# Sistema de voz sobre IP para redes inalámbricas en zonas rurales aisladas

**Rodrigo de Salazar**
**Departamento de Ingeniería y Sistemas Telemáticos de la Universidad Politécnica de Madrid**

**Sistema de voz sobre IP para redes inalámbricas en zonas rurales aisladas**

por Rodrigo de Salazar

Publicado 22 de Mayo de 2005
Copyright © 2005 UPM

En este proyecto se estudia un sistema para integrar un sistema de comunicación de voz de bajo costo sobre redes privadas inalámbricas tipo 802.11, desplegadas en zonas rurales aisladas de países en desarrollo.

El uso de este documento se rige por la licencia Attribution-ShareAlike de Creative Commons. Para ver una copia de la licencia, visite http://creativecommons.org/licenses/by-sa/1.0 o pídala por correo a Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

# Tabla de contenidos

**1. Introducción ..... 1**

1.1. Objetivo específico ..... 1

1.2. Estructura del proyecto ..... 1

**2. Análisis ..... 3**

2.1. Servicios deseables ..... 3

2.1.1. Voz ..... 4

2.1.2. Videoconferencia ..... 4

2.1.3. Charla interactiva ..... 4

2.1.4. Correo electrónico y web ..... 5

2.1.5. Protocolos de terminal ..... 5

2.1.6. Acceso a ficheros remotos ..... 5

2.2. Características geográficas ..... 5

2.2.1. Selva ..... 6

2.2.2. Sierra ..... 8

2.3. Características de hardware ..... 9

2.4. Características del software y los protocolos ..... 10

**3. Voz sobre IP ..... 12**

3.1. Arquitectura de redes VoIP ..... 12

3.2. Códecs ..... 15

3.2.1. Audio ..... 15

3.2.2. Vídeo ..... 18

3.3. RTP/RTCP ..... 19

3.4. Norma H.323 ..... 22

3.4.1. Red básica H.323 ..... 23

3.4.2. Red H.323 con GateKeeper ..... 27

3.4.3. Elementos avanzados ..... 29

3.4.3.1. Pasarelas a la Red Telefónica Conmutada (RTBC) ..... 29

3.4.3.2. Intermediarios H.323 ..... 30

3.4.3.3. Unidades de Control Multipunto ..... 30

3.4.3.4. Firewalls ..... 31

3.4.3.5. Mecanismos de NAT/PAT ..... 32

3.5. Protocolo SIP ..... 32

3.5.1. Llamada básica en SIP ..... 33

3.5.2. Elementos adicionales de la arquitectura SIP ..... 34

3.5.2.1. Llamada con intermediario ..... 35

3.5.2.2. Llamada con redirección ..... 36
3.5.2.3. Registro de localización ..... 36
3.5.3. Elementos avanzados ..... 37
3.6. Comparación entre H.323 y SIP ..... 37
3.7. Implementaciones ..... 39
3.7.1. Productos software libres ..... 39
3.7.1.1. H.323 ..... 39
3.7.1.2. SIP ..... 40
3.7.1.3. Otros ..... 41
3.7.1.4. Terminales software propietarios ..... 41
3.7.1.5. Terminales hardware ..... 42
3.7.2. GateKeepers, servidores SIP y pasarelas ..... 43
3.7.3. Estudio de las aplicaciones H.323 más conocidas ..... 43
3.7.3.1. Netmeeting ..... 43
3.7.3.2. Gnomemeeting ..... 44
3.8. Calidad de servicio en VoIP ..... 45
3.8.1. Calidad de servicio en IP ..... 46
3.8.2. Aplicación a nuestro caso ..... 46
**4. Batería de pruebas ..... 51**
4.1. Diseño de las pruebas ..... 51
4.2. Herramientas ..... 54
4.3. Implementación y automatización ..... 55
4.3.1. Banco de pruebas ..... 55
4.3.1.1. Banco inalámbrico básico ..... 57
4.3.1.2. Banco inalámbrico con teléfonos VoIP ..... 58
4.3.2. Pruebas con iperf ..... 60
4.3.2.1. Pruebas objetivas ..... 60
4.3.2.2. Pruebas subjetivas ..... 63
4.3.3. Casos de estudio ..... 63
4.4. Resultados ..... 69
4.4.1. Pruebas objetivas ..... 69
4.4.1.1. Prueba sin conformado de tráfico ..... 71
4.4.1.2. Prueba con disciplina de colas PRIO ..... 73
4.4.1.3. Prueba con disciplina de colas HTB priorizando UDP ..... 75
4.4.1.4. Prueba con disciplina de colas HTB priorizando la voz ..... 76
4.4.1.5. Prueba con disciplina DSMARK con ancho de banda compartido ..... 78

4.4.1.6. Prueba con disciplina DSMARK sin compartir ancho de banda ..... 80
4.4.2. Pruebas subjetivas ..... 82
4.5. Conclusiones ..... 84
**5. Diseño ..... 86**
5.1. Los nodos ..... 86
5.1.1. Hardware ..... 86
5.1.1.1. Versión de pruebas ..... 87
5.1.1.2. Versión de campo ..... 88
5.1.1.3. Subsistema solar ..... 89
5.1.2. Software ..... 90
5.1.3. Autoconfiguración de la red ..... 90
5.1.4. Implementación de QoS en los nodos ..... 91
5.2. Los terminales ..... 93
5.2.1. Terminal único ..... 94
5.2.2. Computación separada ..... 94
5.3. Las pasarelas ..... 95
5.3.1. Pasarela única ..... 96
5.3.2. VoIP separada ..... 96
**6. Presupuesto ..... 97**
6.1. Presupuesto de contratación ..... 97
6.1.1. Coste de ejecución material ..... 97
6.1.2. Gastos generales ..... 98
6.1.3. Beneficio industrial ..... 98
6.2. Honorarios ..... 98
6.3. Presupuesto total ..... 98
**7. Bibliografía ..... 99**
Referencias ..... 99

