UNIVERSIDAD CARLOS III DE MADRID

UNIVERSIDAD POLITÉCNICA SUPERIOR

INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN:

TELEMÁTICA

PROYECTO FIN DE CARRERA

IMPLEMENTACIÓN DE UN SERVIDOR DE VoD PARA REDES CON IMS

Autora: Mª Nieves Santillán Celada

Tutor: Ignacio Soto Campos

Título: Implementación de un servidor de VoD para redes con IMS

Autor: Mª Nieves Santillán Celada

Director: Ignacio Soto Campos

EL TRIBUNAL

Presidente: _____

Vocal:

_____

Secretario:

_____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

Lo primero, a mi tutor Ignacio Soto Campos,por su orientación e infinita paciencia.

Lo segundo, a mis padres, por su eterna pregunta … “y qué tal llevas el proyecto?”.

En tercer lugar a mis “compañeras” de proyecto, Maia, mi hija, y Puti, mi perra, que han estado conmigo en todo momento.

Por último y no menos importante, a José, mi compañero, por ser el blanco de iras y momentos de frustración durante la elaboración de este proyecto.

Resumen

El proyecto fin de carrera consiste en la implementación de un servidor de vídeo bajo demanda (VoD) para redes con IMS.

Se ha desarrollado un servidor que ofrece vídeo y su manipulación en tiempo real a los clientes que así lo soliciten dentro de una red con IMS. Por otra parte, se ha desarrollado un cliente que solicita el vídeo al servidor y lo presenta.

Tanto el servidor de vídeo como el cliente actúan como usuarios de la red IMS, implementando toda la funciones de un cliente SIP/IMS.

La comunicación de control entre cliente y servidor se realiza empleando señalización SIP. Esta comunicación incluye por un lado la negociación del vídeo a transmitir, así como su formato, y por otro la manipulación del vídeo por parte del cliente.

El cliente solicita al servidor un vídeo en un determinado formato, estableciéndose una sesión en caso de llegar a un acuerdo. Esta negociación se lleva a cabo empleando SDP.

El cliente podrá manipular el vídeo indicando si quiere iniciarlo, pararlo o pausarlo.

La transmisión de vídeo se realiza a través de los protocolos RTP y RTCP, empleando la librería de código abierto *live555 streaming media*.

Adicionalmente se incorpora una base de datos que contiene toda la información multimedia de vídeos que el servidor es capaz de ofrecer.

Palabras clave: IMS (IP Multimedia Subsystem), SIP (Session Initiation Protocol), RTP (Real-Time transport protocol), VoD (Video on Demand)

Abstract

This work involves the design and implementation of a video on demand (VoD) server for IMS-enabled networks.

This server offers video to its clients. It also allows the users to control the playing of the video in real time through the IMS network. Aditionally, a client that requests videos to the server and plays them has also been developed.

The video server and the client are both IMS users, therefore they implement all the functions of an IMS client.

The signalling between the client and the server is done through SIP messages. This communication includes, on one hand, the negotiation of the video to transmit, as well as its codecs;, and, on the other hand, the control of the playing of the video.

The client requests a video to the server asking for particular codecs, and sets up a session with the server if they agree to do so. This negotiation is carried out using SDP.

A user will be able to control the playing of the video through the functionality provided by the client program, being able to ask for stopping it, starting it or pausing it.

The video transmission is done through RTP and RTCP, using the open source library *live555 streaming media*.

Additionally, a data base has been incorporated. It contains all the multimedia information about the videos that the server is able to offer.

Keywords: IMS (IP Multimedia Subsystem), SIP (Session Initiation Protocol), RTP (Real-Time transport protocol), VoD (Video on Demand)

Índice general

Índice de figuras

Índice de tablas

Acrónimos

AS: Application Server.

CSCF: Call Session Control Function.

DNS: Domain Name System.

E-CSCF: Emergency Call Session Control Function.

HSS: Home Subscriber Subsystem.

IMS: IP Multimedia Subsystem.

IP: Internet Protocol.

I-CSCF: Interrogating Call Session Control Function.

IPTV: Internet Protocol Television.

LTE: Long Term Evolution.

NGN: Next Generation Networking.

P-CSCF: Proxy Call Session Control Function.

RTP: Real-time Transport Protocol.

RTCP: RTP Control Protocol.

S-CSCF: Serving Call Session Control Function.

SDP: Session Description Protocol.

SIP: Session Initiation Protocol.

TISPAN: Telecommunication and Internet Converged Services and Protocols for Advanced Networks

UMTS: Universal Mobile Telecommunications System.

UDP: User Datagram Protocol.

Capítulo 1

Introducción y Objetivos

1.1 Introducción

Este proyecto ha consistido en la implementación de un servidor de vídeo bajo demanda (VoD) para redes con IMS. La idea es emplear la señalización SIP para controlar el envío de vídeo y su manipulación a través de la red IMS.

1.2 Motivación del proyecto

IMS es una arquitectura que proporciona el plano de control para ofrecer voz y otros servicios multimedia a usuarios fijos y móviles sobre redes IP.

Aparece por primera vez en el año 2002, en la Release[1] 5 de 3GPP[2], siendo modificado y mejorado a lo largo de diferentes releases. Actualmente se está desarrollando la Release 13, prevista para finales del 2015.

IMS ha sido adoptado por otros organismos de estandarización como por ejemplo el ETSI[3],

1 Release es como denomina el 3GPP a una versión de sus estándares.

2 3GPP: The 3rd Generation Partnership Project, http://www.3gpp.org

más concretamente TISPAN[4], que tiene como base IMS.

La clave principal de IMS es la convergencia. Tradicionalmente, las operadoras han ofrecido los servicios multimedia sobre redes de conmutación de circuitos y los servicios de datos sobre redes de paquetes. IMS permite proveer los servicios multimedia sobre redes de paquetes. Esto permite a las operadoras la convergencia de sus redes a redes basadas en conmutación de paquetes.

Por otra parte, cabe destacar los RCS[5]. Se trata de servicios [1] basados en IMS que permiten a los usuarios mantener conversaciones de texto o de voz e intercambiar imágenes o vídeos durante las mismas.

El éxito inicial de IMS no fue el esperado debido a una serie de razones.

La principal fue que precisaba una revisión bastante significativa de la red, así como del modelo de negocio, lo que implicaba una gran inversión.

Además, las redes de conmutación de circuitos funcionaban correctamente generando beneficios.

Finalmente no existían ni terminales IMS ni servicios competitivos para ofrecer a los usuarios.

Este escenario hacía que las operadoras se mantuvieran reticentes a la espera de ver qué hacían otras operadoras.

Con el paso del tiempo el panorama ha cambiado radicalmente y las operadoras se muestran más receptivas a IMS.

Las aparición de aplicaciones OTT[6] ha sido en parte responsable de este cambio. Se trata de servicios de vídeo, televisión, mensajería que se proveen a través de Internet. Este tipo de aplicaciones, entre las que destacan las destinadas a mensajería, están teniendo un impacto negativo en los ingresos de las operadoras. Los RCS representan un medio para hacer frente a la competencia de las OTTs. En la actualidad todas las operadoras ofrecen RCS, con diferentes niveles de éxito. Algunos ejemplos son :

- Joyn, que ofrece mensajería instantánea,
- “Te lo guardo”, permite realizar backups de nuestras fotos y contactos

Sin embargo, el cambio de parecer de las operadoras, en particular de las móviles, radica realmente en la estandarización por parte de 3GPP de LTE (4G), más específicamente de VoLTE[2], que consiste en que en redes LTE el servicio de voz se ofrece sobre la red de paquetes

3 European Telecommunications Standars Institute

4 Telecommunication and Internet Converged Services and Protocols for Advanced Networks

5 Rich Communications Services

6 Over the top

usando IMS.

VoLTE sustituye la conmutación de circuitos presente en 2G/3G, ofreciendo una mayor calidad y un menor tiempo de establecimiento de llamada. Además permite la creación de nuevos servicios avanzados que facilitan la comunicación multimedia y en tiempo real.

Actualmente, varias operadoras están planeando su lanzamiento o ya lo han realizado:

- Vodafone Hutchiston Australia ha finalizado las pruebas correctamente y se propone empezar a emplearlo durante este año [3].
- La operadora japonesa NTT DOCOMO lanzó VoLTE en Junio de 2014, siendo el primero proveedor de VoLTE en Japón. [4]

Es en este contexto en el que surge el presente Proyecto Fin de Carrera, en el que el objetivo es desarrollar un servicio muy atractivo para ofrecer sobre IMS: vídeo bajo demanda (VoD).

1.3 Objetivos del proyecto

El objetivo del proyecto es la implementación de un servidor de VoD para redes IMS.

El servidor de VoD debe ser capaz de utilizar el IMS de la red para llevar a cabo el registro en la red IMS y aceptar o rechazar sesiones de clientes. Estas sesiones son usadas por los clientes para solicitar vídeos al servidor. Esto implica que el servidor debe enviar mensajes SIP a la red IMS, así como recibirlos e interpretarlos.

El servidor debe también ser capaz de acceder a la base de datos que contiene la información multimedia para comprobar si puede ofrecer el vídeo solicitado por el cliente o no.

Ofrecerá el vídeo solicitado al cliente a través de una comunicación RTP/RTCP, facilitando una serie de comandos para su manipulación. Estos comandos serán PLAY, PAUSE y STOP y serán comunicados por el cliente al servidor usando mensajería SIP.

Otro objetivo del proyecto ha sido desarrollar un cliente para poder probar el funcionamiento del servidor.

El cliente se comunica con la red IMS, por lo que deberá registrarse.

También debe ser capaz de iniciar una sesión con el servidor, especificando a través de SDP las características de la misma, así como el vídeo solicitado.

Finalmente, el cliente mostrará el vídeo tal y como lo recibe, y lo manejará a través del envío de los comandos PLAY, PAUSE Y STOP al servidor empleando mensajería SIP.

1.4 Estructura del proyecto

La memoria se estructura en 6 capítulos.

El capítulo 1 (este capítulo)es la introducción al proyecto. En este capítulo se detalla la motivación del proyecto y el objetivo del mismo.

El capítulo 2 es el estado del arte. En este apartado se explica la arquitectura IMS y los protocolos principales de IMS. Nos servirá de base técnica para entender el proyecto.

El capítulo 3 describe las herramientas utilizadas para el proyecto. En él se describen el Open IMS core de Fokus, una implementación de IMS empleada en el entorno de pruebas; la librería live555, utilizada en el servidor y en el cliente para la comunicación con IMS y la transmisión de vídeo; la base de datos; y el reproductor de vídeo empleado en el cliente.

El capítulo 4 describe detalladamente el diseño y la implementación del servidor VoD. Se divide en dos apartados. El primero introduce la comunicación con la red IMS y el segundo explica todo lo relacionado con el vídeo.

El capítulo 5 muestra las pruebas realizadas con el servidor de VoD que nos han permitido evaluar la implementación del mismo.

El capítulo 6 se divide en dos apartados. Un primer apartado analiza el cumplimiento de los objetivos iniciales del proyecto y las conclusiones del mismo. En el segundo apartado se proponen diferentes aplicaciones futuras del proyecto.

Además se pueden encontrar 3 anexos:

El anexo A incluye la planificación del proyecto, explicando cómo se ha realizado la división de tareas e incluyendo un cuadro resumen de la planificación del proyecto.

El anexo B contiene el presupuesto del proyecto.

El anexo C contiene un manual de instalación que explica cómo instalar el software necesario para el despliegue del servidor de VoD.

El anexo D contiene una explicación a bajo nivel de la librería live555.

El anexo E muestra los mensajes SIP intercambiados entre cliente y servidor durante las pruebas realizadas.

El anexo F presenta una bibliografía, que incluye un listado con todas las referencias, documentos y libros que han sido utilizados como soporte para la elaboración del proyecto.

Capítulo 2

Estado del arte

2.1 Introducción

A continuación se ofrece una descripción detallada del entorno tecnológico en el que se ha desarrollado el proyecto. Se describirá la arquitectura IMS, detallando los elementos que presenta y sus funciones. Finalmente se explicarán los protocolos empleados en el sistema como SIP, SDP y RTP.

2.2 IMS[7]

2.2.1 ¿Qué es IMS?

IMS [5] es una especificación, inicialmente del 3rd Generation Partnership Project (3GPP) aunque adoptada por otros organismos de estandarización, que proporciona servicios multimedia sobre redes de paquetes fijas y móviles, soportando un gran número de tecnologías de acceso sobre estándares abiertos.

IMS proporciona una verdadera convergencia fijo - móvil. La capa de control y la capa de

7 *IP Multimedia Subsystem*

aplicación van a trabajar igualmente en el mundo móvil que en el mundo fijo.

Así mismo, proporciona una convergencia a una red de paquetes que ofrece todo tipo de servicios, incluidos los tradicionales de conmutación de circuitos.

Por otra parte, IMS tiene en cuenta que diferentes tipos de acceso precisan de diferentes requisitos. Algunos servicios requieren un gran ancho de banda mientras otros lo que necesitan es una baja latencia. La funcionalidad de multiacceso de IMS hace que los diferentes servicios se ejecuten correctamente, dado que conoce las características de cada acceso.

Cabe destacar que habilitar una única red para todos los tipos de acceso significa un gran beneficio para los operadores de red, permitiendo ahorrar costes en equipos, operación y mantenimiento. IMS proporciona mecanismos comunes para gestión de red, provisión de servicio, cargos y control de acceso.

En cuanto a la creación de servicios, IMS facilita su creación e implantación. Los servicios son alojados en servidores de aplicaciones y son accedidos de forma estandarizada. Además, dado que toda la infraestructura ya está presente (autenticación, autorización, etc), es muy simple habilitar un nuevo servicio en un servidor de aplicaciones. Por último facilita la interconexión de servicios a través de toda la comunidad de operadores.

Para terminar, destacamos que está basado en estándares abiertos, como por ejemplo SIP y Diameter.

2.2.2 Arquitectura IMS

IMS presenta cuatro capas totalmente diferenciadas, cada una con una funcionalidad propia, como se observa en la ilustración 1 :

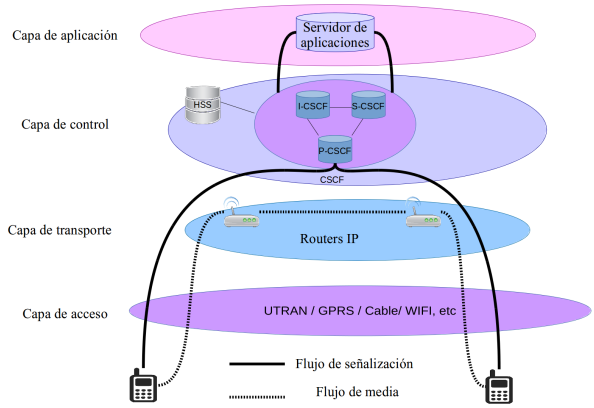


Ilustración 1: Capas IMS

La capa de acceso representa todo acceso de alta velocidad basado en conmutación de paquetes, como UTRAN o redes de banda ancha. En caso de tratarse de una red basada en conmutación de circuitos se precisará de un elemento adicional que transforme la señalización a conmutación de paquetes.

La capa de transporte se trata de una red IP.

La capa de control es el núcleo de IMS. Se encarga del encaminamiento de la señalización entre usuarios y de la invocación de servicios.

La capa de aplicación está formada por los servidores de aplicación que proporcionan servicios a los usuarios.

2.2.3 Entidades y sus funcionalidades

Esta sección explica las diferentes entidades IMS y sus funcionalidades principales.

Las entidades podrían ser divididas en cuatro categorías:

1. Gestión de la sesión y encaminamiento de señalización
2. Bases de datos
3. Servicios.
4. Funciones de interacción

2.2.3.1 Gestión de la sesión y encaminamiento de señalización

Existen 4 entidades dentro de este grupo, cada una con su tarea específica. La ilustración 2 muestra cómo están conectadas:

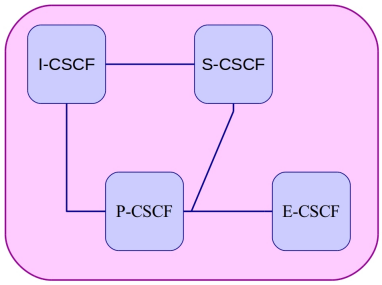


Ilustración 2: Entidades IMS

P-CSCF (Proxy Call Session Control Function)

Es la primera entidad con la que los usuarios interaccionan en IMS. Todo el tráfico de señalización SIP de los usuarios será recibido por esta entidad. De la misma forma, todo el tráfico de señalización SIP de IMS será enviado al usuario final a través del P-CSCF.

Sus tareas principales son:

- ✓ La compresión y descompresión de mensajería SIP.
- ✓ Mantiene las asociaciones de seguridad y aplica integridad y protección a la señalización SIP.
- ✓ Detecta las sesiones de emergencia y las encamina al E-CSCF, explicado más adelante.

I-CSCF (Interrogating Call Session Function)

Se trata de un nodo intermedio que ayuda a otros nodos a determinar el siguiente salto de los mensaje SIP y a establecer un camino para la señalización. Tiene asignadas 3 funciones:

- ✓ Obtener del HSS el nombre del siguiente salto de la señalización, ya sea un S-CSCF o un servidor de aplicaciones.
- ✓ Asigna un S-CSCF en función del perfil obtenido del HSS.
- ✓ Encamina peticiones entrantes al S-CSCF o al servidor de aplicaciones.

S-CSCF (Serving Call Session Function)

Se trata del elemento clave en el control de sesión que realiza IMS. Sus tareas principales son:

- ✓ Maneja el proceso de registro.
- ✓ Toma decisiones de encaminamiento de la señalización SIP.

- ✓ Mantiene el estado de las sesiones.
- ✓ Almacena los perfiles de servicio.

E-CSCF (Emergency Call Session Control Function)

Se trata de una funcionalidad dedicada a manejar peticiones de emergencia dentro de IMS, tales como llamadas a policía, bomberos o ambulancias.

Su principal tarea es seleccionar un centro de emergencia al que la petición debería ser dirigida.

2.2.3.2 Bases de datos

Existen dos bases de datos principales dentro de la arquitectura IMS:

HSS

Almacena los datos de los subscriptores y servicios de la red IMS. Esta información incluye:

- ✓ Identidades de usuarios, tanto públicas como privadas.
- ✓ Parámetros de acceso utilizados para establecer sesiones, como por ejemplo la autenticación del usuario.
- ✓ Lanzadores de servicio. Filtros que definen las condiciones en las que se habilita o no la ejecución de un servicio, en función de su configuración.

SLF

Es utilizado como un mecanismo de resolución que permite a el I-CSCF, el S-CSCF y a los servidores de aplicaciones encontrar la dirección del HSS que mantiene la información de un suscriptor dada una identidad de usuario. Es necesario cuando existen múltiples HSS dentro de una red.

2.2.3.3 Servicios

Aquí encontramos las siguientes funciones:

MRFC

Interpreta la señalización SIP recibida del S-CSCF y controla el MRFP. Soporta servicios tales como conferencias o notificaciones a usuarios.

MRFP

Mezcla flujos de diferentes conexiones y procesa los flujos multimedia que recibe.

AS

Implementan lógica de servicios. Definen qué hacer ante la recepción de mensajes SIP o ciertos eventos, para proporcionar servicios multimedia de valor añadido en IMS, pudiendo estar en la red local o en otra red.

2.2.3.4 Funciones de comunicación con red telefónica tradicional

A continuación vamos a describir una serie de entidades cuyas funciones son necesarias para habilitar la comunicación de voz y vídeo entre IMS y redes de conmutación de circuitos.

BGCF y MGCF

El S-CSCF decide cuándo una petición debe salir a la red de conmutación de circuitos. Para ello envía una petición SIP al BGCF, que seleccionará la pasarela de salida. Esta salida puede ser en la misma red IMS donde se encuentra el BGCF o en otra distinta. Cuando se produce en la misma red, el BGCF selecciona un MGCF para que interactúe con la red de conmutación de circuitos.

Si hay que enviarlo a otra red IMS, el BGCF reenvía la petición SIP al BGCF de la red seleccionada.

El MGCF realiza la conversión de la señalización del dominio de conmutación de circuitos (ISUP/SS7) a la señalización SIP y viceversa, empleando el SGW, que realiza la interconexión física entre las dos redes de señalización.

2.3 Protocolos principales de IMS

2.3.1 SIP[8]

Protocolo de señalización [6] [7] [8] desarrollado por el grupo de trabajo MMUSIC del IETF. Se trata de un protocolo abierto y ligero. Seleccionado por el grupo 3GPP como protocolo principal para el control de la sesión en IMS.

Su principal característica es que nos permite establecer, modificar y finalizar sesiones multimedia.

SIP utiliza la arquitectura cliente/servidor, donde el cliente genera un mensaje y el servidor responde a ese mensaje y lo reenvía.

2.3.1.1 Direcciones SIP

Las direcciones en SIP, también conocidas como SIP URI, utilizan el concepto de URI

8 *Session Initiation Protocol*

(Universal Resource Indicator).

Su formato es como sigue:

sip:[userinfo]@[hostport][parameterers]

userinfo: almacena la información del usuario.

Hostport: contiene el nombre del dominio o una dirección IP. Puede contener un puerto.

Parameterers: parámetros adicionales, separados cada uno por un ";".

2.3.1.2 Mensajería SIP

La mensajería SIP es legible y presenta el siguiente formato:

Línea de inicio

Cabeceras SIP

Línea en blanco

Cuerpo del mensajería

Encontramos mensajes SIP de petición y mensajes SIP de respuesta, siendo la línea de inicio su principal, aunque no única, diferencia.

A continuación comentamos cada uno de ellos.

Mensaje SIP de petición

La lína de inicio sigue el siguiente formato:

MÉTODO Request-URI SIP-VERSION

A continuación se explica detalladamente la línea de inicio para la mensajería SIP de petición.

El método indica la acción a realizar. Los principales métodos se muestran en la Tabla 1:

Método	Descripción
INVITE	Petición de inicio de sesión
ACK	Mensaje enviado por el cliente para indicar que una respuesta de éxito a un INVITE ha sido recibida.
OPTIONS	Petición al servidor de sus características.
BYE	Liberación de la sesión
CANCEL	Cancelación de cualquier petición pendiente.
REGISTER	Utilizado por el cliente para registrarse en la red IMS.
INFO	Se emplea para transportar información a nivel de aplicación dentro de una sesión SIP.

Tabla 1: Métodos de petición en SIP

Además de estos métodos, existen otros métodos definidos en extensiones a la especificación SIP, tales como : NOTIFY, UPDATE y MESSAGE.

El *Request-URI* es el destinatario de la petición.

La versión de SIP, es la versión que se está empleando del protocolo SIP.

Finalmente, se introduce un nueva línea para separar la línea de inicio de la cabecera.

En cuanto a las cabeceras SIP, en los mensajes SIP de petición las obligatorias se muestran en la Tabla 2:

Cabecera	Descripción
Via	Muestra la ruta tomada por la petición.
From	Origen de la petición
To	Destino de la petición
Call-ID	identificador único generado por el cliente.
Cseq	Número de secuencia. Es generado por el cliente e incrementado en cada petición sucesiva

Tabla 2: Cabeceras obligatorias en mensajes de petición SIP

Mensaje SIP de respuesta

La lína de inicio sigue el siguiente formato:

SIP-VERSION CODIGOESTADO RAZON

A continuación se explica detalladamente la linea de inicio para la mensajería SIP de petición.

SIP-VERSION es la versión del protocolo SIP que se está empleando.

El código de estado es un valor numérico que contiene la respuesta a la petición. Los principales códigos de estado se muestran en la Tabla 3:

Código	Tipo	Descripción
1xx	Informativo	Para indicar que una petición ha sido recibida o que se está procesando la petición.
2xx	Éxito	Este código indica que la petición a la que hace referencia fue recibida y aceptada.
3xx	Redirección	Otras acciones deben ser llevadas a cabo para completar la petición a la que hace referencia.
4xx	Error de cliente	Código que indica que la petición a la que hace referencia presenta una sintaxis errónea o no puede ser llevada a cabo por el servidor.
5xx	Error de servidor	El servidor fallo al procesar una petición aparentemente correcta.
6xx	Error global	La petición a la que hace referencia la respuesta es inválida en cualquier servidor

Tabla 3: Códigos de estado en mensajes de respuesta SIP

La razón es un texto descriptivo que acompaña al código de estado.

Finalmente, se introduce un nueva línea para separar la línea de inicio de la cabecera.

En cuanto a las cabeceras SIP, las obligatorias son las mismas que aparecen en el mensaje de petición.

2.3.1.3 Operaciones principales SIP en IMS

A continuación vamos a mostrar las operaciones principales en SIP entrando en detalle en la mensajes intercambiados.

Registro de un usuario en la red IMS

Para acceder a la red IMS, un usuario debe registrarse.

El proceso de registro de un usuario en la red IMS se divide en dos partes. Una primera en la

que el usuario indica su identidad y dominio en un mensaje tipo REGISTER. La red IMS rechazará el registro retando al usuario para que introduzca las credenciales y un código de autenticación en un segundo mensaje REGISTER. El usuario enviará este segundo mensajes y será entonces cuando, en caso de haber introducido la información solicitada correctamente, estará registrado en la red IMS.

A continuación, la ilustración 3 muestra el intercambio de mensajería acompañado de una descripción de la misma.

