1. Buenos días, me llamo Diego Poderoso Novo y les voy a exponer en lo que ha consistido el proyecto “Diseño de estrategias para la gestión del audio en teléfonos inteligentes sobre el Sistema Operativo Android”. Los contenidos que se van a abordar a lo largo de la presentación son los siguientes:

2. Introducción, dónde se comentarán la motivación y los objetivos de este proyecto

Situación actual del audio en el Sistema Operativo Android

Pasaré a explicar tres casos prácticos que se han abordado en la realización de este TFG como son: los modos de audio, las curvas de volumen y las rutas del audio, en los cuales se expondrá un problema o mejora y se implementará una solución.

Finalmente procederé con las conclusiones y las futuras líneas de desarrollo.

3. Comencemos

4. Este proyecto se ha realizado en el entorno de la Cátedra BQ. BQ es una empresa española que tiene en su catálogo de productos una amplia gama de teléfonos inteligentes con el Sistema Operativo Android, el cual está presente en el 87% de smartphones hoy en día.

En estos dispositivos el audio es un tema muy importante, pues han aparecido nuevas formas de comunicación, a parte de las llamadas, que requieren de una buena calidad de audio. Por otra parte, mucha gente utiliza su teléfono como centro multimedia en su día a día.

El caso es que el código Android es muy complejo, pesando en torno a 100 gigas, y por tanto surge la duda ¿Cómo gestionamos el audio?

5. Por tanto, el objetivo principal de este proyecto es el diseño de nuevas estrategias de gestión del audio para teléfonos inteligentes sobre Android. Para ello se analizará su funcionamiento, se detectarán los problemas existentes o posibles mejoras, y se realizarán las implementaciones necesarias para solventarlos.

De esta manera se pretende hacer de nexo entre el departamento de audio, para los cuales el sistema operativo Android es una caja negra y el departamento de firmware.

Este proyecto no se ha limitado a ninguna de las capas del Sistema Operativo, y se ha trabajado sobre las versiones 6.0 y 7.1.1, que son las últimas versiones disponibles.

6. El audio en el Sistema Operativo Android se divide en cinco capas diferentes.

7. La capa superior es el framework de la aplicación, que será la capa sobre la que el desarrollador de aplicaciones trabaje.

El JNI o Capa nativa de Java se encarga de traducir las peticiones del framework de la aplicación a peticiones al framework del sistema, normalmente descrito en C o C++.

El framework nativo del sistema incluye todo lo referente a la política de audio, rutado, volumen, comunicación entre dispositivos, abrir flujos de datos, etcétera.

La HAL proporciona una capa de abstracción del hardware del dispositivo, que es utilizada, entre otros usos, para configurar las distintas rutas del audio.

Finalmente, la capa inferior es el Kernel de Linux modificado para soportar los controladores del hardware del dispositivo.

8. El primer caso de estudio ha sido el tratar con los modos de audio. En este apartado se ha trabajado en la capa superior del sistema, la JNI ya comentada.

9. La necesidad que se ha encontrado en las asociaciones de los distintos modos de audio, que permiten clasificar el audio dentro del sistema operativo,  no se diferencia entre un tono de llamada Voz IP del resto de modos de audio. Por tanto se ha desarrollado un nuevo modo de audio para este caso de uso.

Esta implementación se ha realizado sobre la versión 6.0.

Para poder diferenciar cuando se trataba de un tono de llamada Voz IP se ha hecho uso de los tipos de flujos de audio, ya que existe uno que identifica todos los tonos de llamada.

10. Como el tono de llamada tradicional ya tiene un modo de audio, si el modo de audio es el normal y además el flujo es de tono de llamada, se clasifica como el modo de audio de tono de llamada IP. Para salir de él puede ser, bien porque no se coja la llamada, o bien si se coge se pasa al modo de audio en comunicación, que es propio de las llamadas Voz IP.

11. Las pruebas realizadas han consistido en realizar llamadas Voz IP al dispositivo que incluía dicha implementación, mediante las aplicaciones de Skype y Hangouts.

<< LLAMAR CON SKYPE>>

La prueba con Skype ha sido satisfactoria.

Sin embargo al recibir una llamada de Hangouts

<<LLAMAR CON HANGOUTS>>

se produce un cierre forzoso de la aplicación. Esto se debe a que se modifican las APIs, la interfaz de programación para aplicaciones, sin notificarlo a los desarrolladores, y por tanto se está aplicando un modo de audio distinto al que la aplicación ha pedido.

