2 Estructura y limitaciones en un cliente de VoIP.

La calidad de servicio es la principal motivación para la realización de este proyecto fin de carrera. Con la QoS podemos conseguir minimizar el ancho de banda conservando la calidad en la conversación, pero esto no podemos conseguirlo a menos que comprendamos bien la estructura interna de un cliente de VoIP, así como sus limitaciones a la hora de la realización y desarrollo.

2.1 Estructura genérica de un cliente de VoIP.

La idea de transmitir voz y vídeo a través de Internet no es nueva. Los primeros experimentos para la transmisión de paquetes de voz sobre tramas de red se remontan a los años 70, en esta misma época, en 1977, se desarrolla la primera RFC sobre la transmisión de voz sobre protocolos de red, NVC *(network voice protocol)*[4]. El audio fue lo primero en lo que se pensó para ser trasmitido por la red, ya que ocupaba menor ancho de banda que el resto del contenido multimedia.

Poco tiempo después un grupo de investigadores propondrían añadir el vídeo a la transmisión por la red, aunque la tecnología sea relativamente nueva ha generado una experiencia en el sector de al menos 10 años.

El proyecto NVC fue pensado en sus inicios para ser transmitido sobre la red ARPANET, el predecesor de Internet. ARPANET tenía una torre de protocolos análoga al TCP/IP, pero su debilidad consistía en introducir demasiado retardo, tanto que hacia que los paquetes se volvieran incontrolables, haciendo muy difícil o casi imposible el desarrollo de un servicio adecuado al actual modelo de transmisión de datagramas UDP/IP usado en RTP (*real time protocol*).

Todos estos experimentos estaban limitados a uno o dos canales de voz, ello sin tener en cuenta la limitación del bajo régimen binario que presentaban las redes de datos en sus inicios. Fue entonces cuando en 1980 la creación de un sistema de transmisión capaz de alcanzar los 3Mbps, la entonces denominada *wideband satellite network,* abrió la puerta no solo al uso de múltiples canales de voz sino que permitió el desarrollo de la transmisión de paquetes de vídeo.

En 1989-1990, la *satellite network* fue reemplazada por una red de banda ancha territorial (*Terrestrial Wideband Network)* denominada DARTNET y la ST (*satellite network*) fue llamada ST-II. Mientras tanto el sistema de videoconferencia fue puesto en producción para ser capaz de soportar reuniones desde diferentes localizaciones geográficas de forma simultánea.

En Marzo de 1992,durante la celebración de un congreso de IETF, se consiguió transmitir la

4 D. Cohen. "Specifications for the Network Voice Protocol," Network Working Group, RFC 741, November 1977

voz a través de Internet a 20 puntos geográficos distribuidos en 3 continentes, usando túneles multicast, conocido esto como "multicast backbone". Siendo este congreso el comienzo del desarrollo del protocolo RTP.

En toda comunicación siempre debe de existir un emisor y uno o varios receptores, RTP es el protocolo encargado de transportar los paquetes de voz de un extremo a otro, en definitiva lo que proporciona RTP es una capa de transporte para el contenido multimedia independientemente de la aplicación.

En la siguiente figura podemos observar la arquitectura de red para la transmisión de contenidos multimedia a través de Internet.

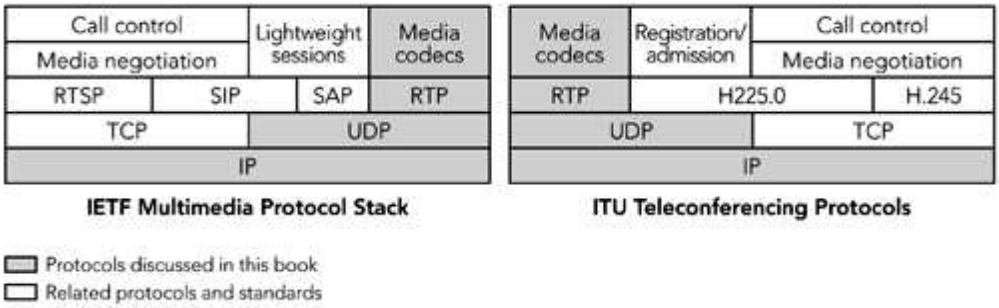


Ilustración 5: Protocolos para el transporte de audio/vídeo en Internet

El emisor es el responsable de capturar y transformar la información audiovisual para la correcta transmisión, así como el encargado de generar los paquetes RTP correspondientes. Incluso a veces el emisor puede participar en la corrección de errores y en el control de la congestión adaptando el flujo de transmisión a las condiciones de la red utilizando los informes que le envía el receptor.

En el siguiente diagrama se detalla el proceso que debe seguir todo emisor para transmitir el audio/vídeo a través de Internet.