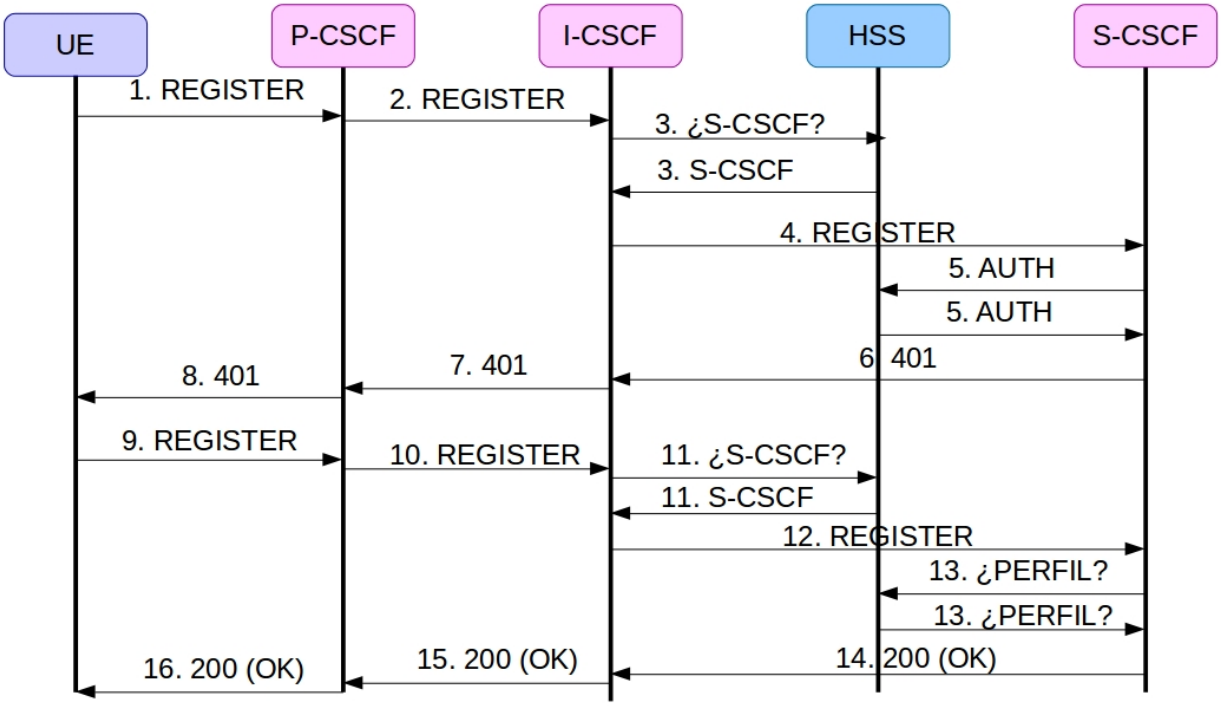


Ilustración 3: Registro de un usuario en IMS

1. El usuario que quiere registrarse envía un mensaje tipo *REGISTER* con su identidad y el nombre de su dominio a la red IMS.

2. El P-CSCF procesa el mensaje y emplea el nombre del dominio para obtener el I-CSCF, enviándole el mensaje.

3. El I-CSCF se comunicará con el HSS para seleccionar el S-CSCF adecuado.

4. Una vez obtenido el S-CSCF, el I-CSCF le enviará el mensaje register.

5. El S-CSCF comprueba que el usuario NO está autenticado, obteniendo la información de

autenticación del HSS.

6. Genera un mensaje tipo 401 indicándolo así y enviándolo al I-CSCF.

7. El I-CSCF se lo envía al P-CSCF.

8. El usuario obtiene la respuesta 401 junto con el reto agregado por el S-CSCF.

9. El usuario envía al IMS un segundo mensaje tipo REGISTER con la respuesta al reto.

10. El P-CSCF es el que recibe el mensaje, enviándoselo al I-CSCF.

11. El I-CSCF se encarga de encontrar el S-CSCF obteniendo dicha información del HSS.

12. El I-CSCF envía el nuevo mensaje REGISTER al S-CSCF encontrado.

13. El S-CSCF comprueba la respuesta al reto y, en caso de ser correcta, descarga el perfil del usuario del HSS.

14. El S-CSCF envía un mensaje OK indicando que el registro se ha realizado correctamente.

15. El I-CSCF recibe el mensaje y se lo re-envía al P-CSCF.

16. El P-CSCF recibe el mensaje para comunicárselo finalmente al usuario.

Establecimiento y finalización de una sesión entre dos usuarios

A la hora de iniciar una sesión, un cliente debe enviar un mensaje tipo INVITE. Para finalizar una sesión se empleará un mensaje tipo BYE. El intercambio de mensajería durante el establecimiento y finalización de una sesión, en el supuesto de que ambos usuarios se encuentren registrados en la misma red, se muestra en la ilustración 4:

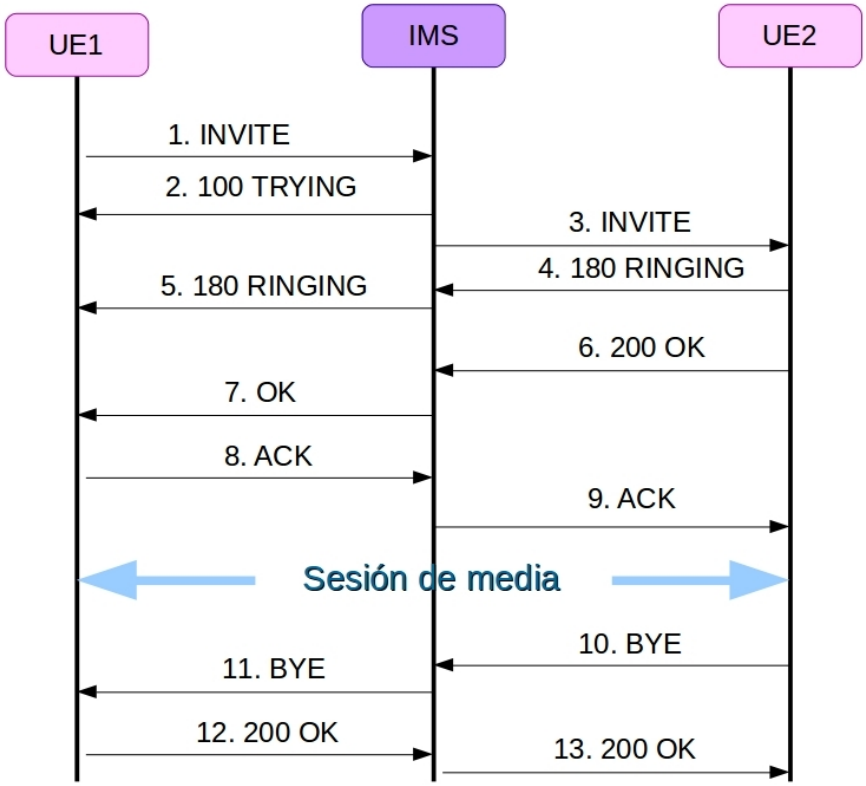


Ilustración 4: Establecimiento de sesión entre dos usuarios

A continuación se destallan los pasos:

1. El usuario que quiere iniciar la sesión envía un mensaje tipo INVITE a la red IMS indicando destinatario.

2. El usuario recibe de IMS un mensaje tipo 100, que le indica que se ha iniciado el proceso.

3. La red IMS localiza al destinatario y le envía el mensaje INVITE.

4. El usuario destino envía un mensaje tipo 180 indicando que ha recibido la llamada.

5. Este mensaje llega al usuario origen a través de IMS.

6. El usuario destino indica que acepta la sesión a través de un mensaje tipo 200.

7. El usuario origen recibe el mensaje de aceptación de sesión vía IMS.

8. El usuario origen envía un mensaje de petición tipo ACK para indicar que ha recibido la aceptación de la sesión.

9. El usuario destino recibe el ACK de IMS.

Ahora la sesión multimedia ha sido iniciada. La descripción de esta sesión se habrá realizado mediante SDP, explicado en la siguiente sección.

10. Cuando alguna de las partes quiere finalizar la sesión, en este caso el destino, envía un mensaje tipo BYE.

11. El usuario del otro extremo recibe la petición de finalización de la llamada.

12. Este usuario indica que ha recibido la petición a través de un mensaje tipo 200.

13. La contraparte recibe el mensaje tipo 200 y la sesión se finaliza.

2.3.3 SDP[9]

Protocolo [6] [9] de texto definido por el IETF y estandarizado en la RFC 4566. Se utiliza principalmente para la descripción de una sesión multimedia.

La descripción de la sesión realizada por SDP consiste en una serie de líneas con formato:

tipo=valor

Donde tipo es un campo de la cabecera de SDP y valor, el valor asociado al campo.

Dentro de la mensajería SIP, el protocolo SDP aparecerá incrustado en el cuerpo del mensaje.

La Tabla 4 muestra los principales campos de cabecera SDP:

9 Session Description Protocol

Campo	Descripción
Versión	v=0
Origen	o= <nombreUsuario> <id sesion> <version> <tipo red> <tipo dirección> <dirección>
Nombre de sesión	s = nombre de sesión
Tiempos	t = <tiempo incio> <tiempo fin>
Datos de conexión	C = <tipo red> <tipo dirección> <dirección>
Contenido	m = <contenido> <puerto> <transporte> <lista de formatos de media>

Tabla 4: Campos SDP

Prestaremos especial atención al campo media dado que juega un papel muy importante en la descripción de la sesión en el ámbito de este proyecto. El formato de esta línea es el que sigue:

m = <contenido> <puerto> <transporte> <lista de formatos de contenido>

- contenido: parámetro que indica el flujo a transmitir, pudiendo tomar los siguientes valores: audio, video, text, application y message.
- Puerto: puerto a través del cual se enviará el flujo multimedia.
- Transporte: Protocolo a emplear en el envío de flujo multimedia. Puede tomar diferentes valores como:

Protocolo	Descripción
UDP	No se especifica ningún protocolo en concreto
RTP/AVP	RTP para el envío de audio y/o vídeo
RTP/SAVP	RTP para el envío de audio y/o vídeo en modo seguro

Tabla 5: Protocolo transporte en SDP

- lista de formatos de media: Los formatos en los que el media se puede codificar. La Tabla 6 muestra los valores principales:

Valor	Nombre	Tipo
0	PCMU	audio
1	reservado	audio
2	reservado	audio
3	GSM	audio
4	G723	audio
5	DVI4	audio
6	DVI4	audio
7	LPC	audio
8	PCMA	audio
9	G722	audio
10	L16	audio
11	L16	audio
12	QCELP	audio
13	CN	audio
14	MPA	audio
15	G728	audio
16	DVI4	audio
17	DVI4	audio

Valor	Nombre	Tipo
18	G729	audio
25	CELB	vídeo
26	JPEG	vídeo
28	Nv	vídeo
31	H261	vídeo
32	MPV	vídeo
33	MP2T	Audio/vídeo
34	H263	vídeo

Tabla 6: Lista de formatos de contenido

2.3.4 Otros protocolos

Además de los protocolos de señalización, el servidor de VoD emplea otro tipo de protocolos en el plano de datos. En esta sección nos vamos a centrar en RTP y RTCP como protocolos para el transporte de datos en tiempo real, como audio y vídeo sobre UDP.

2.3.3.1 RTP[10]

Desarrollado [6] [10] por el grupo de trabajo de trasporte de audio y video del IETF y publicado en la RFC 3550. Se trata de un protocolo utilizado para la transmisión de la información en tiempo real.

Generalmente se implementa sobre UDP/IP, aunque también puede ir sobre TCP.

RTP permite a cada fuente abrir uno o varios flujos, en función de si se quiere separar el audio y el vídeo, como ocurriría con MPEG1 y MPEG2, o se quieren enviar en un mismo stream, como es el caso de TS.

Es importante tener en cuenta que RTP no provee ningún mecanismo para asegurar la entrega de paquetes en destino o cualquier otra calidad de servicio.

10 *Real-time Transport Protocol*

Finalmente se destaca que puede ser empleado tanto en aplicaciones unicast como multicast.

La Ilustración 5 muestra la cabecera RTP:

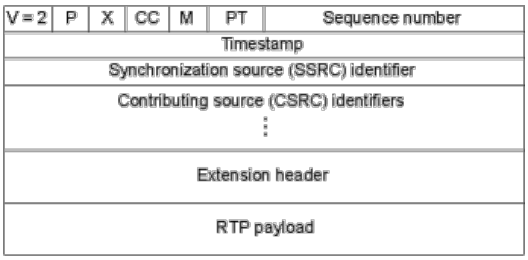


Ilustración 5: Cabecera RTP

La Tabla 7 muestra el significado de los campos:

Campo	Descripción
V	Versión del protocolo
P	Indica la presencia de relleno
X	Especifica si la cabecera es seguida por una extensión
M	Marcador que indica el inicio de imagen si se transmite vídeo
CC	Número de fuentes
PT	Tipo de dato multimedia que se está transportando
Sequence number	Número de secuencia. Utilizado por el receptor para detectar paquetes perdidos o desordenados.
Timestamp	Permite la sincronización de diferentes streams, así como la reproducción de las muestras en el intervalo apropiado.
SSRC	Identificador de la fuente de sincronización. Identifica de manera única una sola fuente en un stream RTP.
CSRC	Identificador de fuente contribuyente. Se utiliza sólo cuando varios streams pasan por un mezclador.

Tabla 7: Descripción de campos RTP

2.3.3.2 RTCP

Proporciona información de control asociada con un flujo de datos RTP transmitiendo mensajes de control periódicos a todos los participantes de la sesión.

Informa de estado de la red, indicando los paquetes que se pierden, un cambio de codificación en el caso de que la calidad se degrade, o un aumento del tamaño del buffer de recepción si aumenta el retardo.

Por otra parte facilita la sincronización, indicando el tiempo en que se deben reproducir las muestras RTP.

Finalmente presenta un mensaje para indicar que uno de los participantes abandona la comunicación.

Los tipos de mensajes se muestran en la Tabla 8:

Mensaje	Descripción
SR	"Sender report". Especifica el número de paquetes y octetos enviados. También envía la marca de tiempos real.
RR	"Receiver report". Contiene información de la calidad de servicio de la comunicación, como paquetes recibidos, perdidos, etc..
SDES	"Source description". Identifica a la fuente.
BYE	Indica que uno de los participantes abandona la comunicación.
APP	Paquetes específicos de una aplicación.

Tabla 8: Tipo de mensajes RTCP

2.4 Arquitectura TISPAN[11] para IPTV[12]

TISPAN es un grupo de estandarización dentro de ETSI[13]. Se encarga principalmente de la creación de especificaciones de redes de próxima generación, NGN[14] a partir de ahora.

IPTV es un sistema que distribuye señales de televisión y vídeo a través de Internet basándose en el protocolo IP. Permite visualizar vídeos bajo demanda, televisión en directo o diferida, entre otras cosas.

TISPAN define dos soluciones IPTV. Una basada en las NGN pero no en IMS y otra basada en IMS. De aquí en adelante, nos centraremos en la solución basada en IMS, pues es la que nos interesa.

11 TISPAN: Telecommunication and Internet Converged Services and Protocols for Advanced Networks.

12 Internet Protocol Television

13 European Telecommunications Standards Institute

14 Next Generation Network

IPTV basada en IMS [11] [12] emplea el protocolo SIP para la señalización y RTSP para el control de vídeo. Esta integración está definida en el documento **Controlling Session Initiation Protocol (SIP)-Established Content Delivery Channels Using the Real-Time Streaming Protocol (RTSP)[13]**. Esta especificación nos presenta dos enfoques, uno "ligero" que prescinde de las peticiones SETUP y TEARDOWN de RTSP y otro que utiliza la totalidad del protocolo. En ambos casos se emplea SIP/SDP para establecer el flujo RTSP. En adelante, nos centraremos en el enfoque ligero, dado que es más sencillo y contiene la información que se quiere mostrar.

La ilustración 6 contiene el flujo de un establecimiento de sesión:

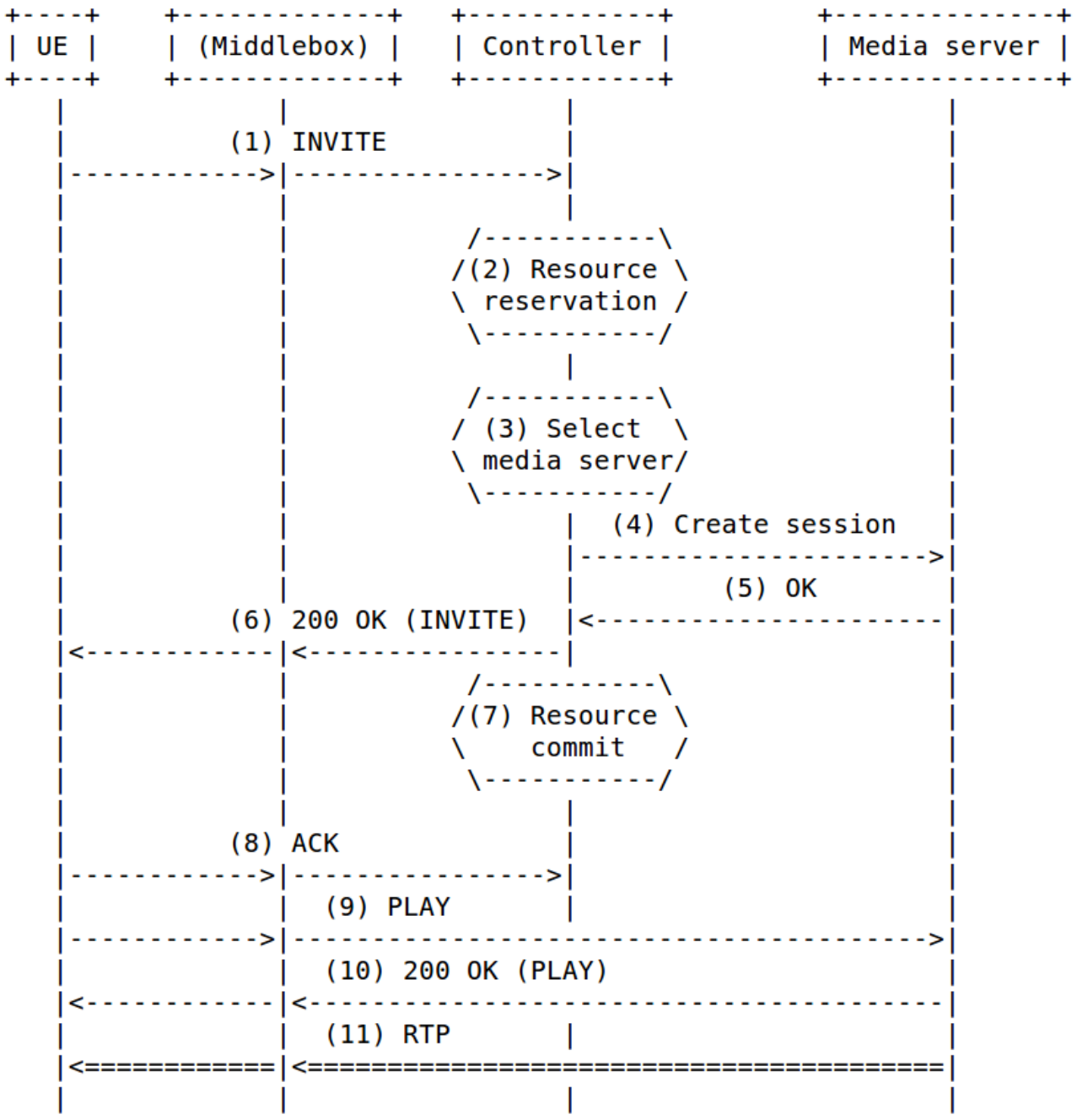


Ilustración 6: Establecimiento de sesión SIP + RTSP (fuente : https://tools.ietf.org/html/draft-lindquist-sip-rtsp-02#section-7)

Podemos ver cómo el equipo de usuario envía un mensaje tipo INVITE al servidor de media para iniciar la sesión. Con este mensaje se establecer el flujo RTP a través de la información contenida en la oferta SDP. Cuando el servidor de media acepte el mensaje INVITE, se establecerá la sesión.

Para empezar a recibir el flujo, el cliente deberá enviar la petición PLAY del RTSP. Una vez

que el servidor de media reciba la petición PLAY empezará a transmitir el contenido.

La ilustración 7 muestra el flujo de finalización de una sesión

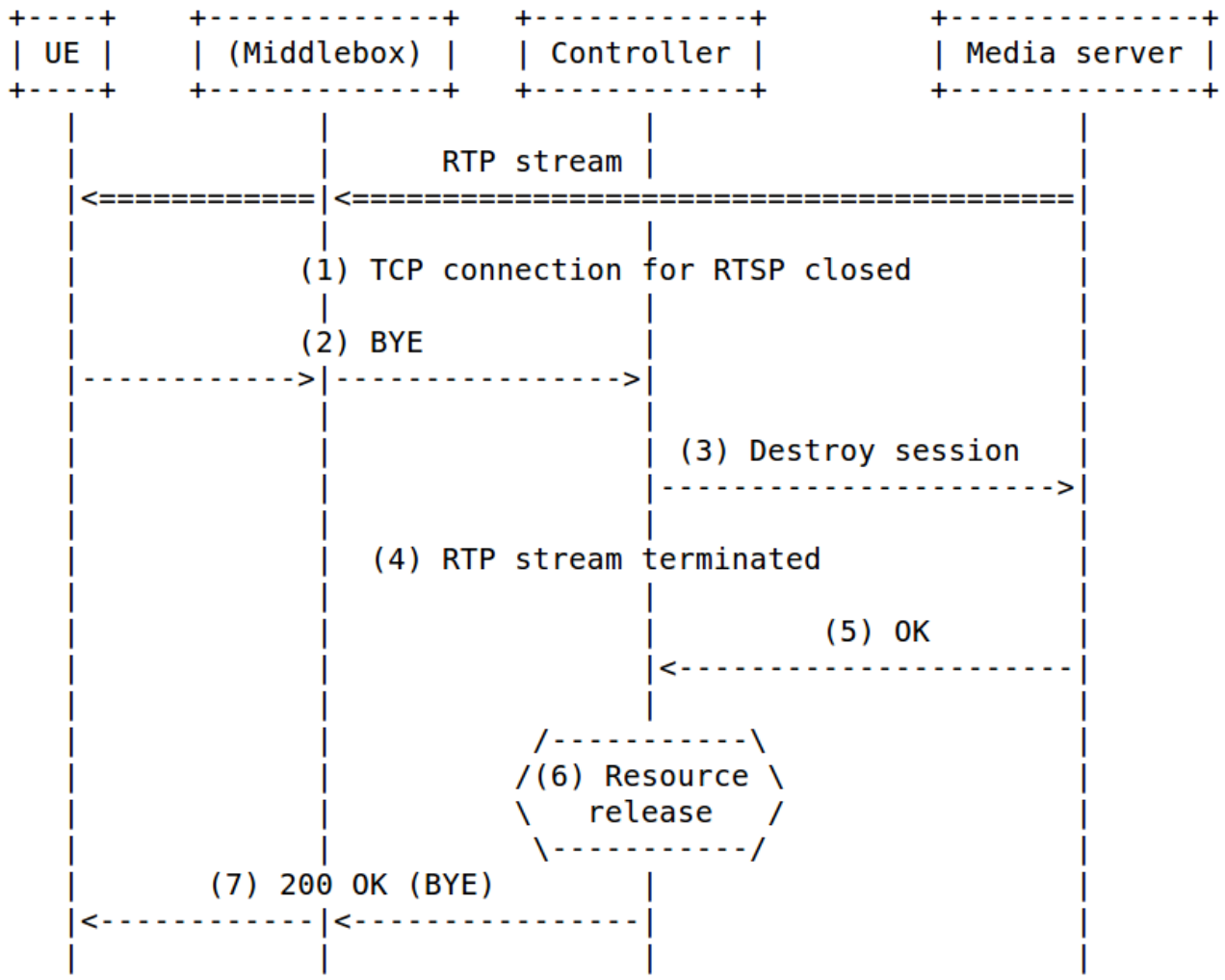


Ilustración 7: Finalización sesión SIP + RTSP (fuente: file:///home/usuario/proyecto/imagenes/lindFin.png)

En la imagen se puede ver cómo el equipo de usuario envía un mensaje BYE del protocolo SIP cuando quiere finalizar la sesión.

Esta especificación presenta un solapamiento de funciones entre la señalización SIP y RTSP La principal se da en el establecimiento de sesión para la transmisión de vídeo. Si nos fijamos en la ilustración 6, en el paso 1, SIP establece los parámetros necesarios para realizar la transmisión. En el paso 4, estos parámetros se vuelven a establecer a través de RTSP.

2.5 Conclusiones

En este capítulo se ha ofrecido una visión de los principales protocolos empleados en nuestra solución.

Por una parte hemos visto los protocolos empleados en el plano de control, SIP y SDP y por otra parte los protocolos empleados en el plano de datos, RTP y RTCP.

Finalmente hemos visto una especificación existente en la que podríamos basar nuestra solución. Sin embargo, para evitar los inconvenientes comentados en el apartado anterior, este proyecto se basa en un único sistema de señalización : SIP.

En los siguientes capítulos se describe el uso de los diferentes protocolos.

Capítulo 3

Herramientas utilizadas en el proyecto

3.1 Introducción

El objetivo de esta sección consiste en describir todas las herramientas utilizadas para el desarrollo del proyecto.

Empezaremos con el Open IMS core, que ha sido un elemento clave en el entorno de pruebas, al facilitarnos el entorno IMS.

A continuación detallaremos la librería live555. Tanto en el cliente como en el servidor, toda interacción con la red IMS, así como la transmisión de vídeo se basa en esta librería.

Seguiremos con MySQL, que se trata de la base de datos seleccionada para almacenar información de los vídeos que va a manejar el servidor.

Finalmente presentaremos la librería libVLC, que permite al cliente mostrar el vídeo recibido.

3.2 Open IMS

Open IMS Core [14] consiste en un proyecto de código abierto desarrollado por FOKUS (Fraunhofer institute Für Offene Kommunications Sisteme) . Contiene una implementación de

las entidades funcionales de la arquitectura IMS: las Call Session Control Functions y el Home Subscriber Server, todos ellos basados en código abierto (SIP Express Router y MySQL).

