 **ESTRUCTURA DE**



**Trabajo realizado por:**

- Miguel Arredondo Calderón

- Jose Miguel Hernández García

- Antonio Jaimez Jiménez

- Robin Costas del Moral

**ÍNDICE**

Introducción e historia

* Introducción e historia......................................3
* Estructura...................................................4
* Gestión de Archivos..........................................5
* Memoria......................................................6
* Gestión de Procesos..........................................7
* Seguridad....................................................9
* Bibliografía.................................................11

**Introducción e historia**

Android es un sistema operativo desarrollado por Google para su uso en dispositivos móviles, tablets y demás dispositivos electrónicos inteligentes. Es decir, diseñado para dispositivos con memoria y procesador más reducidos y menos potentes que los de un ordenador de sobremesa o portátil. Manteniendo dichas limitaciones en mente, la visión que Google tiene en Android es tener un enorme conjunto de APIs (Application Programming Interfaces) de programación y una interfaz de usuario sencilla y fácil de utilizar.

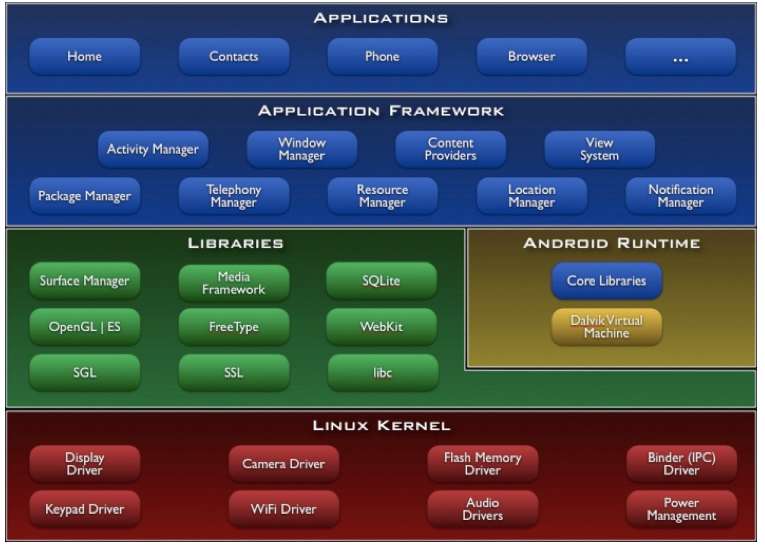
Android se basa en el kernel de Linux, pero en realidad no es puramente una distribución más de Linux. Una distribución de Linux estándar tiene un sistema de ventanas nativo, glibc, y algunas utilidades de serie. Además no tiene una capa de abstracción entre las aplicaciones a nivel de usuario y de las bibliotecas. Existen importantes modificaciones que se han aplicado en el kernel de Linux para adaptarlo a Android, como la forma en que se gestiona la memoria y los procesos, los permisos de seguridad que se basan en el modelo del dispositivo y además es de código abierto.

En Julio de 2005 Google compra Android a sus desarrolladores. Tras esta compra, en 2007 se lanza el Android Software Development Kit y se crea el Open Handset Alliance, un consorcio de varias marcas que desarrollaron los estándares abiertos para los dispositivos móviles y poco más de un año después aparece la beta con versión 0.9 de Android. Un mes más tarde la versión 1.0 es lanzada. Desde entonces Android ha pasado por diferentes versiones hasta llegar a la más reciente a día de hoy, que es la versión 7.1 (Nougat). Algunas de las más famosas son las siguientes:

* 1.0 (Apple Pie)
* 1.1 (Banana Bread)
* 1.5 (Cupcake)
* 1.6 (Donut)
* 2.0/2.1(Eclair)
* 2.2 (Froyo)
* 2.3 (Gingerbread)
* 3.0-3.2 (Honeycomb)
* 4.0 (Ice Cream Sandwich)
* 4.1-4.3 (Jelly Bean)
* 4.4 (KitKat)
* 5.0-5.1 (Lollipop)
* 6.0 (Marshmallow)
* 7.0 (Nougat)

**Estructura**

A continuación se muestra una imagen donde es posible apreciar cómo se estructura el sistema operativo Android:



**Kernel**: Suministra servicios genéricos como los Drivers de los distintos componentes del sistema, tales como la cámara, la memoria flash… Además del manejo de la seguridad, los procesos, la memoria… También provee servicios propios de Android, como el control del consumo de batería, entre muchos otros.

