Carpetas values con los diferentes idiomas

Un requisito para poder programar de esta manera será que en lugar de poner el texto que se desea que salga por pantalla directamente en el archivo .java

como se ve en la figura 4.8 se tiene que referenciar al archivo strings.xml tal y como se ve en la figura 4.9.

```
Toast.makeText(getApplicationContext(), "cargando web ATENEA" , Toast.LENGTH_SHORT).show();
```

Fig. 4.8 Forma incorrecta de mostrar los datos

```
Toast.makeText(getApplicationContext(), R.string.mensaje_netarea , Toast.LENGTH_SHORT).show();
```

Fig. 4.9 Forma correcta de mostrar los datos

En la figura 4.10 hay ejemplo del archivo strings.xml en el que se observa que al hacer la llamada de la cadena de texto se referencia al atributo name.

```
<?xml version="1.0" encoding="UTF-8"?>
<resources>
    <string name="hello">Hello World!</string>
    <string name="app_name">EPSC WIDGET</string>
    <string name="boton1">NETAREA</string>
    <string name="boton2">EPSC</string>
    <string name="boton3">ATENEA</string>
    <string name="boton4">BIBLIOTECNIA</string>

    <string name="mensaje_netarea">cargando web NETAREA</string>
    <string name="mensaje_epsc">cargando web EPSC</string>
    <string name="mensaje_atenea">cargando web ATENEA</string>

    <string name="widget_epsc">Widget EPSC</string>
</resources>
```

Fig. 4.10 Ejemplo de fichero strings.xml

#### 4.4. Generar un .apk

Para poder instalar las aplicaciones creadas en la ROM, es necesario crear un archivo .apk que es el que contendrá la aplicación compilada y es el archivo que se colocara en la carpeta /system/app o bien /data/app de la ROM. Para hacer este proceso se a trabajado con el entorno de programación Eclipse que es el recomendado por Google.

Cuando el programa este finalizado, hay que situarse sobre la carpeta del proyecto presionar con el botón derecho, seleccionar Android Tools y clicar en Export Signed Application Package tal y como muestra la figura 4.11.

Como medida de seguridad en Android todas las aplicaciones tienen que estar firmadas para poder ser instaladas en un teléfono, ya que así es posible saber de forma inequívoca quien a creado la aplicación. Si se desea firmar después la aplicación también sería posible hacerlo para ello se usaría una aplicación diferente. En ese caso se debería de seleccionar Export Unsigned Application Package.

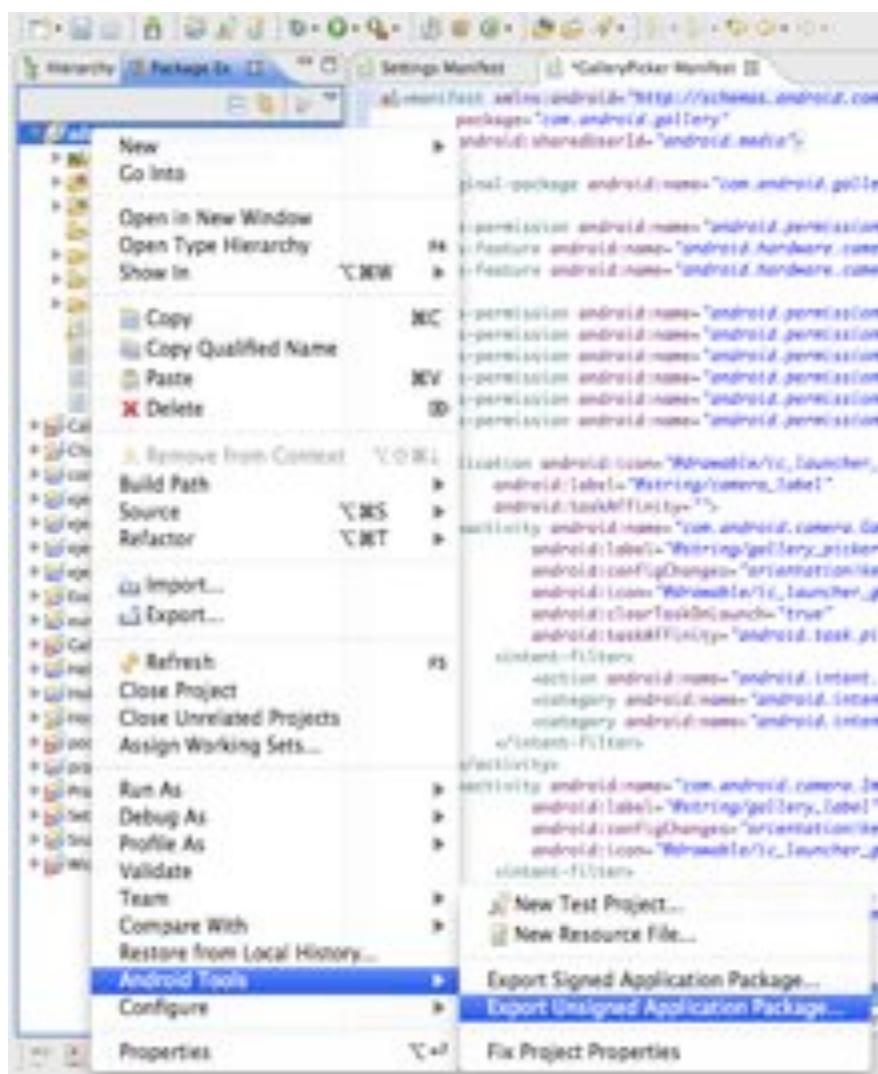


Fig. 4.11 Menú Android tools

En ese momento aparecerá una ventana en la que se tiene que seleccionar el proyecto que queremos exportar, como en la figura 4.12.