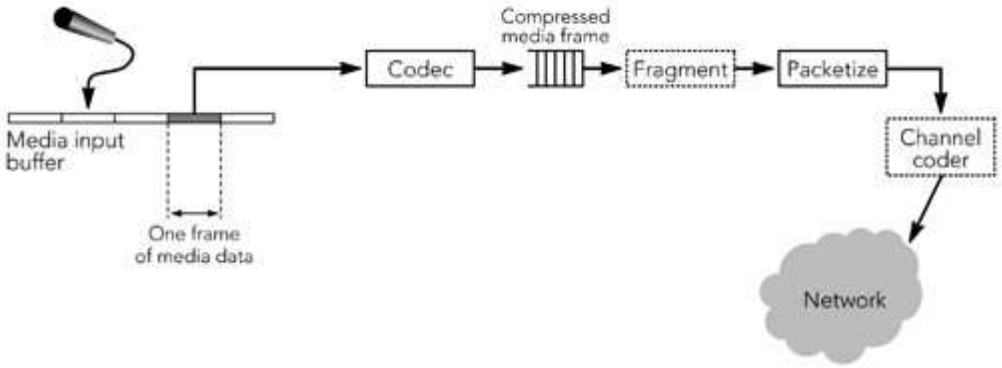


Ilustración 6: Diagrama del Emisor RTP

Como podemos observar, los datos de audio y vídeo son capturados y almacenados en un buffer a la espera de ser codificados, cada trama de audio/vídeo puede ser codificada de diferentes maneras dependiendo del algoritmo utilizado para dicho fin.

Una vez que las tramas han sido codificadas estamos en disposición de encapsular dichas tramas en paquetes RTP listos para ser enviados por la red. Si las tramas son demasiado grandes pueden ser fragmentadas en varios paquetes RTP, y viceversa si las tramas son muy pequeñas podemos introducir varias tramas de voz codificada en un solo paquete RTP.

El emisor es responsable de generar periódicamente informes sobre el estado del flujo multimedia creado, incluyendo todos aquellos que sean necesarios para mantener la sincronización de la comunicación.

Además recibirá informes de calidad de la comunicación procedente de otros miembros involucrados en la conversación permitiendole al mismo adaptarse al comportamiento fluctuante de la red.

El receptor es el responsable de recolectar todos los paquetes RTP recibidos a través de la red, corrigiendo cualquier pérdida, ajustando los tiempos, descomprimiendo las tramas multimedia y presentando todos los resultados al usuario. Incluso algunas veces el propio receptor transmite pequeños paquetes de calidad de servicio, que van a permitir al emisor adaptarse a la transmisión del receptor y poder mantener así una correcta base de datos de los participantes en la sesión de comunicación.

En la siguiente figura, se puede ver el diagrama de bloques pertenecientes a un receptor.

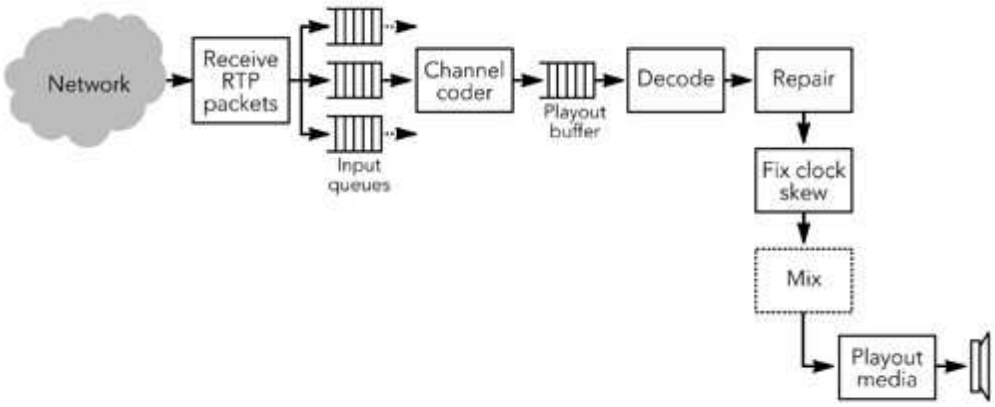


Ilustración 7: Diagrama de bloques para un receptor RTP

El primer paso del proceso es la recepción de los paquetes RTP desde la red, una vez que han sido correctamente validados, son introducidos en una de las colas de llegada de paquetes correspondientes a cada uno de los emisores, a continuación cada uno de los paquetes es decodificado mediante el algoritmo adecuado, algunos incluso son capaces de corregir las perdidas que durante la transmisión se hayan producido.

Una vez que los paquetes han sido decodificados, son introducidos en el buffer de playout ordenados según el timestamp de dicho paquete, éstos esperan en el buffer hasta que les toca su turno de reproducción, con estos buffer se consiguen corregir los jitter (retrasos entre emisor y receptor, producto de la congestión en la red) siendo este valor de espera uno de los puntos más críticos en el diseño de una aplicación de VoIP con RTP.