**Librerías nativas:** Se trata de un conjunto de librerías que se encuentran escritas en lenguaje C y C++ usadas por el SO para crear procesos e hilos, asignar memoria, modelado de gráficos en 3D, etc.

**Runtime de Android:** Es la capa compuesta por librerías escritas en JAVA que facilitan el trabajo de creación de nuevas apps ya que las aplicaciones de Android están escritas en JAVA y es por ello que se necesita una Máquina Virtual que ejecute estos programas. El problema es que Android no puede correr máquinas virtuales muy potentes, por lo que Google creó específicamente para Android, una MV denominada Dalvik que permite ejecutar las aplicaciones en dispositivos con Android.

**Entorno de aplicaciones:** Está formado por una enorme cantidad de software que las apps necesitan, el framework como pueden ser los botones, iconos, los gestores de ubicación, de notificaciones, etc.

**Aplicaciones:** Es el bloatware del sistema, se trata de las aplicaciones propias de Android como el teléfono, el gesto de correo, el navegador…

**Gestión de archivos en Android**

Android usa diferentes particiones para organizar los archivos y las carpetas. Cada partición tiene su propia funcionalidad y hay seis particiones principales:

* /boot
* /system
* /recovery
* /data
* /cache
* /misc

A continuación se detallan un poco estas particiones:

**BOOT**

Esta es la partición de arranque de cualquier dispositivo Android, incluye el kernel y el “swap”. Sin esta partición sería imposible que el dispositivo arrancara.

**SYSTEM**

Esta partición contiene el Sistema Operativo completo, incluyendo la GUI (Interfaz gráfica de usuario) y las aplicaciones de sistema que vienen preinstaladas en el dispositivo.

**RECOVERY**

Esta partición está especialmente diseñada para las copias de seguridad.

**DATA**

Es la partición que guarda datos del usuario, contactos, SMS, ajustes y todas las aplicaciones que hayas instalado. Cuando se restablecen los datos de fábrica, en realidad lo único que se hace es borrar esta partición

**CACHE**

Esta es la partición donde Android almacena datos frecuentemente usados y componentes de aplicaciones. Borrar esta partición no afecta a tus datos personales, simplemente elimina los datos aquí almacenados, que además, volverán a aparecer automáticamente a medida que se usa el dispositivo.

**MISC**

Esta partición contiene ajustes misceláneos del sistema en forma de interruptores on/off. Estos incluyen configuraciones USB y ajustes del hardware. Si esta partición falla o se encontrase corrupta, varias funcionalidades del dispositivo podrían verse afectadas.

Otras particiones adicionales que se encuentran en la tarjeta SD son:

**SDCARD**

En términos de uso, este es el espacio que se usa para almacenar archivos en general, documentos, etc. Borrarlo es perfectamente seguro mientras guardes todos los archivos que necesites en otro lugar de la memoria interna. Sin embargo algunas aplicaciones pre-instaladas pueden almacenar sus ajustes en esta partición y borrarla hará que perdamos dichos datos.

Si el dispositivo tiene tanto una tarjeta SD interna como otra externa: /sdcard se refiere a la partición de la tarjeta interna y /sdcard/sd o /sdcard2 para la externa.

**Memoria**

Android es un Sistema operativo basado en Linux con el kernel 2.6.x, simplificada para manejar la mayoría de tareas. Utiliza bibliotecas nativas en C abiertas. Todas las operaciones básicas del sistema operativo como de E / S, gestión de memoria, y así sucesivamente, son manejados por el de kernel de Linux.

**La gestión de memoria** Al igual que Java y NET., Android utiliza el entorno de ejecución y la máquina virtual para gestionar la memoria de la aplicación. A diferencia de cualquiera de estos dos marcos, el entorno de ejecución de Android también maneja los tiempos de vida del proceso. Android asegura la respuesta de la aplicación, deteniendo y matando a los procesos que obstaculizan la fluidez y libera recursos para las aplicaciones de mayor prioridad. Cada aplicación Android se ejecuta en un proceso independiente dentro de su propia instancia de Dalvik, renunciando a toda responsabilidad de la memoria y la gestión de procesos.

Dalvik y el entorno de ejecución de Android se posicionan en la parte superior de un núcleo de Linux que se encarga de la interacción de bajo nivel del hardware, incluyendo los drivers y la gestión de memoria, mientras que el conjunto de API proporciona acceso a todos los servicios de bajo nivel, características y hardware.

