PROTOTIPO DE SOFTWARE PARA LA ENTREGA DEDICADA DE AUDIO Y VIDEO ORIENTADO A DISPOSITIVOS MÓVILES

Autores:

Julián Ricardo Hernández Mariño
Carlos Mario Paredes Suárez

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2012

PROTOTIPO DE SOFTWARE PARA LA ENTREGA DEDICADA DE AUDIO Y VIDEO ORIENTADO A DISPOSITIVOS MÓVILES

Autores:

Julián Ricardo Hernández Mariño
Carlos Mario Paredes Suárez

Tesis de grado presentada como requisito para optar al tíitulo de:

Ingeniero de Sistemas

Director:

Mcc. Fernando Rojas Morales
Escuela de ingeniería de Sistemas e informática
Facultad de Ingenierías Físico-Mecánicas

Codirector:

Ing. Sergio Antonio Pino Gallardo
Escuela de ingeniería de Sistemas e informática
Facultad de Ingenierías Físico-Mecánicas

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2012

Dedicatoria

Dedico este trabajo a Dios, pues su amor es el que hace que mi vida cobre sentido. A mi Madre Maria Gladys Mariño, y a mi hermano Oscar Leonardo Gil Mariño, quienes son la razón de mis logros, el motor de mi vida y la razón por la que lucho día a día por salir adelante.
A mi compañero de trabajo de grado y amigo, Carlos Mario, por haber sido mi compañía en el logro de este sueño, quien me levanto en los momentos difíciles, y por ser uno de mis ejemplos a seguir.
A Fernando Rojas y Sergio Antonio Pino, ya que las lecciones que aprendí de ellos fueron fundamentales para el desarrollo del proyecto, y para entender la vida.

Julián Ricardo

A mi Creador, porque me dio fuerzas cuando estaba cansado, multiplicándolas cuando no tenía ninguna. El mismo Dios que ha puesto en mi camino a mis familiares y seres queridos, que sin dudarlo han hecho grandes esfuerzos para que este día exista.

Carlos Mario

Agradecimientos

"A Dios, pues a él todo en mi vida se lo debo. A mi Madre Maria Gladys Mariño, y a mi hermano Oscar Leonardo Gil Mariño, quienes son la razón de mis logros, el motor de mi vida y la razón por la que lucho día a día por salir adelante.

A mi Director Fernando Rojas y mi codirector Sergio Pino, pues fueron quienes nos guiaron en todo el proceso de este trabajo.

Al grupo de investigación en ingeniería biomédica GIIB, por haber permitido que este proyecto fuera desarrollado dentro de su dominio.

A Carlos Mario, mi compañero y apoyo para el presente trabajo de grado. A mis amigos y compañeros, que en algún momento fueron de ayuda y soporte para el cumplimiento de mis metas propuestas."

Julián Ricardo

"Agradezco principalmente a Dios, el Eterno que me ha acompañado durante todo el recorrido de mi vida, dando fortaleza y bendición junto a mis progenitores Sally y Carlos.

A mi hermana Sally por su paciencia y respeto.

También agradezco a mis familiares, por dar palabras de apoyo y confianza.

A mis amigos y compañeros de estudio, con los cuales he aprendido y mejorado, particularmente a Julián Ricardo quien aportó verdadero valor a este proyecto.

A mis maestros, en especial al director y al codirector del proyecto Fernando Rojas y Sergio Pino, quienes con tiempo, y conocimiento guiaron este estudio final.

A la Escuela de Ingeniería de Sistema e Informática de la Universidad Industrial de Santander por tan grande oportunidad de pertenecer a ella.

A todos gracias!"

Carlos Mario

Tabla de Contenido

Índice de Cuadros

Índice de Figuras

Resumen

Título: Prototipo de software para la entrega dedicada de audio y video orientado a dispositivos móviles.*

Autores: Julián Ricardo Hernández Mariño, Carlos Mario Paredes Suárez**

Palabras claves: Entrega bajo demanda, Streamin, JME, Video en dispositivos móviles, API Multimedia.

Descripción: Este trabajo de investigación presenta el diseño e implementación de un prototipo de software que permite el unicast o entrega bajo demanda de contenido multimedia de audio y video entre dispositivos móviles.

El prototipo de software desarrollado se presenta como una aplicación móvil que funciona como alternativa de comunicación. El cual fue desarrollado en dos secciones analizadas con mayor detalle en este documento. La primera sección, describe los hallazgos y dificultades encontradas en el desarrollo de un módulo para compartir un video, que consiste en la selección de un archivo de video localizado en el dispositivo móvil y su posterior almacenamiento en un servidor dedicado, haciendo uso del Protocolo de Transferencia de Hipertexto (HTTP). La sección que continúa, documenta la forma cómo se implementó el módulo de streaming, el cual se conecta a un servidor especializado y hace uso de los protocolos especializados como Protocolo de Transporte de Tiempo Real (RTP) y Protocolo de Flujo de Tiempo Real (RTSP) para visualizar un video en el dispositivo móvil.

La aplicación está desarrollada en lenguaje Java para Móviles (JME), el cual se adapta a los recursos y requerimientos de hardware y software de los dispositivos que posean una máquina virtual de Java, sin embargo es necesario que los mismos cuenten con ciertas especificaciones técnicas para que el prototipo funciones de manera satisfactoria.

*Trabajo de Investigación

**Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería de Sistemas. Director: Fernando Rojas Morales. Codirector: Sergio Antonio Pino

Abstract

Title: DESING AND DEVELOPMENT OF A SOFTWARE PROTOTYPE TO UNICAST MULTIMEDIA AUDIO AND VIDEO BETWEEN MOBILE PHONES[***]

Authors: Julián Ricardo Hernández Mariño, Carlos Mario Paredes Suárez[****]

Index terms: Mobile Unicast, Streaming, JME, Multimedia API, Mobile Video.

Description: This research project presents the design and implementation of a software prototype that allows multimedia, audio and video, unicast between mobile phones in which the application have been installed.

The developed software prototype shows an alternative means of communication. Consist of two major elements analyzed throughout this document. The first section describes the findings and difficulties encountered in the development of a module to share video content from a mobile device, which consists of selecting a video file located in the phone and its subsequent sending and storage in a dedicated server, using the Hypertext Transfer Protocol (HTTP). The next section shows the streaming module; which connects to Darwin Streaming Server (DSS), a specialized server, and makes use of specific protocols such as Real Time Transport Protocol (RTP) and Real Time Streaming Protocol (RTSP) to play and stop a video content in a mobile devices using streaming.

The application is developed in Java Micro Edition (JME) and uses the Multimedia API (MMAPI), which adapts to the resources and requirements of hardware and software devices that have a Java virtual machine, however it is necessary that they have certain specifications for the prototype functions satisfactorily.

[***] Research Project

[****] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería de Sistemas. Director: Fernando Rojas Morales. Codirector: Sergio Antonio Pino

INTRODUCCIÓN

Los dispositivos móviles y la conexión a internet en la última década han mejorado la comunicación entre las personas, permitiendo cada vez más calidad y eficiencia en el proceso.

La tecnología móvil de vanguardia ha avanzado en múltiples ámbitos dando paso a la creación de herramientas software útiles e innovadoras que realizan procesos de entrega dedicada o unidifusión de contenido multimedia sin mayores complicaciones, lo cual en tiempos pasados era tan solo fruto de la imaginación o procesos de investigación.

El estudio de esta área de conocimiento, es cada vez más atractivo y motivador, pues la idea de movilidad, comunicación y practicidad son conceptos que enriquecen la experiencia de los usuarios.

La tecnología Java ME proporciona un entorno robusto y flexible para aplicaciones que se ejecutan en dispositivos móviles. Es por eso que el presente estudio demuestra los alcances y detalles tecnológicos necesarios para realizar procesos de unidifusión usando dispositivos móviles, soportados en plataformas de desarrollo basadas en Java ME.

La aplicación realizada consiste en un prototipo de software basado en una arquitectura cliente/servidor que permite al usuario compartir un elemento multimedia de video, obtenido con un dispositivo móvil para ser observado posteriormente en una plataforma de caracteristicas similares, usando la técnica de streaming, optimizando recursos y evitando el almacenamiento en la memoria de este dispositivo final.

El prototipo de software no pretende reemplazar la comunicación convencional, se presenta como una extensión a esta y da una experiencia nueva y enriquecedora al usuario.

1. DESCRIPCIÓN DEL PROYECTO

1.1. Alcances del Proyecto

El constante aumento de la tecnología en dispositivos móviles ha permitido un intenso estudio y una ardua investigación en temas relacionados. La sociedad actual dispone de herramientas que mejoran su estilo de vida, permitiendo esencialmente el aumento favorable en la calidad y capacidad de comunicación.

Además, la comunicación no está simplemente limitada a un único sentido, esta por el contrario se extiende en diferentes direcciones. Dando nuevas posibilidades de aprendizaje y métodos más didácticos y eficientes para entregar la información. Sin embargo la necesidad de mejorar estos servicios y la interacción social entre las personas es un factor importante en el estudio de las telecomunicaciones.

En la actualidad tanto emisor como receptor tienen la tecnología para compartir y recibir contenido multimedia usando dispositivos móviles, siendo esto tan importante, su valor agregado es el uso y optimización de los recursos. A pesar de estos avances, se hace necesaria la creación de una herramienta que permita mejorar la comunicación en escenarios como aulas de clase, salas de conferencias y reuniones, entre otras, permitiendo la distribución de elementos multimedia como audio y video.

Es por esta razón que la investigación esta orientada a la realización de un prototipo que permita la entrega de este contenido para ser consultado remotamente, contribuyendo el aprovechamiento de la tecnología en situaciones como las anteriormente mencionadas.

1.2. Objetivo General

Diseñar e implementar un prototipo software que permita la entrega dedicada de audio y video sobre dispositivos móviles, con el fin de establecer comunicación entre personas que no se encuentran presentes en el mismo lugar.

1.3. Objetivos Específicos

- Elaborar y obtener los requisitos funcionales y no funcionales de un prototipo software que permita acceder al servicio de unicast de audio y video en dispositivos móviles.
- Diseñar e implementar el modulo inicial de carga del video capturado por la cámara del dispositivo móvil.
- Diseñar e implementar el segundo modulo encargado de realizar la descarga y visualización del video utilizando la técnica de streaming desde el servidor al dispositivo móvil.

- Verificar el desempeño de la herramienta desarrollada, realizando pruebas en simuladores y comparando resultados con dispositivos físicos.
- Optimizar la herramienta implementada, ajustando y mejorando el código del prototipo aplicando la técnica de refactorización, para obtener un prototipo final más estable.
- Realizar la documentación técnica del prototipo desarrollado para proporcionar información que tiene que ver con el código y la arquitectura utilizada para el desarrollo del mismo, así como también el manual de usuario correspondiente.

2. MARCO TEÓRICO

2.1. Telefonía Móvil

2.1.1. Introducción a la telefonía móvil

Desde sus inicios, la telefonía móvil ha revolucionado de manera drástica las actividades cotidianas de los seres humanos, ya que hace algunas décadas parecía difícil logar comunicación sin cables. A pesar de que fue pensada solo para voz debido a las limitaciones tecnológicas que existían en la época, en la actualidad se habla de comunicación móvil al conjunto de redes que permiten transferencia no solo de voz sino de datos y archivos multimedia a grandes velocidades, y hoy en día sigue el perfeccionamiento de estas redes para alcanzar aun mayores velocidades de transferencia. Los dispositivos móviles se han convertido en una herramienta primordial para las personas que desean mantenerse en contacto sin que las distancias sean un impedimento.

El reporte de los operadores figura 1 indica que a 30 de junio de 2011 existían 46.714.934 abonados en telefonía móvil (líneas activas), según informes debidamente auditados. Se presentó un aumento de 1.293.840 abonados entre el primer y el segundo trimestre de 2011, que equivale al 2,85 %.

Colombia Móvil S. A. ESP. (TIGO), Comunicación Celular S. A. Comcel S. A. y Telefónica Móviles Colombia S. A. (Movistar) son los operadores de la telefonía móvil en Colombia.

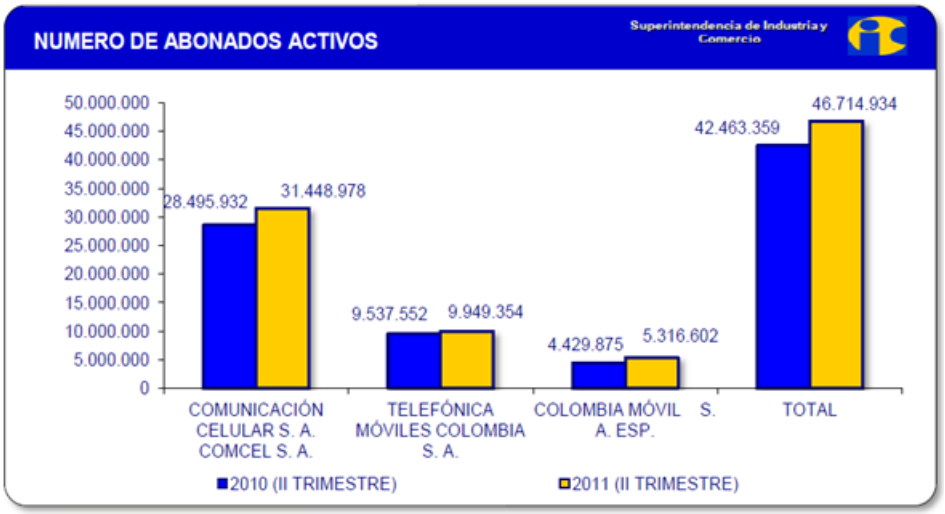

Figura 1: *Estadísticas de usuarios abonados en telefonía móvil.*Fuente: Superintendencia de Industria y Comercio [1]

2.1.2. Generaciones de la telefonía móvil

Primera Generación (1G): esta primera generación hizo su aparición en 1979, pero tuvo su real importancia durante los años 80. Esta generación se encargo de introducir como tal a los

teléfonos celulares, basados en redes celulares con múltiples estaciones de base relativamente cercanas unas de otras, y protocolos para el traspaso entre las celdas cuando el teléfono se movía de una a otra.
La principal característica es que durante esta generación la transferencia era analógica y estrictamente para voz. Así mismo, tanto la calidad de enlaces, como la velocidad de conexión eran muy reducidas. Otro de los inconvenientes a resaltar es que esta primitiva telefonía celular no contaba con medidas preventivas relacionadas con seguridad. La tecnología predominante de esta generación es AMPS[5][2].

Segunda Generación (2G): La segunda generación llego hasta 1990, y la diferencia más importante con respecto a la primera es que se caracterizó por ser digital. El sistema 2G utiliza protocolos de codificación más adecuados. Las tecnologías predominantes para esta generación son: GSM[6]; IS-136[7] y CDMA[8] y PDC[9]. Los protocolos empleados en los sistemas 2G soportan velocidades de transmisión de voz más altas, pero limitados en comunicación de datos. Se puede ofrecer servicios auxiliares, como los SMS . La mayoría de los protocolos de 2G ofrecen diferentes niveles de encriptación.

Debido a un tema de costos, los operadores de telefonía móvil migraron primero de 2G a una generación de transito 2.5G antes de implantar la 3G, esta generación es más rápida que las anteriores. En esta generación se incluyen nuevos servicios como EMS[10] y MMS[11] . Para poder prestar estos nuevos servicios se hizo necesaria una mayor velocidad de transferencia de datos, que se hizo realidad con las tecnologías GPRS[12] y EDGE[13].

Tercera generación (3G): La tercera generación nace de la necesidad de aumentar la capacidad de transmisión de datos para ofrecer servicios como la conexión a internet desde el móvil, la videoconferencia, la televisión y la descarga de archivos. En este momento el desarrollo tecnológico ya posibilita un sistema totalmente nuevo: UMTS[14]. UMTS utiliza tecnología CDMA, lo cual le hace alcanzar velocidades realmente elevadas (de 144 Kbps hasta 7.2 Mbps).

Cuarta generación (4G): La cuarta generación es un proyecto a largo plazo que estará basado totalmente en IP, siendo un sistema de sistemas y una red de redes. 4G será la evolución tecnológica que ofrecerá al usuario de telefonía móvil un mayor ancho de banda que permitirá, entre muchas otras cosas, la recepción de televisión en alta definición.

[5]AMPS *Advanced Mobile Phone System*
[6]GSM *Global System for Mobile Communications*
[7]IS-136 *Interim Standard 136*
[8]CDMA *Code Division Multiple Access*
[9]PDC *Personal Digital Communications*
[10]EMS *Enhanced Messaging Service*
[11]MMS *Multimedia Messaging Service*
[12]EMS *General Packet Radio Service*
[13]EDGE *Enhaced Data rates for GSM evolution*
[14]UMTS *Universal Mobile Telecommunications System*

2.1.3. Internet móvil

Con el pasar del tiempo, las tecnologías emergentes que facilitan las comunicaciones sin importar distancias evolucionan para hacernos las cosas un poco más fáciles. El internet móvil es una de estas cosas que han llegado para quedarse y hacer de nuestras comunicaciones un proceso más sencillo. Las mejoras en las características de los dispositivos móviles como su capacidad computacional, su memoria de almacenamiento y su versatilidad, hacen que el internet móvil sea posible en esta clase de dispositivos. El crecimiento del ancho de banda disponible para la transferencia de datos en las redes actuales hace también que el internet móvil haya dejado de ser una utopía y se adapte a los requerimientos que estos necesitan para la distribución de información a través de la red.

El internet móvil es usado en aplicaciones empresariales como fuerza de ventas, en m-learning, m-commerce o entretenimiento para descarga de aplicaciones, juegos, audio o video, sin dejar atrás los servicios relacionados con la localización a través de mapas, todas estas, aplicaciones que están en pleno auge en nuestros días.

