Diseño e implementación de una aplicación distribuida de video bajo demanda basada en la arquitectura cliente-servidor

Omar P. Montalvo, Byron P. Vicente,
Ingeniería Eléctrica y Electrónica, Escuela Politécnica Nacional, Quito - Ecuador

***Abstract*- El presente proyecto abarca lo concerniente a la entrega del servicio de video bajo demanda mediante una aplicación que usa los protocolos estandarizados por la IETF (*Internet Engineering Task Force*): RTP (*Real-Time Transport Protocol*), RTCP (*Real Time Control Protocol*) y RTSP (*Real Time Streaming Protocol*), basado en la arquitectura cliente-servidor usando software libre y software propietario.**

La aplicación fue desarrollada siguiendo las etapas de definición y análisis de requerimientos, diseño, implementación y pruebas.

Se utilizaron técnicas de diseño y desarrollo de software ágiles convenientes para este tipo de proyecto como programación extrema.

I. INTRODUCCIÓN

En la actualidad los sistemas de distribución de contenido multimedia bajo demanda son un servicio esencial en especial en lugares de hospedaje en los cuales a los clientes se ofrece la oportunidad de acceder a diferentes tipos de contenidos multimedia mediante su televisor como por ejemplo a determinada película o pista musical; el presente proyecto pretende solventar dicha necesidad mediante la creación de un sistema que permita a uno o más usuarios acceder al contenido que deseen.

II. REDES DE VIDEO BAJO DEMANDA

Las redes actuales han evolucionado de modo que soportan los requerimientos que la transmisión de video necesita. La forma de crecimiento que ha tenido Internet crea la atractiva posibilidad de usar la misma plataforma para entregar nuevos servicios. Las nuevas tecnologías que las redes LAN (*Local Area Network*) y WAN (*Wide Area Network*) brindan, permiten que los usuarios accedan a servicios que Internet ofrece sin que el usuario tenga que invertir en nuevos elementos de hardware de red.

A. VoD (Video On Demand)

Es la denominación más común para sistemas de distribución por suscripción de video utilizando suscripciones de banda ancha sobre el protocolo IP.

El servicio de video bajo demanda se lo puede definir también como una aplicación que espera, procesa y sirve peticiones de uno o varios clientes, donde dichas peticiones solicitan un video que el cliente que emitió la petición desea recibir, una vez que el servidor ha recibido la petición del cliente este responderá con la emisión del contenido; estos datos deberán ser almacenados de forma permanente en el servidor y de modo temporal en el cliente.

Fig. 1 Diagrama básico de un sistema de VoD

En la Fig. 1 se muestra un esquema general de cómo debería estar estructurado el servicio de video bajo demanda.

III. PROTOCOLOS PARA STREAMING

La necesidad de transmitir información multimedia a través de las redes tanto públicas como privadas han motivado la implementación de nuevos protocolos que puedan permitir que las aplicaciones multimedia sean capaces de utilizar las redes para distribuir su contenido.

La IETF ha desarrollado una serie de protocolos que posibilitan la transmisión de contenido multimedia en tiempo real a través de redes de conmutación de paquetes basadas en TCP/IP. Los tres principales protocolos son: RTP *(Real-Time Transport Protocol*), RTCP (*Real Time Control Protocol*) y RTSP (*Real Time Streaming Protocol*).

A. RTP (Real-Time Transport Protocol)

Surgió como idea de crear un protocolo específico para la demanda de aplicaciones en tiempo real (música, video, telefonía IP, videoconferencia, multimedia). El protocolo RTP fue definido en los RFC 1889,1890 y 2250 durante los años 90, sin embargo en julio de 2003 se publicó el RFC 3550 que hizo obsoletos a los RFC's 1889 y 1890. RTP fue pensado para brindar servicios de video bajo demanda, control de maquinaria, telemedicina, video conferencia, monitoreo y administración de paquetes y servicios interactivos para redes de conmutación de paquetes basadas en la arquitectura TCP/IP.

RTP fue pensado para brindar servicios de transporte en tiempo real basándose en 4 escenarios:

1) *Conferencia de audio multicast:* En este escenario RTP se vale de direcciones IP multicast y de puertos UDP, donde se utiliza un puerto para datos y otro para control.
2) *Conferencia de audio y video:* Considerando el caso de que ya no se transmita solo audio sino video también como es el caso claro de una video conferencia, los componentes de audio y video serán enviados de forma separada, usando un puerto para el audio y otro para el video.
3) *Mixer y translators:* Cuando un usuario no tiene el mismo ancho de banda que los demás o enlaces de alta velocidad, las señales de audio deben volver a ser sincronizadas para esto se utilizan los *mixers.* En lugares donde los enlaces son de alta velocidad los paquetes no sufren retardos, pero pueden verse retrasados debido al direccionamiento IP del tipo *multicast* o *firewalls*, en este caso se utiliza *translators*.
4) *Codificación de capas:* Las aplicaciones multimedia deben ser capaces de adaptar la velocidad de transmisión a los diferentes receptores o ajustarlas a la congestión de red.

B. RTCP (Real Time Control Protocol)

RTCP es un protocolo basado en el intercambio periódico de paquetes de control del protocolo RTP, el cual se realiza entre los participantes de una sesión. El objetivo principal de RTCP es intercambiar información relacionada con los paquetes RTP y su contenido entre los participantes de estas sesiones.

En la Fig. 2 se observa una sesión RTP entre el cliente y el servidor; se observa que el servidor envía paquetes RTP al cliente y este, a su vez envía paquetes RTCP al servidor con información asociada al servicio.

