UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR
INGENIERÍA SUPERIOR DE TELECOMUNICACIÓN

DESARROLLO DE UNA APLICACIÓN DE AUDIOCONFERENCIA BASADA EN MULTICAST PARA ESCENARIOS DE RED CON SOPORTE IMS

PROYECTO FIN DE CARRERA

Autor: Mario Ramos Benito
Tutor: Iván Vidal Fernández
Marzo 2009

Resumen

Hoy en día la demanda de nuevos servicios en las redes de telecomunicaciones está llevando a los operadores a buscar nuevas fórmulas para satisfacer las necesidades de sus clientes. Para ello los operadores están buscando unificar todos los servicios que ofrecen tanto las redes de telefonía fija, la red de telefonía móvil e Internet. En este marco aparecen las Redes de Nueva Generación (NGN) que se presentan como un tipo de redes cuya arquitectura permite la unificación de los tres dominios mencionados anteriormente y además permite la creación y acceso a nuevos servicios de valor añadido. Este tipo de redes aseguran la calidad de servicio extremo a extremo (*QoS End-to-End*) siendo capaces de soportar servicios en tiempo real como la transmisión de la voz sobre IP (VoIP) asegurando unos parámetros de latencia y *jitter*. Otro tipo de servicios sensibles al retardo y el *jitter* tales como el video sobre demanda (VoD) o la televisión sobre IP (IPTV) también están soportados en este tipo de redes.

Dentro de las redes de nueva generación, el 3GPP (*Third Generation Partnership Project*) [19] propone una nueva arquitectura denominada subsistema multimedia IP (*IP Multimedia Subsystem*, en adelante IMS); mediante la producción de especificaciones y reportes técnicos. Gracias a esta arquitectura sería posible acceder a multitud de servicios, tanto servicios *peer-to-peer* como puede ser la transmisión de voz, como a los distintos servicios de valor añadido proporcionados por un proveedor, y por supuesto los anteriormente mencionados VoD y IPTV siguiendo en todos los casos procedimientos sencillos de control.

El presente proyecto tiene como objetivo la implementación una aplicación que sea capaz de generar la señalización adecuada (basada en los protocolos SIP [5] y SDP [18]) para permitir el establecimiento de una sesión de voz multiusuario. Esta herramienta utiliza el soporte de un servidor de aplicación desarrollado para gestionar sesiones multiusuario [11, 12, 13]. Del mismo modo se trata la emisión y recepción de los paquetes de voz enviados por multidifusión [15, 16] a los participantes de la conversación.

Palabras clave

Redes de Nueva Generación (NGN), Calidad de servicio extremo a extremo (QoS *End-to-End*), voz sobre IP (VoIP), Subsistema multimedia IP (IMS), Protocolo de Inicio de Sesión (SIP), Protocolo de Descripción de Sesión (SDP)

Abstract

Nowadays, the demand of the new services in the telecommunications networks tends the operators to search for new formulas in order to satisfy the customer necessities. To carry out this, the operators are trying to merge all the services that the landline, mobile infrastructure and Internet can offer. In this framework, the New Generation Networks (NGN) provides the capabilities to merge the three domains mentioned and besides allows the creation and the access to a new added value services. This kind of networks ensure end to end quality of service (QoS end to end) and can support real time services such as the transmission of the voice over IP packets (VoIP) guarantying certain parameters as the latency and jitter. Other type of services very sensitive to these parameters such as the Video on Demand (VoD) and the television over IP (IPTV) are also supported in these networks.

In the NGN framework, the 3GPP (Third Generation Partnership Project) [19] purposes a new architecture called IP Multimedia Subsystem (IMS) by means of delivering specifications and technical reports. With this architecture it is possible to access to many type of services, such as peer-to-peer with the voice over IP transmission, the new added value services provided by the operator or third party and of course the previous ones mentioned VoD and IPTV implementing in all the cases simple control mechanisms procedures.

The goal of this project is the implementation of an application capable of generating the proper signalling (based on the protocols SIP [5] and SDP [18]) to permit the establishment of a multiparty voice conference. This tool utilizes the support from an application sever developed to handle multiparty sessions [11, 12, 13]. In addition, it is also studied the transmission and reception of the voice packets to a multicast IP address to all the participants of the session.

Keywords

New Generation Networks (NGN), Quality of Service End to End (QoS End to End), voice over IP (VoIP), IP Multimedia Subsystem (IMS), Session Initiation Protocol (SIP), Session Description Protocol (SDP).

Agradecimientos

La defensa de este proyecto supone el fin de un largo camino cuyo comienzo tuvo lugar en septiembre de 1998. Durante todo este tiempo se han mezclado todo tipo de sensaciones: alegría, tristeza, cansancio, deseperación, frustración…etc.

Quiero agradecer en primer lugar el apoyo que me han brindado mis padres, quiero darles las gracias por todos los consejos que me han dado, por darme ánimos cuando me han visto decaido y por darme la oportunidad de haber estudiado una carrera que me apasiona y me ha posibilitado posicionarme profesionalmente en algo que me fascina y me ilusiona.

También quiero tener unas palabras de agradecimiento a mi hermana y a mis abuelos que siempre me han apoyado en los momentos difíciles.

Es cierto que la carrera me apasiona, pero lo mejor sin duda de este periodo universitario ha sido la gente con la que me he encontrado. No quiero ponerme a enumerar porque no me gustaría olvidarme de nadie, pero quien lea este documento sabe bien a quien me refiero.

Quiero mencionar tambien a una persona muy especial para mi, Anabel, que en esta última fase de la carrera me ha presatado un gran apoyo.

Por último, a toda la comunidad universitaria, desde el personal docente por haberme ayudado a formarme hasta el resto del personal de la universidad. Muchas gracias también a Iván Vidal por haberme ayudado y aconsejado en esta última fase de mi carrera.

Tabla de contenidos

Índice de figuras

Índice de tablas

1. Introducción

En este primer capítulo exponen las razones que motivaron la realización de este proyecto y después se definen los objetivos que se desean alcanzar así como la estructura del documento.

Además se exponen los elementos claves en el desarrollo del proyecto y se define el marco tecnológico alrededor del cual va a girar la aplicación implementada.

1.1 Motivación

Desde la eclosión a nivel usuario de Internet a principios de los años 90 y hasta el día de hoy, el ser humano ha sentido la necesidad a medida que han pasado los años de acceder a contenidos y servicios que hace unas décadas parecían una utopía.

Si pensamos en las comunicaciones móviles la aparición de GSM supuso una revolución en las comunicaciones interpersonales entre los individuos. En un principio el solo hecho de poder establecer comunicaciones de voz de manera independiente de la ubicación del individuo supuso un logro que impactó en muchos ámbitos de la sociedad. Y no solo eso, los servicios de mensajería instantánea SMS supusieron una revolución ya no solo entre la comunicación entre personas sino que abrieron nuevos canales de publicidad y nuevas formas de llegar al usuario final que hace no mucho tiempo parecía poco menos que imposible.

A medida que los seres humanos nos hemos ido habituando a lo que hace unos años era una utopía hemos ido buscando nuevos retos y a la vez la sociedad nos ha ido creando unas necesidades que antes no teníamos. A día de hoy es raro pensar que alguien no dispone de un terminal de telefonía móvil.

La evolución de las comunicaciones móviles ha dado lugar a que no solo se conciban este tipo de redes para gestionar las comunicaciones de voz si no que actualmente se conciben como redes de datos que proveen acceso a todo tipo de contenidos y servicios. Esto ha sido posible gracias a la aparición de las redes GPRS 2.5G y en último lugar a las redes 3G de tercera generación.

Sin embargo desde el primer momento, se produjo una segregación entre lo que es el dominio de circuitos CS provenientes de la red GSM y el dominio de paquetes que surge con la aparición de las redes de datos 2G y 3G.

Por otra parte, la eclosión de Internet en la sociedad coincide con la aparición del GSM. Internet se concibe hoy en día como una red basada en tecnología IP para el intercambio de contenidos multimedia, navegación, email, etc… Sin embargo con el paso de los años y a medida que la sociedad demanda una serie de servicios, se pretende que Internet sea capaz de transportar cualquier tipo de servicio asegurando calidad extremo a extremo y que garantice unos parámetros de calidad que aseguren la transmisión de aplicaciones de voz en tiempo real bajo ciertas restricciones de retardo y *jitter*.

Las redes de nueva generación (NGN) aseguran calidad de servicio extremo a extremo y surgen debido a la necesidad de proporcionar una serie de servicios que las redes de datos que existían previa a su aparición no eran capaces de soportar.

El IMS *(IP Multimedia Subsystem)* nace como arquitectura de plano de control en el proceso de estandarozación de UMTS (3G). Posteriormente se utilizó como arquitectura de referencia en varias propuestas NGN relevantes (ej. La desarrollada por el grupo TISPAN de ETSI). El objetivo consiste en aunar las características de lo que podríamos llamar dominio fijo (Internet) y el dominio móvil permitiendo al usuario que se provisione tanto los servicios tradicionales de voz, como de servicios multimedia y de nueva generación o de los servicios tradicionalmente ofrecidos en las redes de paquetes (por ejemplo, los basados en la web).

La arquitectura del IMS utiliza un lenguaje descriptivo llamado SDP (*Session Description Protocol*) [18] para la definición de sesiones multimedia, las cuales posibilitan la entrega de servicios de valor añadido al usuario final. Por tanto, podemos considerar que la arquitectura de IMS está pensada para ser capaz de ofrecer acceso al usuario final a un rango completo y heterogéneo de servicios de valor añadido – tanto servicios de voz clásicos como servicios Web, e incluso a servicios creados a partir de la unión de otros servicios a partir de la definición genérica y estandarizada del servicio o servicios que se quieran proporcionar.

Además, esta arquitectura utiliza como señalización en el plano de control el protocolo SIP (*Session Initiation Protocol*) [5], con el que se logra de una manera sencilla poder

realizar las funcionalidades de control de sesión (establecimiento, mantenimiento y liberación), de manera que la red consigue entregar los servicios asociados a cada red en cuestión.

En este marco, aparece el concepto de las aplicaciones multiusuario en las que varios individuos comparten una serie de contenidos haciendo uso de los protocolos existentes en la red. En lo referido a las comunicaciones de voz parece interesante pensar el hecho de que varios usuarios se puedan unir en una sola conferencia y mantener una conversación simultánea entre todos ellos.

Sin embargo, las especificaciones desarrolladas hasta la fecha por el 3GPP no proporcionan una plataforma general que soporte la provisión eficiente de estos servicios al usuario final. Así, las soluciones desarrolladas son típicamente específicas para cada servicio, utilizándose un esquema de transmisión unicast en el plano de usuario. A este respecto, en [11, 12, 13] se propone una plataforma general para la provisión eficiente de servicios multiusuario en redes con plano de control IMS, mediante el empleo de tecnologías multidifución (*multicast*). El componente central de eta propuesta es un Servidor de Aplicación Multiusuario (MAS, Multiparty Application Server), que se encarga de gestionar la señalización SIP necesaria en IMS para configurar apropiadamente el servicio multiusuario. En este contexto, y como veremos a continuación, .el objetivo principal de este Proyecto Fin de Carrera, consiste en el desarrollo de una aplicación de audioconferencia basada en tecnologías de entrega multidifusión sobre entornos de red IMS.

1.2 Objetivos

El objetivo de este proyecto es desarrollar un sistema que permita establecer una sesión de voz multiusuario. El sistema tendrá dos partes bien diferenciadas: un plano de control o señalización en donde se tratará la negociación de la sesión mediante el protocolo de señalización SIP/SDP y un plano de datos en cual la voz que ha sido codificada y encapsulada en paquetes IP se envía a través de multidifusión IP a los usuarios participantes de la sesión.

El marco en el que se encuadra el desarrollo de la aplicación desarrollada gira en torno al IMS. Se dispone de un servidor de aplicaciones: el MAS (*Multiparty Aplication Server*) cuya misión será procesar y reenviar las peticiones y respuestas que son enviadas por los usuarios. Haremos un estudio pormenorizado del IMS y los

servidores de aplicaciones en el capítulo 2. Precisamente el objetivo de este proyecto en lo referido al plano de control es la implementación de estos usuarios o clientes SIP que serán desarrollados en JAVA y su integración con el MAS. Se definen dos tipos de clientes: un cliente llamante que origina la llamada a un grupo se usuarios y el usuario llamado que recibe la invitación de la llamada y acepta el establecimiento de sesión. Por tanto, tendremos 1 usuario llamante y 1 o más usuarios llamados.

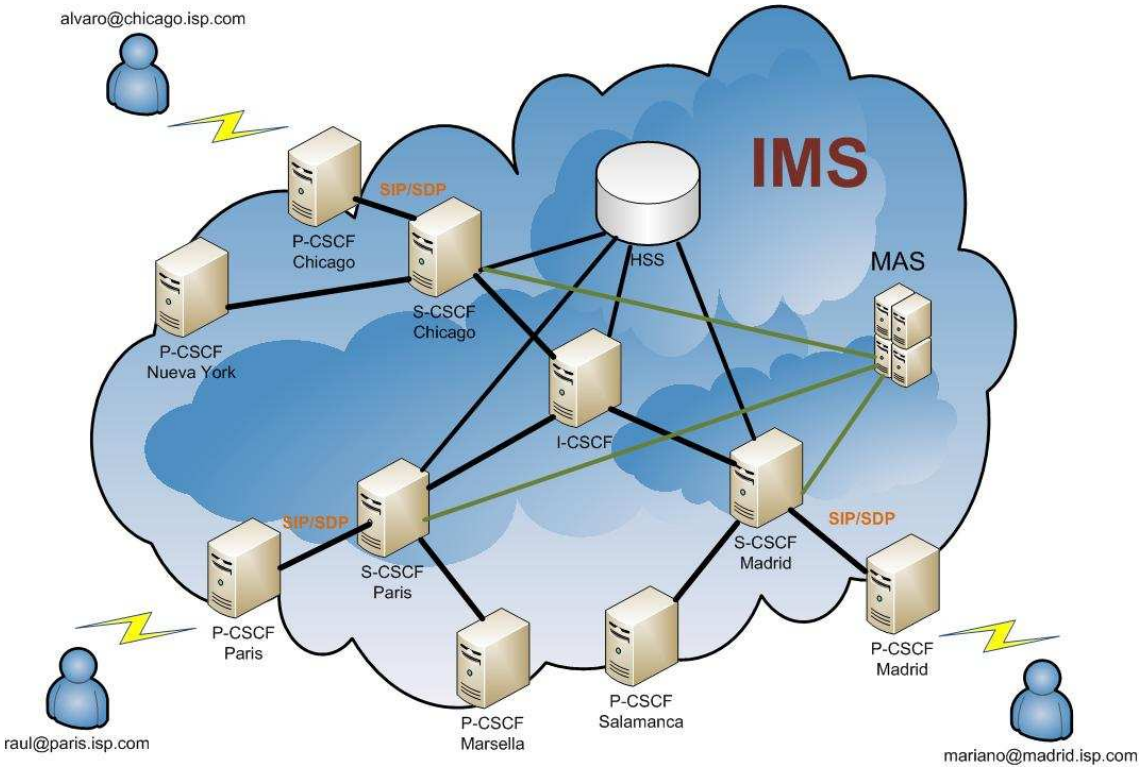

Figura 1 – Sistema IMS

La idea es que Álvaro que se encuentra registrado en el P-CSCF de Chicago establezca una sesión de voz multiusuario con Mariano que se encuentra en Madrid y Raúl que se encuentra en París. En primer lugar se producirá una negociación SIP/SDP y una vez que la sesión se haya establecido, los usuarios estarán en disposición de comenzar la sesión de voz multiusuario.

En el plano de datos se tratará el envío de la voz paquetizada en tramas de UDP sobre IP mediante difusión IP multicast. La voz será codificada mediante un codec que se negoció en el establecimiento de sesión.

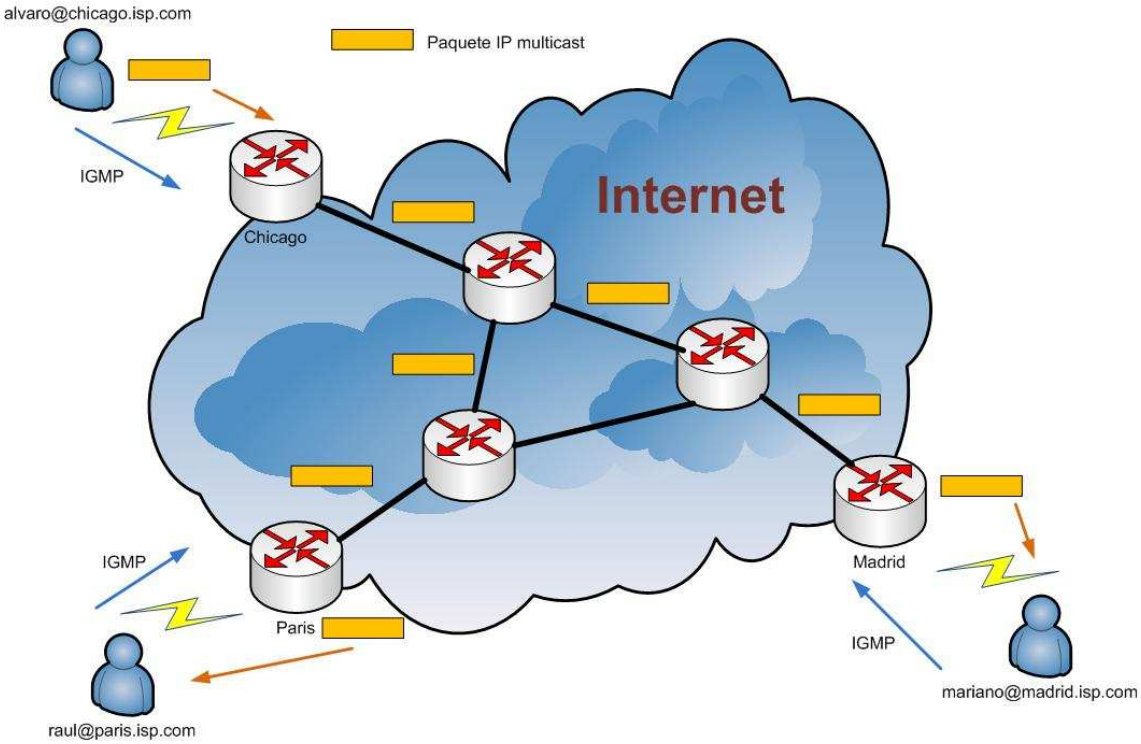

Figura 2 – Sesión multiusuario

Para ello el cliente ha de subscribirse a un grupo *multicast* mediante el protocolo IGMP cuyo funcionamiento será explicado en el apartado 2.6 y para que los paquetes lleguen a los usuarios destinatarios será necesaria la implementación de protocolo que permita la multidifusión de los paquetes de voz a lo largo de toda la red y cuyo estudio tendrá lugar en la sección 2.6.4.

En el ejemplo de la figura 2, Álvaro que se encuentra en Chicago inicia la sesión de voz multiusuario con los destinatarios Raúl que se encuentra en París y Mariano que se encuentra en Madrid. Los paquetes de voz atraviesan la red y son entregados finalmente a los usuarios finales.

1.3 Estructura del documento

El presente documento se encuentra estructurado en varios capítulos. A continuación se describe de manera breve el contenido de cada capítulo:

- Capítulo 1 – Introducción: El presente capítulo donde se establece el marco de estudio del proyecto, presentando la necesidad de la implementación desarrollada así como los objetivos que se intentan alcanzar.

- Capítulo 2 – Estado del arte. Describe el entorno tecnológico en el que el proyecto se va a desarrollar. En el caso que nos ocupa, el núcleo será el IMS cuya arquitectura se describirá en modo detallado así como los protocolos que participan en el sistema, principalmente SIP y SDP. También se trata la multidifusión IP y se hace un estudio de los codecs de voz más representativos.

- Capítulo 3 – Identificación de los requisitos del sistema. Se definen los requisitos del sistema tanto en el plano de señalización como en el plano de datos

- Capítulo 4 – Implementación de los elementos del sistema. Una vez conocidos los requisitos, nos centraremos en la implementación de los mismos. Se describirán las herramientas utilizadas y los procedimientos que se ha llevado a cabo para realizar dicha implementación.

- Capítulo 5 – Escenario de pruebas. Define la creación de un escenario de pruebas donde validar el funcionamiento del sistema acorde a los requisitos. Se especifican la ejecución de una serie de pruebas y se hace una valoración de los resultados obtenidos.

- Capítulo 6 – Conclusiones y líneas futuras. Se hace una presentación de las conclusiones extraídas de la realización del proyecto y se indican las líneas futuras de investigación que pueden ser identificadas con el presente trabajo.

- Capítulo 7 – Referencias. Se listan todas las referencias y documentos que han sido utilizados en el desarrollo de este proyecto.

- Listado de anexos. Se adjuntan un listado de anexos como extensión a ciertos capítulos del documento.

2. Estado del arte

En este apartado se pretende dar una descripción detallada del entorno tecnológico en el que se ha desarrollado el proyecto. Se hará un estudio de la evolución del sistema IMS, de los elementos que forman la arquitectura, las funciones de cada uno de ellos y los protocolos empleados en el sistema.

2.1 ¿Qué es el IMS?

Podemos definir el IMS como una especificación del *3rd Generation Partnership Project* 3GPP que permite el provisionamiento de servicios con mecanismos de calidad de servicio (*Quality of Service* QoS) y facilita el establecimiento de sesiones multimedia en redes de IP tales como voz sobre IP (VOIP), *Push to Talk* (PTT), *Video Calling, Video Streaming, Video Sharing, Audio Streaming,* etc... Se puede decir que el IMS pertenece al *core* de red y que pretende servir como nexo de unión de diferentes tecnologías.

Desde el punto de vista del usuario final, el IMS permite el establecimiento de sesiones multimedia entre dos o más usuarios finales o el acceso a contenidos desde un usuario final a un servidor de aplicaciones donde el usuario puede descargar fotos, videos o cualquier combinación de estos. Además proporciona una plataforma de servicios de modo que el usuario pueda acceder a ellos. Los servicios se encuentran en una capa lógica y los encargados de llevar a cabo la implementación de los servicios son los servidores de aplicaciones (*AS Application Servers*).

Además se persigue desarrollar un tipo de sistema basado en una tecnología IP de conmutación de paquetes para ofrecer un tipo de servicio que tradicionalmente correspondían a las redes de telefonía de circuitos conmutados (llamadas de voz). El IMS también contempla la integración de ambos dominios proporcionando un *break-out* y un *break-in* hacia/desde la PSTN (*Public Switched Telephone Network* – Red Pública de Circuitos Conmutados).

2.2 Origen y evolución del IMS

Se puede decir que el origen de las redes móviles y más tarde de las redes de nueva generación parte de la creación de la estandarización de la red GSM a finales de los años 80 y principio de los 90. La última especificación fue definida en 1998. En el mismo año aparece el consorcio 3GPP (*3rd Generation Partnership Project*) que agrupa a diferentes corporaciones de Europa (ETSI), Japón (ARIB/TTC), China (CCSA), Estados Unidos (ATIS) y Corea del Sur (TTA) y que empieza a trabajar en la evolución de GSM hacia las redes 2.5G (GPRS) y 3G (UMTS) [19]

La primera contribución del 3GPP es la *Release 99*. Principalmente aparece una nueva red de acceso radio (UTRAN) que se conecta a la red de paquetes a través de la SGSN y a la red de circuitos mediante el MSC. Otro aspecto especialmente novedoso de UMTS, es la definición de una nueva arquitectura de servicios abierta (OSA, *Open Service Architecture*) y flexible que permita la creación de una gran variedad de servicios y aplicaciones para el usuario utilizando un conjunto de capacidades de servicio que son las entidades a estandarizar y un proceso de gestión denominado Entorno Propio Virtual (*VHE, Virtual Home Environment*). El VHE proporciona al usuario la posibilidad de disponer de un entorno de servicios personalizado cualquiera que sea la red o terminal que esté usando, y cualquiera que sea su posición. La estandarización de un interfaz OSA entre las aplicaciones y los servidores de capacidades de servicio facilita la incorporación al sector de las telecomunicaciones móviles 3G de nuevos proveedores de servicios que pueden desarrollar y ofrecer nuevos servicios haciendo uso de dicha interfaz OSA.

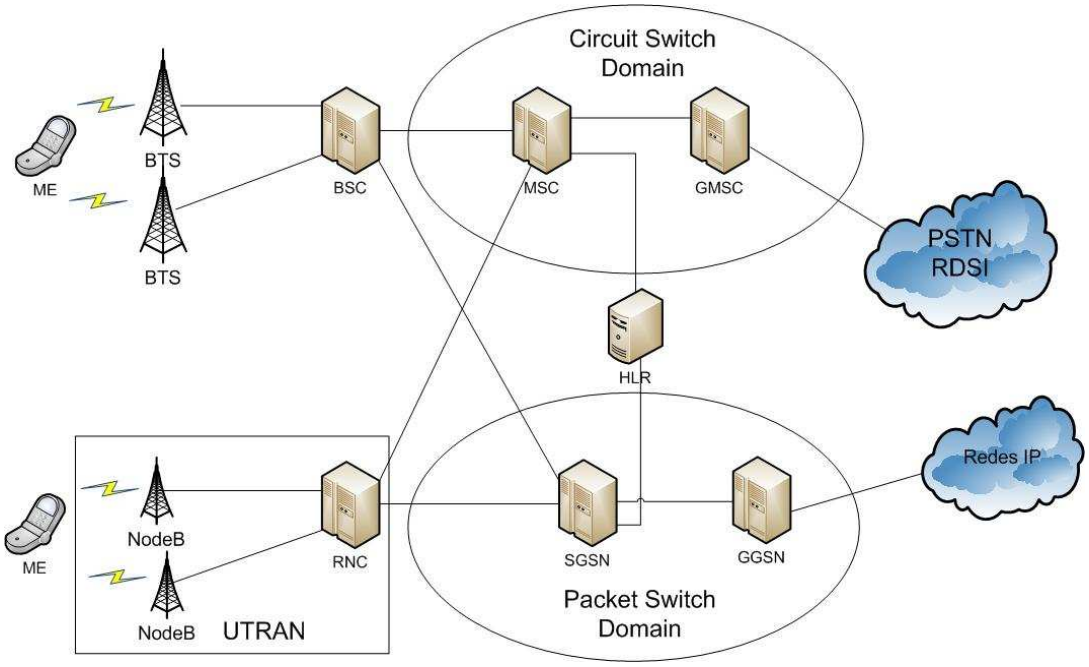

Figura 3 – 3GPP Release 99

Tras la R99 llega la *Release 4* cuya principal novedad supone la división del MSC en dos entidades:

- *MSC Server* que se encarga del plano de control
- *Media Gateway* (MGW) cuya misión es gestionar el tráfico de usuario así como hacer *transcoding* de voz y cancelación de eco.

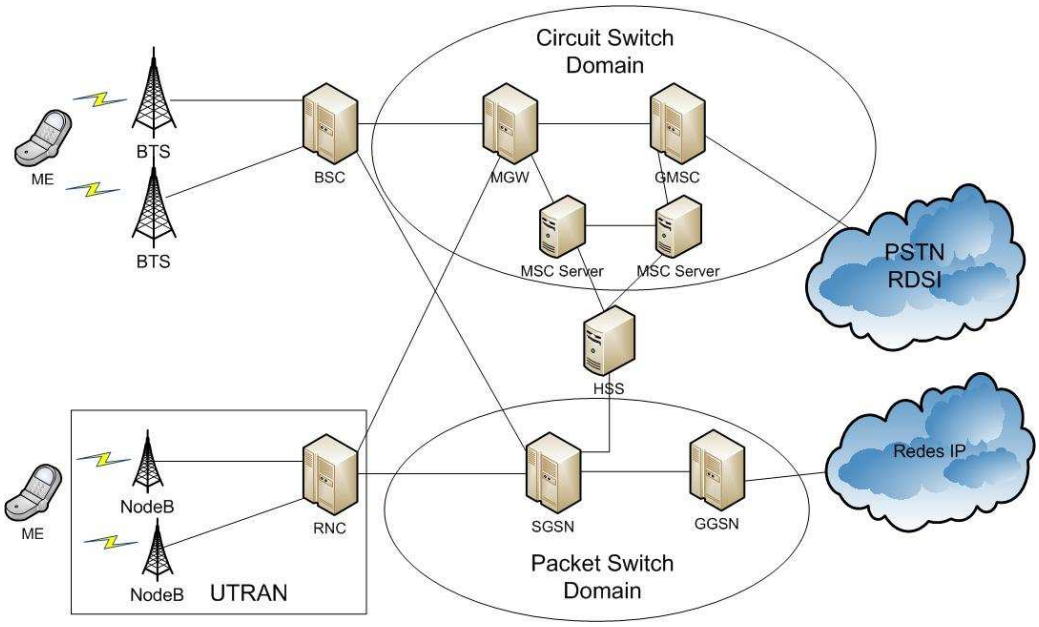

Figura 4 – 3GPP Release 4

Finalmente la *Release 5* del 3GPP introduce el IMS. El IMS se concibe como una plataforma estandarizada e independiente cuya arquitectura esta basada en tecnología IP que permite la interconexión con diferentes tipos de redes tanto de voz o datos tales como la PSTN, RDSI, Internet y redes móviles como por ejemplo GSM, CMDA, etc… El IMS permite establecer conexiones IP *peer-to-peer* con cualquier tipo de cliente que requiera calidad de servicio QoS. Además de gestionar el establecimiento de sesión, el IMS también se hace cargo de otras funcionalidades tales como el registro, seguridad, facturación, roaming, etc... Además en esta *Release 5* se definen los interfaces que conectan los distintos elementos de la arquitectura y los protocolos participantes en ella. El protocolo predominante en el IMS es SIP (*Session Initiation Protocol*) [5] que se encarga de varios aspectos tales como el registro, subscripción, negociación de parámetros de calidad de servicio, liberación de sesión, etc.…

La *Release 6* del 3GPP añade aspectos y funcionalidades que quedaron por definir en la *Relase 5* como por ejemplo el *Push To Talk* (PTT). Por tanto, se puede ver la R6 como un complemento a la R5 y que termina de especificar la definición de servicios y define la interconexión a otro tipo de redes de acceso como las WLAN.

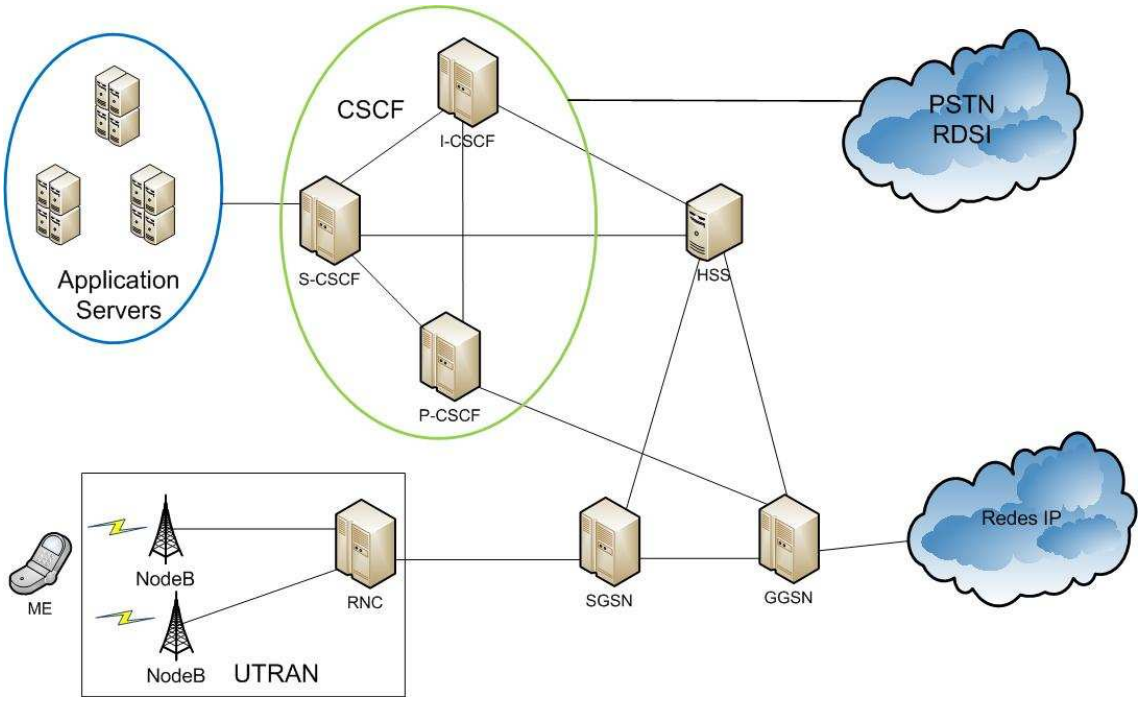

Figura 5 – 3GPP Release 5/ Release 6

Tras la *Release 6* de UMTS llega la *Release 7* con la aparición de HSPA que incorpora sobre todo mejoras en el interfaz radio aumentando el *throughput* (hasta 14 Mbps en el

DL) y mejorando la capacidad del medio haciendo posible la transmisión de la voz sobre ip en tiempo real y con unas prestaciones de retardo y *jitter*. La mejora en el interfaz radio se debe a la aparición de antenas tipo MIMO (*Multiple Input, Multiple Output*) que posibilitan el envío de datos a unas tasas superiores a las obtenidas en *Releases* anteriores. En la parte del *core* se sigue manteniendo la estructura del IMS pero aparece nuevas funcionalidades tales como la continuidad y mantenimiento de las conversaciones de voz cuando un usuario migra de una red a otra (por ejemplo de una WLAN a una 3G).

La última contribución del 3GPP es la *Release8* que supone un cambio importante en la arquitectura de las redes 3G. Aparecen nuevos elementos/nodos tanto en la parte de acceso radio como en el *core*. En la radio se introduce el *evolved NodeB* (eNodeB) que emplea una modulación en el interfaz radio (OFDM *Orthogonal Frequency Division Multiplex*) totalmente distinta a las de las anteriores *releases*. En el *core* aparecen nuevo elementos que se integran y se conectan a los nodos de las anteriores *releases* mediante la definición de nuevos interfaces. Del mismo modo aparece un nuevo interfaz que permite conectar los elementos del *core* con el IMS. La arquitectura resultante es el EPS (*Evolved Packet System*) que comprende la definición de la arquitectura en la parte radio LTE (*Long Term Evolution*) y la arquitectura en el *core* de red EPC (*Evolved Packet Core*).

2.3 Arquitectura del sistema IMS

A continuación se muestra un diagrama detallado de la arquitectura del IMS con sus correspondientes interfaces.

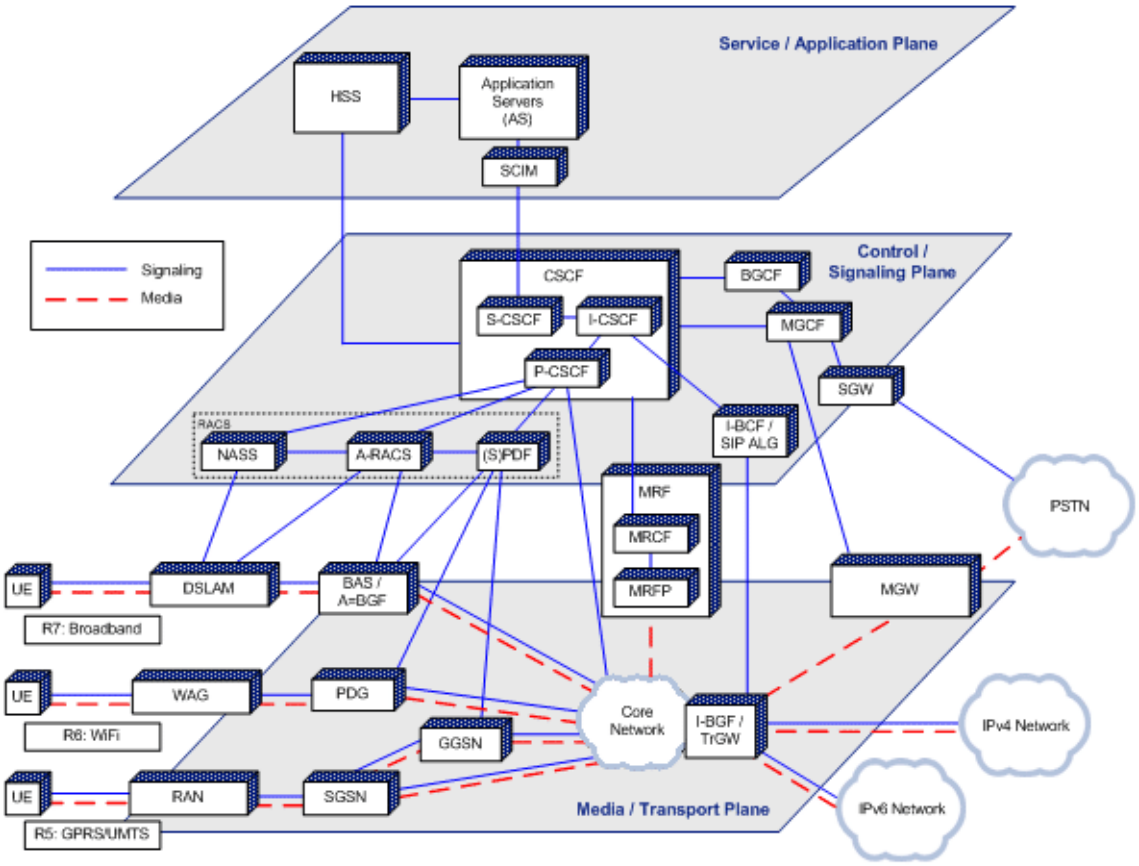

Figura 6 – Arquitectura del IMS

Dentro de la arquitectura del sistema IMS, podemos agrupar las entidades en función del papel que desempeñan.