2.1.4. Computación móvil

La computación móvil es una forma de interacción entre el humano y el computador, en donde se espera que el computador pueda ser transportado durante su uso normal. La computación móvil tiene tres aspectos fundamentales: la comunicación móvil, los equipos móviles, y el software móvil. El primer aspecto se ocupa de cuestiones de comunicación en redes ad-hoc (red inalámbrica descentralizada) y redes de infraestructura, así como propiedades de comunicación, protocolos, formatos de datos y tecnologías concretas. El segundo aspecto es sobre el hardware, por ejemplo, dispositivos móviles o componentes de dispositivo. El tercer aspecto se refiere a las características y requerimientos de las aplicaciones móviles.[3]

Debido a la evolución que los dispositivos móviles han experimentado en los últimos años en términos de procesamiento y memoria y al gran avance de las comunicaciones inalámbricas como las redes de telefonía celular, Wi-Fi, Wi-Max y Bluetooth, el paradigma de la computación móvil ha tomado mucha fuerza, y es así como las organizaciones han optado por implementar este tipo de tecnologías para aumentar la productividad y optimizar procesos.

La movilidad es el referente de este paradigma, haciendo posible que los usuarios de terminales móviles conectados a internet puedan realizar actividades de procesamiento de información sin importar donde se encuentren. Así pues la computación móvil es un recurso que permite comunicar, procesar datos y acceder a información remota cuando se requiera a través de dispositivos móviles.

2.1.5. Limitaciones de la computación móvil

- **Ancho de banda insuficiente:** El acceso a internet móvil es generalmente más lento que mediante conexiones de cable directos, usando tecnologías como GPRS Y EDGE, y más recientemente redes HSDPA[15] y HSUPA[16].

- **Estándares de seguridad:**La seguridad es una preocupación importante, ya quecuando está trabajando un móvil, este es dependiente de redes públicas, requiriendo un uso cuidadoso de las VPN[17].

- **Consumo de energía:** Cuando una toma corriente o un generador de energía portátil no está disponible, las computadoras móviles deben depender enteramente de la energía de la batería; esto, en combinación del tamaño compacto de muchos dispositivos móviles, a menudo significa que baterías muy caras deberían ser usadas para obtener la vida de la batería necesaria.

- **Interferencias en la transmisión:** El terreno, el clima, y el rango desde el punto de señal más cercano pueden interferir con la recepción de la señal. La Recepción en túneles, algunos edificios, y áreas rurales suelen ser deficientes.

- **Interfaces pequeños:** Debido a que las pantallas y los teclados tienden a ser pequeños, puede a veces resultar difícil el uso de los dispositivos móviles. Métodos alternativos para la entrada de datos como reconocimiento de voz requieren entrenamiento[4].

2.2. Streaming

2.2.1. Definición

El streaming es una técnica utilizada para la distribución de contenido multimedia en Internet, con la característica de poder visualizar y/o escuchar estos contenidos en el cliente cuando la información aun está siendo trasmitida por la red, de tal manera que no es necesaria la descarga completa del contenido multimedia en el cliente para que este pueda empezar su reproducción.[5]

2.2.2. Funcionamiento

Para la distribución del contenido, el primer paso tiene que ver con la conexión del cliente con el servidor de streaming, para que este le empiece a enviar el fichero en el momento en que el cliente lo desee. Una vez que el cliente comienza a recibir el fichero, construye un buffer donde empieza a guardar la información. Así pues, cuando se ha llenado el buffer con una pequeña parte del archivo, el cliente puede acceder a este, y lo puede escuchar y/o visualizar

[15]HSDPA *High Speed Downlink Packet Access*
[16]HSUPA *High-Speed Uplink Packet Access*
[17]VPN *Virtual Private Network*

mientras que la descarga continua realizándose. El sistema esta sincronizado para que el archivo se pueda escuchar y/o visualizar mientras que el archivo se descarga; entonces una vez finaliza la descarga, también finalizara el acceso a dicho fichero figura 2[5].

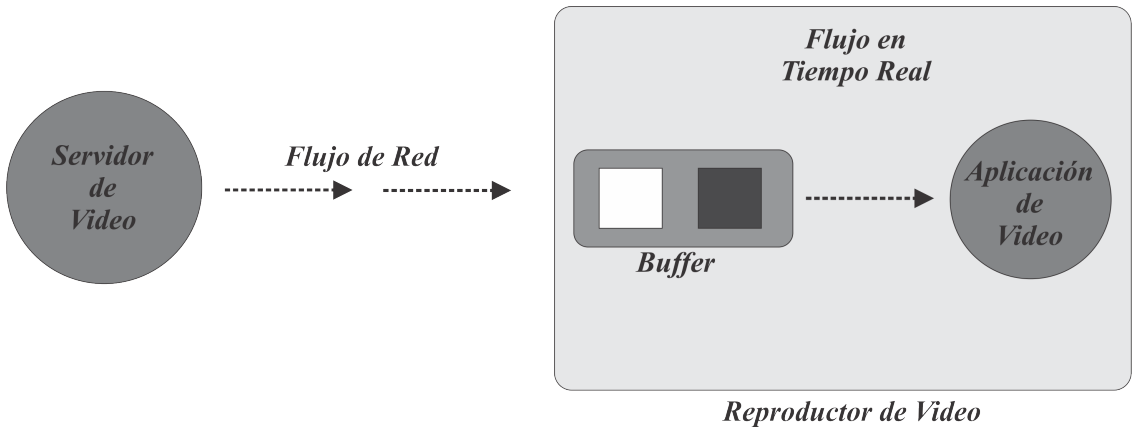

Figura 2: *Funcionamiento del streaming de video.* Fuente: Evaluación de servidores de streaming de video orientado a dispositivos móviles, proyecto de grado de la universidad de Antioquia.

Hablar de "streaming en tiempo real" es tal vez no hablar con exactitud, ya que en realidad el contenido multimedia se transmite en forma confluente debido a que existen factores que retrasan el acceso a estos ficheros, como lo son el ancho de banda, congestión en la red, o capacidades de recepción del cliente.

2.3. Componentes de un sistema streaming

Un esquema general de un servicio de streaming consta de tres componentes, la generación del contenido, la disposición y administración del contenido en unservidor y la reproducción figura 3.

Generación del Contenido: La generación del contenido es realizada por cualquier dispositivo apto para capturar y almacenar en memoria cualquier tipo de contenido multimedia, como un teléfono celular o una cámara de video.Además es preciso editar y/o codificar los datos multimedia según los formatos designados para el sistema teniendo en cuenta aspectos como el ancho de banda de la red y los formatos soportados por el servidor y el reproductor del cliente. Si la fuente genera contenido análogo, es necesario digitalizarlo.

Disposición y administración del contenido en un servidor: Se debe tener en cuenta que para la difusión de contenido multimedia por la red se hace imprescindible de servidores de streaming que gestione las conexiones yanchos de banda de los usuarios conectados al sistema. Además de estar codificado el video en los formatos exigidos, algunos servidores de streaming exigen que para que los archivos puedan ser obtenidos mediante streaming deben haber pasado por un proceso denominado hinting, el cual permite el proceso sin generar errores.

Reproducción: El reproductor del cliente es la última etapa de unsistema streaming. Lo más importante a tener en cuenta respecto alreproductor son los tipos de formatos de archivo

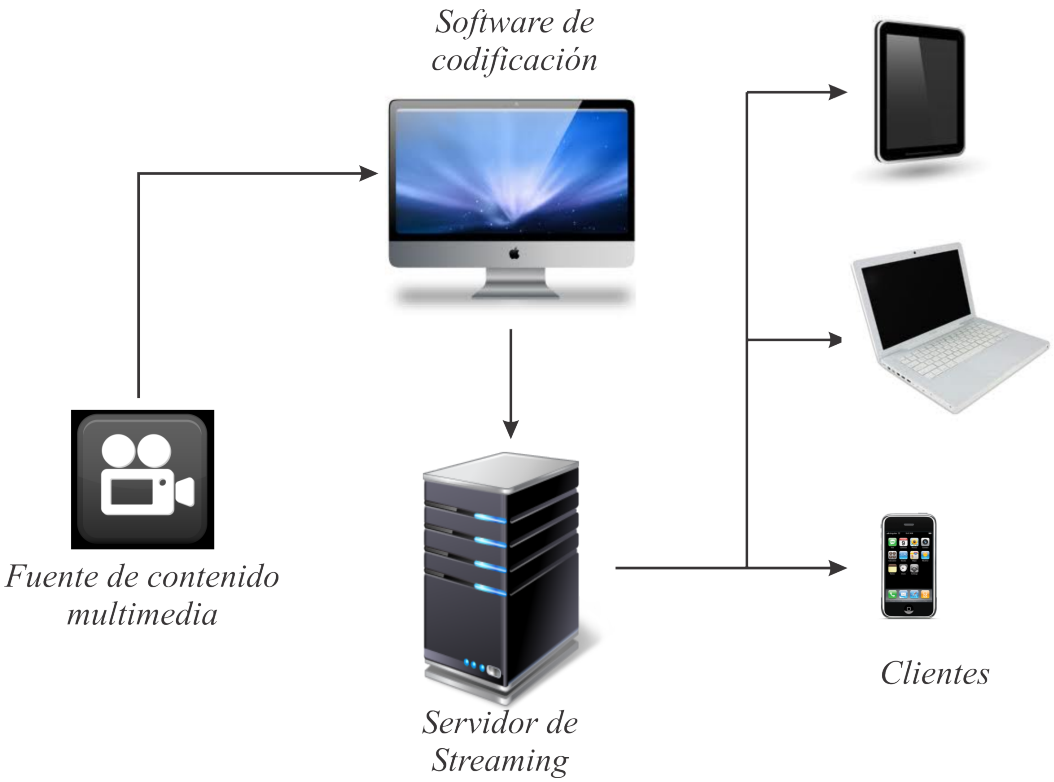

Figura 3: *Servicio de streaming.* Fuente: Evaluación de servidores de streaming [6] editado por Autores del Proyecto

que este puede reproducir.El reproductor es parte fundamental de un sistema streaming pues entresus responsabilidades está la interacción con el usuario a través de una interfaz, el manejo de errores en la trasmisión y la decodificación de losdatos recibidos.

2.4. Esquemas de distribución

2.4.1. Broadcast

En redes de computadores el término Broadcasting hace referencia a la difusión de paquetes de información por red que serán entregados simultáneamente a todos los clientes que pertenezcan a esta red. La difusión a través de Boadcasting está limitada a un dominio bradcast. Generalmente las redes que soportan Broadcasting están asociadas a redes de área local.[5]

2.4.2. Unicast

Por contraposición a broadcast y multicast, unicast es la comunicación establecida entre un solo emisor y un solo receptor en una red. Esto significa que cada cliente recibe un flujo distinto por parte de un servidor, y lo reciben solo los clientes que así lo soliciten.

2.4.3. Multicast

Multicast es un servicio de red en el cual un único flujo de datos, proveniente de una determinada fuente, puede ser enviado simultáneamente para diversos destinatarios. El multicast es dirigido para aplicaciones del tipo uno-para-varios y varios-para-varios, ofreciendo ventajas principalmente en aplicaciones multimedia compartidas. Para realizar una distribución multicast, es necesario que los routers en el trayecto entre el servidor y el grupo estén habilitados para distribución multicast.

2.5. Protocolos de streaming

La gran ruptura que dio paso a la revolución del Streaming fue la adopción del protocolo UDP[18] junto con nuevas técnicas de codificación que comprimen audio y video en paquetes pequeños de datos. UDP hace el Streaming de multimedia posible permitiendo la transmisión de datos más eficientemente que protocolos previos, en situaciones donde se envían flujos de datos desde un servidor sobre internet hasta el reproductor del cliente final. Protocolos más recientes como el RTSP[19] hacen que la transmisión de datos sea aun más eficiente.

UDP y RTSP son ideales para el Streaming, ya que ellos son protocolos no orientados a conexión, colocan una alta prioridad al envió de flujos de manera continua e ininterrumpida, en vez de colocar la atención exclusivamente en la seguridad. Así pues, a diferencia de las transmisiones mediante protocolos que si son orientados a conexión como TCP[20], cuando un paquete de audio/video UDP se pierde, el servidor sigue enviando paquetes nuevos, causando solamente un pequeño lapso de silencio en el cliente en vez de un espacio prolongado que se produciría si el servidor intentara reenviar el paquete perdido. Por otro lado, TCP intenta permanentemente el reenvió de paquetes perdidos antes de enviar uno nuevo, causando grandes espacios de silencio en los reproductores del cliente y rupturas en el Streaming de audio o video. En la figura 4 se representa el stack de protocolos.

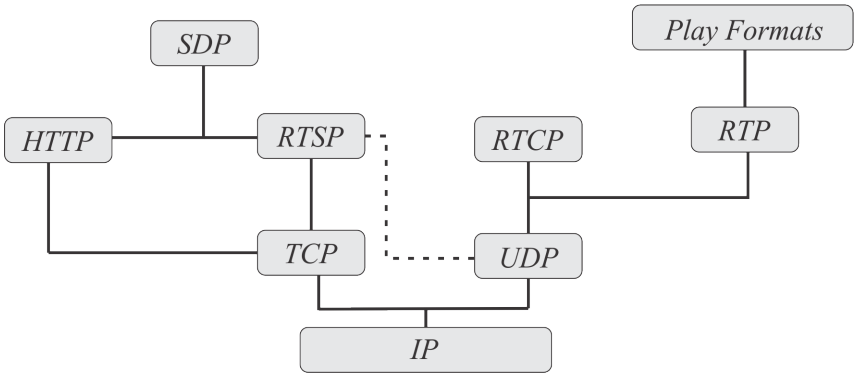

Figura 4: *Pila de protocolos para el estándar 3GPP PSS* Fuente: Medialab[7]

[18]UDP *User Datagram Protocol*
[19]RTSP *RealTime Streaming Protocol*
[20]TCP *Transmission Control Protocol*

Sin embargo, antes de utilizarse las transmisiones bajo UDP y RTSP, los datos fueron enviados sobre la web exclusivamente vía TCP y HTTP[21]. Hoy en día Las transmisiones TCP, en contraste a las transmisiones UDP y RTSP, son designadas principalmente para la transferencia fiable de documentos de texto, email, y paginas HTML[22] sobre internet, siempre colocando la máxima atención a la fiabilidad e integridad de los datos, en vez de la puntualidad de estos. Así pues, debido a que las transmisiones HTTP se basan en TCP, este tipo de transmisiones no se adaptan bien para presentaciones multimedia utilizando Streaming ya sea Unicast, Multicast o Broadcast.

Algunas tecnologías de Streaming como Real Audio y Windows Media utilizan servidores dedicados que soportan transmisiones UDP y RTSP. Otros formatos son designados para Streaming desde un servidor web HTTP estándar. Mientras esos formatos son de menos calidad y a menudo más fáciles de usar ya que no requieren de la instalación de un nuevo servidor, ellos no son usados en situaciones profesionales de Broadcasting que requieren la distribución de cientos o miles de streams simultáneos.

Streaming bajo HTTP es a menudo referido como pseudo-Streaming, ya que aunque técnicamente es posible, no es la mejor alternativa[8].

2.5.1. RTP

El protocolo RTP[23] provee servicios de entrega de datos con características de tiempo real desde un dispositivo final a otro, como por ejemplo audio y video interactivo. Estos servicios incluyen identificación del tipo de carga útil, numeraciones de secuencia, "time stamping" y monitoreo de entrega. Las aplicaciones normalmente corren RTP en la cima de UDP para hacer uso de su multiplexación y servicios de control; la implementación del protocolo RTP tiene como protocolo subyacente a UDP que a su vez trabaja sobre IP[24]. RTP debe ser usado con otros protocolos. RTP soporta transferencia de datos hacia múltiples destinos usando distribución multicast, siempre y cuando sea permitido este tipo de distribución por la red subyacente.

RTP por sí mismo no provee ningún mecanismo para asegurar la entrega oportuna o proveer otras garantías de calidad de servicio, pero se basa en servicios de capas inferiores para hacerlo. Esto no garantiza la entrega o la prevención de entregas de paquetes fuera de orden, ni asume que la red subyacente es fiable para entregar los paquetes en secuencia. Sin embargo, los números de secuencia incluidos en RTP permiten al receptor la reconstrucción de la secuencia de paquetes del emisor, pero los números de secuencia pueden también ser usados para determinar la ubicación apropiada de un paquete, por ejemplo en la decodificación del video, sin necesariamente decodificar los paquetes en secuencia. El transporte de datos es mejorado mediante un protocolo de control conocido como RTCP[25], El cual permite mejoras en el monitoreo de la entrega de datos en una manera escalable para redes multicast extensas, y provee un control y una funcionalidad de identificación.