12. El segundo caso de estudio ha consistido en comprender las asociaciones de las curvas de volumen, así como su configuración. Este apartado abarca parte de la política de audio definido en el framework nativo.

13. Las curvas definidas en el sistema tienen una relación directa con la barra de volumen de la interfaz de usuario. De esta manera, si el usuario modifica la barra de volumen, se atenuará o aumentará el volumen según se haya definido en la curva asociada a cada caso de uso.

La manera de definir las curvas de volumen consiste en asociar un conjunto de puntos, cada uno con su índice y su atenuación, y realizar una interpolación lineal para obtener la atenuación en cada segmento definido. Por ejemplo, si el usuario tiene la barra de volumen de la alarma a la mitad, eso se corresponderá con un índice de 50 y una atenuación asociada de -25 decibelios.

14. El cambio de versión, de Android 6.0 a 7.1.1 trajo consigo un cambio en la forma de definir las curvas de volumen del sistema, que son las que marcan cómo se aumenta o disminuye el sonido. Por tanto el objetivo de este apartado ha consistido en comprender que ha cambiado, y configurar las nuevas curvas de volumen de la misma manera que se tenían en la versión anterior.

15. Las diferencias, y mejoras que ha traído el cambio de versión son las siguientes:

Están descritas en XML en lugar de C++, lo que permite que no sea necesario recompilar el código cuando se modifiquen dichas curvas, sino que se puede sustituir el fichero en el sistema.

En Android 6.0 existían dos limitaciones por parte del código, el límite inferior de la curva no se aplicaba, sino que se aplicaba una atenuación máxima de -758 decibelios. Con el límite superior pasaba algo similar, imponiéndose un nivel de 0 decibelios.

Con el cambio de versión, la limitación superior desaparece, y se aplica el máximo volumen configurado en la curva.

Las curvas de volumen se definen como un conjunto de cuatro puntos en la versión 6.0. En la versión 7.0 se pueden definir con dos o más puntos, lo que permite una mayor flexibilidad.

El último cambio tiene que ver con la manera de definir la atenuación, en la nueva versión se utilizan los milibelios en lugar de los decibelios, lo que permite operar con números enteros, lo que es más eficiente.

16. El tercer caso de estudio ha consistido en comprender cómo se ruta el audio a bajo nivel. Esta lógica de rutado tiene lugar en la HAL y afectará al kernel según la ruta escogida.

Un esquema del rutado del audio a bajo nivel es el que tenéis a continuación.

17. Las rutas, tanto a nivel de DSP como a nivel hardware, se seleccionan lo primero mediante un fichero de configuración del microcontrolador.

Las rutas, o topologías, del DSP se configuran mediante fichero ACDB que podremos modificar.

Las rutas hardware, que definen por donde de rutarse en el códec de audio, se configuran mediante un fichero de configuración XML que también se puede modificar.

En el dispositivo con el que se ha trabajado incluye dos códecs de audio distintos, el WCD que es propietario del fabricante del micro, Qualcomm, y el TFA que se utiliza en exclusiva para el altavoz, y que incluye su propio DSP.

Una vez se ha introducido el rutado del audio, voy a explicar como se ha aplicado esto a problemas que se han encontrado a este nivel.

18. La supresión de ruido para grabación en el DSP no estaba activa, como consecuencia de que tardaba unos segundos en activarse y por tanto el ruido era muy apreciable durante esos primeros segundos, produciéndose un cambio brusco al empezar a hacer efecto.

Tras descubrir la existencia de una nueva versión de dicho módulo de supresión de ruido, se han creado dos tipologías distintas que hacen uso de dicho módulo. Una para la grabación mono y otra para la estéreo.

Estas topologías se han aplicado a la grabación de audio en las distintas aplicaciones de mensajería, a la grabación con la aplicación Grabadora de Android, y a la grabación de un audio con el micrófono integrado de los cascos.

Para aplicar estos cambios ha sido necesario modificar los ficheros ACDB que configuran el DSP.

19. En la topología que incluye supresión de ruido para la grabación destacan: un filtro paso alto, un amplificador con ganancia de 18 decibelios, el módulo de supresión de ruido en su versión dos, y un compresor multibanda.