La ilustración 8 muestra los 4 elementos fundamentales que conforman Open IMS core:

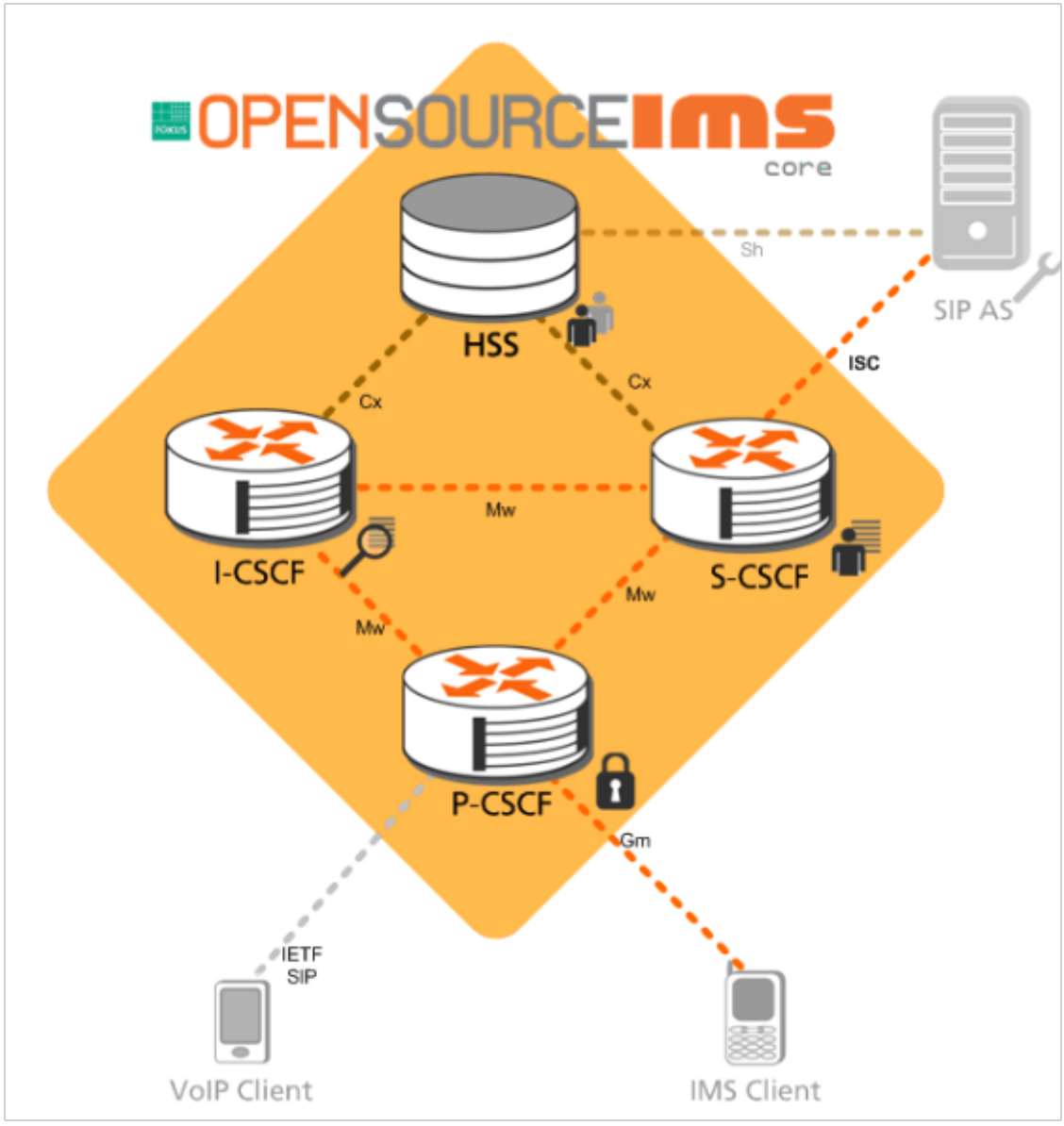


Ilustración 8: Componentes Open IMS Core (fuente: http://www.openimscore.org/)

Open IMS core proporciona un entorno de pruebas de la red IMS, facilitando el testeo de servicios basados en esta red. Además, proporciona una interfaz gráfica que facilita la gestión de usuarios, servicios y configuración de red.

El código fuente está disponible en su página web. También provee una máquina virtual donde el núcleo se encuentra previamente instalado y listo para probar.

Para que Open IMS core funcione correctamente precisa de MySQL como base de datos y de un servidor DNS. Ambos se comentan en los siguientes apartados.

Configuración

En el directorio */opt/OpenIMSCore/ser_ims/cfg* se encuentran las configuraciones de los elementos *Call Session Control Function* que componen Open IMS core:

pcscf.cfg : fichero de configuración del elemento P-CSCF.

icscf.cfg: fichero de configuración del elemento I-CSCF.

scscf.cfg: fichero de configuración del elemento S-CSCF.

Estos ficheros se proporcionan configurados con "open-ims.test" como dominio por defecto para la loopback local. Esta configuración (dominio e IP) se puede modificar. Para ello se facilita el script */opt/OpenIMSCore/ser_ims/cfg/configurator.sh* o bien, se puede modificar manualmente.

En el directorio */opt/OpenIMSCore/FHoSS/deploy* se encuentra la configuración del elemento HSS (Home Suscriber Server). Permite, entre otras cosas, modificar la IP y el puerto de la interfaz gráfica del HSS. Inicialmente es la 127.0.0.1 y el puerto 8080.

Finalmente, la configuración inicial incluye 2 usuarios, Alice y Bob. Se pueden crear tantos usuarios como se quiera a través de la interfaz gráfica del HSS, a la que se accede vía web.

Uso de Open IMS core

Open IMS Core se ha empleado como entorno de pruebas para nuestra aplicación servidor y cliente. Open IMS Core nos ha facilitado las funciones de registro de usuarios, inicio, mantenimiento y negociación de sesión.

3.3 Librería live555

Live555[15] es una librería desarrollada en C++ que permite manejar streaming multimedia empleando protocolos estándares(RTP/RTCP, RTSP y SIP). Así mismo, puede ser utilizada para enviar, recibir y procesar MPEG, H264, H.263+, DV y JPEG. Permite ser ampliada fácilmente para soportar otros formatos de audio o vídeo. Finalmente puede ser empleada para desarrollar clientes y servidores RTSP o SIP.

Esta librería ha sido utilizada para desarrollar aplicaciones como "Live555 Media Server", "Live555 Proxy Server" y vobStreamer.

Las librerías están disponibles para Linux, Windows y QNX bajo licencia GNU LGPL.

Está estructurada en tres librerías: UsageEnvironment, Groupsock y Media. Finalmente, presenta un conjunto de programas de prueba que permite comprobar el alcance de la librería.

A continuación se describen cada una de las librerías.

15 http://live555.com/

UsageEnvironment & TaskScheduler

Estas dos clases son la base de la librería. Nos permiten manejar el entorno.

Proveen las siguientes funciones:

- programación de tareas
- asignación gestores para eventos de lectura asíncrona
- Mostrar errores y avisos por la salida estándar.
- Define una estructura tipo “Hashtable” (tabla hash), utilizada en el resto del código.

Groupsock

Permite comunicaciones multicast y unicast para el envío y recepción de datos. Encapsula interfaces de red y sockets.

LiveMedia

Conjunto de clases que se encargan del flujo de media y de los formatos. Permiten transmitir audio y vídeo en diferentes formatos.

También presenta un cliente SIP, permitiendo realizar un registro, iniciar una sesión y detenerla.

Uso de la librería live555

La librería live555 se ha empleado tanto en el lado del cliente como en el lado del servidor.

En el lado del servidor se ha empleado para manejar el intercambio de mensajes SIP con IMS y para la transmisión de media.

En el lado del cliente se ha empleado para manejar el intercambio de mensajes SIP con IMS.

Se ha empleado live555 en el proyecto porque sus módulos ofrecen una forma sencilla de enfrentar el envío y recepción de mensajería a través de sockets y el envío/ recepción de vídeo. También se ha tenido en cuenta su licencia y su amplia documentación.

3.4 MySQL

MySQL [15] consiste en un sistema de gestor de bases de datos relacional, multihilo y multiusuario. Es muy conocido y ampliamente usado por su simplicidad y rendimiento.

Presenta una licencia dual. Por un lado se ofrece bajo licencia GNU GPL, disponible para aquellos programas que lo incorporen y presenten esta misma licencia y por otro lado existe la licencia comercial, disponible para aplicaciones comerciales.

Está disponible para múltiples plataformas.

MySQL se ha empleado en el servidor para almacenar la información de contenidos disponible.

Se ha elegido MySQL porque, además de cubrir nuestras necesidades sobradamente, su licencia es gratuita y su instalación sencilla.

3.5 LibVLC

LibVLC [16] consiste en un framework que permite insertar capacidades multimedia en una aplicación. Se trata de la interfaz de programación externa del programa VLC.

VLC es un reproductor multimedia de código abierto y gratuito. Codifica, transmite y reproduce audio y vídeo. Funciona sobre todas las plataformas populares y puede leer casi todos los archivos, Cds, DVDs, tarjetas capturadoras y transmisiones de red. Es creado por voluntarios de la comunidad VideoLAN.

Todas las características propias de VLC están disponibles a través de libVLC.

LibVLC se ha empleado en el cliente del servidor de vídeo. El cliente emplea la librería libVLC para reproducir el vídeo enviado por el servidor vía RTP/RTCP.

Se ha elegido este framework debido a que cubre perfectamente nuestras necesidades, además de otras cualidades como su uso extendido, su licencia, el gran número de formatos interpretados, así como su fácil instalación y accesibilidad.

3.6 BIND9

Bind9 [17] es un servidor DNS. Un servidor DNS es un servidor que, entre otras cosas, traduce nombres de dominio a direcciones IPs y viceversa. Presenta 3 modos de funcionamiento: maestro, esclavo y servidor caché DNS.

En el modo maestro, el servidor se comportará como un auténtico servidor DNS en nuestra red local. Atenderá las peticiones de resolución de direcciones pertenecientes a la red local y reenviará a servidores DNS externos el resto.

En el modo esclavo actuará como un servidor espejo de un servidor DNS maestro.

En el modo servidor caché DNS, el servidor se comportará como si fuera un auténtico servidor DNS para nuestra red local aunque no sea un servidor propiamente dicho. Cuando reciba una petición, la trasladará al DNS maestro obteniendo la respuesta, almacenándola y proporcionándosela quien hizo la petición. En caso de recibir una segunda vez dicha petición,

recurrirá a su memoria caché para dar la respuesta.

El archivo de configuración del DNS se llama *named.conf* y se encuentra en */etc/bind/named.conf*. Este fichero hace referencia a una serie de ficheros que se encuentran en la misma ruta y se comentan a continuación:

- named.conf.option: Contiene las opciones genéricas. Aquí se define el directorio por defecto para ubicar los ficheros de zonas.
- named.conf.local: Contiene la especificación particular del servidor DNS. Incluye las zonas y las zonas inversas que va a manejar así como el nombre de los archivos donde se definen.
- Archivos para zonas: Contienen la información relativa a las zonas que el servidor va a manejar.
- Archivos para zonas inversas: Contienen la información relativa a las zonas inversas que el servidor va a manejar.

Open IMS Core precisa de un servidor DNS maestro. Se ha elegido bind9 debido a su amplia distribución y la gran cantidad de documentación que existe en la web.

3.7 Conclusiones

En este capítulo se han descrito las herramientas empleadas para el desarrollo del proyecto. Todas las herramientas son necesarias y cumplen con los requerimientos iniciales del proyecto. En el siguiente capítulo se explica como se han integrado tanto en el servidor como en el cliente.

Capítulo 4

Diseño e implementación del servidor VoD para redes IMS

4.1 Introducción

El capítulo presente realiza una descripción completa del diseño e implementación del servidor y del cliente.

Empezaremos con el diseño del servidor de VoD, ofreciendo primero una visión conceptual para luego entrar en la interacción con el cliente.

Finalmente se mostrará la implementación, dedicando un primer apartado al servidor de VoD y el siguiente al cliente.

4.2 Diseño

El servidor de VoD consiste básicamente en un usuario de IMS que envía información multimedia a otros usuarios que así lo soliciten vía RTP/RTCP permitiéndoles su manipulación. Además, dispone de una conexión a base de datos, donde almacena el contenido multimedia que puede ofrecer a sus clientes.

El cliente se trata de otro usuario de IMS que inicia una sesión con el servidor para que este último le envíe un determinado contenido multimedia.

Una visión de la arquitectura de nuestro sistema para ofrecer vídeo bajo demanda es la que se muestra en la ilustración 9.

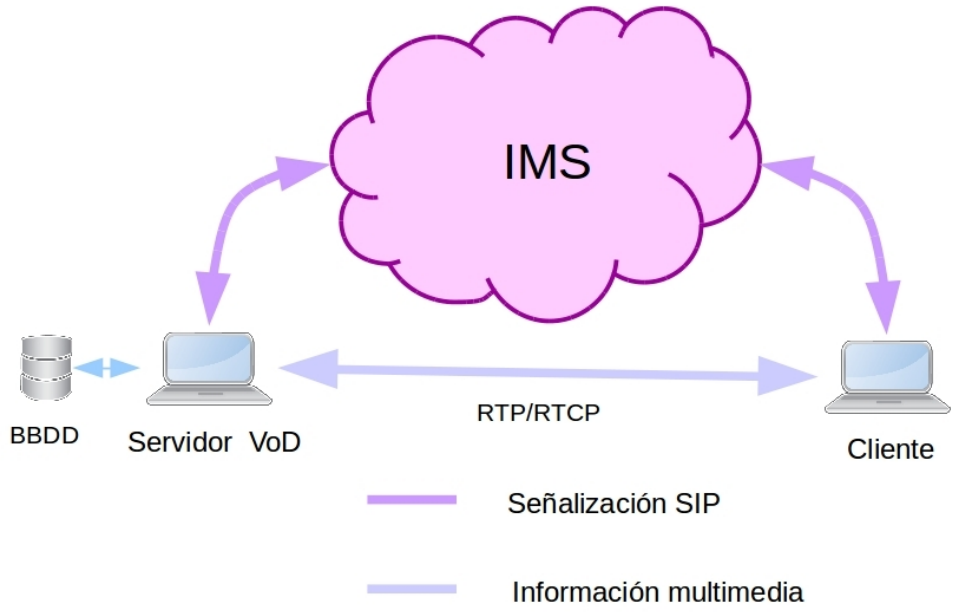


Ilustración 9: Visión de la arquitectura de nuestro sistema

4.2.1 Base de Datos

El servidor de VoD presenta una base datos que contiene información sobre el contenido multimedia que pueden solicitar los clientes.

El servidor realiza dos acciones:

-Comprobar si existe el vídeo solicitado por el cliente en el formato multimedia indicado.

-Obtener la ubicación del vídeo solicitado por el cliente.

La base de datos empleada es MySQL estando estructurada como sigue:

Tabla vídeos

Contiene información sobre los vídeos que ofrece el servidor. Su estructura es como sigue:

Campo	Descripción
Id	Identificador único del vídeo
Name	Nombre del vídeo
Path	Ubicación del vídeo en el servidor

Tabla videoPayLoad

Almacena los formatos multimedia en los que están disponibles los vídeos. Tiene la siguiente estructura:

Campo	Descripción
Id	Identificador del vídeo
Payload	Codec en el que el vídeo está disponible

4.2.2 Usuario IMS

Tanto el servidor de VoD como el cliente son un usuarios de IMS, por lo que deben registrarse en la red IMS. Para ello enviará un mensaje SIP de petición tipo REGISTER siguiendo el esquema que se muestra en la ilustración 10.

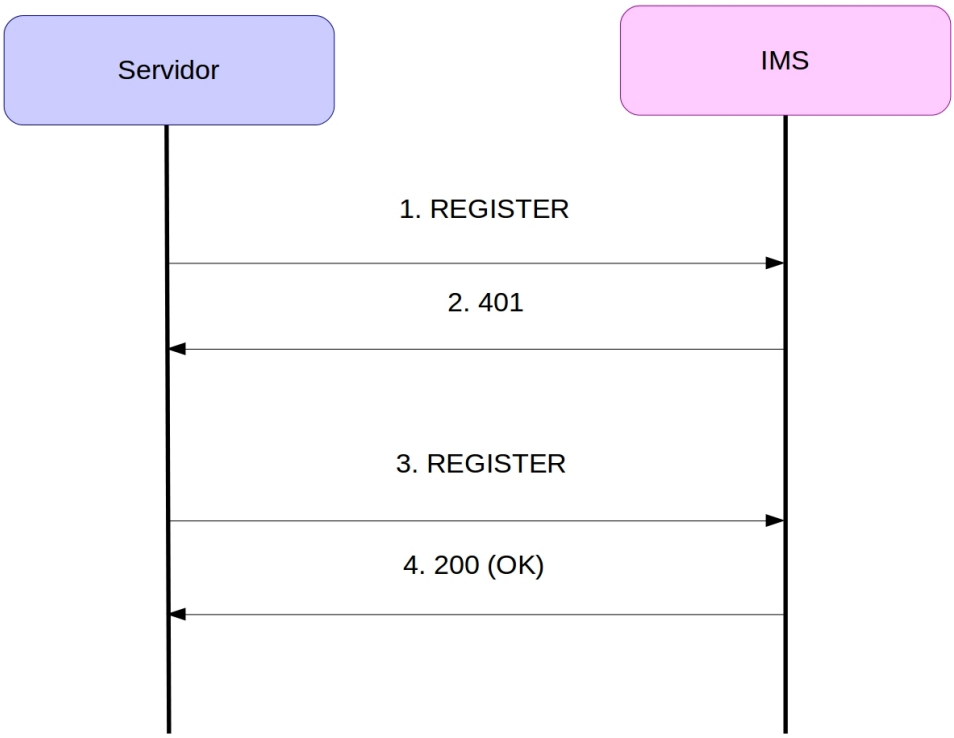


Ilustración 10: El servidor se registra en IMS

Cuando finalicen su trabajo, se des-registrará de la red IMS enviando un mensaje SIP de tipo REGISTER estableciendo la cabecera *Expires* a 0. El mensaje se muestra en la ilustración 11.

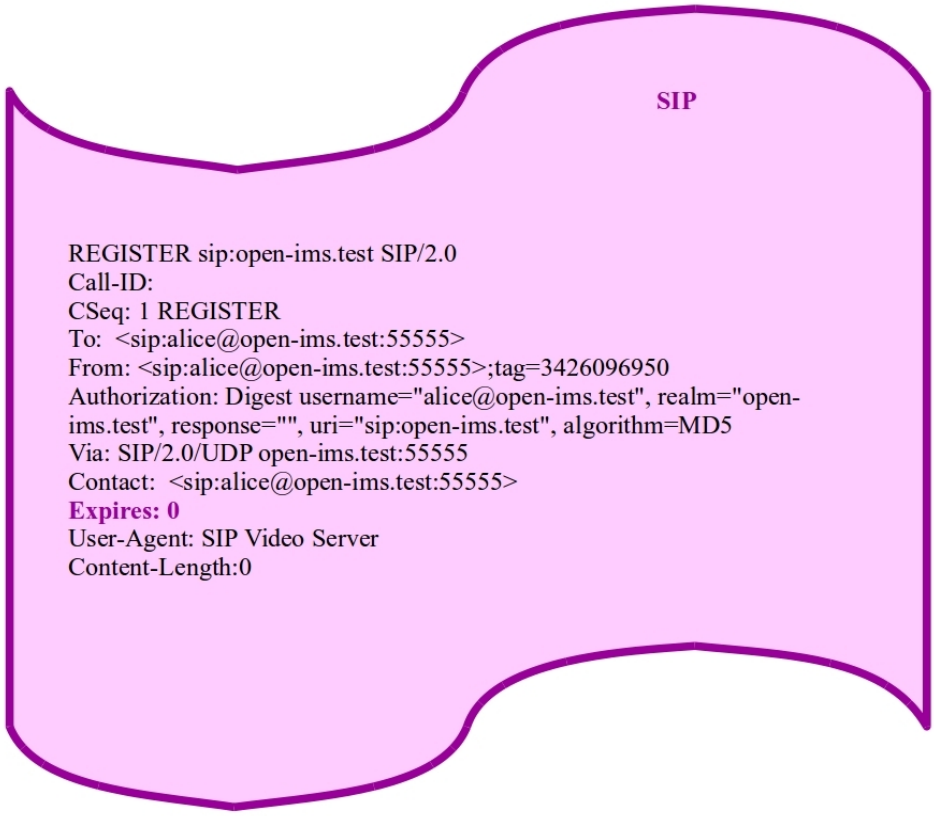


Ilustración 11: Mensaje para des-registrarse de IMS

Una vez registrado,el servidor quedará a la espera de peticiones de inicio de sesión por parte de los diferentes clientes. Una petición de inicio de sesión consiste en un mensaje SIP de petición tipo INVITE con cuerpo SDP, que contendrá la información relativa al vídeo y al formato de media.

El cliente debe indicar en qué formato desea recibir el media y los puertos destinados para la recepción.

La información se encapsula tal y como muestra la ilustración 12:

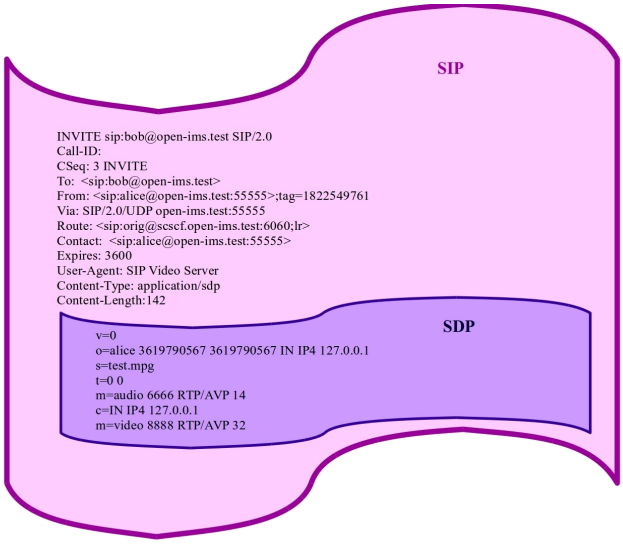


Ilustración 12: Mensaje SIP INVITE

La Tabla 9 continuación se muestran qué campos SDP se emplean y cómo se emplean:

Campo	Descripción
s	Nombre del vídeo solicitado
m	Contiene el tipo de media y la información sobre transporte siguiendo el esquema: <contenido> <puerto> <transporte> <lista de formatos de media> Puede aparecer una o dos veces en función del tipo de formato de media que se haya solicitado. <contenido> puede ser audio o vídeo. <puerto> puerto a donde se ha de transmitir el stream. <transporte> siempre RTP/AVP <lista de formatos de media> Sólo se admiten MPEG Transport Stream -TS o MP2T- (14 para audio y 32 para vídeo).
c	Contiene la dirección donde se debe enviar el media siguiendo el esquema: <tipo red> <tipo dirección> <dirección>

Tabla 9: Uso de los campos SDP por el servidor de VoD

Este mensaje será enviado por uno de los clientes del servidor. Cuando el servidor reciba este mensaje comprobará el vídeo que solicita el cliente y el formato. En caso de que disponga del vídeo solicitado con el formato indicado, aceptará la sesión con un mensaje OK. La ilustración 13 muestra el proceso.

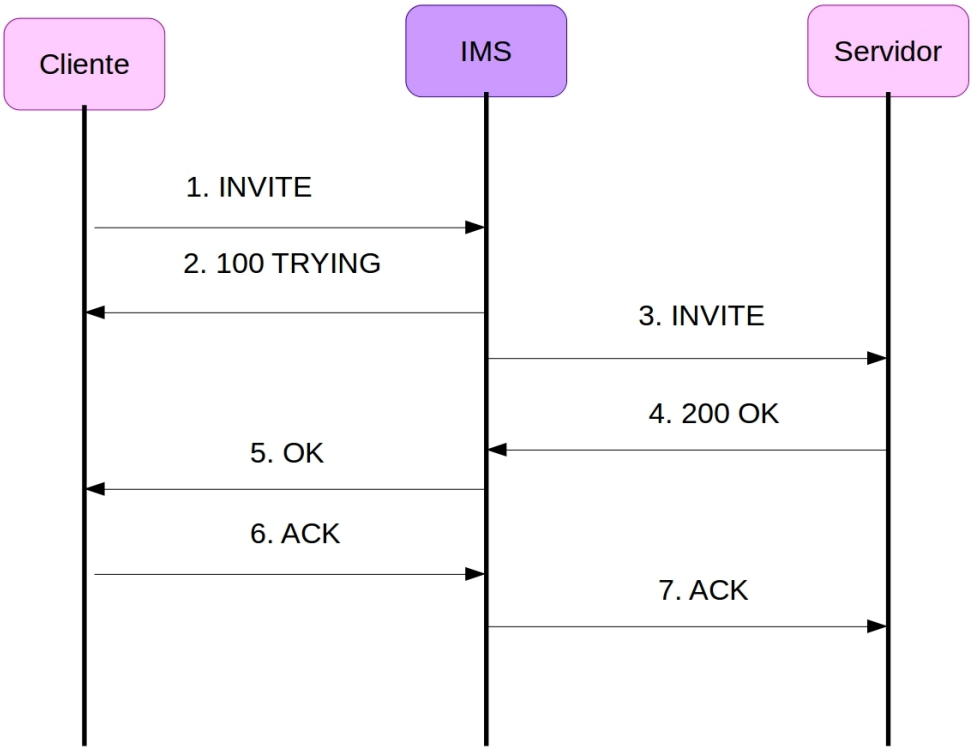


Ilustración 13: Servidor acepta petición del cliente

En caso contrario, es decir, si el servidor no dispone del vídeo o no tiene el formato solicitado , el servidor rechazará la sesión con un mensaje tipo respuesta con código de estado 404 indicando que el vídeo solicitado no se encuentra. El proceso se muestra en la ilustración 14.