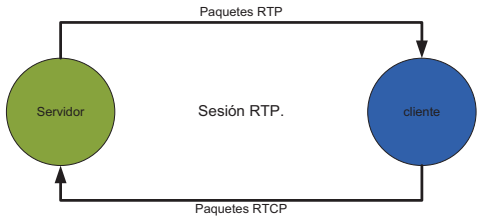

Fig. 2 Intercambio de paquetes RTP y RTCP

Se definen 5 tipos de paquetes RTCP:

1) *SR (Sender Report):* Son paquetes que proporcionan información estadística acerca de cuantos y cómo se transmiten los paquetes RTP.
2) *RR (Receiver Report):* Son paquetes de control generados por los clientes de una sesión RTP, es decir los equipos que no están generando tráfico RTP
3) *SDES (Source Description):* Se emite este paquete con información específica del cliente RTP que lo genera.
4) *BYE (Despedida):* Este paquete es emitido tanto por emisores como por receptores cuando uno de ellos decide abandonar una sesión.
5) *APP (Application-Defined):* Este paquete se utiliza para proporcionar información sobre las funciones de las aplicaciones específicas y nuevas que se están desarrollando.

C. RTSP (Real Time Streaming Protocol)

RTSP es un protocolo de nivel aplicación utilizado para el control de datos que se entregan en tiempo real. RTSP provee un marco de trabajo para aplicaciones de audio y video en tiempo real. La función principal de RTSP es gestionar múltiples sesiones y encontrar las mejores condiciones para los canales (UDP, TCP, unicast, multicast, etc.) y proporcionar un medio para la entrega de los mecanismos basados en RTP. RTSP actúa como un control remoto en la sesión sobre los servidores multimedia.

RTSP soporta las siguientes operaciones para streaming multimedia:

1) *Recuperación desde un servidor multimedia:* El contenido debe ser capaz de ser controlado por RTSP de forma individual es decir, RTSP debe manejar el contenido por cada uno de los *streams* de forma individual.
2) *Invitación de un servidor multimedia a una conferencia:* Un servidor puede ser "invitado" a unirse a una conferencia ya existente, ya sea para reproducir contenido multimedia o para grabar una parte de este contenido.
3) *Adición de medios a una presentación existente:* Especialmente para presentaciones en vivo, es útil para informarle al cliente que existe nuevo contenido multimedia.

RTSP controla un stream que puede ser enviado por un protocolo independiente del canal de control. Por ejemplo,

RTSP puede utilizar TCP como protocolo de trasporte mientras los otros protocolos de tiempo real (RTP y RTCP) pueden utilizar UDP para realizar sus funciones.

Existen varios métodos dentro de RTSP que permiten que el cliente se comunique con su servidor, estos métodos pueden o no influir en el estado del contenido y son los que permiten realizar muchas de las operaciones necesarias en una comunicación de tiempo real entre cliente y servidor, estos métodos son:

- *Setup*
- *Play*
- *Record*
- *Pause*
- *Teardown*
- *Describe*
- *Options*
- *Announce*
- *Get_Parameters*
- *Set_Parameters*
- *Redirect*

D. SDP (Session Description Protocol)

SDP proporciona la representación estándar que indica cómo esta información se transporta través de la red; de hecho SDP es el formato para la descripción de sesiones multimedia; utiliza como protocolo de transporte RTSP entre otros. SDP está destinado a ser de uso general a fin de que pueda ser utilizado en una amplia gama de entornos de red y aplicaciones.

1) Inicio de Sesiones SDP: SDP proporciona el soporte para protocolos de capa aplicación que controlan, crean, modifican y terminan sesiones multimedia.

2) Streaming Multimedia: Cuando un cliente RTSP y su respectivo servidor negocian un conjunto de parámetros relacionados para la entrega de los medios, mediante la sintaxis de SDP se describen dichos parámetros.

3) Sesiones Multicast: SDP permite la comunicación de los diferentes participantes mediante un directorio distribuido de inicio de sesión; por ejemplo, SDP puede enviar periódicamente paquetes que contienen una descripción del grupo multicast y de la sesión.

IV. Herramientas Para el Desarrollo de Aplicaciones Multimedia

A. Gstreamer

Gstreamer es un *framework* de código abierto que permite el desarrollo de aplicaciones multimedia. Gstreamer no solo se dedica a manejar flujos de datos multimedia, sino que puede manejar cualquier tipo de datos y reduce la sobrecarga de datos mediante el diseño de tuberías, mejorando así el rendimiento de las redes y reduciendo el procesamiento de las aplicaciones.

Una tubería es un conjunto de elementos interconectados entre sí mediante propiedades que contienen dichos elementos, cuya finalidad es modificar la información que ingresa a la tubería para obtener un resultado deseado ver Fig. 3.

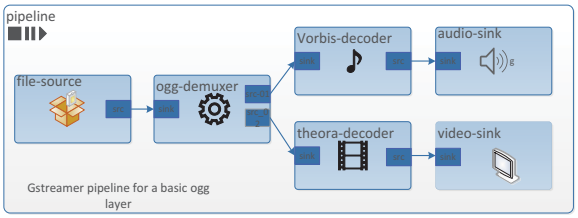

Fig. 3 Tubería Gstreamer

El *framework* Gstreamer se encuentra dividido en los siguientes paquetes:

- **gstreamer:** El paquete núcleo.
- **gst-plugins-base:** Un conjunto de elementos esenciales.
- **gst-plugins-good:** Un conjunto de plugins de buena calidad bajo LGPL (*GNU Lesser General Public License*).
- **gst-plugins-ugly:** Un conjunto de plugins de buena calidad que pueden poseer problemas de distribución.
- **gst-plugins-bad:** Un conjunto de plugins que necesitan mejorar su calidad.
- **gst-python:** Enlazadores para el lenguaje de programación Python.

B. NetBeans [1]

NetBeans IDE es un entorno de desarrollo para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

En el presente proyecto se emplea esta herramienta para compilar la librería Gstreamer RTSP Server la cual cuenta con las siguientes clases:

1) GstRTSPClient: Maneja los eventos que realiza cada cliente, sus peticiones realizadas.

2) GstRTSPMediaFactory: Se encarga de crear las tuberías por cada uno de los contenidos multimedia.

3) GstRTSPMediaFactoryURI: Se encarga de asociar un MediaFactory con el URI solicitado.