2.3.1 CSCF

El CSCF (Call Session Control Function) se puede considerar como el núcleo de IMS. Realiza funciones de control de sesión y de enrutado. A su vez el CSCF se puede descomponer a su vez en tres entidades:

- *Proxy-CSCF*. Se trata del primer punto de contacto del usuario dentro del IMS. Todo el tráfico de señalización de usuario atraviesa el P-CSCF. EL P-CSCF se encarga de procesar los mensajes que envía el usuario validando la petición y reenviándola a los destinatarios.

- *Interrogating*-CSCF. Su función es la determinar dentro de la red del operador contra que S-CSCF el usuario se ha de registrar. Para ello, cuando el I-CSCF recibe la petición de registro del P-CSCF manda una consulta al HSS para determinar si el usuario esta permitido registrarse en la red del operador. En

caso afirmativo el HSS devuelve el S-CSCF asignado al usuario. El I-CSCF también interviene en el proceso de establecimiento de llamada. De esta manera cuando el iniciador de la llamada desea establecer una llamada manda un INVITE indicando en la URI destino el destinatario de la llamada. Este INVITE se manda al P-CSCF de la red del local y el P-CSCF lo manda al S-CSCF de la red local. Es entonces cuando el S-CSCF extrae el dominio y hace una petición DNS para saber a que I-CSCF se lo tiene que mandar. Cuando el INVITE entra por el I-CSCF de la red visitada, interroga al HSS para saber con que dirección se ha registrado el destino y esta información se envía al S-CSCF de la red visitada. Además el I-CSCF puede implementar la funcionalidad llamada *Topology Hiding Inter-network Gateway* (THIG). THIG puede utilizarse para ocultar la configuración de la topología y capacidades. de la red hacia operadores externos.

- *Serving*-CSCF. Se puede considerar como el cerebro del IMS y se localiza en la red del operador. Realiza funciones de control de sesión y registro del usuario. Mientras que un usuario se encuentra activo en la sesión el S-CSCF mantiene el estado de la sesión e interactúa con otras plataformas para llevar por ejemplo el control de la tarificación

2.3.2 HSS

El *Home Subscriber Server* (HSS) es la entidad encargada de almacenar los datos de usuario y de servicios del IMS. Incluye información sobre identidades de usuarios, información de registro, parámetros de acceso y servicios.

Las identidades de los usuarios pueden ser públicas o privadas. La identidad privada se asigna en la red del operador donde el usuario reside y se utiliza para los procesos de registro y autorización mientras que la identidad pública es aquella que otros usuarios pueden utilizar para contactar con el usuario en cuestión. Los parámetros de acceso del IMS se usan para establecer sesiones y mandar información de autenticación, *roaming* y los posibles S-CSCF a los que se puede conectar el usuario de manera que el I-CSCF utilizará esta información para seleccionar el S-CSCF apropiado para el usuario.

Además el HSS puede contener el registro HLR/AuC para la autenticación de usuarios en los dominios CS y PS de GSM/UMTS.

Muy importante es la información que el HSS almacena en los IFC (*Initial Filter Criteria*) cuyo propósito es proveer un mecanismo de filtrado que permite mandar los mensajes SIP al servidor de aplicaciones correspondiente a la hora de provisionar un servicio.

Puede ocurrir que haya más de un HSS en la red de manera que tendremos que saber a que HSS está asociado un determinado usuario. Para ello se utiliza el *Subscription Locator Function* (SLF) que determina a que HSS está vinculado el usuario.

2.3.3 AS

Desde el punto de vista de la arquitectura del IMS, los servidores de aplicaciones (AS) se encuadran en el plano de aplicación. Se tratan de entidades que proporcionan servicios de valor añadido al sistema.

Un servidor de aplicaciones reside en la red del operador al que pertenece el usuario o en la localización de una tercera parte. Esta tercera parte puede ser parte de otra red o un servidor aislado. Las principales funciones del servidor de aplicaciones son [1]:

- Procesamiento de una sesión SIP procedente del IMS.
- Capacidad de generar nuevas peticiones SIP.
- Mandar información de tarificación al CCF (*Charging Collection Funtion*) y al OCS (*On-line Charging System*).

Los servicios ofrecidos no se limitan únicamente a servicios basados en el protocolo SIP. Cabe la posibilidad de hacerlo siguiendo la *Customized Applications for Mobile network Enhaced Logic* (CAMEL) y de la *Open Service Architecture* (OSA).

El uso de la arquitectura OSA permite al operador hacer uso de los servicios de control de llamadas, interacción con el usuario, estado del usuario, control de la sesión de datos, conocer las capacidades de los terminales, gestión de la tarificación, etc... Además puede ser utilizado como elemento para proveer acceso al IMS a terceras partes dado que implementa mecanismos de autenticación, autorización, registro y descubrimiento para las terceras partes dado que el S-CSCF no proporciona autenticación y funciones de seguridad a terceras partes.

Mas detalles de los elementos de la arquitectura del IMS se pueden encontrar en el Anexo C: Elementos del IMS.

2.4 SIP

El protocolo de inicio de sesión (SIP) es un protocolo de señalización, presencia y mensajería instantánea desarrollado para establecer, modificar y cancelar sesiones multimedia. SIP permite el establecimiento de sesiones multimedia entre dos usuarios finales. SIP ofrece los siguientes mecanismos:

- Localización de un usuario final.
- Contacto de usuario final para determinar su deseo de establecer una sesión.
- Intercambio de información sobre el medio para permitir el establecimiento de sesión.
- Modificación de las sesiones existentes.
- Cancelar/terminar sesiones multimedia existentes.

Además soporta mecanismos para pedir y enviar información de presencia (estado *on-line/off-line* del usuario o si el usuario esta suscrito a un determinado grupo). Estas funciones incluyen:

- Publicar y cargar información de presencia.
- Petición de solicitud de información de presencia.
- Presencia y otros eventos de notificación
- Transporte de mensajes instantáneos.

2.4.1 Agentes SIP

En SIP se definen distintos tipos de agentes

2.4.1.1 UA

Un UA es un dispositivo final que soporta la señalización SIP y capaz de establecer sesiones multimedia,

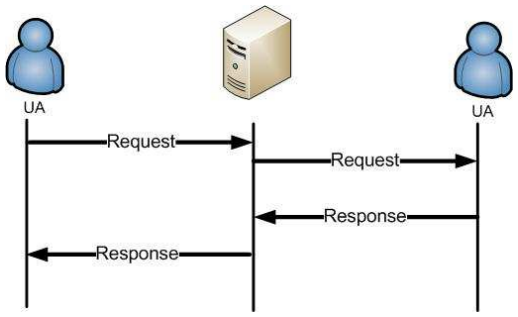

Figura 7 – *User Agent*

2.4.1.2 PA

Agente de presencia (PA). Es una entidad capaz de recibir peticiones de suscripción y generar notificaciones de estado definidas en las especificaciones de eventos SIP. Un agente de presencia responde a peticiones SUBSCRIBE y envía respuestas NOTIFY. Este agente puede recoger información de presencia procedente de diferentes fuentes tales como usuarios que se registran, dispositivos que publican información de presencia, etc…

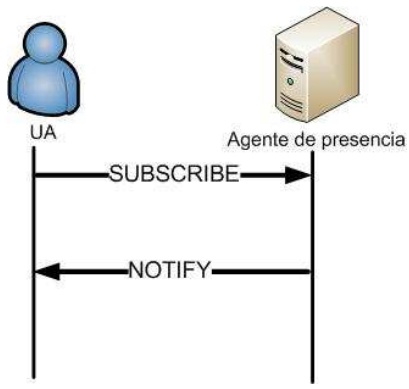

Figura 8 – *Presence Agent*

2.4.1.3 B2BUA

Back to Back User Agent. Se trata de una entidad que recibe una petición SIP, reformula la petición y la reenvía como una petición nueva. Las respuestas a la petición anterior son recibidas por el agente B2BUA que las reformula de nuevo y las envía a la entidad que origino la petición.

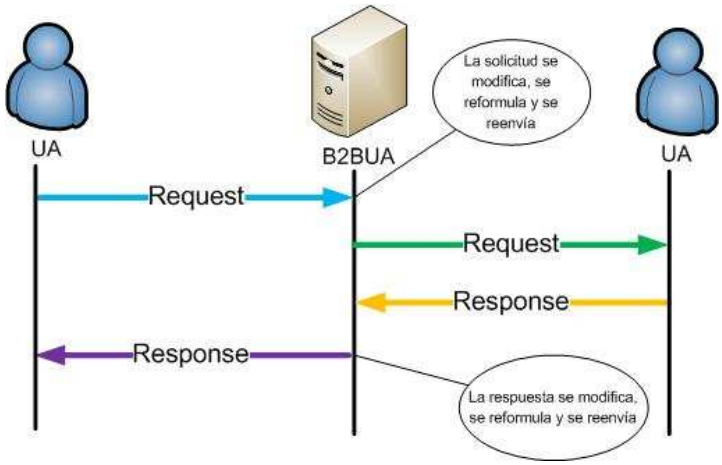

Figura 9 – *Back to Back User Agent*

2.4.1.4 SIP *Gateways*

Una pasarela SIP es una entidad que hace de interfaz entre SIP y otro protocolo. Una pasarela termina la via de la señalización SIP y en general también el *path* multimedia. Un ejemplo de SIP *Gateway* es aquel transforma la señalización SIP en señalización H.323. Sin embargo, en este caso el camino de los paquetes multimedia (que bien pudiera ser paquetes RTP) no termina en el SIP *Gateway* ya que estos paquetes pueden traspasar la pasarela SIP y ser enviados al dominio gobernado por la señalización H.323 que al fin y al cabo se trata de una red IP. No es el caso de las pasarelas SIP que actúan de interfaz con la PSTN (*Public Switched Telephone Network*). En este caso tanto el *path* de señalización como el de datos terminan en este punto ya que las dos redes presentan tecnología distinta: el flujo de datos va sobre una red de paquetes mientras que esos paquetes han de tratarse para ser insertados en una red de circuitos conmutados.

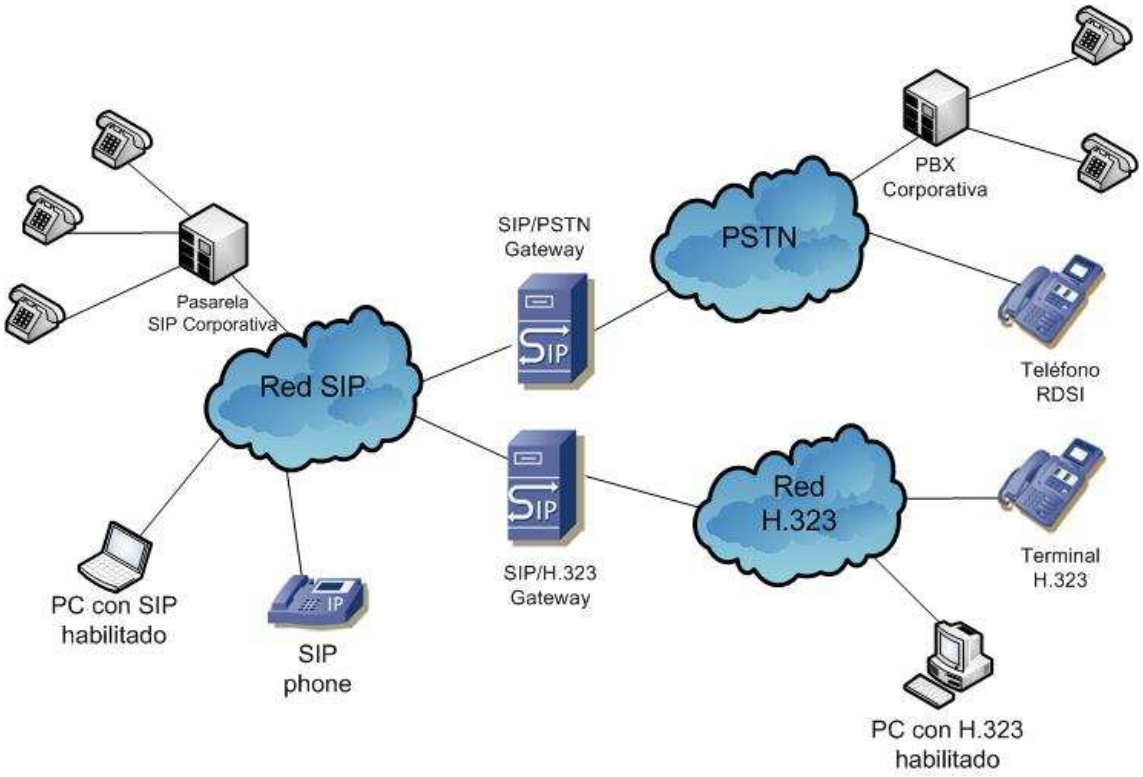

Figura 10 – *SIP Gateway*

2.4.1.5 SIP Servers

Son aplicaciones que aceptan peticiones SIP y dan respuestas a ellas. Un servidor SIP no debe de ser confundido con un UA o un cliente SIP ya que es una entidad distinta. El servidor SIP debe soportar TCP, UDP y TLS como protocolo de transporte. Podemos distinguir varios tipos de servidores SIP:

- Proxy SIP *Servers*. Estos servidores reciben peticiones SIP de un UA o de otro *proxy* y actúa de parte del UA reenviando o respondiendo a la petición. Un *proxy* no es una agente B2BUA ya que solo se le permite modificar peticiones o respuestas de acuerdo a las reglas de la RFC 3261 que define reglas preservan la transparencia *end-to-end* de la señalización SIP. Un servidor *proxy* normalmente tiene acceso a la base de datos o al servidor de localización para ayudarse en el procesamiento de la petición para determinar el siguiente salto. El interfaz entre el *proxy* y el servidor de localización no se define en el protocolo SIP. Las bases de datos contienen información sobre registros SIP, información de presencia y otro tipo de información sobre la localización del usuario. Un *proxy* no necesita entender ninguna petición SIP

para reenviarla, cualquier petición no conocida, el servidor la trata como si fuera una transacción no-INVITE. Un *proxy* no ha de cambiar el orden de las cabeceras, ni modificarlas ni eliminarlas.

Un servidor *proxy* se diferencia de un UA principalmente en:

- Un servidor SIP no genera peticiones SIP, solo responde a peticiones a un UA
- Un *proxy* no puede negociar las capacidades del medio.
- Un *proxy* SIP no puede parsear el contenido de los mensajes, tan solo los campos de las cabeceras.

Además los servidores *proxy* pueden dividirse en *stateless* (sin memoria) o *stateful* (con memoria). El servidor stateless es aquel que procesa cada petición o respuesta basado solamente en el contenido del mensaje. Una vez que el mensaje ha sido parseado, procesado y reenviado o respondido, el servidor no almacena ningún tipo de información sobre la petición o respuesta ni tampoco sobre el diálogo. Un *stateless proxy* nunca retransmite un mensaje y no utiliza ninguna clase de temporizador.

Un *stateful proxy* mantiene información sobre las peticiones y respuestas recibidas y utiliza esa información en el procesamiento de futuras peticiones y respuestas. De este modo un *stateful proxy* inicia un timer en el momento que recibe una petición y la reenvía. Si no recibe ningún tipo de respuesta a esa petición antes de que el temporizador expire, el *proxy* vuelve a retransmitir la petición. También un *proxy stateful* puede requerir la autenticación del usuario. Dentro de este tipo de servidores podemos distinguir los *transaction stateful proxy servers*, es decir, entidades que solo mantienen el estado de la transacción hasta que recibe una solicitud que espera. Una vez recibida la respuesta, el servidor destruye la información relativa al estado de la transacción y liberar información que puede estar utilizando recursos de manera inútil.

Un *proxy stateful* normalmente manda un mensaje de respuestas 100 TRYING cuando recibe una petición INVITE mientras que el servidor *stateless* nunca lo hace. Este mensaje nunca es reenviado a otras entidades ya que es un mensaje de un único salto. Un *proxy* que maneja peticiones TCP tiene que ser *stateful* dado que el UA asume que el transporte es fiable y retransmitirá el

paquete en caso de que no reciba respuesta. Sin embargo, el número de *proxies* que pueden reenviar el paquete viene limitado por el valor de la cabecera *Max-Forwards.* Cada vez que un *proxy* recibe una petición y la reenvía y decrementa el valor de este parámetro. Si en algún momento este contador llega a cero, el *proxy* que recibe este paquete lo descarta y no lo reenvía devolviendo un mensaje 483 TOO MANY HOOPS a la entidad que generó la petición.

Además existe un temporizador de sesión SIP que limita el periodo a partir del cual el servidor mantiene la información de estado. Esta información se encuentra dentro del mensaje INVITE en la cabecera *Session-Expires*. Una vez que expira este temporizador el *proxy* descarta la información de estado. Un UA puede cancelar la sesión si el temporizador expira y volver a enviar mensajes de INVITE.

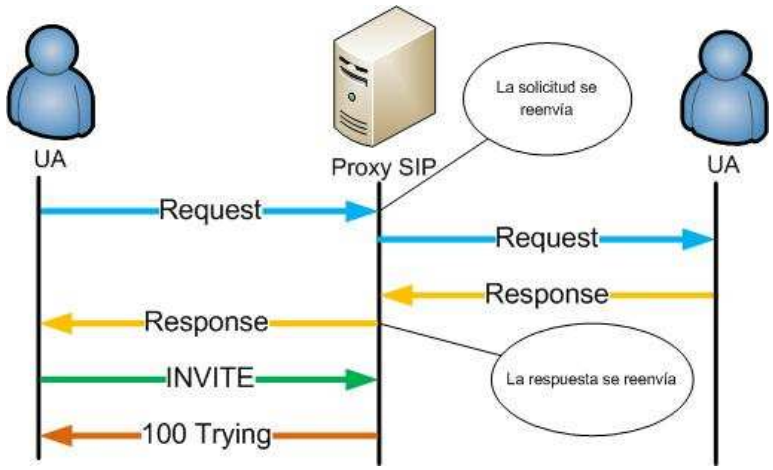

Figura 11 – *Proxy Stateful SIP*

2.4.1.6 Servidor de redirección

Se tratan de entidades que responde a peticiones pero no tienen capacidad de reenvío. Un servidor de redirección utiliza una base de datos o un servicio de localización para saber la ubicación del usuario. La información de localización del usuario es enviada a la entidad llamante mediante un mensaje con código 3xx y ésta responde con un ACK al servidor de redirección una vez sabido cual es la localización del usuario.

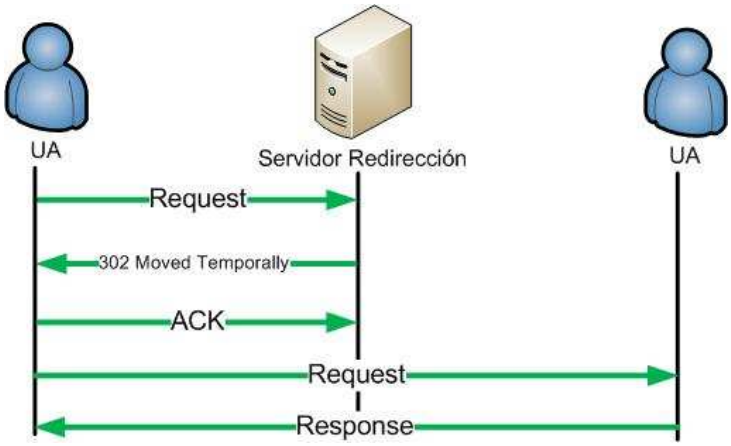

Figura 12 – Servidor de Redirección

2.4.1.7 Servidor de registro

Es un servidor que acepta mensajes SIP REGISTER. El resto de las peticiones son contestadas con un mensaje 501 NOT IMPLEMENTED. La información del campo *Contact* de la petición se hace disponible a otro servidor SIP tales como *proxies* o servidores de redirección en el mismo dominio administrativo. En una petición de registro, la cabecera *To* contiene el nombre del recurso que se desea registrar y la cabecera *Contact* que contiene direcciones alternativas o alias. El servidor de registro crea temporalmente una asociación entre el *Address of Record* (AOR) URI en la cabecera *To* y la URI en la cabecera *Contact*. Los servidores de registro normalmente requieren que el UA se registre para que sea autenticado.

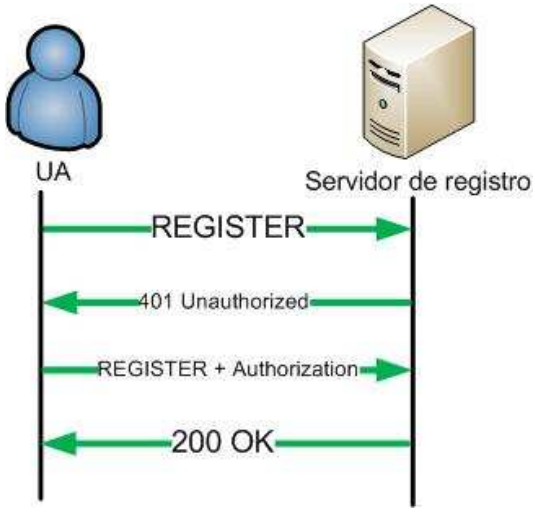

Figura 13 – Servidor de Registro

La mayoría de las solicitudes SIP son mensajes *end to end* entre los *User Agents* por lo que los *proxies* que hay entre los *User Agents* simplemente reenvían las peticiones o respuestas al *User Agent* o a otro *proxy* de manera que delegan la generación de mensajes de asentimientos a los *User Agent*.

SIP define mecanismos de fiabilidad que permiten el uso de protocolos de transporte no fiable. Se puede encontrar mas información acerca la fiabilidad de SIP en el Anexo D y en la referencia [9].

2.4.2 Solicitudes SIP

Se tratan de un tipo de mensajes en los cuales se requiere que el receptor del mensaje lleve a cabo una determinada acción. Originalmente había un total de 6 SIP *Request*: INVITE, REGISTER, BYE, ACK, CANCEL y OPTIONS. Más tarde aparecieron las extensiones de SIP que se encuentran definidas en otras RFCs y que introducen 7 nuevas peticiones REFER, SUBSCRIBE, NOTIFY, MESSAGE, UPDATE, INFO y PRACK.

2.4.2.1 INVITE

Se utiliza para establecer sesiones multimedia entre dos o más agentes usuarios (UAC). Un INVITE normalmente posee un cuerpo dentro del mensaje que contiene la información multimedia del usuario que origina la llamada. Además este cuerpo también puede contener información de calidad de servicio (QoS) o información de seguridad. Una sesión se considera establecida cuando el UAC recibe un mensaje de aceptación 200 OK por parte del servidor. En este momento se establece un diálogo entre los UAC que han acordado la sesión cuya duración expira en el momento que cualquiera de ellos decide abandonar la sesión enviando un BYE.

2.4.2.2 BYE

Este mensaje se utiliza para terminar una sesión previamente establecida. Para liberar la sesión el UA ha de introducir en la primera línea del BYE la URI del usuario contra el que quiere terminar la sesión que ha de ser la que le pasa en la cabecera *Contact.* El formato de esta primera línea se define en la siguiente tabla.

	Formato cabecera	Ejemplo
SIP URI	sip:[usuario]@dominio[:puerto(opcional)][SIP version]	sip:usuario1@192.168.1.3 SIP/2.0

Tabla 1 – Formato de la solicitud BYE

2.4.2.3 ACK

Se trata de un mensaje que se envía para asentir respuestas finales a una solicitud INVITE. Respuestas finales a otro tipo de respuestas finales nunca son asentidas. Las respuestas finales quedan definidas por los códigos 2xx, 3xx, 4xx, 5xx y 6xx. El valor de *CSeq* nunca se incrementa cuando se envía un ACK pero el método en la cabecera *CSeq* si es cambiado a ACK.

2.4.2.4 NOTIFY

La solicitud NOTIFY es utilizada por un UA para transmitir información sobre la ocurrencia de un cierto evento. El NOTIFY siempre se envía dentro de un diálogo cuando existe una suscripción entre el usuario y el que notifica. Dado que es un mensaje que se manda dentro de un diálogo estarán presentes las cabeceras *To, From,* y *Call-ID.*

La respuesta al NOTIFY es un 200 OK indicando que el mensaje ha sido recibido. Si se recibe un mensaje con el código *481 Dialog/Transaction Does Not Exist* la suscripción se termina automáticamente y no se envían más NOTIFY.

Las peticiones NOTIFY contienen una cabecera *Event* indicando el tipo de evento notificado y una cabecera *Subscription-State* que denota el estado actual de la suscripción.

Los tipos de eventos posibles se lista en la siguiente tabla:

Conference	***Información de llamada: lista de participantes en la llamada, información política y de control***
Dialog **message-summary** **Presence** **Refer** **Reg**	***Información de diálogo e información de identificación*** ***Notificación de mensajes usado en con el indicador de mensaje en espera y el buzón de voz*** ***Información de presencia*** ***Información implícita creada por el REFER*** ***Estado de registro del usuario***
winfo [3, 7]	***Acumula información general de estado***

Tabla 2 – Tipos de eventos

Los posibles valores de la cabecera *Susbcription-State* se definen en la tabla adjunta [6]:

Active	***Estado activo. La suscripción ha sido aceptada y (en general) autorizada.***
Pending	***Estado pendiente. La suscripción ha sido recibida por el notificador pero todavía no ha sido aceptada.***
Terminated	***Estado terminado. La suscripción ha finalizazo.***

Tabla 3 – Valores de la cabecera *Subscription-State*

Además se le pueden añadir parámetros adicionales tales como *expires, reason o retry-after* a cada uno de los estados para añadir información adicional al estado.

Un NOTIFY se envía siempre al comienzo y al final de de la suscripción. Si el NOTIFY contiene un cambio respecto al estado del usuario, el cuerpo del mensaje NOTIFY deberá denotar este cambio utilizando un incremento de una unidad por cada NOTIFY enviado. De esta manera el receptor del NOTIFY puede identificar si algún NOTIFY se ha perdido al estar fuera de secuencia.

Además se ha de definir bajo que supuestos se ha de mandar el NOTIFY. El cuerpo del NOTIFY ha de reportar el estado del recurso que está siendo monitorizado. Cada paquete ha de definir que tipo o tipos de eventos pueden notificar. La semántica de estos paquetes también tiene que estar claramente definida.

2.4.2.5 UPDATE

El mensaje UPDATE es un método que se utiliza para cambiar el estado de la sesión sin modificar el estado del diálogo. Posibles utilidades del UPDATE son silenciar la llamada, poner la llamada en espera y dejar de enviar el flujo de datos o realizar

negociaciones de parámetros de calidad de servicio extremo a extremo antes de establecerse la sesión.

2.4.2.6 PRACK

El mensaje PRACK se utiliza para asentir la recepción de las respuestas fiables provisionales (con código 1xx). La fiabilidad de las respuestas con código 2xx, 3xx, 4xx, 5xx o 6xx se obtiene asintiendo al mandar un ACK.

En caso de tener respuestas provisionales tales como 180 Ringing, es crítico determinar el estado de la llamada y confirmar la recepción de la respuesta provisional. Por tanto, el PRACK puede ser aplicado a todas las respuestas provisionales 1xx salvo para el 100 Trying.

El PRACK lo genera el UAC cuando se recibe una respuesta provisional que contiene las cabeceras *RSeq* (*Reliable Sequence Number*) y la cabecera *Supported:100rel.* Entonces el UAC manda un PRACK asintiendo la respuesta provisional y añade una cabecera *RAck* con el número de secuencia que recibió en la respuesta temporal y el método que la generó. Además incrementa el número de secuencia *CSeq.*

Una vez que el PRACK llega al destinatario, este responde con un 200 OK para informar de que el mensaje ha llegado correctamente.

La combinación del *Call-ID, CSeq* y *RAck* permite al UAC poder distinguir quien envía el 200 OK como respuesta al PRACK.

Dentro del cuerpo de PRACK se puede añadir información relativa a la negociación de parámetros de calidad de servicio.

En el anexo E se pueden encontrar mas detalles acerca de la solicitudes SIP.

2.4.3 Respuestas SIP

Una respuesta SIP es un mensaje originado por el UAS o *proxy* que contesta a una solicitud realizada por un UAC. Esta respuesta puede contener cabeceras adicionales que pueden proporcionar información extra o pueden ser simples asentimientos a solicitudes anteriores.

Existen 5 clases de respuestas SIP. Las primeras 5 clases fueron copiadas del protocolo HTTP mientras que la sexta es propia de SIP. Las clases de respuestas se listan en la siguiente tabla:

Clase	Descripción	Acción
1xx	Información	Indica el estado de la llamada antes de que se complete. Es la primera información que se recibe y es provisional.
2xx	Éxito	La solicitud se ha completado con éxito
3xx	Redirección	El servidor ha cambiado de localización. El cliente ha de dirigir la solicitud a otro servidor.
4xx	Error en el cliente	La solicitud ha fracasado por fallo en el cliente. El cliente puede intentar hacer otra solicitud.
5xx	Error en el servidor	La solicitud ha fracasado por fallo en el servidor. La solicitud puede ser redirigida a otro servidor
6xx	Fallo global	La solicitud no debe volverse a mandar a ningún servidor.

Tabla 4 – Respuestas SIP

Cada código de cada respuesta SIP no es entendida por el UAC, es decir, el UAC no entiende el código en particular, tan solo la clase.

Analizaremos algunas de las respuestas SIP ya que estarán presentes en el desarrollo de este proyecto.

2.4.3.1 100 TRYING

Se trata de una respuesta provisional que se propaga solo de un salto a otro y nunca se reenvía. Además nunca contiene ninguna información en el cuerpo del mensaje. Esta respuesta puede ser generada tanto por un *proxy Server* (cuando por ejemplo recibe un INVITE) como por un UA.

2.4.3.2 180 RINGING

Esta respuesta se manda para indicar que el INVITE ha sido recibido por el UA y alertar al iniciador de la llamada. Cuando el UA acepta la llamada, se deja de enviar el RINGING y se manda un 200 OK

El cuerpo del mensaje puede llevar información de calidad de servicio, información de seguridad, puede llevar adjunto un tono de llamada o cualquier tipo de animación que mande el UAS al UAC.

2.4.3.3 183 SESSION IN PROGRESS

Esta respuesta proporciona información sobre el progreso de la llamada. Al contrario que el 100 TRYING, el 183 SESSION IN PROGRESS es un mensaje extremo a extremo dentro de un diálogo por lo que las cabeceras *To* y *Contact* han de estar presentes en el mensaje.

Este mensaje se puede utilizar para permitir al UAC escuchar el tono de llamada, tono ocupado, o escuchar cualquier tipo de anuncio o mensaje de la PSTN.

También este mensaje se puede utilizar para enviar repuestas en modo fiable. Cuando el UA inicia la llamada enviando el INVITE, el destino de la llamada puede contestarle con un 183 SESSION IN PROGRESS para indicarle sobre el progreso de la llamada. Esta respuesta actualizará el valor de la cabecera *RSeq* incrementando el valor de este campo si fuera necesario. Además añadirá la cabecera *Require* que incluirá el parrámetro *100rel* para denotar que la respuesta es fiable. El destinatario de la llamada no enviará más repuestas provisionales en este diálogo hasta que no reciba el PRACK de la parte que inició la llamada.

El 183 SESSION IN PROGRESS puede contener un cuerpo SDP en el mensaje con parámetros de calidad de servicio que se negocian durante el establecimiento de la llamada.

2.4.3.4 200 OK

El uso de este mensaje tiene dos acepciones. Cuando se envía para aceptar una sesión (aceptar el INVITE) lleva un cuerpo en el mensaje con información sobre las capacidades de la parte llamada. Cuando se utiliza para dar respuesta a otras solicitudes indica que la solicitud se recibió y ha sido completada.

Información más detallada acerca del resto de solicitudes y respuestas pueden encontrarse en [5, 6, 9].

2.5 SDP

El protocolo SDP es un protocolo que surge para la negociación y descripción de sesiones multimedia. Mas que un protocolo es un lenguaje con una sintaxis claramente definida que permite la negociación de los parámetros de calidad de

servicio para el establecimiento de una sesión multimedia así como el tipo de flujo que se va a transmitir (audio, video, video-conferencia,…).

La carga SDP se encapsula dentro del cuerpo del mensaje SIP. Esta carga está compuesta por una serie de líneas llamadas campos cuyos nombres dentro del mensaje están abreviados por una letra minúscula indicando el tipo de campo que representan y su valor.

El contenido de la información dentro del cuerpo SDP se puede estructurar de la siguiente manera:

- Descripción de la sesión: contiene información referente a la sesión y al creador de la misma (información de la misma, dirección IP, …)
- Descripción de tiempos: contiene los tiempos de inicio y finalización y las repeticiones.
- Descripción multimedia: contiene información referente al protocolo de transporte o los formatos multimedia soportados.

La estructura de cualquier campo SDP es la siguiente:

x=parámetro1 parametro2 ... parámetrorN

A continuación se listan todos los campos existentes en SDP:

Campo	Nombre
v	*Version*: versión
o	*Origin*: creador/identificador de la sesión
s	*Session Name*: nombre de la sesión
i	*Information*: información de la sesión
u	*URI*: identificador uniforme del recurso
e	*Email*: dirección de correo electrónico
p	*Phone*: número de teléfono
c	*Connection*: información de conexión
b	*Bandwidth*: información de ancho de banda
t	*Time*: información de comienzo y fin de sesión
r	*Repeat times*: número de repeticiones
z	*Time zones*: ajustes de las zonas horarias
k	*Encryption key*: clave encriptada
a	*Attributes*: línea con atributos
m	*Media*: información multimedia

Tabla 5 – Campos SDP

La especificación completa del protocolo SDP se puede encontrar en [18]. También se adjunta el Anexo E en el que se pueden encontrar una breve descripción de cada uno de los campos.

Prestaremos especial atención al campo *Attributes* ya que juega un papel esencial en la negociación de la calidad de servicio y de los requisitos.

Los atributos sirven para extender tanto la descripción de sesión como las descripciones multimedia dentro de la carga SDP. Los atributos pueden ser definidos para ser utilizados tanto a nivel de sesión como a nivel multimedia o ambos.

Por tanto, se pueden encontrar los atributos en dos formatos: los de propiedad que se muestran de la forma a=<atributo> para indicar si se soporta una determinada propiedad, o los de valor, que extienden a los anteriores de la forma a=<atributo>:<valor>. Estos últimos no sólo especifican que se encuentra una determinada propiedad sino que además se incluye el valor correspondiente.

Dentro de los del primer grupo podemos encontrar los que definen la dirección del flujo multimedia que se pretende establecer en una descripción multimedia y que son cuatro:

- a=sendonly: sólo se desea enviar flujo multimedia
- a=recvonly: solo se desea recibir flujo multimedia
- a=sendrecv: se desea enviar y recibir flujo multimedia
- a=inactive: no se desea ni enviar ni recibir flujo multimedia

Para controlar la calidad de servicio tanto en el lado local como en el otro extremo se utilizan los atributos con las precondiciones [8]. En ellos se indican el estado actual de la calidad de servicio en ambos extremos y la que se desea para que se establezca la sesión multimedia. A continuación se muestra un ejemplo con posibles combinaciones de precondiciones tanto en el extremo local como en remoto:

```
a=curr:qos local none
a=curr:qos remote none
```

Estas dos líneas indican que actualmente (curr) no hay calidad de servicio garantizada (none) tanto en local como en remoto

```
a=des:qos mandatory local send
a=des:qos none remote sendrecv
```

Este par de líneas manifiestan que se desea (des) calidad de servicio obligatoria para mandar en local (qos mandatory local send) y todavía sin especificar en remoto (qos none remote sendrecv).

```
a=conf:qos remote sendrecv
```

Esta última línea expresa que el extremo llamado (remote) deberá mandar un mensaje de confirmación (conf) en el momento que los recursos (qos) hayan sido reservados en ambas direcciones de envío y recepción (sendrecv)

Dentro de los atributos de valor también tenemos aquellos que mapean un determinado número con un formato de codificación (a=rtpmap:). Este mapeo esta definido en [10]. Por ejemplo tenemos:

```
a=rtpmap:0 PCMU
```

En este caso se produce un mapeo entre el payload 0 con la codificación PCMU.

De igual importancia tenemos la línea *media.* Esta línea contiene información sobre el flujo multimedia que se va a transmitir. La línea tiene el siguiente formato:

m=<media> <puerto> <transporte> <lista_de_formatos>

- El parámetro media puede tener varios valores dependiendo del flujo a transmitir: "audio","video", "text", "application" y "message".
- El parámetro puerto indica el puerto por el que se envía el flujo multimedia.
- El campo transporte designa el protocolo de transporte sobre el que se va a enviar el flujo multimedia. Puede tomar distintos valores: UDP cuando no especifica ningún protocolo en concreto, RTP/AVP cuando el protocolo es RTP para el envío

de audio y/o video o RTP/SAVP cuando el protocolo es RTP para el envío de audio y/o video en modo seguro.

