Nahuel Maika Fanessi Villarroel
Erik Alba

a)

Al hablar de la arquitectura de Android podemos dividir dicho tema en 4 capas principales:

1. ***La capa de kernel***
2. ***La capa de librerias nativas***
3. ***La capa de los frameworks de aplicaciones***
4. ***La capa de aplicaciones***

La capa de kernel provee funcionalidades básicas del sistema como el manejo de procesos, el manejo de memoria, el manejo de dispositivos (incluyendo la cámara, la pantalla, el teclado,etc). El kernel utilizado para en Android OS es un kernel Linux 2.6 ya que Linux es muy bueno en operaciones básicas pero igualmente Google mejoró dicho kernel para manejar de mejor manera las necesidades de los móviles, como el manejo de memoria,el de procesos, el manejo de energía, un mecanismo especial para la intercomunicación de procesos en ejecución y un mejor uso de los recursos limitados del sistema. Algunas de las mayores mejoras son las siguientes:

- **Alarm Driver:** El kernel de android lo utiliza para realizar ciertas operaciones y planificar adecuadamente cuando “despertar” una aplicación dependiendo de los eventos que tomen lugar.
- **Binder:** Se implementó el binder para manejar de manera más eficiente la intercomunicación de procesos ya que consumen muchos recursos. Además, hace que las llamadas de mensajes entre procesos sean más rápidas y eficientes al utilizar memoria compartida. Otro motivo para su implementación fue lo inseguro que era el IPC tradicional de linux.
- **Power Managment**
- **Low Memory Killer:** Esta mejora es manejada por el manejador Out of memory (OOM). Cuando un dispositivo se está quedando sin memoria, este elige un proceso y lo elimina, liberando la memoria utilizada. El proceso a eliminar es seleccionado por el kernel quien elimina los procesos en segundo plano que se estén ejecutando pero que no se estén utilizando.
- **Kernel Debbuger**: Tanto android como linux son de código abierto y cualquiera puede realizar cambios en ellos. Por este motivo, para mantenerse al tanto de que los cambios funcionen correctamente se provee con este debbuger.
- **Logger:** Carga en el sistema grandes mensajes con el propósito de generar problemas incluyendo el logger buffer principal, el logger buffer de eventos, el de radio y el buffer del sistema.
- **Ashmen:** Se trata de la memoria compartida de android, la cual fue agregada en el kernel para facilitar compartir memoria y la conservación, la cual provee mejor soporte para dispositivos de poca memoria.

La capa de librerías nativas provee librerías que le permite a los dispositivos manejar diferentes tipos de información específica del hardware. Esta capa está dividida en dos partes, una es Android Library y la otra es Android Runtime. Las librerías en Android Library están escritas en C++ y se encargan de realizar todas las tareas pesadas para proveer mucho poder a la plataforma android.
Algunas de las librerías más importantes son:

- **Libc:** Android implementa su propia versión de la librería bionic libc la cual es pequeña en comparación con la librería GNU libs.

- **SQLite:** Es una base de datos relacional utilizada para guardar información que utilizan las aplicaciones según la necesiten.
- **Media Framework:** Provee grabación y reproducción de una gran cantidad de formatos de audio, video e imágenes.
- **Surface Flinger**: Provee un manejador de renderizado que combina superficies 2D con superficies 3D.
- **Webkit:** Es un motor de búsqueda que se utiliza para mostrar contenido HTML.

La segunda parte, Android Runtime, consiste en la Máquina Virtual Dalvik (DVM) y librerías java centrales. Todas las aplicaciones de java y la mayoría de sus frameworks están escritos en Java. En vez de utilizar la Máquina Virtual de Java (JVM), android utiliza su propia DVM que está diseñada para sistemas pequeños que proveen poca memoria RAM y un procesador lento. DVM tiene su propio código en formato byte que se ajusta a las necesidades de los dispositivos con android. Este código está más comprimido que el típico código en formato byte de Java. Además, ejecuta archivos ejecutables Dalvik (.dex) que está optimizado para un uso mínimo de memoria. DVM provee portabilidad, consistencia en ejecución y permite que cada aplicación corra su propio proceso con su propia instancia de DVM.

La capa de los frameworks de aplicaciones provee una gran cantidad de interfaces de programación de aplicaciones (API's) y mejores servicios en forma de clases Java. Estas API's están disponibles para todo el mundo que quiera crear aplicaciones android. Hay diferentes tipos de componentes de aplicaciones y cada uno tiene un ciclo de vida distinto y un propósito distinto. Alguno de estos componentes son:

- **Activity:** Representa una pantalla única con interfaz de usuario. En las aplicaciones, las actividades trabajan en conjunto para mejorar la experiencia del usuario. Siempre hay una actividad principal que se le muestra al usuario cuando corre la aplicación por primera vez y para realizar otras acciones, está actividad principal puede iniciar otras actividades. Cuando una nueva actividad comienza, la anterior se detiene y se guarda en una pila con su estado actual, para algún uso posterior.
- **Servicies:** Un servicio es un componente que corre en segundo plano y no provee una interfaz. Los servicios realizan operaciones largas en ejecución y trabajan para procesos remotos. Además, los servicios corren en el hilo principal de la aplicación por defecto. Una actividad puede conectarse a un servicio en ejecución para comunicarse con otros servicios
- **Content Providers**: Maneja cómo acceder y compartir información desde otras aplicaciones como podría ser la base de datos SQLite, el archivo de sistema o cualquier otra locación de guardado de información persistente.
- **Package Manager:** Android utiliza una extensión especial para los paquetes llamada APK(Android Package). Este componente mantiene información de todos los paquetes disponibles en el sistema y los servicios que los mismos ofrecen.
- **Window Manage**r: Se encarga de dibujar las diferentes ventanas en pantalla para la interacción mientras que Surface Flinger maneja el output de la pantalla.
- **Hardware Services**: Interactúa con la capa de abstracción de hardware para acceder a los dispositivos.
- **Telephony Services:** Maneja todas las actividades relacionadas a llamadas de teléfono y mensajes de texto.

- **Location Services**: Es utilizado para manejar la ubicación del dispositivo. Utiliza el GPS o las torres de teléfono para brindar la ubicación del dispositivo.

La capa de aplicaciones es la capa más visible, contiene aplicaciones pre-instaladas como el marcador de teléfono, la aplicación de mensajes, algún buscador predefinido, etc. La mayoría de los usuarios interactúan con esta capa, incluso descargan aplicaciones desde la Play Store dependiendo de sus necesidades.

b)

Los elementos más representativos para este tipo de sistemas son:

- Dalvik Virtual Machine (DVM)
- Application Framework
- Optimized Graphics
- Integrated Browser
- SQLite
- GSM Technology
- Edge
- 3G
- 4G
- Media Support
- Camera
- Bluetooth
- WiFi

c)

El artículo en sí nos pareció interesante, ya que nos brindó información sobre muchísimas partes de Android que desconocíamos. En un principio brinda información general sobre el sistema operativo Android desarrollado por la compañía Google, cómo está evolucionando el uso de teléfonos móviles en los últimos años y como se utilizan 3.5 veces más los dispositivos móviles que las PC's. Principalmente Android OS está desarrollado principalmente para teléfonos móviles y tablets. Pensado para dispositivos con una batería que se les agota rápidamente y equipado con hardware de posicionamiento global (GPS),WiFi,UMTS, pantalla táctil, entre otras cosas. Las aplicaciones Android son, en su mayoría, desarrolladas usando el lenguaje de programación Java utilizando Android Development Kit (SDK) el cual incluye un debbuger, librerías, QEMU, tutoriales y documentación. Si bien hemos programado en Android, principalmente con Android Studio, nunca nos habíamos adentrado en el tema más allá de la parte de programación de aplicaciones móviles. No sabíamos que se habían realizado tantos cambios en el kernel de Android para poder obtener el sistema operativo que es hoy en día, inclusive lo que más nos sorprendió fue el Binder ya que sin esta mejora de kernel la intercomunicación entre procesos sería muy pesada ya que consumiría muchos recursos y no hubiera sido posible la creación de los primeros smartphones con Android, los cuales tenían poca memoria RAM y un procesador muy lento. Estábamos al tanto de los problemas de la seguridad de Android ya que al ser un sistema de código abierto que puede ser corrido por múltiples dispositivos es lógico que existan grandes vulnerabilidades, es el típico trade-off de la computación. Una gran vulnerabilidad de Android es que al estar basado en el kernel de Linux implementa una

arquitectura monolítica en la que los componentes están interconectados y acoplados en una misma pieza. Muchos desarrolladores de dispositivos android usualmente utilizan drivers personalizados para su hardware y eso puede provocar fallas de seguridad ya que dicho código de los drivers no son controlador y testeados por la comunidad de Linux.
El hecho de rootear un dispositivo móvil también puede comprometer la seguridad del mismo. Algunas aplicaciones o usuarios rootean sus dispositivos, es decir que le/se brinda al usuario permisos de administrador, y esto hace que su dispositivo deje de ser seguro ya que se rompe la barrera de integridad del kernel. Esto causa que el kernel modificado pueda deshabilitar las medidas de seguridad de android y así infectar el dispositivo con algún malware que cause que se filtre información privada. En comparación con otros sistemas operativos móviles como Apple's iOS o Black Berry OS, Android no es seguro. Esto no quiere decir que android sea completamente inseguro, solamente es más vulnerable a fallos en seguridad comparado con otros sistemas operativos. Pero eso también es lo que lo hace un sistema operativo tan versátil, que inclusive uno como programador puede crear su propia ROM, probarla en su propio móvil o descargar una ROM y modificar el sistema operativo a gusto propio.