2. INTRODUCCIÓN A VOIP

En este capítulo vamos a hablar de VoIP. Para ello analizaremos cuáles son las características apropiadas que debe tener una red para que la voz se pueda transmitir de forma correcta considerando sus protocolos de transporte, señalización y los codificadores necesarios para su transmisión. Además comentaremos las ventajas e inconvenientes que caracterizan a VoIP.

2.1. Funcionamiento básico de VoIP

Voz sobre IP es una tecnología que permite realizar llamadas telefónicas sobre redes de datos, como puede ser Internet. Voz sobre IP también es conocida como la telefonía de Internet. La telefonía tradicional consiste en transportar la señal analógica sobre cable de cobre, pero la tecnología de VoIP convierte la voz analógica en paquetes de datos digitales que soportan la comunicación sobre el protocolo IP y UDP (*User Datagram Protocol*) [RFC768]. Además, pueden emplear protocolos para aplicaciones en tiempo real como el RTP (*Real Time Protocol*) [RFC3550].

VoIP presenta la ventaja del acceso abierto. La documentación en un estándar abierto, así que se puede escribir o modificar código sin problemas. Las aplicaciones de VoIP debido a que se encuentran en continuo desarrollo, proporcionan ininterrumpidos servicios de calidad, fiabilidad, seguridad y compatibilidad para teléfonos basados en IP, video y convergencia a aplicaciones de escritorio. Con VoIP se pueden recibir y enviar llamadas independientemente de la ubicación con sólo conectarse a una conexión de banda ancha.

La principal ventaja que esta tecnología ofrece respecto al resto de opciones, la facilidad y flexibilidad para la introducción de nuevos servicios en la red. Entre las características que proporcionan estas ventajas podemos citar:

- Desarrollo abierto: Al tratarse de estándares basados en IP, se dispone de una gran cantidad de profesionales con capacidad en este campo.
- Disponibilidad de Software ya desarrollado: La adopción de VoIP como tecnología de voz permite beneficiarse de la enorme capacidad de desarrollo de toda la comunidad de Internet.

Como cualquier otra tecnología, la arquitectura VoIP también posee ciertos defectos. Durante una comunicación por VoIP puede llegar a producirse retraso en la llegada de los paquetes o incluso cortes de información debido a las pérdidas.

Otro de los aspectos negativos dentro de la comunicación a través de tecnología VoIP es el posible deterioro de la comunicación al ser recibida por el usuario.

Uno de los requisitos más importantes para aportar calidad de servicio a una aplicación de VoIP es proporcionar suficiente ancho de banda a la red para que pueda soportar el tráfico de voz en tiempo real.

Los factores principales de los que depende el ancho de banda son el codificador usado y el número de muestras o tramas de voz que se deseen introducir por paquete. Esto es así porque ambos factores están directamente relacionados con el retardo. Los codificadores en los terminales de VoIP comprimen la señal de voz e introducen dos tipos de retraso:

- procesamiento: tiempo necesario para codificar el código en una sola trama de voz.
- retraso de la trama: tiempo necesario para que el sistema que envía transmita la trama.

Existen tres tipos de codificadores: Los de forma de onda, los de fuente y los híbridos.

Los de forma de onda básicamente muestrean y codifican la señal analógica de entrada sin tener en cuenta cómo se generó la señal en el origen. A continuación, se transmiten los valores cuantificados de las muestras para que en el destino se reconstruyan. Consiguen una calidad de salida muy alta y no son complejos. Su mayor desventaja es que consumen grandes cantidades de ancho de banda en comparación con otros codificadores. Cuando se usan con anchos de banda menores, la degradación de la calidad de voz disminuye de manera significativa.

Los de fuente son aquellos que modelan la señal entrante con un filtro matemático que representa la forma en la que se produce el habla. Por lo general, utilizan un modelo de predicción lineal del tracto vocal con una bandera (flag) de voz/no-voz que representa la excitación aplicada al filtro. La información que se transmite al otro extremo es un conjunto de parámetros del modelo en lugar de una representación de la propia señal. En el otro extremo, utilizando la misma técnica de modelado pero a la inversa, toma los valores recibidos y reconstruye la señal analógica. Los vocoders operan a velocidades muy bajas pero la calidad de voz resultante suele ser también baja (sonidos bastante sintéticos).

Los híbridos intentan proporcionar un equilibrio entre las características de los codificadores de forma de onda y los de fuente. Su funcionamiento es una mezcla de los dos anteriores: algoritmos que usan muestras de la señal de voz con modelos matemáticos de cómo se producen los sonidos.

Entre los codificadores utilizados en VoIP encontramos los G.711 [ITUG.711], G.723.1 [ITUG.723.1] y el G.729 [ITUG.729] (especificados por la ITU-T).