Fig. 4.12 Seleccionar proyecto

Seguidamente hay que crear un keystore si no existe aún, figura 4.13. Para ello pedirá la localización donde se guardara y una contraseña de acceso a este.



Fig. 4.13 Seleccionar keystore

Una vez hecho esto pedirá que se introduzca un alias, una contraseña, durante cuanto tiempo será valido el certificado, nuestro nombre, el nombre de nuestra organización, el nombre de nuestra unidad, la ciudad, el estado y el código del país. Con toda esta información creara un certificado valido para poder firmar las aplicaciones como en la figura 4.14.



Fig. 4.14 Datos para firmar la aplicación

Finalmente preguntara el destino donde guardar la aplicación firmada, como muestra la figura 4.15



Fig. 4.15 Destino de la aplicación .apk

Una vez hecho ya se tiene el archivo con la aplicación firmada y se puede colocar en la carpeta /system/app o /data/app de la ROM y cuando se instale esta la aplicación también será instalada.

#### **4.5. *Conclusiones***

En este capítulo se ha tratado de introducir los conceptos básicos de la programación de aplicaciones en Android, y se ha desarrollado un par de widgets para los usuarios de la EETAC que serán incorporados en la ROM.

De esta manera se demuestra que existe la posibilidad de insertar aplicaciones propias en el sistema operativo Android en el momento de la instalación tal como hacen las operadoras en los terminales, al incorporar programas propios con sus servicios.

## 5. CONCLUSIONES FINALES

En este capítulo se van a explicar las conclusiones a las que se ha llegado en este trabajo final de carrera, los conocimientos adquiridos, así como una visión de futuro del sistema operativo Android y se hace una comparación entre la ROM creada con otras existentes en el mercado.

### 5.1. *Conocimientos adquiridos*

En este trabajo final de carrera se ha tratado de estudiar el sistema operativo Android en su conjunto, desarrollando una versión modificada de este sistema.

En el transcurso de estos meses se ha estudiado la forma de obtener el código fuente del sistema operativo, el uso de la plataforma eclipse para el trabajo de programación con este sistema operativo, así como la obtención del SDK de Google, el proceso de compilación del código fuente para obtener una ROM, los diferentes métodos de instalación de las ROM en el dispositivo, la gestión de las aplicaciones que se quieren instalar en el dispositivo así como configuración de estas.

También se ha tratado de modificar las aplicaciones para mejorar su aspecto visual, además de mejorar aspectos como la gestión de energía o la inclusión de radio FM en el terminal.

Por último se han creado widgets con la temática de la UPC y que da acceso a los principales servicios de la universidad con el fin de poner las bases de un widget disponible en el Market de la UPC pues no existe ninguno y es una buena forma de promocionar la universidad además de dar más servicios a los estudiantes.

Con todo ello creo haber adquirido los conocimientos de las principales áreas de desarrollo del sistema operativo Android.

### 5.2. *Comparativa de la ROM creada con otras existentes en el mercado*

A diferencia de la versión original en la ROM creada se ha modificado el launcher por otro que dispone de widgets mejorando la integración con las principales redes sociales y acercándolo de esta manera a versiones, como la de Sense pero sin ser tan pesadas para el sistema como esta.

En la ROM creada se ha procurado mejorar la gestión de la energía dando la posibilidad a los usuarios de intervenir sobre ella, esto es algo que ninguna ROM integra y puede resultar muy útil para alargar la duración de la batería.

Al ser un estudiante de la UPC todo el aspecto gráfico ha sido modificado con tonos azulados en lugar de verdes, pues es uno de los colores de esta universidad, además de poner un fondo de pantalla con el símbolo de la UPC.

Con la creación de los widgets de la UPC se ha pretendido dar facilidad de acceso a los servicios de la UPC a los usuarios de esta.

### **5.3. Visión de futuro del sistema operativo Android**

Aunque hasta ahora hemos visto casi en exclusiva el sistema operativo Android en móviles, en no demasiado tiempo se empezara a ver el sistema instalado en otro tipo de dispositivos, como pueden ser los tablet o incluso coches o electrodomésticos. Ya que al ser un sistema operativo libre y gratuito es muy fácil para las empresas adaptar e integrar Android con sus dispositivos.

Es muy posible que el sistema operativo Android se expanda tanto como lo ha hecho Windows en los ordenadores pero en dispositivos de menor complejidad, Android al igual que en su día Windows tiene un uso muy sencillo y intuitivo para los usuarios. Los grandes fabricantes empiezan a integrar Android en la televisión, home-cinema, tablets, no sería de extrañar que con la competencia de estos y el interés por sacar productos nuevos y diferenciarse en el mercado surjan dispositivos como:

- Tablet orientado a la cocina que indique recetas y listas de la compra, o incluso neveras que gestionen las compras y los productos que faltan poco para caducar
- Ordenadores de a bordo del coche usando Android para comunicarse con el móvil hacer de GPS, gestión de tráfico, control de menús del propio coche, avisar en caso de accidente a través del móvil
- Relojes que usando Android interactúen con el móvil para gestionar todas las alertas y sucesos directamente en el reloj
- Maquinas expendedoras como las del metro o cajeros usen Android por ser gratuito fácil de modificar y hacer desarrollar aplicaciones, con lo que se ahorren costes de diseño.