# Lista de tablas

3-1. Tabla de códecs de audio ..... 17
3-2. Tabla de códecs de vídeo ..... 18
3-3. Comparación general entre H.323 y SIP ..... 37
6-1. Coste de ejecución material ..... 97
6-2. Fungibles ..... 97
6-3. Coste de ejecución material ..... 98
6-4. Presupuesto total ..... 98

# Capítulo 1. Introducción

Este proyecto se enmarca dentro del programa del Enlace Hispano Americano de Salud, EHAS (http://www.ehas.org)[MVS2002], cuyo objetivo general consiste en mejorar las condiciones de trabajo del personal sanitario rural de salud por medio de las tecnologías de la información y comunicación. EHAS ofrece a los agentes prestadores de servicios de salud soluciones de telecomunicación de bajo coste y un conjunto de servicios de telemedicina adaptados al entorno rural. Con los sistemas EHAS se pretende dotar a puestos y centros de salud rurales de al menos comunicación de voz y correo electrónico, incluso en zonas donde no haya teléfono ni fluido eléctrico. Sobre estos sistemas se construyen sistemas de capacitación a distancia del personal de salud, consultas remotas a especialistas, ayuda en el acceso a documentación sobre salud, sistemas de vigilancia epidemiológica, sistemas de pedido de medicamentos, etc.

Las tecnologías de telecomunicación utilizadas en EHAS hasta ahora se han basado principalmente en enlaces VHF que se utilizan para radiofonía local y para transporte de correo electrónico a la linea telefónica más cercana. Estas tecnologías se están extendiendo por un lado a banda estrecha HF, para abarcar grandes distancias en zonas muy aisladas, así como a banda ancha en microondas, en particular a IEEE 802.11b, para aquellos lugares donde se pueda conseguir visión directa entre los puntos a enlazar y las condiciones legales y tecnológicas lo permitan.

## 1.1. Objetivo específico

El objetivo específico de este trabajo es contribuir a definir e implementar una arquitectura para la prestación de servicios ordinarios de datos y de voz sobre IP utilizando las posibilidades de IEEE 802.11b en entornos rurales. Parte de esa arquitectura será genérica, apta para cualquier tipo de comunicación, pero compatible con un servicio de voz de calidad suficiente y prioritario, por lo que deben implementarse mecanismos de calidad de servicio (QoS). Así mismo, la solución estará, en la medida de lo posible, basada en estándares abiertos, ampliamente implementados, y/o en código abierto, fácilmente adaptable a situaciones cambiantes y apto para su apropiación y desarrollo por los actores interesados en trabajar en este campo.

## 1.2. Estructura del proyecto

Primeramente se examinarán las caracteristicas de los entornos de trabajo típicos, basándonos en zonas en las que ya se ha intervenido o donde hay proyectos pendientes de realizar, especificando el tipo de solución deseable. Esto se corresponde con una fase de análisis del problema a resolver.

Seguidamente se procederá a realizar una revisión de las tecnologías disponibles en el campo de la VoIP, para ello se analizará la arquitectura típica de una red que implemente un sistema de VoIP, así como los códecs habitualmente utilizados, protocolos de tiempo real a nivel de red y de transporte y por supuesto, las dos soluciones más utilizadas en el mundo de la VoIP, la norma H.323 y el protocolo SIP. Todo ello se hará principalmente desde una óptica orientada a evaluar las posibilidades que cada una de estas partes nos ofrecen a la hora de establecer nuestra política de calidad de servicio; una vez extraídas la conclusiones pertinentes, propondremos un esquema de priorización de tráfico que funcione, al menos sobre el papel, a la hora de garantizar dicha calidad.

Una vez examinadas estas tecnologías y extraídas las conclusiones y el sistema de priorización a seguir, se procederá a la realización de una serie de pruebas con distintos esquemas basadas en las posibilidades que nos ofrece el núcleo de Linux para la priorización del tráfico en una red, de forma que podamos evaluar cuál de ellas se acerca más a nuestros objetivos.

Por último se hará una propuesta de diseño de la solución a adoptar, desde un punto de vista tanto de componentes hardware como del software y configuración necesarios, teniendo en cuenta las conclusiones alcanzadas a lo largo del proyecto.

# Capítulo 2. Análisis

Enfocaremos el análisis en examinar las condiciones de trabajo en la cuales se han de proporcionar servicios realizables por medio de tecnologías maduras, económicas y con regulación liberal, basadas en 802.11b e IPv4, sin perder de vista alternativas emergentes, como son 802.16 e IPv6. En todo caso se supondrá que se desean enlaces permanentes en un ámbito geográfico determinado, que posibiliten todo tipo de comunicaciones interactivas, en especial las de voz, sin exigir los elevados anchos de banda de modernas aplicaciones multimedia.

## 2.1. Servicios deseables

Los sistemas basados en VHF y HF desplegados por EHAS sólo disponen de dos tipos de servicios: voz analógica, disponible 24 horas al día en el ámbito del alcance de la radio, y correo electrónico, de ámbito universal, sobre el que se montan los demás servicios, y accesible bajo demanda del usuario, que lo activa para los intercambios. Esto tiene dos efectos laterales deseables:

1. Que no esté la computadora constantemente encendida , ahorrando energía, muy valiosa donde sólo hay alimentación solar.
2. Que la comunicación más importante, la voz, no dependa de elementos frágiles, fácilmente desconfigurables por mal uso, virus, errores de software, etc.

Los servicios basados en microondas no deberían perder estas características, pudiendo ser realizados de dos maneras distintas:

1. Separando las funcionalidades más críticas en elementos preprogramados, de bajo consumo, como serían los de comunicación de datos (routers, convertidores ethernet-inalámbrico) y los de voz (teléfonos IP, alámbricos o inalámbricos), y la computadora, que sólo se enciende cuando se necesite.
2. Integrando al máximo todos los elementos en un sólo dispositivo de computación robusto, sencillo de usar y de muy bajo consumo cuando esté en espera. A ser posible también debe ser no administrable, como lo sería un terminal inalábrico sin almacenamiento local.