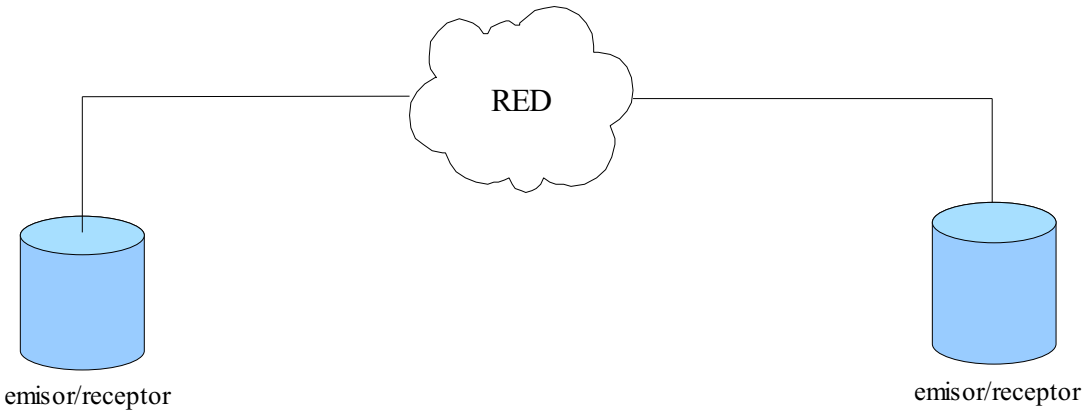
Antes de ser enviados al sistema de audio, los paquetes son reparados, siempre y cuando fuera necesario, y posteriormente se ajusta el reloj para mantener la correcta sincronización en la comunicación, es lo que comunmente se conoce como skrew.

Finalmente el audio/vídeo son reproducidos y presentados al usuario, siendo habitual su presentación de forma individual, aunque podría realizarse una mezcla del audio y el vídeo antes de ser enviado definitivamente al usuario.

Como podemos observar la funcionalidad del receptor es más compleja que la realizada por el emisor, ya que el receptor es el encargado de solventar la variabilidad de las redes IP, tratando de compensar la perdida de paquetes y corregir los retrasos en la medida de lo posible, siendo esto una tarea más compleja y por tanto de mayor dificultad.

En el desarrollo de un cliente de comunicaciones para VoIP no solo se va a tener que realizar un emisor, sino que al mismo tiempo conviven tanto el emisor como el receptor.

Durante cualquier conversación hay momentos en los que se escucha y momentos en los que se habla, es decir, por la propia naturaleza de toda comunicación cada uno de nosotros es tanto un emisor como un receptor.



Cada emisor/receptor contiene un esquema de comunicación igual que el mostrado en las ilustraciones Diagrama del Emisor RTP y Diagrama de bloques para un receptor RTP, cada uno de estos diagramas será implementado en cada cliente de VoIP.

Para que se pueda organizar y sincronizar la comunicación entre los participantes, antes se debe de poder llamar, es decir, cuando se desea ponerse en contacto con una persona determinada, lo primero que se debe hacer es conocer su número de teléfono, es la pieza básica para localizar y establecer una comunicación con él. Una vez que se conoce su identificador en la red telefónica, se realiza una llamada y con el descuelgue del interlocutor del otro extremo es cuando se comienza a transmitir la voz. En las redes IP, Internet, el esquema de funcionamiento es muy parecido, por no decir igual salvo por el medio de transporte de las tramas de voz.

Internet no es más que una abstracción de un conjunto de redes interconectadas entre si, a lo largo del mundo. La siguiente ilustración os va a servir para que os hagáis una idea mejor de lo que se está hablando.

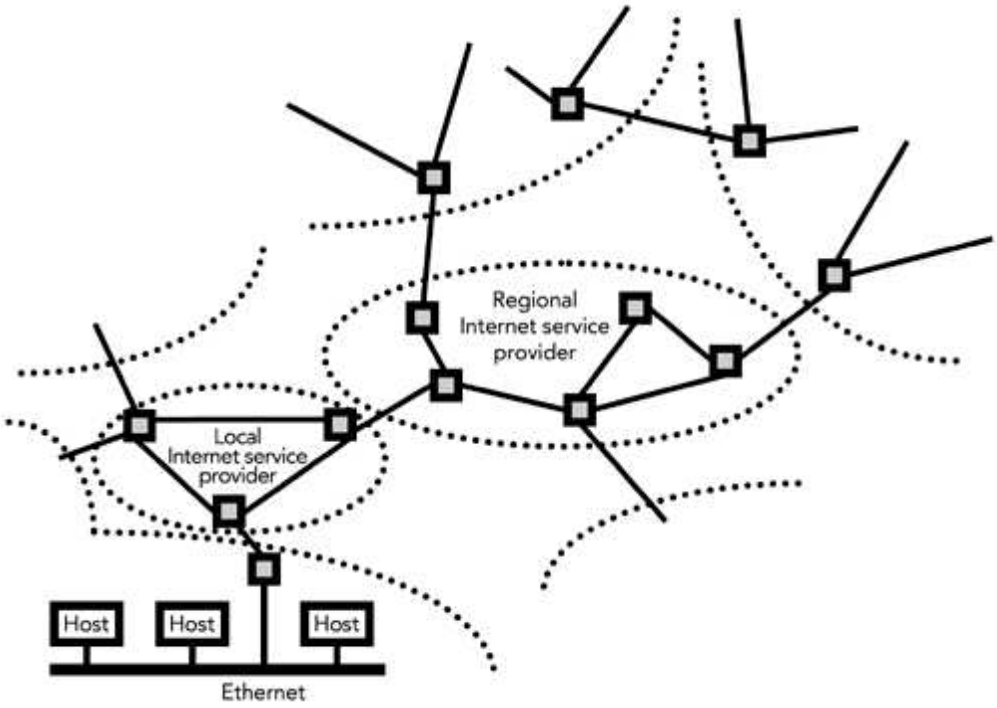


Ilustración 8: Internet, conjunto de redes interconectadas entre si.