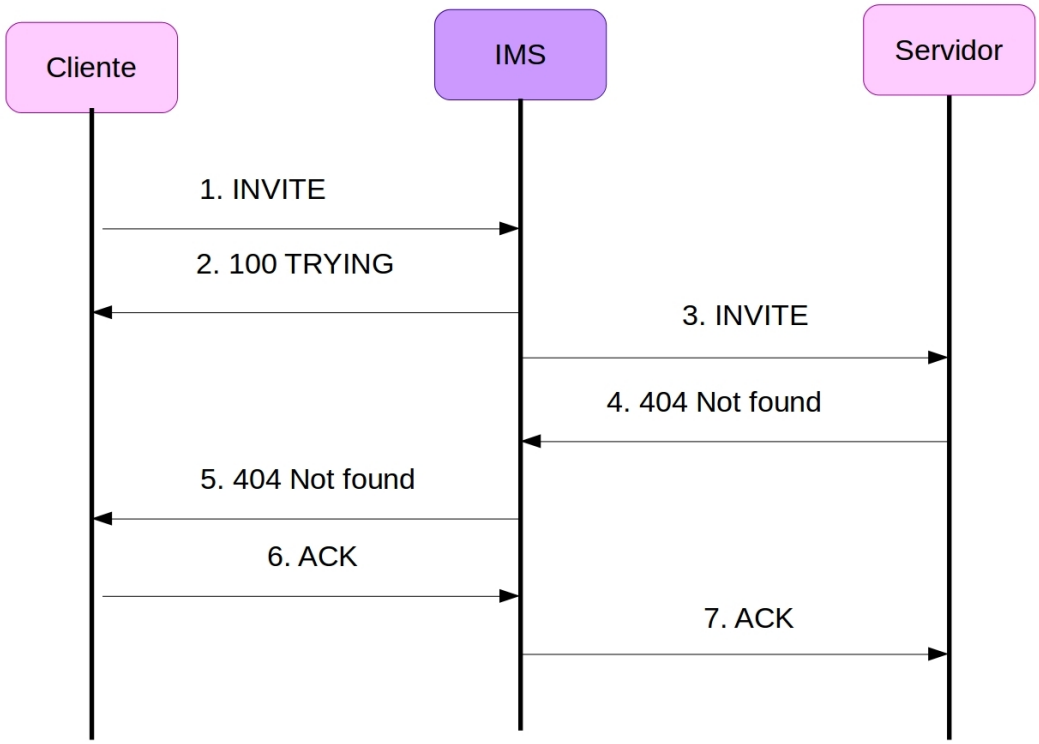


Ilustración 14: Servidor rechaza petición de cliente

Una vez se ha establecido una sesión entre el cliente y el servidor, éste último queda la espera de recibir instrucciones del cliente para reproducir el media.

Las instrucciones que pueden recibir son las que se describen en la Tabla 10:

Instrucción	Descripción
PLAY	Inicia la reproducción del media
PAUSE	Pausa la reproducción de media
STOP	Detiene la reproducción del media.

Tabla 10: Comandos que admite el servidor

Estas instrucciones llegan en forma de mensajes SIP de petición de tipo INFO encapsulándose en el

cuerpo del mensaje, como se muestra en la ilustración 15:

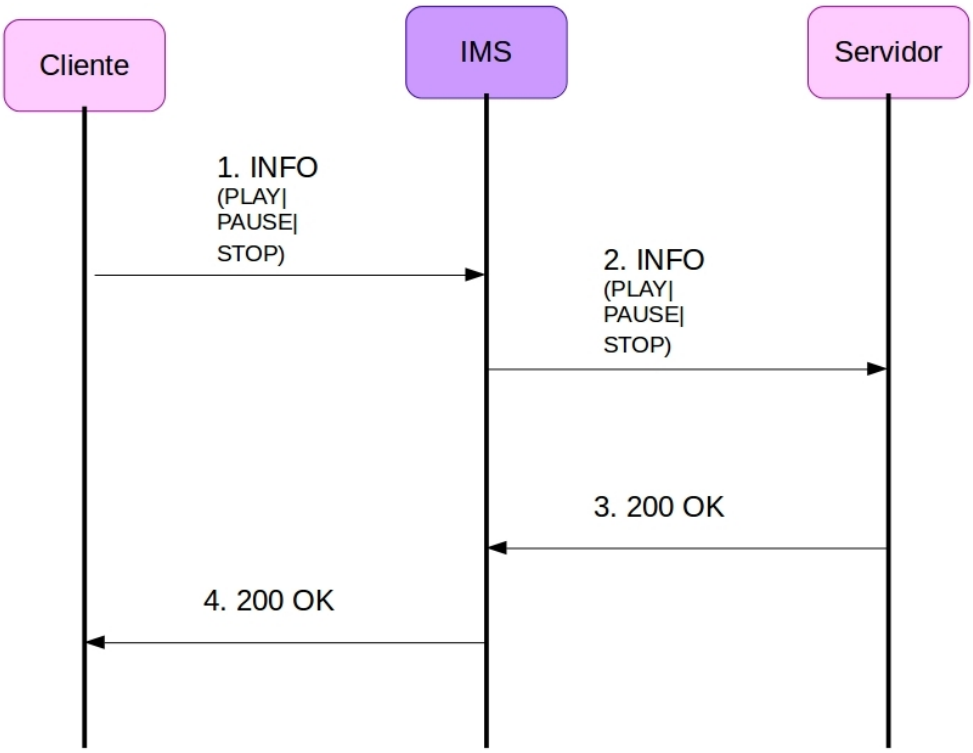


Ilustración 15: Envío de comando de cliente a servidor

Una vez que el cliente finaliza la visualización del vídeo, enviará un mensaje de petición tipo BYE, indicando que finaliza la sesión.

4.2.3 Conexiones con clientes

La transmisión de vídeo se realiza usando RTP/RTCP entre cliente y servidor. Los parámetros de esta comunicación se especifican a través de SDP durante el inicio de sesión como se explicó en el apartado anterior.

Una vez iniciada una sesión con un cliente, el servidor queda a la espera de recibir instrucciones del cliente para la transmisión del vídeo como también se explicó en el apartado anterior.

El cliente consumirá el media enviado por el servidor a través de VLC.

El servidor ofrece dos tipos de formatos multimedia, Transport Stream (TS) y Moving Picture Experts Group 2 (MPEG-2).

Archivos TS

Se crea un stream para transmitir el vídeo. Tanto el servidor como el cliente deberán reservar dos puertos para el envío/recepción del media.

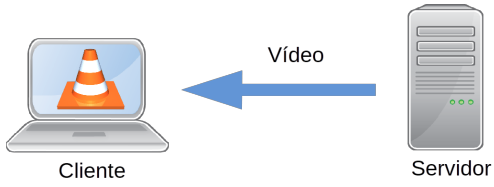


Ilustración 16: Multimedia TS

Archivos MPEG2

Se crea un stream para transmitir vídeo y otro para el audio. El servidor y el cliente deben reservar cuatro puertos para enviar/recibir el media, dos para el audio y otros dos para el vídeo.

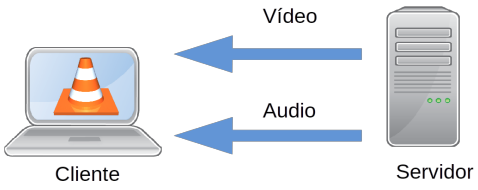


Ilustración 17: Multimedia MPEG2

4.3 Implementación

4.3.1 Servidor

A continuación se describe el funcionamiento del servidor y los módulos que se han realizado para su implementación.

4.3.1.1 Funcionamiento

El funcionamiento del servidor se describe en el esquema que muestra la ilustración 18:

Ilustración 18: Funcionamiento del servidor (I)

Carga de configuración

Lo primero que hace el servidor es cargar la configuración. La configuración se realiza a través de un fichero ubicado en el mismo directorio que la aplicación llamado *configuracion.txt*.

Los datos que componen la configuración del servidor se muestran en la Tabla 11:

Nombre	Descripción
APPNAME	Nombre de la aplicación
URI	URI de la red IMS
ADDRESS	Dirección IP donde se encuentra el P-CSCF de la red IMS del usuario.
SOURCEPORT	Puerto a través del que se va a realizar la comunicación con la red IMS.
USER	Usuario de la red IMS con el que nos vamos a registrar.
PWD	Contraseña del usuario de la red IMS con el que nos vamos a registrar.
IMSPORT	Puerto del P-CSCF de la red IMS del usuario.
BBDDIP	IP de la base de datos
BBDDPORT	Puerto de la base de datos
BBDDUSER	Usuario de la base de datos
BBDDPWD	Clave de la base de datos

Tabla 11: Configuración del servidor

Registro en la red IMS

El usuario debe registrarse en la red IMS tal y como se describe en la ilustración 8.

Espera de clientes

Una vez registrado en la red IMS, el servidor se queda a la espera de clientes.

El funcionamiento del servidor una vez que recibe una petición de cliente se muestra en la ilustración 19:

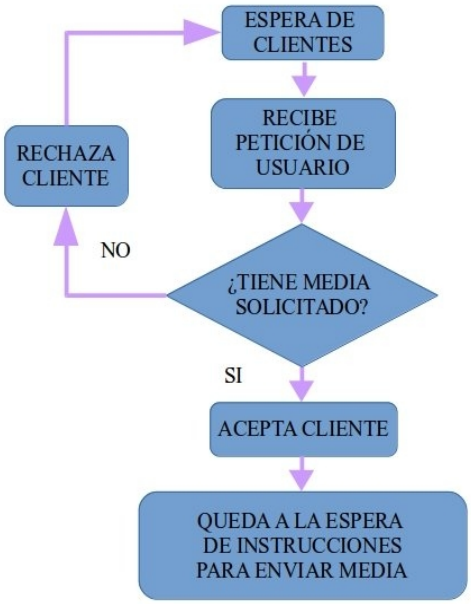


Ilustración 19: Funcionamiento del servidor (II)

El servidor recibirá una petición de inicio de sesión de parte de un cliente. Esta petición contendrá el nombre del media y el formato que desea el cliente. El servidor debe comprobar si tiene el contenido multimedia solicitado. De ser así, aceptará el cliente y por lo tanto la sesión, quedando a la espera de instrucciones para enviar el media. En caso contrario, el servidor rechazará la sesión y por tanto al cliente.

4.3.1.2 Módulos

El servidor de VoD consta principalmente de un módulo usuario IMS, de un módulo servidor, de un módulo conexión con la base de datos y de uno de media. Se describen a continuación.

Módulo usuario IMS

Es el encargado de realizar todas las tareas de iteración con la red IMS gestionando el envío y recepción de mensajería SIP.

Sus funciones principales son:

- Registro.

-Recepción/Envío y gestión de mensajes.

La ilustración 20 refleja su funcionamiento:

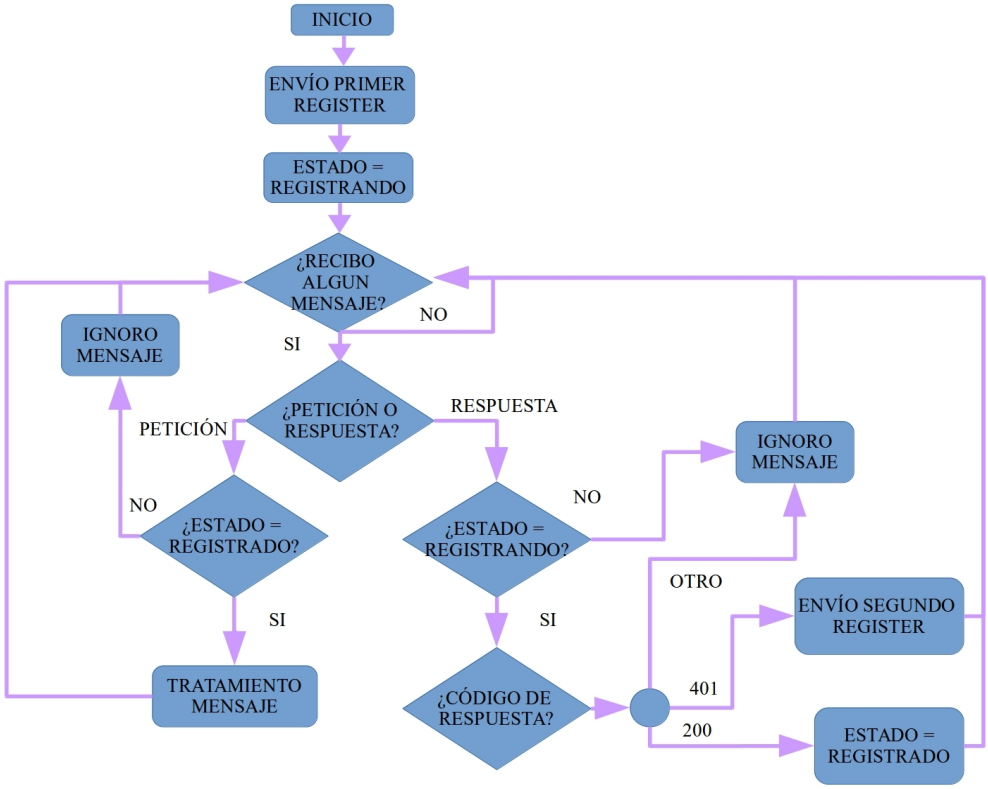


Ilustración 20: Módulo usuario IMS

Librerías empleadas

Este módulo se apoya en la librería live555 para la gestión de sockets. Más concretamente, emplea los módulos *UsageEnvironment y GroupsockHelper*.

El módulo *GroupsockHelper* se emplea para crear el socket de la aplicación destinado a la comunicación con la red IMS.

El módulo *UsageEnvironment* se emplea para controlar la salida por defecto del programa,contribuye en la creación del socket de comunicación con IMS y se encarga de controlar los eventos de entrada/salida, más concretamente la recepción de mensajes.

Módulo servidor

Módulo encargado de la gestión de clientes y de la mensajería destinada al manejo de media.

Recibe los mensajes de petición tipo INVITE, BYE e INFO que le pasa el módulo usuario IMS. A continuación se describe su comportamiento en función del mensaje recibido.

INVITE : Interpreta el mensaje para obtener el media solicitado. Utiliza esta información para interaccionar con el módulo base de datos y poder comprobar si acepta o rechaza la sesión. En caso de aceptar la sesión añade el cliente a su lista de clientes activos. La ilustración 21 muestra el tratamiento del mensaje INVITE por parte del servidor:

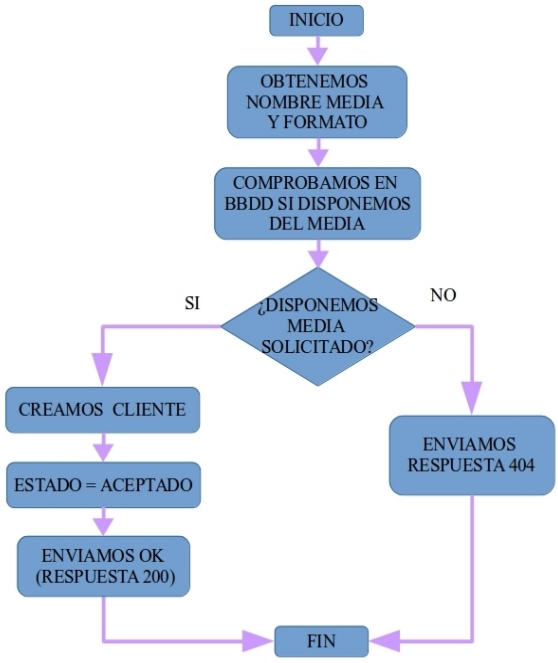


Ilustración 21: Tratamiento mensajes INVITE

BYE: Cuando recibe este tipo de mensajes elimina el cliente correspondiente de la colección. La Ilustración 22 contiene el tratamiento del mensaje BYE:

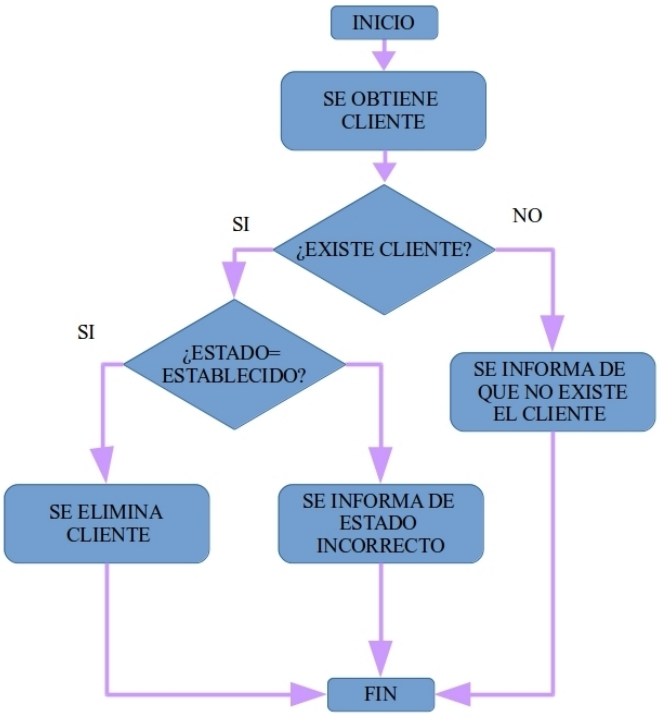


Ilustración 22: Tratamiento mensajes BYE

INFO: Estos mensajes son específicos de la gestión de vídeo. Pueden contener los siguientes comandos: PLAY, PAUSE y STOP. Cuando el módulo servidor recibe este tipo de mensaje comprueba cuál de sus clientes lo envió. Una vez encontrado el cliente, interpreta el mensaje y pasa el comando a su módulo media para que se ejecute la instrucción indicada. El módulo media se describe a continuación. El tratamiento de mensajes tipo INFO se describe en la ilustración 23.

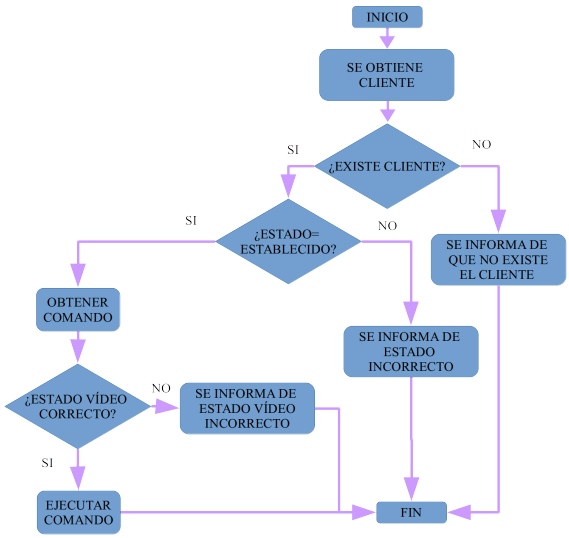


Ilustración 23: Tratamiento mensajes INFO

Librerías empleadas

Emplea la clase *HashTable* ofrecida por el módulo *UsageEnvironment* de la librería live555 para manejar los clientes conectados, indexándolos por nombre de usuario.

Módulo Media

Es el encargado de realizar el envío de media a los clientes a través de una conexión RTP/RTCP. Por otra parte recibe la mensajería tipo INFO, realizando la acción solicitada sobre el media.

Librerías empleadas

Utiliza el módulo *liveMedia* de la librería live555 para la transmisión de vídeo, además de los módulos *UsageEnvironment y GroupsockHelper*.

El módulo *liveMedia* se emplea para el envío de los vídeos. Cabe recordar que la aplicación permite el envío de dos tipos de archivos: TS y MPEG2. En ambos casos existirá una instancia de la clase *RTCPInstance* que se encarga de la gestión de RTCP. El flujo RTP se gestiona de forma diferente en función del tipo de archivo que sea.

Para los vídeos tipo TS se emplea la clase *SimpleRTPSink*.

Los vídeos tipo MPEG2 emplean la clase *MPEG1or2AudioRTPSink* para el audio y la clase *MPEG1or2VideoRTPSink* para el vídeo. Por otra parte hace uso de la clase *MPEG1or2Demux* para demultiplexar el archivo en un stream de vídeo y otro de audio.

El módulo *GroupsockHelper* se emplea para crear los sockets para enviar el vídeo. En el caso de envíar un archivos tipo TS se emplearán 2 sockets, uno para RTP y otro para RTCP. En el caso de que se esté manipulando un archivo tipo MPEG2, se utilizarán 4, dos para el audio (uno para RTP y otro para RTCP) y otros dos para el vídeo (uno para RTP y otro para RTCP).

El módulo *UsageEnvironment* se emplea para controlar la salida por defecto del programa.

Módulo Base de datos

Es el encargado de la gestión de la información multimedia que el servidor es capaz de ofrecer a los clientes.

Realiza dos consultas a la base de datos:

- Una consulta para comprobar si existe el vídeo solicitado por el cliente en el codec indicado.

- Una segunda consulta para obtener la ubicación del vídeo.

Librerías empleadas

La aplicación emplea la librería *MySQL Connector/C++* como interfaz de comunicación con la base de datos.

4.3.2 Cliente

4.3.2.1 Funcionamiento

El funcionamiento del cliente se describe en la ilustración 24:

Ilustración 24: Funcionamiento cliente

A continuación se describe cada paso:

Carga de configuración

Lo primero que hace el cliente es cargar la configuración. La configuración se realiza a través de un fichero ubicado en el mismo directorio que la aplicación. La aplicación solicitará el nombre del fichero de configuración, de esta forma se podrán cargar fácilmente diferentes perfiles.

Los datos que componen la configuración del cliente se muestran en la Tabla 12.

Nombre	Descripción
APPNAME	Nombre de la aplicación. Valor que se envía en la cabecera *User-Agent* del mensaje SIP
SERVERUSER	Usuario que se registra en la red IMS como servidor y con el que el cliente debe iniciar sesión.
URI	URI de la red IMS
ADDRESS	Dirección IP donde se encuentra el P-CSCF de la red IMS del usuario.
SOURCEPORT	Puerto a través del que se va a realizar la comunicación con la red IMS.
USER	Usuario de la red IMS con el que nos vamos a registrar.
PWD	Contraseña del usuario de la red IMS con el que nos vamos a registrar.
IMSPORT	Puerto del P-CSCF de la red IMS del usuario.

Tabla 12: Configuración Cliente

Elección media

El cliente debe elegir el media disponible. Esta información se encuentra almacenada en un fichero llamado *videos.txt*. La aplicación muestra el contenido en pantalla y permite al cliente seleccionar el vídeo que desea.

Elección de formato

El cliente debe elegir el formato de media. Los formatos disponibles se encuentran almacenados en un fichero llamado *formatos.txt*. La aplicación muestra el contenido en pantall y permite al cliente seleccionar el formato de media que desea.

En función del formato seleccionado por el cliente solicitará uno o dos puertos para posteriormente poder realizar la transmisión del media

Registro en IMS

El usuario debe registrarse en IMS siguiendo el mismo proceso descrito en el servidor.

Inicio de sesión con el servidor

El cliente inicia la sesión con el servidor enviándole un mensaje tipo INVITE con la información del media deseado. El servidor comprobará si dispone del media solicitado y en caso afirmativo se

inicia la sesión. En caso de que el servidor no disponga del media, no se iniciará la sesión.

Visualización/Manipulación de vídeo

Una vez iniciada la sesión el cliente podrá visualizar y manipular el media.

La visualización a realizará a través del programa VLC.

La manipulación del media se realizará través de la línea de comandos, seleccionando una de las acciones que se ofrecen.

4.3.2.2 Módulos

El cliente consta principalmente de un módulo usuario IMS y de un módulo reproductor de media.

Usuario IMS

Es el encargado de realizar todas las tareas de iteración con la red IMS gestionando el envío y recepción de mensajería SIP.

Sus funciones principales son:

- Registro.

-Envío y recepción de mensajes. El cliente puede enviar los mensajes SIP de petición INVITE, INFO y BYE. A continuación describimos el comportamiento en cada caso particular.

INVITE: Una vez el cliente elige el media que quiere solicitar y su formato se lo indica al servidor a través de un mensaje tipo INVITE. Una vez el servidor acepte el mensaje, la sesión se establecerá y el cliente deberá enviar los diferentes comandos disponibles para su manipulación.

INFO: El cliente puede manipular el media a través de los comandos PLAY, PAUSE y STOP que se encapsulan en un mensaje SIP de petición tipo INFO.

BYE: Cuando el cliente desee terminar la sesión con el servidor le enviará un mensaje tipo BYE.

Librerías empleadas

Este módulo se apoya en la librería live555 para la gestión de sockets. Más concretamente, emplea los módulos *UsageEnvironment y GroupsockHelper*.

El módulo *GroupsockHelper* se emplea para crear el socket de la aplicación destinado a la comunicación con la red IMS.

El módulo *UsageEnvironment* se emplea para controlar la salida por defecto del programa,contribuye en la creación del socket de comunicación con IMS y se encarga de controlar los eventos de entrada/salida, más concretamente la recepción de mensajes.

Reproductor de media

Es el encargado de reproducir el media recibido en el cliente. Para ello se emplea el programa VLC.

Librerías empleadas

Este módulo emplea la librería LibVLC descrita en el apartado 3.5.

4.4 Conclusiones

El servidor de vídeo para redes con IMS y el cliente se han implementado siguiendo el diseño propuesto en el apartado 4.2. Usando las herramientas elegidas, hemos conseguido implementarlos con la funcionalidad deseada cumpliendo con nuestros objetivos iniciales. Ambos han necesitado el uso de una serie de librerías. A continuación, se exponen las conclusiones a las que hemos llegado después de su uso.

La librería *live555* se emplea en el módulo media y en el módulo usuario IMS.

La funcionalidad ofrecida por *live555* relacionada con el manejo del flujo multimedia ha cumplido con nuestras necesidades de cara a la implementación del módulo multimedia. Facilita la transmisión en tiempo real de contenidos en una gran cantidad de formatos permitiendo su manipulación. Como contra, sólo permite el uso de los *trick mode operations*[16] para un determinado formato de media, siendo su uso complejo.

A la hora de implementar el usuario IMS, se han echado en falta funcionalidades como por ejemplo un generador de mensajes SIP o la gestión de diálogos SIP. Se ha tenido que implementar un constructor de mensajes SIP y un manejador de diálogos SIP, que permitiera gestionar de forma correcta las cabeceras de los mensajes. La librería live555 tan sólo ofrece un cliente SIP con una funcionalidad bastante limitada. Por otra parte, cabe destacar la funcionalidad de esta librería relacionada con las comunicaciones. Su módulo de sockets nos ha facilitado la comunicación del usuario IMS con IMS.

Finalmente, la documentación relativa a *live555* ha resultado escasa y poco orientativa.