2.2.Protocolos de VoIP

Existen numerosos protocolos de señalización, sin embargo a continuación sólo enumeraremos el protocolo más común y que es utilizado por el cliente PJSIP. El protocolo de transporte utilizado será RTP/RTCP.

2.2.1. Señalización SIP

SIP (*Session Initiation Protocol*) [RFC3261] es el estándar que IETF utiliza para establecer las conexiones VoIP. Es un protocolo que pretende desarrollar los *gateways* para que sean más inteligentes.

Se trata de un protocolo de control de la capa de aplicación encargado de crear, modificar y terminar sesiones con uno o más participantes. Estas sesiones multimedia incluyen aplicaciones de audio, video y datos con varios participantes que se pueden comunicar a través de una comunicación *unicast, multicast* o una combinación de ambas.

Este protocolo de transporte usa invitaciones para crear mensajes SDP (*Session Description Protocol)* [RFC4566] para llevar a cabo el intercambio y establecer el uso de canales de control.

SIP es un protocolo punto a punto y por lo tanto la parte de inteligencia está incluida en los terminales. Se definen dos elementos fundamentales para implementar las funcionalidades básicas:

1. User agents-UA: consta de dos partes, el cliente y el servidor. El primero genera peticiones SIP y recibe las respuestas, el otro genera las respuestas a las distintas peticiones.
2. Servidores: aquí nos encontramos con una división conceptual de tres tipos de servidores diferentes. Esta división aporta al conjunto estabilidad y mejora el rendimiento:
    - Proxy Server: tiene la tarea de enrutar las peticiones de otras entidades más próximas a su destino. Actúa como cliente y servidor para el establecimiento de llamadas entre usuarios. Existen los *stateful* que mantienen el estado de las transacciones durante el procesamiento de las peticiones y permiten la división de una petición en varias y el otro tipo son los *stateless*, que al contrario no mantienen estado únicamente se limitan a reenviar los mensajes.
    - Registrar Server: este servidor acepta peticiones de registro de los usuarios y guarda la información de estas para suministrar un servicio de localización y traducción de direcciones en el dominio que controla.
    - Redirect Server: este servidor genera respuestas de redirección a las peticiones que recibe y reencamina las peticiones hacia el próximo servidor.

El protocolo SIP define principalmente seis tipos de solicitudes:

- INVITE: establece una sesión.
- ACK: confirma una solicitud INVITE.
- BYE: finaliza una sesión.
- CANCEL: cancela el establecimiento de una sesión.

- REGISTER: comunica la localización de usuario (nombre de equipo, IP).
- OPTIONS: comunica la información acerca de las capacidades de envío y recepción de teléfonos SIP.

Las respuestas se generan como retorno de una petición devolviendo un código de estado. En este caso la línea inicial recibe el nombre de *status line*, que llevara el SIP utilizado, código de respuesta y una pequeña descripción de ese código. Podemos recibir estas respuestas según el rango:

- 1xx: respuestas informativas.
- 2xx: respuestas de éxito.
- 3xx: respuestas de redirección.
- 4xx: errores de solicitud.
- 5xx: errores de servidor.
- 6xx: errores globales.

SIP comparte con HTTP alguno de sus principios de diseño, siguiendo una estructura petición respuesta con coditos de respuesta similares a los de HTTP. Por ejemplo un código de retorno 200 significa OK y el 404 es no encontrado. Y la localización la basa en DNS. Por lo tanto este protocolo está basado en el intercambio de peticiones y respuestas que consisten en una línea inicial. Recibe el nombre de *request line* e incluyen el nombre de método al que invocan, el identificador del destinatario, el protocolo SIP que se está utilizando. En la figura 2 podemos ver el intercambio de mensajes típicos durante una llamada de VoIP.