4) GstRTSPMediaMapping: Se encarga de asociar una sesión con un recurso multimedia solicitado, el cual es generado mediante la clase *GstRTSPMediaFactory*.

5) GstRTSPMedia: Se encarga de contener los elementos que intervienen en la preparación de los recursos multimedia para su correspondiente entrega.

6) *GstRTSPAuth:* Se encarga de crear un sistema de autenticación básica para que solo los usuarios autorizados puedan realizar conexiones con el servidor.
7) *rtsp-params:* Se encarga de generar los parámetros involucrados en el establecimiento de las sesiones mediante el protocolo RTSP.
8) *rtsp-sdp:* Se encarga de generar los datos descriptivos propios del protocolo SDP, para que el cliente reciba la información respectiva del contenido que le será entregado.
9) *GstRTSPServer:* Se encarga de realizar la tarea de escuchar y aceptar las peticiones de los clientes creando una lista de clientes asignándoles un pool de conexiones y por ende una sesión, permitiendo la comunicación asincrónica con el cliente.
10) *GstRTSPSessionPool:* Mantiene una lista de sesiones. Esta lista de sesiones puede ser manejada por el servidor para realizar un seguimiento de las sesiones activas.
11) *GstRTSPSession:* una sesión por cada una de las peticiones realizadas por los clientes, que se conectan al servidor, asignando un número de identificación.

V. Aspectos Para el Desarrollo de Aplicaciones Web

A. WWW (World Wide Web)

El servicio *World Wide Web*, también conocido como WWW o simplemente Web, es un sistema de información distribuido por Internet basado en hipertexto que proporciona una interfaz común entre los distintos tipos de datos y los servicios de Internet existentes.

B. HTML (Hyper Text Markup Language)

El lenguaje estandarizado para la creación de páginas Web; es el lenguaje de marcas de hipertexto HTML; este es un lenguaje muy sencillo que permite escribir documentos de hipertexto basado 4 preceptos básicos:

- HTML es simplemente texto
- Igualdad de mayúsculas y minúsculas
- No importan los tabuladores y los saltos de línea
- Maneja caracteres especiales

C. Principios de Diseño de Páginas Web

Se dice que una página Web es accesible si se han tenido en cuenta los requisitos para que pueda ser usada por personas con discapacidades físicas o por usuarios que poseen diversas configuraciones de hardware o software.

W3C (*World Wide Web Consortium)* define 14 pautas para el diseño de páginas Web las cuales pueden o no estar asociadas entre sí o estar relacionadas a otros puntos de diseño. Estas pautas son:

a) Proporcione alternativas equivalentes de acceso
b) No basar la página Web solo en el color
c) Utilizar marcadores y hojas de estilo
d) Identificar el idioma utilizado
e) Crear tablas adecuadamente
f) Independencia de navegador
g) Facilidad de manejo para el usuario
h) Asegurar la accesibilidad
i) Independencia de dispositivo
j) Utilizar soluciones provisionales
k) Utilizar tecnología y pautas
l) Proporcionar la información necesaria
m) Proporcionar mecanismos claros de navegación
n) Asegurarse que el contenido sea claro y sencillo

D. ASP.NET Framework

ASP.NET es una plataforma de programación Web que proporciona todos los servicios necesarios para desarrollar aplicaciones Web.

Desarrollar páginas Web que emplean el *Framework.NET* brinda las siguientes ventajas:

- Implementación de Interfaces Web complejas
- Separación entre cliente y servidor
- Ejecución sin estado
- Facilidad de acceso a datos
- Facilidades de escalabilidad

VI. Aplicaciones Distribuidas Multimedia Basadas en la Arquitectura Cliente Servidor

A. Criterios básicos de aplicaciones distribuidas basadas en la arquitectura cliente servidor

Los grandes sistemas informáticos actuales dividen el procesamiento de la información en varios procesos, de modo que se pueden optimizar los recursos sin que estos sean centralizados, evitando que se resten características esenciales a las aplicaciones de gran envergadura. Los sistemas distribuidos además pueden agregar componentes sin causar demasiados problemas.

Existen varias ventajas al momento de distribuir una aplicación respecto a sistemas centralizados entre las cuales se enumeran las siguientes:

- Compartición de recursos
- Apertura
- Concurrencia
- Escalabilidad
- Tolerancia a fallos

De igual forma como existen ventajas también hay varios retos el momento de diseñar un sistema distribuido:

- Complejidad
- Seguridad
- Manejabilidad
- Imprevisibilidad

B. *Arquitectura Cliente Servidor*

La arquitectura cliente - servidor se puede definir como un conjunto de servicios asociados a un conjunto de clientes que acceden a esos servicios donde servidores son aquellos procesos que ofrecen servicios a otros subsistemas, como por ejemplo: servidores de ficheros que permiten el acceso a información y además gestionan dichos ficheros y clientes son aquellos procesos o subsistemas que acceden a los servicios ofrecidos por los servidores.

La aproximación más simple de la arquitectura cliente servidor está formado por dos niveles, de la cual se puede obtener dos subdivisiones:

1) Modelo de cliente ligero: En este modelo el procesamiento del cliente se reduce al mínimo, únicamente el cliente se encarga de la presentación de la información dejando al servidor la tarea de procesar la información; en la Fig. 4 se observa este modelo.

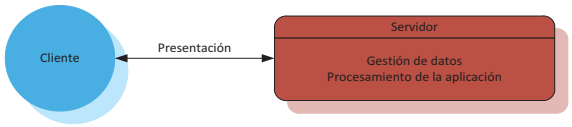

Fig. 4 Modelo de cliente ligero

2) Modelo de cliente pesado: En este modelo el procesamiento de la aplicación es responsabilidad del cliente, liberando de esta carga al servidor; como se observa en la Fig. 5.