A continuación se enumeran y comentan, más o menos ordenados según su importancia, los servicios posiblemente deseables y sus características.

### 2.1.1. Voz

Como dijimos antes es el servicio más importante. Sus características más notables son:

1. Disponibilidad continua, por lo que es tentador ofrecerlo por medio de un dispositivo independiente, robusto, fácil de usar y de bajo consumo.
2. Prioridad absoluta, de modo que otras comunicaciones, como la transferencia de correo, acceso a ficheros remotos, web o vídeo no perturben la posibilidad de comunicarse por voz con ningún punto de la red.
3. Calidad de voz comparable o superior a la de la telefonía móvil.
4. Compatibilidad con aplicaciones de VoIP en computadoras de la red de salud y del exterior, incluidas aplicaciones libres, lo que implica sujección a normas (al menos de facto) y la posibilidad de hablar por medio de códecs y protocolos no patentados. No es necesario garantizar la calidad y prioridad cuando se sale al exterior.
5. Acceso desde la red de VoIP a telefonía convencional, con las llamadas salientes debidamente autorizadas, autenticadas y con la posibilidad de tarificación.
6. Es desable poder establecer multiconferencias, de modo que varios interlocutores puedan reunirse virtualmente.

### 2.1.2. Videoconferencia

Aunque es un servicio de menor importancia, a menudo están ligados a los mismos protocolos que la voz (RTP y H.323 o SIP, normalmente) y, si se usan, deben ser prioritarios sobre otros servicios, excepto el de voz. No se considera que la videoconferencia deba estar accesible permanentemente ni que se pueda propagar al exterior de la red.

### 2.1.3. Charla interactiva

Puede asociarse a sistemas de conferencia (audio y o vídeo) o ser independiente. No tiene requisitos especiales de prioridad y puede utilizarse cuando el servicio de voz no pueda prestarse con calidad suficiente por sobrecarga.

### 2.1.4. Correo electrónico y web

Aunque muy importantes, sobre todo el correo electrónico, son las comunicaciones de menor prioridad. El sistema debe garantizar que usan el ancho de banda marginal disponible y no retrasan los paquetes de voz. Como Web entendemos no sólo HTTP, sino los demás protocolos comunes (FTP, HTTPS, etc). Las mejoras de rendimiento en estos servicios pueden obtenerse con proxies apropiados.

### 2.1.5. Protocolos de terminal

En el caso de considerarse viable una plataforma única de bajo mantenimiento para todo, deberá considerarse la necesidad de utilizar protocolos de terminal (VNC, X, ICA, RDP), que sin tener las características de tiempo real, sí deberían priorizarse sobre el tráfico de correo y web.

### 2.1.6. Acceso a ficheros remotos

En el caso de considerarse viable una plataforma única de bajo mantenimiento para todo (voz y datos), deberá considerarse la necesidad de acceso a ficheros remotos, ya sea para transferir el software sólamente (típicamente TFTP), ya sea para acceder a ficheros remotos donde residan las aplicaciones que se ejecuten localmente y sus datos. En su caso deben determinarse los protocolos concretos a usar, pero se anticipa que no necesitarán calidad de servicio prioritaria.

## 2.2. Características geográficas

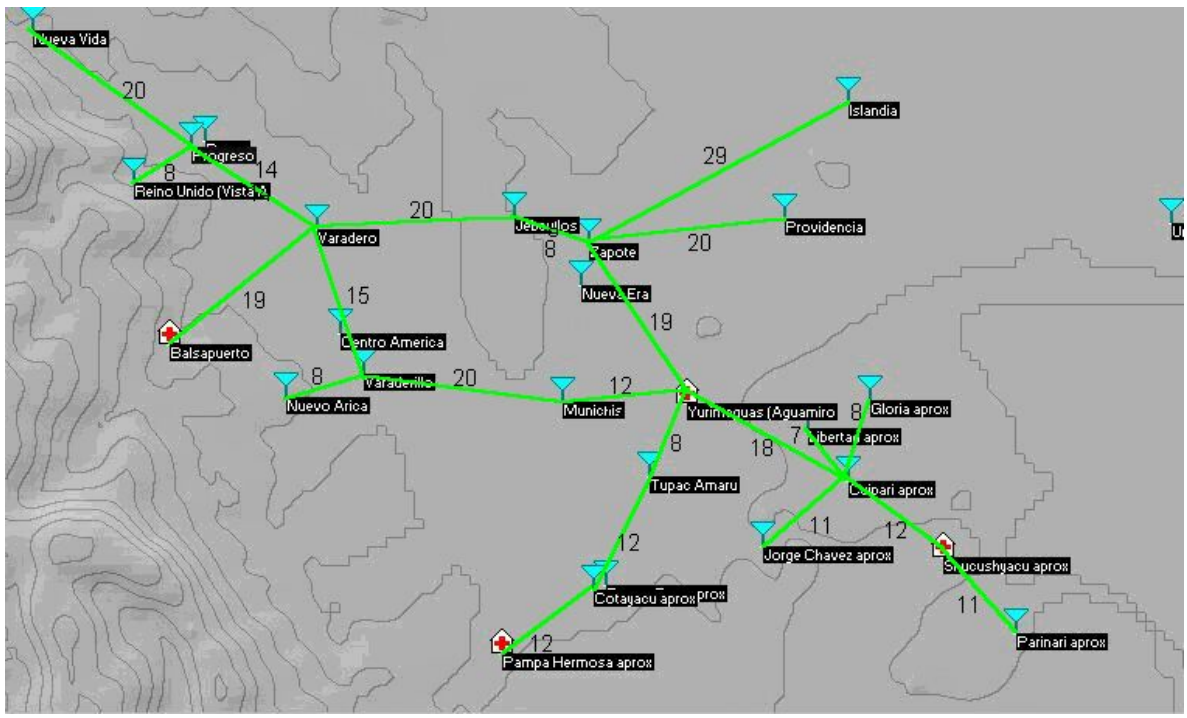
Aunque las características geográficas pueden ser muy variables, siempre nos moveremos en entornos rurales y urbanomarginales. De ellos los más exigentes son

los rurales, ya que cualquier entorno urbanomarginal tendrá posibilidad de acceso a telefonía e internet a menos de unas pocas decenas de kilómetros[1]. Nos centraremos pues en entornos rurales típicos, tomando como modelos aquellos en los que ya se han hecho instalaciones.

