Transmisión de multimedia en Internet usando proyecto FFmpeg.

Mario A. Ungemach M., Sebastián U. Duque R., Redes de Computadores II, *Departamento de Electrónica, Universidad Técnica Federico Santa María.*,

***Abstract*— El presente trabajo trata de la transmisión de multimedia a través de Internet haciendo uso del proyecto FFmpeg, dentro del desarrollo de una herramienta de educación a distancia a través de la red.**

Esta entrega, explica los aspectos relativos a la implementación de la solución, describiendo los principales elementos usados. Además se presenta el tipo de arquitectura utilizada en la implementación, sus limitaciones y eventuales mejoras.

***Index Terms*— captura pantalla, exposiciones, FFmpeg, Internet, software libre, streaming, Unicast, Java.**

I. Introduccion

Este documento presenta la segunda entrega del trabajo desarrollado en el marco de la asignatura Redes de Computadores II del Departamento de Electrónica de la Universidad Técnica Federico Santa María. El que trata de la transmisión de multimedia a través de Internet haciendo uso del proyecto FFMpeg [1]. Se describen principalmente los elementos usados, como la aplicación para conversión de formatos multimedia *ffmpeg*, el servidor de streaming *ffserver*, una aplicación para el control de flujo multimedia escrita en lenguaje *Java*, además de otros elementos que permiten llevar a cabo la solución, tales como el servidor Web Apache y el servidor de dominios DynDns. Se continúa con la descripción de funcionamiento del sistema, aspectos relativos a los requerimientos de BW y por último se exponen las conclusiones generales.

II. Escenario y Objetivos de la aplicación.

Es muy común hoy en día el uso de la Internet para la realización de exposiciones, donde quien presenta se encuentra muy alejado físicamente de su audiencia. En este sentido, es muy usual observar el uso de aplicaciones de tipo vídeo conferencia para realizar este tipo de actividades, las cuales permiten un *feedback* entre quienes presencian la exposición y el expositor mismo.
Esta información viaja a través de la red, grandes distancias, es capturada por la aplicación cliente dentro de un terminal y es luego proyectada para ser observada y oída por los asistentes en un lugar definido.
No obstante, muchas de estas aplicaciones requieren del uso de elementos físicos agregados que permitan el traspaso de esta información, como por ejemplo, el uso cámaras digitales para la captura de imágenes del material de presentación y del expositor mismo y dispositivos de captura de voz como micrófonos.
Incluso en muchas ocasiones, la figura del mismo expositor no es necesaria, solo lo es el material de presentación, su voz y un elemento que sirva para indicar un lugar en la imagen del material del cual se está exponiendo.
Acerca del material que está siendo presentado, este siempre se encontrará en el *top* de la pantalla del computador donde está corriendo la presentación y el elemento indicador puede ser el cursor de la computadora manejado por el mouse.
Acerca del *feedback* entre expositor-audiencia este depende del tipo de presentación y puede ser suplida por otras aplicaciones de comunicación por voz a través de Internet.

A partir de lo anteriormente expuesto, se ve la necesidad de ofrecer una herramienta que permita a uno o varios usuarios (clientes) acceder a las imágenes que están siendo proyectadas en la pantalla de un servidor, junto con el audio proveniente de la entrada de micrófono de este. De esta manera, uno o varios clientes puedan presenciar, sin mayores requerimientos, presentaciones donde quien expone se comunica a través de la pantalla de su computador y de lo que habla a través de su micrófono.
El esquema de funcionamiento de la aplicación se muestra a continuación.

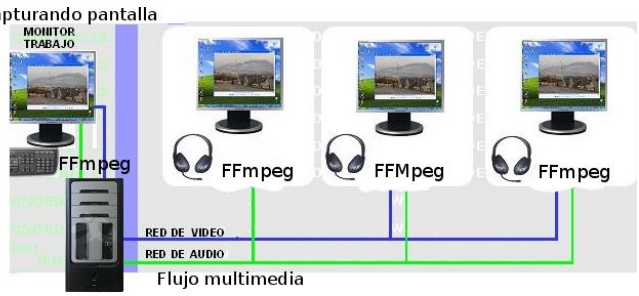

Fig. 1. Esquema de la aplicación del proyecto Ffmpeg. Comunicación de contenidos a distancia.