- Finalmente tenemos el listado de formatos que muestran todos los formatos en que el flujo multimedia se puede codificar.
- Es una cadena de caracteres numéricos separados por un espacio en el que cada número se corresponde a un formato de codificación.

Una posible línea m podría ser la siguiente:

```
m=audio 3458 RTP/AVP 0 96 97 98
```

Esta línea especifica que se puede enviar información de audio al puerto 3458 y que los formatos soportados para codificar la voz son 0, 96, 97 y 98.

2.6 Multidifusión IP

El concepto de multidifusión o *multicast* se basa en el envío de un mismo paquete a un grupo definido de destinatarios.

En lo referido al protocolo IP, existe una solución que es capaz de implementar este comportamiento que se conoce como IP *multicast*.

Los equipos destinatarios se constituyen en un grupo *multicast* de destinatarios (*receivers*), con una dirección IP *multicast* asociada. El originante (*sender*) indicará como dirección destino del datagrama la del grupo *multicast*. Los routers de la red se encargarán de hacer llegar a cada miembro del grupo una copia del mismo de forma eficiente, sin consumir excesivos recursos de la red.

Para la identificación de grupos multidifusión IP se utilizan direcciones de clase D (224.0.0.0 a 239.255.255.255) [17]. Para cada dirección *multicast* puede existir cero o más destinatarios activos, que recibirán una copia del datagrama enviado a dicha dirección. El grupo *multicast* está formado por los miembros activos, y se mantiene de forma dinámica mediante conexiones y desconexiones de los mismos (orientado al receptor). Independientemente de su naturaleza dinámica, los grupos *multicast* pueden ser permanentes o transitorios.

- Grupo permanente: tiene asociada una dirección IP *multicast* fija, independientemente del número de miembros que tenga el grupo. Se asocian a aplicaciones normalizadas. Ejemplos:

  - 224.0.0.1 - Todos los sistemas de la subred
  - 224.0.0.2 - Todos los routers de la subred
  - 224.0.0.4 – Todos los routers DVMRP de la subred
  - 224.0.0.5 - Todos los routers OSPF del dominio

  Se puede hacer uso del servicio DNS para localizar la dirección asociada a un grupo *multicast* permanente (dominio *mcast.net*) y lo mismo para sus resoluciones inversas (*224.in-addr.arpa.*)

- Un grupo transitorio se crea dinámicamente (en el momento que se lanza una aplicación multidifusión), y dejará de existir cuando deje de tener miembros activos

El grupo *multicast* se considera siempre de receptores, no de emisores. Esto implica que el originante de un datagrama *multicast* no tiene por qué ser miembro del grupo al que envía dicho datagrama.

2.6.1 Multicast en un segmento físico

La entrega física de un datagrama IP *multicast* a los miembros de la subred está condicionada por la capacidad multidifusión de la tecnología de red utilizada:

- Se cuenta con capacidad del multidifusión ➔ se mapea sobre ella el envío *multicast*. El problema se centrará en la traslación de direcciones *multicast* IP en direcciones *multicast* propias de la tecnología de red subyacente.

- No se cuenta con capacidad de multidifusión ➔ se implementan soluciones *multicast* IP mediante simulación por múltiples entregas unienvío o por difusión.

Redes Ethernet

Ethernet cuenta con un servicio de envío *multicast* que permite alcanzar todas las máquinas que pertenecen a un grupo dentro del segmento.

El envío de datagrama IP *multicast* sobre Ethernet requiere la conversión de la dirección IP a una dirección MAC *multicast*, su encapsulamiento en una trama y el envío de la misma hacia el segmento. Todos los equipos pertenecientes al grupo *multicast* IP tendrán que configurar en su interfaz Ethernet la atención a esa dirección MAC *multicast* en la que se ha mapeado la dirección IP.

Las direcciones MAC reservadas para *multicast* están comprendidas en el rango 01-00-5E-00-00-00 a 01-00-5E-7F-FF-FF-FF (23 bits menos significativos de las direcciones MAC).

La asociación entre ambas direcciones se realiza mediante un mapeo estático, solapando los 23 bits menos significativos de la dirección IP (de 32 bits) en la parte más baja de la dirección MAC 01-00-5E-00-00-00. Como la dirección IP *multicast* tiene 28 bits variables los 5 bits más significativos de la dirección IP no quedan reflejados en la dirección MAC

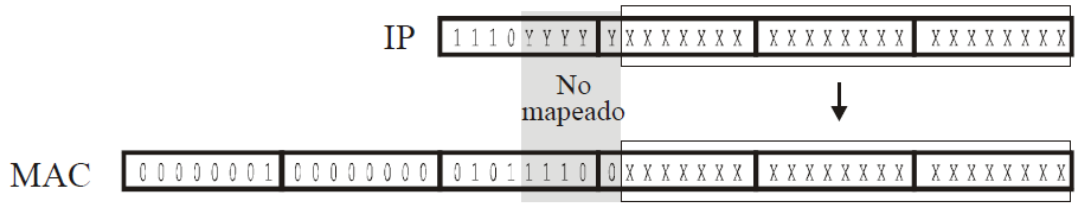

Figura 14 – Mapeo *multicast* IP-MAC Ethernet

Hasta 32 grupos multidifusión IP pueden ser mapeados en la misma dirección *multicast* MAC. Por ello es necesario filtrar a nivel IP los datagramas *multicast* que entrega el interfaz Ethernet, por si no corresponden con los grupos *multicast* IP a los que está asociado el interfaz IP.

2.6.2 Multicast entre segmentos

Será necesario contar con routers multidifusión (*mrouters*) que puedan procesar estos datagramas de modo que una copia de los mismos alcance cada uno de los equipos miembro del grupo multidifusión de forma eficiente a través de rutas sin bucles. Esto se consigue mediante un árbol de entrega *multicast* (*multicast delivery tree)* que se construye a través de los routers y que tiene por ramas todos los equipos que forman parte del grupo *multicast.*

Este árbol de entrega es dinámico, en función de la conexión/desconexión de los miembros del grupo.

Por tanto, para la construcción de este árbol se necesita:

- Un protocolo para determinar la pertenencia a un grupo de multidifusión (IGMP).
- Algoritmos y protocolos de encaminamiento multidifusión para construir y mantener los árboles de entrega (existen dos modelos de algoritmo y múltiples protocolos que los implementan: DVMRP, MOSPF, PIM-DM, PIM-SM,…). Se encargan de mantener las ramas de este árbol y podar las que ya no conducen a miembros del grupo.
- Un mecanismo para establecer el ámbito de la multidifusión, determinando el alcance de la misma.

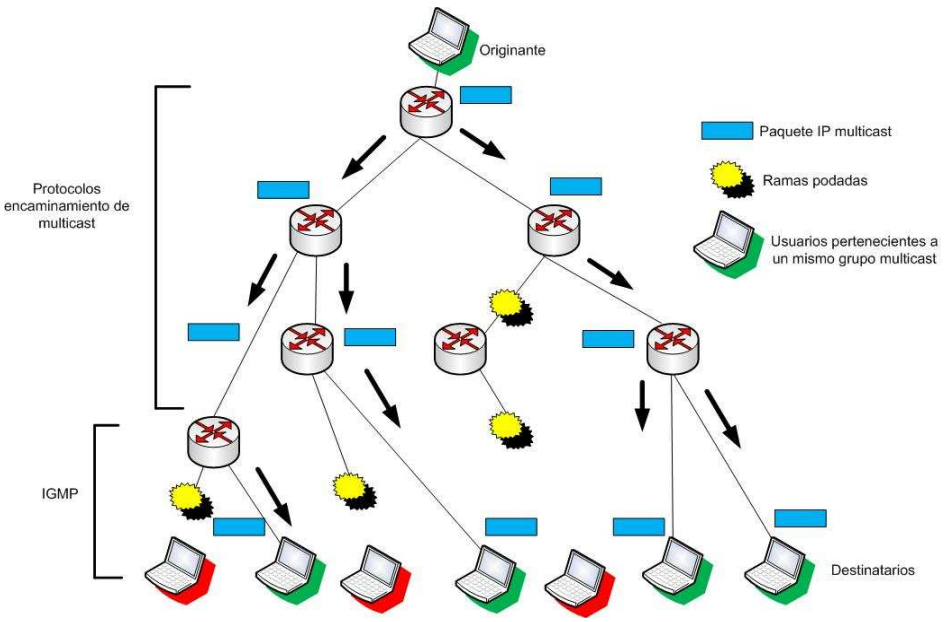

Figura 15 – Árbol de entrega *multicast*

Para el ámbito de multidifusión, puede establecerse de dos modos diferentes

- Mediante el campo TTL del datagrama. Dependiendo de su valor, los routers limitaran el reenvío del mismo:

TTL	Ámbito
0	Restringido al propio equipo. No sale a la subred.
1	Restringido a la misma subred. No será encaminado por los *mrouters*.
32	Restringido a la propia corporación.
64	Restringido a la propia región.
128	Restringido al mismo continente.
255	Ámbito global. Sin restricciones.

Tabla 6 – Valores del TTL para el ámbito de difusión

- Mediante la dirección IP de multidifusión empleada (por ejemplo se puede configurar el rango 224.0.0.0 – 224.0.0.255 para que esté reservado para aplicaciones multidifusión de un solo salto, y los datagramas no serán reenviados por los *mrouters*.)

2.6.3 IGMP (*Internet Group Management Protocol*)

Previo al envío de los paquetes a la dirección IP *multicast* es necesario que los participantes de la sesión se hayan suscrito a ese grupo.

Para llevar a cabo esta subscripción los participantes hacen uso del protocolo de administración del grupo Internet (IGMP) que está recogido en las referencias [14, 15,16]. Establece un mecanismo para el intercambio y actualización de información sobre la pertenencia de los equipos de un segmento a un grupo *multicast*. El diálogo se desarrolla entre los equipos miembros y los *mrouters* de su segmento y posibilita la conexión y desconexión de los equipos a los grupos *multicast*. Los equipos informan acerca de su pertenencia a un grupo, y los routers de multidifusión sondean periódicamente el estado de dichas pertenencias.

Mensajes IGMP

Se transportan encapsulados dentro de los datagramas IP (*protocol* = 2). Su estructura es la siguiente:

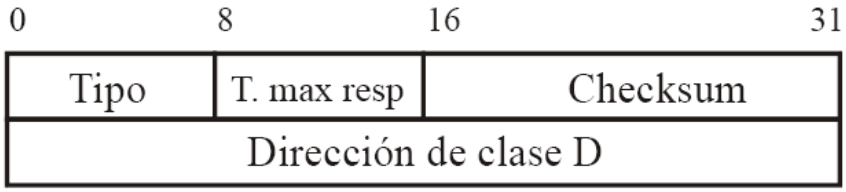

Figura 16 – Formato mensaje IGMP

El campo Tipo establece los distintos tipos de mensaje reconocidos en IGMP:

Tipo	Función
0x16: Informe de pertenencia de equipo (*membership report packet*)	Enviado por un equipo que se une al grupo multidifusión. También reenvía como respuesta a un mensaje de consulta del router *multicast*.
0x11: Consulta de pertenencia de equipos (*membership query packet*)	Enviado periódicamente por los routers para sondear la subred en busca de los miembros del grupo.
0x17: Dejar grupo (*leave Group packet*)	Enviado por un equipo cuando abandona el grupo multidifusión.

Tabla 7 – Tipos de mensajes IGMP

El tiempo máximo de respuesta indica (en décimas de segundo) el tiempo máximo que el router esperará un mensaje IGMP de tipo Informe de pertenencia por parte de los equipos. El campo *checksum* protege el mensaje y el campo Dirección contiene la dirección *multicast* del grupo afectado.

Los *mrouters* mantienen una base de datos de grupos *multicast* para cada uno de sus interfaces, donde se recogen las direcciones multidifusión activas (con una entrada del tipo [grupo *multicast, interface]*). Cuando un equipo desea conectar a un grupo envía un Informe de Pertenencia con la dirección *multicast* del mismo. El router que detecta ese mensaje comprueba si ya existe dicho grupo asociado al interfaz, y si no es así registra la nueva pertenencia. Periódicamente el router chequeará la presencia de miembros de los grupos registrados en cada interfaz (Consulta de Pertenencia) y si deja de recibir respuestas eliminará la anotación asociada.

En la versión 3 de IGMP se permite establecer filtros a los envíos: los equipos pueden limitar desde qué direcciones IP origen aceptarán datagramas multidifusión para cada grupo al que pertenecen.

2.6.4 Algoritmos y protocolos de encaminamiento multidifusión

Algoritmos

Los algoritmos multidifusión permiten mantener los árboles de entrega *multicast* entre los routers para alcanzar a todos los miembros de un grupo. Deben verificar los siguientes requisitos:

- Encaminar los datagramas sólo a los miembros del grupo multidifusión
- Optimizar las rutas desde el origen hasta cada destino, y evitar bucles
- Distribuir la carga por todas las ramas
- Mantener el árbol mediante un modelo de señalización eficiente y escalable

Dos son los algoritmos utilizados por los protocolos multidifusión: Propagación por la Trayectoria Inversa (RPF, *Reverse Path Forwarding*) y Árbol Centralizado (CBT, *Center-Based Tree*)

Mas detalles sobre el funcionamiento de estos protocolos se recoge en el Anexo G.

2.7 Codecs de voz

La voz para ser transmitida ha de someterse a un proceso de digitalización para posteriormente ser paquetizada y ser enviada a través de la red.

El proceso de convertir ondas analógicas a información digital se hace con un codificador-decodificador (el CODEC). Hay muchas maneras de transformar una señal de voz analógica, todas ellas gobernadas por varios estándares. El proceso de la conversión es complejo. Es suficiente decir que la mayoría de las conversiones se basan en la modulación codificada mediante pulsos (PCM) o variaciones.

Además de la ejecución de la conversión de analógico a digital, el CODEC comprime la secuencia de datos, y proporciona la cancelación del eco. La compresión de la forma de onda representada puede permitir el ahorro del ancho de banda. Esto es

especialmente interesante en los enlaces de poca capacidad y permite tener un mayor número de conexiones de VoIP simultáneamente. Otra manera de ahorrar ancho de banda es el uso de la supresión del silencio, que es el proceso de no enviar los paquetes de la voz entre silencios en conversaciones humanas.

2.7.1 PCM

La codificación PCM se realiza mediante tres pasos:

- Muestreo.
- Cuantificación.
- Codificación

Muestreo

El proceso de muestreo consiste en tomar valores instantáneos de una señal analógica a intervalos de tiempos iguales. A los valores instantáneos se les llama muestras.

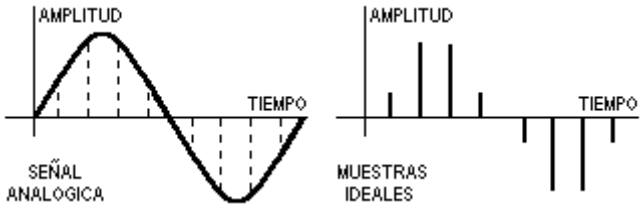

Figura 17 – Muestreo de una señal analógica

El muestreo se efectúa siempre a un ritmo uniforme, que viene dado por la frecuencia de muestreo fm o *sampling rate*.

La condición que debe cumplir fm viene dada por el teorema del muestreo "Si una señal contiene únicamente frecuencias inferiores a f, queda completamente determinada por muestras tomadas a una velocidad igual o superior a 2f."

De acuerdo con el teorema del muestreo, las señales telefónicas de frecuencia vocal (que ocupan la Banda de 300 a - 3.400 Hz), se han de muestrear a una frecuencia igual o superior a 6.800 Hz (2 x 3.400).

En la práctica, sin embargo, se suele tomar una frecuencia de muestreo o *sampling rate* de fm = 8000 Hz. Es decir, se toman 8.000 muestras por segundo que corresponde a una separación entre muestras de

T=1/8000= 0,000125 seg. = 125 µs

Por lo tanto, dos muestras consecutivas de una misma señal están separadas 125 µs que es el periodo de muestreo.

Cuantificación

La cuantificación es el proceso mediante el cual se asignan valores discretos, a las amplitudes de las muestras obtenidas en el proceso de muestreo. Existen varias formas de cuantificar que iremos detallando según su complejidad.

a) Cuantificación uniforme

Hay que utilizar un número finito de valores discretos para representar en forma aproximada la amplitud de las muestras. Para ello, toda la gama de amplitudes que pueden tomar las muestras se divide en intervalos iguales y a todas las muestras cuya amplitud cae dentro de un intervalo, se les da el mismo valor

El proceso de cuantificación introduce necesariamente un error, ya que se sustituye la amplitud real de la muestra, por un valor aproximado. A este error se le llama error de cuantificación.

El error de cuantificación se podría reducir aumentando el número de intervalos de cuantificación, pero existen limitaciones de tipo práctico que obligan a que el número de intervalos no sobrepase un determinado valor.

Una cuantificación de este tipo, en la que todos los intervalos tienen la misma amplitud, se llama cuantificación uniforme.

En siguiente figura se muestra el efecto de la cuantificación para el caso de una señal analógica. El número de intervalos de cuantificación se ha limitado a ocho.

La señal original es la de trazo continuo, las muestras reconstruidas en el terminal distante, se representan por puntos y la señal reconstruida es la línea de trazos.

El error de cuantificación introducido en cada muestra, da lugar a una deformación o distorsión de la señal reconstruida que se representa por línea de trazos y puntos.

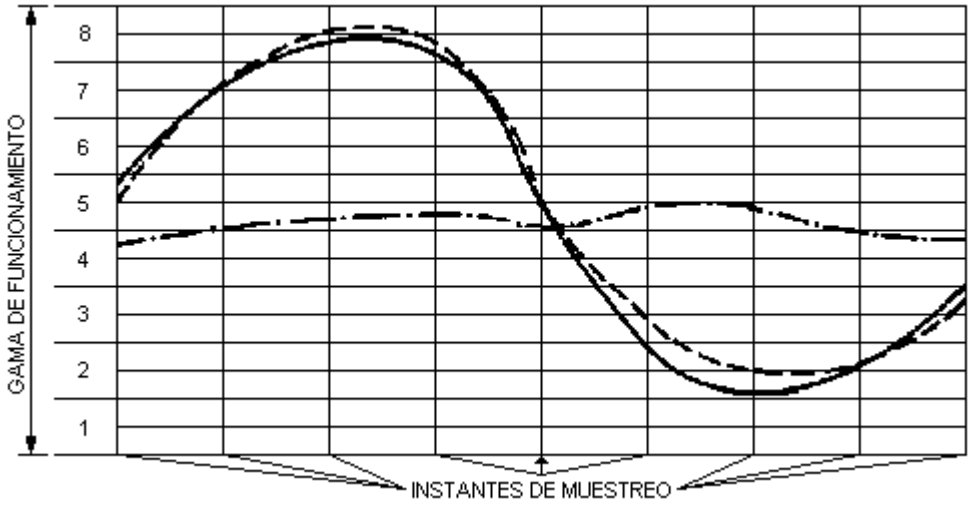

Figura 18 – Señal orginal vs. señal muestreada

b) Cuantificación no uniforme.

En una cuantificación uniforme la distorsión es la misma cualquiera que sea la amplitud de la muestra. Por lo tanto cuanto menor es la amplitud de la señal de entrada mayor es la influencia del error. La situación se haya ya inadmisible para señales cuya amplitud analógica está cerca de la de un intervalo de cuantificación

Para solucionar este problema existen dos soluciones:

- Aumentar los intervalos de cuantificación - si hay más intervalos habrá menos errores pero necesitaremos más números binarios para cuantificar una muestra y por tanto acabaremos necesitando mas ancho de banda para transmitirla.

- Mediante una cuantificación no uniforme, en la cual se toma un número determinado de intervalos y se distribuyen de forma no uniforme aproximándolos en los niveles bajos de señal, y separándolos en los niveles altos. De esta forma, para las señales débiles es como si se utilizase un número muy elevado de niveles de cuantificación, con lo que se produce una disminución de la distorsión. Sin embargo para las señales fuertes se tendrá una situación menos favorable que la correspondiente a una cuantificación uniforme, pero todavía suficientemente

buena.

Ley de codificación o compresión

El proceso de cuantificación no uniforme responde a una característica determinada llamada ley de codificación o de compresión

Hay dos tipos de leyes de codificación: las continuas y las de segmentos.

En las primeras, los intervalos de cuantificación son todos de amplitud distinta, creciendo ordenadamente desde valores muy pequeños, correspondientes a las señales de nivel bajo, a los valores grandes, correspondientes a las señales de nivel alto.

En las segundas, la gama de funcionamiento se divide en un número determinado de grupos y dentro de cada grupo los intervalos de cuantificación tienen la misma amplitud, siendo distinta de unos grupos a otros.

Normalmente se utilizan las leyes de codificación de segmentos.

G.711 Ley A (a-law) y ley μ (u-law)

Actualmente, las dos leyes de compresión de segmentos mas utilizadas son la ley A (a-law) y la ley μ (u-law) que dan lugar al codec g.711. La ley A (a-law) se utiliza principalmente en los sistemas PCM europeos, y la ley μ (u-law) se utiliza en los sistemas PCM americanos.

La ley A está formada por 13 segmentos de recta (en realidad son 16 segmentos, pero como los tres segmentos centrales están alineados, se reducen a 13). Cada uno de los 16 segmentos está dividido en 16 intervalos iguales entre si, pero distintos de unos segmentos a otros.

La formulación matemática de la Ley A es:

y= Ax / 1+ LA ---------- para 0 =< x =< 1/A
y= 1+ L (Ax) / 1+ LA ---------- para 1/A=< x =< 1

siendo L logaritmo neperiano.

El parámetro A toma el valor de 87,6 representando x e y las señales de entrada y salida al compresor

La ley μ se representa matemáticamente como:

y= L(1+μx) / L (1+μ)---------- para 0 =< x =< 1

donde μ= 255

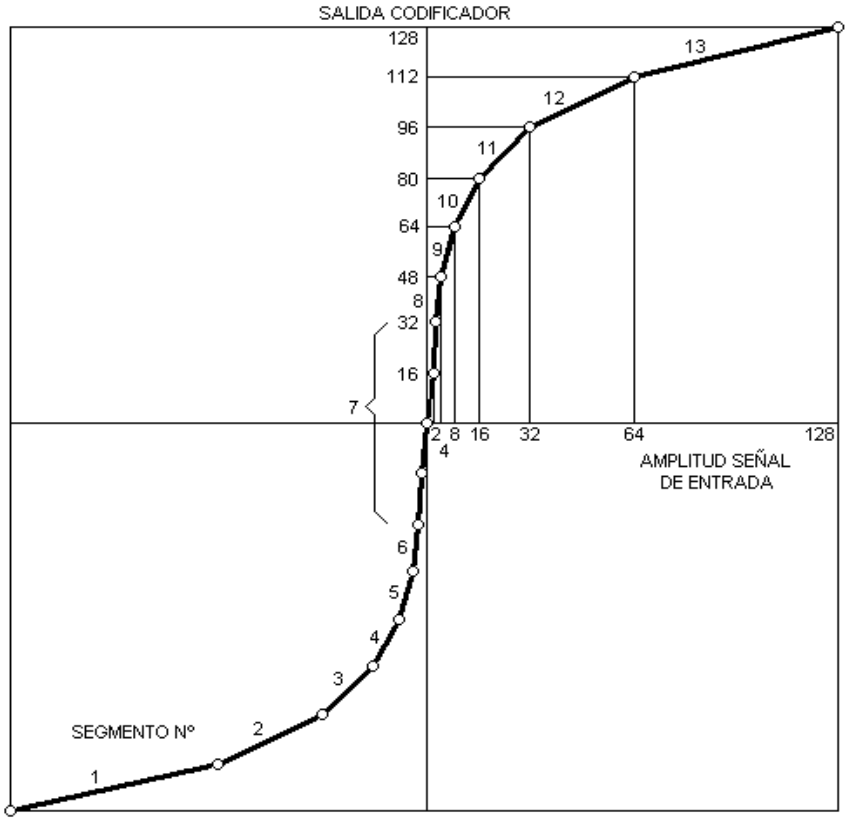

Figura 19 – Amplitud de la señal vs. salida del codificador

Cuantificación diferencial

En las señales de frecuencia vocal, predominan generalmente las bajas frecuencias, por ello las amplitudes de dos muestras consecutivas difieren generalmente en una cantidad muy pequeña. Aprovechando esta circunstancia, se ha ideado la cuantificación diferencial.

En la cuantificación diferencial, en lugar de tratar cada muestra separadamente, se cuantifica y codifica la diferencia entre una muestra y la que le precede. Como el número de intervalos de cuantificación necesarios para cuantificar la diferencia entre dos muestras consecutivas es lógicamente inferior al necesario para cuantificar una muestra aislada, la cuantificación diferencial permite una reducción sensible de la frecuencia de transmisión en línea, ya que esta es proporcional al número de intervalos de cuantificación

Cuantificación diferencial delta y ADPCM (*Adaptative delta PCM*)

Si en un sistema DPCM vamos aumentando la frecuencia de muestreo, llega un momento en que dos muestras consecutivas tienen una amplitud tan próxima, que no se necesita más que un solo intervalo de cuantificación para cuantificar la diferencia.

Este caso solo se necesitaría un bit por muestra, y la velocidad de transmisión en línea (*bit rate*) sería igual a la velocidad de muestreo. Este tipo de modulación se conoce con el nombre de la modulación delta.

La modulación delta descrita, se denomina modulación delta porque la magnitud de la variación producida a la salida es fija. Existen otros tipos de modulación delta mas sofisticados, en los cuales dicha variación no es fija sino que depende de las variaciones de la señal de entrada. Por ejemplo ADPCM o *Adaptative* delta PCM se basa en ajustar la escala de cuantificación de forma dinámica para adaptarse mejor a las diferencias pequeñas o grandes.

Codificación - Decodificación

La codificación es el proceso mediante el cual se representa una muestra cuantificada mediante un número binario.

Cada muestra cuantificada se representa o codifica mediante un número binario. Normalmente en telefonía se utilizan 256 intervalos de cuantificación para representar todas las posibles muestras (por ejemplo para G.711 tanto ley A como ley μ), por tanto se necesitarán números binarios de 8 bits para representar a todos los intervalos (pues $2^8 = 256$). Otros codecs que usan ADPCM o cuantificación delta utilizan menos intervalos y por tanto menos bits.

La decodificación es el proceso inverso, a partir de la secuencia de bits recibida en el decodificador se reconstruyen las muestras y a partir de esta la señal original.

Además de la ejecución de la conversión de analógico a digital, el CODEC comprime la secuencia de datos, y proporciona la cancelación del eco. La compresión de la forma de onda representada puede permitir el ahorro del ancho de banda**.** Esto es especialmente interesante en los enlaces de poca capacidad y permite tener un mayor número de conexiones de VoIP simultáneamente. Otra manera de ahorrar ancho de banda es el uso de la supresión del silencio, que es el proceso de no enviar los paquetes de la voz entre silencios en conversaciones humanas.

A continuación se muestra una tabla resumen con los códecs más utilizados actualmente:

- El *bit rate* indica la cantidad de información (en bits) que se manda por segundo [bits/seg]
- El *sampling rate* representa la frecuencia de muestreo de la señal de voz (cada cuanto se muestrea la señal la señal analógica de voz) [KHz]
- El *frame size* indica cada cuantos milisegundos se envía un paquete de voz [mseg]
- El MOS (*Mean Opinion Score*) indica la calidad general de CODEC (valor de 1 a 5)

CODEC	Descripción	Bit rate (Kb/s)	Sam Rate (KHz)	Frame Size(ms)	Observaciones	MOS
G.711	PCM	64	8	Muestreada	Tiene dos versiones u-law (US, Japan) y a-law (Europa) para muestrear la señal	4.1
G.721	ADPCM	32	8	Muestreada	Obsoleta. Se ha transformado en la G.726	
G.722	7KHz de audio	64	16	Muestreada	Divide los 16 KHz en dos bandas cada una usando ADPCM	
G.722.1	Codificación a 24 y 32 kbit/s para sistemas sin manos con baja perdida de paquetes	24/32	16	20		
G.723	Extensión de la norma G.721 a 24 y 40 kbit/s para aplicaciones en circuitos digitales.	24/40	8	Muestreada	Obsoleta por G.726. Es totalmente diferente de G.723.1.	
G.723.1		5.6/6.3	8	30	Parte de H.324 video conferencing. Codifica la señal usando linear predictive analysis-by-synthesis coding. Para el codificador de high rate utiliza Multipulse Maximum Likelihood Quantization (MP-MLQ) y para el de low-rate usa Algebraic-Code-Excited Linear-Prediction (ACELP).	3.8-3.9
G.726	40, 32, 24, 16 ADPCM	40, 32, 24, 16	8	Muestreada	ADPCM; reemplaza a G.721 y G.723.	3.85
G.728		16	8	2.5	Utiliza Code Excited Linear Prediction (CELP)	3.61
G.729 *		8	8	10	Bajo retardo (15 ms). Utiliza conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP)	3.92
GSM 06.10		13	8	22.5	Regular Pulse Excitation LongTerm Predictor (RPE-LTP). Usado para la telefonía celular	
LPC10		2.4	8	22.5	Linear-predictive codec	
iLBC		8	13.3	30		
EVRC	Enhanced Variable Rate CODEC	9.6/4.8/ 1.2	8	20	Se usa en redes CDMA	
DVI	ADPCM	32	Variable	Muestreada		
L16	Formato no comprimido	128	Variable	Muestreda		

Tabla 8 – Codecs de voz

G729: es el codec original

G729A o anexo A: es una simplificación de G729 y es compatible con G729. Es menos complejo pero tiene algo menos de calidad.

G729B o anexo B: Es G729 pero con supresión de silencios y no es compatible con las anteriores.

G729AB: Es g729A con supresión de silencios y sería compatible solo con G729B.

3. Identificación de los requisitos del sistema.

Como ya se ha comentado en este proyecto se pueden distinguir dos partes bien diferenciadas: por un lado tenemos el plano de control para el establecimiento de sesión de audio basada en el protocolo SIP/SDP y un plano de datos que comprende el procesamiento y el envío de la voz por multidifusión IP a un determinado grupo de usuarios.

3.1 Plano de control o señalización

En lo relativo a la señalización se han identificado varias tareas:

- Implementación de un cliente iniciador de la sesión o llamante
- Implementación de un cliente receptor o llamado.
- Integración de ambos con el servidor de aplicaciones multiusuario (MAS *Multiparty Application Server*)

Antes de proceder a la definición e identificación de los requisitos de los clientes hemos se entender el marco en el que se encuentran los mismos. La idea es que estos clientes interactúen con el MAS y por ello es conveniente que conozcamos su funcionamiento. La figura 21 ilustra el intercambio de mensajes entre los clientes y el MAS. Esencialmente el MAS es un servidor que actúa como B2BUA de SIP. Como ya se ha comentado un B2BUA está formado por un Agente de Usuario Servidor (*UAS, UserAgent Server*) y un Agente de Usuario Cliente (*UAC, User Agent Client*). Ambos UAs se encuentran conectados mediante cierta lógica específica del servicio proporcionado por el B2BUA. En la figura 20 se muestra la arquitectura funcional de un B2BUA.

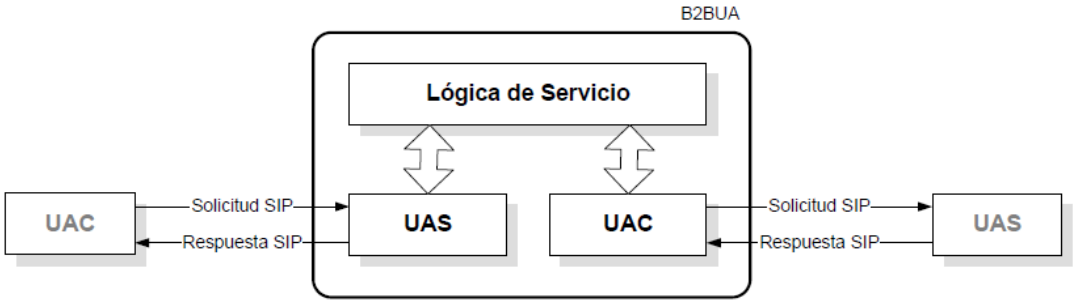

Figura 20 – Arquitectura funcional del MAS

El MAS solo participa en la fase de establecimiento y liberación de sesión. Una vez que la sesión esta establecida el MAS no interviene en el envío de los datos multimedia, en este caso la voz, entre los diferentes participantes en la sesión.

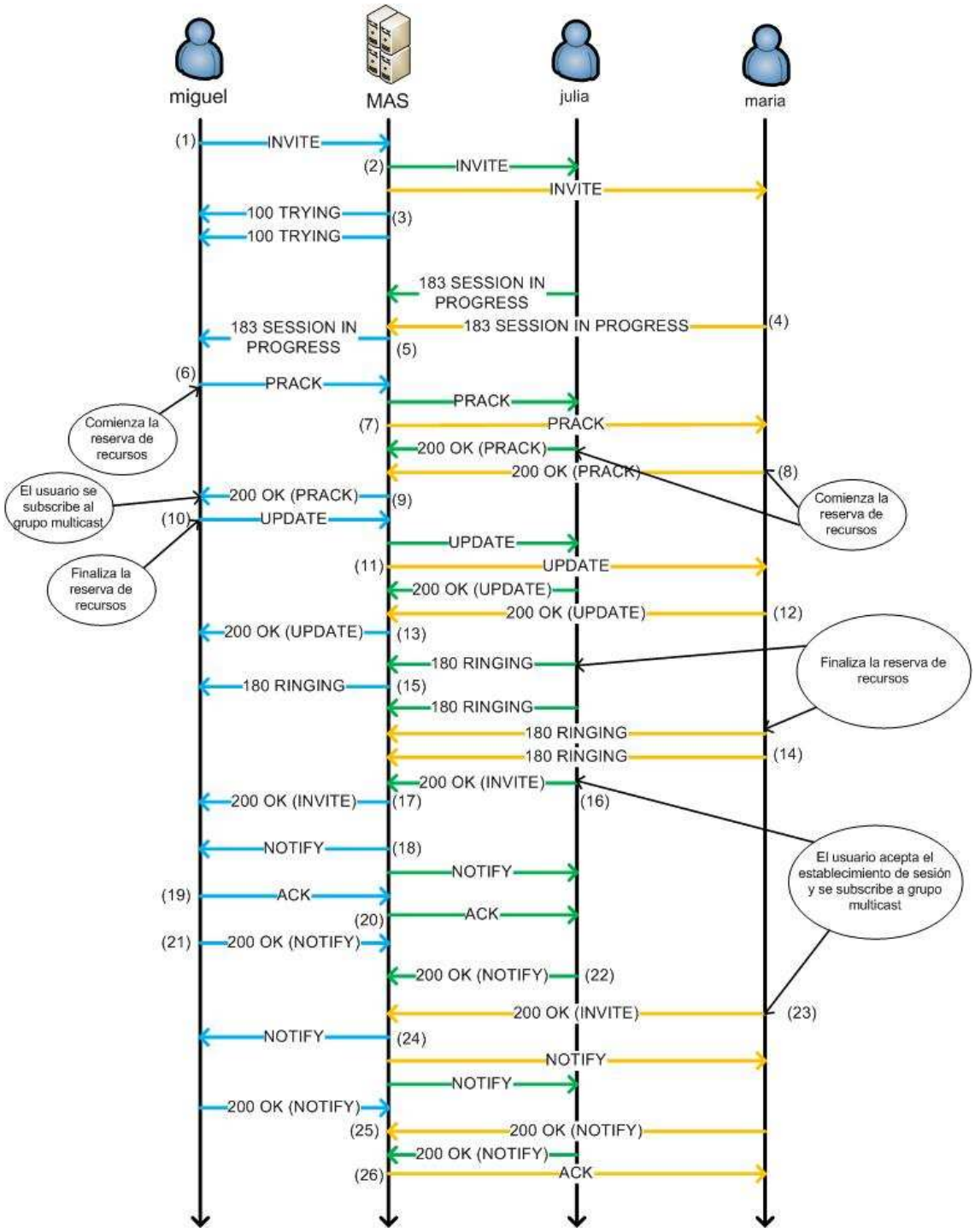

Figura 21 – Establecimiento de sesión multiusuario

Los clientes (llamante y llamado) serán los encargados de generar las solicitudes y respuetas SIP mostrados en el diagrama. La idea es que el llamante (miguel) inicie una sesión con un grupo de usuarios que se encuentran registrados en una dirección de grupo. El MAS básicamente se encarga de identificar la dirección grupo y reenviar las solicitudes a los destinatarios (julia y maria) en concreto. Los destinatarios

responden a estas solicitudes y el MAS se encarga de procesarlas y hacérselas llegar al iniciador de la llamada.

Del mismo modo también se propone una liberación de sesión en caso de que uno de los usuarios decida abandonar la sesión cuyo diagrama de mensajes se muestra en la figura 22.