[21] HTTP *Hypertext Transfer Protocol*
[22] HTML *HyperText Markup Language*
[23] RTP *Real Time Transport Protocol*
[24] IP *Internet Protocol*
[25] RTCP *Real time control protocol*

Mientras RTP es primariamente designado para satisfacer las necesidades de conferencias multimedia de múltiples participantes, no está limitado a solo este tipo de aplicaciones. El almacenamiento de datos continuos, la simulación distribuida interactiva y el control y medición de aplicaciones también pueden encontrar aplicable a RTP[9].
Un encabezado RTP típico, se presenta en la figura 5:

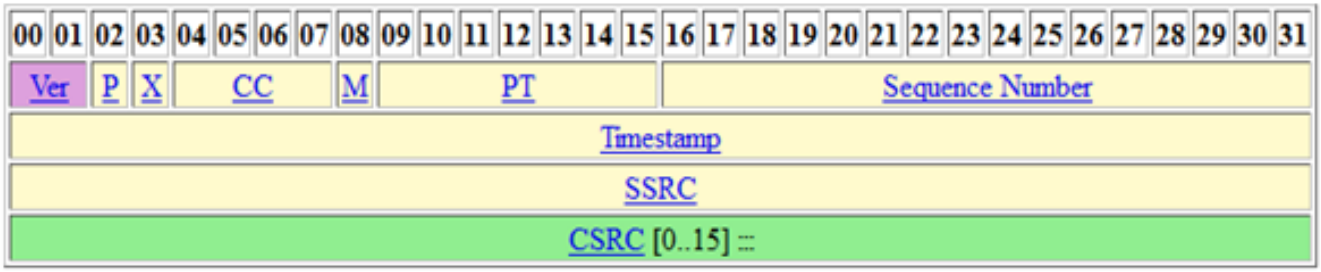

Figura 5: *Encabezado paquete RTP* Fuente: http://www.networksorcery.com/enp/protocol/rtp.htm

Donde:

- **Version:** Versión del protocolo.
- **Padding:** Sí el bit P es puesto, se indica que el paquete se ha rellenado aun múltiplo de 4 bytes. El último byte indica el número de bytes de relleno.
- **Extension:** Este bit X indica que hay un encabezado de extensión.
- **ContributingSourceCount:** Indica cuantos orígenes de contribución (CSRC) están presentes, si este bit es cero, entonces la fuente desincronización es seguida por la carga útil.
- **Marker:** Es un bit de marcador específico de la aplicación.
- **PayLoadType:** Indica cual algoritmo de codificación se ha utilizado para el formato de carga útil. Los tipos de carga útil se especifican en el RFC 3551.
- **SéquenceNumber:** Contador que se incrementa en cada paquete enviado.Se utiliza para detectar paquetes perdidos.
- **TimeStamp:** El origen del flujo produce la marca del tiempo para indicarcuándo se creó la primera muestra en el paquete.
- **SynchronisationSource:** Indica a cual flujo pertenece el paquete. Es el método utilizado para multiplexar y desmultiplexar varios flujos de datos en un solo flujo de paquetes UDP.
- **Content Source:** En caso de que hayan, se utilizan cuando los mezcladores están presentes en el estudio.

RTP en Conferencias de Audio y Video

Si tanto audio como video son usados en una conferencia, ellos son transmitidos en sesiones RTP separadas. Esto es, paquetes RTP y RTCP separados son transmitidos para cada tipo

de multimedia, usando dos diferentes pares de puertos UDP y/o direcciones multicast. Una de las razones de esta separación es la de permitir a los participantes en la conferencia el poder recibir solo un tipo de multimedia si así lo desean. A pesar de esta separación, la reproducción sincronizada de audio y video puede ser lograda usando información almacenada en paquetes RTCP para ambas sesiones.

2.5.2. RTSP

El protocolo RTSP es un protocolo de nivel de aplicación para la configuración y el control de la distribución de datos con propiedades de tiempo real, comúnmente en streaming. RTSP provee un framework extensible para permitir el control sobre la distribución bajo demanda de datos en tiempo real, como audio y video. Las fuentes de datos pueden incluir orígenes de datos en vivo o de datos previamente almacenados. Este protocolo tiene como objetivo el control de múltiples sesiones de distribución de datos, ofreciendo un medio para escoger canales de distribución como lo es UDP, UDP multicast y TCP, y provee un medio para escoger mecanismos de distribución basados en RTP. RTSP actúa como un control remoto de la red para servidores multimedia, similar a un control remoto de un reproductor de DVD. Además de ser usado para transmitir contenido multimedia en tiempo real, RTSP permite hacer un control directo en el cliente de la reproducción del flujo de datos enviado desde el servidor por medio de métodos propios del protocolo. Una ventaja marcada del uso de este protocolo para presentaciones multimedia con características de tiempo real es que el servidor puede utilizar un modo de operación unicast, multicast, e incluso broadcast[10].

Esados RTSP

RTSP controla diferentes flujos que pueden ser enviados por medio de un protocolo separado, independiente del canal de control. Por ejemplo el control RTSP puede ocurrir sobre una conexión TCP mientras los flujos de datos viajan vía UDP. Así mismo, durante el tiempo de vida, un único flujo de datos puede ser controlado mediante peticiones RTSP emitidas de forma secuencial sobre diferentes conexiones TCP. Por lo tanto, el servidor necesita mantener un "estado de sesión" para ser capaz de correlacionar las peticiones RTSP con los flujos. A continuación se definen los métodos que representan dichos estados en una sesión de streaming normal figura 6.

- **DESCRIBE:** Fase Inicial en la comunicación que obtiene la descripción del archivo solicitado. Es un método obligatorio debido a que los métodos subsiguientes dependen de él.
- **SETUP:** Este método cumple la función de hacer que el servidor reserve los recursos necesarios para comenzar la transmisión del flujo, además indica el protocolo a utilizar para esta transferencia (UDP, TCP, HTTP).
- **PLAY:** Este método informa al servidor que puede iniciar el envío de los flujos de datos. Para realizar una solicitud "PLAY" el cliente debe hacer primero una solicitud "SETUP".
- **PAUSE:** Este método interrumpe temporalmente el envío de datos por parte del servidor, sin embargo no libera los recursos reservados para dicha sesión.

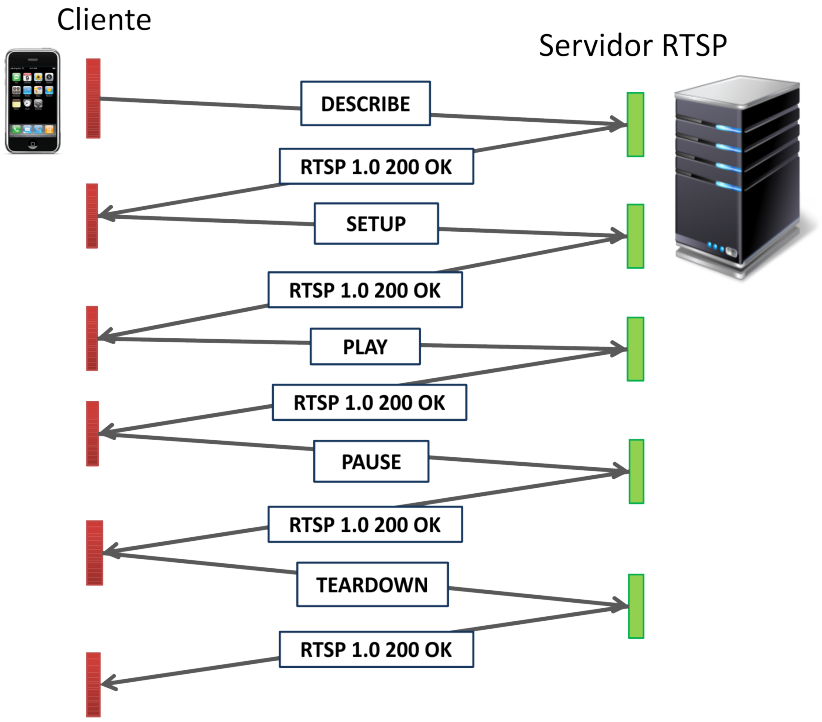

Figura 6: *Uso de los Métodos RTSP en una sesión streaming* Fuente: Autores del Proyecto

- **TEARDOWN:** Método utilizado para detener la transmisión de un recurso determinado, se liberan todos los recursos relativos a dicha comunicación, invalidando cualquier petición realizada sobre dicha sesión.

Mensajes RTSP

RTSP es un protocolo que utiliza el juego de caracteres de la ISO 10646 en la codificación Utf-8. Los mensajes pueden ser peticiones del cliente al servidor y respuestas del servidor al cliente. Las peticiones usualmente son de tipo textual y siempre implican una respuesta por parte del servidor basada en un código especifico. Ver Anexo A.5.

Un mensaje de petición RTSP está compuesto por tres partes: la línea de petición, la cabecera de petición y el cuerpo del mensaje. Un mensaje de respuesta RTSP está conformado también por tres partes: la línea de estado de la respuesta, la cabecera de la respuesta y el cuerpo del mensaje.

2.6. Java Micro Edition (JME)

Introducción a Java Micro Edition

Java Micro Edition (JME) es un subconjunto de JSE[26] orientado al desarrollo de aplicaciones Java destinadas a dispositivos con pocos recursos, con capacidades restringidas, tanto con

[26] JSE *Java Standard Edition*

respecto a la capacidad de memoria disponible, limitaciones de la pantalla gráfica, como con respecto a la capacidad de procesamiento; características típicas de cualquier equipo de electrónica de consumo; por ejemplo, teléfonos celulares, tabletas, etc.

Java Micro Edition no es exactamente una especificación, es un conjunto de especificaciones cada una de ellas aplicable a un conjunto de requisitos. JME no define un nuevo lenguaje sino que mantiene la compatibilidad hacia arriba con la edición JSE, adaptando la tecnología JAVA a su uso en dispositivos móviles. JME elimina partes de la edición JSE que no son aplicables a este tipo de dispositivos de capacidades limitadas, por ejemplo el AWT[27] o las operaciones en coma flotante. Por ello, una aplicación escrita para JME funcionará igualmente en la plataforma JSE, e incluso en la plataforma JEE[28], asumiendo que las clases que se usen están disponibles en cada edición[12].

Arquitectura JME

La arquitectura JME figura 7 se basa en familias y categorías de dispositivos. Una categoría define un tipo de dispositivo particular: teléfonos celulares, tabletas, etc. Una familia de dispositivos está compuesta por un grupo de categorías que tiene requisitos similares de memoria y capacidad de procesamiento.

Figura 7: *Arquitectura JME* Fuente: Autores del Proyecto

Configuración

La configuración es un mínimo grupo de APIs[29], útiles para desarrollar las aplicaciones destinadas a un amplio rango de dispositivos. La configuración estándar para los dispositivos inalámbricos es conocida como CLDC[30]. El CLDC proporciona un nivel mínimo de funcionalidades para desarrollar aplicaciones para un determinado conjunto de dispositivos inalámbricos. Se puede decir que CLDC es el conjunto de clases esenciales para construir aplicaciones[13].

[27] AWT *Abstract Windowing Toolkit*
[28] JEE *Java Enterprise Edition*
[29] API *Application Programming Interface*
[30] CLDC *Connected Limited Device Configuration*

Perfiles

En la arquitectura JME, por encima de la configuración, tenemos el perfil. El perfil es un grupo más específico de APIs, desde el punto de vista del dispositivo. Es decir, la configuración se ajusta a una familia de dispositivos, y el perfil se orienta hacia un grupo determinado de dispositivos dentro de dicha familia. EL perfil, añade funcionalidades adicionales a las proporcionadas por la configuración. La especificación más usada es MIDP[31], la cual describe un dispositivo MIDP como un dispositivo pequeño, de recursos limitados, móvil y con una conexión inalámbrica.

MMAPI

MMAPI es uno de los paquetes opcionales (ver figura 8) para JME que permite a los desarrolladores de aplicaciones móviles, el acceso a las capacidades multimedia del dispositivo siempre que este tenga soporte JAVA. El uso de este paquete permite incluir en aplicaciones móviles la reproducción y captura de diferentes formatos de archivos de audio y video, toma de fotografías a través de la cámara del dispositivo, entre otras utilidades.

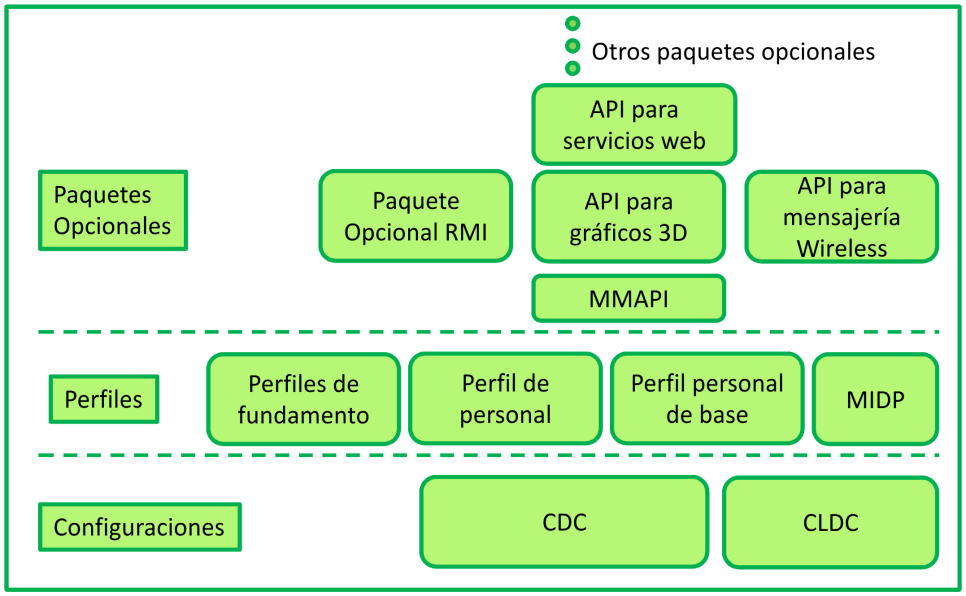

Figura 8: *MMAPI en el bloque de paquetes opcionales para JME* Fuente: Libro Pro Java Me MMAPI: Mobile Media API for Java Micro Edition

Arquitectura MMAPI

El procesamiento de datos multimedia se divide en dos partes:

- Obtener el contenido multimedia: Lectura de los datos multimedia desde una fuente como un archivo o un servidor.

[31] MIDP *Mobile Information Device Profile*

- Mostrar o reproducir este contenido: Análisis o decodificación de los datos multimedia para la visualización en el dispositivo.

Para realizar las operaciones antes mencionadas el MMAPI está constituido por tres paquetes:

Paquete Media: Define las interfaces y clases básicas para acceder y reproducir diferentes formatos de audio y video.

Paquete Control: Define las interfaces necesarias para el control especifico de las funciones asociadas a un determinado tipo de formato de audio o video.

Paquete Protocol: Define las interfaces necesarias para que los desarrolladores puedan crear reproductores propios para tipos de formatos que no estén recogidos en el API.

Además, el paquete MMAPI define otros dos elementos que llevan a cabo las posibles funcionalidades que ofrece el MMAPI:

Manager: Es un objeto que permite el control de los recursos de audio y video disponibles en el dispositivo.

Player: Los reproductores de tipo Player, son solicitados por las aplicaciones a través del objeto Manager para poder conocer las capacidades de audio y video del dispositivo. Además es el encargado de reproducir el contenido multimedia.

2.7. Códecs y formatos contenedores de video

El formato contenedor de un video es la forma en la que el video es guardado en un ordenador o dispositivo; sin embargo un Códec es el tipo de compresión que se ha utilizado para codificar dicho video digital en el archivo (en ese formato contenedor). Los codecs de video más populares son: DivX, Sorenson Video, SorensonSpark, Windows Media Encoder, Real Video, x264, xviD, FFmpeg, Nero Digital y Cinepak[14].

En el cuadro 1 se enuncian los formatos contenedores de video más populares.

Formato	Descripción	Códec Audio	Códec Video
.3gp	3GPP Multimedia File	AMR (-NB, -WB, -WB+	H.263
.asf	Advanced Systems Format File	WMA, MP3	WMV
.avi	Audio Video Interleave File	WAV	MPEG, QPEG
.mov	Apple QuickTime Movie[15]	Flia. AMR, WavPack, WMA	H.263, H.264
.mp4	MPEG-2 Video File	m4a, m4b	H.263, AMR, TX3G
.mpeg	MPEG Movie	MP3, MP2, MP1	MPEG-1 Part 2

Cuadro 1: *Formatos de video más usados.* Fuente: Autores del proyecto, basados en www.fileinfo.com

3. PLANTEAMIENTO DEL PROBLEMA

El constante avance de la tecnología en dispositivos móviles ha permitido un intenso estudio y una ardua investigación en temas relacionados. La sociedad actual dispone de herramientas que mejoran su estilo de vida, permitiendo esencialmente realizar un aumento favorable en la calidad y capacidad de comunicación.

Añadiendo, la comunicación no está simplemente limitada a la manera tradicional, por el contrario, esta se extiende en diferentes maneras como mensajes de texto, mensajes multimedia incluso hasta video llamadas, dando nuevas posibilidades de aprendizaje y métodos más didácticos y eficientes para entregar la información. Sin embargo la necesidad de mejorar estos servicios y la interacción social entre las personas es un factor importante en el estudio de las telecomunicaciones.

En la actualidad tanto emisor como receptor tienen la tecnología para compartir y recibir contenido multimedia usando dispositivos móviles, siendo esto tan importante, el valor agregado es el uso y optimización de los recursos.

A pesar de estos avances, se hace necesaria la creación de una herramienta que permita mejorar la comunicación en escenarios como aulas de clase, salas de conferencias y reuniones, entre otras, permitiendo la distribución de elementos multimedia como audio y video.

Es por esta razón que la investigación realizada propone un prototipo que permita la entrega de este contenido multimedia, para ser consultado remotamente, utilizando la técnica de streaming para dispositivos móviles, contribuyendo el aprovechamiento de la tecnología en situaciones como las anteriormente mencionadas. A lo largo del documento cada vez que se hable de dispositivos móviles se está haciendo referencia a celulares reales y emuladores, a menos que se especifíque lo contrário.

Este capítulo contiene las especificaciones de los problemas encontrados durante el proceso investigativo, dividido en dos secciones referentes al módulo de carga y al módulo de descarga respectivamente.

3.1. Compartir un video, Módulo de Carga

La idea de compartir un video desde un dispositivo móvil implica pensar en los recursos informáticos necesarios que se utilizarán en el proceso. En esta sección se analizan los factores y recursos utilizados para realizar la tarea de carga de un video que un usuario desea publicar en un servidor especializado.