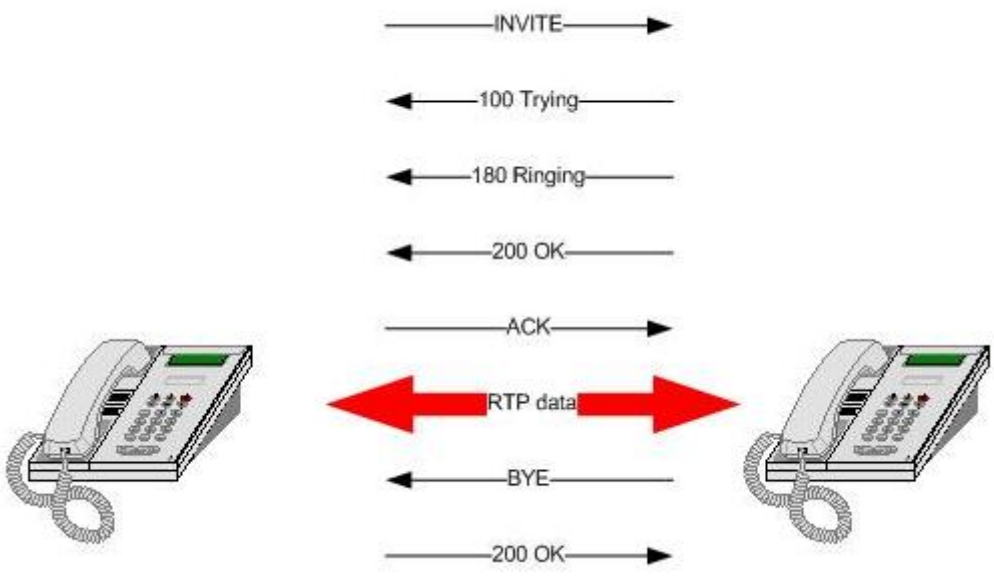

FIGURA 2: SESIÓN SIP ENTRE DOS TERMINALES

2.2.2. Transporte de medios: protocolo RTP/RTCP

Para transportar la voz o el vídeo sobre IP, se utilizan el protocolo IP a nivel 3 y el protocolo UDP a nivel 4 (véase figura 3). Pero estos dos protocolos UDP e IP no son suficientes para asegurar el transporte de la voz. De hecho, UDP es un protocolo sin corrección de errores, y en ningún momento se asegura la llegada de paquetes en su orden de emisión. Para el transporte de datos en tiempo real, como la voz o el vídeo, es necesario utilizar dos protocolos suplementarios: RTP (*Real-Time Transport Protocol*) y RTCP (*RTP Control Protocol*).

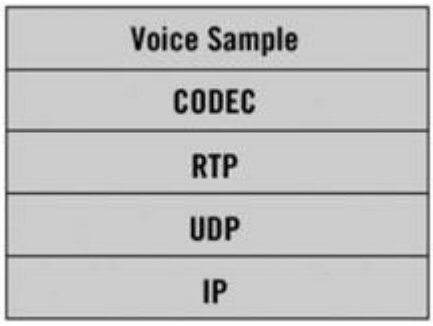

FIGURA 3: TORRE DE PROTOCOLOS DE VOIP

RTP y RTCP son dos protocolos que se sitúan a nivel de aplicación y se utilizan con el protocolo de transporte UDP. RTP y RTCP pueden utilizar el modo *unicast* (punto a punto) y el modo *multicast* (multipunto).

RTP y RTCP utilizan puertos diferentes. RTP utiliza un número de puerto par, y RTCP el número de puerto impar que sigue a continuación. Cuando una sesión RTP es abierta, al mismo tiempo se abre una sesión RTCP implícita.

La función de RTP es proporcionar un medio uniforme de transmisión de datos sometidos a limitaciones de tiempo real (audio, vídeo, etc.).

RTP permite:

- Identificar el tipo de información transportada,
- Añadir marcadores temporales que permitan indicar el instante de emisión del paquete. De esta forma, la aplicación destino podrá sincronizar los flujos y medir los retardos y la fluctuación.
- Incluir números de secuencia a la información transportada para detectar la pérdida de paquetes y poder entregar los paquetes a la aplicación destino.

Además, RTP puede ser transportado por paquetes *multicast* para encaminar conversaciones hacia múltiples destinos. No obstante, RTP no está concebido para realizar reservas de recursos o controlar la calidad de servicio, ni garantiza la entrega del paquete en recepción.

RTP transporta las señales audio o vídeo codificados mediante paquetes RTP que contienen una cabecera RTP (*header*) seguido de estas señales audio o vídeo Un paquete RTP pasa por la capa UDP, que le añade una cabecera UDP. El conjunto es traspasado a la capa IP, que agrega una cabecera IP. Entonces, el datagrama IP es encaminado hacia el destino. En recepción, el paquete es entregado a la aplicación adecuada.

La cabecera de un paquete RTP (RTP *header*) está obligatoriamente constituida de 12 octetos, y eventualmente seguida de una lista de identificadores de fuentes contributivas CSRCs en el caso de un mezclador. A esta cabecera precede el "*payload*" que representa los datos útiles (véase figura 4).

FIGURA 4: FORMATO DEL PAQUETE RTP

Version (V) (2 bits): Este campo indica el número de versión RTP utilizada. La versión actual es la versión 2.

Padding (P) (1 bit): Si vale 1, este campo significa que el campo de datos (*payload*) tiene una parte de relleno. Recordemos que la longitud de datos debe ser un múltiplo de 4 octetos, lo que hace necesarios los octetos de relleno, sobre todo para el último paquete. El último octeto de esta parte de relleno indica el número de octetos de relleno que se deben ignorar.