## 2.2.1. Selva

El primer proyecto piloto de EHAS, con tecnología VHF, se realizó en una zona de selva baja de la provincia de Alto Amazonas en Perú (departamento de Loreto). En ella los establecimientos de salud están estructurados en microrredes de un diámetro del orden de 40 Km, como puede verse en la Figura 2-1 . Esto no era obstáculo para los enlaces VHF, que podían estructurarse en redes punto multipunto, centradas en los centros de salud, con acceso VSAT o telefónico, marcados con una cruz en la figura.

**Figura 2-1. Centros y puestos de salud enlazados en la zona de Alto Amazonas**




A pesar de ser una zona llana, esta distancia es de por sí demasiado grande para enlazar con visión directa, sobre todo en un terreno con árboles mayores de 15 metros.

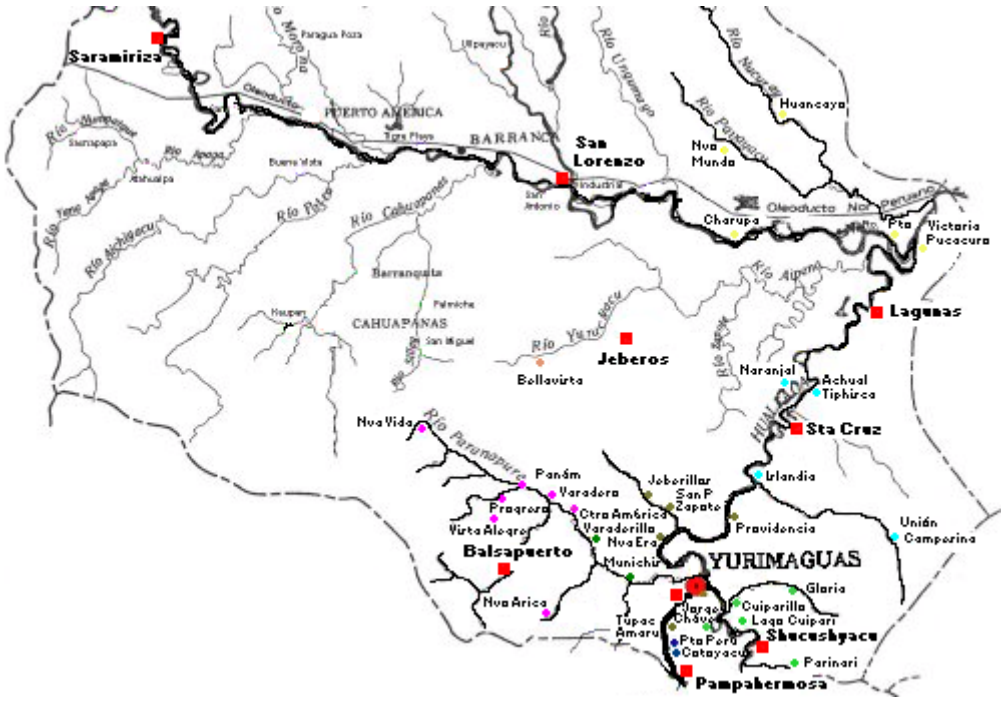
Los problemas de distancia y de visión pueden soslayarse si enlazamos los

establecimientos de forma irregular, aprovechando los tramos más cortos y repitiendo la señal para el siguiente salto. En la Figura 2-1, los enlaces son de entre 8 Km y 20 Km, salvo uno de 29 Km, necesario por la existencia de una colina entre establecimientos más cercanos.

Si estos enlaces pueden establecerse, tendríamos una red redundante, con varios puntos alternativos de acceso a internet (los centros de salud) y, en algunos casos, caminos redundantes, aunque la probabilidad de poderlos realizar es muy escasa, debido a las características geográficas.

En efecto, el patrón de población de estas zonas se circunscribe a las vías fluviales, que se introducen en el territorio de forma arborescente, como puede verse más claramente en la Figura 2-2. Esto sitúa las poblaciones en los puntos más bajos y más difíciles de enlazar, lo que exige torres muy altas.