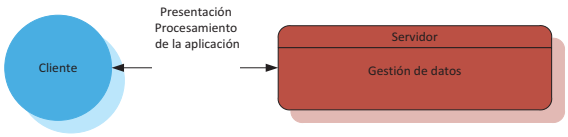

Fig. 5 Modelo de cliente pesado

C. *Aplicaciones distribuidas multimedia*

Los sistemas multimedia distribuidos son sistemas de tiempo real los cuales deben ejecutar tareas y entregar sus resultados de acuerdo con una planificación que se determina externamente.

Los requisitos de las aplicaciones multimedia difieren del resto de aplicaciones en tiempo real:

- Las aplicaciones multimedia son altamente distribuidas y operan sobre entornos de computación distribuida de propósito general. Compiten con otras aplicaciones distribuidas por el ancho de banda de la red y por los recursos de computación tanto en los clientes como en los servidores.
- Los requisitos de recursos de las aplicaciones multimedia son dinámicos. Una aplicación de videoconferencia puede necesitar más o menos ancho de banda, dependiendo del aumento o disminución del número de participantes.
- A menudo los usuarios desean equilibrar los costes en recursos de las aplicaciones multimedia con otras actividades. Pueden querer reducir sus peticiones de ancho de banda conforme las actividades que estos usuarios necesiten.

Las aplicaciones multimedia interactivas plantean problemas mucho más graves. Muchas aplicaciones multimedia son cooperativas (involucran varios usuarios) y sincronizadas. Este tipo de aplicaciones demandan:

- Conmutación con baja latencia
- Estado de sincronización distribuida
- Sincronización de medios
- Sincronización externa
- Sistemas de gestión de QoS

D. *Guía Para El Diseño e Implementación De Una Aplicación Distribuida Para Brindar El Servicio de Video Bajo Demanda*

Para realizar el diseño y la implementación de la aplicación distribuida del presente proyecto se seguirá el siguiente proceso:

1) Definir el método de desarrollo
2) Obtener los requerimientos de la aplicación
   a) Definir los requerimientos funcionales
   b) Definir los roles de la aplicación
   c) Definir los requerimientos no funcionales
3) Analizar los requerimientos obtenidos
4) Definir los subsistemas de la aplicación
   a) Generación de streaming
   b) Interfaz Web de usuario
   c) Bases de datos
5) Definir y analizar los casos de uso y UML[1]
   a) Definir los diagramas secuencias
   b) Definir los casos de uso
6) Diseñar la base de datos
   a) Recopilar información obtenida por los requerimientos
   b) Dividir la información en tablas (entidades)

[1] *Unifed Modeling Language*: Es un lenguaje gráfico utilizado para diseñar, describir y documentar sistemas de software

c) Convertir la información en datos y definir las claves primarias
d) Aplicar reglas de normalización
e) Definir las relaciones entre las tablas
f) Asociar los atributos con las entidades
g) Definir el diagrama relacional

7) Diseñar el subsistema de generación de streaming

a) Diseñar el subsistema de streaming
b) Diseñar la interfaz gráfica
c) Diseñar el subsistema de carga de contenido
d) Diseño de interfaz Web
e) Definir la dinámica de la aplicación
f) Diseñar la composición de páginas Web
g) Definir el manejo de información y las funciones de la aplicación
h) Diseñar los diagramas UML

8) Dimensionar de servidores

a) Definir el número y tipos de servidores
b) Calcular capacidad de procesamientos
c) Calcular capacidad de memoria
d) Calcular capacidad de almacenamiento
e) Determinar las características de la interfaz de red

9) Determinar los requerimientos de un cliente

a) Determinar las características de software del cliente
b) Calcular capacidad de procesamientos
c) Calcular capacidad de memoria
d) Calcular capacidad de almacenamiento
e) Determinar las características de la interfaz de red

10) Definir las plataformas de implementación

11) Implementar la base de datos

a) Crear tipos de datos predeterminados
b) Implementar tablas (entidades) y sus relaciones
c) Implementar complementos de la base de datos

- Reglas para vincular los datos
- Procedimientos almacenados
- Triggers
- Vistas de información

d) Ingresar información ficticia a la base de datos (útil para el proceso de pruebas)

12) Implementar el servidor de streaming

a) Implementar el subsistema de carga de contenido
b) Implementar el subsistema de generación de streaming
c) Implementar la interfaz gráfica par administración del servidor

13) Implementación de la interfaz de usuarios (aplicación Web)

a) Implementar páginas Web de autenticación
b) Implementar páginas Web para menú
c) Implementar páginas Web para gestión y acceso a la información
d) Implementar páginas Web de reproducción de contenido

14) Probar la aplicación implementada

a) Definir la estrategia de pruebas
b) Realizar las pruebas de componentes
c) Realizar las pruebas de sistema
d) Realizar las pruebas basadas en los requerimientos
e) Probar los servidores

- Prueba de máximo número de conexiones
- Pruebas de uso de memoria
- Pruebas de uso de CPU
- Pruebas a la interfaz de red

15) Recopilar información obtenidas en las pruebas

16) Realizar ajustes de diseño e implementación

17) Realizar el análisis de tráfico mediante capturas de paquetes

18) Realizar el análisis de eficiencia de la aplicación respecto a la red

a) Calcular de eficiencias por capas TCP/IP
b) Calcular la eficiencia total
c) Calcular la velocidad efectiva

19) Definir las características mínimas de red para que esta soporte la aplicación

20) Elaborar el presupuesto referencial

a) Costos Servidores
b) Costos asociados al desarrollo de la aplicación
c) Definir costos asociados al diseño y desarrollo
d) Calcular los costos por diseño y desarrollo
e) Realizar el presupuesto total

El proceso antes descrito puede sufrir variaciones en el diseño, implementación y pruebas de la aplicación distribuida de acuerdo a las diferentes necesidades que se presenten en el desarrollo de la misma.

VII. Método de Programación Extrema

La programación extrema (XP) es posiblemente el método más ágil conocido y ampliamente utilizado. El nombre fue acuñado por Beck (Beck 2000) debido a que el enfoque fue desarrollando buenas prácticas reconocidas, como el desarrollo iterativo, y con la participación del cliente en niveles extremos.