Para compartir un video entre los usuarios del sistema, el dispositivo móvil debe tener instalada la aplicación, debe contar con conexión a internet, ya sea por Wi-Fi o la red celular que el operador provea y finalmente el sistema debe estar soortado por un servidor especializado que recibirá y almacenará el archivo que se desea compartir.

En la figura 9 se observa un bosquejo de funcionamiento del sistema, en el que se contemplan las actividades que se realizan durante el proceso de subida, incluyendo los procedimientos posteriores referentes a la carga y la descarga mediante la técnica de streaming.

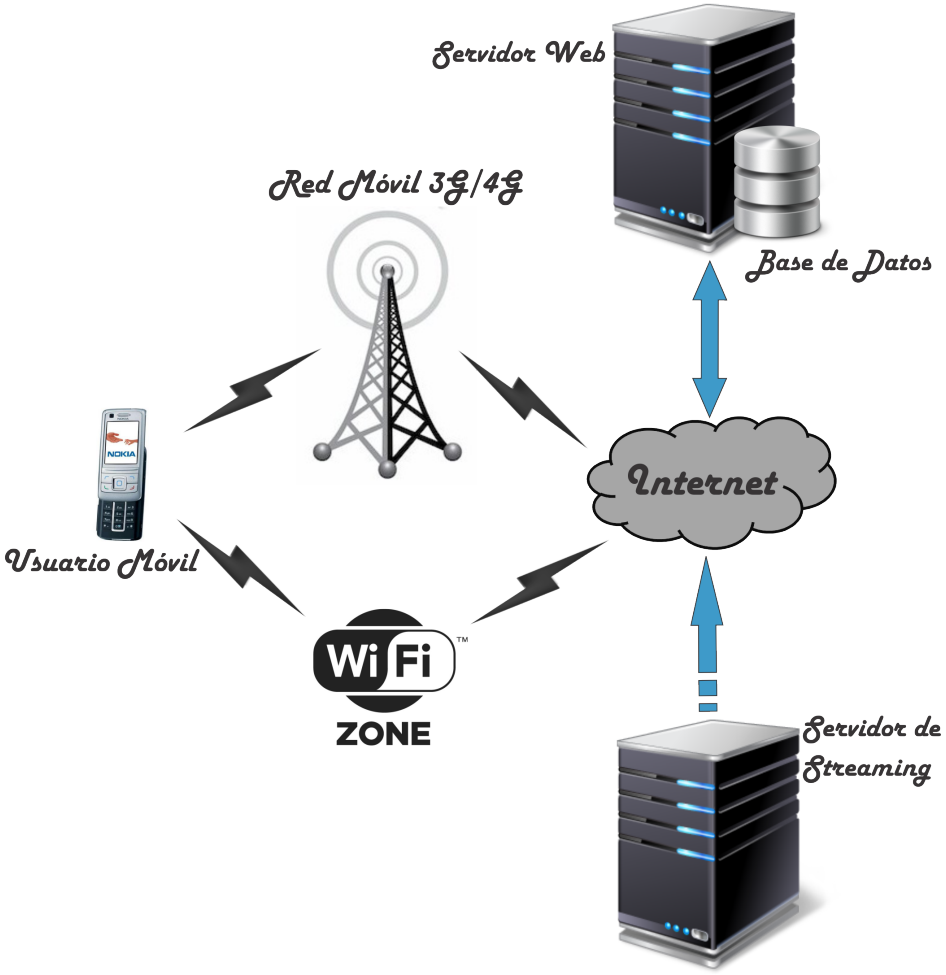

Figura 9: *Arquitectura General de Mobile Tv* Fuente: Autores del Proyecto

De la anterior figura realizaremos un análisis detallado de los recursos usados y las implementaciones adecuadas, para así documentar las dificultades encontradas, y en el siguiente capítulo de este documento presentar sus respectivas soluciones.

El proceso inicia con la selección del video almacenado previamente en la memoria del dispositivo o por el contrario con la creación y captura de uno nuevo. Teniendo en cuenta que los dispositivos móviles tienen capacidades de computación bastante reducidas, el prototipo cuenta con restricciones de uso para que se puedan realizar estos procesos exitosamente, por lo tanto se plantearon ciertas reglas del negocio para el sistema de carga. El video no tiene límite de tamaño si la aplicación se ejecuta en emuladores, mientras que en dispositivos físicos no debe superar un tamaño de 300 Kbits. Además en el momento de la búsqueda, debe encontrarse en la memoria externa del dispositivo, fuera de cualquier carpeta, para ser localizado por el prototipo.

El componente que se analiza a continuación es el servidor web.

Servidor Apache Tomcat

Apache Tomcat es un servidor web de código abierto que posee un contenedor de servlets desarrollado por la Apache Software Foundation (ASF). Tomcat implementa las especificaciones de Java Servlet y Java Server Pages (JPS) y provee un entorno de servicios web para ejecutar código basado en Java.

Apache Tomcat posee componentes especializados que se encargan de ejecutar diferentes tareas. Desde la versión Tomcat 4.x este servidor web cuenta con tres componentes fundamentales: Catalina (Contenedor Servlet), Coyote (Conector HTTP), y Jasper (Motor de JSP).

El Prototipo de Software Mobile Tv usa la versión 7.0.27, la última de Tomcat disponible desde Abril de 2012, la cual tiene un diseño modular bastante estable, dando seguridad y calidad de servicio, tanto al cliente como al administrador. El servidor se encuentra instalado en un Sistema Operativo Ubuntu pues es multiplataforma, donde se encuentran almacenados los servlets referentes al prototipo.

Servlets

Catalina es el contenedor de Servlets de Tomcat. Este contenedor posee dos grandes servlets que son usados por el prototipo planteado, el manejador de la base de datos y el responsable de la carga de archivos multimedia desde el dispositivo móvil respectivamente.

Servlet de Carga

Para cargar un archivo desde el dispositivo móvil se crea una conexión Http basados en la interfaz de java que define los métodos y constantes de una conexión Http.

Por medio de este protocolo, el cual se basa en la Petición-Respuesta, se crean parámetros que son enviados al servidor, interpretados por el servlet y finalmente procesados en una respuesta de confirmación.

En el servlet de carga se realizan dos procedimientos muy importantes, éstos son: el almacenamiento de los datos enviados, y el *hinting*, proceso del cual hablaremos más adelante.

Durante el almacenamiento se hallaron algunos inconvenientes como errores durante la conexión, los cuales se interpretaban por medio de los mensajes de respuesta que arrojaba el servidor, también errores en la inconsistencia de los archivos, comparando el original con la copia creada en el servidor.

En este punto de quiebre, se encontraron dificultades en la lectura del archivo que se pretendía cargar al servidor, pues el planteamiento proponía realizar una lectura byte por byte por medio de una conexión Http. Se observó que el proceso era más lento cuando el archivo a subir sobrepasaba los 10 Mega Bytes, y por lo tanto no brindaba la suficiente satisfacción al usuario.

Servlet Manejador de Base de Datos
Paralelamente al trabajo realizado sobre el servidor web Tomcat, se realiza una conexión con una base de datos.

MySQL es la base de datos de código abierto más popular del mundo. Es un sistema de base de datos relacional que se ejecuta en un servidor que proporciona acceso de múltiples usuarios simultáneamente.

Al ser una base de datos de código abierto es utilizada en algunos de los sitios web más visitados en Internet, incluyendo Flickr, Nokia.com, YouTube, la Wikipedia, Google, Facebook y Twitter.

MySQL viene con una suite de herramientas de línea de comandos para tareas tales como la consulta de la base de datos, realizar copias de seguridad, inspección de estado, la realización de tareas comunes, tales como la creación de una base de datos, y muchas más.[16]

La base de datos creada para el proyecto está compuesta por una única tabla de la que es posible obtener la información correspondiente al video almacenado en el servidor. Esta tabla tiene 5 campos con información de cada video, los campos se observan en el cuadro 2.

Video	
CP	idVideo
CF	nameVideo
CF	authorVideo
O	descriptionVideo
O	urlVideo

Cuadro 2: *Tabla de Video Base de datos Mobile Tv.* Fuente: Autores del proyecto

Para el proyecto Mobile Tv se ha instalado la versión 5.5 de MySQL sobre un Sistema Operativo Ubuntu, es accedida remotamente usando el protocolo HTTP apoyada en un Servlet que realiza procesos CRU (*Create - Read - Update*) creado por los autores del proyecto.

En esta etapa del proyecto no se encontraron dificultades relevantes que documentar, pues el proceso de implementación se llevó a cabo satisfactoriamente.

3.2. Observar un video, Módulo de Descarga

El valor agregado de esta investigación se encuentra en el módulo de descarga, pues la implementación de los protocolos de streaming como RTP y RTSP para visualizar un video, o cualquier contenido multimedia en dispositivos móviles es un objetivo importante a obtener. Observando nuevamente la figura 9 de la sección anterior, se analízan los elementos que faltan.

Streaming implementando protocolos RTP y RTSP

Para la descarga del archivo, se planteó realizar la implementación de los protocolos RTP y RTSP en la aplicación utilizando Java Micro Edition (JME), ya que el servidor streaming DSS utiliza estos protocolos para el envío de datos e información a través de paquetes RTP.

Basados en la documentación de RTP Transport Protocol for Real-Time Applications [17] RTP Protocolo de Transporte para Aplicaciones en Tiempo Real, se encontraron especificaciones correspondientes al tema y el conjunto de normas y modo del protocolo.

Paralelamente se realizó un análisis de la documentación presentada en la Comunidad de Desarrolladores Java (JCP) [18] en la cual Goyal Vikram un experto en el tema de multimedia en dispositivos móviles presenta una posible solución en la implementación de éstos protocolos.

Clases RTP y RTSP

Como anteriormente se mencionó, cada sesión de streaming consiste en que el servidor DSS envía paquetes a su destino basados en una petición previamente realizada.

Por lo tanto se realizó una clase para abstraer objetos con los atributos de un paquete RTP, para ello, se realizó un análisis del formato del paquete, observando la figura 5 de este documento.

Posteriormente, se recurre a una clase abstracta del API de Multimedia de JME, usando la clase DataSource, con la cual se encapsulan las tareas de localización y recuperación de datos. La recuperación de los datos es un proceso, pero el uso de ellos es otro muy diferente, por lo que se hace necesario continuar con la creación de otras clases que se encargan de resolver este asunto.

Internamente cada DataSource o fuente de datos, lee streams individuales, comúnmente llamados flujos de datos. Estos datos son leídos por una clase que lee paquetes RTP directamente, apoyada en la interfaz SourceStream de Java.

Hasta ahora se ha estudiado la creación y recepción de los paquetes RTP en la sección del dispositivo móvil, a continuación se analizan las peticiones y solicitudes realizadas al servidor.

RTSP es el protocolo sobre el cual los comandos de streaming son iniciados. Para simplificar la implementación, en la figura 10, se observa una sesión RTSP típica entre un cliente y un servidor de streaming.

Se realizó una clase controladora que realiza las peticiones y procesa las respuestas por parte del servidor. Esta clase da control al usuario desde que se realiza la primera solicitud al servidor, presentando la información que es necesaria para realizar los procesos referentes al *Describe*, *Setup*, *Play* y *Teardown*.

En conclusión el sistema de transferencia de paquetes funciona de esta manera: El constructor

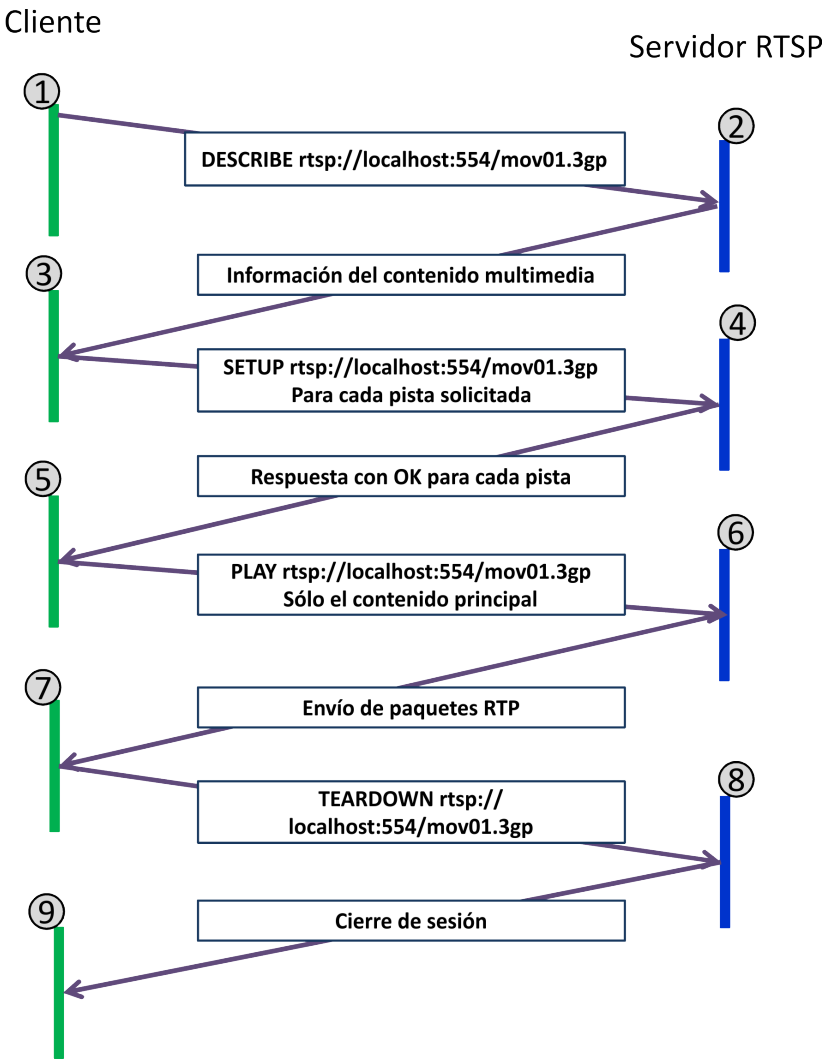

Figura 10: *Sesión típica del Protocolo RTSP para el prototipo Mobile Tv entre Cliente Móvil y DSS*
Fuente: Autores del Proyecto, basados en Vikram Goyal Documentation[18]

de la clase RTP SourceStream crea una conexión por Sockets con el servidor remoto, a continuación, se abre la entrada y los flujos de salida, que se utiliza para crear el manejador RTSP. Por último, con este controlador, se envían los comandos Describe y Setup al servidor remoto para hacer que el servidor quede listo para enviar los paquetes. La entrega real no comienza hasta que el método Start es llamado, lo que abre un puerto local (el caso de Mobile Tv es el 8081) para la recepción de los paquetes y se envía el comando Play para empezar a recibir estos paquetes. La lectura real de los paquetes se realiza en el método de ReadRTP, que recibe los paquetes individuales, que son interpretados en la clase RTP y devuelven los datos de PayLoad o Carga Útil para ser tratados individualmente en un último proceso. Es importante resaltar que el proceso anteriormente descrito, fue implementado y probado satisfactoriamente por los autores del proyecto.

La dificultad del proceso se encuentra en este punto, los datos transferidos son separados por el tipo de carga útil, sólo dos casos se presentan en el proceso: Carga útil de audio y carga útil de video.

Cada paquete de audio se trató de manera especial en una clase que adapta un array de bytes convirtiéndolos en un formato reproducible y reconocible por el dispositivo móvil. El códec de audio es especificado y empaquetado por el formato que es transferido por el protocolo RTP, en este caso se realiza una solicitud sólo de archivos en formato .3gp, según el cuadro 3 el códec de audio transportado es AMR, por lo tanto esta clase desempaqueta estos datos nuevos los cuales son interpretados directamente por el reproductor del API de multimedia.

Formato	**Códec Audio**	**Códec Video**
3gp	AMR	H.263
mp4	AAC	H.264

Cuadro 3: *Formatos contenedores disponibles para realizar streaming mediante RTSP.* Fuente: Autores del Proyecto

Caso contrario se presenta para el paquete de video, pues la carga útil transportada en el formato .3gp analizando el cuadro 3 está códificada bajo H.263. Por lo que se ensayó con la creación de una clase que pudiera codificar estos datos a un formato reproducible pero los resultados no fueron satisfactorios, esto debido a que los datos recibidos no incluyen las cabeceras establecidas por los estandares de compresión que rigen a los formatos reproducibles por los reproductores comunes.

Se propuso usar un componente software basado en Java, Xuggler[19], es un proyecto que comprime, descomprime y codifica cualquier formato de video en otro deseado. El proyecto Xuggler está basado en un API[32] de Java que no está soportado por Java ME. Por lo tanto, los esfuerzos por codificar la carga útil y obtener un formato reproducible dentro del dispositivo móvil fueron fallidos.

[32]API Application Programming Interface

4. PLANTEAMIENTO DE LA SOLUCIÓN

En este capítulo se presentan las soluciones a las dificultades encontradas durante el desarrollo del proyecto. Por lo tanto basados nuevamente en la figura de la arquitectura del prototipo Mobile Tv (Ver figura 9), se analizan con mayor detalle: los formatos de video para realizar streaming, servidores sespecializados, el proceso *hinting* que se realiza sobre los videos cargados y el uso del API de Multimedia ofrecido por la JCP y la manera en que todo esto aporta para la solución definitiva.

4.1. Darwin Streaming Server

Darwin Streaming Server (DSS) es la versión de código abierto de Apple QuickTime Streaming Server, que permite la transmisión de archivos multimedia vía streaming basados en los protocolos RTP y RTSP. Esta versión proporciona un alto nivel de personalización y se ejecuta en diversas plataformas, permitiendo manipular el código para satisfacer las necesidades del proyecto. Se encuentra instalada la versión DSS 6.0.3, sobre un Sistema Operativo Ubuntu.