**Figura 2-2. Cuenca fluvial de Alto Amazonas, con puestos y centros de salud marcados.**

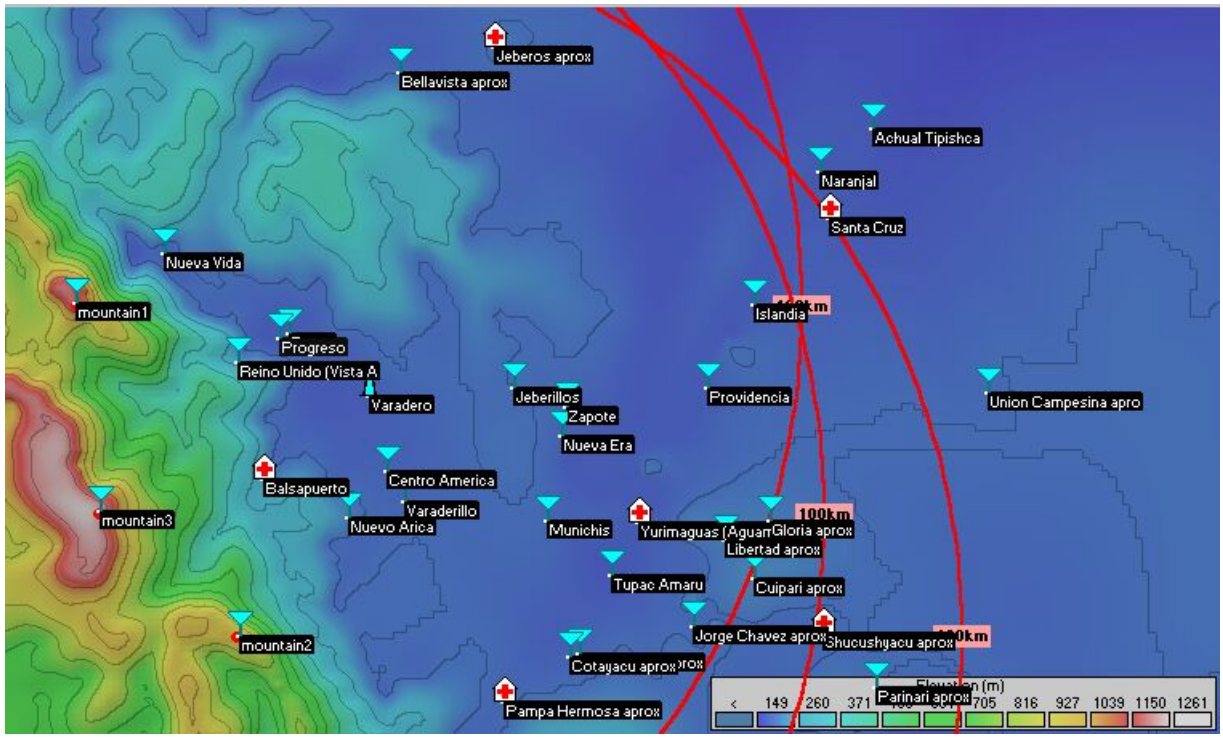



No es raro, sin embargo, encontrar pequeñas colinas situadas a decenas o centenares de metros, donde podrían situarse elementos repetidores relativamente vigilados y mantenibles. Incluso dentro de la misma población, el lugar más alto puede ser un edificio distinto (quizá la iglesia o la municipalidad), donde se puede emplazar un repetidor. Por otro lado, en las orillas de los ríos, siempre hay poblados que podrían

albergar y cuidar repetidores cercanos, pudiéndose así constituir a lo largo de los corredores fluviales verdaderos *corredores inalámbricos*.

Aunque es desable que todo elemento repetidor esté cerca de una población que pueda encargarse de su vigilancia y cuidado, habrá ocasiones en que no sea posible y sea necesario ubicarlos en lugares altos desatendidos. Por ejemplo, el anterior enlace de 29 Km (Zapote-Islandia), podría romperse a través de Providencia si se pone un repetidor en la colina que la separa de Islandia. En otras ocasiones podría recurrirse a alturas más remotas, como se ve en la Figura 2-3, aunque esta es una situación más parecida a las de sierra.

**Figura 2-3. Visibilidad desde las montañas de la zona de Alto Amazonas**

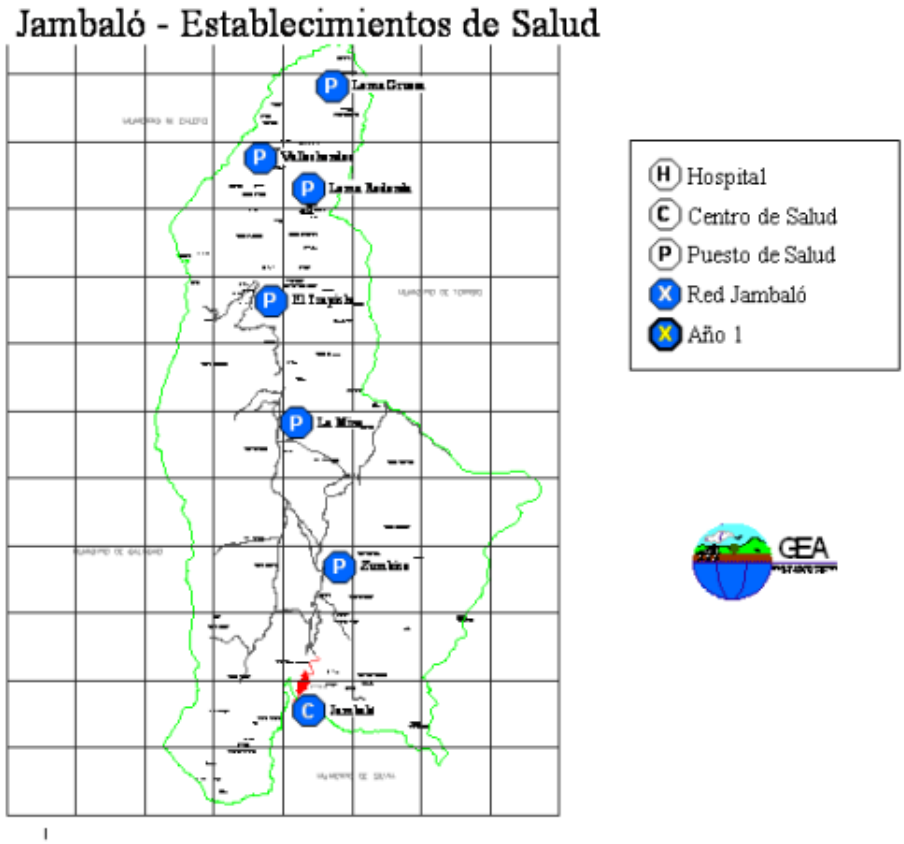



## 2.2.2. Sierra

En sierra la situación es distinta, aunque parecida al último escenario de selva, sólo que sin el problema de los árboles. Normalmente las poblaciones estarán en valles y a veces será posible establecer corredores inalámbricos en los mismos. Esto sucede, por ejemplo, en el proyecto EHAS-Jambaló, en Colombia, como muestra la Figura 2-4. Pero en la mayoría de los casos habrá que recurrir a cumbres elevadas donde se

puedan poner los elementos repetidores que completen los enlaces, con el problema añadido de la inaccesibilidad y la imposibilidad de vigilancia, a veces mitigado por la existencia de torres de radioenlaces, televisión, etc, que pueden utilizarse de mutuo acuerdo.