VIII. Obtención de Requerimientos del Sistema

El video a solicitud a veces se compara con una tienda electrónica de renta de videos. El usuario (cliente) selecciona cualquiera de una gran cantidad de videos y comienza a verlo de inmediato. Usando esta descripción se definen los siguientes requerimientos:

- Es claro que existen diferentes usuarios que van a interactuar con el sistema, es decir, que se pueden definir los siguientes actores: clientes, administradores y operadores.
- El usuario del sistema debe autenticarse proporcionando un nombre de usuario y una contraseña, para lo cual debe acceder a una interfaz que brinde esa capacidad.
- Debe haber un registro de los accesos al sistema indicando: el nombre de usuario y detalles de hora y fecha.
- La información de los usuarios debe estar almacenada en una base de datos la misma que debe estar debidamente diseñada para realizar la tarifación. La base de datos debe almacenar información de la descripción del video.
- Los actores del sistema desempeñarán los siguientes roles:
  - Administrador
  - Operador
  - Cliente
- Luego de comprobada la identidad del usuario se muestra una interfaz gráfica adecuada a su rol respectivo dicha interfaz mostrará un menú en el cual se presentarán las actividades que el usuario puede realizar.

IX. Subsistemas de la Aplicación Distribuida de Video Bajo Demanda

Una vez seleccionada la arquitectura de la aplicación y los diferentes requerimientos se procede a modelar la aplicación relacionando los requerimientos y dividiéndola en niveles y posteriormente en subsistemas específicos que realizan tareas designadas. En la Fig. 6 se define los niveles que la aplicación usará para el presente proyecto.

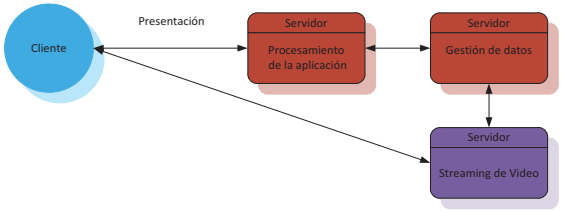

Fig. 6 Niveles de la aplicación de VoD

Al realizar un análisis de los requerimientos (de usuario y no funcionales) definidos en el presente Capítulo se puede obtener un criterio de los diferentes subsistemas que conformarán la aplicación distribuida que se desarrolla en el presente proyecto (Fig. 7).

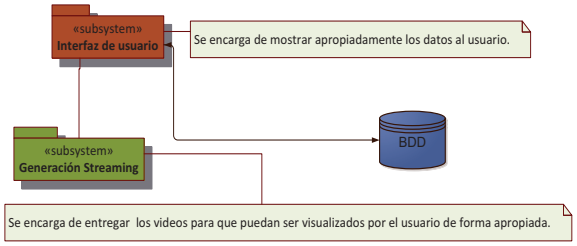

Fig. 7 Subsistemas de la aplicación distribuida

A. *Interfaz de Usuario*

Se encarga de mostrar al usuario las diferentes opciones que, dependiendo del rol está en capacidad de ejecutarlas.

B. *Generación de Streaming*

Se encarga de almacenar los recursos multimedia que serán entregados al cliente. Además se encarga de procesar estos recursos y prepararlos para que puedan viajar a través de la red.

C. *Base de Datos*

Es aquí donde se almacena la información de los usuarios del sistema, la descripción del contenido que será entregado y la relación entre usuarios y contenido.

X. Diagramas de Secuencia y Casos de Uso de la Aplicación Distribuida de Video Bajo Demanda

A. *Diagramas de secuencia*

Los diagramas de secuencia ayudan a visualizar el orden en el cual se deben llevar a cabo las actividades de los diferentes actores que interactúan con la aplicación, en base a cada uno de los requerimientos.

B. *Casos de uso*

Los casos de uso ayudan con el diseño de un sistema de software buscando las formas en que las personas interactúan con el mismo. Además los casos de uso permiten

analizar el comportamiento de la aplicación sin que se tenga que especificar la implementación de la misma.

A continuación en la Tabla I se describen los diferentes casos de uso de la aplicación de video bajo demanda y sus actores.

TABLA I
CASOS DE USO DE LA APLICACIÓN DE VIDEO BAJO DEMANDA

Caso de Uso	Actores
Iniciar Sesión	Administrador, Operador y Cliente
Buscar Contenido	Administrador, Operador y Cliente
Visualizar Contenido	Administrador y Cliente
Gestionar Contenido	Administrador
Establecer Tarifa	Administrador
Gestionar Usuarios	Administrador
Gestionar Clientes	Administrador y Operador
Visualizar Tarifación	Administrador, Operador y Cliente

XI. DISEÑO DE LA BASE DE DATOS

Para diseñar la base de datos para la aplicación de video bajo demanda del presente proyecto se sigue los siguientes pasos:

a) Determinar la finalidad de la base de datos
b) Buscar y organizar la información necesaria
c) Dividir la información en tablas
d) Convertir los elementos de información en columnas y especificar claves primarias
e) Aplicar reglas de normalización
f) Definir relaciones entre las tablas
g) Definir los diagramas
h) Realizar los ajustes de diseño

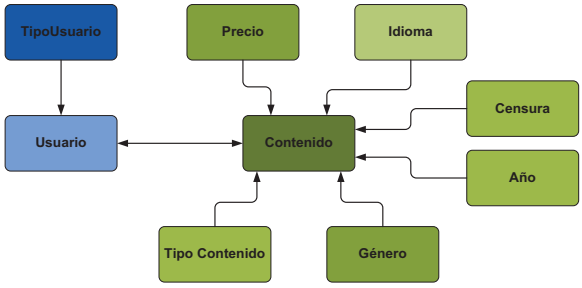

Fig. 8 Relaciones entre tablas

El resultado al aplicar correctamente este proceso es un diagrama relacional de la base de datos para la aplicación; en la Fig. 8 Se observan las relaciones entre las diferentes tablas.

