Sistema de reproducción de archivos mp3

César Ávila Sánchez
Escuela de Ingeniería Electrónica
TEC
Cartago, Costa Rica
cesaravila22@gmail.com

Fabricio Quirós Corella
Escuela de Ingeniería Electrónica
TEC
Cartago, Costa Rica
fabisqc0207@gmail.com

***Resumen*—El presente documento pretende abarcar todo el proceso de desarrollo ejecutado y los resultados obtenidos por el grupo de trabajo, con el objeto de implementar un sistema capaz de reproducir archivos de audio, con extensión .mp3, tanto de manera local en un ambiente servidor, como en uno remoto o cliente.**

Palabras Claves—BeagleBoard-xM, GStreamer, Pipeline, Qt Creator.

I. INTRODUCCIÓN

Como el título de este texto bien lo menciona, asimismo se aclara en el inciso de Resumen respectivo, el objetivo de todo este proyecto básicamente es crear una aplicación que permita la reproducción de audio del tipo mp3, lo cual solamente es posible con un programa que sea capaz de manipular todo los recursos multimedia que incluya una computadora o algún sistema embebido en específico, como lo es, dentro de este contexto, la BeagleBoard-xM.

Es por esta razón que, para efectos de este proyecto, se recurre a una herramienta de software en particular, un framework multimedia, el cual es de carácter libre, que soporta una gama importante de sistemas operativos, procesadores y compiladores, denominado GStreamer, cuya principal funcionalidad consiste en la creación de aplicaciones audiovisuales de todo tipo, tales como aquellas destinadas a la visualización de un video, inclusive reproducción remota a la codificación de cierto tipo de archivos de audio, hasta las que permite efectuar la mezcla de audio y video, por mencionar algunos ejemplos.

Entre las principales características que posee el GStreamer, caben destacar las siguientes: presenta una biblioteca de núcleo compacta que no sobrepasa los 500KB, con aproximadamente 65 000 líneas de código, así como un trasiego de datos extremadamente ligero, lo que implica un alto rendimiento y una acumulación de retardos reducida. De la misma manera, facilita una interfaz de programación de aplicaciones (en inglés, API: *Application Programming Interface*) bastante simple y estable, cuyo acceso es posible realizarlo desde una cantidad considerable de lenguajes, con el objeto de permitir el desarrollo de aplicaciones y complementos (*plugins*),

Otra particularidad importante de la herramienta en cuestión consiste en el hecho que maneja una arquitectura inteligente de complementos, los cuales son cargados dinámicamente y proveen recursos para trabajar con una enorme cantidad de elementos multimedia, además de ello, es posible aumentar las capacidades y la funcionalidad del GStreamer, gracias a la inclusión de nuevos complementos, donde estos están organizados en distintas categorías, tal y como se muestra a continuación.

Primero, se tiene el *gst-plugins-base*, que constituye el conjunto de complementos básico, totalmente soportado por el GStreamer. En segundo lugar, se encuentra el *gst-plugins-good* que incluye todos aquellos complementos bien soportados que emplean licencias libres. Luego, está el *gst-plugins-ugly* que considera todos aquellos plugins que posiblemente presenten inconvenientes en su libre distribución. Y por último, se tiene la categoría de *gst-plugins-bad*, la cual contiene aquel conglomerado de complementos que aún no ha superado rigurosas especificaciones de calidad, impuestas por los desarrolladores de GStreamer.

Como una de las características fundamentales que posee este framework, y que es de suma utilidad a la hora de implementar alguna tarea multimedia en particular, corresponde a la capacidad de la utilización de una ruta total conformada por una serie de elementos vinculados, los cuales son complementos del GStreamer, donde la denominada ruta constituye el *pipeline* de la función, donde existen infinidad de posibilidades de vinculaciones entre plugins; es decir, hay innumerables pipelines a los que se pueden recurrir y crear.

II. DESCRIPCIÓN DEL DISEÑO

En cuanto al diseño estipulado y desarrollado por el grupo de trabajo, el cual igualmente considera una serie de especificaciones determinadas en un instructivo de dicho proyecto, es posible establecer que el sistema de reproductor de mp3 consiste en dos bloques funcionales principales: una unidad que trabaja como un servidor y otra como un cliente, donde básicamente la primera es capaz de ejecutar una reproducción de un archivo de extensión .mp3, ya sea de manera local o remota al cliente respectivo, quien simplemente se encarga de ejecutar el archivo de audio proveniente del servidor.

Dicho bloque que trabaja como servidor consiste en una serie de elementos, como lo son la BeagleBoard-xM, un monitor con una entrada de protocolo DVI, un dispositivo de almacenamiento extraíble del tipo USB, algún medio que permita la audición de estos archivos, ya sea unos audífonos o unos parlantes, y una interfaz gráfica que permite una

interacción amigable con el usuario. De inmediato, se comenta la funcionalidad de cada uno de estos constituyentes: primero, el sistema empotrado BeagleBoard-xM, cuya principal labor es realizar, mediante el uso de los pipelines correspondientes con los complementos de GStreamer, la reproducción de alguno audio mp3 que se encuentra almacenado en una memoria USB y que está conectada al dispositivo embebido en cuestión, ya sea de manera local; es decir, en la misma BeagleBoard, o en el cliente, el cual sería una computadora personal (PC).