Extension (X) (1 bit): Si vale 1, este campo indica que la cabecera fija tiene una parte de cabecera suplementaria.

CSRC Count (CC) (4 bits): Este campo contiene el número de identificadores CSRC que siguen la cabecera fija, es decir, el número de fuentes contributivas ligadas a este paquete.

Marker (M) (1 bit): Se trata de un bit de señalización. Su significado depende de los datos transportados.

Payload type (PT) (7 bits): Este campo identifica el tipo de contenido (audio, vídeo, etc.) que representa el tipo de codificación de información transportada en el paquete.

Sequence Number (16 bits): El valor de este campo se incrementa en 1 por cada paquete RTP enviado, pero su valor inicial es aleatorio. Este campo permite detectar paquetes RTP perdidos.

TimeStamp (32 bits): Un protocolo como RTP utiliza marcas temporales para datar los paquetes emitidos. Estas informaciones son la base de los cálculos que permiten evaluar el retardo y la fluctuación introducidos por un sistema de comunicación. La pertinencia de estas informaciones depende enteramente de la precisión y de la sincronización de los relojes utilizados en las maquinas que van a basar sus cálculos en estos valores.

Synchronization Source (SSRC) (32 bits): Este campo identifica la fuente que ha producido el paquete. Al principio de una sesión, cada participante escoge un número de SSRC. En este caso se habla de sincronización ya que la escala de tiempo establecida por la fuente en estos paquetes va a servir de referencia a los receptores para restituir la información correctamente.

Contributing source (CSRC): De 0 a 15 instancias de este campo pueden estar presentes en la cabecera del paquete RTP. El número está indicado por el campo CC.

Cuando un flujo RTP es el resultado de una agregación por un mezclador RTP, la lista de SSRCs que han aportado su contribución es añadida en la cabecera de los paquetes RTP del flujo resultante (véase figura 5).

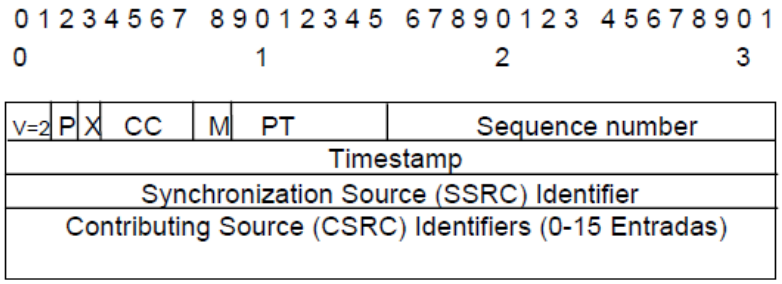

FIGURA 5: CABECERA RTP

El protocolo RTCP está basado en transmisiones periódicas de paquetes de control para todos los participantes en la sesión. Es un protocolo de control de los flujos RTP, que permite transportar informaciones básicas de los participantes de una sesión y de la calidad de servicio.

Existen cinco tipos distintos de paquetes RTCP para cada tipo de información:

- *SR* (*Sender Report*) contiene las estadísticas de transmisión y de recepción para los participantes que son emisores activos.
- *RR* (*Receiver Report*) contiene estadísticas de recepción para los participantes que no son emisores activos pero sí receptores de una sesión.
- *SDES* (*Source Description*) describe la fuente: nombre, email, teléfono, etc.
- *BYE* permite a una estación indicar el fin de su participación en una sesión.
- *APP* es un paquete de señalización específico a una aplicación.

El control de flujo RTP se realiza guardando una evaluación del número de participantes en una sesión (fuentes y receptores). A partir de esta evaluación, se calcula un intervalo de tiempo que sirve de periodo de recurrencia en la difusión de informaciones SR o RR, según el caso. Globalmente, los algoritmos de control limitan el volumen de las informaciones de control transmitidas (los datos RTCP, por lo tanto) a un 5% del volumen global de los intercambios de la sesión. En este volumen, el 25% está reservado a las informaciones de las fuentes (mensajes SR). De esta forma, se garantiza una posibilidad de gestionar grupos de gran tamaño del punto de vista del volumen de

información intercambiada. Cuanto más elevado es el número de participantes, menos precisa es la visión que tiene cada participante del estado de la red.

Los paquetes que se transmiten con más frecuencia son SR y RR. Los participantes en una sesión que a la vez emiten y reciben paquetes RTP, utilizan paquetes RTCP SR. El formato de este paquete está presentado en la figura 6.