20. Con respecto a la topología anterior, la topología para grabación estéreo disminuye la ganancia del amplificador a 10 decibelios, para que al combinar ambos canales no sature el sonido. El resto se mantiene igual.

21. Para probar si el funcionamiento de estas topologías es el esperado, se han realizado dos grabaciones mono, con la aplicación de Grabadora. Una con el módulo de supresión de ruido activo, y otra con dicho módulo desactivado.

<<PONER AMBAS GRABACIONES>>

Como se ha podido observar el nivel de ruido de fondo disminuye considerablemente, y por tanto los resultados son más que satisfactorios.

22. Otro problema en el rutado del audio lo encontramos con algunas aplicaciones que dan servicio de llamadas Voz IP. Ya que, dependiendo de la aplicación, no se interpreta como una llamada al menos a nivel de DSP, sino que se interpreta como una reproducción y una captación de audio simultáneas. Esto nos imposibilita el uso de procesado exclusivo para voz.

Por otro lado, con algunas aplicaciones, cuando se pone el modo manos libres no se capta el sonido por el micro secundario, sino que se mantiene por el principal, lo que provoca que el sonido que sale por el altavoz se realimente por el micro principal. Este comportamiento tampoco es el esperado.

Todos estos fallos se deben a peticiones incorrectas por parte de las aplicaciones, y por tanto no se puede forzar el comportamiento deseado, pues no depende del sistema.

23. La solución que se ha implementado para disminuir las consecuencias ha sido crear nuevos dispositivos a nivel de código, que son los que tienen asociadas una ruta del DSP y una ruta hardware, y configurar dichas rutas como se quiera para obtener una experiencia aceptable. De esta manera se puede asegurar que dicha configuración no afectará al resto de casos de uso.

Ha sido necesario crear dispositivos: para la reproducción de la llamada por el auricular, para la reproducción por el altavoz en modo manos libres y para la captación de audio cuando se está reproduciendo por el auricular. Para la captación de audio cuando se está en modo manos libres no ha sido necesario crear un nuevo dispositivo, pues el existente se aplica sólo en este caso de uso. Lo que sí ha sido necesario es modificar la ruta hardware de dicho dispositivo para que soporte la captación de audio por el micro secundario.

Los cambios realizados han sido tanto a nivel de código como de los ficheros de configuración del DSP.

24. Para probar el comportamiento de las llamadas Voz IP se han realizado llamadas con dos aplicaciones bastante conflictivas, como son Skype y Viber.

Se ha comprobado la asociación de los nuevos dispositivos y si se hace uso de la captación por el micro secundario en el modo manos libres.

Las pruebas han sido satisfactorias, interpretándose como reproducción y captación simultánea pero haciendo que esta circunstancia no afecte al usuario de la aplicación.

<<LLAMAR CON SKYPE Y PROBAR LA CAPTACIÓN ESTÉREO>>

25. Por último, las conclusiones y el futuro desarrollo son los siguientes.

26. Como conclusión principal, se puede decir que se ha mejorado la gestión del audio sobre Android, sin haberse limitado a ninguna de las capas del sistema. Ya que se ha analizado el funcionamiento de los modos de audio, que es una clasificación de alto nivel, las curvas de volumen , definidas en la capa intermedia, y el rutado del audio a bajo nivel.

Se han detectado problemas o posibles mejoras como la distinción del tono de llamada en una llamada Voz IP. Aunque se ha concluido que modificar las APIs no es conveniente, ya que puede dar problemas de incompatibilidad en ciertas aplicaciones.

Se han configurado las nuevas curvas de volumen que ha traído el cambio de versión, viéndose todas las mejoras que incluía.

Se han implementado rutas que soportan la supresión de ruido para las grabaciones, y se ha asegurado el funcionamiento de aplicaciones conflictivas que ofrecen llamadas Voz IP.

27. Como futuro desarrollo se propone estudiar el problema de la latencia del audio en Android, aún demasiado alta y variable para según que aplicaciones.

Por otro lado, se está trabajando en la relación entre la HAL y el kernel. Lo que ha permitido solucionar problemas con las peticiones de escritura en los registros de memoria, y un uso correcto de las interrupciones generadas cuando se pulsa un botón del controlador de volumen integrado en unos cascos.

28. Muchas gracias, y si tienen alguna pregunta estaría encantado de responderles.