XII. DISEÑO DEL SERVIDOR DE STREAMING

Un servidor de streaming es aquel que envía la información requerida por un cliente de forma tal que pueda ser visualizada sin necesidad de que la información sea descargada completamente.

A. Subsistema de Streaming

El subsistema de streaming se encarga de preparar los datos requeridos por los usuarios para que puedan viajar a través de la red de forma adecuada usando los protocolos respectivos de capa aplicación según la arquitectura de red TCP/IP; en la Fig. 9 se muestran los elementos que conforman al subsistema de streaming.

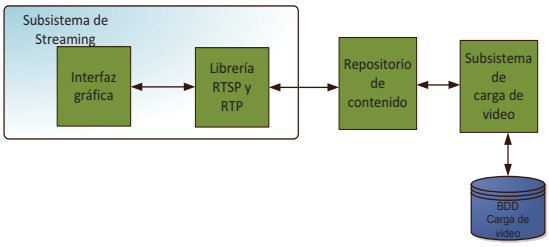

Fig. 9 Elementos del subsistema de streaming

B. Interfaz Gráfica

Esta interfaz debe contar con un formulario principal llamado "Servidor RTSP" que cuente con la opción de arrancar y parar el servidor.

Para la administración del servidor debe permitir especificar: la ubicación del repositorio de videos, número de clientes permitidos y una opción de ayuda de los parámetros de configuración. Adicionalmente debe brindar la siguiente información:

- Número de clientes conectados
- Tiempo de funcionamiento
- Hora y fecha del arranque del servidor
- Estado del servidor

C. Subsistema de carga de contenidos

Este subsistema permitirá almacenar los videos en el repositorio de contenido, debido a que el este se encuentra en un Sistema Operativo basado en Linux se usará Samba para garantizar el acceso desde otros Sistemas Operativos

XIII. DISEÑO DE LA APLICACIÓN WEB

A. Dinámica de navegación de la aplicación

Para definir la dinámica que debe presentar la aplicación es importante tomar en cuenta los roles que desempeñan los usuarios del sistema y así definir como estos roles deben ser plasmados en la aplicación Web. Una herramienta útil que ayuda a definir la dinámica de la aplicación es el análisis de los diferentes casos de uso; en la Fig. 10 se muestra la dinámica que tendrá la aplicación Web

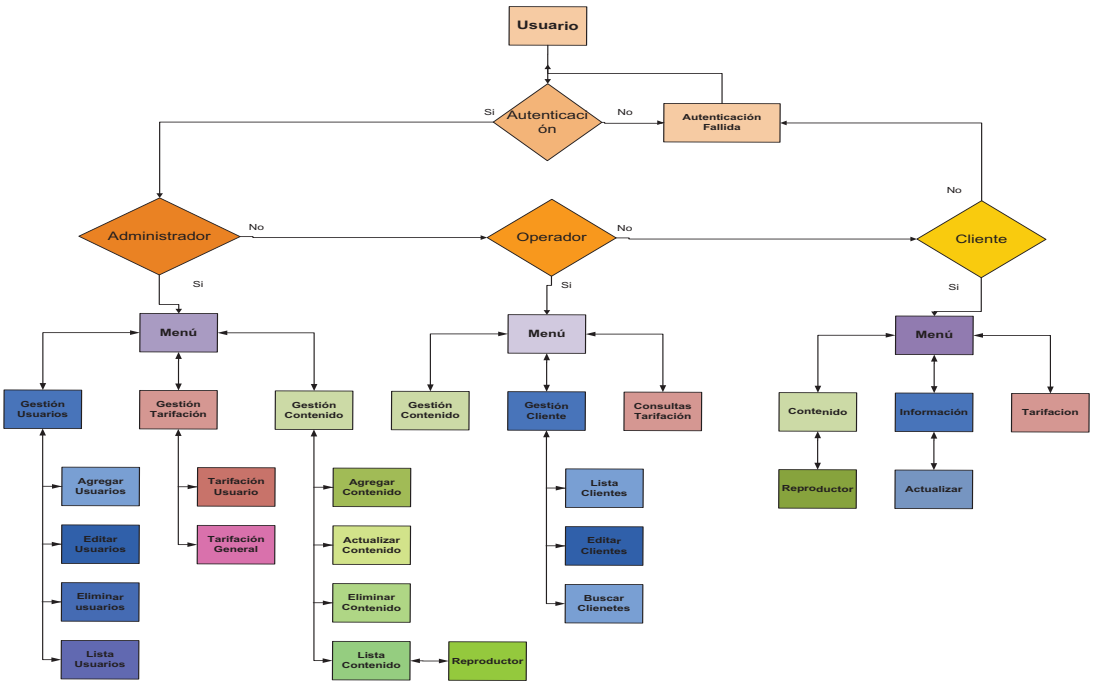

Fig. 10 Dinámica de la aplicación Web

B. *Composición de páginas Web*

La composición de las diferentes páginas a las cuales los diferentes usuarios de la aplicación tendrán acceso debe ser organizada. Para cumplir con este criterio se define la siguiente estructura general de las diferentes páginas Web de la aplicación:

1) *Documento Web:* Esta sección comprende todo el contenido de la página Web, es decir, tanto el encabezado como el cuerpo de la página Web.
2) *Formulario:* Definida dentro del cuerpo de la página, esta sección comprende lo relacionado a la operación de la página y la información relativa a la misma.
3) *Título:* En esta sección se incluye el titulo de la página Web.
4) *Funciones:* Aquí se hallan las diferentes actividades que realiza la página Web; debido a que las diferentes páginas Web que componen la aplicación ejecutan acciones distintas entre sí, esta sección será la que agrupe dichas funciones.
5) *Navegación:* Permitirá al usuario la navegación a través de las diferentes páginas Web de la aplicación, dispondrá de un botón que le permitirá al usuario regresar al menú y así poder viajar a través de las diferentes página del sistema y un botón que le permitirá abandonar el sistema si así éste lo desea.
6) *Información:* Información adicional de la página o del sistema en sí mismo, en esta sección se tendrá los nombres de los desarrolladores del proyecto e información de elementos externos utilizados en la página Web.