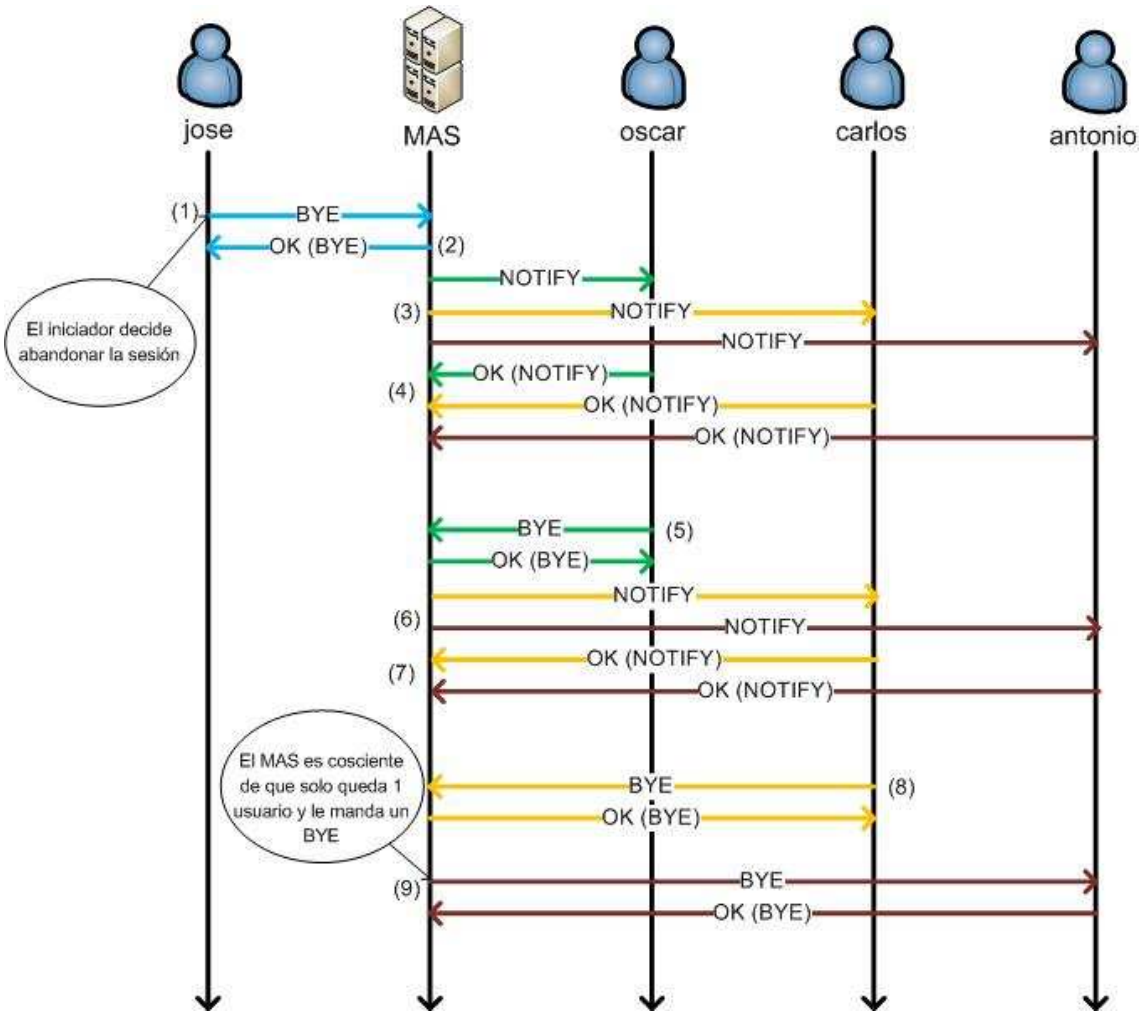

Figura 22 – Liberación de sesión multiusuario

Se puede encontrar un funcionamiento detallado del MAS en las referencias [11, 12, 13] así como en el Anexo H en el que se detalla el contenido de cada solicitud y respuesta SIP en los casos de establecimiento y liberación de sesión.

Una parte muy importante en el establecimiento de la sesión es la negociación de los parámetros de calidad de servicio y prerrequisitos que el iniciador y los llamados llevan a cabo. Esta negociación responde a un modelo ofrecimiento/respuesta (*SDP*

Offer/Answer). El iniciador de la llamada ofrece una serie de parámetros al destino y éste ha de responder al iniciador diciéndole aquello que soporta.

En total se han identificado un total de 3 *SDP Offer* que manda el iniciador y 3 *SDP Answer* con el que responde el destinatario.

3.1.1 Cliente llamante.

El cliente llamante o iniciador de la llamada es la entidad encargada de generar el inicio del establecimiento de la sesión. Se presentará al usuario un interfaz gráfico en el cual el usuario podrá especificar los parámetros de entrada a la aplicación. Este interfaz hará más amigable e interactivo la comunicación entre el usuario de la aplicación y la aplicación en si.

A continuación se definen los requisitos genéricos que tiene que cumplir la aplicación:

REQ.1a. El cliente ha de definir un nombre de usuario con el que se le identificará en la sesión. Este nombre puede ser cualquier cadena alfanumérica y puede contener cualquier tipo de carácter.

REQ.1b. El cliente tiene que especificar en qué puerto esta escuchando las solicitudes y respuestas SIP. Además ha de especificar en que dirección espera escuchar las repuestas a estas solicitudes. Esto se hace en el campo *Contact* del INVITE inicial. En este caso esta dirección será la dirección IP del adaptador que genera la petición, es decir, el cliente espera recibir las respuestas en la misma dirección que las genera.

REQ.1c. El cliente siempre dirige las solicitudes y respuestas SIP al puerto SIP por defecto que es el 5060.

REQ.1d. La solicitud inicial INVITE irá dirigida a una SIP URI que se identificará con una dirección de grupo. En esta dirección de grupo se encuentran suscritos los destinatarios finales a los que se dirige el establecimiento de sesión multiusuario.

REQ.1e. Se podrá especificar el camino que ha de seguir la petición inicial INVITE. Esta información se fijará en el campo *Route* de esta solicitud y definirá la lista de proxies SIP (en formato SIP URI) que la solicitud ha de atravesar.

REQ.1f. El cliente iniciador ha de especificar el puerto donde se desea establecer la sesión de voz multiusuario. Este puerto tiene que ser distinto del puerto donde el cliente escucha las solicitudes y respuestas SIP que quedó definido en el REQ.1b.

REQ.1g. El cliente iniciador ha de ser informado de cuando el primero de los usuarios ha aceptado el establecimiento de la sesión multiusuario. En este momento el cliente estará en disposición de iniciar la sesión de voz.

REQ.1h. Si todos los participantes que se encuentran activos en la llamada deciden finalizar la sesión de manera que quede solo el cliente iniciador activo y además ocurre que haya uno o mas participantes que todavía no hayan aceptado la llamada, el iniciador ha de ser notificado de esto y decidirá si espera a que al menos uno de ellos acepte la llamada o por el contrario no quiere esperar y decida abandonar la sesión.

REQ.1i. El cliente iniciador ha de especificar qué tipo de tráfico multimedia se va a cursar. En este caso será audio.

REQ.1j. Se han de especificar el listado de codecs de voz que soporta el usuario.

REQ.1k. El cliente ha de indicar cuales son los prerrequisitos tanto en el usuario local como en remoto.

REQ.1l. No hay una preferencia por un codec determinado. En el caso de que sean ofrecidos varios codecs, el iniciador seleccionará el primero que se encuentre en la lista. Una vez que se ha negociado el codec de voz que se utilizará en la sesión de voz multiusuario se ha de iniciar la reserva de recursos. La implementación de esta reserva no se ha tratado en el desarrollo de este proyecto y se ha dejado abierta**.**

REQ.1m. No se especificará la dirección en la que el cliente desea recibir los datos de la sesión multimedia ya que esa información la especificará el MAS.

REQ.1n. El cliente vuelca las solicitudes y las respuestas que envía a un fichero de *logs* donde queda constancia de todas las transacciones realizadas y que sirve como un histórico de mensajes enviados y recibidos.

3.1.2 Cliente llamado.

El cliente destinatario será la entidad receptora de la llamada. Al igual que ocurre con el iniciador, éste cliente dispone de una interfaz gráfica en el que el usuario podrá aceptar las llamadas.

Los requisitos genéricos que se definen en la aplicación del destinatario se definen a continuación:

REQ.2a. Ïdem que REQ.1a.

REQ.2b. Idem que REQ.1b. En este caso el *Contact* se especifica en el SESSION IN PROGRESS que el destinatario envía como respuesta al INVITE inicial.

REQ.2c. Idem que REQ.1c.

REQ.2d. Responderá a las solicitudes en base a lo explicado en el capítulo 2.

REQ.2e. Una vez que se ha terminado la reserva de recursos, el destinatario tendrá que alertar sobre esto y para ello enviará una respuesta 180 RINGING. Esta respuesta la enviará el destinatario cada segundo hasta que el propio destinatario decida aceptar el establecimiento de la llamada.

REQ.2f. El MAS indica al cliente el puerto donde se desea establecer la sesión de voz multiusuario.

REQ.2g No se define un temporizador a partir del cual el destinatario deje de mandar el 180 RINGING. Esta respuesta se enviará hasta que el usuario decida aceptar el establecimiento de sesión.

REQ.2h. Idem que REQ.1h.

REQ.2i. Idem que REQ.1i.

REQ.2j. Idem que REQ.1j.

REQ.2k. Idem que REQ.1k.

REQ.2l. Idem que REQ.1h.

REQ.2m. No se contempla el rechazo de llamadas. El destinatario esta obligado bien a aceptar la llamada o bien a quedarse en espera.

REQ.2n. Idem que REQ.1n.

REQ.2o. No se permite poner las llamadas en espera ni tampoco hacer transferencias de llamadas.

3.2 Plano de datos

En el plano de datos se han identificado los siguientes requisitos:

REQ.3a. Todos los clientes han de desarrollar un proceso para subscribirse al grupo *multicast* una vez finalizada la reserva de recursos y conocida por tanto la dirección IP para la multidifusión de los paquetes de voz.

REQ.3b. Del mismo modo los clientes han de poder desvincularse de este grupo una vez abandonen la sesión.

REQ.3c. Procesamiento de la voz. Se ha de desarrollar un mecanismo capaz capturar la voz analógica través de un dispositivo de entrada, en este caso el micrófono, muestrearla, cuantificarla y codificarla de acuerdo al codec negociado transformándola en una secuencia de bits y situar esta ristra de bits en el *payload* de los paquetes UDP para que sean enviados a nivel de transporte a la dirección IP de multidifusión.

REQ.3d. Cuando la aplicación reciba los paquetes de voz ha de ser capaz de realizar el proceso inverso al requisito REQ.3c de tal manera que permita extraer del *payload* de los paquetes UDP la ristra de bits, decodificarlos y a partir de ellos reconstruir la señal original y reproducirla a través de un dispositivo de salida, en este caso el auricular del terminal.

REQ.3e. Se ha de implementar un mecanismo que permita el cese momentáneo del envío de los paquetes de voz para el caso en el que no existan receptores activos en la sesión multiusuario (por ejemplo el caso de que varios hayan abandonado la misma). Del mismo modo, debe existir otro mecanismo que pueda reanudar el envío de los mismos.

4. Implementación de los elementos del sistema

El sistema a implementar se muestra a nivel lógico en la figura 23.

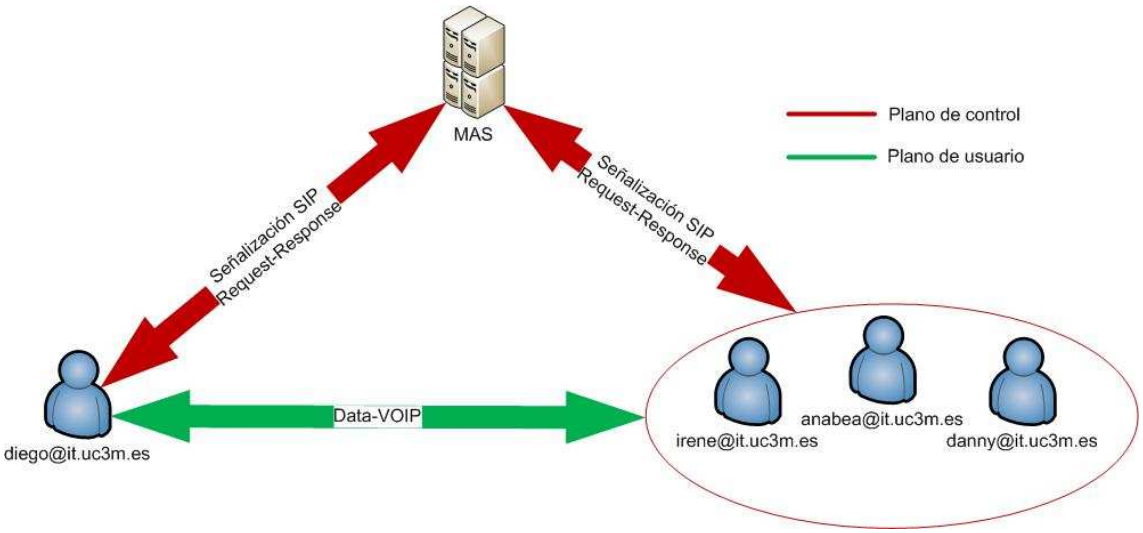

Figura 23 – Arquitectura lógica del sistema a implementar

Tal y como se puede observar, la implementación estará referida a dos planos distintos:

- Implementación de un plano de control mediante señalización SIP/SDP. Se hará uso de este protocolo para gestionar la señalización oportuna previa a la comunicación de voz entre los usuarios. En este proceso participará de manera activa el servidor de aplicaciones multiusuario (MAS), de tal manera que todas las solicitudes SIP que genere el usuario diego@it.uc3m.es serán enviadas al MAS, éste las procesara e identificará al grupo al que va destinado la invitación, en este caso irene@it.uc3m.es, anabea@it.uc3m.es y danny@it.uc3m.es. Del mismo modo las respuestas originadas por los destinatarios serán enviadas al MAS y este las enviará al iniciador de la sesión. En resumidas cuentas, toda la señalización tendrá necesariamente que atravesar el MAS para su procesamiento.

- Implementación de un plano de usuario mediante la transmisión de paquetes de voz sobre ip. Una vez terminada la señalización y asumiendo que los usuarios han aceptado la sesión, los paquetes de voz no seguirán el *path* anterior ya que el MAS no intenvendrá en el envió de datos de usuario. Los paquetes serán enviados a través de multidifusión IP al puerto que se negoció en la fase de señalización.

Pasaremos ahora a describir el entorno sobre el que ha desarrollado la aplicación en su totalidad. La solución está formada por una serie de módulos. Todos ellos están desarrollados en JAVA.

Para gestionar la señalización SIP en Java, existe una interfaz denominada JAIN SIP. Esconde toda la complejidad de la sintaxis de los mensajes SIP y del transporte de los mismos, a través de una interfaz con una pila de protocolo SIP, dejando al usuario gestionar los mecanismos a nivel de transacción. El método de funcionamiento se basa en eventos/mensajes y la implementación de SIP Listener y SIP Provider (figura 24). SIP Provider transforma mensajes SIP que recibe de la red en eventos para pasárselos a la aplicación, mientras que SIP Listener escucha estos eventos y los encapsula dentro de mensajes para que se gestione la comunicación pertinente.

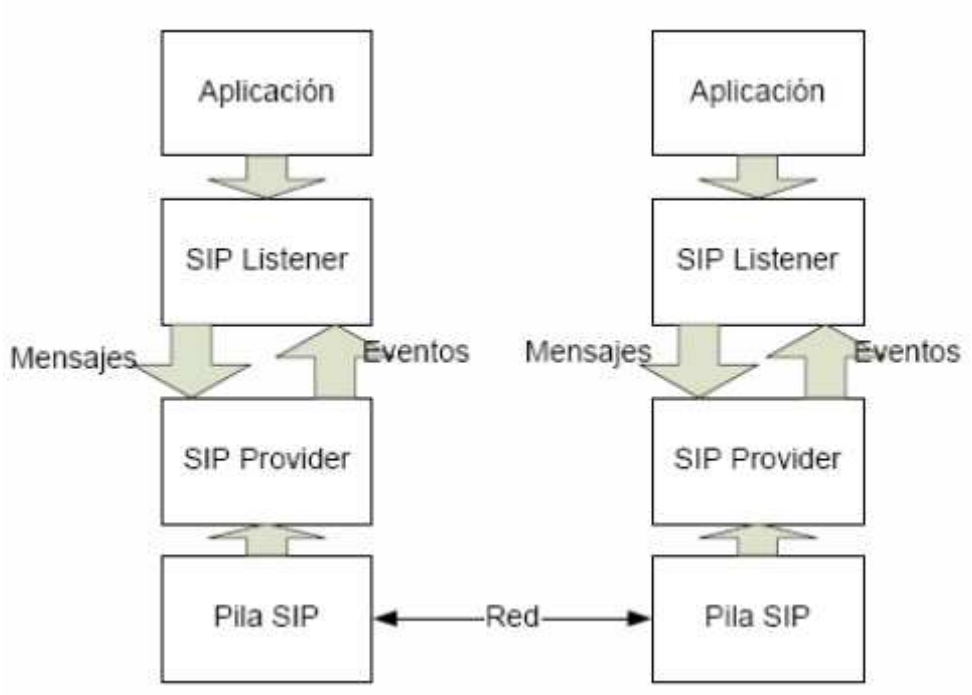

Figura 24 – Implementación pila SIP en JAVA

La colección de librerías empleadas en este proyecto que permiten implementar tanto la descripción de sesiones mediante SDP como la gestión de la señalización de SIP es la siguiente:

- *log4j-1.2.8.jar*. Es una librería que permite centralizar y administrar los mensajes de debugging y avisos de la aplicación. Tiene la capacidad de habilitar y deshabilitar ciertos logs, mientras otros no sufren ninguna alteración. Esto se realiza categorizando los mensajes de logs de acuerdo al criterio del programador.
- *concurrent.jar*. Es una librería que gestiona la ejecución de procesos concurrentes.
- *jainSipApi1.2.jar*. Contiene la implementación completa de la especificación correspondiente al protocolo SIP. Se trata de un API estándar de Java para el acceso a bajo nivel a una pila de protocolo SIP.
- *nist-sdp-1.0.jar*. Implementación de referencia del protocolo SDP del NIST.
- *jainSipRi1.2.jar*. Contiene la implementación de referencia del protocolo SIP.

En el plano de datos la aplicación se ha desarrollado utilizando también las librerias de JAVA que permiten la captura y el procesado de la voz por un lado (*javax.sound*) y la susbcripción y abandono de los grupos de multidifusión utilizando el protocolo IGMP (*java.net*).

La figura 25 representa un esquema lógico general de la aplicación completa con todos los módulos.

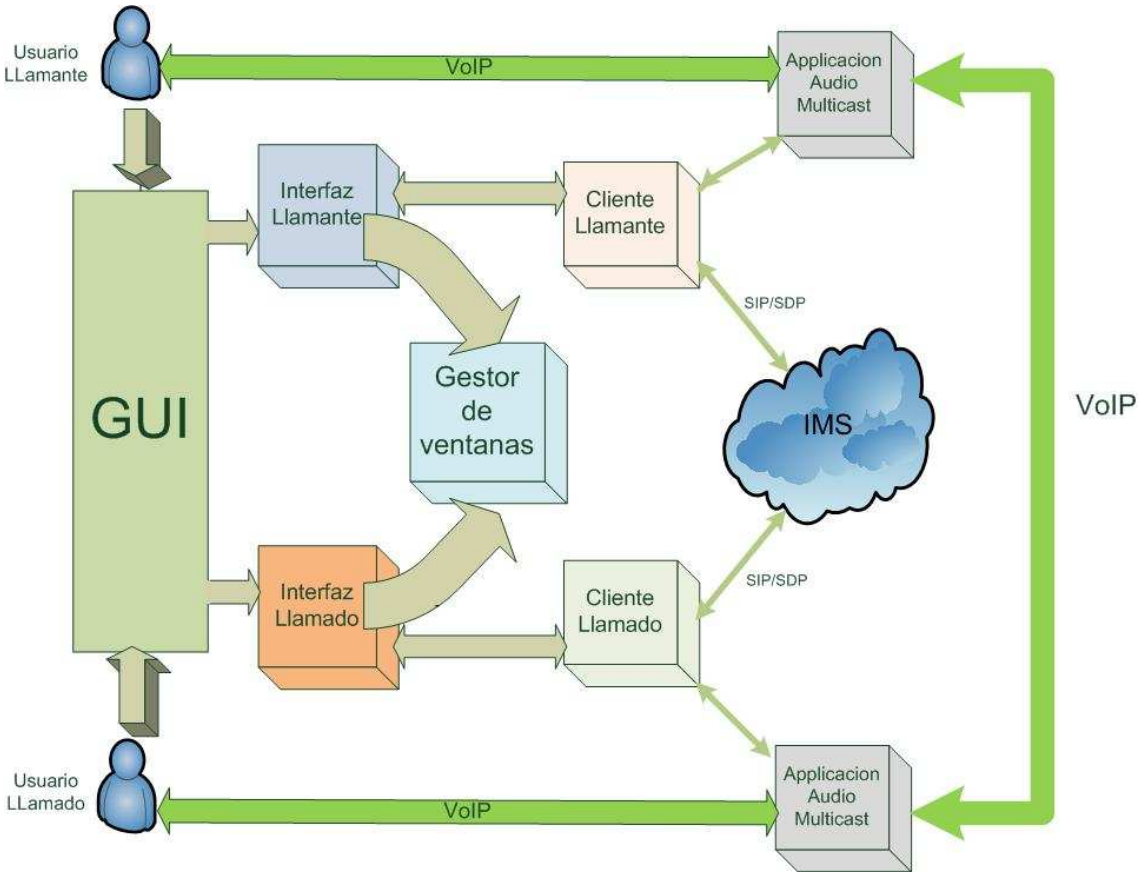

Figura 25 – Esquema lógico general de la aplicación

Como se puede comprobar tanto el usuario llamante como el llamado disponen de una GUI (*Graphical User Interface*) que supone el punto de contacto entre el usuario y la aplicación. Dependiendo si se trata de un usuario llamante o llamado entrarán en juego distintos módulos.

En el caso del usuario llamante la información introducida en la GUI es transferida al módulo interfaz llamante que a su vez se comunica con otros dos módulos: el gestor de ventanas para sacar por pantalla la información pertinente a la llamada y el cliente llamante que es el módulo que se encarga de generar las solicitudes y procesar las respuestas SIP así como gestionar la negociación SDP a través del IMS. Además se comunica con el módulo que gestiona la aplicación de voz *multicast* y que se arrancará una vez que alguno de los usuarios destino acepte el establecimiento de sesión.

Para el usuario llamado el esquema es prácticamente similar al anterior, en este caso se tiene el módulo interfaz llamado en vez del llamante y el módulo cliente llamado en lugar del llamante que desempeñan funciones idénticas pero en este caso referido a la

parte llamada. Además comparten el gestor de ventanas de la aplicación. No solo comparten este módulo, también comparte el módulo que gestiona la aplicación de voz *multicast* que permite establecer la sesión de voz multiusuario. En este caso el cliente llamado llamará a este parte una vez que haya terminado la reserva de recursos y acepte el establecimiento de sesión.

A continuación se describe de manera mas detallada la implementación de cada una de las partes de la aplicación.

4.1 Cliente llamante

4.1.1 Interfaz gráfica

Como ya se ha mencionado la implementación del cliente iniciador está basada en una interfaz gráfica donde el usuario podrá insertar los parámetros relativos al establecimiento de sesión. La aplicación está desarrollada en JAVA de manera que en el momento que el usuario lanza la aplicación se le presenta una ventana. Esta ventana esta compuesta por dos pestañas.

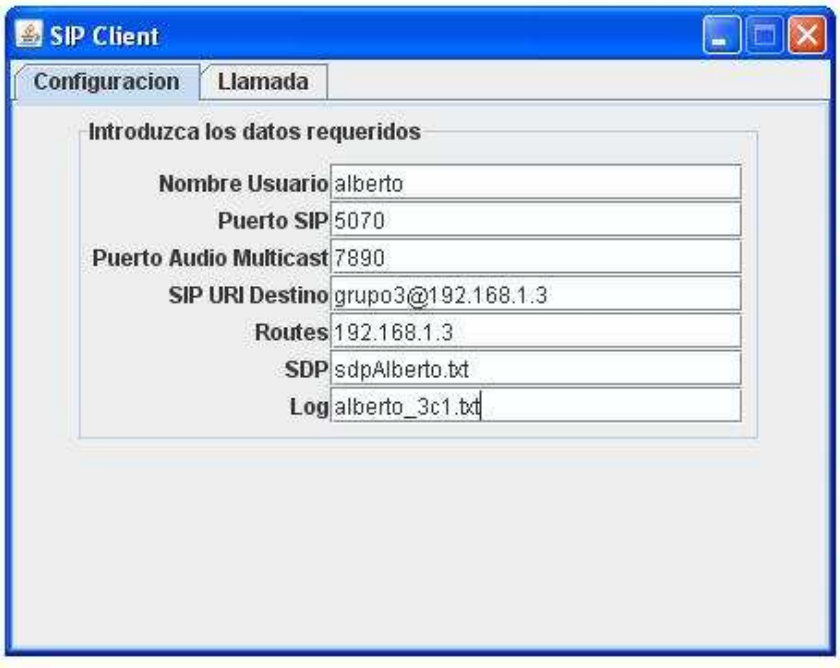

Figura 26 – Pestaña Configuración GUI cliente Llamante

La pestaña *Configuración* permite introducir los parámetros de configuración básicos acorde con los requisitos definidos en el capítulo 3.1.1.

El primer parámetro que se pide es el nombre de usuario (REQ.1a). A continuación se ha de indicar el puerto SIP (REQ.1b). Ha de ser un número que no coincida con un puerto que esté en uso como por ejemplo el puerto 53 de DNS, el puerto 123 de NTP, etc.…

En el campo Puerto Audio Multicast se especifica el puerto de acuerdo con el requisito REQ.1f. Se trata de un valor numérico que ha de cumplir las mismas especificaciones que las del puerto SIP.

También se ha de especificar la SIP URI destino (REQ.1d) a la que se dirige la petición inicial INVITE. La SIP URI tendrá el formato *destinatario@ip*.

El campo *Routes* especificará la lista de proxies SIP de acuerdo con el requisito REQ.1e Cada salto estará separado por comas y sin ningún espacio entre medias. El formato de este campo será *proxy1@ip1,proxy2@ip2,proxy3@ip3....*

El campo SDP especifica el fichero de texto donde se almacena la información SDP que irá encapsulada dentro del cuerpo SIIP. El fichero ha de almacenarse en el mismo directorio en el que se ejecuta la aplicación. El formato que ha de tener el fichero es el siguiente:

```
m=audio RTP/AVP 0 96 97
a=rtpmap:0 PCMU
a=rtpmap:96 G726-32/8000
a=rtpmap:97 AMR-WB
```

La primera línea del fichero contendrá la línea m de la carga SDP. Aquí se ha de especificar el tipo de tráfico enviado acorde al requisito REQ.1i. Además se especifica el protocolo de transporte utilizado (RTP/AVP) y la lista de codecs soportados por el iniciador de la llamada (REQ.1j). A continuación han de parecer tantas líneas *rtpmap* como formatos de codificación soportados por el cliente. Estas líneas realizan el mapeo del formato de codificación con un determinado número.

Finalmente aparece un campo Log donde se especifica el nombre del fichero que almacenará el histórico de la llamada (REQ.1n).

Todos los campos de texto han de rellenarse para que la aplicación se inicie de manera correcta, en caso contrario la aplicación generará un error y no se podrá lanzar.

La aplicación cuenta con otra pestaña denominada *Llamada*.

Figura 27 – Pestaña Llamada GUI cliente Llamante

Esta pestaña cuenta con varios botones:

- Iniciar. Al presionarse este botón se iniciarán las variables del sistema.
- Llamar. El usuario presionará este botón cuando decida iniciar el establecimiento de sesión.
- Colgar. Este botón se ha de presionar cuando el usuario decide abandonar la sesión.
- Salir. Al pulsar este botón se cerrará la aplicación.

Además el sistema dispone de varias ventanas emergentes para advertir al usuario de ciertos eventos.

Tendremos una ventana emergente cuando alguno de los usuarios acepte el establecimiento de la sesión (REQ.1g)

Figura 28 – Llamada aceptada cliente Llamante

Del mismo modo se mostrará una ventana emergente cuando todos los usuarios abandonen la sesión menos uno y además quede algún usuario que no haya aceptado la llamada de acuerdo con el requisito REQ.1h.

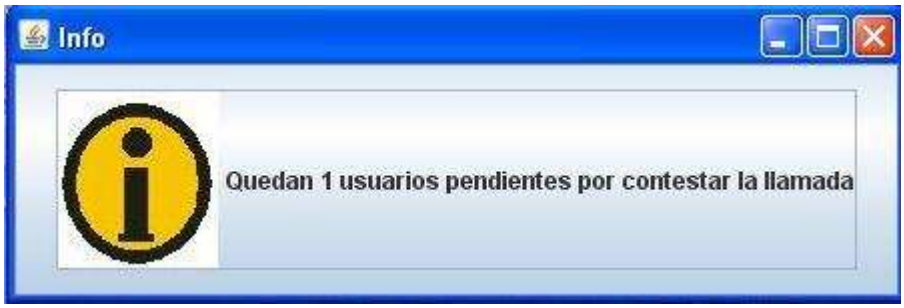

Figura 29 – Usuarios pendientes por aceptar la llamada

4.1.2 Procedimientos en el cliente llamante

Este apartado tiene como objetivo explicar todos los procesos que se suceden desde que el ususario inicia la sesión hasta que la libera.

Establecimiento de sesión.

Antes de empezar a generar la señalización propiamente dicha es necesario que el usuario introduzca los parámetros de configuración expuestos en el apartado anterior 4.1.1 y que presione el botón *Iniciar* de la figura 27 para arrancar las variables del sistema. La figura 30 describe este proceso

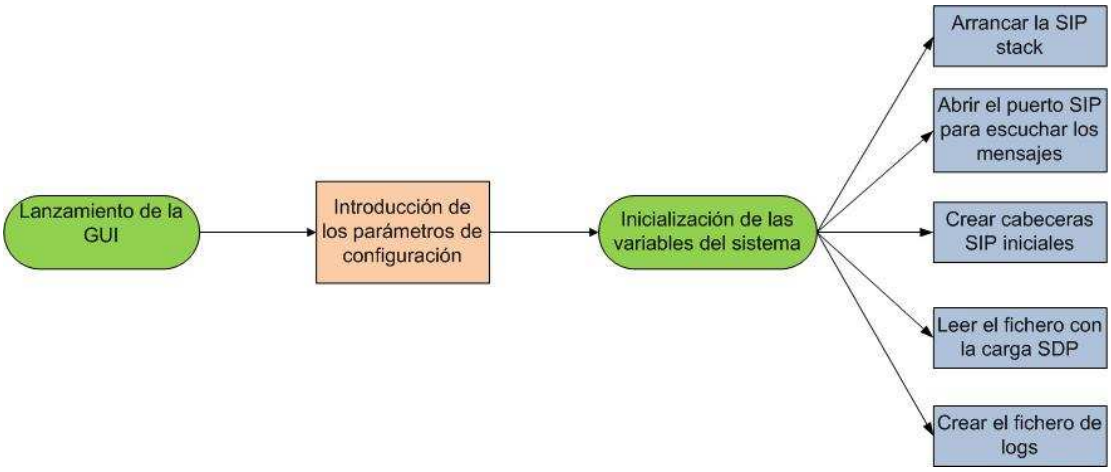

Figura 30 – Inicialización del sistema

Una vez que el sistema ha inicializado las variables, se está en disposición de generar la señalización adecuada. En primer lugar, el usuario llamante genera un INVITE dirigida a la dirección de grupo donde se encuentran registrados los usuarios destinatarios. Este proceso se ilustra en la figura 31.

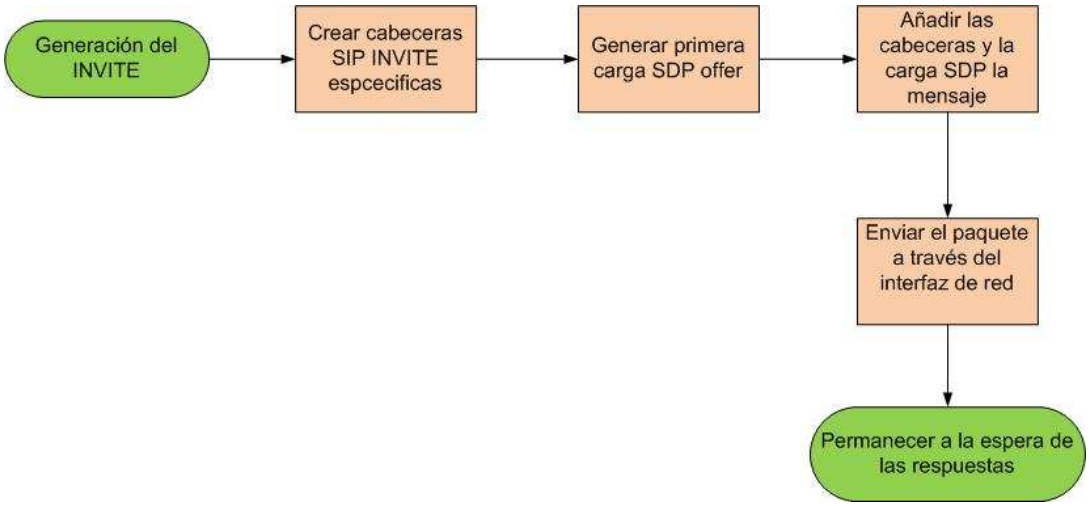

Figura 31 – Generación del INVITE

Cabe destacar que en este paso se crea la primera carga SDP cuyo proceso de generación se muestra en la siguiente figura.

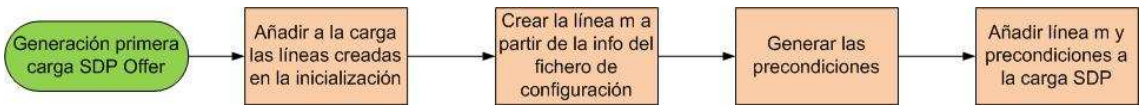

Figura 32 – Generación de la primera carga SDP

En este momento la aplicación pasa a un estado de espera a recibir las respuestas del MAS. Una vez que llega la respuesta combinada SESSION IN PROGRESS procedente del MAS, el cliente llamante la procesa y genera el PRACK tal y como se muestra en la figura 33.

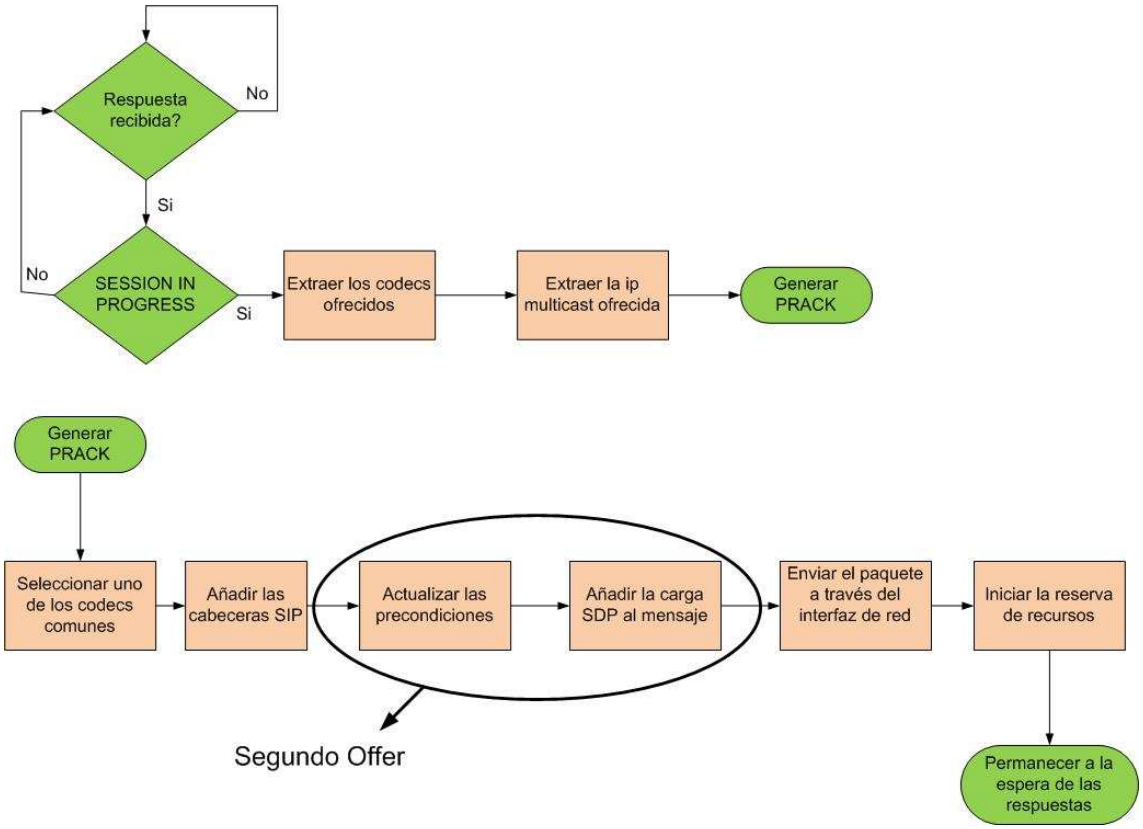

Figura 33 – Procesamiento del SESSION IN PROGRESS y generación del PRACK

En este momento es cuando el usuario inicia la reserva de recursos. Este mecanismo no se ha implementado en este proyecto. Sin embargo, se ha dejado abierto un método de reserva llamado *void reservarRecursos (SessionDescription sd)* al que se le ha pasado un objeto de la clase *SessionDescription* que representa el estado de las precondiciones.