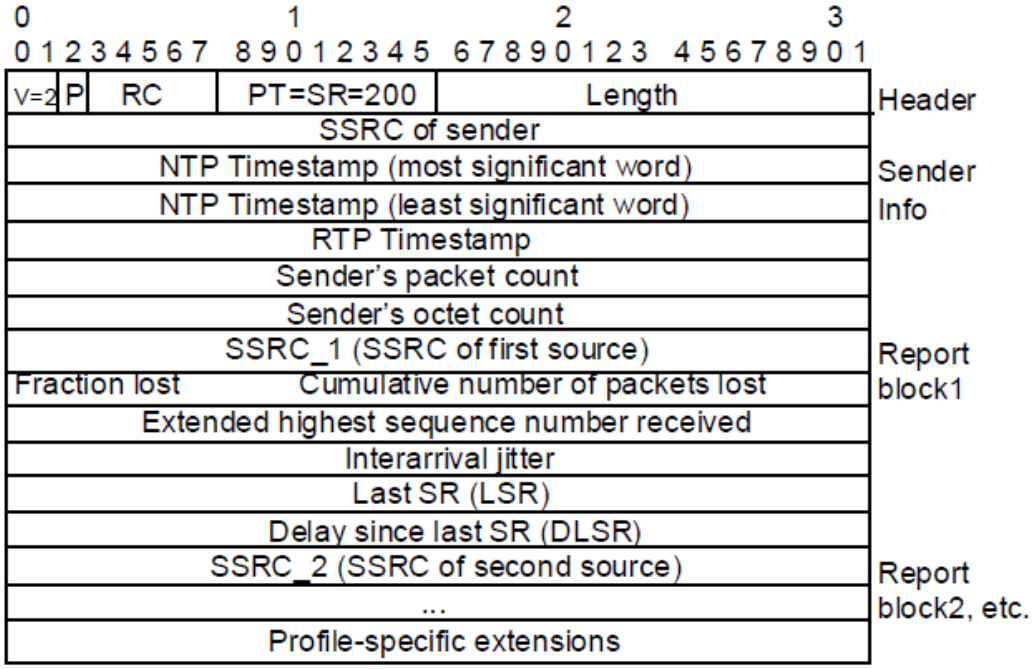

FIGURA 6: PAQUETE RTCP SENDER REPORT

El paquete SR contiene una cabecera, informaciones sobre el emisor, un cierto número de bloques de informes de recepción y opcionalmente una extensión específica al perfil.

La estructura es similar a la de un paquete RTP. La cabecera contiene los campos siguientes:

Version (V) (2 bits): Este campo indica el número de versión RTCP. El valor de la versión actual del protocolo RTCP es 2.

Padding (P) (1 bit): Cuando vale 1, este campo indica que hay un relleno cuyo tamaño es indicado por el último octeto.

Reception report count (RC) (5 bits): Este campo precisa el número de informes de recepción contenidos en el paquete SR, considerando un informe para cada fuente. Por lo tanto, se pueden incluir hasta un máximo de 31 informes en el paquete SR.

Packet type (PT) (8 bits): Este campo indica el tipo de paquete; se trata de un paquete SR, representado por el valor 200.

Length (16 bits): Este campo indica la longitud total del paquete en palabras de 32 bits (cabecera y relleno comprendidos).

Las informaciones sobre el emisor consisten en los campos siguientes:

SSRC of sender (32 bits): Este campo precisa la identificación de la fuente específica del emisor.

NTP timestamp (64 bits): La representación del tiempo utilizado por NTP (Network Time Protocol) es bastante simple: una fecha está codificada en 64 bits y medida en segundos desde las 0h del 1 de enero de 1900. La parte entera de esta fecha expresada en segundos está codificada en 32 bits de peso fuerte (palabra más significante), y la parte fraccionaria (fracción de segundo) en los 32 bits de peso débil (palabra menos significante). Esta representación garantiza una precisión de aproximadamente 200 pico-segundos, probablemente suficiente para la mayor parte de aplicaciones. El problema de acercamiento a cero de esta representación llegará en el 2036. Entonces, será necesario definir una nueva versión del protocolo antes de esta fecha.

RTP timestamp (32 bits): Este campo indica el mismo tiempo que el que indica NTP *Timestamp* precedente, pero utilizando las mismas unidades que las utilizadas para especificar el valor *timestamp* en los paquetes RTP.

Sender's packet count (32 bits): Este campo indica el número total de paquetes RTP transmitido por el emisor desde el inicio de la sesión. Está reinicializado en el contexto de una sesión si el emisor cambia de identificador SSRC.

Sender's octet count (32 bits): Este campo indica el número total de octetos RTP (sólo son considerados los octetos de datos de usuario y no los octetos de cabecera o de relleno) transmitidos por el emisor desde el principio de la sesión. También se reinicializa si el emisor cambia de identificador de SSRC.