C. *Tipos de páginas Web*

Se han definido diferentes tipos de páginas web las cuales realizan diferentes funciones en el sistema. El definir diferentes tipos de páginas Web permite organizar la aplicación de forma precisa y de fácil acceso:

1) *Paginas de login:* En este tipo de páginas los diferentes usuarios ingresaran la información solicitada con el fin de acceder a la aplicación.
2) *Páginas de menú:* En estas páginas los usuarios pueden tener acceso a las diferentes actividades que deseen ejecutar en la aplicación.
3) *Páginas de manejo de información:* En este tipo de páginas los usuarios pueden tener a la información almacenada en la base de datos.
4) *Páginas de reproducción:* La función principal de estas es brindar la reproducción del video solicitado por el usuario.

XIV. PLATAFORMAS DE IMPLEMENTACIÓN

Una vez que se han definido los diseños de los diferentes componentes de la aplicación de video bajo demanda que se desarrolla en este proyecto, el siguiente paso es definir la forma en cómo ésta será implementada; es decir, el software de desarrollo que será utilizado para llevar a cabo la tarea de crear la aplicación basándose en los criterios de diseño expuestos hasta el momento.

De esta forma se tienen los siguientes sistemas operativos y software para la implementación de la aplicación de VoD:

1) *Aplicación Web:* Para el desarrollo de la interfaz de usuario se usará Visual Studio 2010, donde se tiene previsto desarrollar la aplicación web que cumplirá con los requerimientos antes mencionados; el lenguaje será Visual Basic.NET y HTML.
2) *Base de Datos:* Para el caso de la base de datos se desarrollará en el ambiente SQL Server 2008. Estas herramientas se usarán conjuntamente con el Sistema Operativo Windows 7 únicamente para el desarrollo. La implementación deberá ser ejecutada en un sistema operativo de tipo servidor.
3) *Subsistema de Streaming:* Para el caso del subsistema de generación de *streaming* se tiene previsto desarrollarlo en el framework Gstreamer, que usa el lenguaje de programación C. Además se usará el ambiente de desarrollo Netbeans. El sistema Operativo que se usará para este subsistema estará basado en Linux; se utilizó Fedora 16 ya que dispone de las librerias de Gstreamer.

XV. Implementación de la Aplicación Distribuida de Video Bajo Demanda

A. *Implementación base de datos*

Para la implementación de la base de datos se deben realizar los siguientes pasos:

a) Eliminar una base previamente creada con el nombre de la base que se desee
b) Crear la base de datos en el gestor
c) Utilizar la base de datos creada en el paso anterior
d) Crear los tipos de datos definidos por el usuario
e) Crear las entidades y sus respectivas relaciones
f) Crear las reglas que se aplicaran a los tipos de datos definidos por el usuario
g) Relacionar las reglas con los diferentes atributos en las entidades
h) Crear los procedimientos almacenados
i) Crear los Triggers
j) Crear Vistas
k) Insertar los datos en la base creada

B. *Implementación del subsistema de carga de contenido*

Para implementar el subsistema de carga de contenidos (Samba) se deben seguir los siguientes pasos:

a) Instalar el servidor samba y su interfaz gráfica
b) Configure el servidor mediante la edición del archivo smb.

C. *Implementación del subsistema de streaming*

Seguir los siguientes pasos para implementar el subsistema de streaming de la aplicación de video bajo demanda:

a) Instalar Gstreamer y sus componentes
b) Configurar la Variable PKG_CONFIG_PATH
c) Modificar la librería Gst-RTSP-Server para que la librería GStreamer RTSP Server trabaje con varios contenidos para varios clientes que pueden acceder al servidor simultáneamente.
d) Incluir la función analizar_url en la clase GstRTSPClient para obtener una tubería como se muestra en la Fig 11.

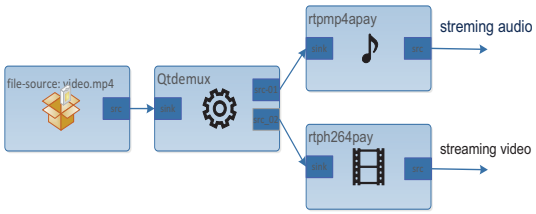

Fig. 11 Tubería de la función analizar_url

e) Implementar la interfaz de Usuario Utilizando NetBeans y QT

D. *Implementación de la aplicación web*

La implementación de la aplicación Web debe realizarse de la siguiente forma:

a) Crear las páginas Web para inicio de sesión de usuarios
b) Crear las funciones para validación de datos de inicio de sesión
c) Definir las conexiones a la base de datos
d) Implementar los formularios Web de menú
e) Implementar los submenús de gestión de contenido
f) Implementar los submenús de gestión de usuarios
g) Implementar los submenús de manejo de información de tarifación
h) Crear las páginas Web que muestran información (videos, usuarios e información de tarifación)
i) Crear los formularios para ingreso, edición y eliminación de usuarios y videos
j) Implementar las páginas Web para la reproducción de videos

XVI. Pruebas a la Aplicación de Video Bajo Demanda

El objetivo principal de probar el software es descubrir errores que pudieron ser cometidos de manera inadvertida en el diseño y la implementación del software; la Fig. 12 muestra la primera idea de una estrategia de prueba de software la cual consiste de 3 etapas:

1) *Pruebas de componentes o unidades:* Prueba los componentes individuales de la aplicación con la finalidad de ver la correcta función de cada uno de ellos.
2) *Pruebas de sistema:* Integrando todas las unidades en un solo sistema estas pruebas pretenden verificar la

correcta interaccione entre estas unidades basándose en los requerimientos funcionales y no funcionales de la aplicación definidos en la etapa de diseño.

3) *Pruebas de aceptación:* Estas pruebas permiten probar la aplicación utilizando información real proporcionada por los diferentes usuarios de dicha aplicación.