El direccionamiento IP es el equivalente al número de teléfono en Internet, es decir, mediante una dirección IP se puede establecer contacto con cualquier persona del mundo. Pero en la telefonía por Internet, no vamos a utilizar un identificador tan complicado y dificultoso de memorizar.

A lo largo de los años han surgido dos protocolos dominantes estos son H.323 y SIP, actualmente estos protocolos son conocidos como de *señalización o de control*, son los encargados de buscar a la persona y hacer los pasos necesarios para que dichas maquinas o dispositivos se comuniquen.

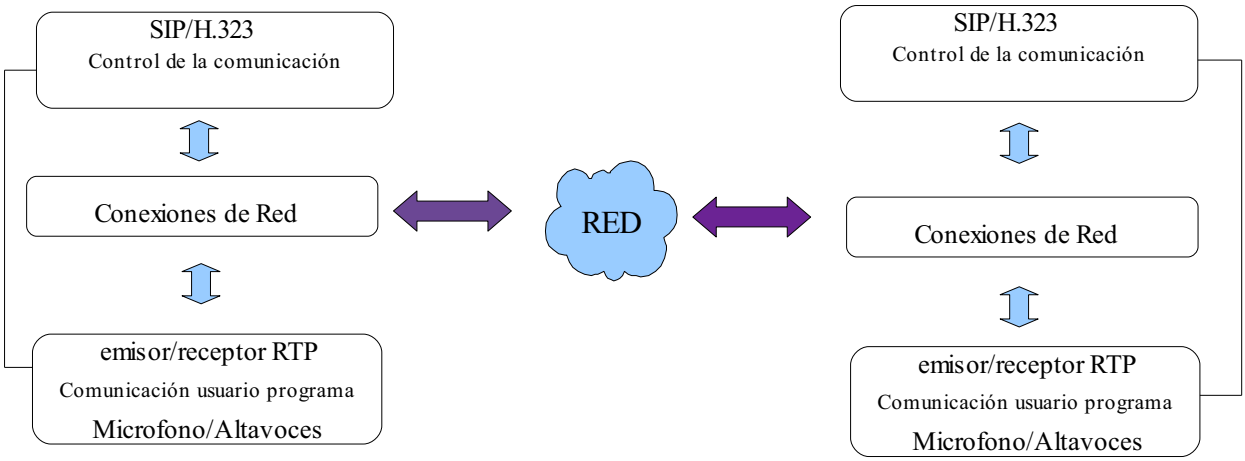
SIP (**Protocolo de Inicialización de Sesiones**) es un protocolo desarrollado por el IETF MMUSIC Working Group con la intención de ser el estándar para la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como el vídeo, voz, mensajería instantánea, juegos online y realidad virtual. SIP es uno de los protocolos de señalización para voz sobre IP, acompañado por H.323. Uno de los objetivos de SIP fue aportar un conjunto de las funciones de procesamiento de llamadas y capacidades presentes en la red pública conmutada de telefonía.

SIP es usado simplemente para iniciar y terminar llamadas de voz y vídeo. Todas las comunicaciones de voz/vídeo van sobre RTP (*Real-time Transport Protocol*).

SIP actúa como envoltura al SDP, que describe el contenido multimedia de la sesión, por ejemplo qué puerto, que IP y que códec se usarán durante la comunicación, etc. En un uso normal, las sesiones SIP son simplemente flujos de paquetes RTP (*Real-time Transport Protocol*). Siendo RTP el portador para el actual contenido de vídeo y audio.

H.323 es una recomendación del ITU-T (International Telecommunication Union), que define los protocolos para proveer sesiones de comunicación audiovisual en cualquier paquete de la red. Como curiosidad histórica podemos decir que H.323 fue el primer estándar de VoIP en adoptar el estándar de IETF el RTP (Protocolo de Transporte en tiempo Real) para transportar audio y vídeo sobre redes IP.

Por tanto un cliente de VoIP no se va a limitar a ser un emisor/receptor sino que además dispondrá de un módulo de control, para la inicialización, control y finalización de las llamadas usando el protocolo SIP o H.323.



Control de la comunicación: En todo programa de VoIP (telefonía *por Internet*) se necesita una serie de primitivas o funciones que nos permitan establecer, mantener y finalizar una llamada de voz. Este módulo se encarga precisamente de la implementación de las funciones que realizan dichas operaciones.

El protocolo SIP o H.323 son los definidos como los estándares a seguir, es por ello que en la amplia mayoría de clientes de voz sobre IP se basan en dichas reglas.