Se ha escogido este servidor entre otros porque posee características especiales que coinciden con nuestros requerimientos como:

- Trabajar como servidor de archivos MP3, MP4, MOV y 3GP *hinted* (Ver sección 4.3 de este documento).
- Robustez, pues es la versión más estable de las usadas en las pruebas, manteniendo la fidelidad de los archivos durante la entrega.
- Simplicidad, no necesita ningún tipo de configuración adicional por parte del cliente.
- Protocolos soportados:

	UDP	**TCP**	**HTTP**
RTSP	x	x	x (Tunneled)*
RTP	x	x	x (Tunneled)*

Cuadro 4: *Protocolos soportados por servidor DSS.* Fuente: Autores del Proyecto *Tunneled: Redes IP privadas

4.2. Formatos de Video

Al seleccionar el formato de video adecuado para la transmisión fue necesario realizar un análisis detallado, evaluando la calidad de video, el tamaño de los archivos a transmitir y los formatos reconocidos por los dispositivos móviles.

Criterios de selección:

Durante la investigación fue necesario realizar el proceso de selección de los formatos de video y audio para realizar el streaming, en la evaluación se tuvieron en cuenta los siguientes aspectos:

- Formatos soportados por el servidor DSS: El servidor DSS, sólo realiza streaming de videos que se encuentran comprimidos y hinted. Darwin soporta una variedad de códecs de video diferentes y tres formatos de video para realizar streaming, estos son:

Formato	**Códec de video soportados por DSS**
.mov	Cinepak, H.261, H.263, MPEG-4 Video, Sorenson Video, Sorenson Video 3, Video
.mp4	MPEG-4 Video
.3gp	MPEG-4 Video

Cuadro 5: *Formatos soportados por Darwin Streaming Server.[21]* Fuente: Autores del Proyecto

- Formatos reconocidos por los dispositivos: Los dispositivos móviles no reconocen todo tipo de formatos, por lo tanto, debemos partir de los disponibles usados por el servidor y realizar una intersección de los más comunes que son usados por dispositivos Nokia, Samsung y Motorola.

  Al realizar un análisis sobre los dispositivos más comunes en el mercado de las diferentes compañías, se encontró que los formatos más comunes soportados por los dispositivos móviles son .mp4 y .3gp, con códec de h.263, h.264 y mpeg-4. Descartando el formato .mov, pues sólo es soportado por dispositivos Apple.

- Calidad y tamaño de Video: Este importante punto de comparación marca una gran diferencia entre el formato y el códec a utilizar durante el proyecto, pues no sólo es necesario tener una buena calidad de video, sino también un tamaño aceptable para poder realizar el streaming, al ser la relación directamente proporcional entre calidad y tamaño de video.

  La siguiente tabla presenta una comparación de formatos y códecs de video analizando los factores de tamaño y calidad.

	H.263		H.264		MPEG-4	
	Calidad	Tamaño	Calidad	Tamaño	Calidad	Tamaño
.3gp	Baja	Baja	Media	Medio	–	–
.mp4	–	–	Media	Medio	Alta	Medio

Cuadro 6: *Comparación de formatos y códecs en cuanto a tamaño y calidad de video* Fuente: Autores del Proyecto

Con base en los aspectos anteriores se ha encontrado que el formato mejor adaptado a los requerimientos del sistema es el .mp4. Dando al usuario calidad de video con un tamaño óptimo para el streaming.

4.3. Hint track

El proceso Hint, traducido como -Indicaciones de la Pista-, es el proceso realizado sobre un archivo multimedia en el cual se adicionan nuevas instrucciones a la pista, permitiendo a los servidores de streaming dividir el contenido total en paquetes que pueden ser enviados por el protocolo de transporte específico. El servidor DSS interpreta en cada archivo multimedia la longitud máxima de bytes y los milisegundos de audio que pueden ser transmitidos por cada paquete, haciendo en este caso, uso del protocolo RTP.

El proceso de Hint se realiza usando componentes software especializadas como lo son el QuickTime Pro y MPEG4IP, los cuales analizaremos a continuación:

QuickTime

QuickTime Player (QT) es un reproductor multimedia. Se puede utilizar para ver muchos tipos de archivos, incluyendo video, audio, imágenes fijas, gráficos, y películas de realidad virtual (VR). QuickTime es compatible con los formatos más populares de Internet para noticias, deportes, educación, trailers de películas y otros elementos multimedia.

QuickTime Pro

QuickTime Pro (QT Pro) es la versión pagada de QuickTime Player en la cual se añaden características útiles como:

- Guardar archivos de Internet.
- Edición de audio y video.
- Añadir efectos especiales
- Crear presentaciones de diapositivas.
- Convertir y guardar vídeo, audio e imágenes en múltiples formatos estándar.

Exportar archivos con QT Pro

Únicamente usando QT Pro se puede exportar una película en diferentes tipos de formatos, entre ellos QuickTime Movie (MOV), MPEG-4, 3GPP, 3GPP2, AVI y DV. En la figura 11 se observan algunos formatos de la opción de Exportar de QT Pro.

Cada uno de los formatos mencionados anteriormente pueden ser seleccionados entre la lista de opciones de exportación. Adicionalmente, se pueden personalizar éstos ajustes, para tener mayor control de la calidad y tamaño de la exportación. Este proceso optimiza el video para ser transmitido por el servidor realizando internamente el proceso Hint, por lo tanto el DSS reconoce los patrones de codificación y puede realizar la entrega del archivo multimedia mediante el protocolo RTSP.

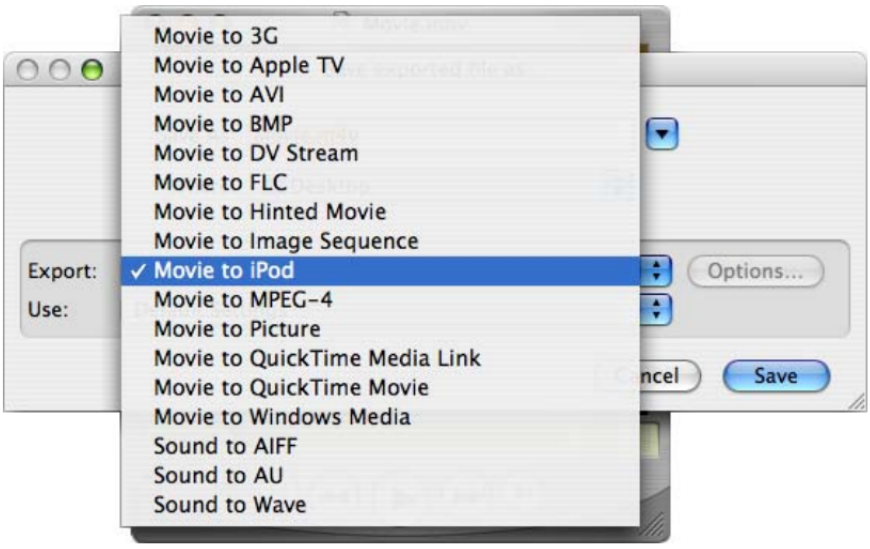

Figura 11: *Opciones Exportación QuickTime Pro*
Fuente: http://images.apple.com/quicktime/pdf/QuickTime7_User_Guide.pdf

MPEG4IP

El proyecto MPEG4IP se ha creado recientemente para ofrecer una herramienta de código abierto basada en estándares para la codificación, transmisión, reproducción e incluso la difusión de audio y vídeo codificado.

La plataforma principal de apoyo de MPEG4IP es Linux, pero ya que el código fuente está disponible, cualquier compilador de lenguaje C/C++ puede ejecutar la herramienta. Los usuarios de Windows cuentan con el apoyo completo de todas las características, pues se ha creado una edición especial para esta plataforma.

El programa está dividido en dos grandes módulos, los cuales se ejecutan en consola sobre un sistema operativo Windows.

El primero de ellos, es el módulo llamado mp4info.exe, se encarga de presentar la información más relevante de la composición del archivo multimedia consultado, presenta por separado los formatos de audio y de video, así como la duración en segundos, el tamaño y la información particular sobre el formato de Hint, detallando el tipo de carga útil de cada pista.[22]

El siguiente módulo es el mp4creator.exe, el cual tiene alrededor de 25 funciones, las más usadas de ellas durante la ejecución del proyecto fueron -extract, -hint, -optimize, y -versión entre otras.

En la figura 12 se pueden observar las opciones presentadas el componente MPEG4IP en ejecución, la versión instalada y las opciones del módulo mp4creator.exe.

Este proceso de Hint se debe aplicar a cada archivo que se desea transmitir, por lo tanto se ha creado un Serlvet que realiza este proceso. En el cual se instancia un objeto de la clase Java Runtime, para ejecutar un Shell que ejecuta la función de mp4creator.exe –hint, insertando en

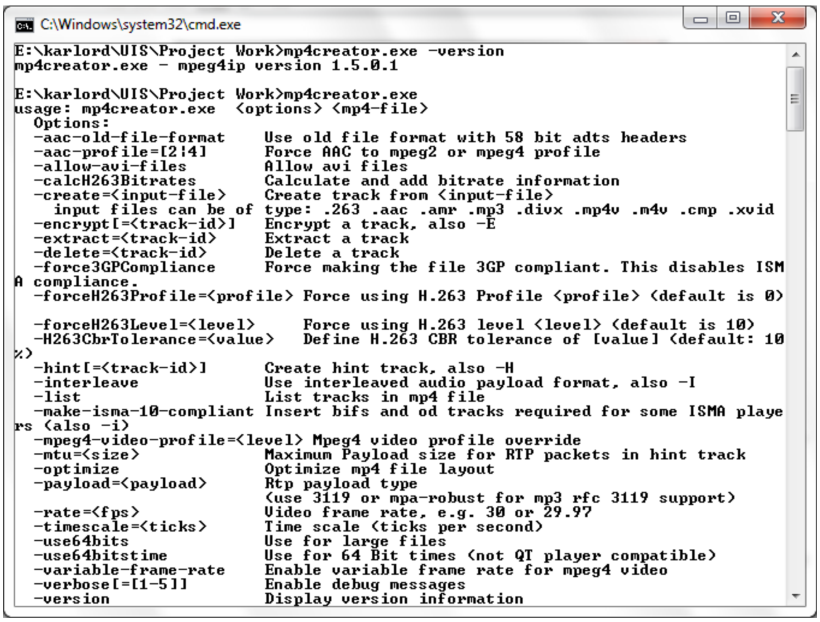

Figura 12: *MPEG4IP opciones sobre un Sistema Operativo Windows 7* Fuente: Autores del Proyecto

cada archivo la información que necesita el servidor para realizar el proceso de streaming.

En el cuadro comparativo 7, se realiza un análisis sobre la herramienta más optima para ser usada por el prototipo de software Mobile Tv.

	QuickTime Pro	MPEG4IP
Open Source	-	x
Sistema Operativo	Sólo MacOs y Windows	Multiplataforma
Realiza Proceso Hint	x	x
Funciona desde línea de comandos	No disponible	x

Cuadro 7: *Comparación entre QuickTime Pro y MPEG4IP*

La selección del componente software se hizo analizando los beneficios presentados anteriormente. Siendo MPEG4IP el componente mejor adaptado a nuestros requerimientos.

Servlet de Carga
Para cargar un archivo desde el dispositivo móvil se crea una conexión Http basados en la interfaz de java que define los métodos y constantes de una conexión Http.

4.4. API de Multimedia para Java Micro Edition MMAPI

La Java Comunity Procces (JCP) es el mecanismo estándar que desarrolla las especificadores técnicas de la tecnología Java. El trabajo de la JCP ayuda a dar estabilidad y compatibilidad

entre las distintas plataformas que soportan Java, lo que permite operar en diferentes dispositivos desde computadores de escritorio hasta robots industriales.

Con el respaldo de la JCP, en una de las Solicitudes de Especificaciones de Java (JSR por sus siglas en Inglés), se encontró la especificación JSR-135 [20] entre la documentación de Oracle, que especifica un API que se ocupa del espacio multimedia en Java Micro Edition (JME) lo que permite un acceso simple y fácil además del control de los recursos básicos de audio y multimedia, al mismo tiempo hacer frente a la escalabilidad y el apoyo de las características más sofisticadas para los dispositivos con plataformas Java.

- Nokia (Specification Lead)
- Aplix Corporation
- Beatnik, Inc.
- France Telecom
- Insignia Solutions
- Mitsubishi Electric Corp.
- Motorola
- Netdecisions Holdings United
- NTT DoCoMo, Inc.
- Openwave Systems Inc.
- PacketVideo Corporation
- Philips
- Siemens AG ICM MP TI
- Smart Fusion
- Sun Microsystems, Inc.
- Symbian Ltd
- Texas Instruments Inc.
- Vodafone
- Yamaha Corporation
- Zucotto Wireless

Información General del MMAPI

Conceptos Básicos: Protocolos y Manejo de Contenidos

Básicamente los procesos de multimedia pueden ser divididos en dos secciones.

1. Manejo del protocolo de entrega de datos.
2. Manejo del contenido de los datos.

Manejo de Protocolo se refiere a la lectura de los datos de una fuente (como un archivo, dispositivo de captura, o algún servidor de streaming) en un sistema de procesamiento de medios de comunicación.

Manipulaciónde contenido por lo general requiere procesar los datos de medios (análisis o decodificación, por ejemplo). Asimismo, como el uso de los recursos electrónicos para la reproducción del contenido.

Dos objetos de alto nivel se utilizan en esta API: *DataSource* y el *Player*. Cada objeto encapsula las dos partes de procesamiento multimedia:

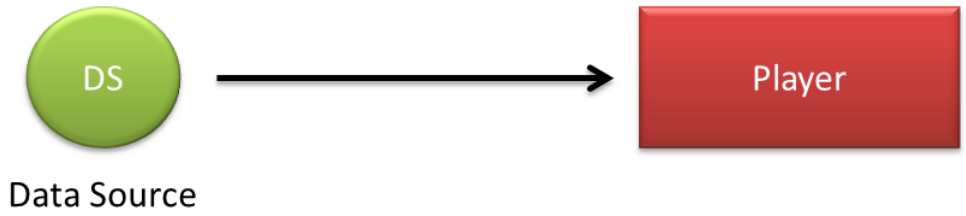

Figura 13: *Estructura de procesamiento mediante el API de Multimedia de Java Micro Edition* Fuente: Autores del Proyecto

1. DataSource para el manejo de los Protocolos.
2. Player para controlar el contenido.

Un DataSource encapsula el protocolo de manejo. Oculta los detalles de cómo se leen los datos de su fuente, si los datos provienen deun archivo o servidor de streaming, el DataSource proporciona un conjunto de métodos que permiten a un Player leer los datos para su procesamiento.

Un Player lee de la fuente de datos, procesa los datos, y hace que los medios de comunicación se reproduzcan en un dispositivo de salida. Proporciona un conjunto de métodos para controlar la reproducción de los medios de comunicación y sincronización básica.

Un mecanismo de fábrica, el Manager o Administrador, crea los Players para cada DataSource.

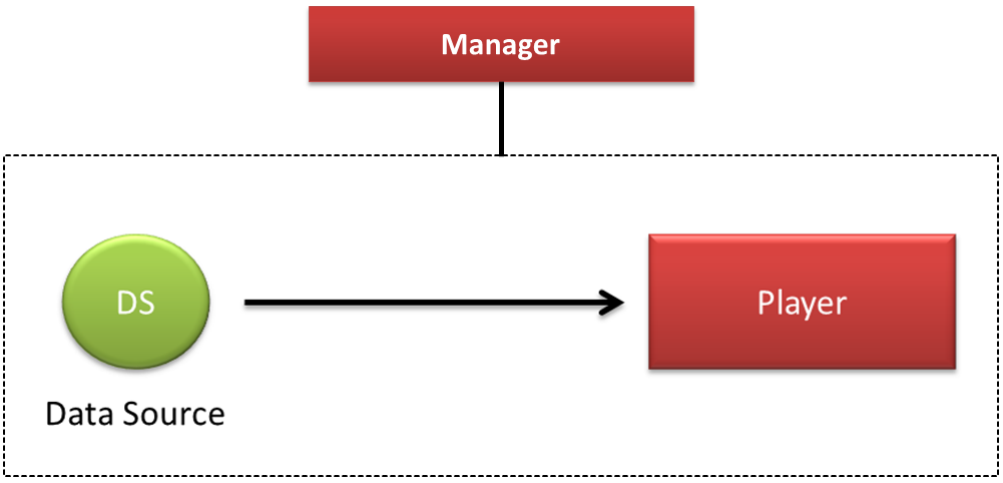

Figura 14: *Manager o Administrador del API de Multimedia de Java Micro Edition* Fuente: Autores del Proyecto

En el capítulo anterior se presentaron los problemas encontrados en cuanto a la reproducción de objetos de video, el API de Multimedia es una solución óptima en cuanto a la codificación de los paquetes de video.

Detalles de la Implementación

La implementación se realizó aplicando un escenario basado en la reproducción de un video usando streaming.

La conexión se realiza al servidor DSS (Darwin Servidor de Streaming) desde un dispositivo celular, se crea un objeto de la clase Player el cual se inicializa con un objeto Manager y el paso del parámetro de la dirección donde se encuentra el archivo consultado. El Manager provee el control necesario para que el video sea presentado en la pantalla, además de los controles de volumen y detener la reproducción.

El API de Multimedia hace uso de los protocolos de streaming RTP y RTSP sobre UDP, esto se descubrió producto del análisis de tráfico de red donde se pudo observar que el MMAPI realiza de la misma manera que la implementación presentada en el capítulo anterior, la división en paquetes RTP, cada uno con su distintiva carga útil, ya sea de video o de audio.

