ELO 330: Programación de Sistemas
TAREA 4

18 de Enero de 2016

Richard Atuan Toledo 201121028-K
Felipe Fernández Pino 201121011-5

Descripción del programa

Los programas ***audioMix.java*** y ***audioTerminal.c*** consisten en dos procesos que permitirán que los usuarios tengan una comunicación de audio básica a través de la red

El programa ***audioMix*** consiste en un servidor TCP ejecutado sobre Java, el cual mezclará los flujos de audio proveniente de diferentes clientes. El diseño estará basado en hebras, las cuales le permitirán al servidor atender a los clientes sin mayor latencia, entregando la posibilidad de recibir nuevos clientes.

Mientras que el programa ***audioTerminal*** ejecutará un cliente TCP que se encontrará corriendo en el computador de cada usuario, tomando el flujo de audio proveniente desde el mircrófono con el proceso ***arecord***, y luego será transmitido al servidor.

El programa ***audioMix***, recibirá los paquetes de audio de todos los clientes y los mezclará, para luego enviarlos de vuelta a cada uno de los clientes conectados, con la particularidad de que la mezcla del audio resultante, no contendrá el audio del cliente destino.

Finalmente, una hebra independiente de cada cliente, recibirá el flujo desde el servidor y los reproducirá en la salida de audio del computador (Parlante) mediante el proceso ***aplay***.

Ambos procesos, tanto cliente como servidor terminarán con un comando ***Control+C***.

Estrategia de solución

I. Ejecución de otros procesos:

Para poder ejecutar otro proceso a partir del programa principal se utilizan las funciones `fork()` y `exec()`. Primero se crean dos procesos hijo a partir del proceso principal (padre) con la función `fork()`, luego, en el proceso hijo, se realizan los ajustes necesarios para la comunicación entre el proceso padre y el proceso hijo. Finalmente en uno de los hijos se ejecuta el programa *arecord* mientras que en el otro hijo se ejecuta *aplay* con el uso de la función `exec()`.

II. Comunicación entre procesos:

Tanto ***audioMix*** y ***audioTerminal*** deberán establecer de alguna forma una conexión para poder transmitir los paquetes de audio. La forma de implementación es a través de sockets con comunicación TCP dada la exigencia del problema. Los datos que serán transmitidos serán valores de 16 bits, en paquetes de 320 datos.

Para poder establecer la comunicación entre el servidor y los clientes, es necesario que los clientes especifiquen una IP del computador donde corre el servidor, y un puerto donde estará escuchando.

Estos datos serán entregados como parámetros en la ejecución del programa.

III. Sincronización entre procesos:

Estos procesos no están sincronizados con uso de *semáforos*, pero la transmisión de paquetes desde el servidor hacia los clientes, no se ejecutará hasta que todos los clientes hayan enviado sus paquetes de audio correspondientes.

Los clientes no serán capaces de enviar más de un paquete de audio, sin haber recibido la respuesta desde el servidor antes.

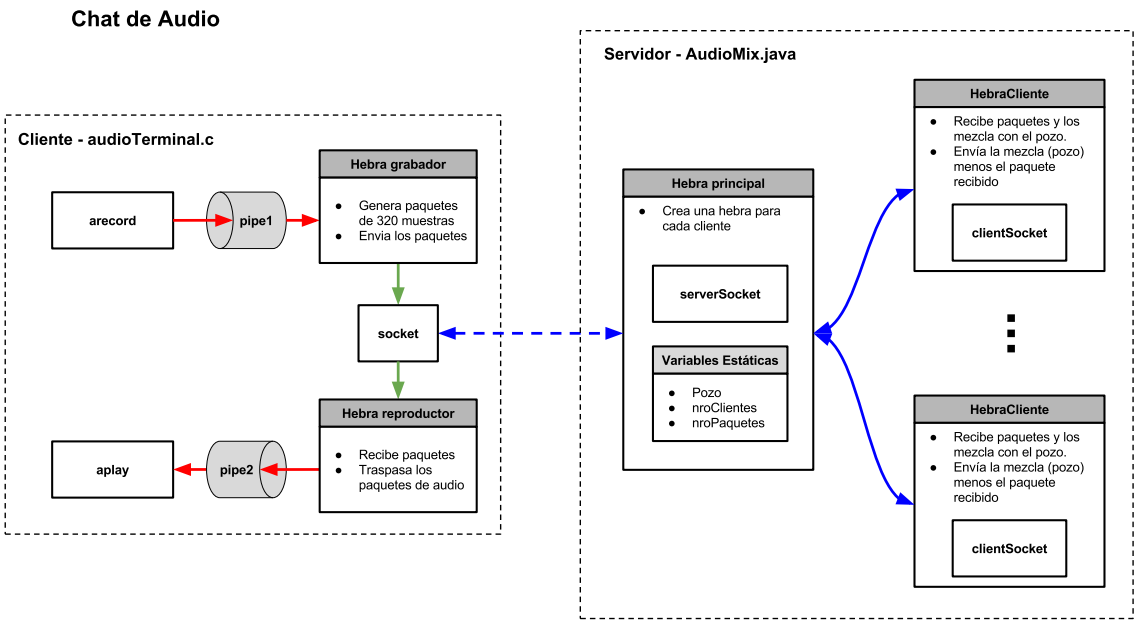

Figura 1: Comunicación entre procesos.

Resultados

Para probar los resultados obtenidos, ejecutamos el servidor en java mediante el comando

```
>java audioMix <Puerto>
```

El servidor quedará esperando hasta que más de un cliente se conecte. Luego se ejecutará en otra terminal comando

```
>./audioTerminal <IP><Puerto>
```

que serán los clientes que se estarán conectando al servidor, para conectarse con más clientes, basta ejecutar el mismo comando en otra terminal.
Luego de realizar la ejecución de los programas, basta con empezar a grabar desde cada cliente y esperar que los demás clientes envíen su audio al servidor, para que luego el servidor nos transmita de vuelta el audio mezclado logrando reproducir en nuestro computador la información capturada.
Tanto el servidor como el cliente terminarán su ejecución con el comando ***Ctrl+C***.

Una vez terminada la aplicación, se procederá a finalizar los procesos *arecord* y *aplay* en cada cliente.