Antes de poder realizar una comunicación de VoIP es necesario localizar al interlocutor, para ello en la inicialización de nuestro cliente nos registraremos en un servidor que nos facilitará un identificador único con el que se nos puede localizar. Todo el diálogo para realizar dicho registro entre el cliente y el servidor va a ser realizado mediante el protocolo SIP.

Durante el establecimiento de una llamada se decide que códec utilizar para la comunicación de voz, eligiéndolo entre los posibles candidatos de cada uno de los clientes, es decir, ambos clientes deben de ponerse de acuerdo sobre cuál es el códec que se va a usar. Ni que decir tiene que ambos clientes deben de tener implementado el mismo códec. Por supuesto todo esta negociación queda perfectamente contemplada en la norma SIP.

La finalización de una llamada no puede acabar de cualquier forma, debe de seguir unas normas o estándares recogidos en el protocolo SIP.

El uso de protocolos estandarizados nos facilitará la intercomunicación entre clientes, es decir, cualquier cliente de comunicaciones que implementase el estándar SIP sería capaz de dialogar y entenderse con cualquier otro cliente que cumpla dichos estándares.

Por tanto, podemos observar que este módulo de control de la comunicación es importante y necesario para el correcto funcionamiento del cliente.

Conexiones de Red: Cualquier cliente de comunicación debe disponer de la implementación de las primitivas o funciones que nos permitan transmitir datos a través de la red.

Este módulo implementa las comunicaciones tanto TCP/UDP, mediante este conjunto de funciones el programa puede enviar un datagrama a otra maquina de la red, y por tanto permitir que comiencen a dialogar y negociar la conexión del flujo de Voz, que a su vez va a necesitar de dichas funciones para comunicarse.

Este módulo es la arteria principal de cualquier programa de comunicaciones, es el encargado de transmitir los datos por la red, y permitir al programa comunicarse con otros clientes a través de Internet. Es la implementación de las capas bajas de la arquitectura de red del modelo OSI.

Este módulo implementa los niveles físico, enlace, red y transporte, de forma que cualquier aplicación necesita dichas capas para poder comunicarse en red.

Comunicaciones usuario/programa: Este módulo es la parte correspondiente a los niveles altos del modelo OSI, es el nivel de aplicación. Va a ser el encargado de presentar la interfaz de usuario con la que interactuar.

Dicho módulo será el encargado de dirigir toda la comunicación, es decir:

- establecimiento de la llamada
- preparación del micrófono para la captura de datos
- recepción y reproducción de las tramas de voz recibidas
- finalización de la llamada
- etc.

Es por tanto el módulo del programa que se implementa como un nivel superior al de control y conexiones de red, es decir será el encargado de usar las funcionalidades aportadas por los otros módulos para establecer la llamada, abrir el flujo de comunicación de voz y la finalización del mismo.

La implementación de un cliente de VoIP puede describirse genéricamente con los módulos aquí definidos.

2.2 Implementaciones en Software Libre.

El objetivo principal del proyecto es la implementación de una solución de VoIP modificando una de las que ya existen y adaptándola a nuestros propósitos.

Debido al gran auge de las comunicaciones VoIP son muchas las soluciones de telefonía que tenemos a nuestra disposición, en el proyecto fin de carrera se ha optado por soluciones basadas en software libre que nos permitan la disponibilidad de consultar y modificar el código fuente a nuestra voluntad para alcanzar así los objetivos establecidos.

La elección del programa que se desea utilizar como base para el proyecto fin de carrera no ha sido una tarea fácil, debido sobre todo a la gran variedad de programas que podemos encontrar.

A continuación se va a enumerar una serie de programas basados en software libre que nos permite realizar llamadas VoIP:

- Kphone v4.2 (*http://www.wirlab.net/kphone/*).
- Twinkle 0.81 (*http://www.twinklephone.com/*).
- Ekiga (*http://www.ekiga.org/*).
- Gizmo (*http://www.gizmoproject.com/*).
- GnoPhone (*http://www.gnophone.com/*).
- Cphone (*http://cphone.sourceforge.net/*).
- LinPhone (*http://www.linphone.org/*).
- Openwengo (*http://www.openwengo.com/*).
- OpenH323 (http://www.openh323.org/).
- PhoneGaim (http://www.phonegaim.com/).
- Tapioca (http://tapioca-voip.sourceforge.net/wiki/index.php/Tapioca).
- ....

De entre todos los aspirantes se decidió comenzar a trabajar con un software básico y sencillo, que tuviera unas características mínimas y una interfaz gráfica simple.

Es por ello que se eligió Kphone como el programa de software por el que iniciar la implementación del cliente de VoIP con QoS.

2.3 Limitaciones de los Clientes Analizados.

La realización de este proyecto comenzó con el análisis del cliente de VoIP "Kphone", pero más tarde se comprobó que este cliente contenía ciertas limitaciones que complicaban el desarrollo normal del proyecto y se decidió analizar otros clientes de entre los cuales se eligío Twinkle. En este apartado se va a exponer las limitaciones encontradas en ambos clientes.