Según el grupo de expertos que hicieron parte del desarrollo del proyecto de MMAPI, sólo un conjunto reducido de proveedores de teléfonos y dispositivos móviles soportan esta implementación. Ya que este tipo de funciones están directamente relacionadas con la casa productora del dispositivo. Por lo tanto sólo algunas gamas de dispositivos reales Nokia soportan este API, mientras que en emuladores este número aumenta con los presentados por Oracle y Nokia, limitando la extensión del API de Multimedia a sólo unos pocos dispositivos.

Durante la implementación se establecieron los tipos y contenedores de archivos de video, con los cuales se realizaron las primeras pruebas, estos fueron contenedores .mp4 y .3gp, puesto que son los más comunes en el mercado y los más soportados por los dispositivos móviles. Las pruebas en emuladores resultaron exitosas, mientras que en dispositivos reales no se tuvieron los mismos resultados.

5. PRUEBAS Y ANÁLISIS DE RESULTADOS

Uno de los objetivos de esta investigación es realizar pruebas de funcionamiento en dispositivos físicos como emuladores, para comprobar el correcto funcionamiendo y el desempeño del prototipo haciendo uso de las redes inhalambricas. Para demostrar de esta forma, el potencial de los protocolos RTP y RTSP para la transmisión de medios por streaming.

5.1. Comparación de Huella Digital

Las comparaciones de la Huella Digital o MD5 han sido ampliamente utilizadas en el mundo del software para proporcionar garantía de que un archivo transferido ha llegado intacto al lugar de destino.

Para el prototipo Mobile Tv se realizaron pruebas de MD5 sobre los archivos que se compartieron, pues durante la transmisión desde el dispositivo celular al servidor donde son almacenados pueden ocurrir errores y verse modificados, de tal manera que los archivos en el servidor no sean los mismos que los archivos originales.

Por lo tanto esta comprobación se distribuyó en tres secciones: Extracción del MD5 del archivo previo a la carga, extracción del MD5 del archivo almacenado en el servidor después del proceso de carga, y finalmente la comparación de los dos MD5. Para realizar esta comparación se utilizó la herramienta software Md5Check, tal como se muestra en la figura 15. La cual permite buscar el archivo que queremos consultar, y calcular su huella o firma digital, en una casilla en blanco se puede pegar otra huella que es comparada con la actual, sí esta coincide, una sección en verde aprobará éstos dos MD5, de lo contrario en un color rojo responderá que existen diferencias en los códigos.

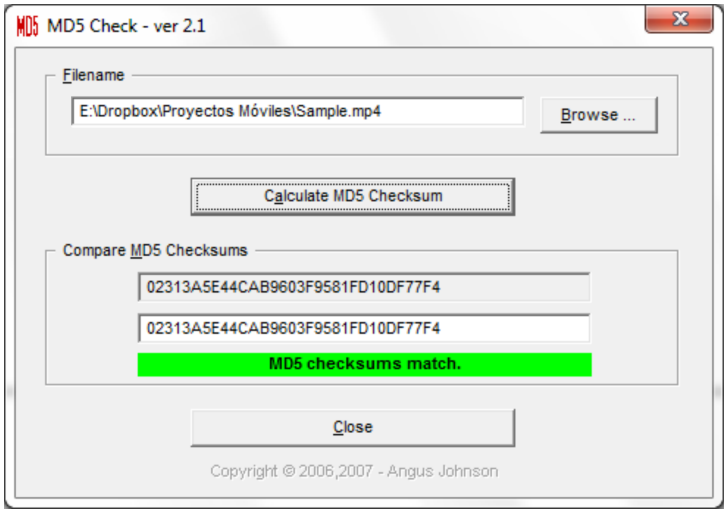

Figura 15: *Comprobación de MD5 con MD5Check*
Fuente: Autores del Proyecto

La especificación documenta que la dificultad de encontrar dos mensajes que tengan la misma huella o firma es del orden de 2^{64} operaciones, y que la dificultad de descifrar cualquier mensaje

dado está en el orden de 2^{128} operaciones.

En las tablas 8 y 9 se presentan la cantidad de errores y aciertos obtenidos en la comprobación de la huella digital de archivos cargados bajo dispositivos reales y emuladores. Para llevar a cabo estas pruebas, se utilizaron diferentes emuladores y dispositivos físicos. Esta prueba se realizó para archivos de formatos .3gp y .mp4 respectivamente.

Emulador	MD5 Antes	MD5 Después	Validación
Oracle Java (TM) Platform Micro Edition SDK 3.0.5	A0D5E7AD3D2CE49 39E892203BF7F69E0	A0D5E7AD3D2CE493 9E892203BF7F69E0	Exitosa
Nokia Symbian Belle SDK v 1.0	737C208512F42155E B96D53236354F43	737C208512F42155E B96D53236354F43	Exitosa
Samsung SDK 1.2.1	B3983C863F437C784 3E8C61A0211F68B		Fallida *
Sun Java Wireless Toolkit	230111940701703C7 782AAA89704D121		Fallida *

Cuadro 8: *Comparación MD5 de archivos compartidos desde emuladores.* Fuente: Autores del Proyecto.

Dispositivo	MD5 Antes	MD5 Después	Validación
Nokia C3	A0D5E7AD3D2CE493 9E892203BF7F69E0	446B9B629849F03B0 C77023279763BC2	Fallida **
Nokia X301	737C208512F42155E B96D53236354F43	89ADAC80690B1915E 26D60D6FE10EA1A	FAllida **
Samsung Galaxy Mini	B3983C863F437C784 3E8C61A0211F68B	BE0F72A81FDD8EEF0 2207CC19BF2AA1A	Fallída ***

Cuadro 9: *Comparación MD5 de archivos compartidos desde dispositivos físicos.* Fuente: Autores del Proyecto.

Convenciones:

* Proceso no terminado, el emulador no tiene soporte a la función de conexión HTTP. Por lo tanto no se pudo hacer la comparación de MD5.

** El archivo no es cargado satisfactoriamente, pues toma más de cinco minutos en realizar el proceso, además de la pérdida de bits durante la transmisión.

*** El archivo es cargado con éxito al servidor. Los MD5 no coinciden debido a un cambio en los útimos bits del archivo. A pesar de ello, el video queda disponible para ser consultado posteriormente

Estas pruebas de comprobación de MD5 sólo se realizaron en los archivos que fueron cargados al servidor (Módulo de Carga), pues en el Módulo de Descarga se realiza streaming y el archivo no queda almacenado en el dispositivo final y no es viable realizar el proceso.

5.2. Reproducción

Un sistema de streaming tiene real importancia en el momento en el que el cliente puede observar en un dispositivo final el archivo multimedia satisfactoriamente. El reproductor es parte fundamental, pues entre sus responsabilidades esta la interacción con el usuario a través de una interfaz gráfica, el manejo de errores en la trasmisión y la decodificación de los datos recibidos.

Para verificar la etapa de reproducción dentro del proceso de streaming, se realizaron pruebas en cuatro emuladores y cuatro dispositivos físicos diferentes. Como se ha dicho, la prueba será exitosa si se reproduce el archivo multimedia, y será fallida en el caso contrario. Para cada emulador/dispositivo se probó el streaming con un archivo .3gp y uno .mp4.

Emulador	Archivo mp4	Archivo 3gp
Oracle Java (TM) Platform Micro Edition SDK 3.0.5	Exitosa	Exitosa
Nokia Symbian Belle SDK v 1.0	Exitosa	Exitosa
Samsung SDK 1.2.1	Fallida *	Fallida *
Sun Java Wireless Toolkit	Fallida *	Fallida *

Cuadro 10: *Streaming de archivos multimedia en emuladores.* Fuente: Autores del Proyecto.

Dispositivo	Archivo mp4	Archivo 3gp
Nokia C3	Exitosa**	Exitosa**
Nokia X301	Exitosa**	Exitosa**
Samsung Galaxy Mini	Fallida *	Fallida *

Cuadro 11: *Streaming de archivos multimedia en dispositivos físicos.* Fuente: Autores del Proyecto.

Convenciones:

* El dispositivo móvil no tiene soporte de conexión por datagramas. Por lo tanto no se pudo hacer streaming.

** La reproducción es satisfactoria, presenta pausas pequeñas debido a la latencia entre el servidor y el dispositivo móvil.

5.3. Seguimiento de Paquetes RTP y RTSP

Las pruebas de seguimiento de paquetes RTP y RTSP se realizaron para tener la certeza de que el prototipo de software Mobile Tv usa estos protocolos para realizar el proceso de streaming.

Wireshark es un software analizador de protocolos de red, que permite capturar y navegar de forma interactiva en el tráfico de paquetes a través una red informática. Es un componente de software libre y multiplataforma.

La prueba se ha dividido en dos secciones a continuación descritas, en las que se analiza con la herramienta Software Wireshark los protocolos RTP y RTSP que usa el prototipo Mobile Tv para la transmisión con streaming.

5.3.1. Envío de paquetes RTP desde el servidor de streaming al emulador.

La primera prueba consiste en analizar, en el lado del servidor, si se hace uso de los protocolos RTP y RTSP en el proceso de transmisión, justo cuando se realiza la petición para observar un video almacenado en el servidor de streaming DSS [33].

La figura 16 presenta los detalles de la captura filtrada por Wireshark, desde un servidor Ubuntu 10.10, en la cual se observan los paquetes RTP transmitidos desde la dirección IP Origen (servidor DSS), a la dirección IP destino (emuladores ubicados en otra máquina). Corroborando de esta manera el uso de los protocolos propuestos en el planteamiento de la solución.

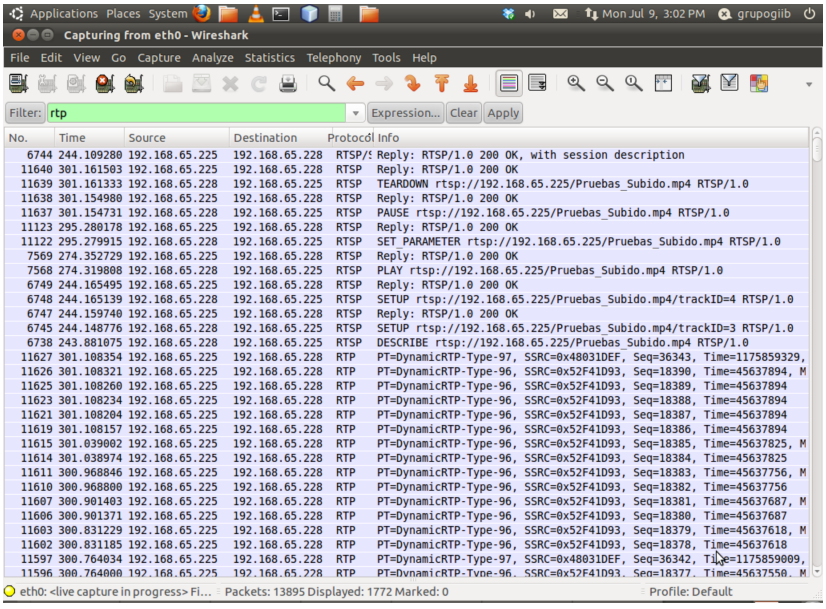

Figura 16: *Captura de paquetes RTP por Wireshark Ubuntu.*
Fuente: Autores del Proyecto

5.3.2. Recepción de paquetes RTP enviados por el servidor al emulador.

No sólo es suficiente demostrar que los paquetes enviados desde el servidor se encuentran empaquetados en el formato requerido, también es relevante asegurar la recepción de los mismos a su destino.

[33] DSS Darwin Streaming Server

En la figura 17 se analizan con detalle los paquetes recibidos por el emulador, que han sido enviados por el servidor DSS, con la herramienta Wireshark instalada en un sistema operativo Windows 7.

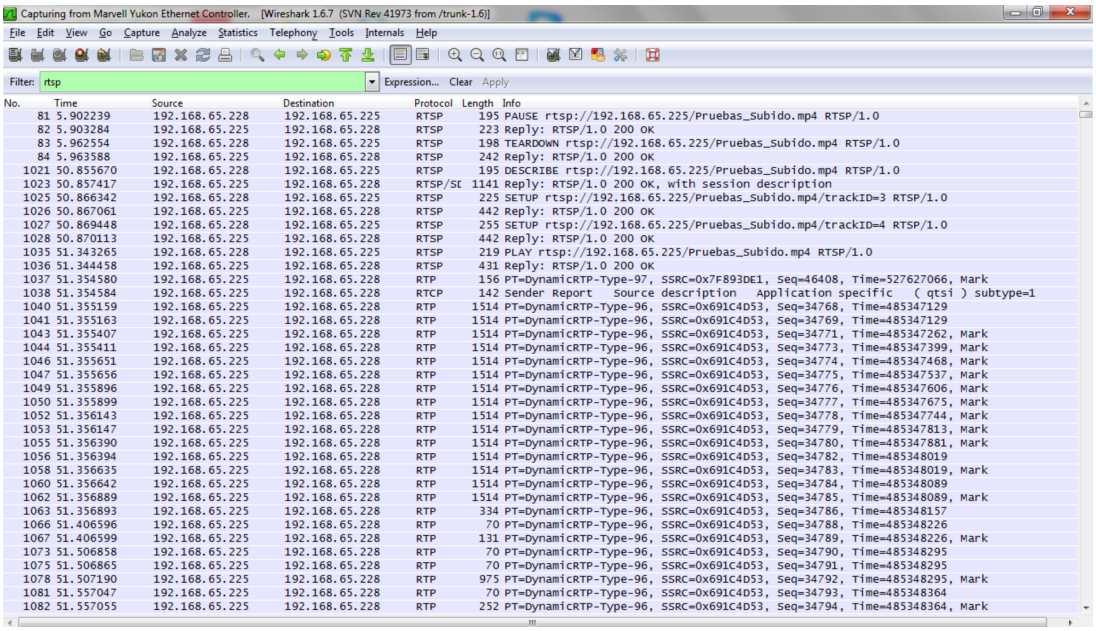

Figura 17: *Captura de paquetes RTP por Wireshark Windows.*
Fuente: Autores del Proyecto

5.4. Pruebas Unitarias.

De acuerdo con la metodología planteada, la realización de pruebas unitarias es un proceso necesario durante la fase inicial del desarrollo, que permite verificar el correcto funcionamiento de los métodos más importantes usados en el prototipo.

Para el proyecto Mobile Tv se utilizó un framework de Java Micro Edition para pruebas llamado JMUnit [34].

En la tabla 12 se listan las clases de pruebas creadas para verificar el funcionamiento de los métodos de conexión de base de datos, reproducción y cargado de video al servidor de streaming.

Las clases prueban los métodos haciendo uso de los *assertions* o *afirmaciones* del lenguaje Java. Estas afirmaciones se usan para verificar los resultados de una operación o un test. Las afirmaciones usadas en las pruebas del prototipo fueron: *assertEquals* y *assertNotNull*, entre otras.

En la figura 18 se observan los resultados de una prueba realizada en un emulador, en la cual, en la primera sección el servidor Tomcat está desactivado, y finalmente en la siguiente

[34] http://jmunit.sourceforge.net/

Clase de Prueba	Prueba realizada
CargaVideoTest()	Comprueba que existe conexión con el servidor Tomcat, la prueba falla sí la respuesta del servidor es diferente de OK.
ManejadorConexionTest()	Verifica sí la base de datos existe y está disponible realizando una solicitud al servidor donde se encuentra instalada.
ReproductorTest()	Comprueba que el servidor de streaming se encuentra disponible para disponible y se solicita una respuesta a una conexión de ejemplo.

Cuadro 12: *Clases de prueba en JMUnit.* Fuente: Autores del Proyecto.

sección la misma prueba con el servidor encendido y el resultado exitoso de la misma.

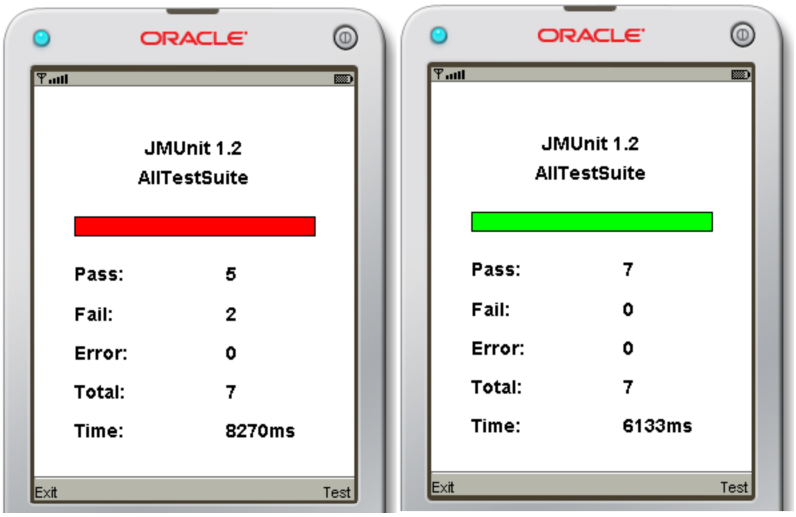

Figura 18: *Pruebas unitarias en emuladores Oracle.*
Fuente: Autores del Proyecto

5.4.1. Comparación de protocolos de transferencia, streaming con TCP y RTP.

Estas pruebas consisten en demostrar la verdadera efectividad y los alcances de los protocolos RTP y RTSP frente a TCP y HTTP, en cuanto al streaming de medios.

Por lo tanto se realizó una comparación entre los dos protocolos (TCP y RTP) en emuladores y dispositivos móviles, creando conexiones HTTP a diferentes archivos, evaluando en cada solicitud calidad de audio y video.

En los cuadros 13 y 14 se presentan los resultados obtenidos de estas simulaciones.

Convenciones:

Protocolo	Formato 3gp	Formato mp4
RTP	Exitosa	Exitosa
HTTP	Exitosa	Fallida *

Cuadro 13: *Pruebas de streaming en emulador Oracle* Fuente: Autores del Proyecto.