**Figura 2-4. Valle de Jambaló**




## 2.3. Características de hardware

El despliegue de esta red requiere de dos tipos de elementos:

- Los que pertenecen al sistema de *encaminamiento exterior*, que deben tener unas características ambientales muy duras, generalmente con temperaturas y

humedades muy altas, lo que exije un hardware *tropicalizado*, muy robusto.

Asímismo están sometidos a frecuentes tormentas eléctricas, lo que requiere una protección adecuada. La necesidad de que en muchos casos deba emplazarse un encaminador inalámbrico solar relativamente cerca de un establecimiento favorece el aislamiento de los equipos de los establecimientos, ya que no hay ninguna conexión eléctrica entre los mismos y la antena elevada. Esto podría sugerir salir de todo establecimiento por medio de unos de estos encaminadores aislados eléctricamente, aunque por cuestiones de alcance no fuera necesario.

Estos elementos siempre tendrán alimentación solar.

- Los que perteneces a los *establecimientos*, que aunque no están sometidos a condiciones ambientales tan duras, no dejan de ser duras (la misma humedad, pero sin lluvia, y temperaturas ambientales, pero a la sombra). La experiencia previa con ordenadores portátiles ha dado bastantes malos resultados en este ambiente.

  Si estuvieran conectados a un elemento radiante externo, deberían tener las protecciones apropiadas (pararrayos, protectores contra sobretensiones, etc), elementos caros y que podrían reemplazarse por enlaces inalámbricos a elementos flotantes activos y baratos.

  Estos elementos tendrán siempre alimentación solar, a menos que se disponga de sistemas alternativos suficientemente fiables. La experiencia de la utilización de los grupos electrógenos comunitarios, muchas veces disponibles sólo algunas horas al día, ha dado muy malos resultados, tanto por la calidad de la energía (estabilidad, sobretensiones), como de disponibilidad (a veces pasan semanas o meses sin combustible para el generador).

En todo caso el hardware a desplegar, además de cumplir las exigencias técnicas y ambientales, debe ser fácilmente adquirible en los países destinatarios, con segundas fuentes, sujeto a normas relevantes, con documentación suficiente, y utilizable por el software necesario.

## 2.4. Características del software y los protocolos

Aunque lo que se plantea es una red privada, se desea que puedan situarse en ella

elementos relativamente heterogéneos. En particular, el programa EHAS considera que a medio plazo pueda utilizarse *software libre* o de *fuente abierta* (FLOSS) en todos los puntos de la red, incluidos los terminales de usuario, ya que no sólo facilita el desarrollo y reduce costes, sino que también representa una oportunidad para los países en desarrollo, reduciendo su dependencia tecnológica y mejorando las posibilidades de desarrollar negocios locales. En particular, en Perú se propuso un proyecto de Ley de uso prioritario de software libre en las administraciones públicas, que, aunque no salió adelante, un próximo cambio de gobierno puede volver a reactivar. Otros gobiernos cercanos, como el del Brasil, ya tienen políticas activas de promoción del software libre.

En todo caso es necesario establecer una infraestructura que permita la transición y la coexistencia con software propietario, siempre que se someta a ciertas normas mínimas.

En el caso del sistema de encaminamiento, al no existir soluciones satisfactorias en el mercado, se va a optar por sistemas basados en Linux empotrado, dados los servicios en él ya implementados y la facilidad de integración de los necesarios. Es por ello que en lo que sigue se hará gran énfasis en cómo se soportan las cosas en Linux.

En el caso de los terminales y las pasarelas con la red telefónica del sistema de voz sobre IP, se estudiarán las posibles ventajas e inconvenientes de usar soluciones propietarias cerradas frente soluciones abiertas, en el caso de que estén maduras y sean rentables. En el caso de los terminales de voz sobre IP, sean hardware o software, se buscará la máxima interoperabilidad, teniendo en cuenta que algunos códecs, por ejemplo, están patentados y no pueden implementarse en softphones libres.

# Notas

1. Hay casos extremos, como la ciudad de Lima, de un diámetro del orden del centenar de kilómetros, pero siempre tienen acesibilidad a distancias mucho menores.

# Capítulo 3. Voz sobre IP

Vamos a analizar cómo se encuentra actualmente el mundo de la VoIP, a todos los niveles, centrándonos en aquellas posibilidades que tengan una solución no propietaria, es decir, de software libre. Actualmente existen dos estándares prominentes en el mundo de la telefonía IP: H.323 y SIP. Pasamos a continuación a verlos en detalle, analizando sus ventajas e inconvenientes, sobre todo en su relación con la red de VoIP que se pretende desplegar.

Nuestro objetivo fundamental es diseñar una red IP con la capacidad y funcionalidad suficiente como para proporcionar un servicio de VoIP al usuario garantizando la calidad de servicio. Para ello se pretende analizar el funcionamiento de las soluciones de VoIP más comunes y extrapolar las medidas a tomar para diseñar una red que sea capaz de dar la prioridad adecuada a los paquetes de voz y de vídeo pertenecientes a una videoconferencia.

Hemos de recordar que no sólo existen esta dos opciones en el mercado, hay otras (MGCP/Megaco, etc.) que por su menor difusión y/o efectividad, tras un estudio preliminar, no hemos considerado como candidatos finales para el despliegue de esta red. Por otra parte mencionar que existen más soluciones de las que vamos a ver aquí, que de nuevo por su perfil, tanto como por simplicidad (por ejemplo, SpeakFreely, aplicación sencilla que permite hablar remotamente con otro usuario) como por dependencia de servidores de terceras partes (por ejemplo, los servicios de mensajería instantánea, que también tienen soporte para videoconferencia, como MSN o Yahoo! Messenger, y el ahora famoso servicio de voz sobre IP Skype), no vamos a analizar tampoco dado que tras este estudio preliminar resulta evidente que no se ajustan a nuestras necesidades.