El monitor constituye un componente de alta relevancia, puesto es el que facilita la comunicación entre el usuario y el servidor, sin la necesidad de recurrir a un ambiente que no es familiar para todas las personas, y finalmente, es el que permitirá visualizar la selección un archivo dentro de la memoria, así como efectuar comandos, entre los que es posible mencionar el de "reproducir" o "detener", igualmente "iniciar la reproducción remota" o detenerla; de la misma manera, ingresar la respectiva dirección IP, con el objeto de poder establecer la comunicación vía Wi-Fi entre servidor y cliente, esto gracias a que ambos están conectados a la misma red.

No obstante, lo descrito con anterioridad no sería posible sin la utilización de una interfaz gráfica en ambas unidades. Tanto para el ambiente de visualización del servidor como para el del cliente, se consideró la herramienta de software conocida como Qt Creator, un ambiente de desarrollo integrado (en inglés, IDE) multiplataforma, el cual representa un programa sencillo de manipular y de emplear a la hora de desarrollar algún tipo de interfaz, donde es posible adaptar, mediante ligeras modificaciones, códigos implementados en el lenguaje de programación C, con el objeto de ejecutar alguna función en particular al oprimir un botón, por nombrar un ejemplo. El siguiente diagrama de bloques funcionales, el cual es sumamente, general pretende sintetizar toda la propuesta de diseño considerada por el grupo de trabajo, así como resumir la información detallada previamente.

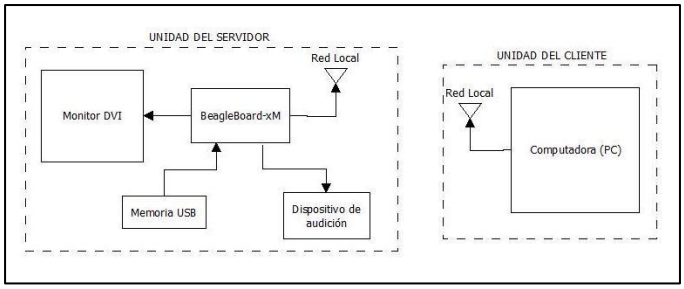

Figura 1. Diagrama de bloques general que resumen todos los constituyentes y la manera en que están dispuestos dentro del reproductor desarrollado.

III. PROCESO DE IMPLEMENTACIÓN

Para implementar nuestra aplicación inicialmente creamos un modelo de compilación cruzada. En este modelo establecimos que el sistema de archivos utilizado por la BeagleBoard-xM se encontraría en la computadora personal; de esta manera nos evitaríamos el problema de modificar continuamente la tarjeta SD. Para crear el sistema de archivos se ejecutó el script llamado setup.sh, el cual nos facilitó esta tarea.

A continuación, y como deseábamos reproducir formato mp3, debíamos asegurarnos que esto fuera posible en el BeagleBoard-xM. Instalamos para este caso gst-plugins-bad, asegurándonos de instalar también las dependencias de mad (mp3 decoder). Las cuales eran libmad y libid3tag.

Luego de encontrar el pipeline adecuado de Gstreamer para reproducir mp3, procedimos a implementarlo en C. Y a continuación buscamos el pipeline para enviar, vía UDP, un flujo de música. Haciendo uso del pipeline anterior y usando el mismo formato de implementación en C utilizado para la reproducción de mp3, logramos crear un programa capaz de enviar y otro capaz de recibir los datos de la canción seleccionada.

Sin embargo, para facilitar la interacción del usuario con el programa, se implementó una interfaz gráfica para ambos programas (tanto el servidor como el cliente). Estas interfaces se realizaron con Qtcreator.

El servidor, que se puede observar en la figura1, debía de contar con un buscador de archivos mp3 y un selector de IP al que se debía de enviar el flujo. Para tal fin se crearon botones que podrían seleccionarse a través de un mouse o una pantalla táctil que permitieran al usuario tanto seleccionar el archivo como digitar la IP del cliente.

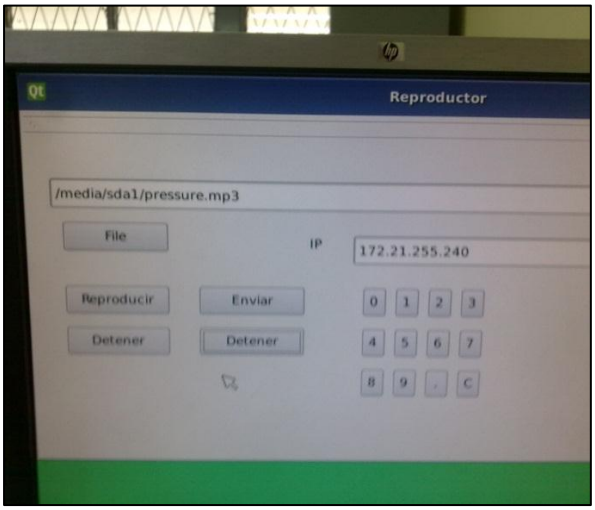

Figura 2. Fotografía de la interfaz gráfica empleada para el servidor, con sus respectivos comandos.