Una vez mandado el PRACK, el cliente se queda a la escucha de los mensajes siguientes. El siguiente mensaje que el cliente iniciador ha de procesar es el 200 OK al PRACK que ha enviado previamente. Una vez recibido y procesado, genera un UPDATE para informar que la reserva de recursos ha terminado y comienza el proceso de suscripción al grupo multicast enviando un mensaje IGMP *report* al router.

Además el tercer crea SDP *Offer* con los prerrequisitos indicando que el cliente iniciador esta listo tanto para enviar como para recibir. Este proceso se muestra en la figura 34.

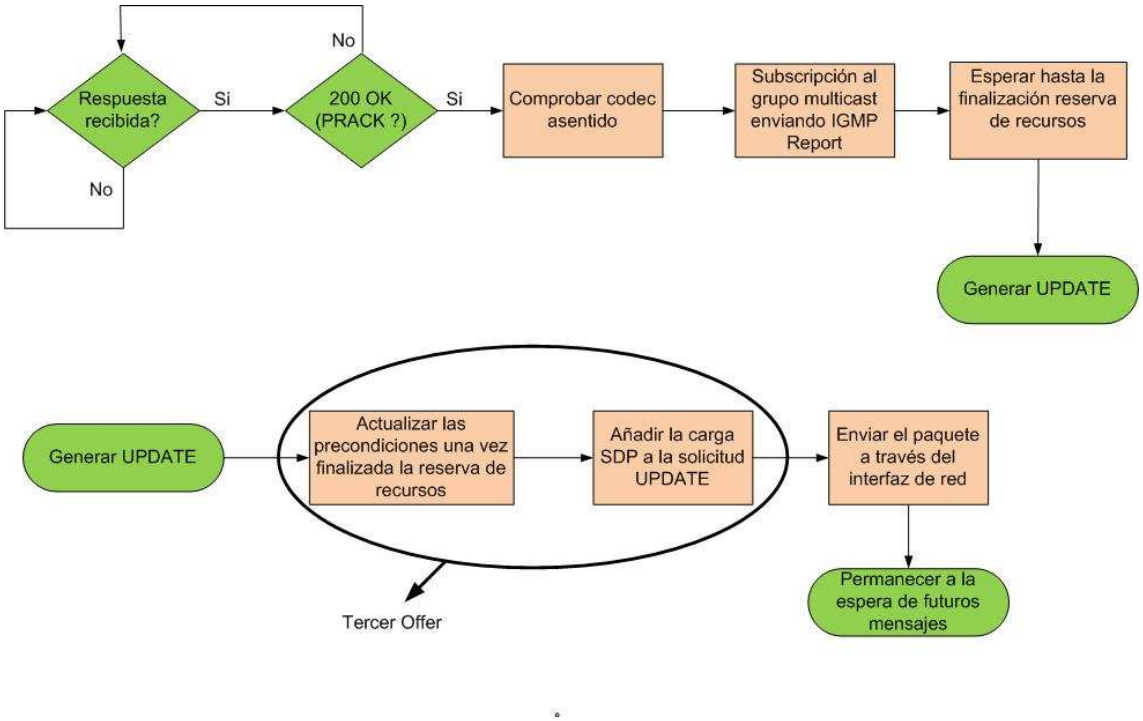

Figura 34 – Procesamiento del 200 OK (PRACK) y generación del UPDATE

Llegado a este punto, el cliente iniciador se pone a la espera de recibir la aceptación del establecimiento de sesión por parte del cliente llamado. Entre tanto el iniciador recibe el RINGING para alertarle que se está a la espera de que uno de los clientes llamados acepte la llamada. En el momento que el cliente iniciador recibe el 200 OK de uno de los destinatarios indicando que ha aceptado el establecimiento de sesión se producen varios eventos:

- El iniciador envía un ACK para concluir con el establecimiento de sesión.
- El inicador procesa el NOTIFY que recibe del MAS y actualiza el estado de la sesión multiusuario.
- Comienzo de la transmisión de los paquetes de voz.
- Contesta al MAS con un 200 OK al NOTIFY.

La figura 35 muestra el proceso entero.

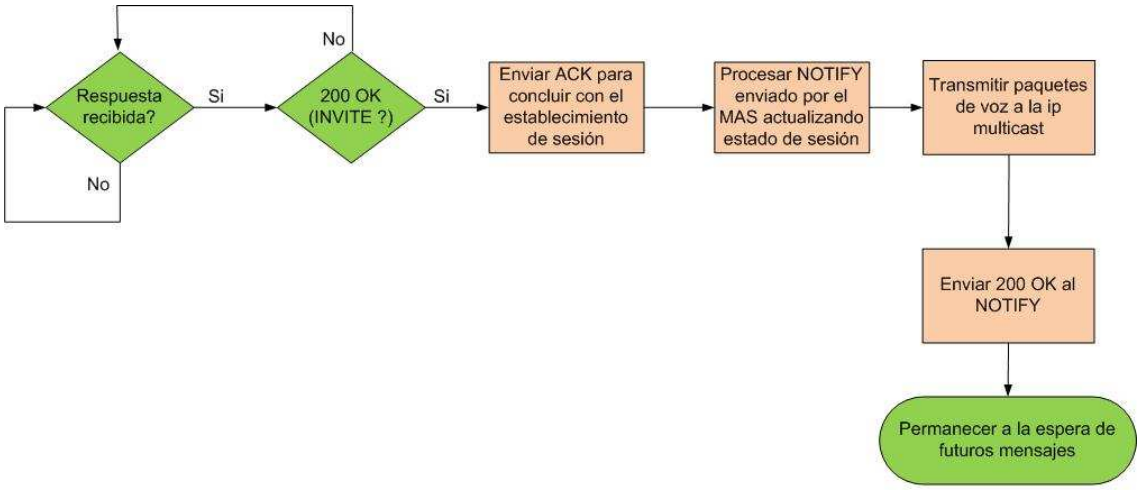

Figura 35 – Procesamiento del 200 OK (INVITE) y procesamiento del NOTIFY

Una vez finalizado este proceso el usuario ha establecido la sesión de voz multiusuario y el se encuentra transmitiendo los paquetes de voz. Además, el cliente se queda escuchando los posibles mensajes SIP ya que el MAS le puede mandar un BYE para indicarle que abandone la sesión.

Liberación de sesión.

Se pueden producir dos casos en la liberación de sesión en función de quien la inicie: iniciada por el cliente o inciada por el MAS.

Si es el usuario el que decide terminar la sesión, presionará el botón *Colgar* de la interfaz gráfica y se generarán los siguientes eventos:

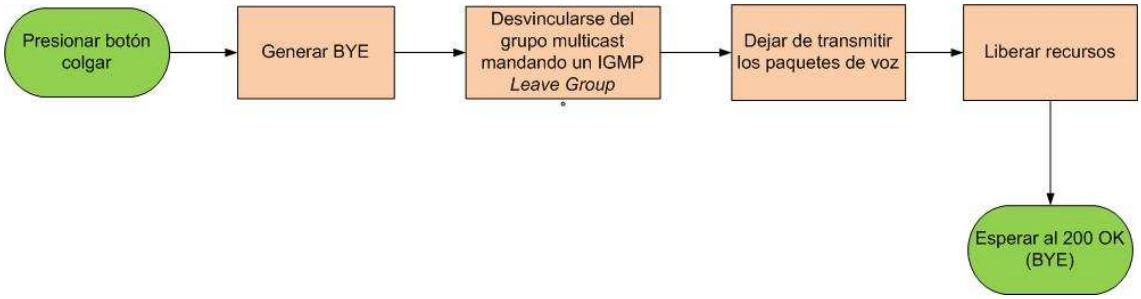

Figura 36 – Generación del BYE iniciado por el cliente

Si embargo, si es el MAS el que inicia la liberación se tiene un diagrama un tanto distinto

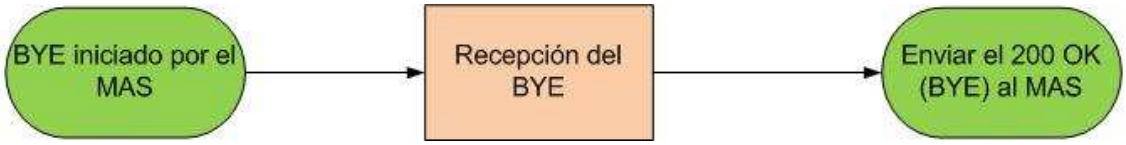

Figura 37 – Procesamiento del BYE iniciado por el MAS

4.2 Cliente llamado

4.2.1 Interfaz gráfica

Al igual que ocurre con el iniciador, se dispone de una interfaz gráfica que hace mas amigable la interacción con el usuario final y que servirá para meter los parámetros de configuración de la aplicación.

El interfaz constará de una ventana que tendrá dos pestañas similares a las del cliente originador de la llamada.

La primera de ellas se corresponde con la pestaña de *Configuración*.

Figura 38 – Pestaña Llamada GUI cliente Llamado

En esta pestaña podemos observar prácticamente las mismas opciones que en el caso del iniciador. Se dispone de un campo para introducir en nombre de usuario de acuerdo al requisito REQ.2a.

Se ha habilitado un campo (Puerto SIP) también para introducir el puerto donde se escucharan las solicitudes y respuestas SIP (REQ.2b). Ha de ser un número con las mismas restricciones que las que tiene el iniciador.

Además tenemos la opción SDP en la que se introducirá un fichero de texto que contiene los parámetros SDP del destinatario. Al igual que ocurre con el iniciador este fichero se debe de encontrar en el directorio desde donde se ejecuta la aplicación. El formato del fichero es exactamente el mismo que tenemos para el iniciador de la llamada.

Por último, tenemos la opción de generar un histórico de todos los mensajes enviados y recibidos por el usuario y escribirlos en el fichero que se especifica en la opción Log al igual que ocurre con el iniciador (REQ.2n). De manera similar el fichero quedará almacenado en el directorio de ejecución de la aplicación.

Todos los campos de texto han de rellenarse para que la aplicación se inicie de manera correcta, en caso contrario la aplicación generará un error y no se podrá lanzar.

Disponemos de otra pestaña denominada Llamada.

Figura 39 – Pestaña Llamada GUI cliente Llamante

Esta pestaña muestra 3 botones con opciones de llamada:

- Iniciar. Al presionarse este botón se inicializarán las variables del sistema
- Colgar. Este botón será presionado cuando el usuario decida abandonar la sesión.
- Salir. Al pulsar este botón se cerrará la aplicación.

Además la aplicación proporciona un sistema de ventanas emergentes que se lanzarán cuando se produzcan ciertos eventos.

En el momento que el usuario ha terminado la reserva de recursos se mostrará una ventana para advertir al usuario final que se tiene una llamada entrante

Figura 40 – Aviso de Llamada Entrante

En el momento que el usuario presione el botón, indicará que acepta la llamada y se estará en disposición de empezar la sesión de voz multiusuario.

Del mismo modo se mostrará una ventana emergente cuando todos los usuarios abandonen la sesión menos uno y además quede algún usuario que no haya aceptado la llamada de acuerdo con el requisito REQ.2h.

4.2.2 Procedimientos en el cliente llamado

Este apartado tiene como objetivo explicar todos los procesos que se suceden desde que el ususario recibe la invitación para el establecimiento de la sesión hasta que la libera.

Establecimiento de sesión

Una vez que se inicia la interfaz gráfica y se inicializan las variables, el cliente llamado permanece en un estado de espera hasta que recibe la primera solicitud que será el INVITE. Cuando esta solicitud llega el cliente actúa tal y como ilustra la figura 41.

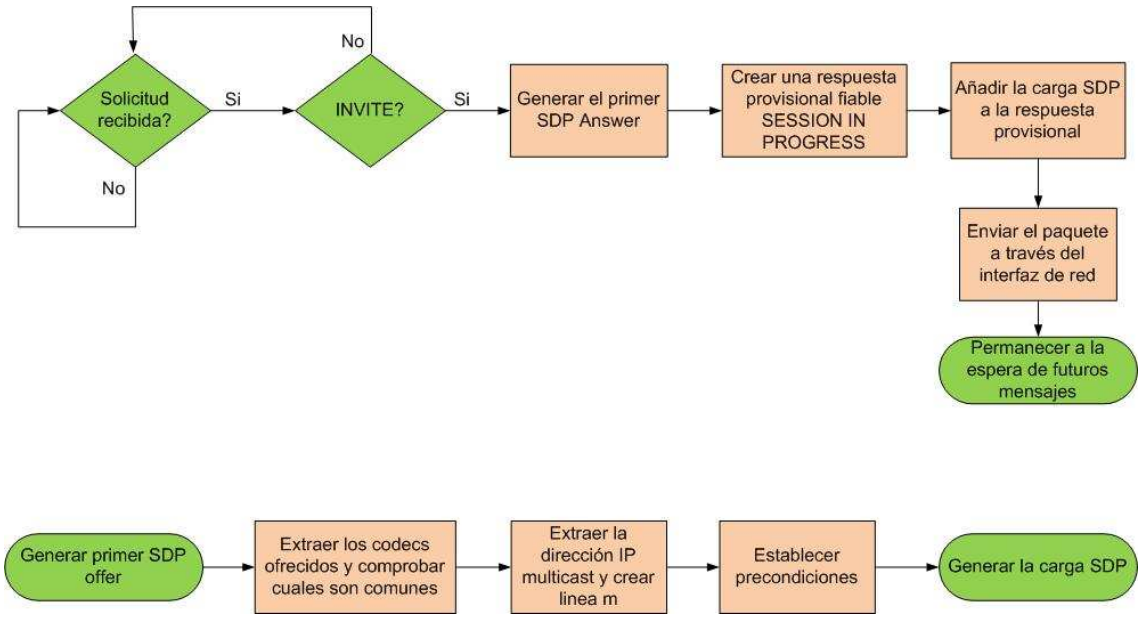

Figura 41 – Procesado del INVITE

Básicamente lo que hace el cliente es mirar el contenido de la carga SDP que recibe en el INVITE y a partir de ello conocer la ip multicast ofrecida por el MAS hacia donde

iran dirigidos los paquetes de voz y ver qué codecs son comunes entre los que ofrece el iniciador y los que soporta el destinatario. Además tambien fija las precondiciones y genera una respuesta provisional fiable 183 SESSION IN PROGRES.

Una vez que finaliza el proceso mostrado en la figura 41, el cliente pasa a un estado de escucha a las siguientes solicitudes SIP.

El siguiente mensaje que debe de recibir el cliente llamado es un PRACK que le envía el MAS. Este PRACK contendrá el codec ofrecido por el cliente iniciador y tendrá que ser uno de los que el llamante ofreció en el mensaje anterior SESSION IN PROGRESS. Además ha de actualizar las precondiciones y con todo esto generar la carga SDP (SDP *Answer*), ensamblarla en el 200 OK (PRACK), enviarla al MAS y comenzar la reserva de recursos La figura 42 muestra este proceso.

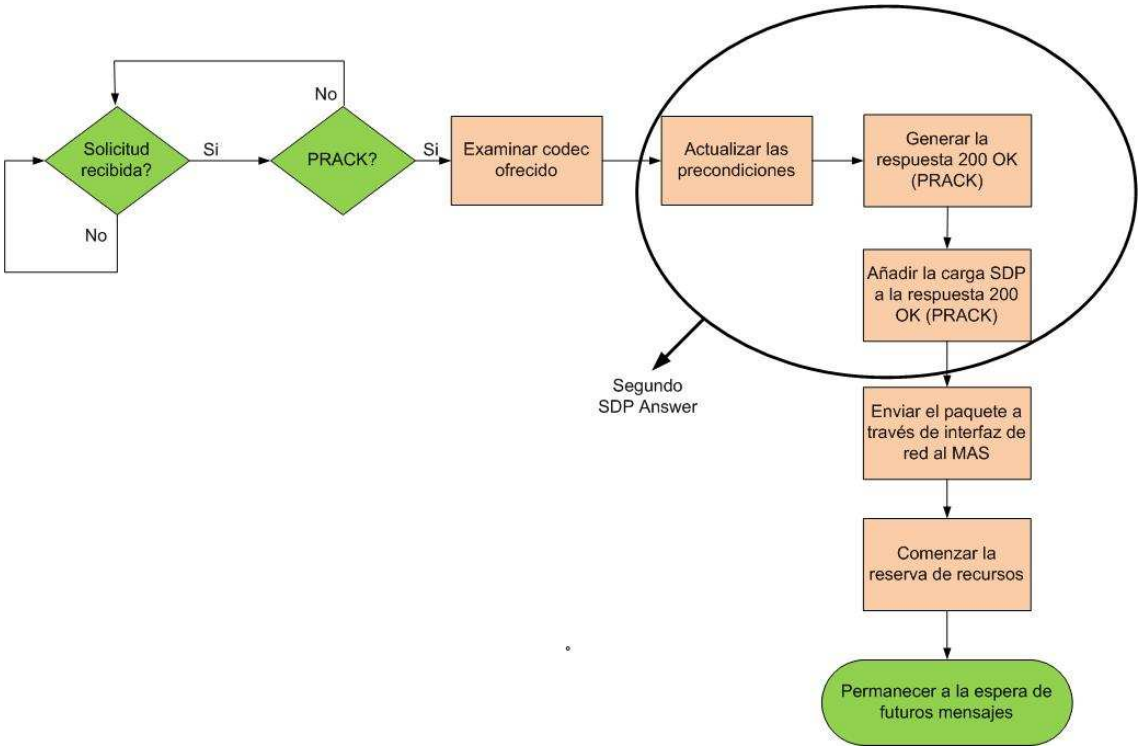

Figura 42 – Procesado del PRACK

Mientras que el llamado hace la reserva de recursos puede ocurrir que reciba la petición UPDATE (tercer SDP *Offer*) por parte del iniciador indicando que la reserva ha terminado y que porta los prerrequistos del iniciador actualizados. En este caso el llamado ha de responder con un 200 OK. El contenido de este mensaje depende del estado actual del llamado: si ha terminado la reserva enviará el 200 OK con el tercer

SDP *Answer* indicando el estado final de las precondiciones. Sin embargo, si no ha terminado la reserva de recursos el 200 OK llevará una carga SDP con las precondiciones actuales del llamado indicando que la reserva no se ha terminado todavía. Dado que en este proyecto no se contempla la implementación de la reserva de recursos, se mandará siempre el primer tipo de respuesta expuesta. La figura 43 muestra los eventos descritos anteriormente.

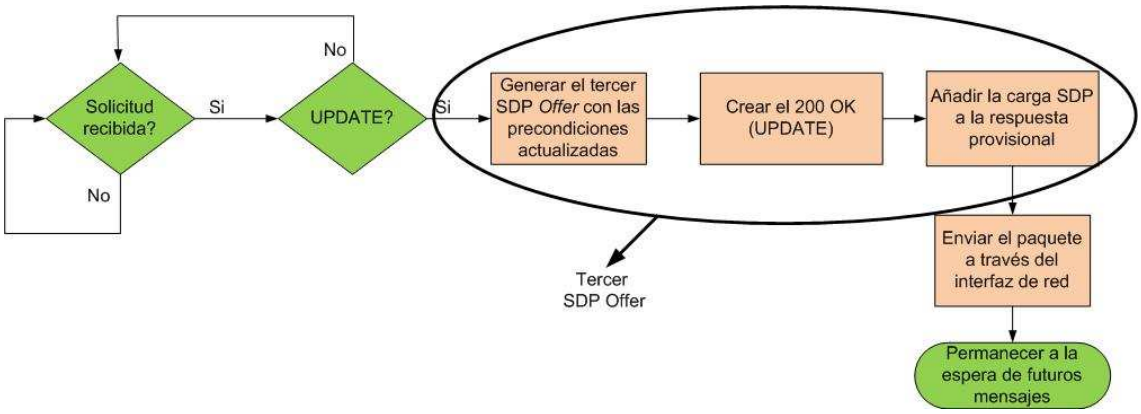

Figura 43 – Procesado del UPDATE

Una vez finalizada la reserva de recursos, el llamado envía un 180 RINGING al llamante para alertarle de que esta reserva ha finalizado. Este RINGING se envía periódicamente cada segundo hasta que el llamado acepta el establecimiento de llamada enviando un 200 OK (INVITE). Esta secuencia de eventos se muestra en la figura 44

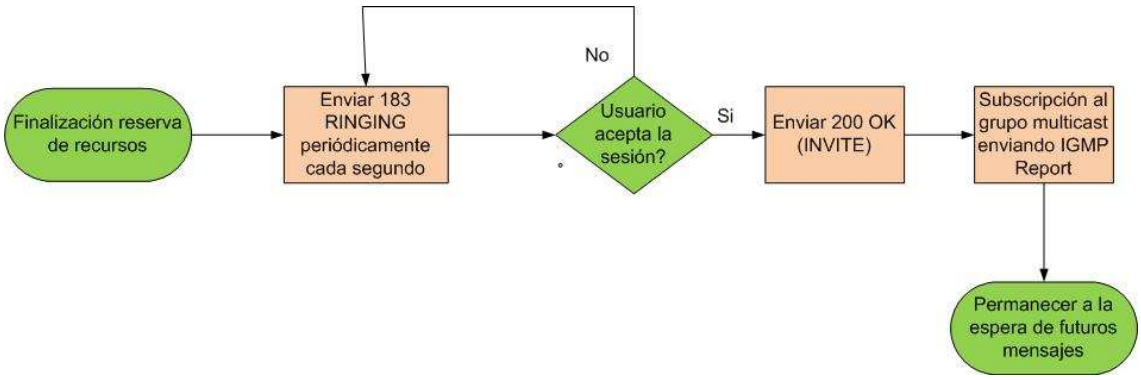

Figura 44 – Finalización de la reserva de recursos, generación del RINGING y aceptación de sesión

En el momento que el MAS recibe el 200 OK (INVITE), este genera un NOTIFY que se envía al llamado para actualizar el estado de la sesión. El llamado lo procesa,

comienza el envío de los paquetes de voz y genera el 200 OK tal y como muestra la figura 45.

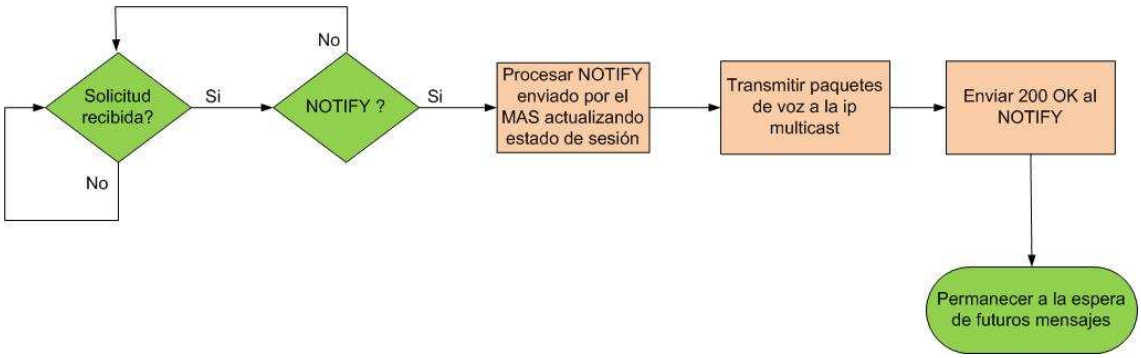

Figura 45 – Procesado del NOTIFY y trasmisión de los paquetes de voz

Una vez llegados a este punto el cliente llamado ha aceptado el establecimiento de sesión y se encuentra mandando tráfico de voz. Para el caso de la liberación de la llamada, los procedimiemtos son idénticos a los del iniciador (figuras 36 y 37)

5. Escenario de pruebas

Para la validar la implementación del sistema desarrollado y comprobar que cumple con los requisitos que se definieron en la sección 3 se ha diseñado un plan de pruebas.

5.1 Definición de las pruebas a realizar. Validación y análisis de los resultados obtenidos.

A continuación se definen el conjunto de pruebas que se realizaron para validar la implementación de la solución propuesta. Se han dividido las pruebas dependiendo del plano al que se refieran:

- Plano de control o señalización SIP/SDP.
- Plano de datos asociado al envío de la VoIP multicast.

El escenario de pruebas se basa en un entorno de red de área local Ethernet en el que los clientes y servidores se encuentran conectados por cable Ethernet a un switch y este switch a un router que soporta el envío de datagramas IP multicast. A continuación se muestra el entorno de pruebas.

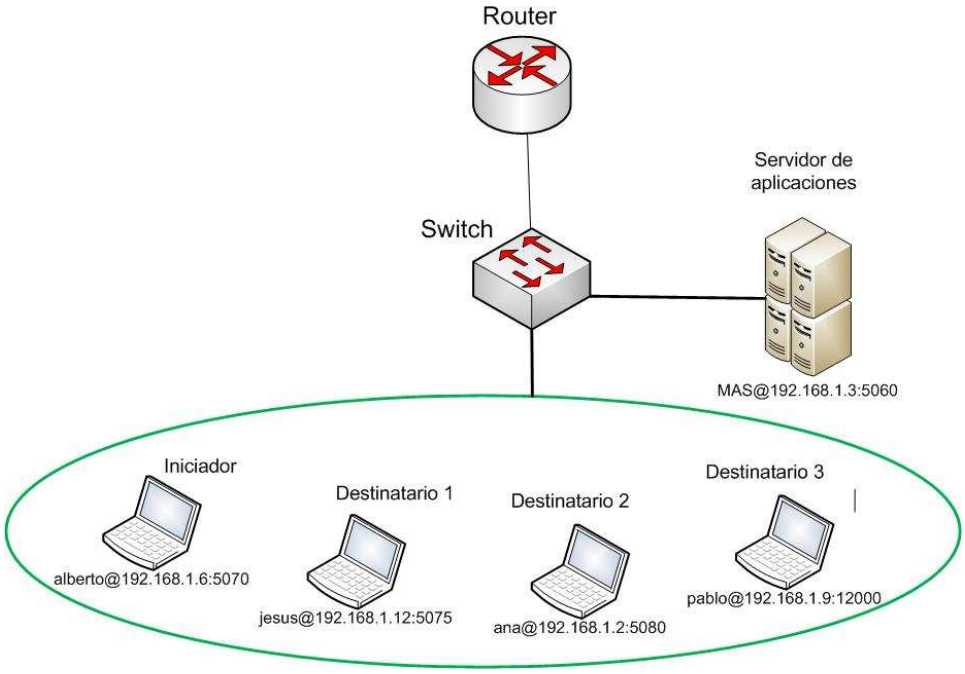

Figura 46 – Escenario de pruebas

5.1.1 Señalización SIP/SDP.

Este conjunto de pruebas pretende demostrar que la aplicación se comporta acorde a lo definido en el capítulo 3. Se presentarán diversos casos de establecimiento y liberación de llamadas y se verificará que la señalización SIP/SDP cumple con los requisitos del diseño.

En los casos de pruebas participan 4 clientes SIP y el servidor de aplicaciones MAS (Multiparty Application Server). Los *hosts* tienen las siguientes propiedades:

Host	Dirección IP	Puerto	Codecs soportados		
alberto	**192.168.1.6**	**5070**	**PCMU**	**PCMA/8000**	**G723/8000**
			GSM/8000	**G722/8000**	**G728/8000**
			G729/8000	**G726-32/8000**	**AMR-WB**
jesus	**192.168.1.12**	**5075**	**PCMU**	**PCMA/8000**	**GSM/8000**
			G728/8000	**G729/8000**	**AMR-WB**
Ana	**192.168.1.2**	**5080**	**PCMA/8000**	**G723/8000**	**GSM/8000**
			G728/8000	**G729/8000**	
pablo	**192.168.1.9**	**12000**	**PCMA/8000**	**GSM/8000**	**G729/8000**
			G726-32/8000	**AMR-WB**	
MAS	**192.168.1.3**	**5060**	**-**		

Tabla 9 – Listado de las propiedades de los clientes

Caso 1. Establecimiento y liberación de sesión entre 4 participantes.

En el presente caso el originador de la llamada (alberto) inicia el establecimiento de la sesión dirigida hacia jesus, ana y pablo todos ellos suscritos al *grupo3*. El primero en aceptar la llamada es jesus, seguido de ana y el último pablo.

En el establecimiento de sesión el codec que negociarán será el **PCMA/8000** ya que es el primero común a todos ellos. El puerto *multicast* que ofrece alberto es el **7890** y la ip multicast que ofrece el MAS para el envío de los paquetes de voz será la **224.10.10.20.**

Una vez que todos los participantes han aceptado la sesión, alberto decide abandonar la sesión enviando un BYE. A continuación lo hace ana, posteriormente pablo. En este momento jesus es el único usuario que queda activo en la sesión. El MAS es consciente de esta situación y envía un BYE a jesus para que abandone la sesión y jesus le ha de responder con un 200 OK.

Validación y análisis de los resultados obtenidos

Establecimiento de sesión

A continuación se muestra un diagrama del flujo de mensajes del establecimiento de la sesión obtenido en la prueba realizada.

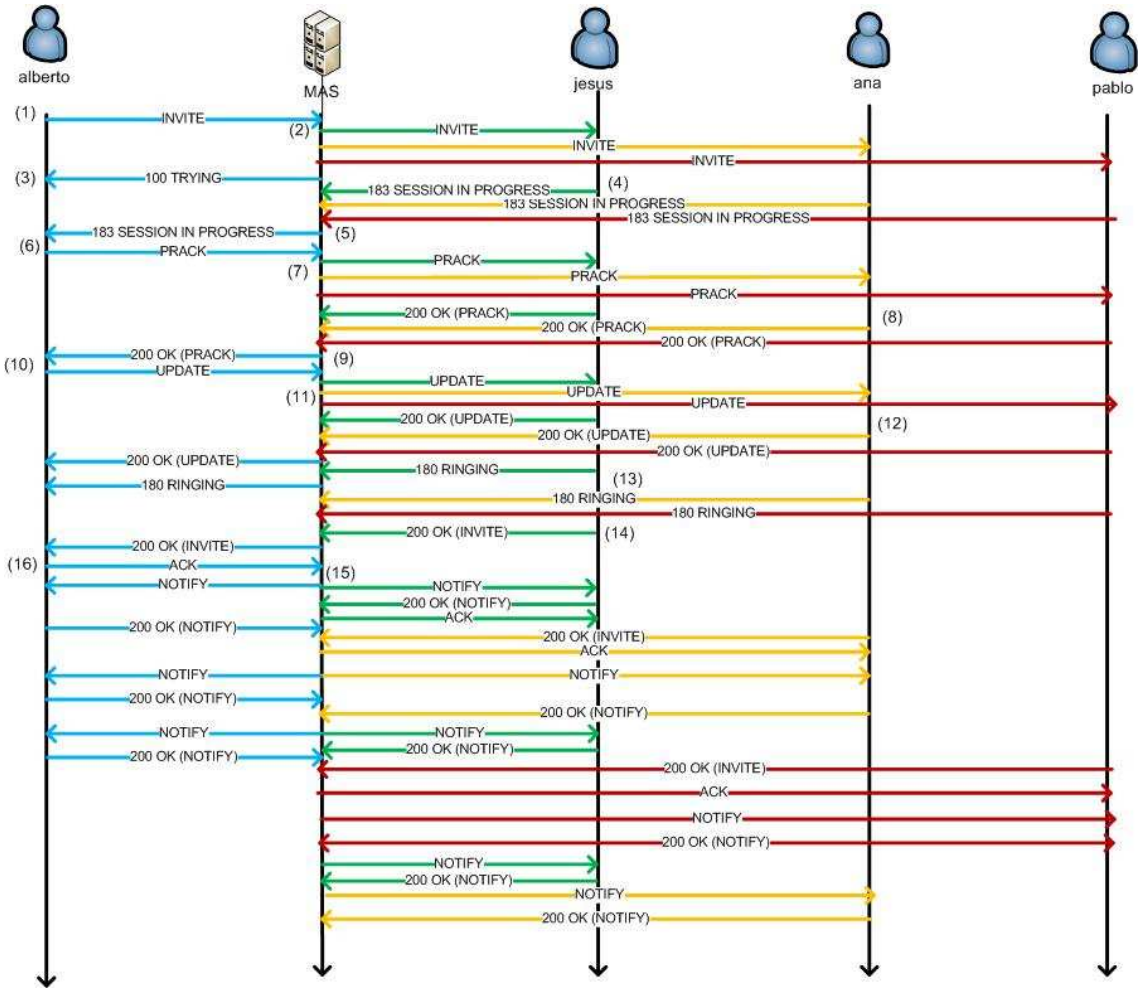

Figura 47 – Establecimiento de sesión 4 usuarios

Pasaremos a analizar el contenido de los mensajes que se cursaron durante el desarrollo de este establecimiento de sesión.

1. alberto inicia el establecimiento de sesión acorde a lo explicado anteriormente enviando un INVITE (punto 1 figura 47) a la dirección de grupo.

```
INVITE sip:grupo3@192.168.1.3 SIP/2.0
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
CSeq: 1 INVITE
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bKd6fa0c5a2b2519620dab7cc208600ef8
Max-Forwards: 70
Contact: <sip:alberto@192.168.1.6:5070>
Route: <sip:192.168.1.3;lr>
Supported: 100rel
Require: precondition
Content-Type: application/sdp
Content-Length: 420

v=0
o=alberto 760638 760638 IN IP4 192.168.1.6
s=-
t=0 0
m=audio 7890 RTP/AVP 0 8 4 3 9 15 18 96 97
a=rtpmap:0 PCMU
a=rtpmap:8 PCMA/8000
a=rtpmap:4 G723/8000
a=rtpmap:3 GSM/8000
a=rtpmap:9 G722/8000
a=rtpmap:15 G728/8000
a=rtpmap:18 G729/8000
a=rtpmap:96 G726-32/8000
a=rtpmap:97 AMR-WB
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos optional remote sendrecv
```

A la vista del primer mensaje se pueden observar varias cosas:

- La solicitud va dirigida a la dirección de *grupo3* tal y como se ve en la SIP URI.
- El campo *Route* indica que el primer salto será directamente el MAS (192.168.1.3)
- La cabecera *Supported:100rel* indica que el UA soporta el mecanismo de respuesta provisional en modo fiable.
- La cabecera *Require:precondition* indicando la presencia de unas precondiciones importantes en la negociación de la calidad de servicio.
- El mensaje contiene una carga SDP de acuerdo a la cabecera *Content-Type*
- De la carga SDP podemos extraer que:
    - alberto soporta los codecs de acuerdo a la tabla 9

- No aparece la línea “c” ya que será el MAS el que fije en que IP *multicast* se establecerá la sesión de voz multiusuario.
- Las precondiciones expresan:
  - Actualmente los requerimientos de calidad de servicio en el usuario local no están reservados (*a=curr:qos local none*).
  - Actualmente los requerimientos de calidad de servicio en el usuario remoto no están reservados (*a=curr:qos remote none*).
  - Se desea (des) que la calidad de servicio (qos) sea reservada en el usuario (local). Se ha de reservar recursos en ambos sentidos, envío y recepción (sendrecv) y la sesión no se iniciará hasta que ambos recursos, recepción y envío, hayan sido reservados (mandatory) (*a=des:qos mandatory local sendrecv*).
  - Se desea (des) que la calidad de servicio (qos) sea reservada en el usuario (remote) de manera opcional (optional) sin que la sesión se pare por ello. Los recursos han de reservarse en ambos sentidos (sendrecv) *(a=des:qos optional remote sendrecv*)

2. El MAS recibe el INVITE y lo reenvía a cada destinatario suscrito al grupo3 (punto 2 de la figura 47).

```
INVITE sip:jesus@192.168.1.12:5075 SIP/2.0
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
CSeq: 1 INVITE
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>
Max-Forwards: 70
Supported: 100rel
Require: precondition
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bK605e684eafb07f3101e5bcea20ddc6a2
Record-Route: <sip:mas@192.168.1.3:5060;lr>
Contact: <sip:8fc5dc77c2def4f5@192.168.1.3:5060>
Content-Type: application/sdp
Content-Length: 443

v=0
o=alberto 760638 760638 IN IP4 192.168.1.6
s=-
t=0 0
m=audio 7890 RTP/AVP 0 8 4 3 9 15 18 96 97
c=IN IP4 224.10.10.20
a=rtpmap:0 PCMU
a=rtpmap:8 PCMA/8000
a=rtpmap:4 G723/8000
a=rtpmap:3 GSM/8000
a=rtpmap:9 G722/8000
a=rtpmap:15 G728/8000
a=rtpmap:18 G729/8000
```

```
a=rtpmap:96 G726-32/8000
a=rtpmap:97 AMR-WB
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos optional remote sendrecv
```

Del mensaje recibido se puede extraer:

- El MAS añade la cabecera *Record-Route* con la URI del MAS.
- El MAS añade la línea c en la carga SDP ofreciendo la IP multicast (224.10.10.20) donde se han de suscribir los participantes en la sesión de voz multiusuario

3. alberto recibe respuestas temporales 100 TRYING por parte del MAS (punto 3 de la figura 47)

```
SIP/2.0 100 Trying
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
CSeq: 1 INVITE
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bKd6fa0c5a2b2519620dab7cc208600ef8
Content-Length: 0
```

4. Cada destinatario responde al INVITE con un 183 SESSION IN PROGRESS (punto 4 figura 47)

```
SIP/2.0 183 Session progress
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
CSeq: 1 INVITE
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bK605e684eafb07f3101e5bcea20ddc6a2
Require: 100rel
Contact: <sip:jesus@192.168.1.12:5075>
To: <sip:grupo3@192.168.1.3>;tag=91691105445
Content-Type: application/sdp
RSeq: 696
Content-Length: 417

v=0
o=jesus 476935 476935 IN IP4 192.168.1.12
s=-
t=0 0
m=audio 7890 RTP/AVP 0 8 3 15 18 97
c=IN IP4 224.10.10.20
a=rtpmap:0 PCMU
a=rtpmap:8 PCMA/8000
a=rtpmap:3 GSM/8000
a=rtpmap:15 G728/8000
a=rtpmap:18 G729/8000
```

```
a=rtpmap:18 G729/8000
a=rtpmap:97 AMR-WB
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
a=conf:qos remote sendrecv
```

En esta respuesta podemos encontrar:

- Una cabecera *Require:100rel* que indica que el UA que reciba la respuesta ha de mandar un mensaje PRACK de confirmación de respuesta provisional recibida.
- Una cabecera *RSeq* para poder distinguir entre respuestas provisionales.
- Una carga SDP (SDP *Answer*) que muestra:
    - Una línea *m* que indica los codecs comunes que soportan jesus y alberto
    - Una línea c que indica la dirección IP *multicast* (224.10.10.20) donde se establecerá la sesión de voz multiusuario.
    - Unas precondiciones que denotan:
        - Actualmente los requerimientos de calidad de servicio en el usuario local no están reservados (*a=curr:qos local none*).
        - *a=curr:qos remote none* ➔ este prerrequisito se origina a partir del primer SDP *Offer* que recibe la entidad llamada e indica que el llamante (en este momento el usuario remoto) no tenia ningún recurso reservado en ese momento.
        - *a=des:qos mandatory local sendrecv* ➔ indica que el usuario llamado necesita hacer una reserva de recursos en ambos sentidos de la comunicación. Es un prerrequisito que el usuario conoce de antemano.
        - *a=des:qos mandatory remote sendrecv* ➔ se genera a partir del primer SDP *offer*. En el se indica que el usuario remoto necesita de la reserva de recursos.
        - *a=conf:qos remote sendrecv* ➔ indica que la entidad que generó la llamada deberá mandar un mensaje de confirmación (*conf*) a la entidad llamada (*remote*) una vez que se haya terminado la reserva de recursos en la entidad llamante.