2.3.2 Limitaciones del cliente Kphone

Kphone fue el primer cliente de software que se ha analizado y estudiado para el desarrollo de este proyecto. La implementación de un cliente de VoIP no es una tarea sencilla, contiene múltiples protocolos y RFC's que tienen que ser implementadas cumpliendo todas las características de dichas normas, para garantizar la mayor compatibilidad posible.

Kphone esta desarrollado en C++ y da la impresión de haberse ido construyendo sin tener en cuenta el proceso típico de desarrollo de aplicaciones de software, es decir, es como si poco a poco se hubieran ido añadiendo partes de código al programa y adaptando este para utilizar las nuevas funcionalidades, en vez de concebir la aplicación desde sus inicios y seguir las etapas para su implementación, diseño e implementación.

Kphone contiene varias partes a diferenciar:

- ✔ Implementación propia de protocolo SIP y RTP.
- ✔ Librerías ALSA para comunicación con el sistema de audio.

Cuando se va a desarrollar un cliente de comunicaciones de VoIP es muy importante plantearse su desarrollo desde la programación multihilo, es por ello que la principal limitación en Kphone es la realización de emisor/receptor RTP en tan solo dos hilos de ejecución.

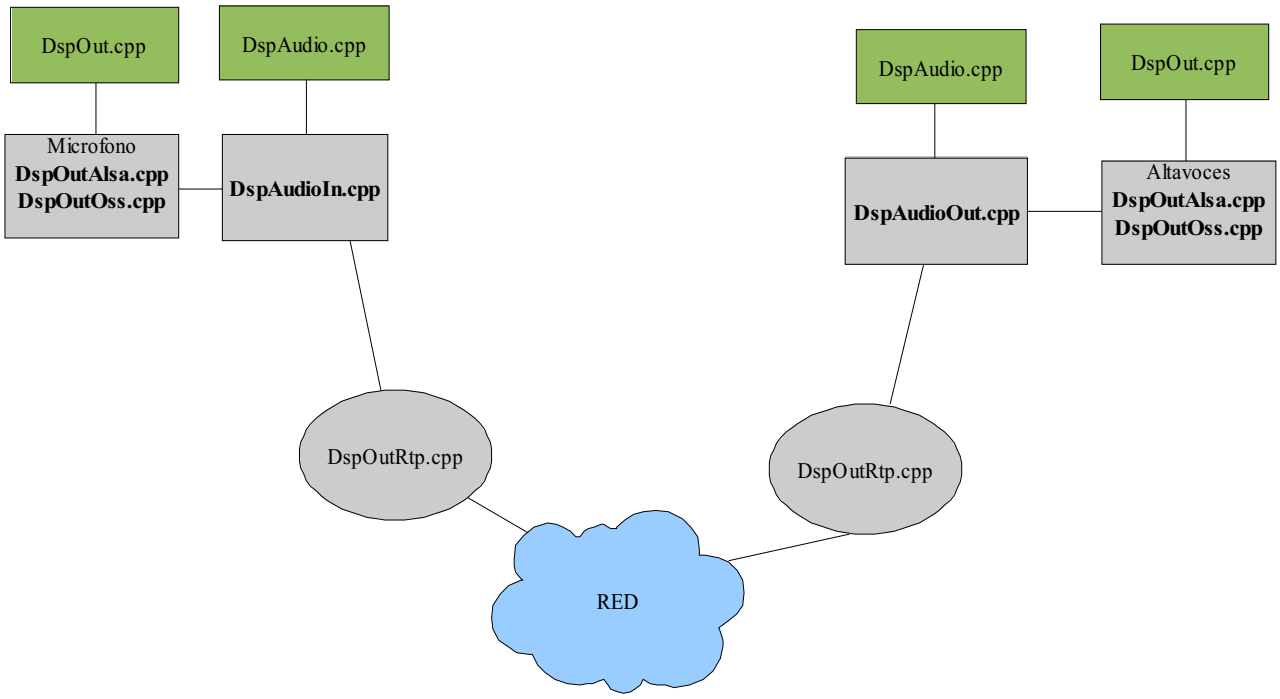
Al analizar la parte correspondiente a la comunicación entre micrófono/altavoz y la red de datos, se comprobó una de las limitaciones que se va encontrar a la hora de realizar un programa de VoIP.

En el apartado 2.1 se ha realizado un repaso sobre la estructura genérica que debe de tener cualquier programa de comunicaciones VoIP, en el caso de Kphone no cumplía estrictamente con dicha estructura.

El flujo del programa si coincidía con la estructura esperada para un cliente de VoIP, pero la

implementación ha sido descuidada permitiendo esto la aparición de la primera limitación encontrada en el cliente Kphone. Esta limitación se centra en la programación multihilos y la comunicación del flujo de datos proveniente de la red y el flujo de datos que debe ser entregado al sistema de altavoces.

En la siguiente ilustración se puede observar la estructura de comunicación entre las librerías de audio y las funciones de red.



Ficheros y estructura de comunicación del cliente Kphone .