Por otro lado, para el cliente se creó una interfaz más simple que solo le permite indicar cuando desea recibir o cuando no los datos desde el servidor. Dicha interfaz se puede observar en la figura2.

Figura2. Interfaz gráfica empleada en la unidad del cliente, con sus dos posibles acciones a efectuar.

Por último, se realizaron los pasos necesarios para que nuestro programa servidor se mostrara al encender la BeagleBoard-xM. Esto fue posible evitando que el script que iniciaba matrix-gui (Interfaz gráfica por defecto de nuestro ambiente de desarrollo) se ejecutara. Básicamente se logró renombrando dicho script ubicado en /etc/rc5.d. A continuación, con ayuda del comando ln, se linkeo un archivo llamado S99reproductor a un script creado por el grupo de trabajo que ejecutaba el programa servidor.

Es importante mencionar que la ubicación de los scripts que se utilizan en la Beagle corresponde a la siguiente dirección /etc/init.d dentro del sistema de archivos. Para finalizar, el cliente solo se ubicaba dentro de la computadora para tal fin y se corría el programa creado.

IV. RESULTADOS OBTENIDOS

Al concluir nuestro proyecto se logró cumplir con las especificaciones y la ejecución de los comandos básicos: enviar, reproducir, detener, entre otros. Se cree que una aplicación de este tipo podría ser útil en un futuro, siempre y cuando se trabaje un poco más en los detalles (tales como manejo de errores e interfaz gráfica). Esto ya que la implementación es bastante sencilla y no fue depurada extensivamente.

Una de las limitantes encontradas en el programa fue que al ejecutar el pipeline, ya sea para enviar o para reproducir, el programa se quedaba ejecutando cierto proceso que al cerrar la aplicación se mantenía corriendo incluso si ya no se veía la interfaz gráfica.

Por otro lado, intentamos ejecutar nuestra aplicación en una pantalla HDMI y no lo logramos, solo fue posible probarla haciendo uso de un monitor con entrada DVI. En cuanto a esto, no estuvimos seguros de cuál fue la razón. Pero se cree tiene que ver de alguna manera con la resolución de la pantalla utilizada.

Para finalizar, en cuanto a la manera en que se oía la reproducción de la música, está fue bastante aceptable. Descubrimos que algunas veces al enviar vía UDP, se oía un tanto intermitente por cierto espacio de tiempo, pero no sucedía así siempre. Probablemente este detalle se deba a que existe algún problema de la red, ya que no se daba con frecuencia.

V. CONCLUSIONES

- El sistema empotrado BeagleBoard-xM contiene una enorme cantidad de recursos que es posible ser utilizados para el desarrollo de aplicaciones relacionadas con la interacción con el usuario.
- El GStreamer representa una herramienta sumamente poderosa al momento de trabajar con recursos multimedia, gracias al importante conjunto de plugins que soporta.
- La técnica de los pipelines es sumamente útil a la hora de recurrir a las diversas funcionalidades que incluyen los complementos que incluye el GStreamer.
- El programa Qt Creator facilita de manera considerable el diseño y la implementación de interfaces gráficas, por su versatilidad y capacidad de incluir funciones definidas por el propio desarrollador en cualquier lenguaje de programación.

REFERENCIAS

[1] Developpez.com, “Qt Creator User Interface”, http://qt.developpez.com/doc/qtcreator-2.3/creator-quick-tour/, recuperado el 22 de setiembre del 2013.

[2] Espacio Linux, “Super tutorial de introducción a GStreamer”, http://www.espaciolinux.com/2012/07/gstreamer-super-tutorial/, recuperado el 13 de setiembre del 2012.

[3] GStreamer Developers, “GStreamer Features”, http://gstreamer.freedesktop.org/features/index.html, recuperado el 16 de setiembre del 2013.

[4] S. Baker, R. Bultje, S. Kost, W. Taymans, A. Wingo, “GStreamer Application Development Manual”, Open Publication License, Setiembre 2013.

[5] Texas Instruments, “Linux EZ Software Development Kit (EZSDK) for Sitara™ ARM® Processors” http://processors.wiki.ti.com/index.php/Example_GStreamer_Pipelines, recuperado el 18 de agosto del 2013.

[6] Texas Instruments, “Examples GStreamer Pipelines”, http://processors.wiki.ti.com/index.php/Example_GStreamer_Pipelines, recuperado el 18 de setiembre del 2013.

[7] Victorhck in the free world, “Git y Github, tutorial básico de uso bajo GNU/Linux”, http://victorhckinthefreeworld.wordpress.com/2012/09/26/git-y-github-tutorial-basicode-uso-bajo-gnulinux/, recuperado el 21 de setiembre del 2013.