Uno o varios bloques de informes de recepción (RR, *Receiver Report*) siguen la información del emisor. Proporcionan a los otros participantes de la sesión la información referente al número de paquetes RTP que han sido emitidos o recibidos con éxito por el emisor del paquete SR.

Los campos siguientes están incluidos en cada bloque RR (véase figura 7).

SSRC_n (32 bits): Este campo especifica la identificación de la fuente en la sesión, que se refiere a los datos incluidos en el bloque RR.

Fraction lost (8 bits): Este campo indica la fracción de paquetes RTP perdidos desde el último informe emitido por este participante. La fracción representa la relación entre el número de paquetes perdidos y el número de paquetes esperados. El número de paquetes perdidos puede ser deducido a partir del análisis del número de secuencia (*Sequence Number*) de cada paquete RTP recibido.

Cumulative number of packets lost (24 bits): Este campo indica el número total de paquetes RTP de la fuente en cuestión, que han sido perdidos desde el principio de la sesión RTP.

Extended highest sequence number received (32 bits): Este campo especifica el número de secuencia del último paquete RTP recibido desde esta fuente SSRC_n.

Interarrival jitter (32 bits): Este campo informa de la variación del retardo de transmisión de los paquetes RTP.

Last SR Timestamp (LSR) (32 bits): Este campo representa los 32 bits del medio el campo NTP *Timestamp* utilizado en el primer paquete SR recibido desde la fuente en cuestión. De esta forma, considera los 16 bits de peso débil de la parte entera de esta fecha (segundos) y los 16 bits de peso fuerte de la parte fraccionaria (fracciones de segundo). Si ningún paquete RTCP SR aun no ha sido recibido, entonces el valor de este campo es igual a 0.

Delay Since Last SR (DLSR) (32 bits): Este campo representa el retardo expresado en unidades de 1/65536 segundos entre el instante de recepción del último paquete SR de la fuente SSRC_n y el instante de emisión de este bloque RR. Si ningún paquete SR aun no ha sido recibido de la fuente SSRC_n, el valor del campo DLSR se sitúa a 0.

El paquete RCTP RR (*Receiver Report*) en cambio, es emitido por un participante en una sesión que recibe paquetes RTP pero que no emite. El formato del paquete está representado en la figura 7. Tiene una estructura similar al paquete RTCP SR, pero indica el valor 201 para el campo *payload type* y no incluye información específica del emisor.

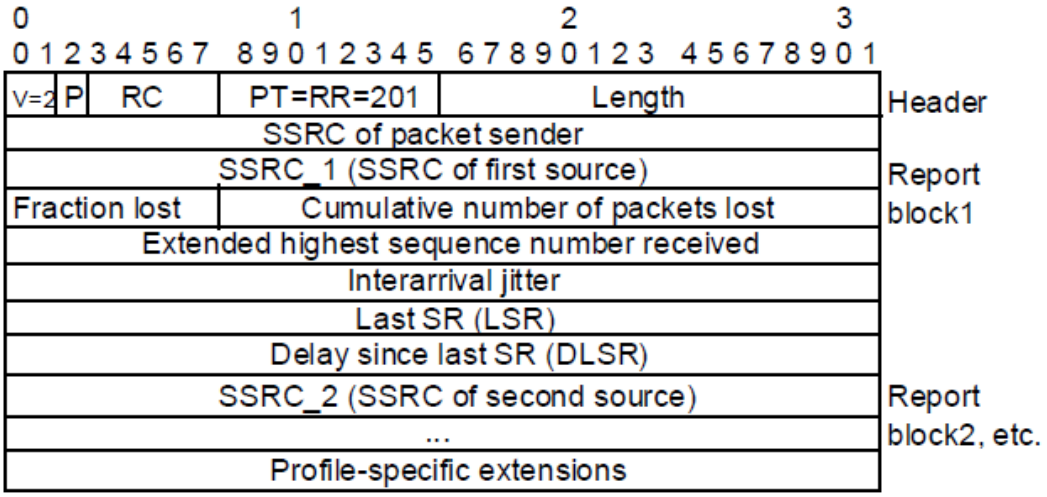

FIGURA 7: PAQUETE RTCP RECEIVER REPORT

En definitiva, los protocolos RTP y RTCP están adaptados para la transmisión de datos en tiempo real. No obstante, funcionan con una estrategia de extremo a extremo y, por tanto, no pueden controlar el elemento principal de la comunicación, la red. Sin embargo, cualesquiera sean los esfuerzos de adaptación de los emisores, o los medios establecidos por los receptores, es en el centro de la red donde se generan las disfuncionalidades críticas.