### **5.4. Estudio medioambiental**

El impacto medioambiental producido por la aplicación de este trabajo final de carrera es muy bajo, se considera que únicamente a tenido un costo energético en el momento de desarrollo del mismo aunque sus efectos pueden ser perfectamente revertidos por el hecho de que uno de los puntos importantes del trabajo final de carrera se basa en alargar la duración de las baterías de los terminales móviles HTC Google Nexus One, lo que se traduce en menos ciclos de carga con el consiguiente ahorro energético en el mismo tiempo de uso del móvil.

Así pues este trabajo final de carrera mas que perjudicar el medio ambiente lo beneficia con el ahorro energético.

## BIBLIOGRAFÍA

1. Libros sobre Android Fecha de ultima consulta (10/02/2011)

Titulo: Pro Android,  
Autores: Sayed Y. Hashimi, Satya Komatineni,  
Editorial: Apress

Titulo: Hello, Android  
Autor: Susannah Davidson  
Editorial: Ed Burnette

Titulo: The Busy Coder's Guide to Android Development  
Autor: Mark L. Murphy  
Editorial: CommonsWare books

2. API oficial de Android: Fecha de ultima consulta (02/07/2010)  
<http://code.google.com/Android>

3. Android Open Source Project: Fecha de ultima consulta (9/10/2010)  
<http://source.android.com/>

4. Open HandsetOfficial Page: Fecha de ultima consulta (24/08/2010)  
<http://www.openhandsetalliance.com/>

5. Wikipedia, versions de Android: Fecha de ultima consulta (26/08/2010)  
<http://es.wikipedia.org/wiki/Android>

6. Blog de Android en español: Fecha de ultima consulta (14/02/2011)  
<http://celutron.blogspot.com/>  
<http://www.androidsis.com/>  
<http://www.elandroidelibre.com/>  
<http://soyandroide.com/>  
<http://www.xatakandroid.com/>

7. Informacion sobre ROMs: Fecha de ultima consulta (28/03/2011)  
<http://www.htcmania.com/foro.php>  
<http://forum.xda-developers.com/index.php>  
<http://android.modaco.com/>  
<http://www.cyanogenmod.com/>

8. Grupos en Google de Android: Fecha de ultima consulta (12/12/2010)

<http://groups.google.com/group/android-espanol-developers-group?lnk=srg>  
<http://barcelona.gtugs.org/>  
<http://catdroid.org/>

9. Tutorial desbloquear el terminal Fecha de ultima consulta (29/08/2010)

<http://ilikemygooglephone.com/2010/01/06/complete-guide-with-images-unlock-bootloader-and-root-google-nexus-one-mac-windows-linux/>  
<http://www.tecnologiablog.com/post/1582/como-desbloquear-el-google-nexus-one>

10. Descargar Kernels: Fecha de ultimita consulta (25/02/2011)

<http://www.mediafire.com/intersectRaven>

11. TFC de android de la UPC: Fecha de ultima consulta (05/01/2011)

<http://upcommons.upc.edu/handle/123456789/178914>  
<http://upcommons.upc.edu/handle/123456789/143404>  
<http://upcommons.upc.edu/handle/123456789/143350>  
<http://upcommons.upc.edu/handle/123456789/143808>  
<http://upcommons.upc.edu/handle/123456789/191364>

12. Tutorial Updater-script Fecha de ultima consulta (25/02/2011)

<http://forum.xda-developers.com/showthread.php?t=641223>

13. Portar android Fecha de ultima consulta (15/1/2011)

[http://www.kandroid.org/android\\_pdk/system\\_requirements.html](http://www.kandroid.org/android_pdk/system_requirements.html)

14. Ejemplos widget: Fecha de ultima consulta (08/02/2011)

<http://unpocodejava.wordpress.com/2010/09/15/android-sdk-tutorial-webkit-2/>  
<http://www.android-spa.com/viewtopic.php?t=2416>

15. Programación en Android: Fecha de ultima consulta (25/2/2011)

[http://www.programacion.com/articulo/introducion\\_a\\_los\\_layouts\\_para\\_android\\_400](http://www.programacion.com/articulo/introducion_a_los_layouts_para_android_400)

16. El crecimiento de Android Fecha de ultima consulta (17/12/2010)

<http://www.muymovil.com/2011/02/24/el-crecimiento-de-android-resumido-en-video>  
<http://www.muymovil.com/2011/02/24/el-crecimiento-de-android-resumido-en-video>

## ANEXOS

### *I. Versiones de Android*

#### a. Android 1.5 (Cupcake)

Debido a las grandes mejoras introducidas en la tercera release de Android, en abril de 2009, el número de versión saltó directamente a la 1.5. Basado en el kernel Linux 2.6.27, las novedades de mayor interés eran las siguientes:

- Rediseño completo de todos los elementos de la interfaz.
- Interfaz de Android 1.1 (izquierda) y Android 1.5 (derecha), vía Android Developers.
- Transiciones animadas entre ventanas.
- Mejoras en la velocidad de la cámara.
- Menor tiempo de búsqueda de los satélites GPS, gracias a la posibilidad de utilizar A-GPS.
- Mejoras en la velocidad del navegador web gracias a la inclusión de la última versión de Webkit, el engine de renderizado, y el nuevo interprete de java SquirelFish.
- Intérprete JavaScript.
- Añadida la posibilidad de copiar y pegar texto y buscar texto dentro de una página web.
- Posibilidad de personalizar los widgets mostrados en la pantalla de inicio.
- Inclusión de teclado en pantalla, con soporte para orientación vertical y apaisada, funcionalidades de auto corrección y soporte de diccionarios del usuario.
- Añadida la posibilidad de grabar y reproducir videos.
- Soporte de Bluetooth Stereo.

#### b. Android 1.6 (Donut)

Lanzada en Septiembre de 2009, está basada en el kernel 2.6.29 de Linux. Se considera una actualización menor, pero aun así desarrollan algunas novedades bastante interesantes:

- Quick Search Box, una caja de búsqueda en la pantalla de inicio que permite buscar entre distintas fuentes (los contactos, el historial del navegador, directamente en Google,). Con autocompletado y capacidad de aprendizaje.
- Mejorada la velocidad de la cámara.
- Posibilidad de conectarse a redes VPN, 802.1x.
- Nueva pantalla para controlar la batería, que permite comprobar qué aplicaciones y servicios son los que más consumen. Desde esta pantalla se puede también parar o desinstalar estas aplicaciones

- Las aplicaciones de Android Market aparecen ahora ordenadas por categorías (Aplicaciones, Juegos y Descargas). Para cada categoría podemos consultar las últimas actualizaciones y las aplicaciones más populares. Además para cada aplicación se muestra ahora capturas de pantalla y reviews de otros usuarios.
- Nuevo motor de texto a voz.

### c. Android 2.0 / 2.1 (Éclair)

En Android 2.0, de noviembre de 2009, se continuó con la tradición de utilizar dulces de repostería como nombres de versión. Las novedades son:

- Rediseñó la interfaz del navegador, contando ahora con soporte para distintas características de HTML5 (entre ellas la etiqueta vídeo), la posibilidad de hacer zoom con una doble pulsación y thumbnails de los marcadores
- Soporte nativo de flash para la cámara
- Zoom digital, modo scene, balance de blanco, efectos de color y modo macro.
- Mejoras en el teclado virtual.
- Soporte para nuevos tamaños y resoluciones de pantalla.
- Contactos rápidos.
- Bluetooth 2.1
- Soporte nativo de Facebook.
- Mejoras en Google Maps, que pasaba a ser multitáctil y soportar capas
- Soporte de Microsoft Exchange.
- Mejoras en el calendario.
- En diciembre de 2009 se publicó una pequeña revisión, Android 2.0.1, que mejoraba la duración de la batería y la estabilidad, la llamada a tres, el GPS, el Bluethooth, la velocidad de disparo y auto focus de la cámara.
- Android 2.1, que llegó a los móviles Android en enero de 2010, también se considera una actualización menor. Entre otras cosas incluye:
  - Reconocimiento de voz. Ahora se puede dictar en lugar de escribir en cualquier campo de texto).
  - Mejoras en el teclado virtual.
  - Galería 3D, al estilo CoverFlow.
  - Uso del gesto de “pellizcar” para hacer zoom en el navegador, la galería y en Google Maps.
  - Nuevas aplicaciones de reloj/tiempo y noticias.
  - Mejoras en Google Maps: sincronización de nuestros sitios favoritos, modo noche y auto completado de búsquedas.
  - Google Goggles.
  - Mejoras en la duración de la batería.

### d. Android 2.2 (Froyo)

Disponible desde finales de Junio del 2010, se citan los siguientes cambios respecto a sus versiones precedentes:

- Actualizaciones automáticas para aplicaciones: Las aplicaciones recibirán actualizaciones automáticas, consiguiendo tener siempre la última versión del software.
- Soporte WiFi IEEE 802.11n
- Soporte para Radio FM.
- Soporte Flash 10.1 y Adobe AIR 2.5
- Soporte de la API gráfica OpenGL 2.0
- Posibilidad de asignar un color de LED en el TrackBall para diferentes eventos del terminal.
- Creación de un compilador JIT el cual se basa en un método para mejorar el rendimiento del tiempo de ejecución de los programas, preparando el código para ser interpretado, esto mejora entre 2 y 5 veces el rendimiento respecto de Eclair.
- Tethering por USB y hotspot WiFi
- Incorporación del mismo motor de JavaScript V8 de Chrome.
- Creación de un sistema de mensajería “en la nube” dotado de un API mediante el cual se puede enviar un mensaje desde la web y hacerlo llegar a un teléfono vía Push.
- Posibilidad de enviar posiciones desde Google Maps al teléfono, usando un plugin del navegador Google Chrome.
- Posibilidad de mover una aplicación instalada desde el teléfono a la tarjeta de memoria, y viceversa.
- Opciones avanzadas de gestión energética

#### e. **Android 2.3 (Gingerbread)**

El 6 de diciembre de 2010, el SDK 2.3 fue liberado, los principales cambios que incorpora el sistema son:

- Actualización del diseño de la interfaz de usuario
- Soporte para pantallas extra grandes y resoluciones WXGA y mayores
- Soporte nativo para telefonía VoIP SIP
- Soporte para reproducción de videos WebM/VP8, y decodificación de audio AAC
- Soporte para Near Field Communication, el NFC es una tecnología de comunicación inalámbrica, de corto alcance y alta frecuencia (13,56MHz) que permite el intercambio de datos entre dispositivos a menos de 10cm, como objetivos tiene el permitir la identificación, funcionar como un método de pago de nuestras compras y permitir el intercambio de datos.
- Funcionalidades de cortar, copiar y pegar disponibles a lo largo del sistema
- Teclado multi-táctil rediseñado
- Soporte mejorado para desarrollo de código nativo

- Mejoras en la entrada de datos, audio y gráficos para desarrolladores de juegos
- Recolección de elementos concurrentes para un mayor rendimiento
- Soporte nativo para más sensores (como giroscopios y barómetros)
- Un administrador de descargas para descargar archivos grandes
- Administración de la energía mejorada y control de aplicaciones mediante la administrador de tareas
- Soporte nativo para múltiples cámaras
- Cambio de sistema de archivos de YAFFS a ext4

#### f. **Android 3.0 (Honeycomb)**

Es la ultima versión de Android fue presentada el 2 de febrero de 2011, su funcionamiento esta orientado principalmente a tablets aunque también funciona en móviles, su principales mejoras son:

- Escritorio 3D con widgets rediseñados
- Sistema multitarea mejorado
- Mejoras en el navegador web predeterminado, entre lo que destaca la navegación por pestañas, autorelleno de formularios, sincronización de favoritos con Google Chrome, y navegación privada.
- Soporte para video llamada a través de Google Talk
- permite encriptar los datos mediante una contraseña

## ***II. Dalvik vs Java Virtual Machine***

La máquina virtual Java se puede encontrar en casi cualquier ordenador de sobremesa y se basa en pila. Por otro lado, la máquina virtual Dalvik utiliza el registro base, ya que los procesadores para móviles están optimizados para la ejecución en registros, permitiendo que se aceleren los tiempos de ejecución.

En general las máquinas basadas en pila deben utilizar instrucciones para cargar los datos en la pila y manipularlos y por tanto, requieren más instrucciones que una máquina de registros para aplicar el mismo código de alto nivel, pero las instrucciones de una máquina de registro deben codificar la fuente y el destino de los registros y por consiguiente tienden a ser instrucciones de mayor tamaño.

Debemos tener en cuenta que la máquina virtual Dalvik se diseñó para ser ejecutada en una CPU de pequeñas dimensiones, en un sistema operativo con poca memoria RAM y poco espacio SWAP; todo ello, alimentado por una batería. La máquina virtual se ejecuta sobre un kernel de Linux 2.6.

### ***III. Uso de herramientas para la creación de una ROM***

#### **a. ADB**

Para poder ejecutar directamente ADB desde línea de comandos necesitaremos que en nuestro PATH este incluida la ruta de la carpeta Tools del SDK de Android en mi caso es:

```
GNU nano 2.0.6          File: .bash_profile
export PATH=$PATH:/Users/miguelfrancisco85/Documents/android-sdk-mac_86/tools
```

**Fig. 3.1 Ejemplo de PATH**

El sintaxis general de un comando es:

```
adb [-d|-e]-s <serialNumber> <command>
```

Categoría	Comando	Descripción	Comentarios
Opciones	-d	Dirigir un comando adb al único dispositivo USB conectado.	Devuelve un error si hay más de un dispositivo USB está conectado.
	-e	Dirigir un comando adb a la única instancia de emulador funcionando.	Devuelve un error si hay más de un emulador de instancia se está ejecutando.

	<code>-s&lt;serialNumber&gt;</code>	Dirigir un comando adb a un determinado emulador / dispositivo, conocido por su número de serie asignado por el ADB (como "emulador-5556").	Si no se especifica, adb genera un error.
--	-------------------------------------	---	---

Para saber la lista de dispositivos o emuladores que tenemos activos usaremos el comando

```
adb devices
```

En respuesta, adb imprime esta información de estado para cada caso:

```
[serialNumber] [state]
```

Número de serie:

Es una cadena creada por el ADB para la identificación exclusiva de un emulador / dispositivo por ejemplo su número de puerto de la consola.

Estado:

El estado de la conexión.

offline –el dispositivo o emulador no está conectado a ADB o no responde.

device – el dispositivo o emulador está relacionado con el servidor "ADB".

Instalación de una aplicación

Se puede utilizar adb para copiar una aplicación desde el equipo de desarrollo y instalarla en un dispositivo de ejemplo o emulador.

```
adb install <path_to_apk>
```

o en el caso que haya más de un emulador o dispositivo

```
adb -s <serialNumber> install <path_to_apk>
```

aquí se puede ver un ejemplo de este proceso:

```
adb -s emulator -5556 install nombre_de_la_aplicacion.apk
```

Hay que tener en cuenta que para poder instalar aplicaciones fuera del Market en un dispositivo este tiene que tener activada la opción de:

Ajustes/Aplicaciones/Orígenes desconocidos

Para copiar un archivo o directorio (recursivamente) desde el emulador o dispositivo, el uso es

```
adb tire <remote> <locales>
```