5. El MAS espera a que se reciban las respuestas individuales de cada destinatario para construir una respuesta 183 SESSION IN PROGRESS conjunta (punto 5 de la figura 47)

```
SIP/2.0 183 Session progress
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
CSeq: 1 INVITE
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=badaaf68218d7113
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bKd6fa0c5a2b2519620dab7cc208600ef8
Require: 100rel
Contact: <sip:8fc5dc77c2def4f5@192.168.1.3:5060>
Record-Route: <sip:mas@192.168.1.3:5060;lr>
Content-Type: application/sdp
RSeq: 749
Content-Length: 416

v=0
o=pablo 183467 183467 IN IP4 192.168.1.9
s=-
t=0 0
m=audio 7890 RTP/AVP 8 3 18
c=IN IP4 224.10.10.20
a=rtpmap:8 PCMA/8000
a=rtpmap:3 GSM/8000
a=rtpmap:15 G728/8000
a=rtpmap:18 G729/8000
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
a=conf:qos remote sendrecv
```

En este mensaje podemos destacar que en la respuesta conjunta del MAS solo aparecen ya los codecs comunes a los 4 participantes

6. alberto envía un PRACK (punto 6 de la figura 47) y comienza la reserva de recursos.

```
PRACK sip:8fc5dc77c2def4f5@192.168.1.3:5060 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bK206cec700a07105443d718d729a66523
CSeq: 2 PRACK
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=badaaf68218d7113
Max-Forwards: 70
Route: <sip:mas@192.168.1.3:5060;lr>
RAck: 749 1 INVITE
Supported: 100rel
Content-Type: application/sdp
Content-Length: 272

v=0
o=alberto 760638 760638 IN IP4 192.168.1.6
s=-
t=0 0
m=audio 7890 RTP/AVP 8
c=IN IP4 224.10.10.20
```

```
a=rtpmap:8 PCMA/8000
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
```

En esta solicitud podemos destacar:

- Se ha seleccionado uno de los codecs comunes (codec 8)
- Hay que notar que ahora el iniciador si especifica la línea “c” dado que ya conoce la ip multicast en la que se va establecer la conferencia multiusuario
- Se ha generado el segundo SDP *Offer* con la siguiente información cuya información es simalar al primer SDP *Offer* salvo que las precondiciones en el iniciador no varían excepto la última línea *a=des:qos mandatory remote sendrecv* en la que se indica que la entidad llamada también necesita hacer reserva de recursos en ambos sentidos.

7. Ese PRACK es reenviado a cada destinatario (punto 7 de la figura 47).

```
PRACK sip:jesus@192.168.1.12:5075 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bK175976c940701e68e0f8d3822ecf0d8e
CSeq: 2 PRACK
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=91691105445
Max-Forwards: 70
RAck: 696 1 INVITE
Content-Type: application/sdp
Content-Length: 272

v=0
o=alberto 760638 760638 IN IP4 192.168.1.6
s=-
t=0 0
m=audio 7890 RTP/AVP 8
c=IN IP4 224.10.10.20
a=rtpmap:8 PCMA/8000
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
```

8. Cada destinatario responde con un 200 OK (punto 8 de la figura 47) y comienza la reserva de recursos en cada destinatario. El MAS recoge cada 200 OK y manda un solo 200 OK a alberto (punto 9 de la figura 47)

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bK206cec700a07105443d718d729a66523
CSeq: 2 PRACK
```

```
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=badaaf68218d7113
Content-Type: application/sdp
Content-Length: 259

v=0
o=user 0 0 IN IP4 192.168.1.3
s=-
t=0 0
m=audio 7890 RTP/AVP 8
c=IN IP4 224.10.10.20
a=rtpmap:8 PCMA/8000
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
```

9. Una vez que alberto ha finalizado la reserva de recursos procede al envío de un UPDATE (punto 10 de la figura 47)

```
UPDATE sip:8fc5dc77c2def4f5@192.168.1.3:5060 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bKed718a243c37970f2929089e481f51c8
CSeq: 3 UPDATE
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=badaaf68218d7113
Max-Forwards: 70
Contact: <sip:192.168.1.6:5070;transport=udp>
Route: <sip:mas@192.168.1.3:5060;lr>
Content-Type: application/sdp
Content-Length: 232

v=0
o=alberto 760638 760638 IN IP4 192.168.1.6
s=-
t=0 0
m=audio 7890 RTP/AVP 8
c=IN IP4 224.10.10.20
a=rtpmap:8 PCMA/8000
a=curr:qos local sendrecv
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
```

En este momento se actualizan el estado de las precondiciones y se genera el tercer SDP *Offer.* Podemos observar las en las precondiciones que en la línea *a=curr:qos local sendrecv* ha pasado de “none” a “sendrecv” indicando que el iniciador ha finalizado la reserva de recursos y esta listo tanto para mandar como para recibir los paquetes de voz.

10. El MAS replica cada UPDATE y lo envía a la lista de destinatarios (punto 11 de la figura 47). Cada uno de ellos contesta con un 200 OK indicando se ha finalizado la reserva de recursos en el lado del destino (punto 12)

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bKf2e95c7c03a1b491008bfc54a64030f7
CSeq: 3 UPDATE
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=91691105445
Content-Type: application/sdp
Content-Length: 257

v=0
o=jesus 476935 476935 IN IP4 192.168.1.12
s=-
t=0 0
m=audio 7890 RTP/AVP 8
c=IN IP4 224.10.10.20
a=rtpmap:8 PCMA/8000
a=curr:qos local sendrecv
a=curr:qos remote sendrecv
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
```

Dado que la reserva ha finalizado en ambas partes se han de actualizar las precondiciones y generar el tercer SDP *Answer.* Se puede comprobar ahora que tanto el estado actual del usuario local como el remoto están listo para enviar y recibir (*sendrecv*)

11. Justo después cada destino alerta a alberto que ha terminado la reserva de recurso enviando un 180 RINGING cada segundo y que está en disposición de aceptar el establecimiento de sesión (punto 13 figura 47).

```
SIP/2.0 180 Ringing
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
CSeq: 1 INVITE
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=91691105445
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bK605e684eafb07f3101e5bcea20ddc6a2
Content-Length: 0
```

12. En el caso que nos ocupa es jesus el que decide coger la llamada en primer lugar enviando un 200 OK al INVITE inicial de alberto (mensaje 14 de la figura 47). En este momento se subscribe al grupo *multicast.*

```
SIP/2.0 200 OK
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
```

```
CSeq: 1 INVITE
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bK605e684eafb07f3101e5bcea20ddc6a2
Record-Route: <sip:mas@192.168.1.3:5060;lr>
Contact: <sip:jesus@192.168.1.12:5075>
To: <sip:grupo3@192.168.1.3>;tag=91691105445
Content-Length: 0
```

13. En cuanto el MAS recibe el 200 OK de jesus, genera un NOTIFY (mensaje 15 de la figura 47) en el que indica entre otras cosas el estado de los usuarios de los participantes en la sesión y que se envía a los que en ese momento se encuentran activos, es decir, alberto y jesus. El contenido del *xml* de este NOTIFY se recoge en detalle en el apéndice H. Podemos observar efectivamente el estado de cada usuario sin mas que examinar el contenido del *xml* que se encuentra encapsulado en el NOTIFY.

```
NOTIFY sip:jesus@192.168.1.12:5075 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bKa89b82a992362ab9d97ae39c42f47e04
CSeq: 4 NOTIFY
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=91691105445
Max-Forwards: 70
Contact: <sip:192.168.1.3:5060;transport=udp>
Content-Type: application/conference-info+xml
Subscription-State: active
Event:  conference
Content-Length: 2041

<?xml version="1.0" encoding="UTF-8"?>
<conference-info entity="sip:8fc5dc77c2def4f5@192.168.1.3:5060"
state="full" version="1" xmlns="urn:ietf:params:xml:ns:conference-
info">
  <users state="full">
    <user state="full" entity="sip:alberto@192.168.1.6:5070">
      <endpoint state="full" entity="sip:alberto@192.168.1.6:5070">
        <joining-method>dialed-in</joining-method>
        <media id="sip:alberto@192.168.1.6:5070">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>connected</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:jesus@192.168.1.12:5075">
      <endpoint state="full" entity="sip:jesus@192.168.1.12:5075">
        <joining-method>dialed-out</joining-method>
        <media id="sip:jesus@192.168.1.12:5075">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
```

```
        <status>connected</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:pablo@192.168.1.9:12000">
      <endpoint state="full" entity="sip:pablo@192.168.1.9:12000">
        <joining-method>dialed-out</joining-method>
        <media id="sip:pablo@192.168.1.9:12000">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>pending</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:ana@192.168.1.2:5080">
      <endpoint state="full" entity="sip:ana@192.168.1.2:5080">
        <joining-method>dialed-out</joining-method>
        <media id="sip:ana@192.168.1.2:5080">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>pending</status>
      </endpoint>
    </user>
  </users>
</conference-info>
```

Dado que jesus es el primero en aceptar la llamada su estado, al igual que el de alberto, es *connected* mientras que ana y pablo se encuentran en *pending.* A medida que los usuarios vayan aceptando la sesión pasarán de un estado *pending* a *connected.*

En este momento los usuarios comienzan a transmitir los paquetes de voz.

14. Este NOTIFY ha de ser contestado con un 200 OK por ambas partes, jesus y alberto. Además alberto ha de mandar un ACK (punto 16 de la figura 47) para terminar con el establecimiento de sesión.

```
ACK sip:8fc5dc77c2def4f5@192.168.1.3:5060 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bK6dbd6ece70cebd87bbf8ddd7824ce9d2
CSeq: 1 ACK
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=badaaf68218d7113
Max-Forwards: 70
Content-Length: 0
```

15. Seguidamente ana acepta el establecimiento de sesión y procede como jesus. El MAS actualiza en estado de cada usuario en el NOTIFY

```
<?xml version="1.0" encoding="UTF-8"?>
<conference-info entity="sip:8fc5dc77c2def4f5@192.168.1.3:5060"
state="full" version="2" xmlns="urn:ietf:params:xml:ns:conference-
info">
  <users state="full">
    <user state="full" entity="sip:alberto@192.168.1.6:5070">
      <endpoint state="full" entity="sip:alberto@192.168.1.6:5070">
        <joining-method>dialed-in</joining-method>
        <media id="sip:alberto@192.168.1.6:5070">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>connected</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:jesus@192.168.1.12:5075">
      <endpoint state="full" entity="sip:jesus@192.168.1.12:5075">
        <joining-method>dialed-out</joining-method>
        <media id="sip:jesus@192.168.1.12:5075">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>connected</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:pablo@192.168.1.9:12000">
      <endpoint state="full" entity="sip:pablo@192.168.1.9:12000">
        <joining-method>dialed-out</joining-method>
        <media id="sip:pablo@192.168.1.9:12000">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>pending</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:ana@192.168.1.2:5080">
      <endpoint state="full" entity="sip:ana@192.168.1.2:5080">
        <joining-method>dialed-out</joining-method>
        <media id="sip:ana@192.168.1.2:5080">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>connected</status>
      </endpoint>
    </user>
  </users>
</conference-info>
```

16. Por último, pablo acepta la llamada y se une a la sesión de voz y su estado se pasa a ser *connected.*

Llegado a este punto podemos concluir que el establecimiento de sesión ha concluido de acuerdo a los requisitos especificados.

Liberación de sesión

Para el caso de la liberación de sesión se obtuvo el siguiente flujo de mensajes:

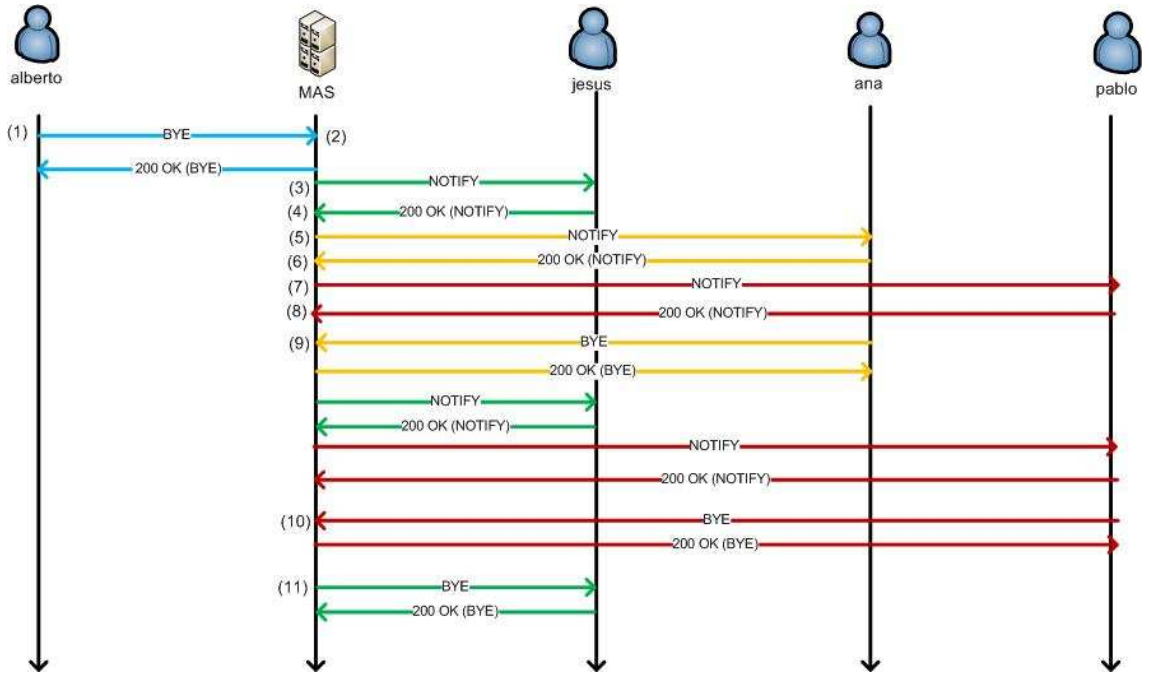

Figura 48 – Liberación de sesión 4 usuarios

1. alberto libera la llamada (punto 1 de la figura 48) enviando un BYE al MAS. El MAS le responde con un 200 OK (punto 2).

```
BYE sip:8fc5dc77c2def4f5@192.168.1.3:5060 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bK53ca6a92b34b01b4d56a873a5a77b647
CSeq: 4 BYE
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=badaaf68218d7113
Max-Forwards: 70
Content-Length: 0

SIP/2.0 200 OK
Via: SIP/2.0/UDP
192.168.1.6:5070;branch=z9hG4bK53ca6a92b34b01b4d56a873a5a77b647
CSeq: 4 BYE
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=badaaf68218d7113
Content-Length: 0
```

2. En este momento el MAS notifica al resto de los usuarios que se encuentran activos (jesus, ana y pablo) que alberto ha abandonado la sesión (puntos 3, 5 y 7 de la figura 48) y actualiza el estado de todos los usuarios, alberto pasa a estado *disconnected* mientras que el resto permanece conectado. Esto se puede comprobar en el mesaje NOTIFY.

```
NOTIFY sip:jesus@192.168.1.12:5075 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bK51fadabe28b24702f6a33d0bc87ae7b7
CSeq: 7 NOTIFY
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=91691105445
Max-Forwards: 70
Contact: <sip:192.168.1.3:5060;transport=udp>
Content-Type: application/conference-info+xml
Subscription-State: active
Event:  conference
Content-Length: 2048

<?xml version="1.0" encoding="UTF-8"?>
<conference-info entity="sip:8fc5dc77c2def4f5@192.168.1.3:5060"
state="full" version="4" xmlns="urn:ietf:params:xml:ns:conference-
info">
  <users state="full">
    <user state="full" entity="sip:alberto@192.168.1.6:5070">
      <endpoint state="full" entity="sip:alberto@192.168.1.6:5070">
        <joining-method>dialed-in</joining-method>
        <media id="sip:alberto@192.168.1.6:5070">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>disconnected</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:jesus@192.168.1.12:5075">
      <endpoint state="full" entity="sip:jesus@192.168.1.12:5075">
        <joining-method>dialed-out</joining-method>
        <media id="sip:jesus@192.168.1.12:5075">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>connected</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:pablo@192.168.1.9:12000">
      <endpoint state="full" entity="sip:pablo@192.168.1.9:12000">
        <joining-method>dialed-out</joining-method>
        <media id="sip:pablo@192.168.1.9:12000">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>connected</status>
      </endpoint>
```

```
    </user>
    <user state="full" entity="sip:ana@192.168.1.2:5080">
      <endpoint state="full" entity="sip:ana@192.168.1.2:5080">
        <joining-method>dialed-out</joining-method>
        <media id="sip:ana@192.168.1.2:5080">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>connected</status>
      </endpoint>
    </user>
  </users>
</conference-info>
```

3. Cada uno de estos NOTIFY es respondido con un 200 OK (puntos 4, 6 y 8 de la figura 48).

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bK51fadabe28b24702f6a33d0bc87ae7b7
CSeq: 7 NOTIFY
Call-ID: 1adc3ebcffe6ae15037afcc00f81bd88@192.168.1.6
From: <sip:alberto@192.168.1.6:5070>;tag=73727875878
To: <sip:grupo3@192.168.1.3>;tag=91691105445
Content-Length: 0
```

4. A continuación ana abandona la sesión (mensaje 9 de la figura 48 y el MAS de nuevo informa de este evento a los usuarios que permanecen conectados enviándoles un NOTIFY.
5. Posteriormente es pablo el que decide desconectarse (punto 10 de la figura 48). En este momento el único usuario que queda activo en la conversación es jesus. El MAS se percata de esto e inmediatamente envía un BYE (mensaje 11 de la figura 48) a jesus para que se desconecte y se de por terminada la llamada.

Con este caso se ha demostrado que tanto el establecimiento como la liberación de sesión funciona acorde a los requisitos establecidos en cuanto a la señalización SIP/SDP.

Caso 2. Liberación de sesión quedando un usuario a la espera de que otro acepte la sesión.

Este caso que se presenta es bastante similar al anterior. El codec negociado será de nuevo el **PCMA/8000**, la dirección ip multicast ofrecida por el MAS la **224.10.10.20** y el puerto el **7890.**

El desarrollo de la llamada será el siguiente: alberto inicia la llamada al grupo3. La llamada será aceptada en primer lugar por jesus y luego por ana mientras que pablo se queda mandando el RINGING a la espera de aceptar la llamada. En un momento determinado ana manda un BYE para abandonar la sesión y posteriormente lo hace jesus. En este momento nos encontramos con alberto que se encuentra activo en la sesión y pablo que de momento no la ha aceptado y permanece mandando el RINGING. Alberto ha de ser alertado de que todavía queda un usuario pendiente de aceptar la llamada y será su decisión quedarse a la espera o abandonar la sesión. Finalmente alberto decide esperar. Pablo acepta la sesión y abre una sesión de voz con alberto. Posteriormente alberto manda un BYE para salir de la sesión y el MAS automáticamente envía otro BYE a pablo ya que es el único usuario que se encuentra conectado.

El flujo de mensajes que se obtuvo en este caso se describe en la figura 49. Obviaremos los mensajes del inicio del establecimiento de sesión y nos centraremos en la llamada en espera.

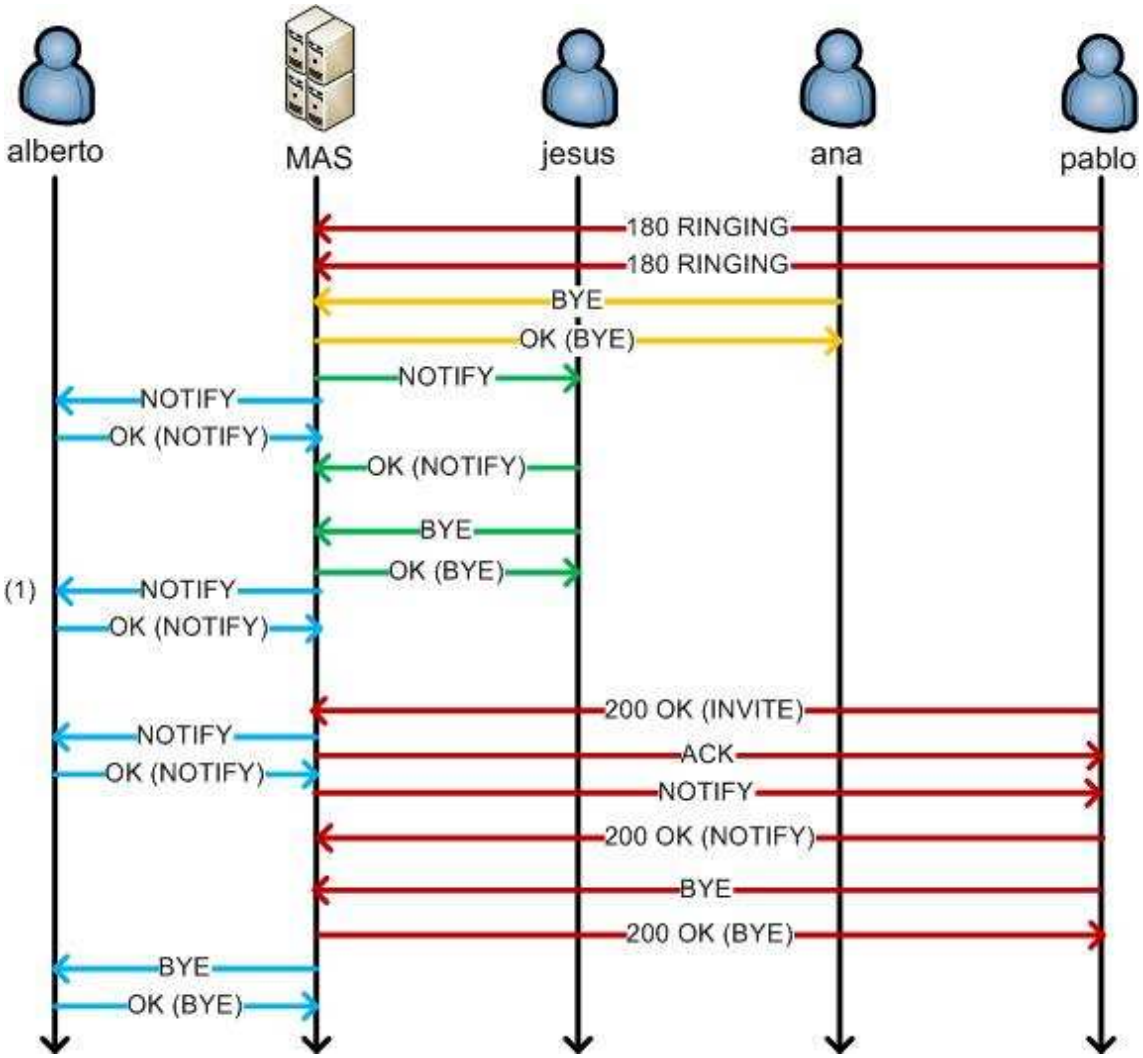

Figura 49 – Liberación de sesión con un usuario enviando RINGING

En el momento que alberto se queda esperando a pablo (punto 1 de la figura 49) se imprime por pantalla el siguiente mensaje

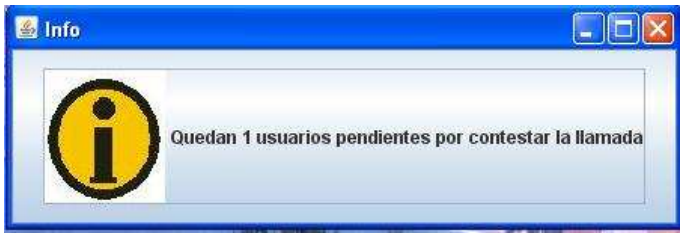

Figura 50 – Usuario pendiente de contestar la llamada

Además podemos comprobar el estado de pablo y del resto de los usuarios en el NOTIFY del punto 1.

```
NOTIFY sip:192.168.1.6:5070;transport=udp SIP/2.0
```

```
Via: SIP/2.0/UDP
192.168.1.3:5060;branch=z9hG4bK9178ffb2a92c078d4efa7aaa1efc8d12
CSeq: 4 NOTIFY
Call-ID: c5926200ff43a0c16ac3fe6d00421b3d@192.168.1.6
From: <sip:grupo3@192.168.1.3>;tag=4a816c88b82d09fe
To: <sip:alberto@192.168.1.6:5070>;tag=101085453249
Max-Forwards: 70
Contact: <sip:192.168.1.3:5060;transport=udp>
Content-Type: application/conference-info+xml
Subscription-State: active
Event:  conference
Content-Length: 2049

<?xml version="1.0" encoding="UTF-8"?>
<conference-info entity="sip:24de3f7f67614eba@192.168.1.3:5060"
state="full" version="4" xmlns="urn:ietf:params:xml:ns:conference-
info">
  <users state="full">
    <user state="full" entity="sip:alberto@192.168.1.6:5070">
      <endpoint state="full" entity="sip:alberto@192.168.1.6:5070">
        <joining-method>dialed-in</joining-method>
        <media id="sip:alberto@192.168.1.6:5070">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>connected</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:jesus@192.168.1.12:5075">
      <endpoint state="full" entity="sip:jesus@192.168.1.12:5075">
        <joining-method>dialed-out</joining-method>
        <media id="sip:jesus@192.168.1.12:5075">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>disconnected</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:pablo@192.168.1.9:12000">
      <endpoint state="full" entity="sip:pablo@192.168.1.9:12000">
        <joining-method>dialed-out</joining-method>
        <media id="sip:pablo@192.168.1.9:12000">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>pending</status>
      </endpoint>
    </user>
    <user state="full" entity="sip:ana@192.168.1.2:5080">
      <endpoint state="full" entity="sip:ana@192.168.1.2:5080">
        <joining-method>dialed-out</joining-method>
        <media id="sip:ana@192.168.1.2:5080">
          <display-text>information field is empty</display-text>
          <type>audio</type>
          <label>label of sdp</label>
        </media>
        <status>disconnected</status>
      </endpoint>
```