III. Esquema de Comunicación

El sistema propuesto está basado en un tipo de comunicación Unicast, donde existe una conexión directa entre emisor y receptor, lo que hace que para cada receptor se debe crear una nueva conexión.

A continuación se presenta un esquema explicativo.

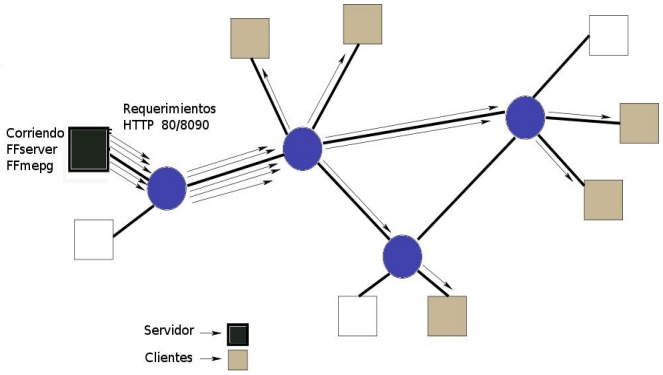

Fig. 2. Esquema de la comunicación de contenidos a distancia. Modo Unicast.

En este esquema, cada vez que un cliente realice una petición al servidor, este creará una nueva conexión para suministrar la información.

IV. Detalle de elementos necesarios.

Por parte del servidor los elementos necesitados para esta solución son:

- Servidor Web (Apache2) + Página alojada (Flash embebido).
- Un dominio público (DynDns, p.e.)
- Herramientas libres para streaming: Proyecto FFmpeg.
- Herramienta para control de servidor de streaming y alimentador de información (Tópicos del curso de Programación de Sistemas: Java-Runtime-Process más adelante).

Por parte del cliente los requerimientos impuestos son:

- Navegador Web + soporte Flash.

La solución presentada como se puede observar coloca al servidor toda la carga de funcionamiento, ya que es donde corren todas las aplicaciones: El servidor de streaming(ffserver), el alimentador (ffmpeg) que entrega la información de la pantalla del computador junto con el audio proveniente de la entrada de micrófono empaquetados en formato *swf*, la herramienta de control de flujo multimedia explicada más adelante y la página alojada en el servidor Web Apache.

Por parte del cliente la solución propuesta minimiza lo más posible sus requerimientos, de manera de que sea una aplicación *plug and play*, de fácil acceso para estos.

Antes de realizar la instalación manual del paquete de herramientas que contiene a *ffpmpeg, ffserver y ffplay* que componen el Proyecto Ffmpeg, es necesario haber instalado el soporte para que se puedan utilizar todas sus funcionalidades, esto es, se deben haber instalado una serie de bibliotecas. Una lista completa de estas es:

dpkg-dev, libimlib2-dev, texi2html, liblame-dev, libfaad2-dev, libmp4v2-dev, libfaac-dev, libxvidcore4-dev, libtheora-dev, libgsm1-dev, libogg-dev, libvorbis-dev, liba52-dev, libdts-dev, libsdl1.2-dev, libraw1394-dev, libdc1394-13-dev y quilt
.

Al momento de compilar las fuentes antes de instalar es necesario para nuestros fines habilitar ciertos módulos, en linux esto se realiza de forma similar al siguiente comando:

```
$ ./configure --enable-gpl --enable-
libmp3lame --enable-x11grab
```

Donde los argumentos son los módulos que se habilitarán de ffmpeg, que serán necesarios para este proyecto.
De la misma forma si se necesitan habilitar otros módulos se deben agregar de forma similar.