2.3. Código abierto VoIP: PJSIP

Como ya se comentó, una de las principales características del estándar VoIP es que es un código abierto (*Open Source*) y que al tratarse de estándares basados en IP, se

dispone de una gran cantidad de profesionales con capacidad en este campo. Todo ello posibilita la disponibilidad de software ya desarrollado que nos permite beneficiarnos de la enorme capacidad de desarrollo de toda la comunidad de Internet. Partiendo de la disponibilidad de protocolos como SIP, se optimiza en costes y en tiempo el esfuerzo dedicado a la realización de nuevos desarrollos.

PJSIP es uno de esos códigos abiertos y gratuitos. Consiste en un conjunto de librerías multimedia de comunicación y utilidades VoIP que implementa protocolos basados en estándares como SIP, SDP, RTP, STUN, TURN, y el ICE. Combina el protocolo de señalización, SIP, con el manejo de flujos multimedia y funcionalidades NAT traversal. Además, es portátil y adecuado para casi cualquier tipo de sistemas que van desde equipos de escritorio, sistemas embebidos o teléfonos móviles. El código fuente PJSIP se encuentra disponible en el directorio *archivos/pjproject1.14* del CD adjunto.

PJSIP trata de prestar a los desarrolladores todo lo necesario para construir aplicaciones de comunicación multimedia en tiempo real en lenguaje de programación C. Puede ser ejecutado prácticamente desde cualquier tipo de sistema, como Windows, Linux o sistemas Unix entre otros.

La estructura de PJSIP consiste en múltiples niveles de API's (véase figura 8), cada una de estas capas se encuentra una encima de otra y relacionadas entre sí. La más básica de todas las capas (PJLIB) comienza definiendo sus propios tipos de datos y estructuras.

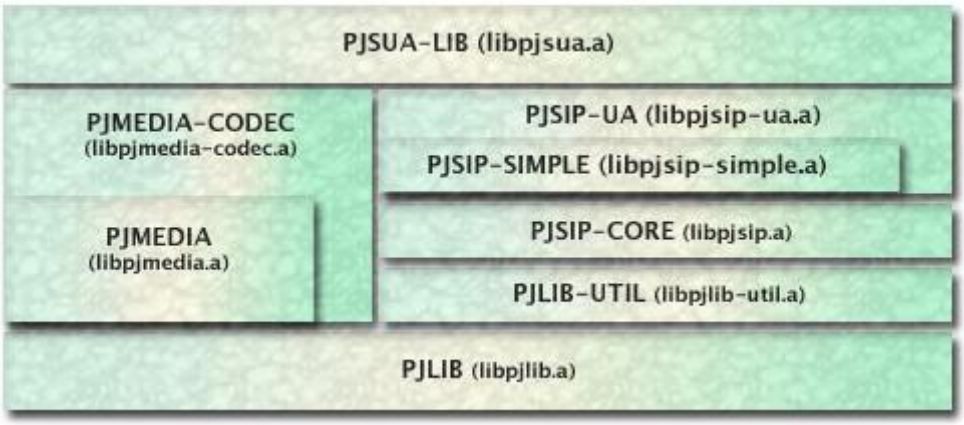

FIGURA 8: LAYOUT PJSIP

PJSIP incluye un potentísimo cliente SIP en línea de comandos llamado PJSUA (véase figura 9) que nos permite realizar llamadas a través de la red. PJSUA es una API de alto nivel para la construcción de agentes de usuario SIP multimedia. Engloba señalización y funciones multimedia en una aplicación fácil de usar. Sus principales características son:

- Administración de múltiples cuentas
- Administración de amigos
- Mensajería instantánea
- Múltiples llamadas

- Retención y transferencia de llamadas
- Conferencias
- Auto-responder con un fichero de audio
- Grabación de voz