```
    </user>
  </users>
</conference-info>
```

Con este caso se ha demostrado la implementación correcta del caso expuesto. En este momento alberto es libre de contestar la llamada. Si la contesta seguirá el procedimiento habitual de establecimiento de sesión explicado en el caso 1, si no es así y alberto decide no esperar y mandar un BYE el MAS le respoderá a alberto con un 200 OK (BYE) y a su vez el MAS le mandará a pablo un BYE ya que será el único usuario activo en la sesión.

Caso 3. Añadir múltiples rutas al mensaje INVITE del iniciador.

Este caso pretende demostrar que el cliente iniciador soporta en envío de la cabecera *Route* con múltiples saltos. Para ello se muestra una captura (figura 51) en la que alberto añade múltiples saltos a la cabecera *Route.*

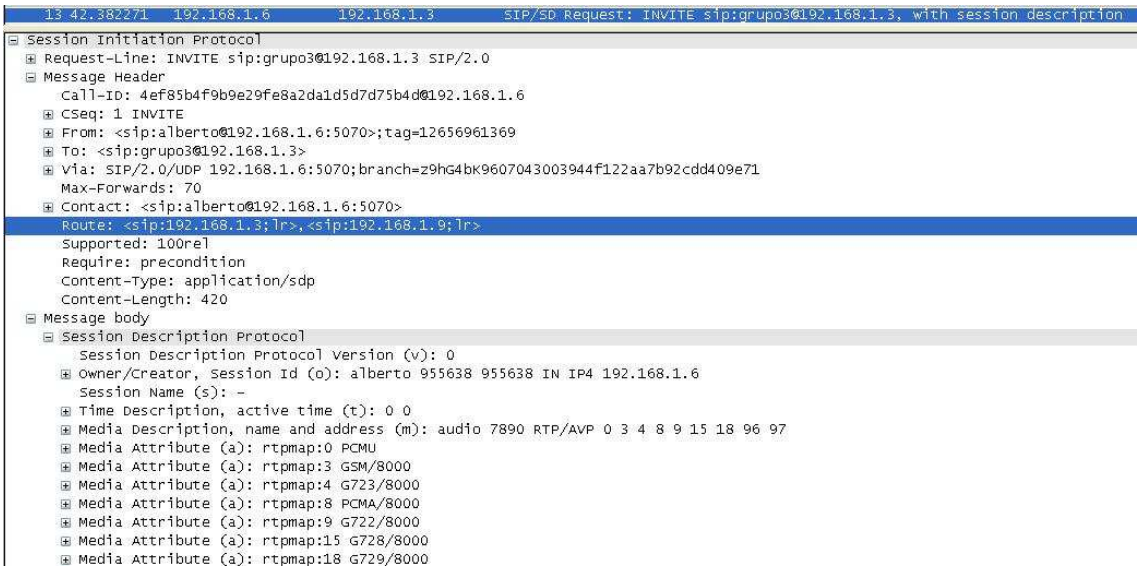

Figura 51 – Múltiples saltos en el campo *Route*

5.1.2 Transmisión multicast de VoIP.

Los siguientes casos de pruebas pretenden demostrar el inicio de la sesión de voz multiusuario mediante el envío de los paquetes de voz de acuerdo a los parámetros en relación a la ip y puerto multicast y codec negociado en el establecimiento de la sesión.

Caso 4. Inicio y finalización de la sesión de voz multiusuario con 3 participantes.

El siguiente caso tiene como objetivo demostrar el desarrollo de una sesión de voz multiusuario una vez que ya se han negociado los codecs de voz, se ha finalizado la reserva de recursos y se conoce la dirección IP y puerto multicast donde se lanzará la aplicación. En el ejemplo que se propone se codificará la voz en formato PCM y se enviará sobre UDP, el puerto donde se recibirán los paquetes de voz será el 7890 y la ip multicast es la 224.0.0.20. Se mostrará también los mensajes *IGMPv2 Report* para la suscripción al grupo multicast así como los de abandono *IGMPv2 Leave Group*.

Concretamente la llamada se desarrolla de la siguiente manera: alberto inicia el establecimiento de sesión con el grupo2 y por tanto llama a jesus y pablo. jesus acepta la llamada (con lo que alberto ya puede hablar con jesus) y seguidamente lo hace pablo. Tras mantener una conversación y pasado un tiempo pablo abandona la sesión, un poco mas tarde lo hace jesus con lo que en este momento solo queda alberto como único participante y también cuelga la llamada.

Seguidamente mostramos el flujo de mensajes IGMP así como algunas muestras de los paquetes UDP de VoIP.

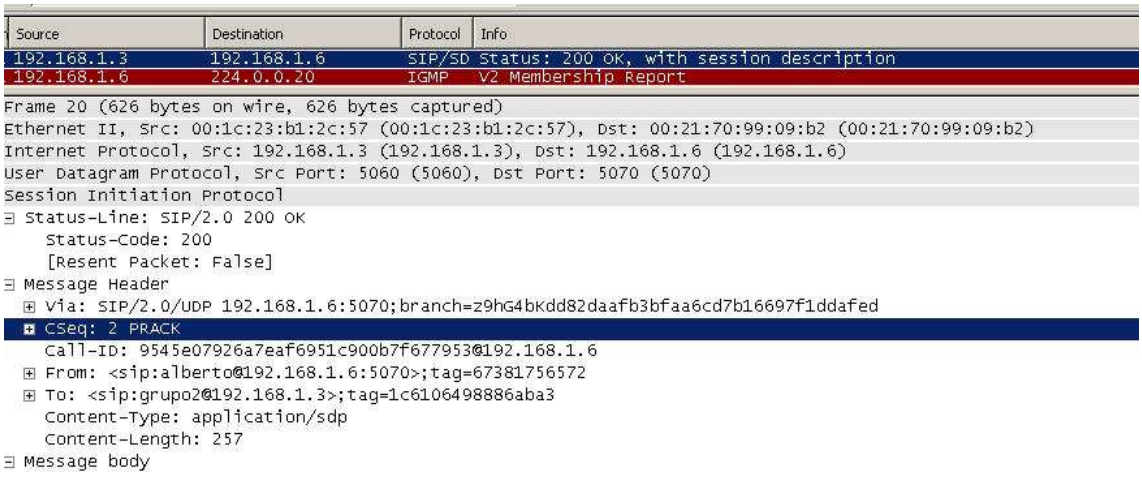

Figura 52 – Suscripción grupo multicast usuario alberto

Podemos observar en la figura 52 que en cuanto alberto recibe el 200 OK del PRACK inmediatamente se subscribe al grupo *multicast* enviando un *IGMP Report* a la dirección de multifusión previamente acordada 224.0.0.20.

En el caso de los destinatarios jesus y pablo, se vincularán al grupo multicast en el momento que envíen el 200 OK (INVITE). Dicha suscripción se muestra en las figuras 53 y 54.

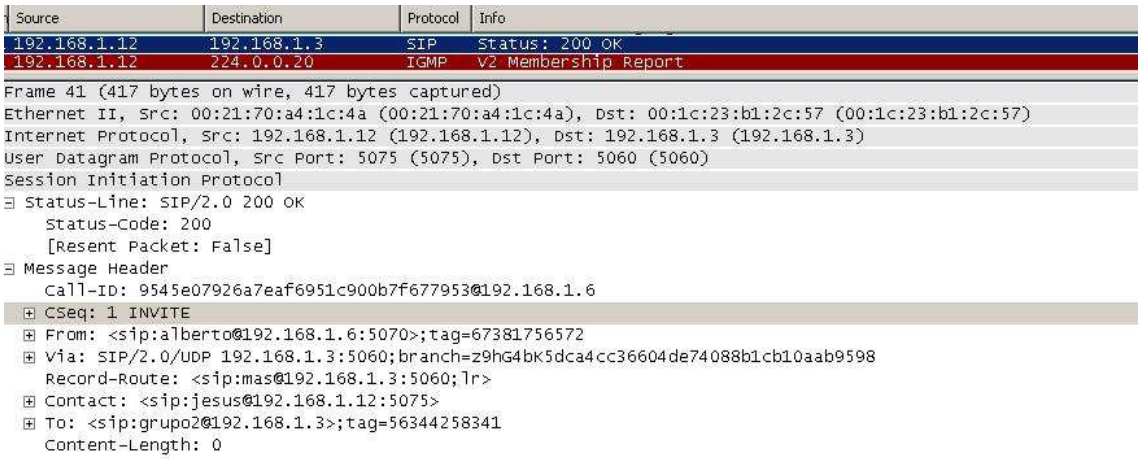

Figura 53 – Suscripción grupo multicast usuario jesus

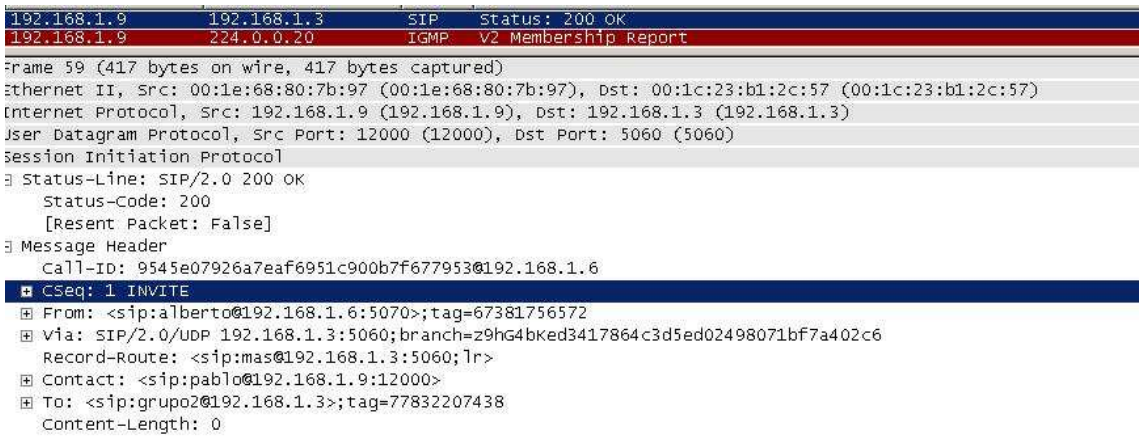

Figura 54 – Suscripción grupo multicast usuario pablo

El MAS ha de mandar un NOTIFY a ambos participantes y estos le han de responder con un OK (NOTIFY). Es justo en este momento cuando ambos usuarios pueden comenzar el envío de los paquetes de VoIP y comenzar la conversación. Del mismo modo actuará pablo cuando acepte la sesión.

La figura 55 muestra a todos los ususarios transmitiendo los paquetes de voz a la dirección ip *multicast* que se negoció en el establecimiento de sesión

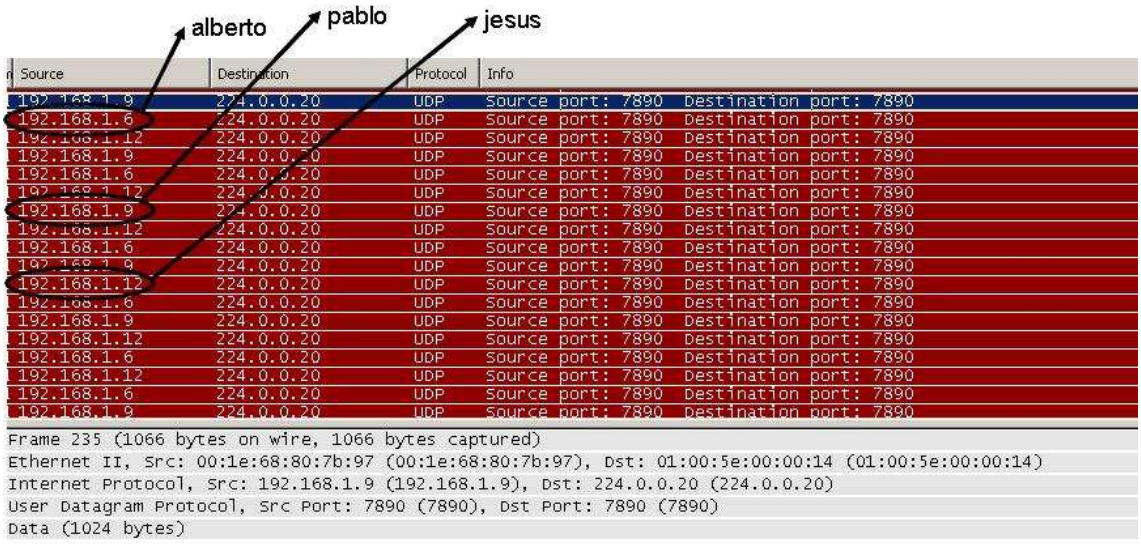

Figura 55 – Usuarios transmitiendo en la sesión de voz multiusuario

A medida que los usuarios abandonan la sesión de voz multiusuario lo hacen también del grupo multicast al que están suscritos. La figura 56 ilustra el proceso en el cual pablo abandona la sesión y sale del grupo *multicast*

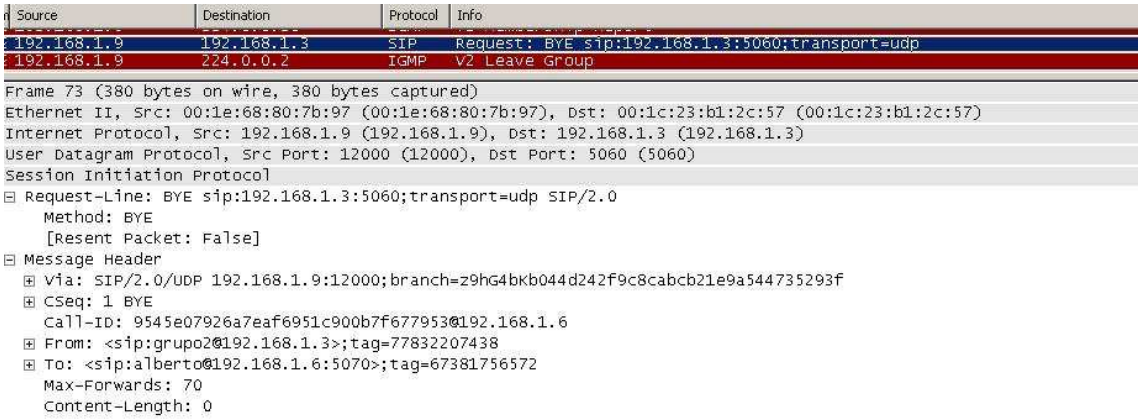

Figura 56 – Usuario pablo abandonando la sesión enviando IGMP *Leave Group*

En el momento que pablo abandona la sesión (enviando un BYE) el *IGMP Leave Group* ha de ir dirigido a la 224.0.0.2 que se refiere a todos los routers de la subred.

A medida que el resto de los participantes abandonan la sesión de voz multiusuario se repite el proceso anterior.

Con este caso se ha demostrado que la aplicación de no solo ser capaz de realizar la negociación SIP/SDP de una manera satisfactoria sino que también implementa los procedimientos necesarios para que los usuarios sean capaces de iniciar y abandonar la sesión de voz acorde a los requisitos establecidos.

Se observó que cuando se a tienen varios usuarios (de 3 en adelante) se produce un deterioro en la calidad de la voz que percibe el receptor. La razón es la implementación que se ha hecho en los clientes de voz. Cada cliente dispone de un *buffer* de *bytes* donde se depositan los paquetes que el cliente recibe por el interfaz de red. Si sólo se tienen dos usuarios en el sistema, el interfaz sólo recibirá datos del otro participante. A medida que aumenta el número de usuarios en la conversación se multiplica por 2 el tráfico recibido en el caso de que el número de participantes sea 3, se multiplicará por 3 si los participantes son 4 y así sucesivamente. Cuando esto ocurre el *buffer* de *bytes* se sobrecarga y se tiran muchos paquetes por lo que la calidad de la voz se degrada de manera notable. Una posible solución que se propone como mejora en este apartado es habilitar un *buffer* de entrada de paquetes por cada usuario participante en la sesión de manera que los paquetes se encolen en el *buffer* correspondiente a cada usuario para evitar sobrecargar un único *buffer*. Esta alternativa plantea una mayor utilización de recursos del cliente (dado que se aumentaría el número de búferes para encolar los paquetes) pero paliaría el efecto que supone la degradación de la voz en la conversación.

6. Conclusiones y líneas futuras

Una vez finalizada la fase de pruebas y validación de resultados es tiempo de extraer las conclusiones del trabajo realizado y proponer una seria de posibles mejoras que sirvan como posibles líneas de trabajo futuras de este proyecto.

En primer lugar hay que afirmar que se han logrado los dos objetivos principales que se plantearon al inicio de este proyecto:

- La implementación de unos clientes SIP que fuesen capaces de generar los mensajes de señalización para el establecimiento y liberación de una sesión de voz multiusuario. En esta fase también se ha tratado la integración con un servidor de aplicaciones multiusuario (MAS) cuya implementación ya estaba desarrollada y que supone un elemento clave en el establecimiento y liberación de la sesión. A estos clientes se les ha dotado de una interfaz gráfica cuyo propósito es la de proveer al usuario final un entorno mas amigable que le permita interactuar de manera mas sencilla con la aplicación.

- Implementación de la aplicación que permite la codificación de la voz utilizando un determinado codec, las suscripción a un determinado grupo multicast y el envío de los paquetes de voz a la ip y puerto multicast que fue negociado. Se logró que todos los participantes pudiesen iniciar y mantener una conversación de voz multiusuario, que se subscribiesen al grupo multicast y que en un determinado momento pudiesen abandonar la sesión dejando de enviar el flujo de paquetes de voz y desvincularse del grupo multicast.

Como posibles mejoras en el plano de señalización que no se han podido llevar a cabo dado el alcance de este proyecto se proponen los siguientes puntos:

- Implementación de casos no contemplados tales como:

    - reserva de recursos tanto en el usuario inicador o llamante como el usuario destinatario o llamado.
    - que uno de los usuarios llamados no acepte el establecimiento de sesión y rechace la llamada.

- que uno de los usuarios llamados esté comunicando.
- posibilidad de hacer transferencia de llamadas.
- posibilidad de poner llamadas en espera.
- estudio de los casos de temporización relacionados con el abandono del establecimiento de llamada.
- mejora de la interfaz gráfica de manera que sea posible configurar los campos SDP en ella en lugar de hacerlo pasando un fichero de texto a la aplicación.

Otras posibles mejoras que se proponen en el apartado relacionado al plano de usuario son:

- Implementación de protocolos de encaminamientos multicast de manera que sea posible la multidifusión de los paquetes de voz en subredes y dominios de difusión distintos.
- Ampliación de los codecs de voz de manera que sea posible hacer la codificación en otros formatos.

A modo general se propone dotar a la aplicación de funcionalidades nuevas como pueden ser:

- Envío conjunto de audio, vídeo y texto al resto de ususarios a través de multidifusión IP.
- Transferencia y compartición de ficheros entre todos los participantes de la sesión.

7. Referencias

[1] Mikka Poikselka, Georg Mayer, Hisham Khartabil, Aki Niemi.The IMS: IP Multimedia Concepts and Services in the Mobile Domain.

[2] Allan B. Johnson. SIP Understanding the Session Initiation Protocol.

[3] Draft ETSI TS 187 006 V0.1.0. Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); NGN Security Countermeasures (Stage 2). Work in progress.

[4] Lucent IMS and RACF Overview. How IMS and RACF provide value to Service Providers

[5] IETF “SIP: Session Initiation Protocol” RFC 3261. J. Rosenberg dynamicsoft , H. Schulzrinne Columbia U., G. Camarillo Ericsson, A. Johnston WorldCom, J. Peterson, Neustar R. Sparks dynamicsoft, M. Handley ICIR, E. Schooler AT&T. June 2002

[6] IETF “Session Initiation Protocol (SIP)-Specific Event Notification” RFC 3265. A. B. Roach dynamicsoft. June 2002

[7] IETF A SIP Event Sub-Package for Watcher Information. Rosenberg dynamicsoft

[8] IETF “Integration of Resource Management and Session Initiation Protocol (SIP)” RFC 3312. G. Camarillo, Ed. Ericsson, W. Marshall, Ed. AT&T, J. Rosenberg dynamicsoft. October 2002

[9] IETF “Reliability of Provisional Responses in the Session Initiation Protocol (SIP)” RFC 3262. J. Rosenberg dynamicsoft, H. Schulzrinne Columbia U. June 2002

[10] IETF “RTP Profile for Audio and Video Conferences with Minimal Control” RFC 3551. H. Schulzrinne Columbia University, S. Casner Packet Design. July 2003

[11] IMS signalling for multiparty services based on network level multicast. Ivan Vidal, Ignacio Soto, Francisco Valera, Jaime Garcia, Arturo Azcorra 3rd EURO-NGI Conference on Next Generation Internet Networks.May 2007. Trondheim, Norway.

[12] Multiparty Services in the IP Multimedia Subsystem Iván Vidal, Ignacio Soto, Francisco Valera, Jaime García and Arturo Azcorra. IP Multimedia Subsystem (IMS) Handbook. Chapter 16, pp. 361-380. 2008. CRC Pr I Llc Books. Ed. Ilyas

[13] Iván Vidal Fernández. Configuración de Transporte Multicast IP con Calidad de Servicio en Arquitecturas de Red con Plano de Control IMS. Tesis Doctoral, Universidad Carlos III de Madrid, Junio 2008.

[14] IETF “Host Extensions for IP Multicasting” RFC 1112. S. Deering Stanford University. August 1989

[15] IETF “Internet Group Management Protocol, Version 2” RFC 2236. W. Fenner Xerox PARC. November 1997

[16] IETF “Internet Group Management Protocol, Version 3” RFC 3376

[17] IETF “IANA Guidelines for IPv4 Multicast Address Assignments” RFC 3171. B. Cain Cereva Networks, S. Deering I. Kouvelas Cisco Systems, B. Fenner AT&T Labs, A. Thyagarajan Ericsson. October 2002

[18] IETF “SDP: Session Description Protocol” RFC RFC 4566. M. Handley UCL, V. Jacobson Packet Design, C. Perkins University of Glasgow. July 2006

[19] Third Generation Partnership Project www.3gpp.org (última visita: 01-02-09)

Anexo A. Planificación del proyecto y presupuesto.

En el presente anexo se pretende describir la planificación del proyecto que abarca desde su concepción inicial hasta la entrega final del mismo. Además se incluye un presupuesto de los costes totales de la implementación y el desarrollo del proyecto.

A.1 Plan de trabajo

El plan de trabajo que se ha seguido se divide en varias etapas cuyo desarrollo se puede observar en la figura 57. En primer lugar se tuvo una fase de documentación para familiarizarse con la arquitectura del IMS y con los protocolos que intervienen en la solución, en especial se estudió de manera pormenorizada el funcionamiento del protocolo SIP/SDP. También se estudió el mecanismo de multidifusión IP así como se hizo una revisión de los codecs de voz existentes prestando especial atención a los que se utilizan de manera más común.

Una vez terminada esta fase se procedió al diseño de la solución en base a unos requisitos previamente definidos. Al término de esta fase, se llevó a cabo a la implementación del sistema y posteriormente se diseñó un plan de pruebas para validar si el sistema se comportaba de manera acorde a los requisitos definidos y una vez finalizada este proceso se hizo una evaluación de las pruebas y se extrajeron unas conclusiones.

Por último se elaboró la presente memoria con el objetivo de recoger todo lo explicado anteriormente.

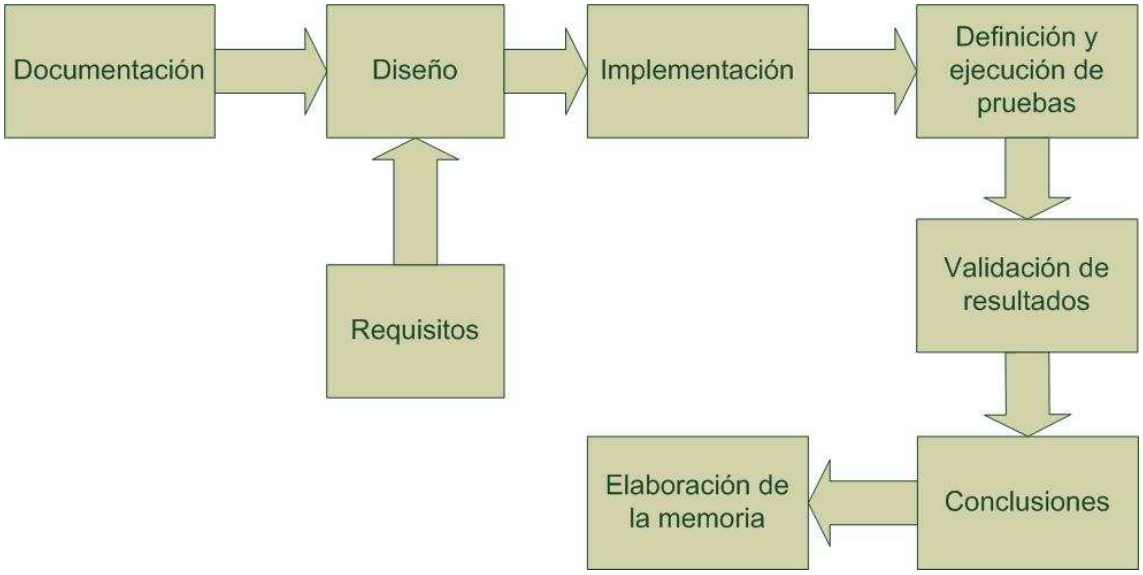

Figura 57 – Fases del desarrollo del proyecto

A continuación definiremos de manera más detallada los procesos descritos en la figura 57 y se proporcionará la carga de trabajo de cada proceso en términos de horas.

Fase 1. Documentación.

En esta fase se procedió a las siguientes tareas:

i. Estudio de la arquitectura de referencia IMS. (64 horas)
ii. Estudio de los protocolos SIP/SDP. (40 horas)
iii. Estudio de los procedimientos de multidifusión IP (16 horas)
iv. Estudio de los codecs de voz existentes (8 horas)
v. Estudio de la plataforma de referencia (JAIN SIP, JAIN SDP y las librerías javax.sound) para la implementación de la solución (16 horas)

Total ➔ 144 horas

Fase 2. Diseño y requisitos.

En esta fase se aborda la identificación de requisitos del sistema y el diseño de la solución a implementar.

i. Analizar y comprender los requisitos del sistema a implementar. (40 horas)

ii. Diseñar una solución que sea capaz de cumplir con los requisitos analizados. (40 horas)

Total ➔ 80 horas

Fase 3. Implementación.

Aquí se tratará el desarrollo de los diferentes módulos que componen la aplicación.

i. Módulo de señalización. Implementación de la parte encargada del establecimiento de sesión (160 horas)
ii. Módulo del plano de datos. Implementación de la parte cuya misión es la del envío de los paquetes de VoIP multicast. (56horas)
iii. Interfaz gráfica. Desarrollo de la interfaz gráfica tanto para el cliente llamante como el llamado. (40 horas)
iv. Integración de todos los módulos descritos anteriormente. (8 horas)

Total ➔ 264 horas

Fase 4. Definición y ejecución de pruebas.

i. Definición de un plan de pruebas para validar la solución (24 horas).
ii. Ejecución del plan de pruebas diseñado (24 horas)

Total ➔ 48 horas

Fase 4. Validación de resultados.

i. Análisis de los resultados obtenidos mediante los logs y trazas recogidos en la fase 4 (16 horas)
ii. Comprobación de si los resultados obtenidos cumplen los requisitos con los que se diseño el sistema. (16 horas).

Total ➔ 32 horas

Fase 5. Conclusiones.

i. Extracción de conlcusiones a la vista de los resultados obtenidos en la fase 4 y líneas futuras (8 horas).

Total ➔ 8 horas

Fase 6. Elaboración de la memoria.

i. Redacción de la presente memoria y revisión de los contenidos (160 horas)

Total ➔ 160 horas

A.2 Presupuesto

El desarrollo del proyecto ha sido realizado por un Ingeniero Superior de Telecomunicación. El presupuesto se calculará acorde al número total de horas empleadas para la elaboración del proyecto.

Fase	Horas
1 – Documentación	**144**
2 – Diseño y requisitos	**80**
3 – Implementación	**264**
4 – Validación de resultados	**32**
5 – Conclusiones	**8**
6 – Elaboración de la memoria	**160**
TOTAL	**688**

Tabla 10 – Horas empleadas

Suponiendo que el sueldo de un ingeniero es de 78 euros/hora (de acuerdo con los últimos baremos obtenidos del COIT – Colegio Oficial de Ingenieros de Telecomunicación) tenemos un presupuesto de la mano de obra de 688 x 78 = **53664 €**

Para calcular el presupuesto total, hay que tener en cuenta también los gastos generales, amortización, las cargas fiscales y jurídicas y gastos de instalación, que se valoran en el 15% del presupuesto de ejecución material.

Costes	Valor
Mano de obra	**53664 €**
Costes generales (15%)	**8049,6 €**
I.V.A (16%)	**9874,16 €**
TOTAL	**71587,76**

Tabla 11 – Presupuesto final

Por tanto, el presupuesto total del proyecto asciende a SETENTA Y UN MIL QUINIENTOS OCHENTA Y SIETE EUROS CON SETENTA Y SEIS CENTIMOS CÉNTIMOS.

Anexo B. Manual de usuario

En este anexo se explica cómo utilizar la aplicación desarrollada.

- Arrancar la aplicación. Será necesario tener instalado una versión de JAVA 1.5 o superior. Además será necesario tener las siguientes librerias .jar en el directorio de ejecución : *log4j-1.2.8.jar, concurrent.jar, JainSipApi1.2.jar, nist-sdp-1.0.jar, JainSipRi1.2.jar*

  En primer lugar ejecutar el siguiente comando para compilar la aplicación y que se generen los correspondientes archivos .class:

  javac -cp .;log4j-1.2.8.jar;concurrent.jar;JainSipApi1.2.jar;nist-sdp-1.0.jar;JainSipRi1.2.jar ClienteLlamante.java ClienteLlamado.java InterfazLlamante.java InterfazLlamado.java MessageFactoryImpl.java MessageProcessor.java AplicacionAudio.java LlamadaEntrante.java LlamadaAceptada.java LlamadaEnEspera.java Chat.java

  A continuación lanzar la aplicación con el siguiente comando:

  java -classpath .;log4j-1.2.8.jar;concurrent.jar;JainSipApi1.2.jar;nist-sdp-1.0.jar;JainSipRi1.2.jar AplicacionAudio

- Selección del tipo de cliente. Una vez ejecutado el comando anterior se le preguntará al usuario que tipo de cliente quiere ejecutar: llamante o llamado

Figura 58 – Ejecución de la aplicación: llamante o llamado

Independientemente del tipo de cliente seleccionado a continuación se le preguntará en qué modo desea ejecutar la aplicación: modo *debug* que implica que se imprimirán el contenido de todos los mensajes recibidos por pantalla (para ello teclear 1) o modo no *debug* en el que simplemente se notificará por pantalla que se ha recibido un mensaje (Request o Response) pero no se imprimirá todo el contenido del mismo (para ello teclear 0).

Figura 59 – Ejecución de la aplicación: debug activado/desactivado

Dependiendo del tipo de cliente seleccionado aparecerá una ventana distinta

- Cliente llamante. Los detalles del funcionamiento de esta ventana así las ventanas asociadas a este modo fueron ampliamente explicados en el apartado 4.1.1
- Cliente llamado. Los detalles del funcionamiento de esta ventana así las ventanas asociadas a este modo fueron ampliamente explicados en el apartado 4.2.1

Anexo C. Elementos del IMS

CSCF

El CSCF (*Call Session Control Function*) se puede considerar como el núcleo de IMS. Realiza funciones de control de sesión y de enrutado. A su vez el CSCF se puede descomponer a su vez en tres entidades:

- *Proxy-CSCF*. Se trata del primer punto de contacto del usuario dentro del IMS. Todo el tráfico de señalización de usuario atraviesa el P-CSCF. EL P-CSCF se encarga de procesar los mensajes que envía el usuario validando la petición y reenviándola a los destinatarios. También se encarga de recibir las respuestas y procesarlas. Además, el P-CSCF puede comportarse como *User Agent* (UA). Un UA es un dispositivo final que soporta la señalización SIP y capaz de establecer sesiones multimedia. En la mayoría de los casos un UA establece sesiones con otro UA que puede ser un usuario final pero también es capaz de negociar una sesión con un usuario intermedio como una SIP *Gateway*. El UA tiene que mantener el estado de las llamadas que inicia o que participa. El estado de la llamada queda definido por un conjunto mínimo de parámetros locales y remotos tales como *Call-ID*, las etiquetas locales y remotas *Cseq* junto con el conjunto de rutas y cualquier información necesario para el medio. Toda esta información es usada por el UA para guardar información relativa al diálogo y para la fiabilidad de la sesión. Los mensajes que recibe el UA procedentes de un diálogo desconocido son descartados o ignorados para salvaguardar la seguridad de la sesión.
  Un UA se compone de dos agentes:
  - UAC (*User Agent Client*): agente que genera las peticiones.
  - UAS (*User Agent Server*): agente que genera las respuestas.

  Durante una sesión el UA actúa conjuntamente como UAC y UAS. Además el UA ha de soportar el lenguaje SDP. Por último, el UA tiene que entender cualquier extensión listada en la etiqueta *Require* y puede anunciar sus capacidades y las características mediante las etiquetas *Allow, Supported* y *Accept* [2]. Las funciones que desempeña el P-CSCF son [1]:

  - Reenviar el SIP REGISTER al I-CSCF basándose en el dominio dentro de la petición.

- Reenviar las peticiones y respuestas que recibe del UE hacia el *Serving*-CSCF (S-CSCF).
- Reenviar las peticiones y respuestas hacia el UE.
- Enviar información de *accounting*.
- Proveer mecanismos de integridad en los paquetes de señalización SIP y mantener asociaciones de seguridad entre el UE y el P-CSCF. La integridad se consigue utilizando el protocolo *IPsec*.
- Compresión y descompresión de los mensajes SIP que envía el UE.
- Realizar políticas de control. El P-CSCF es capaz de examinar el contenido del campo SDP del los paquetes SIP para comprobar si contienen información relativa al trafico que se va a enviar (audio, vídeo, etc) y el tipo de codificación empleada.
- Mantener los temporizadores de sesión. Es capaz de liberar los recursos cuando cualquier temporizador expira.
- Interactuar con el PDF (*Policy Decision Function*). El PDF se encarga de la implementación del SBPL (*Service Based Local Policy*) cuyas funciones son:
  - Mantener información de la sesión establecida así como de los parámetros relativos a la información enviada durante la sesión (direcciones IP, puertos utilizados, ancho de banda utilizado…)
  - Participar en labores de autenticación y de concesión, denegación y/o modificación de *bearers*.
  - Pasar información de tarificación al P-CSCF.
- Interactuar con el A-RACF (*Access Resource and Admission Control Function*) cuya función es la de verificar dinámicamente la disponibilidad de recursos [4].
- Interactuar con el NASS (*Network Attachment Sub-System*) que se encarga de realizar labores de autenticación en el registro de usuario.

- *Interrogating*-CSCF. Su función es la determinar dentro de la red del operador contra que S-CSCF el usuario se ha de registrar. Para ello, cuando el I-CSCF recibe la petición de registro del P-CSCF manda una consulta al HSS para determinar si el usuario esta permitido registrarse en la red del operador. En caso afirmativo el HSS devuelve el S-CSCF asignado al usuario. El I-CSCF también interviene en el proceso de establecimiento de llamada. De esta manera cuando el iniciador de la llamada desea establecer una llamada manda un INVITE indicando en la URI destino el destinatario de la llamada.

Este INVITE se manda al P-CSCF de la red del local y el P-CSCF lo manda al S-CSCF de la red local. Es entonces cuando el S-CSCF extrae el dominio y hace una petición DNS para saber a que I-CSCF se lo tiene que mandar. Cuando el INVITE entra por el I-CSCF de la red visitada, interroga al HSS para saber con que dirección se ha registrado el destino y esta información se envía al S-CSCF de la red visitada. Además el I-CSCF puede implementar la funcionalidad llamada *Topology Hiding Inter-network Gateway* (THIG). THIG puede utilizarse para ocultar la configuración de la topología y capacidades. de la red hacia operadores externos.

- *Serving*-CSCF. Se puede considerar como el cerebro del IMS y se localiza en la red del operador. Realiza funciones de control de sesión y registro del usuario. Mientras que un usuario se encuentra activo en la sesión el S-CSCF mantiene el estado de la sesión e interactúa con otras plataformas para llevar por ejemplo el control de la tarificación. Las principales funciones que realiza el S-CSCF son:
    - Participar en el proceso de registro del usuario. El S-CSCF conoce que P-CSCF el usuario esta utilizando como punto de entrada al IMS.
    - Autentica al usuario mediante el procedimiento *Authentication and Key Agreement* (AKA). AKA define una autenticación mutua ente usuario y red.
    - Recuperar información de usuario del HSS en los procesos de registro y establecimiento de llamada.
    - Es el punto de contacto con el *Breakout Gateway Control Function* (BGCF) cuya misión es decidir si la llamada ha de ser enrutada hacia el dominio de circuitos CS. La decisión puede ser incluso el hacer el enrutado hacia la propia red del operador. Si es así el BGCF selecciona una *Media Gateway Control Function* (MGCF) para que se haga cargo de la sesión. Si por el contrario, el *breakout* se ha de hacer hacia otra red, entonces el BGCF contacta con su homólogo en la red visitada para que se haga cargo de la llamada. El BGCF además soporta mecanismos de envío de información de tarificación.
    - Interactuar con plataformas de servicios. El S-CSCF tiene la capacidad de decidir si ha de enrutar la petición hacia un servidor de aplicaciones (AS) y en ese caso cual es el servidor al cual ha de mandar la petición.
    - Translación de números telefónicos.
    - Control de temporizadores y capacidad de registrar usuarios si algunos de estos temporizadores expiran.

- Ejecución de políticas de control sobre el tráfico que se va enviar entre los usuarios. El S-CSCF es capaz de examinar el contenido del cuerpo SDP dentro de la petición SIP para ver que tipo de información multimedia se va a intercambiar (audio, voz,..) y la que codecs se van a utilizar. Si los codecs ofrecidos no cumplen la política del operador, el CSCF puede rechazar el establecimiento de la sesión mandando un mensaje 488 a los usuarios.
- Manda información de tarificación a las plataformas correspondientes.

MRFC

El MRFC (*Multimedia Resource Function Controller*) soporta los servicios de *bearers* tales como conferencias, notificaciones a usuarios o *transcoding*. El MRFC interpreta la señalización SIP recibida del S-CSCF y utiliza el protocolo MEGACO (*Media Gateway Control Protocol*) para controlar el MRFP (*Multimedia Resource Funtion Processor*) cuya misión principal es la de mezclar flujos de diferentes conexiones de distintos usuarios, generar ciertos flujos multimedia (notificaciones) y procesar los flujos multimedia que recibe.

MGCF

El MGCF (*Media Gateway Control Function*) es la pasarela que permite la comunicación entre usuarios en el IMS y el CS. Maneja la señalización entre ambos dominios y realiza la conversión de la señalización del dominio CS a la señalización SIP y viceversa y se conecta a la *Signalling Gateway* (SGW) que realiza la interconexión física entre las dos redes de señalización.

IMS-MGW

El IMS-MGW (*IMS Multimedia Gateway*) proporciona la conexión en el plano de usuario entre el dominio CS (PSTN, GSM) y el IMS. Es el punto de terminación de los canales *bearers* procedentes del CS y del los flujos multimedia procedente de redes IP (flujos RTP) o de redes ATM (AAL2/ATM). Realiza el *transcoding* entre ambos dominios y además es capaz de enviar tonos y anuncios al dominio CS. Está controlada por la MGCF.

Anexo D. Fiabilidad de SIP

Cuando SIP utiliza como protocolo de transporte TCP o TLS, los mecanismos de fiabilidad no se usan ya que TCP o TLS son protocolos fiables en si y retransmiten el paquete si este se pierde o informan si el servidor no está disponible.

En el caso de que el protocolo de transporte sea UDP, siempre existe la posibilidad de que los paquetes se pierdan o que se reciban fuera de secuencia. UDP sólo garantiza que el datagrama no sea erróneo pero no soporta mecanismos de retransmisión. El UAS valida y parsea las peticiones SIP para asegurar que el UAC haya creado correctamente la petición, haya añadido todas las cabeceras que son obligatorias y no haya cometido algún error de sintaxis. Los mecanismos de fiabilidad en SIP incluyen:

- Retransmisión de temporizadores.
- Incremento de los números de secuencia en la cabecera *CSeq*.
- Asentimientos positivos.

De este modo el temporizador T1 comienza cuando se genera o se recibe una petición en un servidor *proxy stateful*. Si pasado este temporizador no se recibe una respuesta a esta petición, la petición es reenviada. Si se recibe una respuesta provisional (1xx), el UAC o el servidor *proxy stateful* ignora el temporizador T1 y comienza un nuevo temporizador T2. Ninguna retransmisión se envía hasta que T2 expira.

Una vez que se retransmite la petición, el temporizador se dobla hasta que alcanza un valor de T2. Después de eso, las retransmisiones restantes ocurren a intervalos de T2. Un servidor *proxy stateful* que recibe una retransmisión de una petición descarta la retransmisión y continúa su esquema de retransmisión basado en sus propios temporizadores. Típicamente, reenviará la última respuesta provisional.

Anexo E. Solicitudes SIP

INVITE

En el INVITE se ha de especificar contra quien queremos establecer la sesión. Esto se hace incluyendo una SIP URI (*Request URI*) en la línea inicial de la solicitud INVITE.

En el momento que se genera el INVITE se crea el *Call-ID* que es un único identificador global que se es utilizado mientras la llamada sigue activa. Esta formado por un identificador aleatorio que consiste en un número criptográficamente aleatorio que se genera de manera segura para evitar ser suplantado por terceros. A continuación se pone el dominio que puede ser bien un nombre o una dirección IP en la que esta localizado el usuario que inicia la llamada.

Además dentro de las cabeceras del mensaje podemos encontrar el campo *Cseq* cuyo valor ha de ser un entero y actúa como un contador que se incrementa cada vez que se genera una nueva petición perteneciente al diálogo que se ha establecido.

También están presentes las cabeceras *To* y *From* que representan las direcciones local (la que correspondiente a la que genera el INVITE) y remota (a la que va dirigida la petición). A ambas cabeceras se la añade una etiqueta *tag*.

Una etiqueta es un número aleatorio de al menos 32 bits que identifica de manera única al diálogo. La cabecera *To* del mensaje INVITE que genera el iniciador de la llamada no contiene la etiqueta *tag* pero si la contiene la cabecera *From*. El resto de peticiones y respuestas, salvo la respuesta temporal 100 TRYING, deberán incluir las dos etiquetas *tag* en las cabeceras *From* y *To*. La etiqueta *tag* de la cabecera *To* se crea en el momento en el que el UAS responde al INVITE. Las etiquetas nunca se copian a lo largo de la llamada. Cualquier respuesta generada por cualquier *proxy* intermedio contendrá una etiqueta propia generada por el *proxy.*

La etiqueta *tag* de la cabecera *To* la fija el UAS o destinatario de la llamada en el mensaje de respuesta al INVITE. Una vez creada esta etiqueta el diálogo queda unívocamente definido por el *Call-ID* y las etiquetas de las cabeceras *To* y *From*.

Esto permite que el UAC o iniciador de la llamada pueda establecer con un solo INVITE múltiples diálogos en el caso de que el INVITE sea clonado y enviado a múltiples UAS. De esta manera cada UAS responde al INVITE con un valor distinto de la etiqueta *tag* en el cabecera *To*. El iniciador entonces es capaz de identificar de quien proviene la respuesta al mensaje INVITE que generó. De esta manera los diálogos tendrán la misma cabecera *From*, *Call-ID* y *CSeq* pero distinto *To*.

La cabecera *Via* se añade al INVITE. En ella el UA graba su propia dirección para asegurar que la respuesta a la petición INVITE sigua exactamente el mismo camino que el INVITE. Se usa principalmente para enrutar los mensajes SIP. Además aparece otro parámetro llamado *branch* cuyo valor se calcula como el resultado de una función hash de los campos *Request-URI, To, From, Call-ID* y *CSeq*. Opcionalmente se pueden insertar otros parámetros tales como *maddr* y *ttl* para el soporte multicast.

La cabecera *Contact* especifica la dirección de contacto para la respuesta inmediata, es decir, indica a que URI el iniciador espera recibir las respuestas referidas al diálogo.

Para evitar que los paquetes SIP entren en un bucle, se define la cabecera *Max-Forwards* que indica el máximo número de saltos que el paquete puede alcanzar. Por cada salto este contador se decrementa en una unidad de manera que cuando un *proxy* recibe un paquete cuyo valor de *Max-Forwards* es cero lo descarta.

En el mensaje INVITE también se puede encontrar la cabecera *Route* cuya misión es dotar de información de enrutado al paquete INVITE. En esta cabecera define una o varias URIs a través de las cuales en paquete puede ser enrutado. La RFC 3261 introduce dos tipos de enrutado: enrutado estricto (*strict*) y enrutado libre (*loose*).

El enrutado *strict* puede tomar la primera URI de la lista y reescribir la *Request URI* para luego enviarla a esa URI. En este caso el camino seguido por el paquete ha de ser exactamente el especificado en la lista de *Route* de manera que se reescribe en cada salto.

En el enrutado libre, el *proxy* que recibe el paquete no reescribe la *Request UR*I y reenvía la solicitud a la primera URI definida en la lista de *Route* o puede reenviarla a otro elemento que soporte enrutado libre y más tarde enrutar la solicitud basándose en la lista de URI definidas en la cabecera *Route*. El UAC o *proxy* puede indicar si el

siguiente elemento soporta enrutado libre añadiendo el parámetro *lr* en la cabecera *Route.*

Por último, la cabecera *Content-Type* especifica el tipo de información que se envía en el cuerpo del mensaje SIP. Esta información puede ser información de aplicación, información útil en formato *xml*, en formato texto, etc...

Además de las cabeceras descritas, el INVITE puede llevar otra serie de cabeceras con otros propósitos que no detallaremos ya que no nos aportarán ningún tipo de información en el desarrollo del presente proyecto.

Los formatos de las cabeceras descritas así como varios ejemplos se especifican en la tabla mostrada a continuación.

	Formato cabecera	Ejemplo
*** Request URI**	**sip:[usuario][:clave(opcional)]@dominio[:puerto(opcional)][SIP version]**	**sip:usuario1@192.168.1.3 SIP/2.0**
*** Call ID**	**[numero_aleatorio]@dominio**	**367dc933584475c4c01ca500ead6ef9a@192.168.1.6**
*** CSeq**	**[numero_secuencia] INVITE**	**CSeq: 56 INVITE**
*** From**	**sip:[usuario]@dominio[;tag]**	**<sip:user1@192.168.1.6:5090>;tag=8d1e42771585f4c9**
*** To**	**sip:[usuario]@dominio[;tag]**	**<sip:user2@192.168.1.3>;tag=6efc23889a3fe0ef**
*** Via**	**[SIP version][]protocolo_transporte][dirección][:puerto][;parámetros]**	**SIP/2.0/UDP 192.168.1.6:5090;branch=z9hG4bK41a737ffa6baecd1e48e0ef55838f515**
*** Max-Forwards**	**[numero]**	**70**
*** Contact**	**sip:[usuario]@dominio[:puerto]**	**<sip:user1@192.168.1.6:5090>**
Route	**sip:[dirección1][;parametros] ,[dirección2][;parametros]….**	**<sip:192.168.1.3;lr>,<sip:172.16.0.5;lr>**