Dalvik Virtual Machine es una máquina virtual basada en registros que ha sido optimizada para asegurar que un dispositivo puede ejecutar múltiples instancias de manera eficiente. Se basa en el kernel de Linux para la gestión de memoria de bajo nivel.

**La máquina virtual Dalvik**

Uno de los elementos clave de Android es la máquina virtual de Dalvik. En lugar de utilizar una tradicional máquina virtual Java (VM), tales como Java ME (Java Mobile Edition), Android utiliza su propia máquina virtual personalizada y diseñada para asegurar que la multitarea se ejecuta de manera eficiente en un único dispositivo.

La máquina virtual Dalvik utiliza el dispositivo del kernel de Linux subyacente para manejar bajo nivel de funcionalidad, incluyendo la seguridad, la planificación de procesos, y la gestión de la memoria.

Todo el hardware de Android y acceso a los servicios del sistema se gestiona mediante Dalvik como un nivel intermedio. Mediante el uso de una máquina virtual para organizar la ejecución de aplicaciones, los desarrolladores tienen una capa de abstracción que asegura que nunca tendrá que preocuparse de una aplicación de hardware en particular. La máquina virtual Dalvik lanza procesos ejecutables Dalvik, un formato optimizado para asegurar la mínima huella en la memoria. Los ejecutables .Dex se crean mediante la transformación de las clases de Java lenguaje, compilado utilizando las herramientas proporcionadas en el SDK.

**Gestión de procesos**

Cuando se inicia el componente de alguna aplicación, y esta no tiene ningún otro componente en ejecución, el sistema creará un nuevo proceso para dicha aplicación con un único hilo de ejecución, por defecto, todos los componentes de una misma aplicación se ejecutan dentro de este mismo proceso y a medida que se inician más componentes, se anexan al proceso.

En un momento dado es posible que el SO quiera finalizar algún proceso, cuando la memoria es baja y la requieren otros procesos que son inmediatamente más necesarios para lo que el usuario desea. En consecuencia, los componentes de la aplicación que se ejecutan en el proceso que se cancela son destruidos. Un proceso se inicia de nuevo para esos componentes cuando vuelve a haber trabajo para ellos.

Android pondera la importancia relativa de los procesos para el usuario. Por ejemplo, cierra más antes un proceso que aloja actividades que ya no se ven en la pantalla que un proceso que sí muestra actividades visibles. La decisión de si se debe finalizar un proceso, por lo tanto, depende del estado de los componentes que se ejecutan en ese proceso.

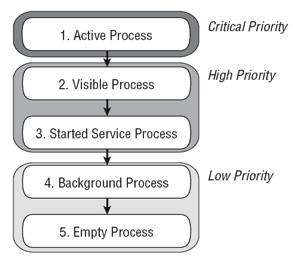
**Ciclo de vida de los procesos**

El sistema Android trata de mantener el proceso de una aplicación el mayor tiempo posible; pero, finalmente, necesita quitar los procesos viejos a fin de recuperar memoria para procesos nuevos o más importantes. Para determinar qué procesos mantener y cuáles finalizar, el sistema coloca cada proceso en una "jerarquía de importancia" según los componentes que se ejecutan en el proceso y el estado de esos componentes.

Los procesos con la importancia más baja se eliminan primero, luego, los siguientes con la importancia más baja y así sucesivamente, según sea necesario para recuperar recursos del sistema.

En caso de haber varios procesos con el mismo nivel de prioridad, se mata al que menos se haya usado recientemente.

Las prioridades son:



**Primer Plano:** Son los procesos que contienen las aplicaciones o servicios que el usuario está utilizando en ese momento o contienen algún servicio del que depende el proceso de la aplicación con la que el usuario está interaccionando. Generalmente solo existen unos pocos procesos en primer plano, en un momento determinado. Se cancelan como último recursos cuando el sistema ha llegado a un punto de paginación de memoria, por lo que finalizar estos procesos es necesario para mantener la capacidad de respuesta de la IU.

**Visible:** Estos son procesos que contienen actividades que aunque no están en primer plano, son visibles para el usuario, o bien contienen algún servicio del que depende otro proceso visible. Un proceso visible se considera extremadamente importante y no se lo finalizará a menos que sea necesario para mantener todos los procesos en primer plano en ejecución.