Para copiar un archivo o directorio (recursivamente) *hacia* el emulador o dispositivo, el uso es

```
adb push <locales> <remote>
```

En los comandos <locales> y <remote> se refieren a las rutas a los archivos de destino / directorio en el equipo de desarrollo (local) y en el dispositivo de ejemplo / emulador (a distancia).

```
adb push nombredelarchivo.txt /sdcard/nombredelarchivo.txt
```

ADB puede ejecutar comandos en la Shell del dispositivo o emulador esto nos proporciona poder testear las aplicaciones dentro del mismo dispositivo

Para acceder a la Shell del dispositivo o emulador

```
adb [-d| -e] -s {<serialNumber>} Shell
```

o por el contrario podemos ejecutar directamente un comando con la siguiente sintaxis:

```
adb [-d| -e] -s {<serialNumber>} Shell <shellCommand>
```

## b. Monkey

La sintaxis básica de la herramienta monkey es:

```
adb shell monkey [opciones] <eventos-cuenta>
```

Un ejemplo de uso seria:

```
adb shell monkey -p your.package.name -v 500
```

Con esto conseguiríamos lanzar 500 eventos pseudo-aleatorios sobre el paquete de nombre your.package.name, -p indica que la prueba se limita al paquete your.package.name si este llama a otros no se testearan. -v indica el nivel de detalle de la información de los resultados siendo 0 el nivel por defecto y con un nivel de detalle menor.

Opciones a nivel de tipos de eventos:

```
adb shell monkey -p your.package.name --pct-touch 30 500
```

Las opciones del tipo --pct permiten asignar el porcentaje de eventos que se desarrollan y estos deben ser de un tipo concreto. En este caso --pct-touch 30 indica que el 30% de los eventos serán del tipo táctil sobre la pantalla.

Opciones a nivel de depuración:

```
adb shell monkey -p your.package.name --ignore-crashes 500
```

Las opciones del tipo --ignore indican a monkey que ignore el tipo de error y continue con lanzando eventos pseudo-aleatorios hasta el final. En este caso --ignore-crashes que continue con las pruebas aunque la aplicación se bloquee o experimente algún tipo de excepción no controlada. Existen más opciones del tipo --ignore como --ignore-timeouts para errores de tiempo de espera o --ignore-security-exceptions parará cuando la aplicación de experiencias de cualquier tipo de error de permisos

### c. Fastboot

El uso genérico de la herramienta fastboot es:

```
fastboot [ <option> ] <command>
```

Para actualizar el terminal desde un archivo update el uso de fastboot es:

```
fastboot update <filename>
```

Para flashear el system el recovery y el boot a la vez la sintaxis es:

```
fastboot flashall
```

Para poder flashear el system o el recovery o el boot se aria de la siguiente forma:

```
fastboot flash <partition> [ <filename> ]
```

Por ejemplo para flashear el recovery se aria:

```
fastboot flash recovery nombre_del_archivo
```

Si lo que se pretende es borrar la partición se tiene que usar:

```
fastboot erase <partition>
```

#### ***IV. Características de las ROMS de Cyanogen***

Las ROM Cyanogen son modificaciones del código fuente de Android hechas por un grupo de desarrolladores conocidos por este seudónimo.

He elegido basarme en esta y no en la publicado por AOSP (Android Open Source Project) que es la versión publicada por Google, porque la versión de Cyanogen modificaciones que he considerado beneficiosas como son:

- Privilegios de ROOT, esto permite al usuario tener un control total sobre el terminal pues por razones de seguridad ciertas acciones no están permitidas en la versión original de Android.
- Modificación del Kernel, esto significa que el núcleo del sistema operativo ha sido modificado en especial el JIT ha sido bastante mejorado.
- Modificación del apartado visual, el desarrollador a decidido modificar los fondos de pantalla existentes, integrar nuevos además de cambiar el Launcher por defecto.
- Modificación de la aplicación Ajustes, en la que se ha incluido un apartado con nuevos ajustes que el sistema no trae por defecto.

Todas las ROM de Cyanogen son publicas y se pueden descargar bien en forma de código fuente o ya preparadas para instalar en el móvil.

## V. Habilitar los privilegios de administrador en el terminal

Para poder instalar una ROM diferente de la original o las que se instalan por medio de OTA, tenemos que tener el terminal NEXUS ONE “rooteado”, al aplicar este método hay que tener en cuenta que esto tendrá los siguientes efectos en el terminal:

- Este proceso anula la garantía.
- La ROM podremos instalar ROMS con privilegios de superusuario(root).
- Al desbloquear el bootloader, se realiza un reseteo de fábrica. Es decir: se perderán todos los datos guardados.
- Perderemos la posibilidad de instalar actualizaciones vía OTA

Para poder hacer este proceso tendremos que tener instalado previamente el SDK de Android, ya que necesitaremos las herramientas fastboot y adb que vienen incluidas en el.

Conectaremos el teléfono apagado al ordenador y lo encendemos manteniendo pulsadas las teclas de encendido y la de TrackBall (la bolita blanca) para hacer que el NEXUS ONE entre en modo bootloader tal y como se muestra en la fig. 5.1.