Content-Type	**Type/subtype**	**application/sdp**

Tabla 12 – Campos obligatorios en la solicitud INVITE

(*) Cabeceras obligatorias

REGISTER

Este método se utiliza para que el UAC notifique a las entidades SIP sobre la URI de su *Contact* actual (dirección IP).También puede añadir o borrar esta asociación.

CANCEL

El método CANCEL se utiliza para terminar búsquedas pendientes o intentos de establecimiento de llamadas. Puede ser generada tanto por UA como por *proxies*. Las cabeceras que han de estar de manera obligatoria son: *Call-ID, CSeq, From, To, Via,* y *Max-Forwards*

OPTIONS

El mensaje OPTIONS se utiliza para preguntar a cualquier UAC o UAS sobre las capacidades que soportan y saber su disponibilidad. La respuesta a esta solicitud es una lista con las capacidades que soporta el UAC o UAS. Un *proxy* nunca genera una solicitud OPTIONS. Ele tipo de respuesta que recibe es similar a la que se recibe cuando se manda un INVITE.

REFER

El método REFER se utiliza por los UA para solicitar a otro UA el acceso a un recurso localizado en una URI o URL.

SUBSCRIBE

El mensaje SUBSCRIBE se usa por un UA para establecer una subscripción a un cierto servicio para recibir notificaciones (a través del método NOTIFY) sobre eventos en particular. Una vez que el usuario ha completado la suscripción se establece un diálogo entre UAC y UAS.

La solicitud SUBSCRIBE contiene la cabecera *Expires* cuya función es fijar la duración de la suscripción. Una vez transcurrido este periodo, la suscripción caduca automáticamente.

El tipo de evento al que el usuario se suscribe se especifica en la cabecera *Event.* Cada evento define una serie de parámetros asociados a dicho evento.

El servidor debe indicar que tipo de eventos soporta. Esta información se almacena en la cabecera *Allow-Events.*

MESSAGE

El método MESSAGE se utiliza para transportar mensajes instantáneos (IM) utilizando SIP. Un mensaje instantáneo consiste en un mensaje corto intercambiado entre los participantes de una sesión. Este mensaje puede ser enviado tanto dentro como fuera de diálogo y no crea un diálogo adicional entre los participantes en la sesión.

El contenido del mensaje se transporta dentro del cuerpo del MESSAGE como un adjunto de tipo MIME. Cualquier UA que sea capaz de soportar el método MESSAGE ha de soportar también el formato en texto plano y opcionalmente otros formatos tales como HTML, cpim (*Common Presence and Instant Messaging*), etc...

Si el mensaje se recibe de manera correcta, el receptor manda un mensaje 200 OK de respuesta al originador del mensaje. Normalmente en el cuerpo de este mensaje no se incluirá ningún IM. En caso de que el destino quiera contestar al iniciador con otro IM le mandará un MESSAGE.

INFO

El método INFO sirve para que un UA mande información de señalización a otro usuario con el que tiene establecido la sesión. Normalmente el mensaje INFO contiene un cuerpo que almacena información de señalización. Este método siempre incrementa el número de secuencia *CSeq.*

.

Solicitud SIP	Cabeceras obligatorias
INVITE	***Call-ID, CSeq, From, To, Via, Contact, Max-Forwards***
REGISTER	***Call-ID, Cseq, From, To, Via, Max-Forwards***
BYE	***Call-ID, Cseq, From, To, Via, Max-Forwards***
ACK	***Call-ID, Cseq, From, To, Via, Max-Forwards***
CANCEL	***Call-ID, Cseq, From, To, Via, Max-Forwards***
OPTIONS	***Call-ID, Cseq, From, To, Via, Max-Forwards***
REFER	***Call-ID, Cseq, From, To, Via, Max-Forwards, Refer-To***
SUBSCRIBE	***Call-ID, Cseq, From, To, Via, Contact, Allow-Events, Event, Max-Forwards***
NOTIFY	***Call-ID, CSeq, From, To, Via, Contact, Subscription-State, Event, Max-Forwards.***
MESSAGE	***Call-ID, Cseq, From, To, Via, Max-Forwards***
UPDATE	***Call-ID, Cseq, From, To, Via, Contact, Max-Forwards***
INFO	***Call-ID, Cseq, From, To, Via, Max-Forwards***
PRACK	***Call-ID, Cseq, From, To, Via, Max-Forwards, Rack***

Tabla 13 – Descripción cabeceras obligatorias solicitudes SIP

Anexo F. Campos SDP

version

El campo versión almacena la versión del protocolo SDP. Dado que la versión actual del protocolo es 0, el valor ha de ser v=0.

origin

Este campo contiene información sobre el originador e identificadores de la sesión. Este campo se utiliza para identificar de manera unívoca a la sesión. El formato que sigue es el siguiente:

o= <username> <id. sesión> <vers. sesión> <tipo red> <tipo dirección> <dirección unicast>

- username: nombre del usuario que genera la sesión.
- Identificador de sesión: cadena numérica de manera que el conjunto de todos los parámetros que forman el campo o constituyen un identificador único.
- Versión de la sesión: número que identifica la versión de la sesión.
- Tipo de red: identifica en formato de texto el tipo de red. Su valor generalmente es IN (Internet).
- Tipo de dirección: define el tipo de dirección en formato texto (IP4 o IP6)

Dirección *unicast*: representa la dirección desde la cual se creo la sesión. Si se trata de una dirección IPv4 puede valer tanto la notación numérica como la FQDN (*Fully Qualified Domain Name*), es decir, el nombre de la máquina asociado a la representación de la IP en formato numérico.

session name

Representa el nombre de la sesión y no puede quedarse vacío, al menos ha de tener un carácter.

information

Proporciona información adicional acerca de la sesión.

URI

Este campo apunta donde encontrar mas información adicional acerca de la sesión.

email

Contiene la dirección de correo electrónico de la persona que es responsable del establecimiento de la sesión.

phone

Contiene el teléfono de contacto de la persona que es responsable del establecimiento de la sesión.

connection

Almacena datos sobre la conexión de datos. El formato es el siguiente:

c=<tipo de red> <tipo de dirección> <dirección de la conexión>

- Tipo de red: ha de ser IN referida a Internet
- Tipo de dirección: IP4 para versión 4 o IP6 para la versión 6.
- Dirección de conexión. Representa de que IP se envían y en que IP se reciben los datos. Hay que notar que puede ser una IP totalmente distinta a la que se utilizó para el establecimiento de sesión (la dirección IP fuente de los paquetes SIP). Además esta IP puede ser una IP *unicast* o una IP *multicast*. En el caso de que sea una IP *multicast*, los paquetes serán enviados y recibidos a esa IP y el usuario tendrá que estar suscrito a ese grupo. Además es necesario especificar el TTL (*Time to Live*) de los paquetes *multicast* y será un número comprendido entre 0 y 255. Para aplicaciones que requieran subscripciones a múltiples grupos *multicast* se puede utilizar una notación indicando el número de grupos al que el usuario se quiere subscribir. Por tanto, el formato de este parámetro sería el siguiente:

  <ip multicast base>[/<ttl>]/numero de direcciones.

De esta manera un campo c=IN IP4 224.2.1.1/127/3 indicaría que el usuario se subscribiría a tres grupos de direcciones *multicast* (224.2.1.1, 224.2.1.2, 224.2.1.3) cuyo TTL es 127.

bandwidth

Se trata de un campo que almacena el ancho de banda propuesto para utilizar en la sesión durante la transmisión de los datos. El formato de los parámetros de la línea b es el siguiente:

b=<tipo bw>:<ancho de banda>

- El tipo de ancho de banda puede ser CT cuando se tienen varias sesiones en paralelo de manera que indica el ancho de banda total que se puede usar por todos los participantes de la sesión o AS cuando se refiere al ancho de banda utilizado por una única sesión.

El parámetro ancho de banda es el valor numérico expresado en *kilobytes* por segundo.

time

Este campo especifica los tiempos de comienzo y finalización de la sesión. Utiliza marcados con formatos NTP. El aspecto de esta línea es el siguiente:

t=<tiempo comienzo> <tiempo finalización>

Si los tiempos de comienzo y finalización son cero significa que la sesión es permanente.

repeat time

Esta línea representa cada cuanto se tiene que repetir una determinada sesión siguiendo el siguiente patrón:

r=<intervalo repetición> <duración activa> <*offsets* desde inicio>

Aquí se especifica cada cuanto se tiene que repetir la sesión (intervalo repetición), durante cuanto tiempo (duración activa) y el *offset* en caso de que haya mas de una sesión activa.

time zone

Se utiliza para hacer ajustes de tiempo de diferentes zonas horarias.

encryption key

La línea k contiene la clave utilizada para la encriptación de los datos. El formato se muestra a continuación:

k=método:clave_encriptada

El método puede ser *clear, base64, uri,* o *prompt*. Si es *prompt*, la clave no se especifica en el parámetro clave_encriptada, en cualquiera de los otros casos la clave se especifica en este parámetro.

Anexo G. Algoritmos de protocolos de multidifusión

Propagación por la Trayectoria Inversa (RPF)

El algoritmo RPF construye un árbol de entrega desde el origen hasta cada miembro del grupo. Para ello cada datagrama multidifusión recibido en un router se reenviará por las restantes interfaces que cuenten con miembros del grupo, siempre que la interfaz por la que ha llegado es la utilizada por el router para enviar datagramas unienvío hacia el origen del datagrama *multicast* (trayectoria inversa).

Los árboles así obtenidos mantienen rutas óptimas, y además son dependientes de cada origen (distintos orígenes pueden dar lugar a diferentes árboles), con lo cual se distribuye mejor la carga multidifusión por toda la red.

Árbol Centralizado (CBT)

En este algoritmos se selecciona un router que actuará como punto central (*center point*) o raíz del árbol de entrega para un grupo multidifusión.

Cuando un equipo envía un Informe de Pertenencia IGMP para conectarse a un grupo, su petición irá “ascendiendo” hacia el punto central: en caso de que no exista, se irán estableciendo nuevas ramas del árbol, en el camino hacia el punto central. Para ello, si el router que recibe la solicitud ya forma parte del árbol de entrega de ese grupo (ya tiene registrados otros miembros del mismo) verifica si el interfaz por el que ha recibido la petición está asociado al grupo, y si no es así o asocia (añade una nueva rama). Si el router no forma parte del árbol, además reenvía la petición hacia otro router multidifusión que se encuentre en el camino hacia el punto central (creando igualmente una nueva rama).

El árbol así obtenido es óptimo desde el punto central a los miembros del grupo, pero obliga a que el envío *multicast* desde un origen se envíe al punto central, para su posterior reenvío (trayectos sub-óptimos). Otra característica de CBT es que el árbol obtenido es único para el grupo, independientemente de el (o los) originante(s).

Protocolos

Son los responsables del mantenimiento de los árboles, utilizando los modelos de algoritmo descritos o variantes de los mismos. Como soporte en la toma de decisión de los algoritmos, el router puede hacer uso de información de encaminamiento unienvío (las tablas de encaminamiento *unicast*), u obtener la información necesaria mediante el propio protocolo *multicast.*

En función del ámbito de aplicación y de las dimensiones de la red, los distintos protocolos de encaminamiento *multicast* se clasifican en "densos" (*dense mode*) y "esparcidos" (*sparse mode*).

- Soluciones "densas": diseñadas para entornos en los que existe una buena representación de miembros del grupo en la red, y se cuente con ancho de banda suficiente. La estrategia que se sigue es partir de un árbol muy denso (por ejemplo, un árbol de difusión que conecte todas las subredes) e ir podándolo al recibirse notificaciones explícitas por parte de los routers de que no hay actividad en una rama concreta. Habitualmente se utilizan algoritmos RPF en estas soluciones.

- Soluciones "esparcidas": se utilizan en redes de gran cobertura en las que no se cuenta con una gran presencia de miembros del grupo. Construyen el árbol en función de la pertenencia al grupo con notificaciones explícitas por parte de los routers *multicast*, reduciendo la carga de control necesaria. Se basan en algoritmos CBT.

DVMRP (*Distance Vector Multicast Routing Protocol*)

DVMRP es un protocolo de encaminamiento multidifusión de tipo "denso", muy conocido y utilizado en pequeños o medianos sistemas autónomos (SA), ya que se implementa mediante *mrouted*, disponible la mayoría de las versiones *Unix*.

Está basado en un modelo de vector distancia, como ocurre con el protocolo de encaminamiento unienvío RIP. De hecho, tiene muchas similitudes con RIP-2 (vector distancia, métrica de saltos, soporte de máscaras...) y un ámbito de aplicación equivalente.

La obtención del árbol de entrega *multicast* se realiza utilizando como punto de partida un árbol de difusión, y realizando a continuación podas para limitar su alcance. Para ello se construye un árbol de difusión desde el origen basado en RPF, y a continuación se van podando sus ramas para que sólo contenga las activas (que alcanzan miembros del grupo).

La estrategia es la siguiente:

Cada router envía periódicamente su tabla de rutas (vector distancia) a los routers vecinos a través de la dirección 224.0.0.4 (que escuchan todos los routers DVMRP) y, con los vectores distancia recibidos, cada router calcula las rutas óptimas hacia las distintas redes destino existentes (como en RIP). Pero para el intercambio de datagramas *multicast* utiliza RPF: al recibir un datagrama *multicast* por uno de sus interfaces, se extrae la dirección origen del mismo y, si ha entrado por el interfaz óptimo para llegar a esa dirección como destino, se reenvía por los restantes interfaces. En otro caso, se descarta.

Con el envío de un primer datagrama multidifusión de un grupo se establece un árbol que alcanza todas las subredes, aunque no tengan miembros activos de este grupo. A continuación habrá que “podar” las ramas inactivas. Cada router que ha recibido un paquete de multidifusión de un grupo y no cuenta con miembros del mismo en sus interfaces enviará un mensaje de poda al router superior (por donde le llegó el datagrama). Lo mismo hará el superior si recibe mensajes de poda por todos los interfaces por donde reenvió el datagrama, y así hasta cerrar el árbol para este grupo.
Los routers tiene que recordar los mensajes de poda que han enviado, porque estas podas tienen una validez limitada (caducan a las dos horas), y tendrán que renovarse transcurrido ese tiempo. Igualmente, si se realiza la conexión de un nuevo equipo al grupo multidifusión (notificada por un Informe de Pertenencia IGMP) y está podada la rama, el router deberá enviar un nuevo mensaje de conexión o “injerto” que reconstruirá nuevamente la antigua rama podada.

Protocolo MOSPF (Multicast OSPF)

Extensión multidifusión del protocolo OSPF, basado igualmente en algoritmos de estado de enlace. Su modo de operación es el siguiente:

Los routers difunden dentro de cada área OSPF información sobre los grupos de multidifusión activos y sus miembros. De este modo, cada router multidifusión cuenta con dicha información y puede calcular de forma independiente las rutas de camino más corto desde un origen hacia todos los receptores miembros del grupo y reenviar adecuadamente los datagramas *multicast* que reciba por cualquiera de sus interfaces.

Para el intercambio de información sobre grupos *multicast* (y sus receptores) con routers de otras áreas OSPF se utilizan los ABR (routers de borde de área). Para evitar la difusión innecesaria de tráfico *multicast*, uno de los ABR asume la tarea de centralizar la información y distribuir los envíos *multicast* entre áreas. MOSPF se suele clasificar como un protocolo "denso" porque utiliza técnicas de difusión para propagar la información sobre grupos *multicast* y receptores, aunque no utiliza técnicas de difusión para el envío de los datagramas *multicast* a cada miembro del grupo, sino todo lo contrario: los datagramas se envían exclusivamente a cada miembro del grupo y por el camino más corto. Desde este punto de vista puede considerarse un protocolo "esparcido".

En todo caso es un protocolo con dependencia plena de OSPF, y por ello sólo se implementa en redes con encaminamiento unienvío OSPF.

PIM (*Protocol Independent Multicast*)

Precisamente debido a la complejidad de MOSPF, y a su dependencia de un modelo de encaminamiento unienvío, se desarrolló de PIM, un protocolo *multicast* independiente del protocolo unienvío utilizado. PIM define dos modos de operación, Denso (DM, *Dense mode*) y Esparcido (SM, *Sparse mode*), para adecuarse a la densidad de equipos de un grupo multidifusión presentes en un Sistema Autónomo.

Ambos modelos (denso y esparcido) pueden convivir en un sistema autónomo (por ejemplo, para distintos grupos multidifusión), siendo incluso posible conmutar entre ambos modos, lo que permitiría usar ambas soluciones dentro de un mismo grupo.

a) PIM - DM (*Protocol Independent Multicast — Dense Mode*)

PIM-DM implementa el algoritmo RPF. Para determinar si un datagrama *multicast* ha entrado por la ruta correcta se utiliza la información disponible: la tabla de encaminamiento mantenida con el protocolo unienvío adoptado en el sistema. De manera similar a la del protocolo DVMRP, PIM-DM construye inicialmente un árbol de difusión que lleva desde el origen a todos los routers *multicast*, y posteriormente se utiliza la información recolectada a través de IGMP y difundida desde los miembros del grupo hacia el origen para podar las ramas inactivas. Debe mantenerse información sobre las ramas podadas para un grupo, por si hay que volver a reactivarlas (injertos).

PIM-DM es un protocolo muy simple de implementar. Su enfoque denso lo hace adecuado para entornos en que hay pocos grupos y muchos receptores en los mismos, poca distancia entre emisores y receptores, y alto volumen de tráfico *multicast.*

b) PIM — SM (*Protocol Independent Multicast — Sparse Mode*)

PIM-SM utiliza una variante del algoritmo CBT basada en un RP (Punto de Reunión, *Rendezvous Point*), que tiene una función equivalente al *Center Point* del CBT. El RP es la localización donde los originantes conectan con los receptores multidifusión.

Operación

Todos los originantes deben registrarse en el RP. Igualmente todos los receptores se integrarán en un árbol con raíz en el RP, enviándose para ello mensajes de conexión en el momento en que un nuevo integrante se da de alta. Estos mensajes progresan hacia el RP y sirven para abrir nuevas ramas o integrar la nueva ruta en una rama ya existente hacia el RP.

Inicialmente, el tráfico desde el originante se envía al RP, y, a través del árbol de difusión, se reenvía a los destinatarios.

Mantenimiento del árbol

Según lo descrito, el árbol sólo mantiene ramas hacia los equipos que han solicitado explícitamente su conexión al grupo. Para descartar (podar) ramas obsoletas, los routers integrados en el árbol deben enviar mensajes periódicos hacia el RP para confirmar el mantenimiento de las ramas. Si se dejan de recibir estos mensajes las

ramas afectadas se podan. Tampoco se guarda información sobre las podas realizadas, ya que una reactivación de la rama por nuevos miembros se indicará explícitamente.

Optimización

Como ya se comentó en el apartado de algoritmos, los caminos originante-destinatarios son sub-óptimos ya que el tráfico se envía a través del RP, no necesariamente el camino más corto entre origen y destinos. Una vez que comienzan los envíos, es posible rehacer el árbol con raíz en RP para que se convierta en otro árbol que tenga como raíz al originante (caminos óptimos). Para ello se utilizan los mensajes periódicos de conexión desde la fuente para activar nuevos caminos óptimos hacia los destinos y podar los que utilizan el RP como raíz.

PIM-SM es adecuado en entornos con pocos receptores en cada grupo *multicast* y ancho de banda escaso.

Anexo H. Establecimiento y liberación de sesiones multimedia multiusuario.

Establecimiento de sesión.

El inicio del establecimiento de sesión comienza con el envío de la solicitud INVITE por parte del UA iniciador tal como indica el punto 1 de la figura 21. Este INVITE contiene las siguientes cabeceras:

- Una *Request URI* en la línea inicial que contiene la dirección de grupo al que están suscritos los usuarios remotos con los que el iniciador quiere establecer la sesión de audio *multicast*.
- Una cabecera *Via* que almacena la dirección del UA originador de la llamada.
- Un conjunto de cabeceras *Route* que indican la ruta que ha de seguir el INVITE. En el IMS esta cabecera contiene las direcciones del P-CSCF y del S-CSCF en el que está registrado el UA. En la figura 21 hemos supuesto que el UA contacta directamente con el MAS por lo que la cabecera *Route* almacenará la dirección del MAS. No obstante el cliente iniciador puede insertar un conjunto de direcciones en la cabecera *Route* indicando los saltos que ha de seguir el INVITE. Las cabeceras *Route* llevan consigo el parámetro lr (loosing route) cuyo significado se puede encontrar en el Anexo E.
- Una cabecera *Contact* indicando en que dirección y puerto el UA desea recibir las respuestas dentro de este diálogo.
- Una cabecera *Supported:100rel* indicando que el UA soporta el mecanismo de respuesta provisional en modo fiable. Esto es importante ya que el iniciador tiene que recibir una respuesta provisional por parte del destinatario, en este caso el 183 SESSION IN PROGRESS que contiene una carga SDP crucial para la negociación de los parámetros de calidad de servicio.
- Una cabecera *Require:precondition* indicando la presencia de unas precondiciones importantes en la negociación de la calidad de servicio.
- Además las cabeceras *CSeq, From, To, Call-ID, Max-Forwards* y *Content-Type* cuya utilidad se describe en el Anexo E.
- Se incluye también un cuerpo SDP con información multimedia y con información que se utilizará para la negociación de la calidad de servicio y de los prerrequisitos.

Cuando el INVITE llega al MAS (punto 2 en la figura 21) entre otras cosas, examina la cabecera *Request URI* para saber a que grupo de usuarios ha de reenviar el INVITE. Junto a esto el MAS genera una serie de respuestas provisionales 100 TRYING que le envía al iniciador de la llamada (punto 3 en la figura 21) y añadirá a la cabecera *Record-Route* su propia dirección. También asignará una ip *multicast* a la conexión de audio que será donde los participantes de la multiconferencia enviarán los paquetes de voz y añadirá toda la información pertinente en las cabeceras *Via* y *Record-Route* para hacer posible el routing.

Una vez que los destinatarios reciben el INVITE generan una respuesta provisional 183 SESSION IN PROGRESS (punto 4 de la figura 21). Estos mensajes contienen las siguientes cabeceras:

- La cabecera *Contact* indicando en que dirección y puerto el UA desea recibir las respuestas dentro de este diálogo.
- Una cabecera *Require:100rel* que indica que el UA que reciba la respuesta ha de mandar un mensaje PRACK de confirmación de respuesta provisional recibida.
- Una cabecera *RSeq* para poder distinguir entre respuestas provisionales.
- Las cabeceras cabeceras *CSeq, From, To, Call-ID,* y *Content-Type.*
- Por último, se tiene una carga SDP en el cuerpo del mensaje con los prerrequisitos.

El MAS espera hasta que se reciben las respuestas 183 SESSION IN PROGRESS de cada participante (punto 5 de la figura 21). Cuando se han recibido todos los mensajes, el MAS genera una respuesta conjunta que es enviada al iniciador de la llamada. Entre otras acciones el MAS generará una nueva SIP URI que identificará unívocamente a la sesión. Esta SIP URI se incluirá en la cabecera *Contact* del 183 SESSION IN PROGRESS de manera que las futuras solicitudes enviadas por el iniciador de la sesión pondrán este identificador en el campo SIP URI de la solicitud.

En el momento que el UA iniciador recibe la respuesta conjunta 183 SESSION IN PROGRESS responde con un mensaje PRACK (punto 6 de la figura 21). Esta solicitud contiene las cabeceras *Via, CSeq, Call-ID, From, To, Max-Forwards* y *Route* (copiada de la cabecera *Record-Route* del 183 SESSION IN PROGRESS). Además añade la cabecera *RAck* en respuesta unívoca al 183 SESSION IN PROGRESS. Por último incluye una carga SDP en el cuerpo del mensaje como parte del protocolo de

negociación de los parámetros de calidad de servicio que se verá mas adelante. En este punto exactamente comienza la reserva de recursos.

Este PRACK se envía al MAS. Cuando el MAS recibe esta solicitud reenvía de nuevo a todos los participantes el mensaje tal y como indica el punto 7 de la figura 21. La SIP URI fijada en la *Request-URI* será ahora la propia para cada destinatario.

Cada destinatario responderá al PRACK con una respuesta 200 OK tal y como se muestra en el punto 8 de la figura 21. En este instante comenzará la reserva de recursos en cada uno de los destinatarios.

Una vez que todos los 200 OK llegan al MAS procedentes de todos los destinatarios, el MAS genera una única respuesta tal y como se indica en el punto 9 de la figura 21 que se reenviará al UA iniciador.

El UA recibe este mensaje 200 OK y espera a terminar con la reserva de recursos. Una vez finalizado esto envía al MAS una solicitud UPDATE (punto 10 de la figura 21) para notificar que la reserva de recursos ha finalizado y comienza la suscripción al grupo *multicast*. Este mensaje lleva consigo un cuerpo SDP con los parámetros definitivos de la calidad de servicio. El MAS de nuevo reenvía a todos los participantes el UPDATE (punto 11 de la figura 21).

Cada participante responde con un 200 OK en el que se asiente los parámetros definitivos de la calidad de servicio (punto 12 de la figura 21) dentro de la carga SDP. Cuando todos los 200 OK llegan al MAS se genera de nuevo una única respuesta (punto 13 de la figura 21) que se reenvía al UA iniciador.

Para alertar de que la reserva de recursos ha terminado en el lado de los destinatarios cada UA envía un RINGING al MAS (punto 14 de la figura 21). Esta alerta se repite periódicamente hasta el momento que el usuario acepta el establecimiento de sesión.

En el momento que el primer RINGING llega al MAS, éste lo reenvía al UA iniciador. Sin embargo, el MAS tan solo reenviará la primera réplica del RINGING (punto 15 de la figura 21). Las réplicas subsecuentes no las reenviará.

En el momento que el usuario acepta la llamada (punto 16 de la figura 21) envía un 200 OK para informar al iniciador que ha aceptado el establecimiento de la sesión. El MAS recibe esta respuesta y la reenvía al iniciador.

Seguidamente el MAS envía un mensaje NOTIFY (punto 18 en la figura 21) a ambos UA (el iniciador de la llamada y el destinatario que la aceptó). Este mensaje indica el estado de la sesión multimedia del usuario. La información relativa a este estado se almacena en formato *xml* dentro del cuerpo del mensaje SIP. El significado de cada campo se puede encontrar de manera detallada en la referencia [13]. La figura 60 ilustra la estructura de los elementos de información:

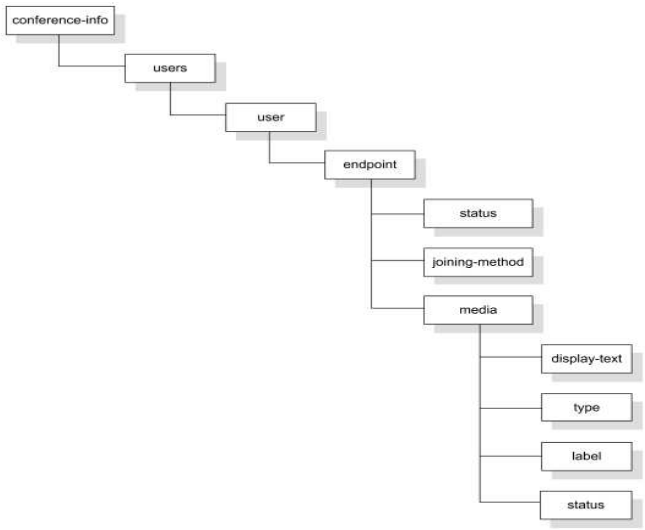

Figura 60 – Estructura del contenido xml en el NOTIFY

El documento de información sobre la sesión multimedia multiusuario comienza con el elemento raíz <conference-info>, para el cual se utilizarán los siguientes atributos:

- *entity: c*ontiene la URI SIP generada por el MAS para la sesión multimedia multiusuario.
- *state:* indica si el documento contiene la información completa sobre el estado de la sesión (valor "*full*") o únicamente la información que ha cambiado desde el documento previo (valor "*partial*").

- *version:* permite al receptor del documento ordenar apropiadamente las notificaciones recibidas.

El MAS debe incrementar el número de versión para cada notificación enviada que reporta un cambio en el estado de la sesión a un UE suscriptor.

El elemento de información <conference-info> contendrá un elemento hijo <users>. Éste, a su vez, es un contenedor de elementos hijo <user>, cada uno de ellos describiendo un único usuario participante en la sesión. Se utilizarán los siguientes atributos para el elemento <user>:

- *entity:* contiene la URI SIP del usuario al que el elemento <user> hace referencia.
- *state:* indica si el documento contiene la información completa sobre el usuario (valor "*full*"), o únicamente la información que ha cambiado desde el documento previo (valor "*partial*").

Para cada elemento <user>, se define el elemento hijo <endpoint>. Este elemento será utilizado en el contexto de la sesión multimedia multiusuario para almacenar la información sobre el UE de un usuario y sobre los parámetros asociados con la sesión que han sido acordados por éste. Para este elemento de información, se utilizarán los siguientes atributos:

- *entity*: el MAS incluirá aquí la URI SIP indicada por el UE en la cabecera *Contact* recibida desde el <endpoint> en una solicitud INVITE (en caso del ser el UE el iniciador de la sesión) o en una respuesta SESSION IN PROGRESS (en caso de no ser el UE el iniciador de la sesión).
- *state*: indica si el elemento contiene la información completa sobre el UE (valor "*full*"), o únicamente la información que ha cambiado desde el documento previo (valor "*partial*").

Asociados con el elemento <endpoint> se utilizarán los siguientes elementos hijo:

- <status>*:* contiene el estado del UE y tomará uno de los siguientes valores:
    - *pending*: este es el estado inicial de cualquier UE participante. En el caso de un UE destino, este estado indica que el UE no ha aceptado el establecimiento de la sesión. En el caso del UE iniciador, este estado indica que ningún UE destino ha aceptado el establecimiento de la sesión.
    - *connected*: este estado indica que el UE es un participante activo en la sesión multimedia multiusuario. Así, si el UE es el iniciador de la sesión,

éste pasa a estado *connected* si algún UE destino ha aceptado el establecimiento de la sesión. Si el UE no es el iniciador de la sesión, pasa a estado *connected* cuando su usuario correspondiente acepta el establecimiento de la sesión.

- o *disconnected*: no existe diálogo SIP ente el UE y el MAS. El UE pasa a este estado tras abandonar la sesión multimedia multiusuario.

- <joining-method>: indica el modo en el que el que el UE se convirtió en participante de la sesión multimedia multiusuario. Puede tomar los siguientes valores:

  - o *dialed-in*: el UE es el iniciador de la sesión multimedia multiusuario (esto es, envió el mensaje INVITE inicial al MAS)
  - o *dialed-out*: el UE recibió una invitación para participar en la sesión multimedia multiusuario (esto es, el MAS envió una solicitud INVITE al UE).

- <media>: contiene información sobre un componente de información, y se incluye para cada componente de información acordado con el UE. Se define un único atributo para el elemento <media>, esto es, el atributo "*id*". Este atributo es el identificador del componente de información en el contexto del elemento <endpoint> en el que se incluye. Se utilizarán los siguientes elementos hijo para el elemento de información <media>:

  - <display-text>: contiene un texto explicativo para el componente de información. El valor de este elemento se corresponde con el del atributo "i" de SDP para un componente de información.
  - <type>: contiene el tipo del componente de información. El valor de este elemento debe ser uno de los valores registrados para el atributo "m" de SDP, esto es, "*audio*", "*video*", "*application*" o "*message*".
  - <label>: contiene un identificador único para el componente de información en el contexto de la sesión multimedia multiusuario. El valor de este elemento es asignado por el MAS, y es indicado por éste a los distintos UEs participantes en la sesión mediante el atributo "*label*" de SDP.

- <status>: indica la direccionalidad del componente de información desde el punto de vista del UE. Puede tomar los valores "*sendonly*", "*recvonly*" o "*sendrecv*", según se definen en la especificación de SDP.

Volviendo al punto 18 de la figura 21, el MAS envía un NOTIFY a cada uno de los participantes activos con la información descrita en el cuerpo del NOTIFY. La etiqueta <status> bajo la etiqueta <endpoint> nos servirá para saber en que estado están todos los participantes de la sesión y quien está en disposición de empezar a mandar la voz a la dirección IP *multicast*. De esta manera cada participante sabrá el estado del resto.

Además el UA iniciador de la llamada asiente el 200 OK del INVITE (punto 17 de la figura 21) enviando una respuesta ACK (punto 19 de la figura 21).

El ACK alcanza el MAS y éste lo reenvía al usuario que generó el 200 OK del INVITE (punto 20 de la figura 21)

Por otra parte, cada NOTIFY que el UA recibe ha de ser respondido con un 200 OK. Por tanto, se generarán 2 mensajes 200 OK que se recibirán en el MAS como respuesta al NOTIFY que generó (puntos 21 y 22 de la figura 21)

El punto 23 de la figura 21 indica la aceptación de la sesión por parte del segundo usuario que fue invitado. Como consecuencia de esto, el MAS genera un NOTIFY (punto 24) para cada usuario que se encuentra activo en la sesión para indicar la aceptación de la llamada del segundo usuario. Ahora el valor de la etiqueta <status> de este usuario es *connected* en el NOTIFY que el MAS envía a todos los usuarios. Los usuarios online responden al MAS con un mensaje 200 OK (punto 25 de la figura 21).

Por último, el MAS autogenera un mensaje de asentimiento ACK que es enviado al usuario que aceptó la llamada (punto 26 de la figura 21). Realmente este mensaje lo construye el MAS en el momento que el iniciador envía el ACK y lo envía a cada usuario que acepta la sesión

Liberación de sesión.

Se propone el caso de que se tenga una sesión entre 4 participantes como la mostrada en la figura 21.

En un momento dado el iniciador de la llamada (jose) decide abandonar la sesión enviando un BYE (punto 1 de la figura 21). El MAS le confirma con un 200 OK (punto 2) y el usuario queda fuera de la sesión. En este momento el MAS ha de avisar al resto de los participantes (oscar, carlos y antonio) que uno de ellos ha abandonado la sesión. Para ello manda un NOTIFY (punto 3 de la figura 21) a cada uno de los participantes activos. Este comportamiento es idéntico al caso en que un usuario se une a la sesión y que ha sido descrito en el establecimiento de sesión La única diferencia es que ahora el estado del usuario iniciador es *disconnected* y es lo que reciben el resto de los usuarios que están online en el cuerpo del NOTIFY.

Todos los usuarios *online* contestan con un 200 OK (punto 4 de la figura 21). En el punto 5, oscar, decide abandonar la sesión y de nuevo el MAS notifica a el resto de los usuarios (puntos 6 y 7 de la figura 21).

En este momento quedan 2 usuarios *online*. En el punto 8 de la figura 21 carlos manda un BYE para salir de la sesión. En este punto el MAS sabe que solo queda un usuario online. La implementación del MAS permite enviar un BYE al único usuario (antonio) que queda online (punto 9 de la figura 21) dado que carece de sentido que permanezca un solo usuario activo en la sesión. Este destinatario (antonio) responde con un 200 OK y la sesión queda cancelada totalmente.