La implementación del servidor ha requerido la utilización de la librería *MySQL Connector/C++* para la comunicación con la base de datos. Esta librería nos ha facilitado el acceso a la base de datos. Cabe destacar su sencillez y la documentación disponible en internet.

En la implementación del cliente se utiliza la librería libVLC. Esta librería nos ha proporcionado las herramientas necesarias para que nuestro cliente pudiera mostrar el flujo o los flujos multimedia que recibe. El uso de esta librería ha resultado sencillo a pesar de que no existe una gran cantidad de documentación.

16 Avance y retroceso rápidos

Capitulo 5

PRUEBAS REALIZADAS

5.1 Entorno de pruebas

El entorno de pruebas conceptual es el que se muestra en la ilustración 25.

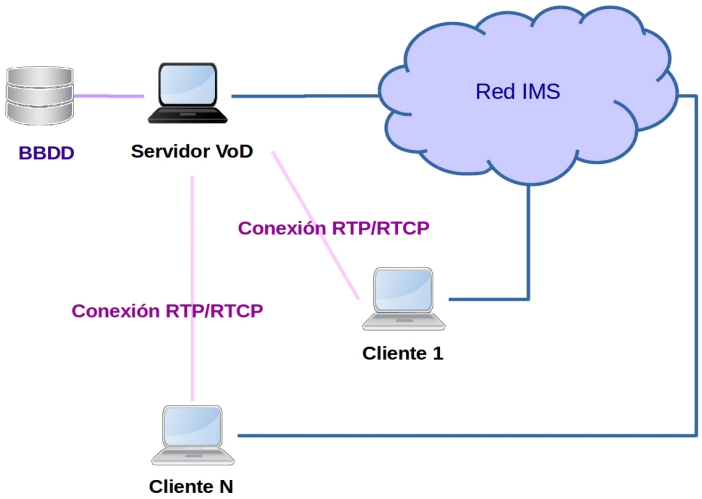


Ilustración 25: Entorno de pruebas conceptual

Sin embargo, por simplicidad las pruebas se han realizado con todos los componentes en un único equipo. Este equipo tenía instalada la plataforma IMS, el cliente, el servidor y la base de datos.

En cuanto a la plataforma IMS tenía configurados tres usuarios: Alice, Bob y Nsc.

La ilustración 26 muestra las identidades públicas de los usuarios en el entorno de pruebas

Public User Identity - Search Results

ID	Identity	Implicit-Set ID	Type	Reg. Status	Barring
1	sip:alice@open-ims.test	1	Public_User_Identity	Registered	false
2	sip:bob@open-ims.test	2	Public_User_Identity	Registered	false
3	sip:nsc@open-ims.test	3	Public_User_Identity	Registered	false

Rows per page
1 20

Ilustración 26: Identidades públicas en el entorno de pruebas

La ilustración 27 muestra las identidades privadas de los usuarios en el entorno de pruebas.

Private User Identity - Search Results

ID	Identity	Auth. Scheme	SQN
4	alice@open-ims.test	4	0000000003fe
2	bob@open-ims.test	20	00000000181f
5	nsc@open-ims.test	127	000000000233

Rows per page
1 20

Ilustración 27: Identidades privadas en el entorno de pruebas

En nuestras pruebas el servidor se registra como Bob y los clientes como Alice o Nsc.

Para realizar las pruebas se ha utilizado un portátil con las siguientes características:

- Sistema operativo Linux Ubuntu 14.04.2 LTS.
- Número de procesadores: 4.

- Versión del kernel de Linux 3.13.0-49-generic.
- Memoria RAM 4GB
- Disco duro 500 GB
- 2 interfaces de red.

Para comprobar el correcto funcionamiento, además de las pruebas visuales, se ha empleado el programa wireshark. Este programa nos ha ayudado a comprobar que los mensajes intercambiados entre cliente y servidor son los esperados.

5.2 Pruebas realizadas

5.2.1 Registro en IMS

Tanto el servidor de VoD como el cliente deben registrarse en la red IMS.

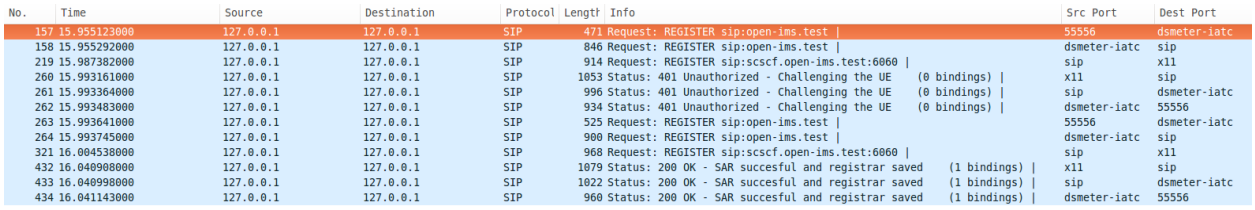
Registro del servidor de VoD en IMS

Lo primero que hace el servidor es registrarse en la red IMS. A continuación, permitirá escuchar clientes o salir de la aplicación. La ilustración 28 muestra este primer paso.

Ilustración 28: El servidor se registra en IMS

Para realizar el registro se sigue el proceso mostrado en la ilustración 10.

La ilustración 29 se muestra las tramas capturadas por wireshark:



No.	Time	Source	Destination	Protocol	Length	Info	Src Port	Dest Port
157	15.955123000	127.0.0.1	127.0.0.1	SIP	471	Request: REGISTER sip:open-ims.test |	55556	dsmeter-iatc
158	15.955292000	127.0.0.1	127.0.0.1	SIP	846	Request: REGISTER sip:open-ims.test |	dsmeter-iatc	sip
219	15.987382000	127.0.0.1	127.0.0.1	SIP	914	Request: REGISTER sip:scscf.open-ims.test:6060 |	sip	x11
260	15.993161000	127.0.0.1	127.0.0.1	SIP	1053	Status: 401 Unauthorized - Challenging the UE (0 bindings) |	x11	sip
261	15.993364000	127.0.0.1	127.0.0.1	SIP	996	Status: 401 Unauthorized - Challenging the UE (0 bindings) |	sip	dsmeter-iatc
262	15.993483000	127.0.0.1	127.0.0.1	SIP	934	Status: 401 Unauthorized - Challenging the UE (0 bindings) |	dsmeter-iatc	55556
263	15.993641000	127.0.0.1	127.0.0.1	SIP	525	Request: REGISTER sip:open-ims.test |	55556	dsmeter-iatc
264	15.993745000	127.0.0.1	127.0.0.1	SIP	900	Request: REGISTER sip:open-ims.test |	dsmeter-iatc	sip
321	16.004538000	127.0.0.1	127.0.0.1	SIP	968	Request: REGISTER sip:scscf.open-ims.test:6060 |	sip	x11
432	16.040908000	127.0.0.1	127.0.0.1	SIP	1079	Status: 200 OK - SAR succesful and registrar saved (1 bindings) |	x11	sip
433	16.040998000	127.0.0.1	127.0.0.1	SIP	1022	Status: 200 OK - SAR succesful and registrar saved (1 bindings) |	sip	dsmeter-iatc
434	16.041143000	127.0.0.1	127.0.0.1	SIP	960	Status: 200 OK - SAR succesful and registrar saved (1 bindings) |	dsmeter-iatc	55556

Ilustración 29: [Pruebas realizadas] El servidor se registra en IMS

A continuación se detalla línea a línea el mensaje de petición REGISTER enviado por el servidor

Método del mensaje acompañado de la uri de petición y la versión de SIP

REGISTER sip:open-ims.test SIP/2.0

Identifica de forma única la petición. Las respuestas a este mensaje presentarán el mismo valor.

Call-ID:

Número de secuencia

CSeq: 1 REGISTER

Receptor de la solicitud. Al tratarse de un mensaje de registro el valor es el mismo que el de la cabecera From.

To: <sip:bob@open-ims.test:55556>

Iniciador de la solicitud. En este caso, el usuario IMS del servidor es Bob

From: <sip: bob@open-ims.test:55556>;tag=2512109996

Cabecera para autenticación. Informa del usuario y del realm. Además indica que emplea el algoritmo MD5 para obtener la respuesta al reto que enviará IMS.

Authorization: Digest username="bob@open-ims.test", realm="open-ims.test", response="", uri="sip:open-ims.test", algorithm=MD5

Camino de la solicitud hasta el momento presente

Via: SIP/2.0/UDP open-ims.test:55556

URI donde el usuario puede ser localizado

Contact: <sip:bob@open-ims.test:55556>

Período de validez del registro

Expires: 3600

Cliente que realiza la comunicación, esto es, nuestra aplicación.

User-Agent: SIP Video Server

Contenido del cuerpo del mensaje. En este caso es 0 por que no presenta información adicional.

Content-Length:0

Registro del cliente en la red IMS

Lo primero que hace el cliente es elegir el contenido que desea visualizar. A continuación elige el formato y por último el puerto o los puertos. Una vez realizados estos pasos, el cliente se registra automáticamente en IMS. La ilustración 30 muestra el proceso.

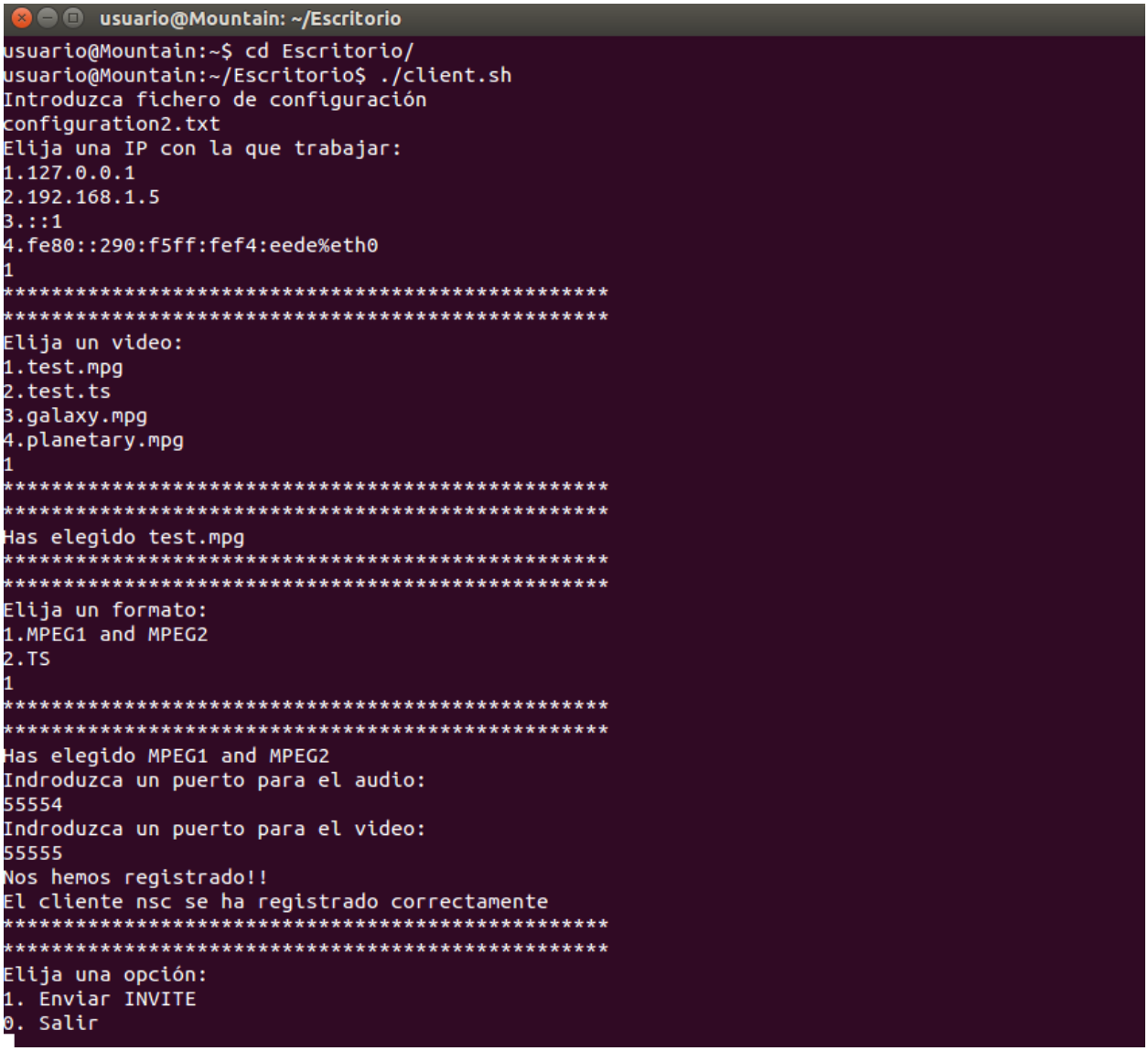
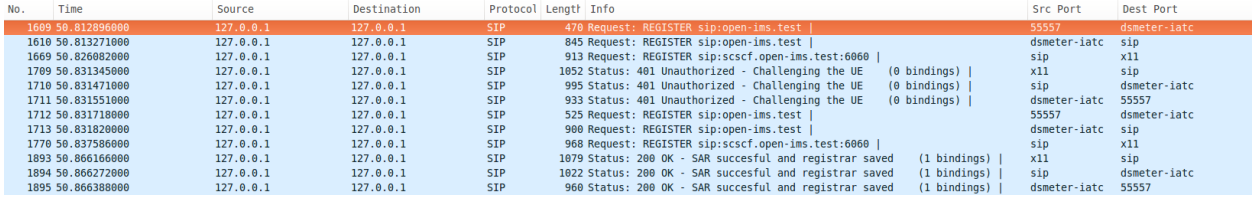


Ilustración 30: El cliente se registra en IMS

La ilustración 31 muestra las tramas intercambiadas entre cliente e IMS.



No.	Time	Source	Destination	Protocol	Length	Info	Src Port	Dest Port
1609	50.812896000	127.0.0.1	127.0.0.1	SIP	470	Request: REGISTER sip:open-ims.test |	55557	dsmeter-iatc
1610	50.813271000	127.0.0.1	127.0.0.1	SIP	845	Request: REGISTER sip:open-ims.test |	dsmeter-iatc	sip
1669	50.826082000	127.0.0.1	127.0.0.1	SIP	913	Request: REGISTER sip:scscf.open-ims.test:6060 |	sip	x11
1709	50.831345000	127.0.0.1	127.0.0.1	SIP	1052	Status: 401 Unauthorized - Challenging the UE (0 bindings) |	x11	sip
1710	50.831471000	127.0.0.1	127.0.0.1	SIP	995	Status: 401 Unauthorized - Challenging the UE (0 bindings) |	sip	dsmeter-iatc
1711	50.831551000	127.0.0.1	127.0.0.1	SIP	933	Status: 401 Unauthorized - Challenging the UE (0 bindings) |	dsmeter-iatc	55557
1712	50.831718000	127.0.0.1	127.0.0.1	SIP	525	Request: REGISTER sip:open-ims.test |	55557	dsmeter-iatc
1713	50.831820000	127.0.0.1	127.0.0.1	SIP	900	Request: REGISTER sip:open-ims.test |	dsmeter-iatc	sip
1770	50.837586000	127.0.0.1	127.0.0.1	SIP	968	Request: REGISTER sip:scscf.open-ims.test:6060 |	sip	x11
1893	50.866166000	127.0.0.1	127.0.0.1	SIP	1079	Status: 200 OK - SAR succesful and registrar saved (1 bindings) |	x11	sip
1894	50.866272000	127.0.0.1	127.0.0.1	SIP	1022	Status: 200 OK - SAR succesful and registrar saved (1 bindings) |	sip	dsmeter-iatc
1895	50.866388000	127.0.0.1	127.0.0.1	SIP	960	Status: 200 OK - SAR succesful and registrar saved (1 bindings) |	dsmeter-iatc	55557

Ilustración 31: [Pruebas realizadas]El cliente se registra en IMS

A continuación detallamos el segundo mensaje de petición REGISTER enviado por el cliente con la respuesta al reto lanzada por IMS.

Método del mensaje acompañado de la uri de petición y la versión de SIP
REGISTER sip:open-ims.test SIP/2.0
Identifica de forma única la petición. Las respuestas a este mensaje presentarán el mismo valor.
Call-ID:
Número de secuencia. En este caso es el segundo mensaje que se envía
CSeq: 2 REGISTER
Receptor de la solicitud. Al tratarse de un mensaje de registro el valor es el mismo que el de la cabecera From.
To: <sip:nsc@open-ims.test:55557>
Iniciador de la solicitud. En este caso, el usuario IMS del servidor esNsc
From: <sip:nsc@open-ims.test:55557>;tag=3647288890
Cabecera para autenticación. Informa del usuario y del realm. En este caso incorpora el campo nonce enviado previamente por IMS y la respuesta al reto en el campo response.
Authorization: Digest username="nsc@open-ims.test", realm="open-ims.test", nonce="db95bf28a4a4c34e149fad84fb7be772", response="1d0f4521531c7a8b5eac771f09030e0f", uri="open-ims.test"
Camino de la solicitud hasta el momento presente
Via: SIP/2.0/UDP open-ims.test:55557
URI donde el usuario puede ser localizado
Contact: <sip:nsc@open-ims.test:55557>
Período de validez del registro
Expires: 3600
Cliente que realiza la comunicación, esto es, nuestra aplicación.
User-Agent: SIP Video Server
Contenido del cuerpo del mensaje. En este caso es 0 por que no presenta información adicional.
Content-Length:0

5.2.2 Inicio de sesión entre cliente y servidor

Cuando el cliente ha elegido el media y el formato que desea recibir, envía un mensaje tipo INVITE al servidor de VoD para iniciar una sesión. El servidor comprobará si tiene el contenido indicado. En caso de ser así, acepta el cliente iniciando la sesión. La ilustración muestra el proceso.

CAPITULO 5: PRUEBAS REALIZADAS

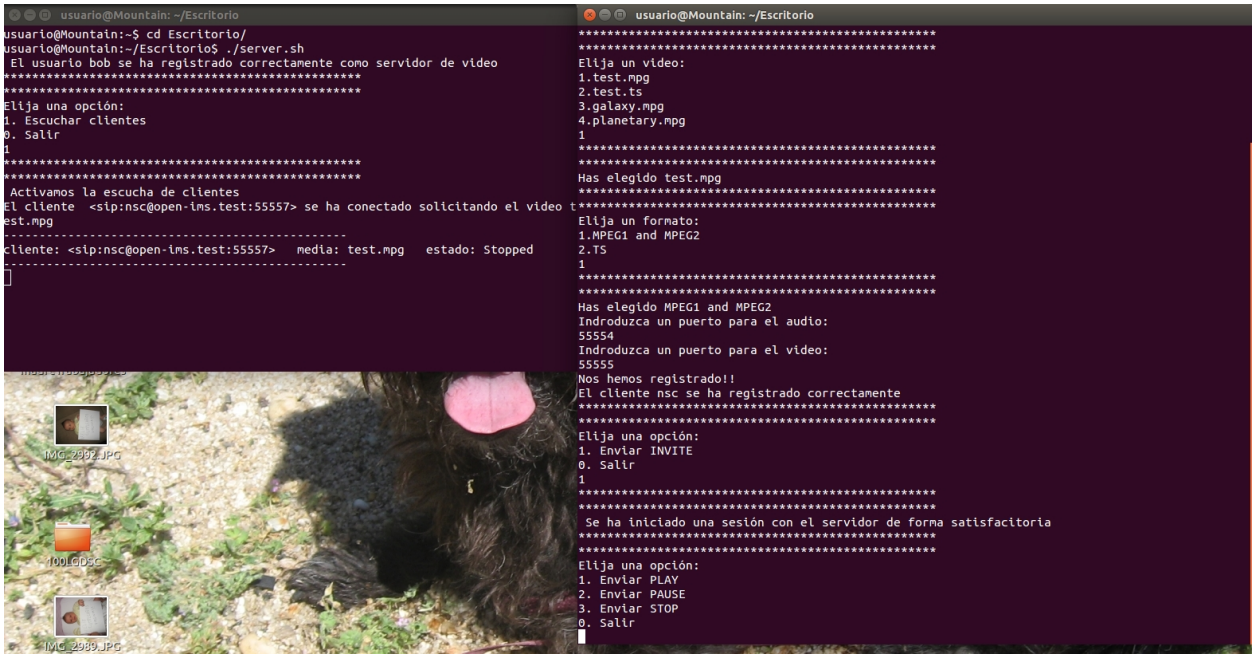
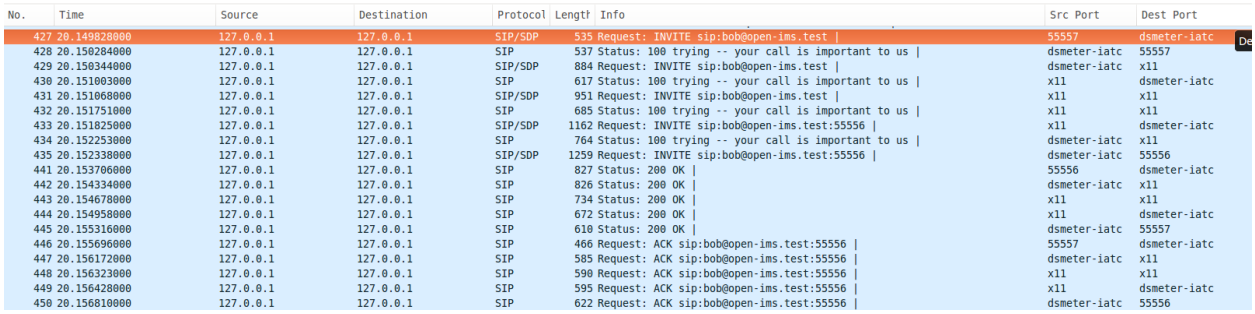


Ilustración 32: Inicio sesión entre cliente y servidor

La ilustración 33 muestra el intercambio de mensajes entre cliente y servidor.



No.	Time	Source	Destination	Protocol	Length	Info	Src Port	Dest Port
427	20.149828000	127.0.0.1	127.0.0.1	SIP/SDP	535	Request: INVITE sip:bob@open-ims.test |	55557	dsmeter-iatc
428	20.150284000	127.0.0.1	127.0.0.1	SIP	537	Status: 100 trying -- your call is important to us |	dsmeter-iatc	55557
429	20.150344000	127.0.0.1	127.0.0.1	SIP/SDP	884	Request: INVITE sip:bob@open-ims.test |	dsmeter-iatc	x11
430	20.151003000	127.0.0.1	127.0.0.1	SIP	617	Status: 100 trying -- your call is important to us |	x11	dsmeter-iatc
431	20.151068000	127.0.0.1	127.0.0.1	SIP/SDP	951	Request: INVITE sip:bob@open-ims.test |	x11	x11
432	20.151751000	127.0.0.1	127.0.0.1	SIP	685	Status: 100 trying -- your call is important to us |	x11	x11
433	20.151825000	127.0.0.1	127.0.0.1	SIP/SDP	1162	Request: INVITE sip:bob@open-ims.test:55556 |	x11	dsmeter-iatc
434	20.152253000	127.0.0.1	127.0.0.1	SIP	764	Status: 100 trying -- your call is important to us |	dsmeter-iatc	x11
435	20.152338000	127.0.0.1	127.0.0.1	SIP/SDP	1259	Request: INVITE sip:bob@open-ims.test:55556 |	dsmeter-iatc	55556
441	20.153706000	127.0.0.1	127.0.0.1	SIP	827	Status: 200 OK |	55556	dsmeter-iatc
442	20.154334000	127.0.0.1	127.0.0.1	SIP	826	Status: 200 OK |	dsmeter-iatc	x11
443	20.154678000	127.0.0.1	127.0.0.1	SIP	734	Status: 200 OK |	x11	x11
444	20.154958000	127.0.0.1	127.0.0.1	SIP	672	Status: 200 OK |	x11	dsmeter-iatc
445	20.155316000	127.0.0.1	127.0.0.1	SIP	610	Status: 200 OK |	dsmeter-iatc	55557
446	20.155696000	127.0.0.1	127.0.0.1	SIP	466	Request: ACK sip:bob@open-ims.test:55556 |	55557	dsmeter-iatc
447	20.156172000	127.0.0.1	127.0.0.1	SIP	585	Request: ACK sip:bob@open-ims.test:55556 |	dsmeter-iatc	x11
448	20.156323000	127.0.0.1	127.0.0.1	SIP	590	Request: ACK sip:bob@open-ims.test:55556 |	x11	x11
449	20.156428000	127.0.0.1	127.0.0.1	SIP	595	Request: ACK sip:bob@open-ims.test:55556 |	x11	dsmeter-iatc
450	20.156810000	127.0.0.1	127.0.0.1	SIP	622	Request: ACK sip:bob@open-ims.test:55556 |	dsmeter-iatc	55556

Ilustración 33: [Pruebas realizadas]Inicio de sesión cliente-servidor

A continuación se detalla el mensaje de petición INVITE que envía el cliente al servidor para iniciar una sesión.

Método del mensaje acompañado de la uri de petición y la versión de SIP

INVITE sip:bob@open-ims.test SIP/2.0

Identifica de forma única la petición. Las respuestas a este mensaje presentarán el mismo valor.