Protocolo	Formato 3gp	Formato mp4
RTP	Exitosa **	Exitosa **
HTTP	Fallida	Fallida

Cuadro 14: *Pruebas de streaming en Nokia C3* Fuente: Autores del Proyecto.

* Hubo pérdida total de los paquetes de audio, sólo se reproduce el video.

** La reproducción es satisfactoria, presenta pausas pequeñas debido a la latencia entre el servidor y el dispositivo móvil.

6. CONCLUSIONES

- Durante el proceso investigativo, se determinó la conveniencia de estudiar los protocolos RTP y RTSP que permiten llevar a cabo el proceso de streaming de un archivo multimedia en cualquier clase de dispositivo. A partir de ese estudio se decidió como alternativa inicial que era necesaria la implementación de dichos protocolos por parte de los autores para lograr desarrollar un prototipo funcional, robusto, integro y confiable que ofreciera servicios de entrega dedicada de audio y video orientados a dispositivos móviles. Durante el proceso de desarrollo, se llegó a la conclusión de que esta alternativa no era la mejor opción, ya que aunque se implementaron los dos protocolos de manera satisfactoria y los archivos se transferían desde el servidor hasta el cliente utilizando la técnica de streaming sin problemas , la imposibilidad para reproducir los paquetes de video del archivo multimedia en el cliente significó en su momento una dificultad importante para la finalización del proyecto y que el prototipo quedara desarrollado bajo esta alternativa inicial en un 80 %. Este problema se originó debido a que los reproductores que soportan el API de multimedia MMAPI de JAVA ME no tienen implementado un códec que interprete y reestructure la información que viene dentro de los paquetes RTP recibidos desde el servidor, ya que los datos recibidos no incluyen las cabeceras establecidas por los estándares de compresión que rigen a los formatos reproducibles por la mayoría de los reproductores.

- Este trabajo de investigación permitió experimentar el alcance de la entrega dedicada de audio y video orientada a dispositivos móviles, el intento inicial de desarrollar un prototipo que incluyera la implementación por parte de los autores del proyecto de los protocolos que permiten el proceso de streaming y el desarrollo final de un prototipo basado en el API de multimedia que ofrece el lenguaje de programación utilizado (JAVAME). El prototipo desarrollado le permite a los usuarios acceder a contenidos de video que estén disponibles en un servidor de streaming. El prototipo sirve como complemento a la aplicación M-learning desarrollada dentro del grupo de investigación de Ingeniería Biomédica (GIIB) en su línea de dispositivos móviles, permitiéndole a estudiantes y profesores además de compartir contenidos de texto, exámenes y notas, compartir videos con el propósito de romper con limitantes de distancia y tiempo entre personas, logrando que el maestro pueda crear videos de sus clases, y que estudiantes puedan acceder a dichos contenidos en cualquier momento y desde cualquier lugar regulándose por ciertas reglas del negocio.

- Además de lograr un prototipo que le ofrece al usuario el acceso a los contenidos de video, este trabajo se enfocó siempre en una relación directa entre dos personas, una que comparta el video que desee, y otra que acceda a este contenido desde cualquier lugar y en cualquier momento, acabando con estrictas necesidades de las relaciones interpersonales, como el que coincidan siempre en tiempo, espacio y disponibilidad. Este proceso se completó con dos componentes adicionales, un modulo de carga implementado en el prototipo, el cual le permite a los usuarios emisores cargar un video desde el dispositivo, y con un servlet que aplica satisfactoriamente un proceso Hinting, deja listo dicho archivo para que cualquier usuario receptor pueda acceder a él mediante la técnica de streaming. De manera adicional, el prototipo le ofrece al usuario la posibilidad de grabar y cargar un nuevo video, sin necesidad de otro programa o salirse de la aplicación. Esto mejora considerablemente la usabilidad del prototipo y comodidad del usuario, optimizando su experiencia con la aplicación.

- Por medio de las pruebas realizadas de verificación de envío y recepción de paquetes RTP se comprobó que los datos circulan por la red desde el servidor hasta el cliente aprovechando los servicios de entrega eficiente de datos que ofrece el protocolo RTP. Así mismo, las pruebas de verificación del flujo de peticiones y respuestas RTSP permitieron comprobar que desde el principio se está estableciendo una configuración controlada de las comunicaciones y un control sobre los paquetes RTP transferidos, añadiendo robustez y eficiencia al prototipo.

- Las pruebas realizadas que comparan el funcionamiento del Streaming de video entre el prototipo utilizando RTP y RTSP como protocolos base de Streaming, y el mismo prototipo utilizando HTTP y TCP para el mismo propósito, permiten corroborar que RTP es el protocolo que transmite más eficientemente datos bajo la técnica de streaming, ya que al ser dos protocolos no orientados a conexión, colocan una alta prioridad al envío de flujos de manera continua e ininterrumpida, en vez de colocar la atención exclusiva a la seguridad. Al ser necesario un reenvío de paquetes perdídos o dañados en el prototipo que funciona bajo TCP, la transmisión se interrumpía parcial o totalmente.

- La metodología agil de desarrollo extreme Programming y su técnica "Test First Programming", permitió sacar el mayor de los provechos a los procesos de planificación pruebas y desarrollo, ya que durante todo el trabajo se verificó el buen funcionamiento de los diferentes módulos del prototipo, siempre pensando en el cumplimiento de los objetivos propuestos. Además, el empleo específico de la técnica "Pair-Programming" dio lugar a una programación organizada y eficiente.

7. RECOMENDACIONES

Finalizado el proceso de desarrollo e investigación y de realizar una evaluación del producto terminado, el paso siguiente fue establecer recomendaciones y futuros alcances de este proyecto.

- Teniendo en cuenta los protocolos de transporte y de streaming, RTP y RTSP respectivamente, y la implementación de los mismos para Java Micro Edition. Se recomienda realizar una implementación al prototipo que permita tener el control de la reproducción, como lo es el botón de pausa, adelantar o retrasar. Esta opción refleja el verdadero alcance de estos protocolos, en los cuales el MultiMedia API (MMAPI) no está implementado, por lo tanto este no permite pausar la reproducción y continuar con el hilo, sino que es necesario reproducirla desde el inicio.

- Analizando los problemas encontrados para llevar a cabo la decodificación y reproducción de los paquetes recibidos en el proceso de streaming, después de haber realizado la implementación de los protocolos RTP y RTSP para el prototipo, es recomendable realizar en trabajos futuros la creación de un reproductor o player que haga el proceso de conversión y reproducción de las cargas útiles de audio y video simultáneamente, además de tener el control total de la cantidad y tamaño de los paquetes que deben transmitirse en una determinada solicitud. Pues debido a las razones documentadas anteriormente es de gran utilidad tener este tipo de control durante una sesión de streaming.

- El proceso Hinting para los archivos de video que se comparten, permite su posterior reproducción en dispositivos finales. Sin este proceso, se reducen las posibilidades de una exitosa reproducción en caso que sea realizado de manera errónea. Es recomendable, encontrar un servidor que permita realizar el proceso de streaming sin tener la necesidad de aplicar el hinting, esto aumentaría la calidad de uso, pues habría menos riesgo de fallas en la solicitud y disponibilidad del video.

- Es conocido que la tecnología móvil está ahora direccionada en los nuevos dispositivos con sistemas operativos Android, Ovi, Windows Phone y IOS entre otros, en los cuales el hardware del dispositivo es mejor aprovechado, aparte las variadas aplicaciones que se tienen en el mercado. Por lo tanto se sugiere extender esta implementación a la nueva tecnología de tablets y smartphones. La ventaja de la programación en los nuevos dispositivos móviles es que cuentan con máquinas virtuales de Java y pueden soportar aplicaciones tan robustas como la planteada en el proyecto, aparte de usar el mismo paradigma de programación orientado a objetos usado para el desarrollo de la aplicación móvil.

Referencias Bibliográficas

[1] Superintendencia de Industria y comercio. Disponible en http://www.sic.gov.co/oldest/archivo_descarga.php?idcategoria=23658 [Fecha consuta: Mayo de 2011]

[2] Proyecto de investigación "Diseño e implementación de una aplicación M-Learning para el acompañamiento de clases presenciales", Disponible en biblioteca Universidad Industrial de Santander. [Fecha consuta: Abril de 2011]

[3] Técnicas y pricncipios de computación Móvil. Disponible en: http://www.acis.org.co/archivosAcis/ComputacionMovilTEcnicasyPrincipios.pdf [Fecha consuta: Abril de 2011]

[4] Hands free Info Disponible en: http://handsfreeinfo.com/study-handheld-cell-bans-have-no-effect [Fecha consuta: Junio de 2011]

[5] Streaming media bible Disponible en: MACK, Steve. Streaming media bible. Hungry Minds, 2002. 869p. ISBN9780764536502. [Fecha consuta: Junio de 2011]

[6] Evaluación de servidores de streaming de video orientado a dispositivos móviles. Proyecto de grado de la universidad de Antioquia. [Fecha consuta: Junio de 2011]

[7] Pila de protocolos para el estándar 3GPP Disponible: http://www.medialab.sonera.fi/workspace/PacketSwitchedStreamingWP.pdf. [Fecha consuta: Junio de 2011]

[8] Streaming Media Disponible en: http://all-streaming-media.com/articles/Streaming-Media-Intro_Streaming-protocols.htm[Fecha consuta: Marzo de 2011]

[9] RFC Paquete RTP Disponible en: http://www.ietf.org/rfc/rfc3550.txt [Fecha consuta: Agosto de 2011]

[10] Protocolo de Streaming en Tiempo Real Disponible en: http://www.ietf.org/rfc/rfc2326.txt [Fecha consuta: Mayo de 2011]

[11] "Desarrollo de un sistema streaming de audio orientado a dispositivos móviles"Proyecto de Investigación desarrollado en la línea de investigación de Dispositivos móviles del GIIB. Disponible en biblioteca Universidad Industrial de Santander [Fecha consuta: Junio de 2011]

[12] Libro J2ME Java 2 Micro Edition – AgustinFroufe Quintas, Patricia Jorge Cardenas –Alfaomega, Ra-Ma. [Fecha consuta: Abril de 2011]

[13] Libro Introducción a J2ME – Manuel Prieto – Noviembre 2002 [Fecha consuta: Abril de 2011]

[14] Comparación de códecs y formatos de video Dsiponible en: http://en.wikipedia.org/wiki/Comparison_of_video_codecs [Fecha consuta: Junio de 2011]

[15] QuickTime File Format - Disponible en: http://en.wikipedia.org/wiki/QuickTime_File_Format [Fecha consuta: Enero de 2012]

[16] Documentación MySQL Disponible en: http://www.mysql.com/whymysql/ [Fecha consuta: Mayo de 2011]

[17] RFC RTP A Transport Protocol for Real-Time Applications Disponible en: http://www.ietf.org/rfc/rfc3550.txt [Fecha consuta: Mayo de 2011]

[18] Experiments in Streaming Content in Java ME Disponible en: http://today.java.net/pub/a/today/2006/08/22/experiments-in-streaming-java-me.html?force=542 [Fecha consuta: Septiembre de 2011]

[19] Xuggler Documentation Disponible en :http://www.xuggle.com/xuggler/ [Fecha consuta: Diciembre de 2011]

[20] Documentation JSR Multimedia API Java ME Disponible en: http://docs.oracle.com/javame/config/cldc/opt-pkgs/api/mm/jsr135/index.html [Fecha consuta: Julio de 2011]

[21] La compresión y hinitng de medios para la transmisión mediante Streaming Disponible en: http://soundscreen.com/streaming/compress_hint.html [Fecha consuta: Julio de 2011]

[22] Documentación y uso del componente software MPEG4IP Disponible en: http://mpeg4ip.sourceforge.net/documentation/MPEG4IP_Guide.pdf [Fecha consuta: Julio de 2011]

[23] Extreme Programming Disponible en: http://wiki.netbeans.org/Refactoring [Fecha cosulta: Enero 2011]

[24] Introduction of MVC structure in J2ME client Articulo de Moto coder Staff, Año de publicación Marzo de 2006 Disponible en http://markmail.org/download.xqy?id=yqka6wgrs5r4bc4h&number=2 [Fecha consulta Abril 2011]

Bibliografía

AGBINYA JHONSON I, IP Communications and Services for NGN, Auerbach Publications, 2009.

FERNÁNDEZ OVIEDO Estefanía, GÓMEZ PARRA Laura, Diseño e implementación de una aplicación M-Learning para el acompañamiento de clases presenciales, Universidad Industrial de Santander, 2010.

FROUFE QUINTAS AGUSTIN J2ME Java 2 Micro Edition. Patricia Jorge Cardenas Alfaomega, Ra-Ma, 2006.

GOYAL VIKRAM, Experiments in Streaming Content in Java ME, 2006. Consulta: http://today.java.net/2006/08/22/experiments-in-streaming-java-me.html?force=542#-rtsp-rtp-rtcp, 2006.

JSR DOCUMENTACIÓN , Multimedia API Java ME Consulta: http://soundscreen.com/streaming/compress_hint.html, 2009.

MACK STEVE. Streaming media bible. Hungry Minds, 2002, 869p.

MOTO CODER STAFF, Introduction of MVC structure in J2ME client, 2006.

NETWORK WORKING GROUP, Real Time Streaming Protocol (RTSP), Columbia University, 1998, 28p.

NETWORK WORKING GROUP, RTP: A Transport Protocol for Real-Time Applications, Columbia University, 2003, 35p.

O'DOCHERTY MIKE, Object-Oriented Analysis and Design - Understanding System Development with UML 2.0, John Wiley & Sons Ltd, 2005, 438P.

PRIETO MANUEL, Libro Introducción a J2ME, Universidad Autónoma de México, 2002.

QUINTERO ORTIZ Juan Pablo and CASTRO SERNA Cristian Andrey, Evaluación de Servidores de Streaming de Video Orientado a Dispositivos Móviles, Medellín: Universidad de Antioquia, 2006.

WAKE WILLIAM C. Extreme Programming Explored, 2000.

A. Anexos

A.1. METODOLOGÍA DE DESARROLLO

Extreme Programming (XP) es un metodología ágil para el desarrollo de software. XP se caracteriza por hacer hincapié en la satisfacción del cliente, la realimentación iterativa y el trabajo en equipo para maximizar el valor de los resultados basándose en pilares como el acompañamiento del cliente, enfoque de planeación y realización de pruebas constantes.

Esta metodología faculta a los desarrolladores para responder con seguridad a las diversas necesidades cambiantes de los clientes, incluso a finales del ciclo iterativo. Se basa en el trabajo en equipo, el cual se auto-organiza en torno al problema para solucionarlo lo más eficientemente posible.

XP mejora el proceso de software en cinco formas esenciales, la comunicación, la simplicidad, la realimentación, el respeto y valor. Durante el proceso investigativo fue necesario adaptar la metodología XP a nuestras necesidades, pues esta se basa en 5 pilares fundamentales de los cuales para el proyecto se implementaron sólo Planeación, Codificación en Parejas, Pruebas y Refactorización.

1. Planeación
   En la fase de planeación se tuvieron en cuenta procesos adaptados a nuestro grupo de trabajo como:

   - Las Historias de Usuario tienen el mismo propósito que los Casos de Uso, pero son escritas por los usuarios. Tienen como objetivo obtener los requerimientos del sistema, así como las cosas que el sistema tiene que hacer. Aplicando este concepto a nuestro proyecto, se encontró que el proceso es mucho más útil realizando el enfoque tradicional, basados en Casos de Uso, pues con las historias de usuario no se tendrían los mismos resultados.

     Los Casos de Uso para el prototipo de software Mobile-Tv se presentan en el anexo A.2 de este documento.

   - Programación en parejas.
     La filosofía de XP está fundamentada en la propiedad colectiva del código. Esta particular forma de trabajo mejora el rendimiento en la producción del software. El equipo de trabajo se fortalece aún más cuando don miembros se sientan frente a una misma pantalla y con un solo teclado se concentran en completar el objetivo del día.

     Esta forma de programación en parejas se adaptó en cada una de las fases de implementación, incluso durante la elaboración de cada una de las pruebas realizadas.

2. Pruebas
   Las pruebas unitarias consisten en el uso de un framework de Java Micro Edtion llamado JMUnit, que permiten comprobar el correcto funcionamiento de las principales clases del

proyecto.

Características de JMUnit:

- Funciona en los emuladores como en los dispositivos reales.
- Tiene una gran colección de Métodos *"Assert"* o Métodos de Aceptación para controlar los fallos de las pruebas.

3. Refactorización
   Refactor[23] o Refactorización es una técnica disciplinada para mejorar la estructura del código existente sin cambiar el funcionamiento del mismo.

   Esta técnica se utilizó para optimizar el diseño del software, aumentando la comprensión del código fuente, además permite un fácil mantenimiento. La refactorización tiene métodos que permiten encontrar errores de manera más sencilla, los más utilizados durante el proyecto fueron *Rename* (Renombrar), *SafelyDelete* (eliminación segura), *Introduce Method* (Crear Método).

   En la figura 19 se pueden observar las opciones presentadas por el IDE Netbeans para aplicar la técnica de Refactoring sobre el código fuente.

4. Herramientas Adicionales

   Netbeans
   El Entorno de Desarrollo Integrado Netbeans(IDE) (Integrated Development Environment) es una herramienta libre, de código abierto para desarrolladores de código. Este IDE está diseñado para crear aplicaciones de escritorio, empresariales, web y aplicaciones móviles entre otras.

   En el desarrollo del proyecto se usó la versión Netbeans 7.1.2, en particular en la creación de proyectos Java ME (Micro Edition)y Java Web.

   Java ME permite la creación de aplicaciones, pruebas y depuración para los Perfiles Móviles (Mobile Information Device Profile(MIDP) 1.0,2.0, 2.1, la configuración Connected Limited Device(CLDC) 1.0 y 1.1, y la configuración del dispositivoconectado(CDC).