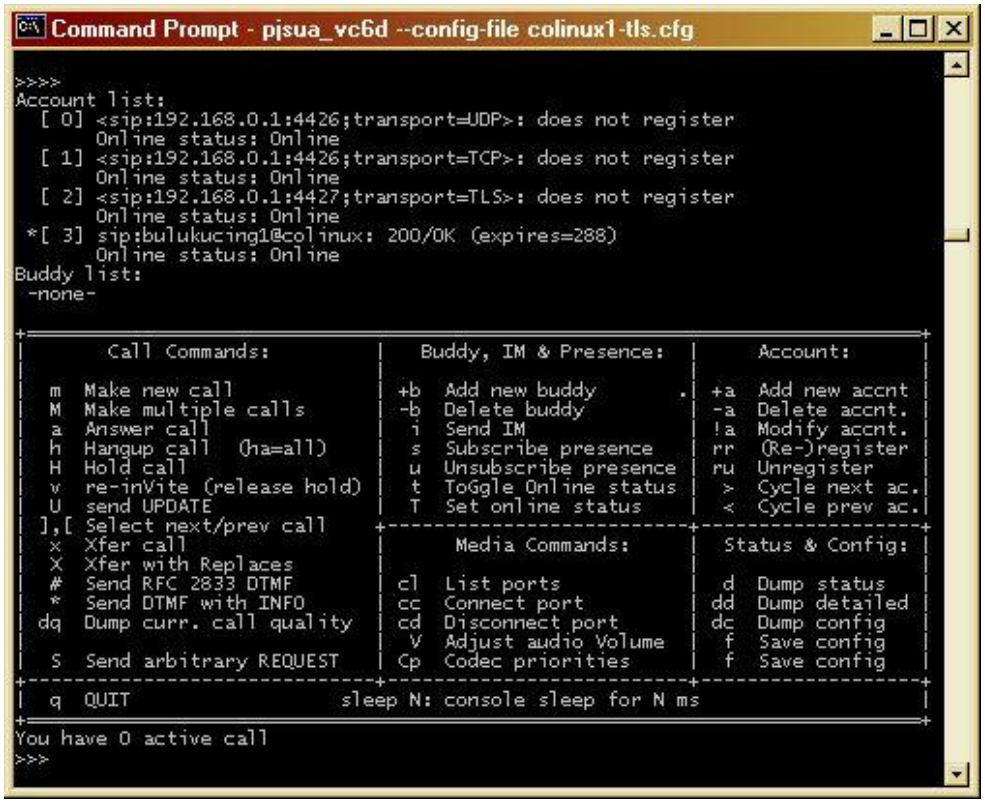

FIGURA 9: CONSOLA DE COMANDOS PJSUA

Las partes implicadas en una llamada son siempre dos agentes de usuario clientes. La funcionalidad requerida sería la siguiente:

- Establecer una llamada: Un cliente deberá poder enviar una solicitud de llamada enviando una invitación a otro cliente, conociendo la dirección IP de éste. También debe poder aceptar una invitación recibida.
- Mantener una conversación: Debe cumplir con la funcionalidad necesaria para el mantenimiento de una conversación, que consiste simultáneamente en:
    - Capturar muestras del micrófono del terminal, codificarlas, formar un paquete agrupando varias de ellas y enviarlas a través de la red.
    - Recibir paquetes a través de la red, decodificar las muestras presentes en éste, y reproducirlas por el altavoz del dispositivo.
- Finalización de una llamada: Cualquiera de los participantes en una llamada debe poder finalizar la misma en el momento en el que lo desee.

- Rechazar una llamada: Un cliente que esté recibiendo una llamada no tiene por qué aceptar la misma, pudiendo rechazarla y comunicándoselo al cliente que la solicitó.
- Cancelación de una llamada no establecida: Si un cliente invita a otro a una llamada, tendrá la posibilidad de cancelar esta invitación antes de que la misma sea aceptada.

De las capas definidas anteriormente, la de mayor interés de esta aplicación se corresponde con el módulo PJMEDIA, donde se implementa toda lo referente a la parte de los flujos de datos multimedia. Además, desarrolla tanto protocolos de señalización como SIP o SDP, como los protocolos de transporte RTP/RTCP y UDP.

La gestión del flujo de tramas de voz dentro del módulo PJMEDIA puede observarse en la figura 10, donde además se detallan los archivos de código fuente que implementan cada uno de los módulos necesarios para gestionar el flujo multimedia.

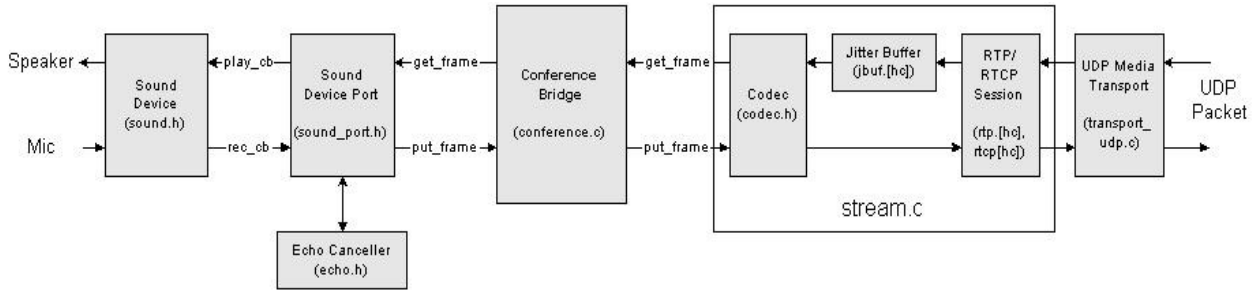

FIGURA 10: FLUJO MULTIMEDIA