Call-ID:

Número de secuencia. En este caso es el tercer mensaje que se envía

CSeq: 3 INVITE

Receptor de la solicitud. En este caso el usuario IMS del servidor

To: <sip:bob@open-ims.test>

Iniciador de la solicitud. En este caso el cliente

From: <sip:nsc@open-ims.test:55557>;tag=3231895250

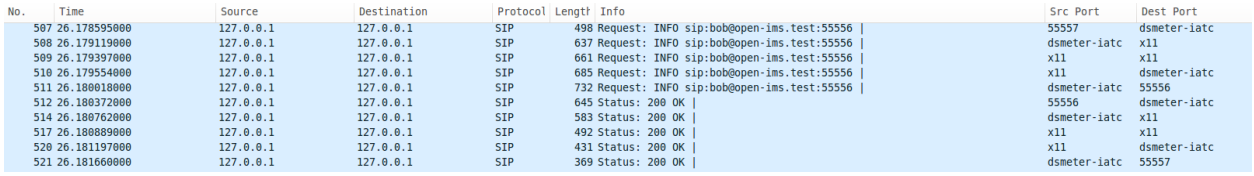
```
Camino de la solicitud hasta el momento presente
Via: SIP/2.0/UDP open-ims.test:55557
Ruta devuelta por IMS durante el registro.
Route: <sip:orig@scscf.open-ims.test:6060;lr>
URI donde el usuario puede ser localizado
Contact:  <sip:nsc@open-ims.test:55557>
Período de validez
Expires: 3600
Cliente que realiza la comunicación, esto es, nuestra aplicación.
User-Agent: SIP Video Server
Tipo de contenido de nuestro mensaje. En este caso es sdp
Content-Type: application/sdp
Longitud del contenido de nuestro mensaje
Content-Length:119

v=0
o=nsc 3619790567 3619790567 IN IP4 192.168.1.5
Nombre del vídeo a reproducir
s=test.ts
t=0 0
Dirección IP donde se debe enviar el contenido
c=IN IP4 192.168.1.5
Tipo de contenido, puerto y formato del contenido
m=video 55557 RTP/AVP 33
```

5.2.3 Envío de comandos del cliente al servidor

Una vez que la sesión está establecida entre cliente y servidor,el cliente enviará comandos para manipular el vídeo. La ilustración 34 muestra la lista de comandos que puede enviar el cliente.

Ilustración 34: Menú con los comandos a enviar por el cliente

La ilustración 35 muestra el intercambio de tramas entre cliente y servidor durante el envío de un comando.



No.	Time	Source	Destination	Protocol	Length	Info	Src Port	Dest Port
507	26.178595000	127.0.0.1	127.0.0.1	SIP	498	Request: INFO sip:bob@open-ims.test:55556 |	55557	dsmeter-iatc
508	26.179119000	127.0.0.1	127.0.0.1	SIP	637	Request: INFO sip:bob@open-ims.test:55556 |	dsmeter-iatc	x11
509	26.179397000	127.0.0.1	127.0.0.1	SIP	661	Request: INFO sip:bob@open-ims.test:55556 |	x11	x11
510	26.179554000	127.0.0.1	127.0.0.1	SIP	685	Request: INFO sip:bob@open-ims.test:55556 |	x11	dsmeter-iatc
511	26.180018000	127.0.0.1	127.0.0.1	SIP	732	Request: INFO sip:bob@open-ims.test:55556 |	dsmeter-iatc	55556
512	26.180372000	127.0.0.1	127.0.0.1	SIP	645	Status: 200 OK |	55556	dsmeter-iatc
514	26.180762000	127.0.0.1	127.0.0.1	SIP	583	Status: 200 OK |	dsmeter-iatc	x11
517	26.180889000	127.0.0.1	127.0.0.1	SIP	492	Status: 200 OK |	x11	x11
520	26.181197000	127.0.0.1	127.0.0.1	SIP	431	Status: 200 OK |	x11	dsmeter-iatc
521	26.181660000	127.0.0.1	127.0.0.1	SIP	369	Status: 200 OK |	dsmeter-iatc	55557

Ilustración 35: [Pruebas realizadas]Envío de comandos cliente-servidor

Envío de comando PLAY

A continuación se muestra el mensaje de petición tipo INFO que envía el cliente al servidor para iniciar la reproducción de vídeo.

Método del mensaje acompañado de la uri de petición y la versión de SIP
INFO sip:bob@open-ims.test:55556 SIP/2.0
Identifica de forma única la petición. Las respuestas a este mensaje presentarán el mismo valor.
Call-ID:
Número de secuencia.
CSeq: 4 INFO
Receptor de la solicitud. En este caso el usuario IMS del servidor
To: <sip:bob@open-ims.test>;tag=1220444376
Iniciador de la solicitud. En este caso el cliente
From: <sip:nsc@open-ims.test:55557>;tag=3231895250
Camino de la solicitud hasta el momento presente
Via: SIP/2.0/UDP open-ims.test:55557
Ruta que debe seguir el mensaje hasta el destino.
Route: <sip:mo@pcscf.open-ims.test:4060;lr>,<sip:mo@scscf.open-ims.test:6060;lr>,<sip:mt@scscf.open-ims.test:6060;lr>,<sip:mt@pcscf.open-ims.test:4060;lr>
Período de validez
Expires: 3600
Cliente que realiza la comunicación, esto es, nuestra aplicación.
User-Agent: SIP Video Server
Tipo de contenido de nuestro mensaje. En este caso es texto.
Content-Type: plain/text
Longitud del contenido de nuestro mensaje
Content-Length:4

PLAY

Los mensajes enviados para pausar y parar la reproducción de vídeo son iguales a excepción del cuerpo del mensaje, que contiene el comando adecuado.

5.2.4 El servidor envía contenido multimedia al cliente

Con la sesión establecida, cuando el cliente envía el comando PLAY al servidor, éste último empieza a transmitir el contenido multimedia. Esta transmisión de contenido se realiza a través de una conexión RTP previamente acordada a través del contenido SDP del mensaje INVITE.

Si recuperamos el contenido SDP del mensaje INVITE :

```
v=0
o=nsc 3619790567 3619790567 IN IP4 192.168.1.5
s=test.ts
t=0 0
c=IN IP4 192.168.1.5
m=video 55557 RTP/AVP 33
```

Podemos ver que el cliente ha solicitado el vídeo *test.ts* con formato MP2T a través del puerto 55557.

La respuesta del servidor ha sido positiva, por lo que en el momento en el que el cliente envíe el comando PLAY, el servidor debe enviar el contenido multimedia.

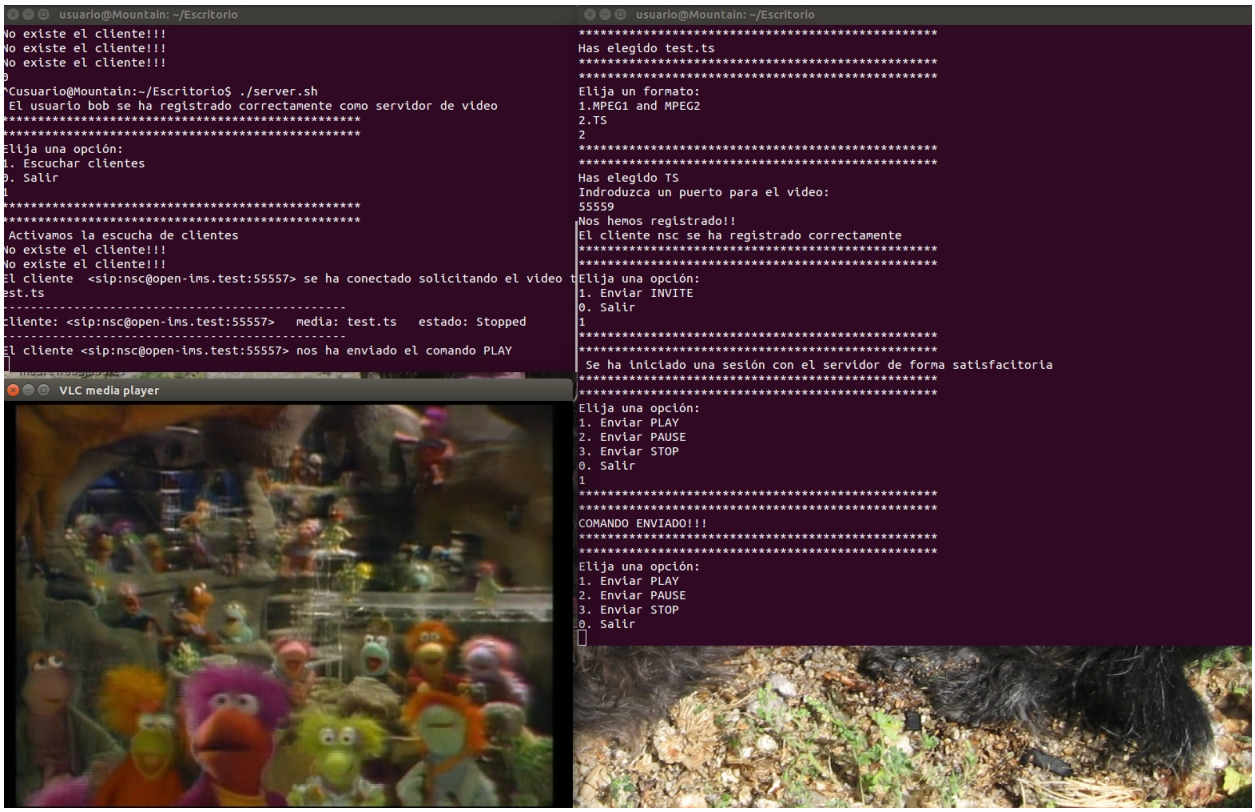


Ilustración 36: Cliente reproduciendo el contenido enviado por el servidor

La ilustración 37 muestra el envío del contenido multimedia con formato MP2T del servidor al cliente a través del puerto 55557.

No.	Time	Source	Destination	Protocol	Length	Info	Src Port	Dest Port
573	27.056612000	192.168.1.5	192.168.1.5	MPEG TS	1372	PT=MPEG-II transport streams, SSRC=0xE85561A7, Seq=48361, Time=283932330	55557	55557
574	27.068630000	192.168.1.5	192.168.1.5	MPEG TS	1372	PT=MPEG-II transport streams, SSRC=0xE85561A7, Seq=48362, Time=283932438	55557	55557
575	27.080635000	192.168.1.5	192.168.1.5	MPEG TS	1372	PT=MPEG-II transport streams, SSRC=0xE85561A7, Seq=48363, Time=283932546	55557	55557
576	27.092667000	192.168.1.5	192.168.1.5	MPEG TS	1372	PT=MPEG-II transport streams, SSRC=0xE85561A7, Seq=48364, Time=283932655	55557	55557

Ilustración 37: [Pruebas realizadas] Envío contenido multimedia

5.2.5 Finalización de sesión entre cliente y servidor

Cuando el cliente no quiera visualizar mas el vídeo, podrá cerrar la sesión con el servidor. Esto lo consigue enviando un mensaje tipo BYE

La ilustración 38 muestra el intercambio de tramas que se realiza para finalizar la sesión.

6981	57.553228000	127.0.0.1	127.0.0.1	SIP	451	Request: BYE sip:bob@open-ims.test:55556 |	55557	dsmeter-iatc
6982	57.553465000	127.0.0.1	127.0.0.1	SIP	590	Request: BYE sip:bob@open-ims.test:55556 |	dsmeter-iatc	x11
6983	57.553572000	127.0.0.1	127.0.0.1	SIP	615	Request: BYE sip:bob@open-ims.test:55556 |	x11	x11
6984	57.553646000	127.0.0.1	127.0.0.1	SIP	640	Request: BYE sip:bob@open-ims.test:55556 |	x11	dsmeter-iatc
6985	57.553817000	127.0.0.1	127.0.0.1	SIP	687	Request: BYE sip:bob@open-ims.test:55556 |	dsmeter-iatc	55556
6986	57.553960000	127.0.0.1	127.0.0.1	SIP	646	Status: 200 OK |	55556	dsmeter-iatc
6987	57.554093000	127.0.0.1	127.0.0.1	SIP	584	Status: 200 OK |	dsmeter-iatc	x11
6988	57.554129000	127.0.0.1	127.0.0.1	SIP	492	Status: 200 OK |	x11	x11
6989	57.554211000	127.0.0.1	127.0.0.1	SIP	430	Status: 200 OK |	x11	dsmeter-iatc
6990	57.554305000	127.0.0.1	127.0.0.1	SIP	368	Status: 200 OK |	dsmeter-iatc	55557

Ilustración 38: [Pruebas realizadas] Finalización de sesión entre cliente y servidor

A continuación se describen los mensajes intercambiados.

El cliente manda un mensaje tipo BYE al servidor.

Método del mensaje acompañado de la uri de petición y la versión de SIP
BYE sip:bob@open-ims.test:55556 SIP/2.0
Identifica de forma única la petición. Las respuestas a este mensaje presentarán el mismo valor.
Call-ID:
Número de secuencia.
CSeq: 8 BYE
Receptor de la solicitud. En este caso el usuario IMS del servidor
To: <sip:bob@open-ims.test>
Iniciador de la solicitud. En este caso el cliente
From: <sip:nsc@open-ims.test:55557>;tag=1850389210
Camino de la solicitud hasta el momento presente
Via: SIP/2.0/UDP open-ims.test:55557
Ruta que debe seguir el mensaje hasta el destino.
Route: <sip:mo@pcscf.open-ims.test:4060;lr>,<sip:mo@scscf.open-ims.test:6060;lr>,<sip:mt@scscf.open-ims.test:6060;lr>,<sip:mt@pcscf.open-ims.test:4060;lr>
Período de validez
Expires: 3600
Cliente que realiza la comunicación, esto es, nuestra aplicación.
User-Agent: SIP Video Server
Longitud del contenido de nuestro mensaje
Content-Length:0

5.3 Conclusiones

En este capítulo se ha mostrado el resultado de las pruebas realizadas. Estudiando los diferentes apartados podemos llegar a la siguiente conclusión: tanto el servidor de VoD como el cliente realizan de forma correcta las funciones de un usuario IMS (registro , establecimiento y finalización de sesión) y las funciones relacionadas con el tratamiento de contenido multimedia.

Este capítulo es la base del siguiente, que nos ofrece las conclusiones generales del proyecto.

Capítulo 6

Conclusiones y líneas futuras

6.1 Conclusiones

La conclusión principal que se obtienen de la evaluación de la implementación es que se ha conseguido el objetivo del proyecto, esto es, desarrollar un servidor de VoD para una red IMS.

Una segunda conclusión es que se puede desarrollar un servidor de VoD para una red IMS empleando SIP + RTP. En un principio se pensó en desarrollar una solución SIP + RTSP, basándonos en la especificación [13]. Sin embargo se optó por una solución basada únicamente en SIP + RTP. Esta opción es mas sencilla y ajustada a nuestras necesidades. La sencillez radica en que así manejamos un único protocolo de señalización.

El proyecto incluye el uso de la librería live555 para el intercambio de mensajes y la transmisión de vídeo. Esta librería ha sido de gran ayuda en relación a la transmisión de vídeo. Sin embargo, en lo relativo a su manipulación resulta mas compleja de lo esperado. Permite iniciar, pausar y parar el vídeo de forma sencilla. Sin embargo, el avance y retroceso rápidos, además de no poderse realizar en todos los formatos, implica la creación de terceros ficheros, lo que incrementa la complejidad. En cuanto al intercambio de mensajes, nos ha facilitado la comunicación con la red IMS, sin embargo no facilita la generación de mensajes SIP dado que no existe ningún módulo destinado a la creación de mensajes SIP. Para la implementación de la solución se ha tenido que crear un generador de mensajes y un manejador de diálogos, que controla las cabeceras de los diferentes mensajes SIP. Otro inconveniente de la librería es la falta de documentación y ejemplos de uso.

Cabe destacar la implementación de IMS empleada en el proyecto. Open IMS core es una plataforma relativamente sencilla de instalar y manejar. Representa un excelente entorno de pruebas para servicios basados en IMS. Además existe una gran cantidad de documentación y foros de ayuda. Por otra parte, ofrece una máquina virtual con IMS instalado y listo para probar para aquellos que no quieran realizar el proceso de instalación.

6.2 Líneas futuras

A continuación vamos a comentar posibles trabajos futuros relacionados con este proyecto.

- Ampliación de los formatos ofrecidos. El proyecto actual ofrece únicamente dos formatos. Un posible trabajo futuro consistiría en ampliar el número de formatos ofrecidos.
- Ampliación de los comandos de manipulación del vídeo. El servidor admite los comandos PLAY, PAUSE y STOP. Sería interesante incluir más comandos, como por ejemplo avance y retroceso rápido.
- Generación de una interfaz gráfica. El cliente obtiene el contenido ofrecido por el servidor de un fichero. Sería interesante que el servidor presentara una interfaz web ofreciendo contenidos y formatos. Por otra parte, también se podrían introducir elementos de gestión del servidor en la interfaz, como visualización de clientes conectados, registro en la red IMS o de-registro.
- Generación automática de guía electrónica de contenidos cuando se añaden al servidor, que pudiera ser consultada por clientes (por ejemplo, vía HTTP) para saber los contenidos disponibles en el servidor.

Apéndice A

Planificación del proyecto

Vamos a utilizar el diagrama de Gantt para mostrar el tiempo de dedicación para las diferentes tareas en las que se ha dividido el proyecto. La ilustración 39 muestra las diferentes tareas de las que se ha compuesto el proyecto:

APÉNDICE A: PLANIFICACIÓN DEL PROYECTO

Nombre	Fecha de inicio	Fecha de fin
Inicio del proyecto	12/01/15	12/01/15
Fase de análisis	13/01/15	29/01/15
Propuesta del proyecto	13/01/15	13/01/15
Estudio del estado del arte	13/01/15	26/01/15
Instalación de software	27/01/15	29/01/15
Fase de diseño	30/01/15	26/02/15
Diseño del servidor	30/01/15	12/02/15
Diseño de la BBDD	5/02/15	6/02/15
Diseño del cliente	13/02/15	26/02/15
Fase de implementación	27/02/15	25/05/15
Implementación de la BBDD	27/02/15	2/03/15
Implementación del servidor	3/03/15	13/04/15
Implementación del cliente	14/04/15	25/05/15
Fase de pruebas	26/05/15	8/06/15
Documentación	12/01/15	19/06/15
Reuniones	13/01/15	18/06/15
Reunión 1	13/01/15	13/01/15
Reunión 2	30/01/15	30/01/15
Reunión 3	26/02/15	26/02/15
Reunión 4	26/05/15	26/05/15
Reunión 5	8/06/15	8/06/15
Reunión 6	19/06/15	19/06/15
Fin del proyecto	22/06/15	22/06/15

Ilustración 39: Listado de tareas

Se debe tener en cuenta que cada día de trabajo corresponde a 4 horas. Por otra parte, el calendario se corresponde con un calendario laboral, donde los sábados y domingos son días de descanso.

La ilustración 40 muestra la planificación del proyecto completo.

APÉNDICE A: PLANIFICACIÓN DEL PROYECTO

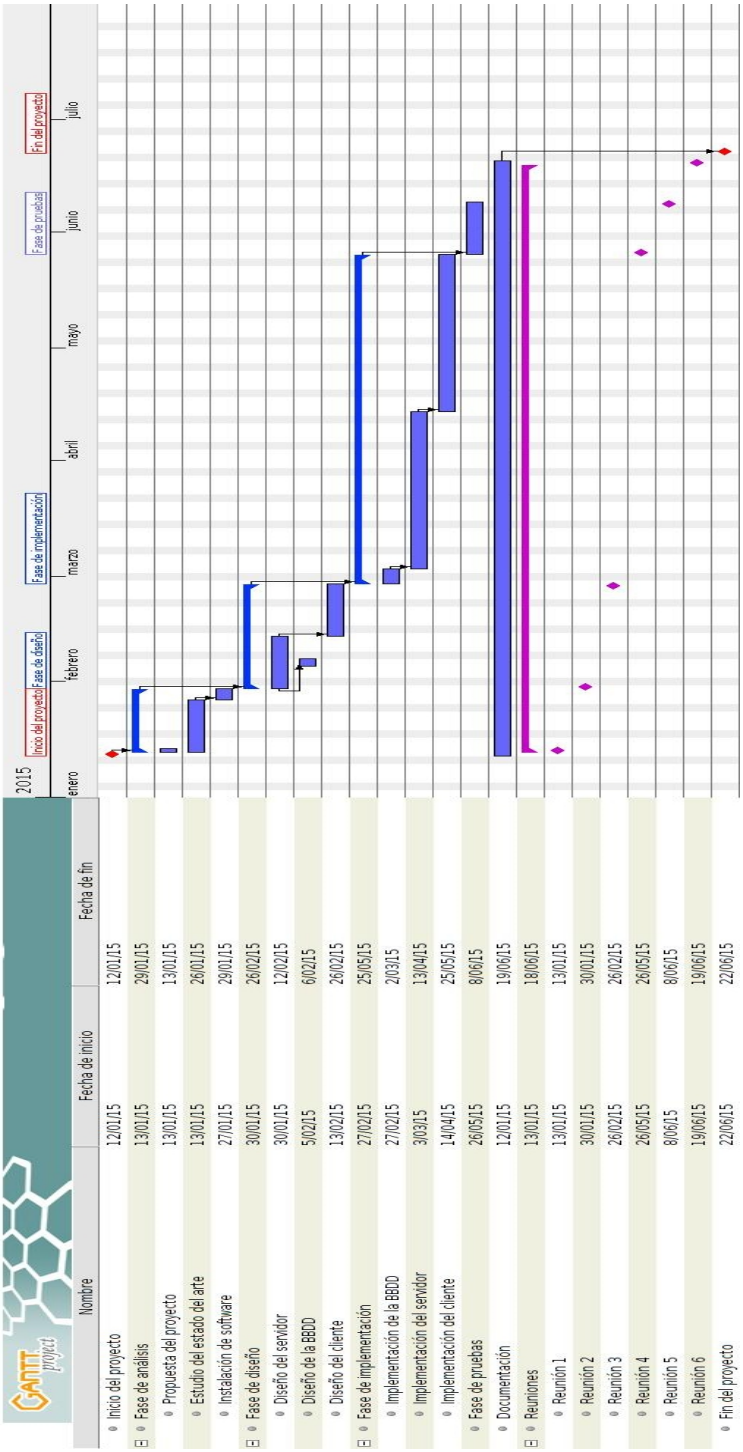


Ilustración 40: Diagrama de Gantt

Según el diagrama de Gantt mostrado en la ilustración anterior, el proyecto ha tendido una duración en días de 115 y en horas de 460.

Apéndice B

Presupuesto del proyecto

Introducción

En este capítulo se va a describir de forma detallada el presupuesto del proyecto. Para llevarlo a cabo se ha empleado la plantilla proporcionada por la universidad.

Horas dedicadas

Basándonos en el diagrama de Gantt expuesto en el apartado anterior el número de horas totales dedicadas al proyecto son las que siguen:

Fase de análisis: 13 días x 3 horas = 39 horas

Fase de diseño: 20 días x 3 horas = 60 horas

Fase de implementación: 72 días x 3 horas = 216

Fase de pruebas: 10 días x 3 horas = 30 horas

Documentación: 115 días x 1 horas = 115 horas

Reuniones: 6 reuniones x 1 horas = 6 horas.

El total de horas dedicadas será la suma de las horas de cada fase, esto es, 460 horas

Personal

En la Tabla 13 se muestra el personal destinado a proyecto. Los salarios por hora son los dictados en la plantilla proporcionada por la universidad

Cargo	Nº horas	Dedicación (Hombre/mes)	Coste hombre mes	Coste total
Ingeniero Senior	36	0.45	4.289,54 euros	1930,293
Ingeniero	400	5	2.694.39 euros	13471,95
			Total	15.402,24

Tabla 13: Presupuesto personal

La dedicación se ha calculado contando que una persona trabaja al mes 80 horas.

Hardware

En la Tabla 14 se muestra el hardware empleado en el proyecto:

APÉNDICE B: PRESUPUESTO DEL PROYECTO

Descripción	Unidades	Coste (€)	Dedicación (meses)	Coste imputable
Ordenador Portátil Mountain	1	1559	6	155.90
Ordenador sobremesa Intel	1	900	6	90
			TOTAL	245,9

Tabla 14: Presupuesto de Hardware

Software

La Tabla 15 que se muestra a continuación contiene las herramientas software necesarias para el proyecto.

Descripción	Unidades	Coste (€)	Coste Total (€)
LibreOffice Writer	1	0	0
LibreOffice Calc	1	0	0
Eclipse Kepler	1	0	0
Open IMS core	1	0	0
MySQL	1	0	0
		TOTAL	0

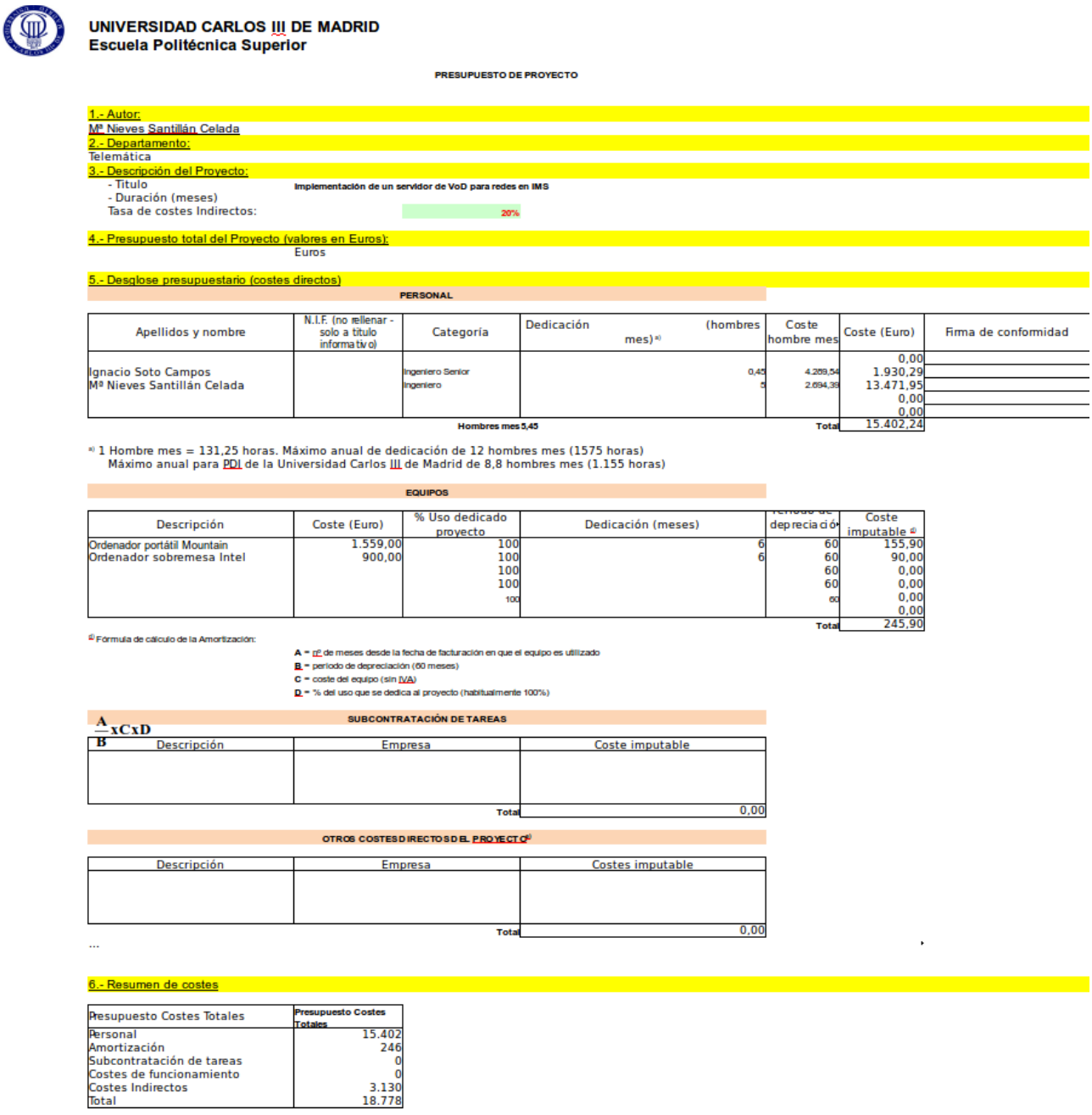
Tabla 15: Presupuesto software

Al tratarse de software libre el coste total de las herramientas ha sido 0 €.