En un sistema de comunicaciones de Voz, uno de los puntos más importantes es reproducir los paquetes recibidos en los instantes de tiempo correspondientes, para esta tarea se ha visto que se utiliza el protocolo RTP para el encapsulado de tramas de voz codificadas y posterior transmisión por la red de datos.

Para una correcta recepción de los datos se debe de implementar el receptor dividiendo las tareas a realizar en dos partes:

1. Recepción de los paquetes RTP y procesamiento del paquete.
2. Reproducción de las tramas de voz en los instantes de tiempo correspondientes.

En el cliente Kphone el diseño de estas funcionalidades se realiza como una acción secuencial y

continuada en el tiempo, esta forma de implementar la reproducción de las tramas recibidas se convierte en una importante limitación a la hora de añadir y/o adaptar el código fuente para el soporte de la calidad de servicio (QoS). Esto es, para la realización de dichas tarea es necesario identificar por separado cada una de las partes que anteriormente se han enumerado.

En el programa Kphone la *limitación* consistía en haber implementado bajo un mismo hilo secuencial la recepción de los paquetes de red, su decodificación y su reproducción. Esto junto con el *error de diseño*[5] del programa hacia muy complicado añadir la parte de código que controlase adecuadamente la recepción y la reproducción de las tramas de voz para el soporte de calidad de servicio (QoS) necesario.

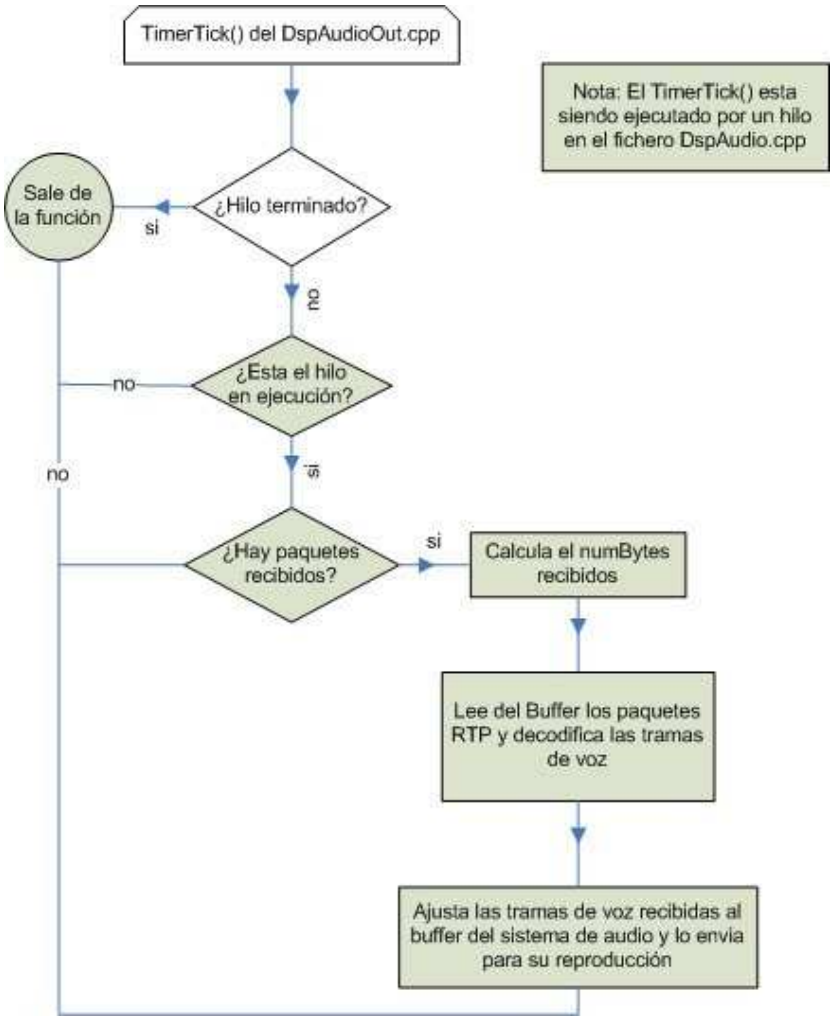


Diagrama de flujo de la recepción de tramas de audio en Kphone.

En la ilustración anterior se observa el diagrama de flujo correspondiente al proceso de

5 Error al no considerar el programa desde el principio multihilo, sino añadir dicha funcionalidad solamente para la parte correspondiente al procesamiento de los paquetes y las tramas de voz.

recepción, decodificación y reproducción de las tramas de voz. En dicha ilustración se puede ver la limitación así como la ausencia de un Buffer de Playout.

Está limitación hace que el programa Kphone ocupe un mayor ancho de banda y puedan descartarse un determinado número de paquetes de voz válidos, haciendo que la calidad de voz obtenida sea muy pobre cuando se establece una comunicación a través de Internet.

La realización de recepción de paquetes y posterior procesamiento de las tramas en un solo hilo corre el peligro de no reproducir las tramas de voz en el tiempo correcto. De hecho, si en el proceso se produce un retraso mayor del esperado, la voz llegará entrecortada y no se entenderá lo que el otro interlocutor esta tratando de decir.