Tal como se dijo en [2] Ffmpeg es un programa sin interfaz gráfica que permite convertir o transformar formatos multimedia, tanto de video como de audio. Aunque existen otros programas, algunos sin necesidad de usar comandos, es una de las opciones con más posibilidades y con una *performance* muy rápida.

La sintaxis en línea de comandos es la siguiente:

```
$ffmpeg [[infile options][`-i' infile]]...
{[outfile options] outfile}...
```

Un ejemplo de uso es el siguiente:

```
$ffmpeg -i test1.mpg -vcodec mpeg4 -s
320x240 -b 300k -r 10 -acodec mp3 -ar
22050 -ab 64k -f avi test1.avi
```

En donde
vcodec :Especifica el codec de video para compresión usado.
s :Especifica el size de compresión del vídeo de entrada.
b :Especifica el bitrate de vídeo de compresión.
r :Especifica la taza de cuadros por segundo de captura.

acodec :Especifica el codec de audio usado.
ar :Especifica la taza de audio usada.

ab :Especifica bitrate de audio usado.
f :Fuerza al formato de salida.

Por otra parte tal como se menciona en [2] FFserver es un componente anexo que permite servir flujos de vídeo y audio a través de HTTP/RTP/RTSP. Soporta archivos multimedia almacenados o que están siendo recibidos en tiempo real.
Para su uso es necesario conocer acerca de su archivo de configuración, el cual por defecto es llamado *ffserver.conf*, si bien puede ser llamado posteriormente de cualquier manera dándoselo a ffserver como argumento de entrada con la opción -f.

Un segmento importante de de este archivo especifica el puerto usado por ffserver para realizar el streaming, la dirección de *bind*, la cual es necesaria si tenemos más de una interfaz de red, la máxima cantidad de clientes conectados, el ancho de banda máximo utilizado entre todos los clientes y por último si este funcionará como demonio o será iniciado manualmente.

Un ejemplo de esto es el siguiente:

```
Port 8090
BindAddress 0.0.0.0
MaxClients 1000
MaxBandwidth 1000
NoDaemon
```

El resto del archivo de configuración contiene dos importantes secciones:

<Feed> : Cada Feed contiene una secuencia de Video y/o Audio proveniente de la salida de alguna instancia de ffmpeg. Es una sección del archivo de configuración.

Un ejemplo de Feed es el siguiente.

```
<Feed feed1.ffm >

File /tmp/feed1.ffm

FileMaxSize 200K

ACL allow 127.0.0.1

<Feed >
```

<Stream> : Aquí se definen los parámetros de reproducción de los streams provenientes de los archivos previamente codificados por Ffmpeg.

Un ejemplo de Stream podría ser,

```
# ASF compatible
<Stream test.asf >
Feed feed1.ffm
Format asf
VideoFrameRate 15
VideoSize 352x240
VideoBitRate 256
VideoBufferSize 40
AudioBitRate 64
<Stream >
```

Por cada Feef pueden existir varios Streams que determinan varios archivos de salida con distintos formatos o de diferentes características.
Para realizar el streaming de un archivo guardado en disco se ejecutan los comandos:

```
$./ffserver -f doc/ffserver.conf &
$./ffmpeg -i INPUTFILE
http://localhost:8090/feed1.ffm
```

Notar que en este caso el archivo de configuración de ffserver se le da como argumento mediante la opción -f y puede estar alojado en cualquier directorio en disco.

Para poder visualizar el contenido que está siendo generado, un cliente solo debe colocar en un navegador o mediante alguna aplicación apropiada la dirección de este host, el puerto asociado al streaming y el archivo generado.
Siguiendo el ejemplo del archivo de configuración de ffserver se debiera colocar en el navegador o aplicación:

```
http://<dirección del host>:8090/test.asf
```

Para controlar el inicio y término de la transmisión de contenido multimedia, para los fines de este proyecto se generó una herramienta de control de flujo de audio/vídeo. Mediante esta Api se puede controlar el streaming de manera eficiente. En caso de necesitar cambiar los parámetros del flujo para modificar las características del vídeo a enviar hacia los clientes.
Se muestra a continuación el aspecto de esta aplicación gráfica.