APÉNDICE B: PRESUPUESTO DEL PROYECTO

Plantilla presupuesto

La Ilustración 41 presenta la plantilla de presupuesto del proyecto.



UNIVERSIDAD CARLOS III DE MADRID
Escuela Politécnica Superior

PRESUPUESTO DE PROYECTO

1.- Autor:
Mª Nieves Santillán Celada
2.- Departamento:
Telemática
3.- Descripción del Proyecto:
- Título: Implementación de un servidor de VoD para redes en IMS
- Duración (meses):
Tasa de costes Indirectos: 20%

4.- Presupuesto total del Proyecto (valores en Euros):
Euros

5.- Desglose presupuestario (costes directos)

PERSONAL

Apellidos y nombre	N.I.F. (no rellenar - solo a título informativo)	Categoría	Dedicación (hombres mes) [a]	Coste hombre mes	Coste (Euro)	Firma de conformidad
					0,00	
Ignacio Soto Campos		Ingeniero Senior	0,45	4.269,54	1.930,29	
Mª Nieves Santillán Celada		Ingeniero	5	2.694,39	13.471,95	
					0,00	
					0,00	
		Hombres mes 5,45		Total	15.402,24	

[a] 1 Hombre mes = 131,25 horas. Máximo anual de dedicación de 12 hombres mes (1575 horas)
Máximo anual para PDI de la Universidad Carlos III de Madrid de 8,8 hombres mes (1.155 horas)

EQUIPOS

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Período de depreciación	Coste imputable [d]
Ordenador portátil Mountain	1.559,00	100	6	60	155,90
Ordenador sobremesa Intel	900,00	100	6	60	90,00
		100		60	0,00
		100		60	0,00
		100		60	0,00
				Total	245,90

[d] Fórmula de cálculo de la Amortización:

$\frac{A}{B} xCxD$

A = nº de meses desde la fecha de facturación en que el equipo es utilizado
B = período de depreciación (60 meses)
C = coste del equipo (sin IVA)
D = % del uso que se dedica al proyecto (habitualmente 100%)

SUBCONTRATACIÓN DE TAREAS

Descripción	Empresa	Coste imputable
	Total	0,00

OTROS COSTES DIRECTOS DEL PROYECTO[e]

Descripción	Empresa	Costes imputable
	Total	0,00

...

6.- Resumen de costes

Presupuesto Costes Totales	Presupuesto Costes Totales
Personal	15.402
Amortización	246
Subcontratación de tareas	0
Costes de funcionamiento	0
Costes Indirectos	3.130
Total	18.778

Ilustración 41: Plantilla presupuesto

Apéndice C

Manual de instalación

Este apéndice explica cómo realizar la instalación del servidor de VoD, del cliente, así como de las herramientas necesarias para su correcto funcionamiento.Las instrucciones de instalación asumen un sistema operativo Ubuntu 14.04.2 LTS, y deberían ser fácilmente adaptables para otros sistemas Linux.

MySQL

MySQL se instala con la siguiente instrucción:

```
sudo apt-get install mysql-server-5.5
```

BIND9

Para instalar Bind se ejecuta el siguiente comando

```
sudo apt-get install bind9
```

Más adelante, en el apartado destinado a Open IMS core se detallará cómo configurar el servidor Bind.

VLC

VLC se instala con la siguiente instrucción:

```
sudo apt-get install vlc browser-plugin-vlc
```

Open IMS core

Vamos a dividir la instalación de Open IMS core en 3 fases. Una primera fase de instalación del software previo que es necesario, una segunda fase de instalación del código y una última de configuración

Fase 1 : Instalación del software previo requerido

Open IMS core presenta los siguientes requerimientos software

GCC 3 o 4

Se emplea para compilar los CSCFs.

```
sudo apt-get install libcurl4-gnutls-dev
```

Make

Se emplea para compilar los CSCFs.

```
sudo apt-get install build-essential
```

Ant

Se emplea para compilar el HSS.

```
sudo apt-get install  ant
```

JDK 1.5 o superior

Se emplea para compilar el HSS.

```
sudo add-apt-repository ppa:webupd8team/java
sudo apt-get update
sudo apt-get install oracle-java7-installer
```

MySQL

Se emplea para almacenar información como por ejemplo las cuentas de usuario. Su instalación se ha explicado en un apartado anterior.

bison, flex

Se usan para escanear mensajes SIP y para la configuración.

```
sudo apt-get install bison
sudo apt-get install  flex
```

libxml2

Se emplea para procesar los ficheros de configuración.

```
sudo apt-get install  libxml2-dev
```

libmysql

Se usa para integrar los CSCFs con MySQL.

```
sudo apt-get install  libmysqlclient15-dev
```

bind 9

Su instalación se ha explicado en un apartado anterior.

Subversion

Se utiliza para descargar el código. Para obtenerlo:

```
sudo apt-get install subversion
```

Fase 2 : Instalación del código.

APÉNDICE C: MANUAL DE INSTALACIÓN

A continuación vamos a enumerar los pasos para la instalación de Open IMS core.

Paso 1: Crear estructura de directorios.

Paso 1.1 Crear directorio principal.

sudo mkdir /opt/OpenIMSCore

cd /opt/openIMSCore

Paso 1.2 Crear un directorio para los CSCFs y otro para el HSS.

sudo mkdir ser_ims

sudo mkdir FhoSS

Paso 2: Descargar el código

Se deben descargar dos partes de código, una primera serán los CSCFs y la otra el HSS. Ambos se encuentran en los repositorios svn.

sudo svn checkout https://svn.code.sf.net/p/openimscore/code/ser_ims/trunk/ ser_ims

sudo svn checkout https://svn.code.sf.net/p/openimscore/code/FHoSS/trunk/ FhoSS

Paso 3: Instalar el IMS Core

```
cd ser_ims
sudo make install-libs all
cd ..
```

Paso 4: Compilar y desplegar el HSS

```
cd FHoSS
sudo ant compile deploy
sudo sed -i 's/JAVA_HOME\/bin\/java/JAVA_HOME\/usr\/bin\/java/g' deploy/startup.sh
cd ..
```

Fase 3: configuración de Open IMS core

Una vez completa la fase de instalación, se necesita configurar la instalación.

Cabe recordar que la configuración inicial está basada en el dominio *open_ims.test*. Si se desea modificar el dominio, se deberán modificar los ficheros de configuración.

A continuación, vamos a describir los pasos a seguir para realizar la configuración inicial de open IMS core.

APÉNDICE C: MANUAL DE INSTALACIÓN

Paso 1. Configuración del DNS:

Se debe copiar el fichero de configuración del DNS. Para ello se ejecutará el siguiente comando:

sudo cp ser_ims/cfg/open-ims.dnszone /etc/bind/

sudo sed -i '3azone "open-ims.test" {\n\ttype master;\n\tfile "\/etc\/bind\/open-ims.dnszone";\n};' /etc/bind/named.conf.local

sudo sed -i '2a127.0.0.1\topen-ims.test mobicents.open-ims.test ue.open-ims.test presence.open-ims.test icscf.open-ims.test scscf.open-ims.test pcscf.open-ims.test hss.open-ims.test' /etc/hosts

Finalmente, reiniciaremos el servidor DNS.

sudo /etc/init.d/bind9 restart

Paso 2: Configurar la base de datos

Los scripts de configuración de la base de datos ya se encuentran descargados, sólo habrá que ejecutarlos:

mysql -u root -p -h localhost < /opt/OpenIMSCore/ser_ims/cfg/icscf.sql

mysql -u root -p -h localhost < /opt/OpenIMSCore/FHoSS/scripts/hss_db.sql

mysql -u root -p -h localhost < /opt/OpenIMSCore/FHoSS/scripts/userdata.sql

Paso 3: Configuración de Open-IMS core.

Para configurar Open IMS core disponemos de una serie de ficheros que nos descargamos previamente. Habrá que copiarlos en el directorio principal:

sudo cp ser_ims/cfg/.cfg ./*
sudo cp ser_ims/cfg/.xml ./*
sudo cp ser_ims/cfg/.sh ./*

Ejecución Open IMS core

Una vez instalado y configurado Open IMS core, sólo nos queda ejecutarlo. Para ello habrá que iniciar los diferentes servidores:

sudo ./pcscf.sh
sudo ./scscf.sh
sudo ./icscf.sh
cd FHoSS/deploy
sudo sh startup.sh

Para acceder al FhoSS:

http://hssAdmin:hss@localhost:8080/hss.web.console/

APÉNDICE C: MANUAL DE INSTALACIÓN

Servidor de VoD

El servidor de VoD precisa de la librería live555, de la librería MySQL para c++. A continuación describimos su instalación.

Live555

Para instalar la librería live555 debemos seguir los siguientes pasos:

Paso 1: Descargar la librería de www.live555.com .

Paso 2: Descomprimirla:

```
tar -x live555-latest.tar.gz
```

Paso 3: Ejecutar el siguiente comando para generar los ficheros make:

```
./genMakefiles linux
```

Paso 4:Ejecutar make e instalar las librerías

```
make install
```

Librería MySQL para C++

Para instalar esta librería se debe descargar la librería de http://dev.mysql.com/doc/connector-cpp/en/. A continuación ejecutar el comando *cmake* para generar el fichero make. Una vez tengamos el fichero make, lanzar un *make install* para que la librería se quede instalada.

Cliente

El cliente precisa de la librería live555 descrita en el apartado anterior para poder ser ejecutado. Además precisa de la librería libVLC para poder visualizar el vídeo. Para instalar esta librería habrá que descargarla de www.videolan.org y realizar el "*make install"* correspondiente.

Apéndice D

Librería live555

La librería live555 está estructurada en tres librerías: UsageEnvironment, Groupsock y Media. Además, presenta un conjunto de programas de prueba que permite comprobar el alcance de la librería.

A continuación se describen cada una de las librerías.

UsageEnvironment & TaskScheduler

Estas dos clases son la base de la librería. Se trata de clases abstractas para manejar el entorno.

Proveen las siguientes funciones:

- programación de tareas
- asignación gestores para eventos de lectura asíncrona
- Mostrar errores y avisos por la salida estándar.
- Define una estructura tipo "Hashtable" (tabla hash), utilizada en el resto del código.

UsageEnvironment

Al tratarse de una clase abstracta no presenta constructor. Los métodos de *UsageEnvironment* empleados en el proyecto se muestran en la Tabla 16:

Método	Uso
virtual UsageEnvironment & **operator<< (char const *str)**=0	Mostrar avisos y errores por la salida estándar
TaskScheduler & **taskScheduler ()** const	Obtener *TaskScheduler*.

Tabla 16: Métodos principales UsageEnvironment

El diagrama de herencia de *UsageEnvironment* se muestra en la ilustración 42

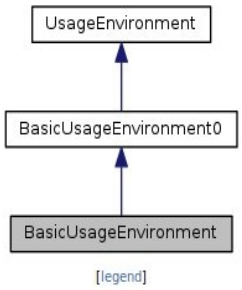


Ilustración 42: Diagrama de herencia UsageEnvironment (fuente: www.live555.com)

El diagrama de colaboración de *UsageEnvironment* se muestra en la ilustración 43.

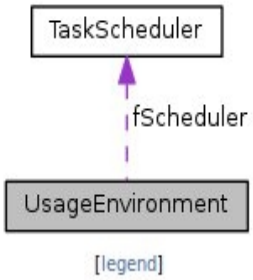


Ilustración 43: Diagrama de colaboración UsageEnvironment (fuente: www.live555.com)

TaskScheduler

Al tratarse de una clase abstracta no presenta constructor. Los métodos de *TaskScheduler* empleados en el proyecto se muestran en la Tabla 17:

Método	Uso
Void **turnOnBackgroundReadHandling** (int socketNum, BackgroundHandlerProc *handlerProc, void *clientData)	Activa la lectura de un determinado socket y le asigna un manejador
void **turnOffBackgroundReadHandling** (int socketNum)	Desactiva la lectura de un determinado socket.
virtual void **setBackgroundHandling** (int socketNum, int conditionSet, BackgroundHandlerProc *handlerProc, void *clientData)=0	Establece la lectura de un eventos de un determinado socket
virtual void **doEventLoop** (char *watchVariable=NULL)=0	Bucle que hace que se quede a la espera de que se produzcan los eventos activados.
virtual EventTriggerId **createEventTrigger** (TaskFunc *eventHandlerProc)=0	Crea un lanzador de eventos.
virtual void **triggerEvent** (EventTriggerId eventTriggerId, void *clientData=NULL)=0	Lanza un evento.
virtual void **deleteEventTrigger** (EventTriggerId eventTriggerId)=0	Elimina un lanzador de eventos

Tabla 17: Métodos TaskScheduler

El diagrama de herencia de la clase *TaskScheduler* se muestra en la Ilustración 44

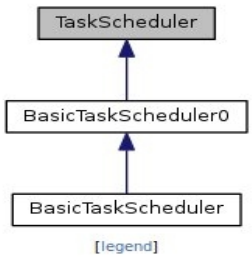


Ilustración 44: Diagrama herencia TaskScheduler (fuente: www.live555.com)

Uso de UsageEnvironment y TaskScheduler

A partir de la clase UsageEnvironment, surge la clase BasicUsageEnvironment. De la misma forma, BasicTaskScheduler hereda de TaskScheduler. Ambas se tratan de implementaciones específicas para desarrollar aplicaciones de consola.

Para empezar a emplearlas se procede como sigue:

```
TaskScheduler* scheduler = BasicTaskScheduler::createNew();
UsageEnvironment* env = BasicUsageEnvironment::createNew(*scheduler);
```

Una vez que tenemos el puntero a un objeto o el objeto *UsageEnvironment* se pueden realizar las diferentes funciones enumeradas anteriormente.

A continuación se muestran algunos ejemplos:

Para mostrar avisos o errores por la salida estándar:

```
*env << "Esto es un aviso";
```

Para obtener el *TaskScheduler*:

```
TaskScheduler& sched = env.taskScheduler();
```

Para activar la lectura de un determinado socket:

```
sched.turnOnBackgroundReadHandling(mSrcSocket->socketNum(),&messageHandler,
this);
```

Para activar bucle de lectura:

```
env.taskScheduler().doEventLoop(&mEventLoopStopFlag);
```

Groupsock

Encapsula interfaces de red y sockets. Permite comunicaciones multicast y unicast para el envío y recepción de datos.

Las clases principales de las librerías son *Groupsock* y *Port*.

Port

Clase que representa un puerto. Proporciona un constructor:

```
Port::Port       ( portNumBits  num)
```

Se emplea para construir los sockets de la aplicación, como se muestra a continuación

Groupsock

Representa un socket. Proporciona dos constructores:

```
      Groupsock (UsageEnvironment &env, struct in_addr const &groupAddr, Port port, u_int8_t ttl)


      Groupsock (UsageEnvironment &env, struct in_addr const &groupAddr, struct in_addr const &sourceFilterAddr, Port port)
```

Los principales métodos se listan en la Tabla 18:

Método	Uso
virtual Boolean **output** (UsageEnvironment &env, u_int8_t ttl, unsigned char *buffer, unsigned bufferSize, DirectedNetInterface *interfaceNotToFwdBackTo=NULL)	Escribe en el socket
int **socketNum** () const	Devuelve el descriptor del socket
virtual Boolean **handleRead** (unsigned char *buffer, unsigned bufferMaxSize, unsigned &bytesRead, struct sockaddr_in &fromAddress)	Lee el contenido del socket

Tabla 18: Métodos Groupsock

El diagrama de herencia se muestra en la ilustración 45.

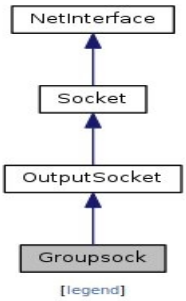


Ilustración 45: Diagrama herencia GroupSocket (fuente www.live555.com)

El diagrama de colaboración se muestra en la ilustración 46:

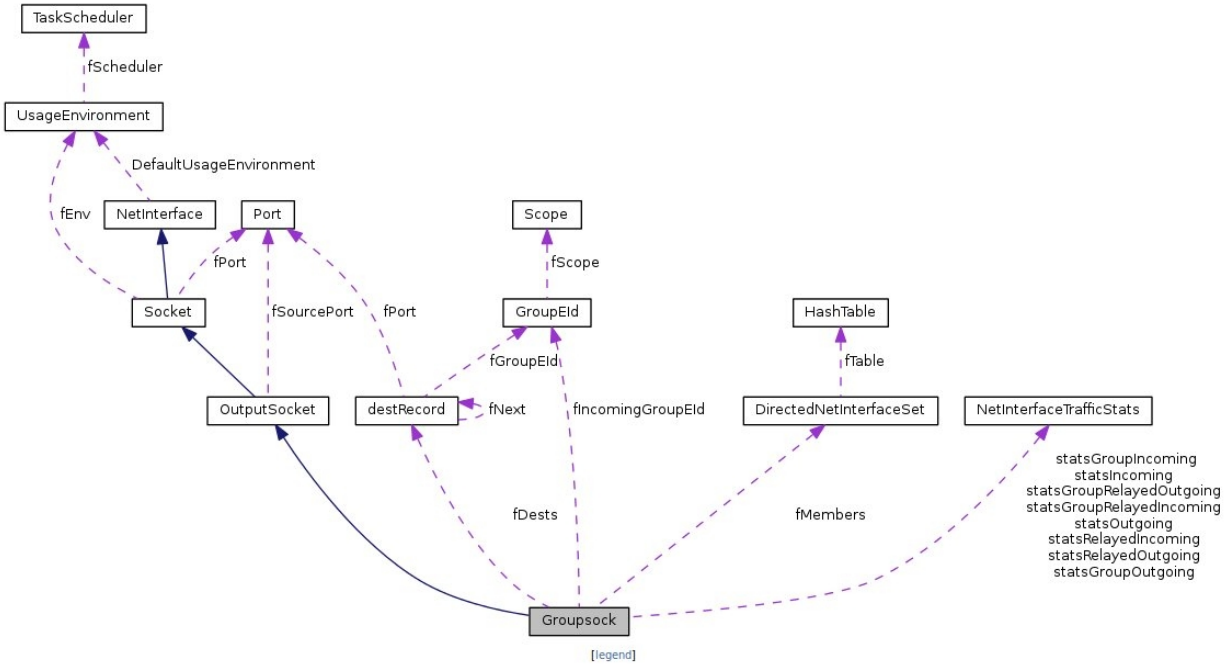


Ilustración 46: Diagrama herencia GroupSocket (fuente www.live555.com)

LiveMedia

Conjunto de clases que se encargan del streaming de media y de los formatos. La clase base de este conjunto es la clase *Medium.* A partir de esta clase *Medium* surge el resto de clases.

APÉNDICE D: Librería live555

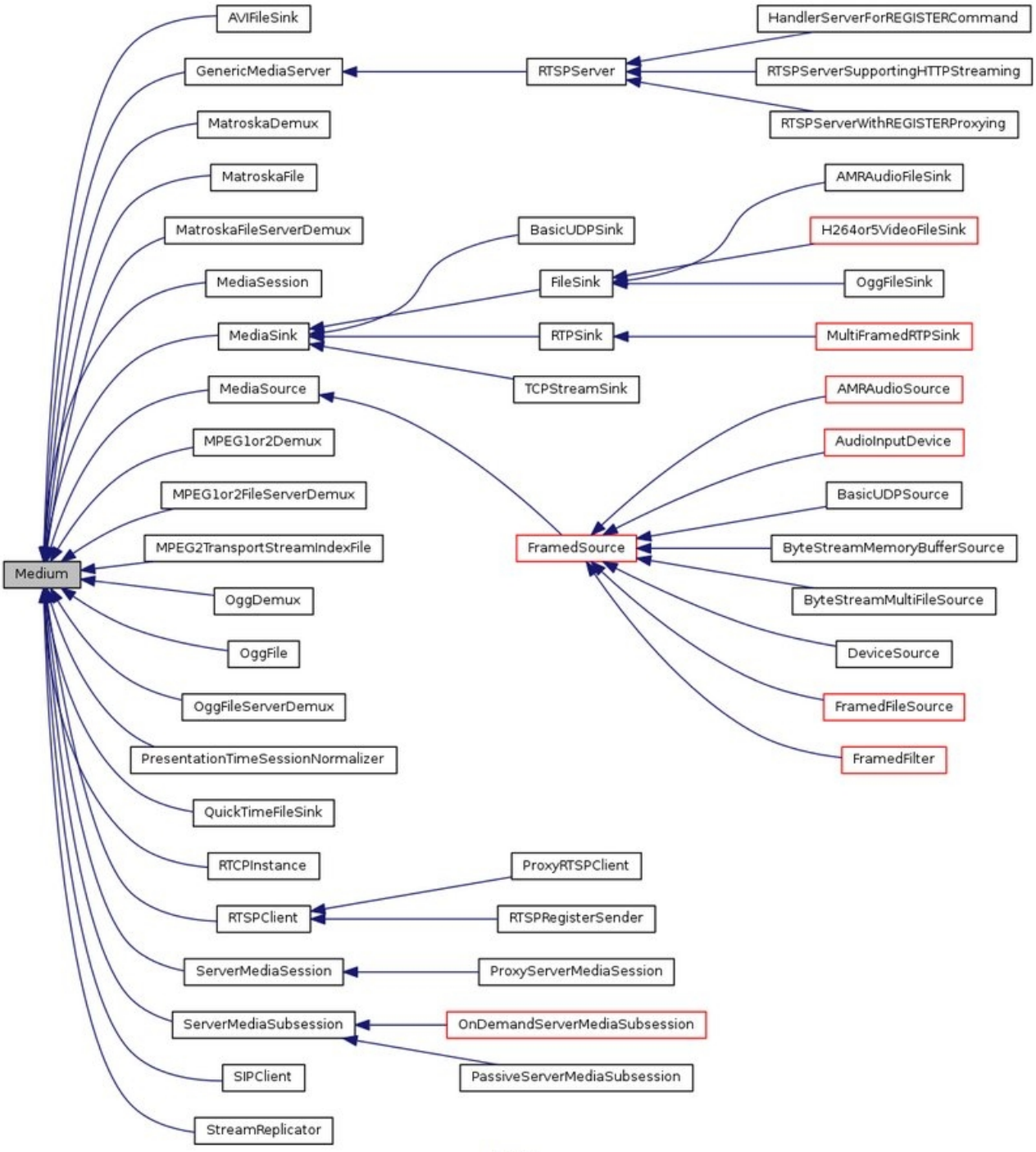


Ilustración 47: Diagrama de herencia de la clase Medium (fuente www.live555.com)

El diagrama de herencia se muestra en la ilustración 47:El diagrama de colaboración se muestra en la ilustración 48:

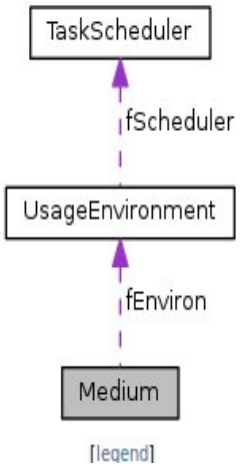


Ilustración 48: Diagrama de colaboración de la clase Medium (fuente www.live555.com)

Cabe destacar las clases que se muestran en la Tabla 19:

Clase	Uso
SIPClient	Encapsula un cliente SIP que realiza registro e inicio de sesión.
RTCPInstance	Representa una instancia RTCP.
RTPSink	Representa un origen de datos RTP. De esta clase heredan indirectamente, a través de la clase **MultiFramedRTPSink**, **SimpleRTP**, empleada para la transmisión de vídeo TS, y **MPEG1or2AudioRTPSink** y **MPEG1or2VideoRTPSink**,empleadas para la transmisión de vídeo MPEG1.
MPEG1or2Demux	Divide un flujo MPEG en audio y vídeo.