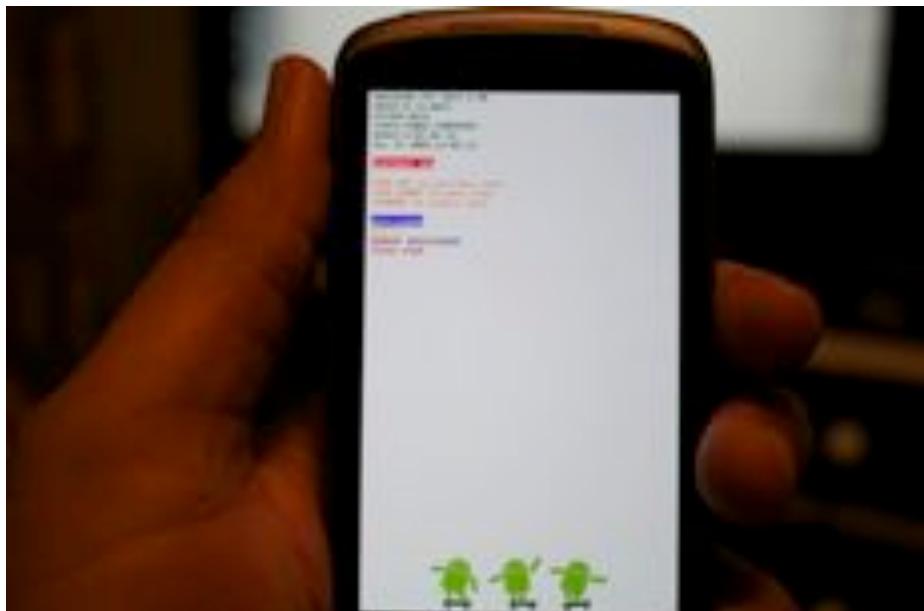


Fig. 5.1 Modo bootloader

En nuestro ordenador abriremos un terminal o ventana de comandos, nos situaremos en la carpeta donde tengamos instalado el SDK y abriremos la carpeta tools:

```
cd <ruta donde tenemos descargado SDK>
```

```
cd /tools/
```

Una vez hecho esto, ejecutaremos comandos fastboot, fastboot-mac, fastboot-linux dependiendo de nuestro sistema operativo, en este ejemplo se toma como ejemplo una maquina Macintosh.

```
./fastboot-mac devices
```

Con este comando conseguiremos ver si el ordenador detecta nuestro terminal NEXUS ONE. Si nuestro terminal es detectado tendría que aparecernos en pantalla un mensaje del tipo de la fig. 5.2.

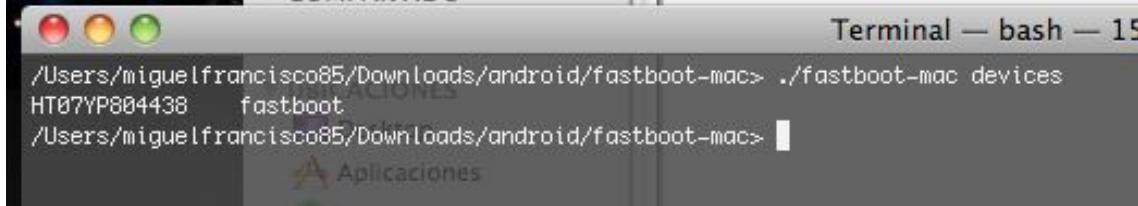


Fig. 5.2 Detección de terminal

El siguiente paso será, desbloquear el terminal, para ello introduciremos el comando:

```
./fastboot-mac oem unlock
```

Una vez introducido esto nos aparecerá en el ordenador un mensaje de error y un mensaje de advertencia en el terminal tal y como muestran las figuras 5.3 y 5.4.

```
Kirans-MacBook-2:fastboot kkiran$ ./fastboot-mac oem unlock
< waiting for device >
... INFOErasing userdata...
ERROR: usb_read failed with status e00002ed
FAILED (status read failed (No such file or directory))
```

Fig. 5.3 Mensaje de error en el ordenador



Fig. 5.4 Mensaje de advertencia en el terminal

Pulsaremos en Yes y apretaremos el TrackBall del terminal, con esto perderemos la garantía pero desbloquearemos el terminal NEXUS ONE.

A continuación cambiaremos el recovery del terminal, para ello introduciremos el comando:

```
./fastboot-mac flash recovery archivo_del_recovery  
./fastboot flash recovery
```

Una vez que se haya instalado el nuevo recovery, apagaremos el teléfono.

## **VI. Ciclo de vida de una aplicación Android**

Una aplicación Android se ejecuta dentro de su propio proceso Linux. Este proceso se crea cuando parte del código de una aplicación necesita ser ejecutado y continuará en funcionamiento hasta que no sea requerido.

Una característica importante, y poco usual, de Android es que el tiempo de vida de un proceso no es controlado directamente por la aplicación. En lugar de esto, es el sistema quien determina el tiempo de vida del proceso basado en el conocimiento que tiene el sistema de las partes de la aplicación que se están ejecutando, qué tan importante es la aplicación para el usuario y cuánta memoria disponible hay en un determinado momento.

Para determinar que procesos deberían ser eliminados ante una condición de baja memoria, Android ordena los procesos en una estructura jerárquica y

asignándole a cada proceso una determinada "importancia", de la cual se distinguen los siguientes tipos de procesos:

#### **a. Foreground process (proceso de primer plano)**

Se trata de un proceso que contiene una Activity con la cual el usuario está interactuando (su método "onResume ()" ha sido llamado) o un IntentReceiver ha sido llamado. Este tipo de procesos son los de prioridad más alta pues están siendo usados en el momento por lo que en caso de falta de memoria serían los últimos en eliminarse.