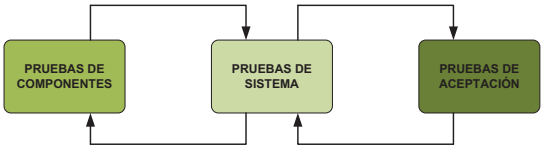

Fig. 13 Proceso de pruebas

Las pruebas de realizadas a la aplicación de video bajo demanda del presente proyecto son:

A. Pruebas de componentes

- Pruebas de la base de datos
- Pruebas de carga de contenidos
- Pruebas de streaming
- Pruebas de autenticación
- Pruebas acceso a datos
- Pruebas de tarifación
- Pruebas de presentación de contenido

B. Pruebas de sistema

- Pruebas de integración de los servidores
- Pruebas de entrega de información
- Pruebas de rendimiento

C. Pruebas de aceptación basadas en requerimientos

- Pruebas a los requerimientos de usuarios
- Pruebas de acceso según perfil de usuario
- Pruebas de funciones según el perfil del usuario
- Pruebas de selección y reproducción de contenido (audio y video)
- Pruebas a los requerimientos de la aplicación
- Pruebas de independencia de sistema operativo
- Pruebas de confiabilidad y entrega de información

Se comprueba el funcionamiento de la aplicación mediante la captura y análisis de paquetes para lo cual se capturaron los siguientes tipos de paquetes:

- SDP
- RTSP
- RTP
- RTCP
- HTTP

En base a los resultados obtenidos mediante esta captura de paquetes y el análisis de estos se determinan los requerimientos mínimos de una red LAN basada en TCP/IP para que esta pueda entregar los servicios brindados por la aplicación; en la Tabla II se observan los resultados.

TABLA II
CARACTERÍSTICAS DE LA RED

Característica:	Descripción:
Tipo de Red de comunicaciones	Área local
Número máximo de usuarios de la red	200
Retardo máximo permitido	150 mili segundos
Velocidad efectiva mínima	1.11 Mbps
Backplane mínimo necesario	53,28Mbps
Eficiencia de red	95,98%
Tipo de puerto de conmutación	Ethernet
Velocidad de puertos en equipo de conmutación	100 Mbps
Arquitectura de red	TCP/IP
Densidad de puertos del equipo de conmutación	48
Velocidad de reenvío	100 Mbps
Índice de simultaneidad	25%
Capacidad de memoria de almacenamiento en el equipo de conmutación	8,23 MB
Soporte para VLAN	802.1q

XVII. CONCLUSIONES

La trasmisión de datos en tiempo real se puede definir como información que llega a su destino de forma rápida y con ciertos parámetros determinados como son: baja distorsión, orden, sin pérdida o que el contenido sea capaz de recuperarse cuando existen estas pérdidas e incluso de errores.

VoD (Video On Demand) es la denominación más común para sistemas de distribución por suscripción de video utilizando suscripciones de banda ancha sobre el protocolo IP. Es decir el trasporte de VoD se da sobre redes basadas en IP capaces de soportar los requerimientos de este servicio. El servicio de video bajo demanda se lo puede definir también como una aplicación que espera, procesa y sirve peticiones de uno o varios clientes.

El desarrollo de software tiene 3 etapas principales diseño, implementación y prueba del mismo; donde cada etapa depende de la anterior. Es importante definir los requerimientos tanto funcionales y no funcionales ya que estos permiten tener una primera aproximación de la forma en cómo el software funcionará, en cómo éste estará estructurado y como el mismo deberá cumplir con sus funciones.

Debido al desarrollo iterativo que XP propone, la documentación de un proyecto se complica, puesto que las diferentes modificaciones que se llevan a cabo en cada iteración no siempre son definitivas, pueden existir cambios significativos en cada iteración.

Probar el software basándose en los requerimientos definidos durante el diseño permite verificar que el software implementado cumpla con los mismos, estas pruebas deben ser realizadas una vez que se han finalizado con todas las pruebas de unidad e integración, en este proyecto las pruebas basadas en los requerimientos son realizadas junto con las pruebas del sistema y los resultados son utilizados para el análisis de la eficiencia y capturas de paquetes.

El método de programación extrema permite realizar los ajustes necesarios de diseño e implementación en la aplicación distribuida ya que la facilidad de realizar iteraciones y la posibilidad de ajustes de software que plantea XP permitió solucionar errores detectados durante la etapa de pruebas y ajustarlos sin la necesidad de replantear todo el proyecto.

XVIII. REFERENCIAS

[1] NetBeans: http://netbeans.org/index_es.html. [agosto 2012].
[2] Word Wide Web Consortium: http://www.w3.org/1999/05/WCAG-REC-fact. [septiembre 2012].
[3] NetBeans: http://netbeans.org/index_es.html. [septiembre 2012].

XIX. BIOGRAFÍAS

Omar P. Montalvo
Nació el 12 de agosto de 1984 en la ciudad de Atuntaqui ubicada en la provincia de Imbabura - Ecuador; realizó sus estudios secundarios en el Colegio Nacional Juan Pio Montúfar en la ciudad de Quito; realizó sus estudios de universitarios de Ingeniería Electrónica y Redes de Información en la Escuela Politécnica Nacional, culminando los mismos en el año 2012. A finales del año 2012 ingreso a ejercer su profesión en Compuequip DOS como ingeniero de servicios especialista en networking, puesto que lo ocupo hasta septiembre de 2013 donde cambio sus funciones por Ingeniero de diseño y preventa de la unidad de redes y telecomunicaciones.

Byron P. Vicente
Nació el 29 de agosto de 1984 en la ciudad de Quito, provincia de Pichincha - Ecuador; realizó sus estudios secundarios en el Instituto Tecnológico Superior Central Técnico de la ciudad de Quito; estudió Ingeniería Electrónica y Redes de Información en la Escuela Politécnica los cuales los finalizó en el año 2012; actualmente desempeña el cargo de Analista en la Unidad de Tecnologías de la Información y Comunicación en el Consejo de Evaluación, Acreditación y Aseguramiento de la Calidad de la Educación Superior.