**Servicio:** Son procesos que contienen servicios que aunque no interaccionan directamente con el usuario le incumben directamente a él, como puede ser el reproductor de música. El sistema los mantiene en ejecución a menos que no haya suficiente memoria para retenerlos junto con los procesos en primer plano y visibles.

**Segundo plano:** Estos procesos, cuyos componentes no afectan directamente a la experiencia de usuario, suelen ser los que se “matan” de forma más común, por lo que están siempre preparados para reanudarse en el punto en el que se habían bloqueado cuando vuelvan a iniciarse. Por lo general, se ejecutan muchos procesos en segundo plano, por lo que se los mantiene en una lista LRU (Less Recently Used) para garantizar que el proceso con la actividad que el usuario vio más recientemente sea la última en finalizarse.

**Vacío:** Un proceso que no tiene ningún componente de la aplicación activo. El único motivo para mantener este tipo de proceso activo es por fines de la memoria caché, para mejorar el tiempo de inicio la siguiente vez que un componente necesite ejecutarlo. El sistema suele finalizar estos procesos para equilibrar los recursos generales del sistema entre las memorias caché de procesos y las memorias caché de kernel subyacentes.

La prioridad que se le asigna a un proceso es la de sus componentes con mayor prioridad. Esto quiere decir que, si uno de sus componentes con mayor prioridad es visible, entonces la prioridad del proceso de esos componentes es visible.

Puede darse el caso de que un proceso sea dependiente de otro que tenga menor prioridad, estos casos, la prioridad del segundo proceso tendría que igualarse a la del proceso dependiente puesto que un proceso no puede tener menor prioridad que aquellos que dependan de él.

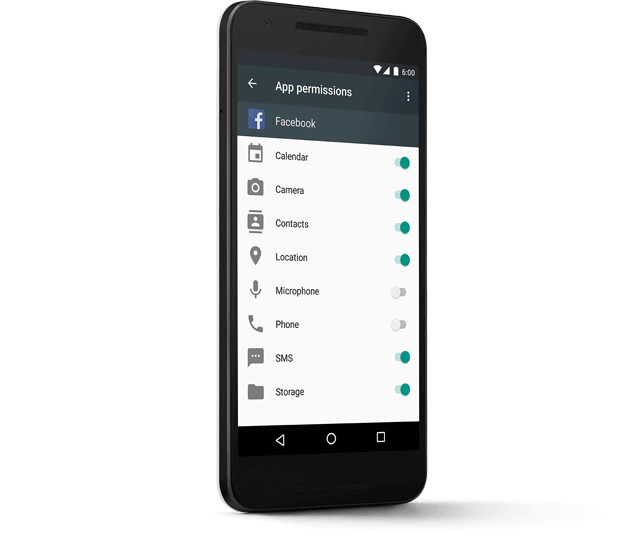
**Seguridad en Android**

La seguridad es un aspecto clave en cualquier sistema operativo, si no se le dedica tiempo a proteger el sistema este queda expuesto a todo tipo de riesgos que pueden dañarlo o exponer nuestra información más personal.

En las plataformas móviles, una de las principales formas de conseguir romper la seguridad es mediante las aplicaciones maliciosas. Esto ocurría sobre todo antiguamente, cuando empezaban a salir los primeros SO para Smartphones, sin embargo hoy en día se ha avanzado mucho en este aspecto y es uno de los principales motivos por los que se han de mantener actualizados dichos SO, vamos a estudiar como nuestro sistema operativo Android ha avanzado en este campo.

Si nos vamos a otras plataformas, y en este caso a una de las dominantes del mercado (junto con Android), iOS, vemos que toda aplicación debe ser validada por Apple para que esta pueda ser publicada en su tienda, y además no se permite instalar aplicaciones exteriores a dicha tienda. Sin embargo cuando nos fijamos en Android nos encontramos con un sistema planteado como software libre, por lo que esto presenta varias vulnerabilidades a la hora de proteger a sus usuarios.

Todas las aplicaciones en un terminal con Android se someten a un sistema de permisos, por el cual es el propio usuario el que elige a qué si y a que no puede acceder una aplicación.