Tabla 19: Clases de la librería Media

Uso de liveMedia

Gestión de vídeo

Cabe destacar el uso de las clases que heredan de la clase RTPSink. Estas representan un origen de contenido que se transmite a través de RTP. La ilustración 49 muestra el diagrama de herencia de la clase RTPSink:

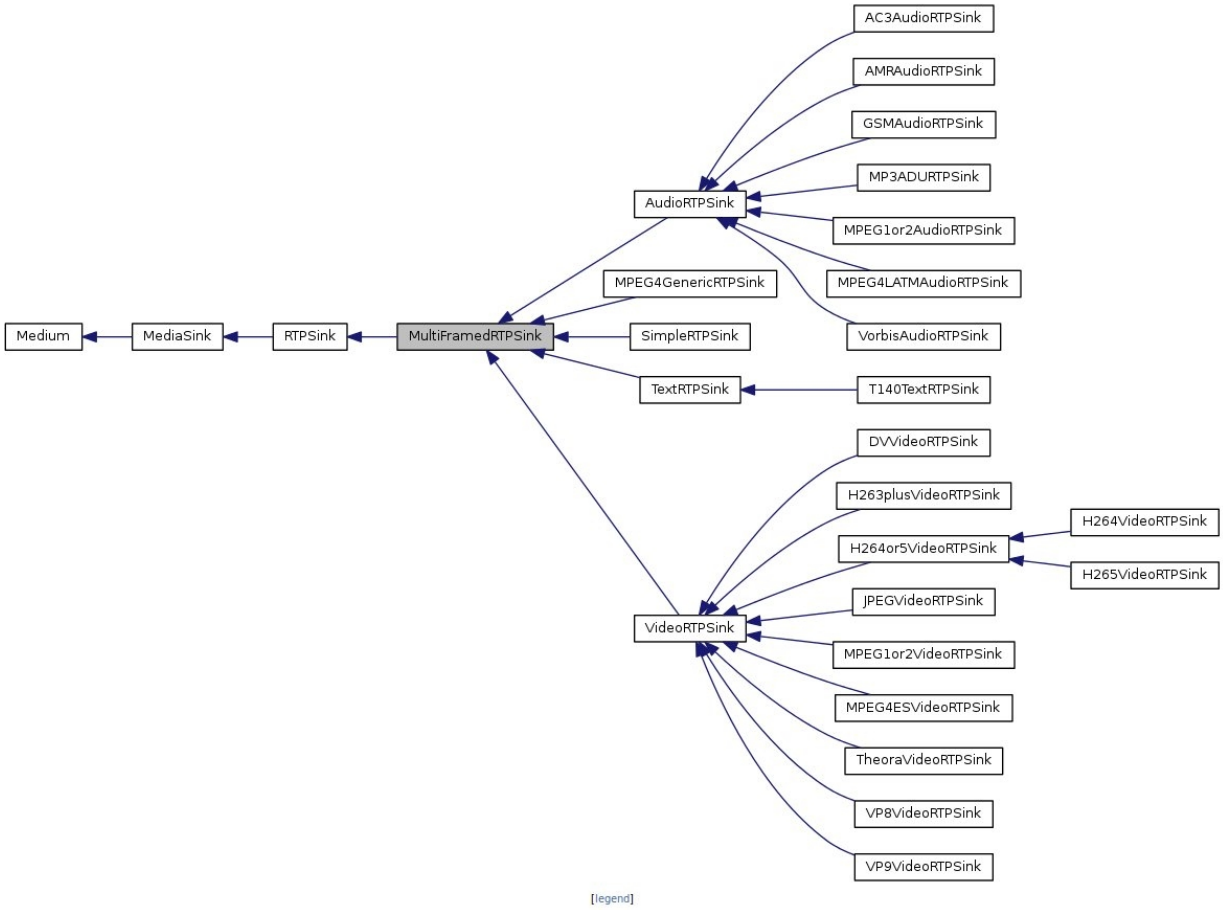


Ilustración 49: Diagrama de herencia de RTPSink

La clase empleada para transmitir contenido TS es la clase **SimpleRTP**.

Proporciona un constructor protegido:

SimpleRTPSink (UsageEnvironment &env, Groupsock *RTPgs, unsigned char rtpPayloadFormat, unsigned rtpTimestampFrequency, char const *sdpMediaTypeString, char const *rtpPayloadFormatName, unsigned numChannels, Boolean allowMultipleFramesPerPacket, Boolean doNormalMBitRule)

Podemos acceder a este constructor a través del método estático **createNew**:

static SimpleRTPSink * **createNew** (UsageEnvironment &env, Groupsock *RTPgs, unsigned char rtpPayloadFormat, unsigned rtpTimestampFrequency, char const *sdpMediaTypeString, char const *rtpPayloadFormatName, unsigned numChannels=1, Boolean allowMultipleFramesPerPacket=True, Boolean doNormalMBitRule=True)

Un objeto de este tipo nos permitirá iniciar y parar el flujo a través de las funciones **startPlaying** y **stopPlaying.**

A continuación mostramos su uso:

```
// Creamos el origen del contenido

RTPSink* videoSinkTS = SimpleRTPSink::createNew(*env, &rtpGroupsock, 33, 90000,
"video", "MP2T", 1, True, False);

….

// Inicia transmisión de contenido

videoSinkTS->startPlaying(*videoSourceTS, afterPlaying, videoSinkTS);

….

// Para transmisión de contenido

videoSinkTS->stopPlaying();
```

Las clases empleadas para transmitir MPEG1 y MPEG2 son **MPEG1or2AudioRTPSink** y **MPEG1or2VideoRTPSink**.

MPEG1or2AudioRTPSink hereda de la clase **AudioRTPSink**, que representa un origen de contenido de tipo audio.

Proporciona un constructor protegido:

MPEG1or2AudioRTPSink (UsageEnvironment &env, Groupsock *RTPgs)

Se puede acceder a él a través del método estático **createNew**:

static MPEG1or2AudioRTPSink * **createNew** (UsageEnvironment &env, Groupsock *RTPgs)

MPEG1or2VideoRTPSink hereda de la clase **VideoRTPSink**, que representa un origen de contenido de tipo vídeo.

Proporciona un constructor protegido:

MPEG1or2VideoRTPSink (UsageEnvironment &env, Groupsock *RTPgs)

Se puede acceder a él a través del método estático **createNew**:

static MPEG1or2VideoRTPSink * **createNew** (UsageEnvironment &env, Groupsock *RTPgs)

Para separar el vídeo del audio de un contenido tipo MPEG1 o MPEG2 se emplea la clase **MPEG1or2Demux**. Esta clase hereda directamente de la clase *Medium*.

Proporciona un constructor protegido:

MPEG1or2Demux (UsageEnvironment &env, FramedSource *inputSource, Boolean reclaimWhenLastESDies)

Se puede acceder a él a través del método estático **createNew**:

static MPEG1or2Demux * **createNew** (UsageEnvironment &env, FramedSource *inputSource, Boolean reclaimWhenLastESDies=False)

A continuación mostramos su uso:

```
// Creamos el origen del audio
RTPSink*      audioSink      =      MPEG1or2AudioRTPSink::createNew(*env, &rtpGroupsockAudio);
// Creamos el origen del video
RTPSink*      videoSink      =      MPEG1or2VideoRTPSink::createNew(*env, &rtpGroupsockVideo);
...
// Iniciamos vídeo
videoSink->startPlaying(*videoSource, NULL, this);
// Iniciamos audio
audioSink->startPlaying(*audioSource, afterPlaying, this);
```

```
....
// Paramos audio
audioSink->stopPlaying();
// Paramos vídeo
videoSink->stopPlaying();
```

Gestión SIP

Para la gestión SIP LiveMedia nos proporciona una clase, *SIPClient,* que encapsula un cliente SIP. Proporciona mecanismos para registrarse en IMS, iniciar y finalizar sesión.

La Ilustración 50 muestra el diagrama de herencia:

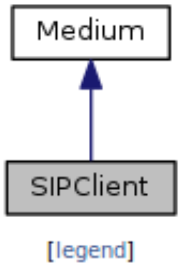


Ilustración 50: Diagrama de herencia de la clase SIPClient (fuente: http://live555.com)

La Ilustración 51 muestra el diagrama de colaboración.

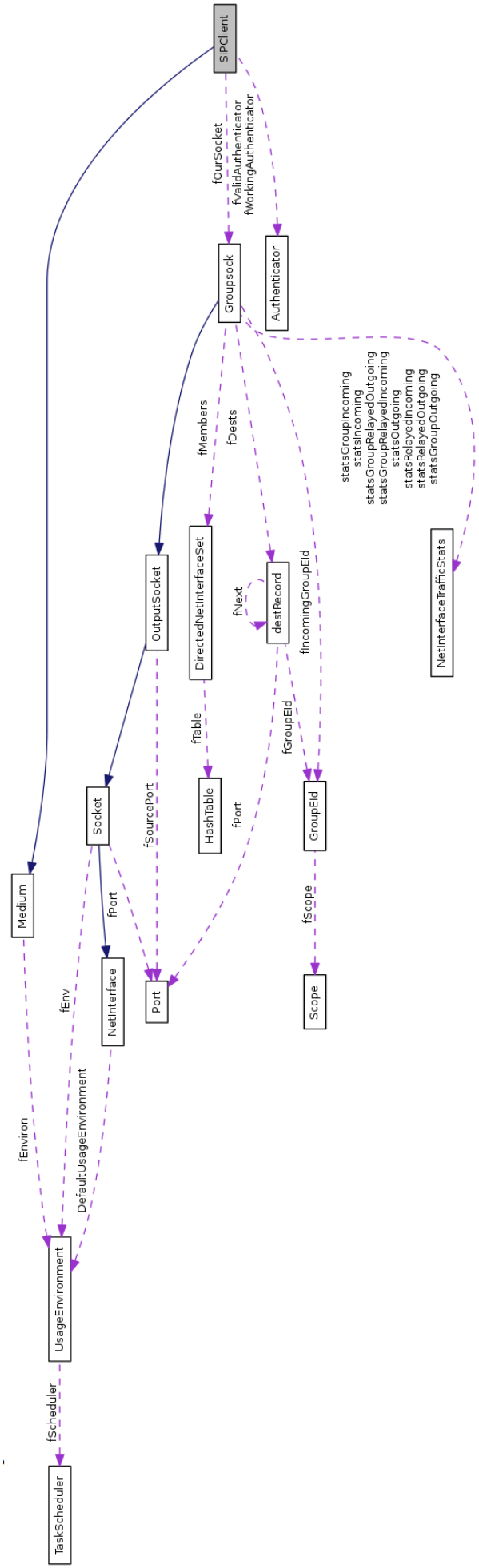


Ilustración 51: Diagrama de colaboración de SIPClient (fuente: www.live555.com)

Apéndice E

Intercambio de mensajes

Registro en IMS

A continuación detallamos los mensajes enviados y recibidos por el cliente.

Cliente envía REGISTER a IMS

```
REGISTER sip:open-ims.test SIP/2.0
Call-ID:
CSeq: 1 REGISTER
To:  <sip:nsc@open-ims.test:55557>
From: <sip:nsc@open-ims.test:55557>;tag=3335584429
Authorization: Digest username="nsc@open-ims.test", realm="open-ims.test", response="",
uri="sip:open-ims.test", algorithm=MD5
Via: SIP/2.0/UDP open-ims.test:55557
Contact:  <sip:nsc@open-ims.test:55557>
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

Recibe respuesta 401 de IMS

APENDICE E: INTERCAMBIO DE MENSAJES

```
SIP/2.0 401 Unauthorized - Challenging the UE
Call-ID:
CSeq: 1 REGISTER
To: <sip:nsc@open-ims.test:55557>;tag=d7837ce6bbd631122d10546eb75bb4cf-37a6
From: <sip:nsc@open-ims.test:55557>;tag=3335584429
Via: SIP/2.0/UDP open-ims.test:55557;received=127.0.0.1;rport=55557;received=127.0.0.1
Path: <sip:term@pcscf.open-ims.test:4060;lr>
Service-Route: <sip:orig@scscf.open-ims.test:6060;lr>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, PUBLISH, MESSAGE, INFO
Server: Sip EXpress router (2.1.0-dev1 OpenIMSCore (x86_64/linux))
Content-Length: 0
Warning: 392 127.0.0.1:6060 "Noisy feedback tells: pid=4381 req_src_ip=127.0.0.1 req_src_port=5060 in_uri=sip:scscf.open-ims.test:6060 out_uri=sip:scscf.open-ims.test:6060 via_cnt==3"
WWW-Authenticate: Digest realm="open-ims.test", nonce="db95bf28a4a4c34e149fad84fb7be772", algorithm=MD5, qop="auth,auth-int"
```

Envía respuesta al reto:

```
REGISTER sip:open-ims.test SIP/2.0
Call-ID:
CSeq: 2 REGISTER
To: <sip:nsc@open-ims.test:55557>
From: <sip:nsc@open-ims.test:55557>;tag=3647288890
Authorization: Digest username="nsc@open-ims.test", realm="open-ims.test", nonce="db95bf28a4a4c34e149fad84fb7be772", response="1d0f4521531c7a8b5eac771f09030e0f", uri="open-ims.test"
Via: SIP/2.0/UDP open-ims.test:55557
Contact: <sip:nsc@open-ims.test:55557>
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

Recibe OK del servidor:

```
SIP/2.0 200 OK - SAR succesful and registrar saved
Call-ID:
CSeq: 2 REGISTER
To: <sip:nsc@open-ims.test:55557>;tag=d7837ce6bbd631122d10546eb75bb4cf-5dcf
From: <sip:nsc@open-ims.test:55557>;tag=3647288890
Via: SIP/2.0/UDP open-ims.test:55557;received=127.0.0.1;rport=55557;received=127.0.0.1
P-Associated-URI: <sip:nsc@open-ims.test>
```

```
Contact: <sip:nsc@open-ims.test:55557>;expires=3600
Path: <sip:term@pcscf.open-ims.test:4060;lr>
Service-Route: <sip:orig@scscf.open-ims.test:6060;lr>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, PUBLISH, MESSAGE, INFO
P-Charging-Function-Addresses: ccf=pri_ccf_address
Server: Sip EXpress router (2.1.0-dev1 OpenIMSCore (x86_64/linux))
Content-Length: 0
Warning: 392 127.0.0.1:6060 "Noisy feedback tells:  pid=4380 req_src_ip=127.0.0.1 req_src_port=5060 in_uri=sip:scscf.open-ims.test:6060 out_uri=sip:scscf.open-ims.test:6060 via_cnt==3"
```

Inicio de sesión entre cliente y servidor

A continuación se muestran los mensajes que envían cliente y servidor.

Cliente envía mensaje INVITE al servidor indicando el media a recibir y el formato.

```
INVITE sip:bob@open-ims.test SIP/2.0
Call-ID:
CSeq: 3 INVITE
To:  <sip:bob@open-ims.test>
From: <sip:nsc@open-ims.test:55557>;tag=3231895250
Via: SIP/2.0/UDP open-ims.test:55557
Route: <sip:orig@scscf.open-ims.test:6060;lr>
Contact:  <sip:nsc@open-ims.test:55557>
Expires: 3600
User-Agent: SIP Video Server
Content-Type: application/sdp
Content-Length:119

v=0
o=nsc 3619790567 3619790567 IN IP4 192.168.1.5
s=test.ts
t=0 0
c=IN IP4 192.168.1.5
m=video 55557 RTP/AVP 33
```

Cuando el servidor recibe este mensaje, comprueba de la existencia del vídeo seleccionado. Como tiene el vídeo con el formato indicado, envía OK para indicar que acepta la sesión:

```
SIP/2.0 200 OK
Call-ID:
CSeq: 3 INVITE
To:  <sip:bob@open-ims.test>;tag=1220444376
From: <sip: <sip:nsc@open-ims.test:55557>>;tag=3231895250
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bK252.f38e2822.0
Via: SIP/2.0/UDP 127.0.0.1:6060;received=127.0.0.1;rport=6060;branch=z9hG4bK252.281ee5b3.0
Via: SIP/2.0/UDP 127.0.0.1:6060;branch=z9hG4bK252.181ee5b3.0
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bK252.e38e2822.0
Via: SIP/2.0/UDP open-ims.test:55557;received=127.0.0.1;rport=55557;received=127.0.0.1
Record-Route: <sip:mt@pcscf.open-ims.test:4060;lr>,<sip:mt@scscf.open-ims.test:6060;lr>,<sip:mo@scscf.open-ims.test:6060;lr>,<sip:mo@pcscf.open-ims.test:4060;lr>
Contact:  <sip:bob@open-ims.test:55556>
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

El cliente recibe el OK del servidor. Entonces envía un ACK, estableciéndose así la sesión.

```
ACK sip:bob@open-ims.test:55556 SIP/2.0
Call-ID:
CSeq: 3 ACK
To:  <sip:bob@open-ims.test>;tag=1220444376
From: <sip:nsc@open-ims.test:55557>;tag=3231895250
Via: SIP/2.0/UDP open-ims.test:55557
Route: <sip:mo@pcscf.open-ims.test:4060;lr>,<sip:mo@scscf.open-ims.test:6060;lr>,<sip:mt@scscf.open-ims.test:6060;lr>,<sip:mt@pcscf.open-ims.test:4060;lr>
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

Envío de comandos del cliente al servidor

Envío de comando PLAY

A continuación se muestran los mensajes intercambiados cuando un cliente envía un comando PLAY al servidor.

Cliente envía comando PLAY para iniciar reproducción de contenido:

```
INFO sip:bob@open-ims.test:55556 SIP/2.0
Call-ID:
CSeq: 4 INFO
To:  <sip:bob@open-ims.test>;tag=1220444376
From: <sip:nsc@open-ims.test:55557>;tag=3231895250
Via: SIP/2.0/UDP open-ims.test:55557
Route: <sip:mo@pcscf.open-ims.test:4060;lr>,<sip:mo@scscf.open-ims.test:6060;lr>,<sip:mt@scscf.open-ims.test:6060;lr>,<sip:mt@pcscf.open-ims.test:4060;lr>
Expires: 3600
User-Agent: SIP Video Server
Content-Type: plain/text
Content-Length:4

PLAY
```

El servidor envía respuesta OK para indicar que ha recibido el comando.

```
SIP/2.0 200 OK
Call-ID:
CSeq: 4 INFO
To:  <sip:bob@open-ims.test>
From: <sip: <sip:nsc@open-ims.test:55557>>;tag=3231895250
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bKf42.80b6ade7.0
Via: SIP/2.0/UDP 127.0.0.1:6060;received=127.0.0.1;rport=6060;branch=z9hG4bKf42.5cd30c5.0
Via: SIP/2.0/UDP 127.0.0.1:6060;branch=z9hG4bKf42.4cd30c5.0
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bKf42.70b6ade7.0
Via: SIP/2.0/UDP open-ims.test:55557;received=127.0.0.1;rport=55557;received=127.0.0.1
Contact:  <sip:bob@open-ims.test:55556>
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

Envío comando PAUSE

A continuación se muestran los mensajes intercambiados cuando un cliente envía un comando PAUSE al servidor.

Cliente envía comando PAUSE para pausar la reproducción de contenido.

```
INFO sip:bob@open-ims.test:55556 SIP/2.0
Call-ID:
CSeq: 5 INFO
```

```
To:  <sip:bob@open-ims.test>;tag=1220444376
From: <sip:nsc@open-ims.test:55557>;tag=3231895250
Via: SIP/2.0/UDP open-ims.test:55557
Route: <sip:mo@pcscf.open-ims.test:4060;lr>,<sip:mo@scscf.open-ims.test:6060;lr>,<sip:mt@scscf.open-ims.test:6060;lr>,<sip:mt@pcscf.open-ims.test:4060;lr>
Expires: 3600
User-Agent: SIP Video Server
Content-Type: plain/text
Content-Length:4

PAUSE
```

El servidor envía respuesta OK para indicar que ha recibido el comando.

```
SIP/2.0 200 OK
Call-ID:
CSeq: 5 INFO
To:  <sip:bob@open-ims.test>
From: <sip: <sip:nsc@open-ims.test:55557>>;tag=3231895250
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bK052.d575f717.0
Via: SIP/2.0/UDP 127.0.0.1:6060;received=127.0.0.1;rport=6060;branch=z9hG4bK052.e89e1865.0
Via: SIP/2.0/UDP 127.0.0.1:6060;branch=z9hG4bK052.d89e1865.0
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bK052.c575f717.0
Via: SIP/2.0/UDP open-ims.test:55557;received=127.0.0.1;rport=55557;received=127.0.0.1
Contact:  <sip:bob@open-ims.test:55556>
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

Envío comando STOP

A continuación se muestran los mensajes intercambiados cuando un cliente envía un comando PLAY al servidor.

Cliente envía comando STOP para parar la reproducción de contenido.

```
INFO sip:bob@open-ims.test:55556 SIP/2.0
Call-ID:
CSeq: 7 INFO
To:  <sip:bob@open-ims.test>;tag=1220444376
From: <sip:nsc@open-ims.test:55557>;tag=3231895250
Via: SIP/2.0/UDP open-ims.test:55557
```

```
Route: <sip:mo@pcscf.open-ims.test:4060;lr>,<sip:mo@scscf.open-ims.test:6060;lr>,<sip:mt@scscf.open-ims.test:6060;lr>,<sip:mt@pcscf.open-ims.test:4060;lr>
Expires: 3600
User-Agent: SIP Video Server
Content-Type: plain/text
Content-Length:4

STOP
```

El servidor envía respuesta OK para indicar que ha recibido el comando.

```
SIP/2.0 200 OK
Call-ID:
CSeq: 7 INFO
To: <sip:bob@open-ims.test>
From: <sip: <sip:nsc@open-ims.test:55557>>;tag=3231895250
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bKe42.f881a111.0
Via: SIP/2.0/UDP 127.0.0.1:6060;received=127.0.0.1;rport=6060;branch=z9hG4bKe42.82c09a43.0
Via: SIP/2.0/UDP 127.0.0.1:6060;branch=z9hG4bKe42.72c09a43.0
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bKe42.e881a111.0
Via: SIP/2.0/UDP open-ims.test:55557;received=127.0.0.1;rport=55557;received=127.0.0.1
Contact: <sip:bob@open-ims.test:55556>
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

Finalización de sesión entre cliente y servidor

A continuación se describen los mensajes intercambiados.

El cliente manda un mensaje tipo BYE al servidor.

```
BYE sip:bob@open-ims.test:55556 SIP/2.0
Call-ID:
CSeq: 8 BYE
To: <sip:bob@open-ims.test>
From: <sip:nsc@open-ims.test:55557>;tag=1850389210
Via: SIP/2.0/UDP open-ims.test:55557
Route: <sip:mo@pcscf.open-ims.test:4060;lr>,<sip:mo@scscf.open-ims.test:6060;lr>,<sip:mt@scscf.open-ims.test:6060;lr>,<sip:mt@pcscf.open-ims.test:4060;lr>
```

```
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

El servidor enviará un mensaje OK, cerrándose así la sesión entre cliente y servidor.

```
SIP/2.0 200 OK
Call-ID:
CSeq: 8 BYE
To:  <sip:bob@open-ims.test>
From: <sip: <sip:nsc@open-ims.test:55557>>;tag=1850389210
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bKc52.7ae7f9d1.0
Via: SIP/2.0/UDP
127.0.0.1:6060;received=127.0.0.1;rport=6060;branch=z9hG4bKc52.ace21205.0
Via: SIP/2.0/UDP 127.0.0.1:6060;branch=z9hG4bKc52.9ce21205.0
Via: SIP/2.0/UDP 127.0.0.1:4060;branch=z9hG4bKc52.6ae7f9d1.0
Via: SIP/2.0/UDP open-ims.test:55557;received=127.0.0.1;rport=55557;received=127.0.0.1
Contact:  <sip:bob@open-ims.test:55556>
Expires: 3600
User-Agent: SIP Video Server
Content-Length:0
```

Apéndice F

Referencias

[1]. GSM Association (2014). Rich Communications. <http://www.gsma.com/network2020/rcs/>. [Consulta: 15 de Enero de 2015]

[2]. GSM Association (2014). Voice and Video calls over LTE. <www.gsma.com/network2020/volte>. [Consulta: 15 de Enero de 2015]

[3] Vodafone (2014). Vodafone 4G Calling: Successful start to VolTE trials.<http://www.vodafone.com.au/doc/Vodafone_VoLTE_Trial.pdf>[Consulta: 17 de Enero de 2015]

[4] DOCOMO (2014). DOCOMO to Launch Japan's First VoLTE Service.<https://www.nttdocomo.co.jp/english/info/media_center/pr/2014/0514_00.html>. [Consulta: 20 de Enero de 2015]

[5] Mikka Poikselkä, Georg Mayer, Hisham Khartabil, Aki Niemi (2009). The IMS: IP Multimedia Concepts and Services, 3rd Edition. John Wiley & Sons.

[6] Kurose, J.F. y Ross, K. W (2013). Computer Networking. A Top-Down approach, 6th Edition. Pearson.

[7]. J. Rosenberg,H. Schulzrinne,G. Camarillo,A. Johnston,J. Peterson,R. Sparks,M. Handley,E. Schooler (Junio 202). SIP: Session Initiation Protocol, RFC 3261, Disponible en: *http://tools.ietf.org/html/rfc3261*

[8]. Into the 3GPP Evolved Packet System. <*http://www.in2eps.com/*>. [Consulta: 28 de Marzo de 2015]

[9].M. Handley,V. Jacobson,C. Perkins (Julio 2006). SDP: Session Description Protocol

APÉNDICE F: REFERENCIAS

RFC 4566. Disponible en: *https://tools.ietf.org/html/rfc4566*

[*10*]. H. Schulzrinne,S. Casner,R. Frederick,V. Jacobson (Julio 2003). RTP: A Transport Protocol for Real-Time Applications. RFC 3550. Disponible en: *https://www.ietf.org/rfc/rfc3550.txt*

[11] ETSI TS 182 027 V2.0.0 (2008-02). Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); IPTV Architecture; IPTV functions supported by the IMS subsystem.

[12] ETSI TS 183 063 V2.4.1 (2009-05). Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); IMS-based IPTV stage 3 specification.

[13]. J. Lindquist,J. Maenpaa,P. Rajagopal, X. Marjou (Junio 2010). Controlling Session Initiation Protocol (SIP)-Established Content Delivery Channels Using the Real-Time Streaming Protocol (RTSP). Disponible en: <*http://tools.ietf.org/html/draft-lindquist-sip-rtsp-02*>

[14]. The Open Source IMS Project. <http://www.openimscore.org/>. [Consulta: 28 de Marzo de 2015]

[15]. MySQL. <http://www.mysql.com/>[Consulta: 20 de Febrero de 2015]

[16]. VideoLAN. <*http://www.videolan.org/index.html*>. [Consulta: 21 de Febrero de 2015]

[17]. Bind9 – Debian Wiki. <*https://wiki.debian.org/Bind9*>. [Consulta: 22 de Febrero de 2015]