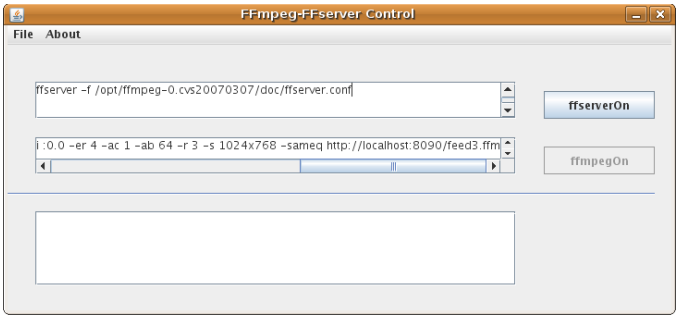

Fig. 3. Aplicación gráfica de control de streaming.

Esta aplicación hace uso de la Clase Runtime de Java, con la cual se ejecutan los programas ffserver y ffmpeg los cuales están originalmente escritos en lenguaje c, compilados e instalados en Linux, tal como se dijo anteriormente.
De manera de controlar mediante dos botones, uno para el control de ffserver y otro para ffmpeg, el estado de la transmisión.
El diagrama de la aplicación, luego de comenzar el streaming es el siguiente.

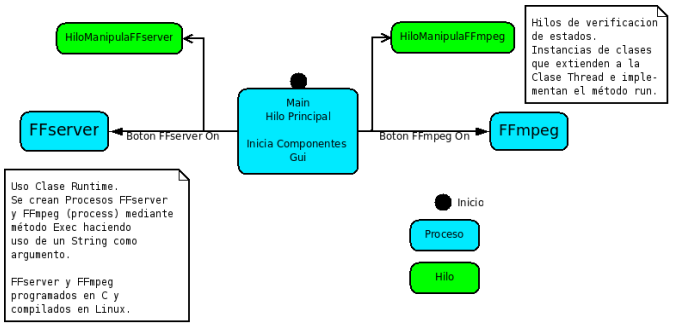

Fig. 4. Diagrama de la aplicación, estado luego de comenzar el streaming.

El diagrama de estados general se muestra a continuación:

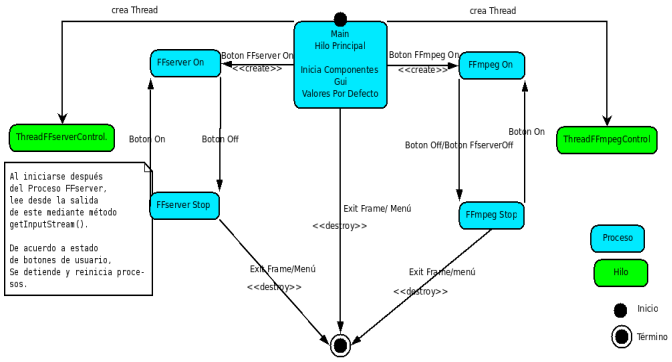

Fig. 5. Diagrama de estados de la aplicación de control.

V. Funcionamiento y resultados.

Una vez todos los elementos estén correctamente configurados e instalados, el primero de estos en iniciarse será la aplicación que comandará al servidor de streaming. De esta manera se debe solo presionar los botones de encendido de ffserver y ffmpeg. Así estará todo preparado para que los clientes puedan acceder al flujo de audio y vídeo.

Cabe recalcar que este contenido específicamente contendrá la información existente en la pantalla del servidor y del audio proveniente de la entrada de señal de micrófono, empaquetados ambos en formato *swf*.
El cliente, quien tiene el único requisito de contar con un navegador con soporte Flash, podrá acceder a esta información sólo colocando en la barra de navegación la dirección de la página (nombre de dominio público del host) la que debe ser conocida, o estando dentro de una LAN alternativamente mediante la IP del servidor.