## 3.1. Arquitectura de redes VoIP

En general un sistema de voz sobre IP (Voice over IP, VoIP) consiste en establecer una conferencia de audio entre dos terminales conectados a una red. En su forma más básica, todo lo que requiere es que sea bidireccional, que el retardo sea prácticamente constante y muy bajo, y que la calidad subjetiva de audio sea suficientemente buena para entender al interlocutor. El problema principal de encargar este cometido a una red IP consiste en que estas redes son de tipo *best-effort*, y no proporcionan por tanto calidad de servicio, necesaria para garantizar que los paquetes de voz llegan en el

orden adecuado y con retardo mínimo y común a todos ellos.

Vamos a analizar esta arquitectura desde una aproximación de arriba hacia abajo, viendo sus distintos elementos de forma general y resaltando aquellos que sean más importantes:

En el nivel de aplicación será donde nos encontremos el grueso de elementos necesarios para constituir la arquitectura, necesitaremos una aplicación que tenga los siguientes elementos:

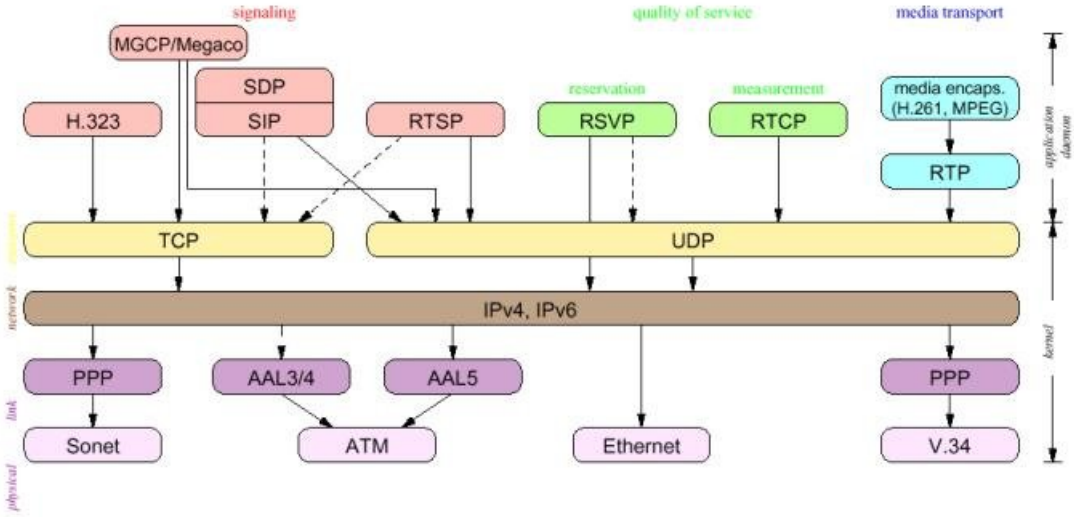
- Interfaz con el usuario: En principio puede tratarse de un dispositivo físico, como un teléfono o videoteléfono, o de un dispositivo lógico, es decir, una aplicación (llamadas *softphones*), ejecutándose en un PC o máquina similar, con la ayuda de micrófono y altavoces, y opcionalmente una cámara para vídeo.
- Opcionalmente, la aplicación puede incluir un protocolo de control/señalización, como se muestra en la Figura 3-1 (con los ejemplos típicos: H.323, SIP, MGCP/Megaco), para la gestión de las conferencias, pero en general esto no es necesario para establecer una comunicación de VoIP, prueba de ello son todas las aplicaciones que no utilizan estos sistemas, y que mencionábamos más arriba: Yahoo! Messenger, SpeakFreely, etc. que o bien desarrollan sus propios protocolos de gestión o incorporan funcionalidad muy reducida, la mínima imprescindible para establecer sesiones de voz punto a punto. Hay que hacer notar que esto no entra dentro de la definición estricta de una red de VoIP, donde entran en juego consideraciones adicionales, desde tarificación hasta gestión de ancho de banda.
- Codificador/Decodificador de señal: Conocidos normalmente como códecs, son uno de los elementos críticos de una red de VoIP, ya que se encargan de codificar la señal de audio o vídeo con dos objetivos principales en mente, mantener un nivel alto de calidad y reducir el ancho de banda utilizado por el flujo. Estos dos conceptos suelen ser excluyentes, el resultado dependerá del algoritmo escogido para implementar el códec. Es tan grande su importancia que les dedicaremos su propio apartado.
- Por último, en el nivel más bajo de aplicación, necesitaremos un mecanismo que permita enviar los datos en tiempo real, para ello recurriremos a la solución propuesta en [SCF2003], el protocolo RTP (*Real Time Protocol*), utilizado en todas las aplicaciones con requerimientos de tiempo real. También veremos sus características más relevantes en detalle más adelante.

Mención aparte merece el tema de la identificación de usuarios: ¿cómo puede saber la aplicación dónde se encuentra el interlocutor? Para ello se habrá de definir primero un sistema de nombramiento, que permita identificar unívocamente a los distintos usuarios de una red VoIP, y además un sistema de resolución de nombres, siendo necesario tener un direcotorio donde se hallen almacenadas estas correspondencias, para poder localizar a los usuarios. Veremos también en otro apartado diversos mecanismos para implementar esta función.

El siguiente nivel será el de transporte, necesitamos un protocolo de transporte que respete los requerimientos de tiempo real, esto es, que no añada excesiva sobrecarga a nuestros paquetes, y que los vaya entregando al nivel superior según llegan lo más rápidamente posible, en este sentido habremos de escoger entre TCP y UDP, los dos protocolos de transporte típicos, TCP es fiable, y orientado a conexión, pero no cumple la premisa de la entrega de paquetes a medida que llegan, se encarga de ordenarlos y entregarlos al nivel superior sin los límites temporales que precisamos aquí, además a tal efecto pide incluso la retransmisión de los paquetes que no hayan llegado, por motivos obvios, esto es absurdo en una comunicación en tiempo real. UDP es no fiable y no orientado a conexión, pero añade menos sobrecarga a los paquetes y los entrega según los recibe, lo cual si no ayuda al flujo de tiempo real, al menos no lo entorpece.