“**Cada aplicación de Android se aloja dentro del**

**entorno protegido sandbox que ayuda a proteger tus**

**datos personales. De este modo, las aplicaciones que**

**instales no podrán acceder a información como tus**

**fotos o tu ubicación a menos que les des permiso para hacerlo.**

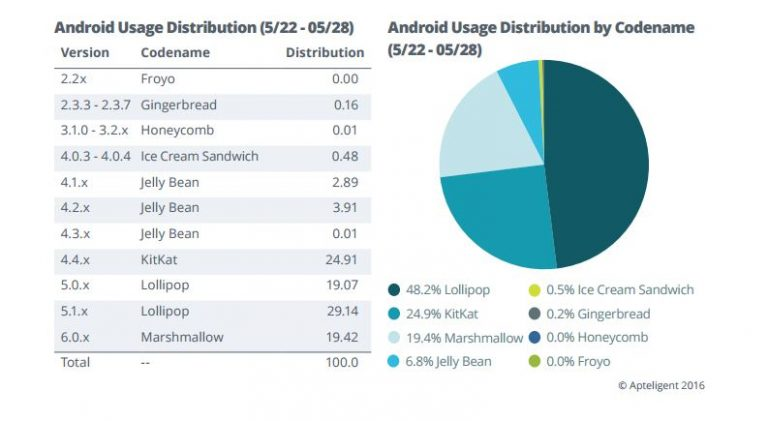
**Se trata de tu privacidad, así que tú tienes el**

**Control.”** - Página web de Android

Sin embargo no ha sido hasta una de las más recientes versiones, con tan solo un par de años de longevidad, la versión 5.0 de Android, cuando Google (actual dueño del SO) empezó a concienciarse sobre la seguridad y hasta dónde podía llegar una aplicación.

Si cogemos por ejemplo el **Stagefright**, una vulnerabilidad bastante grave que hasta hace poco era un peligro en cualquier dispositivo con Android, en el cual el atacante era capaz de ejecutar código de forma remota en nuestro dispositivo y así, controlarlo, vemos que afecta a la inmensa mayoría de dispositivos (desde la versión 2.2 hasta la 5.1 de Android). Al menos ha servido para que Google pueda centrarse en la seguridad de su sistema.

Pero sin duda el mayor peligro que posee este sistema es la **Fragmentación**. Como ya se sabe, Android posee un gran número de versiones, y al contrario que otros SO, este no te obliga a tener la versión más actualizada del mismo, por lo que los dispositivos quedan distribuidos desde la versión 2.2 hasta la última 7.1:

U5j35w8PGcahQ9jxHAglQkKDJLPXhaQu0HISCNbmtoIjj9oORT4gf4M0LaN6oCThd23iw6EkDf-DByCq31TmMnJZyry8N-azYEpOvltN5pYeNsGrjehFXcM-zCBQsWigxbxaHlMJ_JIjrN2_ow**-Gráfica de un estudio realizado por Apteligent a mediados de 2016(por lo que no se incluye la versión 7.x todavía).**

Vemos que tan solo el 19.4% usa la última versión de Android, es decir que el 80.6% de los usuarios de este SO se encuentran expuestos a las posibles brechas de seguridad que se hayan podido solventar en la nueva versión. Esto es culpa de los fabricantes, que no optan por adaptar dichas versiones a sus dispositivos para así lanzar dicha actualización al público.

Tras analizar la seguridad de Android en profundidad, nos damos cuenta que poco a poco están optando a tomar un mayor control sobre su dispositivo. En la propia página web oficial de Android encontramos lo siguiente: “**Todas las aplicaciones de Android pasan por un proceso de rigurosas pruebas de seguridad antes de aparecer en Google Play Store. Comprobamos todos los desarrolladores de aplicaciones y suspendemos a aquellos que infringen nuestras políticas. De esta forma, antes de instalar cualquier aplicación, sabes que hemos verificado que es segura”**. Lo cual nos recuerda un poco a lo anteriormente mencionado sobre iOS, porque tal vez la mejor forma de proteger a los usuarios sea tener un mayor control sobre su dispositivo y lo que realizan con el mismo.

**Bibliografía**

<http://www.elandroidelibre.com/2016/06/fragmentacion-en-android-problema-menor.html>

<https://www.android.com/intl/es_es/security/overview/>

<http://www.androidcurso.com/index.php/tutoriales-android/41-unidad-7-seguridad-y-posicionamiento/280-los-tres-pilares-de-la-seguridad-en-android>

<https://www.xataka.com/moviles/se-abre-una-nueva-era-para-la-seguridad-en-android-y-todo-gracias-a-stagefright>

<http://www.howtogeek.com/161225/htg-explains-how-android-manages-processes/>

<https://developer.android.com/guide/components/processes-and-threads.html>