Por el lado del cliente en su pantalla se observaría algo como lo mostrado a continuación:

Fig. 6. Flujo de vídeo obtenido mediante el navegador cliente.

Los parámetros usados para la compresión fueron tales que permitían obtener del lado del cliente, una imagen satisfactoria para por ejemplo la visualización de documentos con una nitidez adecuada.
No obstante también se debe tener en cuenta que las dimensiones y características del flujo, tales como el *bitrate* de vídeo/audio, el *framerate* de vídeo están muy relacionadas con los anchos de banda necesarios para satisfacer la necesidad de múltiples clientes conectados.

Para tener una medida de la relación existente entre este flujo definido mediante parámetros de compresión dados a ffmpeg, se realizaron pruebas para medir el ancho de banda de subida usado por parte del servidor, con distintos números de clientes conectados. Lo que se muestra a continuación en la gráfica.

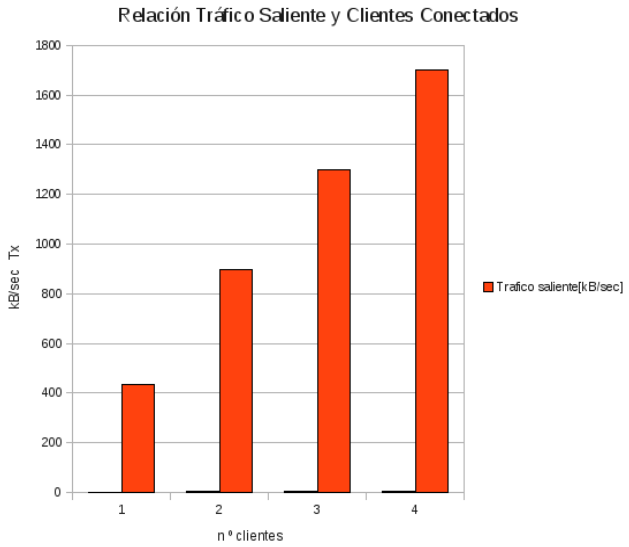

Fig. 7. Relación de BW de subida del servidor V/S Nº de clientes conectados.

Se vislumbra entonces la necesidad de implementar un esquema tipo Multicast para aplicaciones donde el número de clientes suba considerablemente en ciertas situaciones.

IV. Conclusiones

Como se hizo notar anteriormente, se vislumbra la necesidad de usar un tipo de esquema Multicast (multipunto) para proveer un servicio donde el número de clientes se incremente considerablemente, ya que esto no sería soportado eventualmente por parte del servidor, sabiendo que el ancho de banda de subida proveído por las empresas suministradoras de Internet, es generalmente mucho menor que el de bajada.

No obstante, esta implementación en modo Unicast, puede ser útil en muchos casos y es suficiente para proveer una herramienta para realizar exposiciones a través de Internet donde por ejemplo:

- Un expositor está muy lejos (otra región del territorio u otro país) de un grupo de personas que concurren a presenciar la exposición juntas a un lugar y que obtienen el flujo a través de un host del cual pueden proyectar la imagen.

- Un anfitrión desea mostrar a un grupo reducido personas, un trabajo que está realizando, y no están próximos entre ellos.

Se debe hacer notar que esta solución es una de las tantas posibles que pueden existir, pero que tiene como característica principal, la unión de herramientas y aplicaciones libres para su implementación, lo que hace que sea un desarrollo potencialmente depurable, en cualquiera de sus elementos, desde su interfaz de presentación hasta su interfaz de control de flujo.

References

[1] FFMpeg Proyect [Online]. Disponible en : http://ffmpeg.mplayerhq.hu/
[2] Informe Nº1. Transmisión de multimedia en Internet haciendo uso del proyecto Ffmpeg. Redes de Computadores II segundo semestre 2008.