   El IDE NetBeans viene con el último SDK de Java ME 3.0.5 que soporta CLDC y el desarrollo de los CDC. Adicionalmente se pueden agregar otras plataformas móviles como emuladores Nokia, Motorola, Samsung entre los más conocidos.
   Java Web permite crear fácilmente aplicaciones en el lado del servidor.Las tecnologías usadas para pruebas y desarrollo del proyecto investigativo son: Páginas JSP y Servlets.

   GitHub
   GitHub es el controlador de versiones más rápido, estable y eficiente integrado a Netbeans.

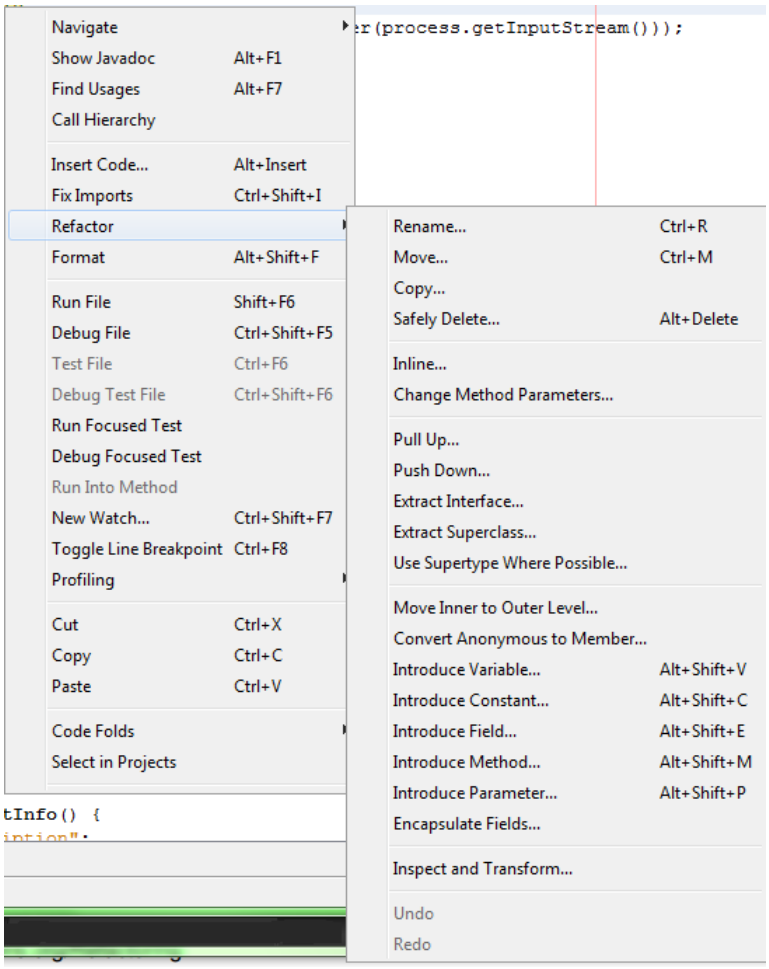

Figura 19: *Técnica de Refactor en Netbeans* Fuente: Autores del Proyecto

Este manejador de repositorios fue útil para realizar copias de seguridad y actualizar los cambios hechos por los programadoresdesde diferentes estaciones de trabajo.

Dropbox

Dropbox es un servicio de alojamiento de archivos operado por Dropbox, Inc. que ofrece almacenamiento en la nube y la sincronización de archivos.

Dropbox ofrece el software de cliente para Microsoft Windows, Mac OS X, Linux, Android, iOS y BlackBerry OS, y los navegadores web.

El servicio de Dropbox brindó el soporte necesario para llevar la documentación a lo largo del proyecto.

A.2. ESPECIFICACIÓN DE REQUISITOS SOFTWARE

De acuerdo con la metodología utilizada para realizar el proyecto, se crearon Casos de Uso para obtener los requerimientos del sistema.

En la figura 20 se presenta el diagrama de Casos de Uso del Prototipo de software Mobile Tv.
Consecutivamente se presentan de las tablas asociadas que especifican los casos de uso más relevantes.

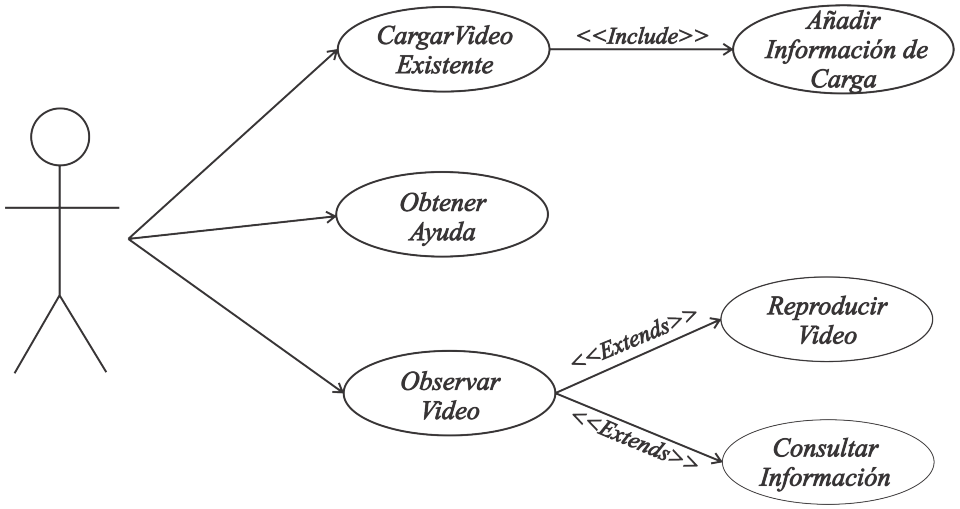

Figura 20: *Caso de uso POrototipo de Software Mobile Tv* Fuente: Autores del Proyecto

1. Arquitectura del prototipo Mobile-Tv.

   La topología de red indica como el sistema está compuesto en diferentes componentes tanto físicos como lógicos.

   En la actualidad los sistemas modernos utilizan una arquitectura cliente-servidor. Esta arquitectura está dividida en niveles distribuidos de la siguiente manera: el primer nivel, Nivel de Lógica y Datos, consta de un servidor en el cual se almacena la información más relevante del sistema, esta puede ser accedida me manera segura y generalmente se trata de una Base de Datos, también se ejecutan múltiples tareas que consumen mayores recursos computacionales. El siguiente nivel, es el Cliente, presenta la interfaz de usuario, en la cual él introduce los datos y realiza peticiones.

   Existen algunos beneficios que se obtienen al adaptar esta arquitectura:

   - Uso de los recursos especializados: Cada máquina está diseñada para realizar un trabajo determinado. Los recursos deben ser optimizados, especialmente cuando se trabaja con dispositivos móviles, porque el verdadero poder de éstos es la conectividad y el uso de la red.

Caso: Compartir Video Existente
Actor: Usuario móvil
Descripción: Se selecciona la actividad Compartir Video, en el cual se elige un video almacenado en el dispositivo para cargarlo al servidor de streaming.
Precondiciones: El video se debe encontrar almacenado en la tarjeta SD del dispositivo.
Flujo Normal: 1. El actor selecciona la actividad Compartir Video Existente. 2. El prototipo busca en la tarjeta SD los archivos de multimedia con formato .mp4 disponibles para que el usuario seleccione el deseado. 3. El sistema solicita completar un formulario que contiene información correspondiente al video, como Autor, Descripción General del Video y Título.
Flujo Alternativo: Cancelación de la acción. El prototipo vuelve al formulario inicial.
Postcondiciones: Almacenamiento en Base de Datos y creación de archivo en servidor multimedia. Notificación de confirmación de subida por parte del sistema.

Cuadro 15: *Comparación entre QuickTime Pro y MPEG4IP*

Caso: Compartir Video Nuevo
Actor: Usuario móvil
Descripción: Se selecciona la actividad Compartir Video, en la cual se puede subir un video al servidor de streaming.
Precondiciones: El dispositivo debe soportar la captura de video.
Flujo Normal: 1. El actor selecciona la actividad Compartir Video Nuevo. 2. El prototipo abre la comunicación con la cámara de video para iniciar la captura. 3. El actor almacena el video, y completa el formulario que contiene información correspondiente al video, como Autor, Descripción General del Video y Título.
Flujo Alternativo: Cancelación de la acción. El prototipo vuelve al formulario inicial descartando toda la actividad.
Postcondiciones: Almacenamiento en Base de Datos y creación de archivo en servidor multimedia. Notificación de confirmación de subida por parte del sistema.

Cuadro 16: *Comparación entre QuickTime Pro y MPEG4IP*

Caso: Obtener Ayuda
Actor: Usuario móvil
Descripción: Se selecciona obtener ayuda, para aclarar las funciones del prototipo y restricciones.
Precondiciones:
Flujo Normal: 1. El actor selecciona la actividad Obtener Ayuda. 2. El prototipo responde al usuario con un formulario lleno de información correspondiente al funcionamiento y utilidades del sistema.
Flujo Alternativo: La actividad Obtener Ayuda puede cancelarse un cualquier momento después de ser activada.
Postcondiciones: El prototipo vuelve al formulario inicial.

Cuadro 17: *Comparación entre QuickTime Pro y MPEG4IP*

Caso: Streaming de Video
Actor: Usuario móvil
Descripción: Se accede a la base de datos de Videos donde se consulta que archivos multimedia se encuentran disponibles para realizar streaming.
Precondiciones: El usuario debe seleccionar la opción Observar Video.
Flujo Normal: 1. El actor escoge la actividad Observar Video. 2. El prototipo realiza una búsqueda en la base de datos del sistema y presenta en pantalla los videos disponibles. 3. El actor selecciona un video,que es reproducirlo en el dispositivo móvil.
Flujo Alternativo: Detener la reproducción del video consultado. El prototipo vuelve al estado inicial.
Postcondiciones: El prototipo obtiene el tamaño de la pantalla para optimizar el uso de la misma, donde se presenta mediante streaming el video solicitado.

Cuadro 18: *Comparación entre QuickTime Pro y MPEG4IP*

- Mayor seguridad: Por lo general, en una arquitectura de tres niveles los clientes realizan operaciones mediante el uso del internet. Por lo tanto es necesario poseer una política de seguridad rigurosa, para proteger los equipos internos, nuestros programas y nuestros datos.

Entre otros beneficios, este tipo de arquitectura es recomendada para múltiples sistemas, incluso los más pequeños. Después de todo, posee mayor adaptabilidad comparada con otras como la arquitectura de tres niveles.

La topología cliente-servidor permite que un gran número de clientes realicen solicitudes a un servidor común que procesa las peticiones realizadas por los usuarios finales.

Al seleccionar la arquitectura para el prototipo se analizaron los términos de funcionalidad, comparados con el alcance del software. Fue necesario seleccionar un servidor especializado para la distribución de medios y poseer un manejador de base de datos.

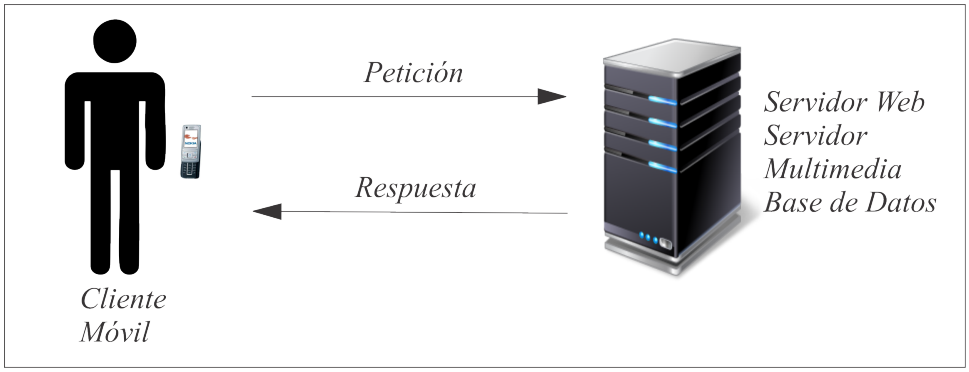

Figura 21: *Arquitectura general del sistema Mobile Tv* Fuente: Autores del Proyecto

2. Modelo del Sistema.

El patrón de diseño de Interfaces de Usuario (UI) adaptado al prototipo fue Modelo – Vista – Controlador (MVC). El patrón MVC es la manera para dividir la aplicación, o partes de la aplicación en tres secciones: Modelo, en el cual está el cuerpo de la aplicación, y donde se maneja la lógica del negocio; Vista, se encarga de presentar las UI al usuario; Controlador, su trabajo es procesar la información entregada por el usuario y delegar el trabajo al Modelo del sistema, para actualizar finalmente la vista del usuario.

Prototipo de Software Mobile Tv

Basado en un controlador, llamado comúnmente Controlador del Sistema¸ es el responsable de recepción y el procesamiento todos los eventos del sistema. Este controlador, normalmente delega la mayor cantidad de trabajo al Modelo, y sólo gestiona la salida en la pantalla, procesando y distribuyendo los eventos del sistema.

A.3. DIAGRAMAS DE CLASES

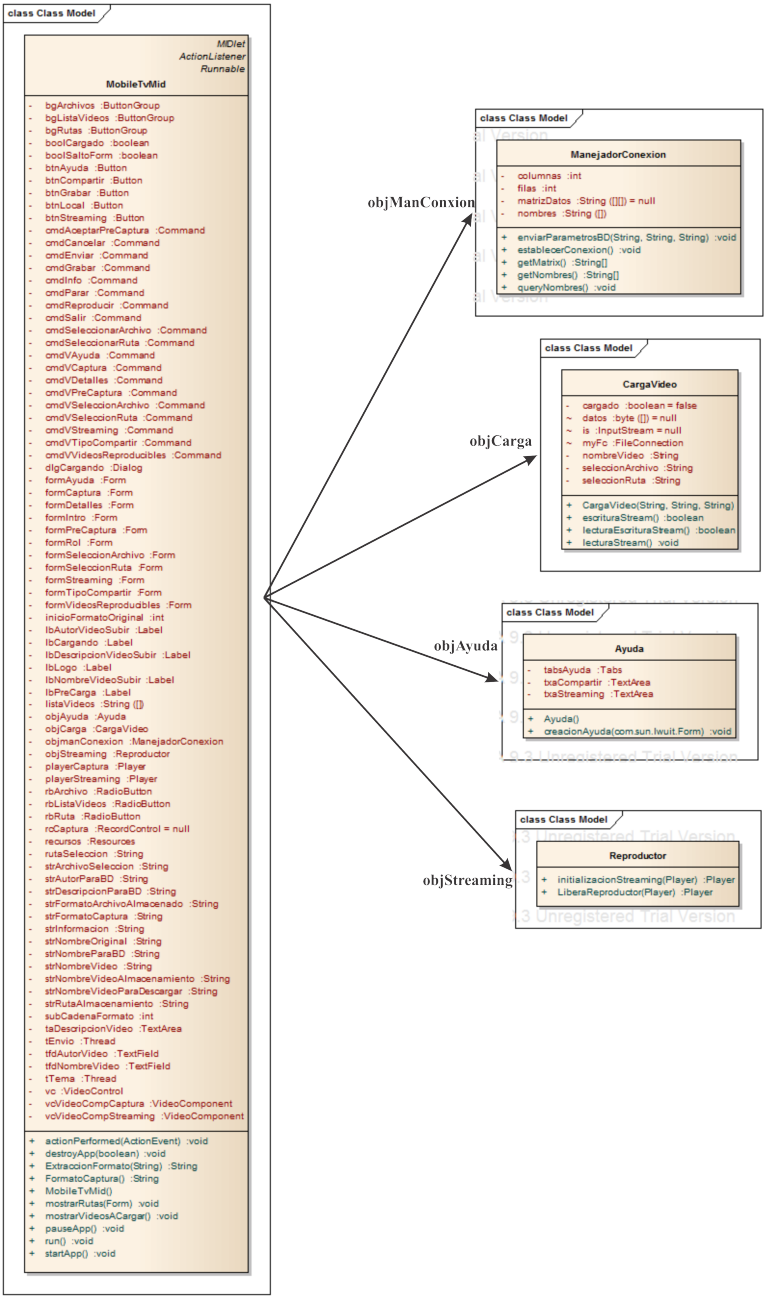

Figura 22: *Diagrama de Clases del protoripo de software Mobile Tv.*
Fuente: Autores del Proyecto

A.4. MANUAL DE USUARIO

Ver CD.

A.5. TABLA DE CÓDIGOS DE RESPUESTA PROTOCOLO RTSP

Código	Estado	–	Código	Estado
100	Continue		411	Length Required
200	OK		412	Precondition Failed
201	Created		413	Request Entity Too Large
250	Low on Storage Space		414	Request-URI Too Large
300	Multiple Choices		415	Unsupported Media Type
301	Moved Permanently		451	Parameter Not Understood
302	Moved Temporarily		452	Conference Not Found
303	See Other		453	Not Enough Bandwidth
304	Not Modified		45x	Session Not Found
305	Use Proxy		45x	Method Not Valid in This State
400	Bad Request		45x	Header Field Not Valid for Resource
401	Unauthorized		45x	Invalid Range
402	Payment Required		45x	Parameter Is Read-Only
403	Forbidden		45x	Aggregate operation not allowed
404	Not Found		45x	Only aggregate operation allowed
405	Method Not Allowed		500	Internal Server Error
406	Not Acceptable		501	Not Implemented
407	Proxy Authentication Required		502	Bad Gateway
408	Request Time-out		503	Service Unavailable
410	Gone		504	Gateway Time-out
			505	RTSP Version not supported

Cuadro 19: *Los códigos de estado y su uso con los métodos de RTSP* Fuente: Autores del Proyecto