Para corregir los retrasos de las tramas de voz se utilizan lo que se denominan *"buffer de playout"* y este buffer es el encargado de corregir esos retrasos haciendo posible mantener una comunicación fluida entre dos extremos de la red.

En el momento que se introduce un *"buffer de playout"* la implementación del cliente no es tan sencilla como recibir, procesar y reproducir. En este caso los paquetes se almacenan en el buffer a la espera de su turno de reproducción, haciendo necesario distinguir entre el proceso de recepción de paquetes de la red y la reproducción de las tramas de voz.

Kphone no distinguía estas dos partes, por lo que se nos presenta una limitación muy importante a la hora de desarrollar un cliente con calidad de servicio (QoS), de hecho, la no concepción del programa desde sus inicios como un programa multihilo hace prácticamente imposible añadir una estructura multihilo y adaptarla a nuestros propósitos. Se intento realizar la adaptación al sistema multihilos, pero el planificador de tareas no ejecutaba correctamente el control de tiempos entre los hilos en ejecución haciendo prácticamente imposible implementar por separado ambas tareas.

Por tanto para adaptar el código fuente del Kphone sería necesario una reestructuración desde el inicio del programa y desde su concepción, por estos motivos se paso a buscar un segundo programa que nos permitiera incorporar una estructura multihilo sin alterar su funcionamiento y por supuesto sin diseñarlo desde el inicio. El propósito de este proyecto no es la creación de un programa de VoIP sino la adaptación de un cliente de software libre ya existente.

Tras analizar el cliente Kphone se llegó a la conclusión que no era un buen candidato para añadir la calidad de servicio, entendiéndose por tal:

- Control dinámico de los buffer de playout
- Empaquetado de tramas de voz
- Codificación adaptativa a la situación

Más adelante se profundizará en estos conceptos, concretamente en la calidad de servicio, en que consiste y como se implementa.

Comprobado que Kphone no era el candidato, se propuso un segundo aspirante en este caso llamado *twinkle.*

2.3.3 Limitaciones del cliente Twikle

Twinkle[6] es un cliente de VoIP diseñado y concebido como una aplicación multihilo, estructurada desde su base como un hilo más dentro del diseño. De este modo disponemos de una estructura que divide el programa en subprocesos, todos independientes y comunicados gracias a un sistema de colas que hacen posible el intercambio de información entre los hilos de manera cómoda y segura.

Desde el comienzo *twinkle* presentaba una estructura ideal para la implementación de la QoS en el cliente, tal y como viene siendo el propósito de este proyecto.

Twinkle parte con cierta ventaja:

- La torre de protocolos RTP estaba implementada usando una librería opensource, concretamente ccRTP, esto permite disponer de un código fuente robusto pero a su vez menos flexible y modificable.
- Las librerías de audio han sido implementadas como una estructura independiente.

Rápidamente podemos intuir cual va a ser la limitación para este cliente de VoIP, la librerías utilizadas para proporcionar el soporte RTP.

Estas librerías crean un marco de limitación para la manipulación de los paquetes. El empaquetado es una tarea que se aplica directamente sobre la torre de protocolos RTP y es por ello que supondrá una pequeña limitación, pues el procesamiento de los paquetes se van a realizar a un nivel por encima del deseado, sin poder modificar ciertos campos de datos de la cabecera RTP, pero que a pesar de no disponer de esa posibilidad, la implementación de la calidad de servicio (QoS) se puede realizar tan solo teniendo en cuenta algunas condiciones que se deben de cumplir, y que más adelante serán tratadas.

La gran limitación en cualquier programa de comunicaciones de VoIP es realizar una correcta reproducción de las tramas de voz, insertando en los instantes de tiempo adecuados las tramas de voz correctas. La implementación de las librerías que manipulan el sonido presentan un gran

6 www.twinklephone.com

obstáculo a superar.

Por ello, comprender correctamente el concepto de establecer y mantener una comunicación por VoIP puede ser fundamental para superar la limitación de sincronización con los hilos que realizan la recepción y reproducción de las tramas de voz.

Twinkle, presenta una estructura ideal para la implementación de la QoS en el cliente de VoIP, no obstante se debe de tener en cuenta que para la sincronización entre hilos es necesario el ajuste adecuado de los tiempos en los que cada hilo debe de suspenderse y permitir de este modo que el planificador del sistema adjudique correctamente los tiempos de ejecución a los mismos, siendo esto fundamental para el desarrollo de un cliente con calidad de servicio.

Es precisamente la sincronización y ajuste de tiempos entre los procesos del programa, lo que representa la *gran limitación* de cualquier cliente de VoIP.

Para finalizar, antes de comenzar a implementar la calidad de servicio en un programa de VoIP, se tiene que estar seguro de contar con un programa concebido para ser multihilo, es decir, diseñado desde el inicio como un cliente capaz de ejecutarse en varios hilos y por tanto ser capaz de adaptar la ejecución de un nuevo hilo sin más que crearlo, configurarlo y ejecutarlo.

En este apartado se ha resumido brevemente algunas de las limitaciones encontradas y que serán analizadas en profundidad en el apartado 4.4 de esta memoria.