#### **b. Visible process (proceso visible)**

Es un proceso que contiene una Activity visible en pantalla pero no en primer plano (su método "onPause ()" ha sido llamado), ya que puede darse el caso que la actividad principal no ocupe toda la pantalla y pueda verse esta Activity de fondo. Este proceso es casi igual de importante que los foreground, por lo que solo se eliminaría para mantener a los de primer plano corriendo.

#### **c. Service process (proceso de servicio)**

Es un proceso que aloja un Service que ha sido iniciado con el método startService (). Este tipo de procesos no suelen ser visibles para el usuario pero suelen ser importantes (tal como mantener una conexión con servidores, o reproducir música).

#### **d. Background process (proceso de fondo)**

Consiste en un proceso que contiene una Activity que no está visible para el usuario (su método "onStop ()" ha sido llamado). Normalmente la eliminación de estos procesos no supone un gran impacto para la actividad del usuario.

Generalmente, hay muchos de estos procesos corriendo, por lo tanto el sistema mantiene una lista "LRU" para asegurar que el último proceso visto por el usuario sea el último en ser eliminado en caso que se requiere recolectar memoria.

#### **e. Empty process (proceso vacío)**

Es un proceso que no aloja ningún componente de la aplicación activo. La razón de existir de este proceso es tener una cache disponible de la aplicación para su próxima activación. Es común, que el sistema elimine este tipo de procesos con frecuencia para obtener memoria disponible.

En el esquema de la figura 6.1 se muestran los estados por los que pasa una activity, dónde inicialmente se llama a la función `onCreate()` que es la que llama al `onStart()`, función que permite ejecutar la actividad. Se puede apreciar las variantes que pueden suceder cuando una activity es pausada (`onPause()`) como ser eliminada si es necesario para el sistema.

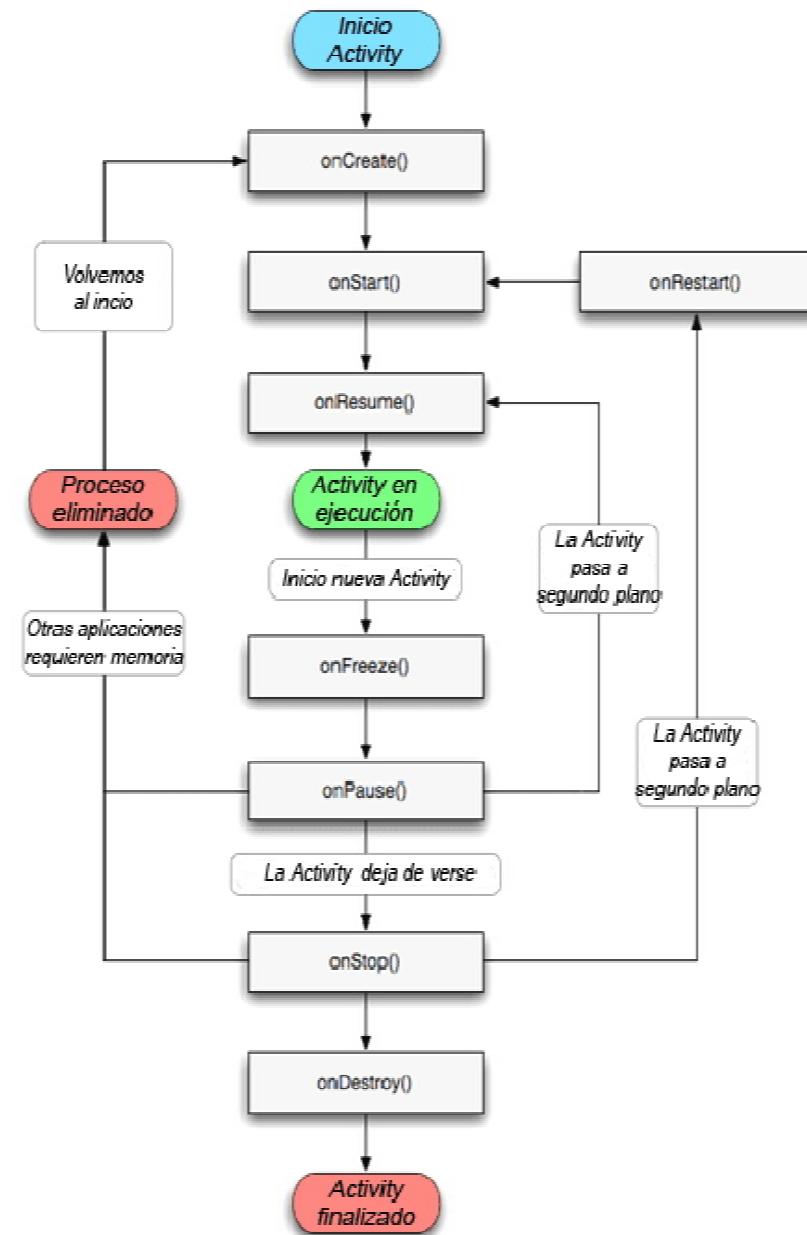


Fig. 6.1 Ciclo de vida de una Activity