Por último tenemos el nivel de red, evidentemente al tratarse de una red VoIP el protocolo elegido es IP, en principio IPv4, aunque cabe la posibilidad de ampliar el estudio, o al menos reseñar las principales diferencias, con el emergente IPv6.

**Figura 3-1. Arquitectura de redes VoIP**




# 3.2. Códecs

Vamos a analizar sucintamente el funcionamiento de un códec. Los paquetes de voz (y en su caso de vídeo) son transportados por RTP codificados de alguna manera. Esta codificación determina el ancho de banda utilizado y la calidad de la comunicación. Generalmente se requiere que los interlocutores negocien el mismo códec para poder comunicarse, lo que tiene consecuencias en la interoperabilidad de los terminales. Así mismo algunos códecs de calidad están sujetos a patentes que imposibilitan su uso en software libre. Veamos a continuación los códecs más conocidos y utilizados, con sus características más relevantes.

## 3.2.1. Audio

La gran mayoría de las aplicaciones de telefonía de VoIP, implementan la recomendación G.711 del CCITT de 1984, diseñada para transportar telefonía digital en canales de 64 Kbps, codificando muestras independientemente (Pulse-Coded Modulation o PCM) de forma no lineal (leyes A y mu). Recomendada para redes con ancho de banda suficiente, suele usarse como referencia de calidad

para otros métodos que comprimen las muestras basándose en su historia.

G.726 es una codificación adaptativa diferencial (ADPCM) a diferentes tasas de bit. Su rendimiento y calidad a 40 Kbps son comparables a G.711. A tasas de error altas, puede tener una calidad subjetiva mayor, tanto a 40 Kbps como a 32 Kbps. Antecesor suyo es G.721, que sólo funciona a 32 kbps. Este códec lo implementa Gnomemeeting (http://www.gnomemeeting.org), y algunas aplicaciones más, sin embargo, su difusión es más limitada que la del G.711.

G.728 comprime según una Low-Delay Code Excited Linear Prediction (LD-CELP), usándose a 16 Kbps. Comparado con G.721, G.728 tiende a ser objetivamente peor, pero es mejor en pruebas subjetivas. Tiende a funcionar peor en presencia de ruido.

G.723.1 provee de dos tasas distintas (5.3 y 6.4 Kbps). Además la velocidad en cada sentido no tiene porqué ser la misma y puede variar entre tramas. Con respecto a los códecs G.729/G.729A, ofrece tonos DTMF (los tonos utilizados por la marcación telefónica digital) con menor distorsión, lo cual permite mayor fiabilidad al acceder a servicios de red inteligente en telefonía, por ejemplo.

G.729 es un códec compresor de tipo Conjugate-Structure Algebraic-Code-Excited Linear Prediction (CS-ACELP). El G.729 Anexo A (G.729A normalmente) es una versión de complejidad reducida. Se desarrolló para usarse en situaciones de multimedia con voz y datos.

Ambos tienen buena relación ancho de banda calidad, y están implementados normalmente tanto en hardware como en software propietario. En software libre so se pueden utilizar con tarjetas de telefonía que los implementen, como las tarjetas QuickNet (http://www.linuxjack.com/), que funcionan en Linux.

El códec *Global System for Mobile communications* (GSM) se usa en telefonía móvil digital en Europa extensivamente, y en otras partes del mundo. Es muy utilizado también en el mundo de VoIP, por ser libre y tener una buena relación calidad/ancho de banda.

Speex es un códec de software libre diseñado para voz, con la idea de permitir la entrada al mercado de aplicaciones de voz a más gente, al proveer de una alternativa gratuita a los códecs patentados. Entre sus puntos fuertes se encuentra la capacidad de ofrecer una tasa de bits variable en la misma comunicación, por ejemplo. Existen dos modos principales de funcionamiento para redes de banda ancha (Speex Wide, a 16 KHz) y de banda estrecha (Speex Narrow, a 8 KHz), proporcionando diferentes tasas de bit en cada caso. Los decodificadores Speex habrán de proporcionar,

como mínimo, decodificación para todos los modos de banda estrecha, y si el decodificador es de banda ancha, ha de decodificar banda estrecha también. Es un códec emergente que se está incluyendo en la mayor parte de las aplicaciones libres de VoIP, facilitándose también módulos para que las aplicaciones de Windows, lo puedan utilizar.

iLBC (Internet Low Bitrate Codec) es un reciente códec diseñado por GIPS (Global IP Sound) y sometido al IETF para su consideración como norma para bajo ancho de banda. En la actualidad existe una realización gratuita y abierta de GIPS para uso personal. GIPS tiene también otros códecs propietarios.

LPC-10 es un códec diseñado por el DoD americano para una banda muy estrecha (2.4 kbps) y bajo gasto de procesador.

La Tabla 3-1 muestra, de forma resumida, las características más sobresalientes de estos códecs.

**Tabla 3-1. Tabla de códecs de audio**

| **** Nombre*** | **** Tasa binaria (kbps)*** | **** Tamaño de paquete (octetos sin cabeceras)*** | **** Duración de un paquete (ms)*** | **** Puntuación MOS*** | **** Patentado*** |
|---|---|---|---|---|---|
| G.711 | 64 | 160 | 20 | 4.1 | No |
| G.723.1 | 5.3/6.4 | 24 | 30 | 3.65/3.9 | Sí |
| G.726 | 16/24/32/40 | 80 | 20 | 3.85 para 32 kbps | Sí |
| G.728 | 16 | 60 | 30 | 3.61 | Sí |
| G.729A | 8 | 20 | 20 | 3.7 | Sí |
| GSM | 13 | 33 | 20 | 3.5-3.7 | No |
| iLBC | 13.3/15 | 50 | 30/20 | 4.1/4.2 | Dudoso |
| LPC-10 | 2.4 | 30 | 30 | 2.5 | No |
| Speex-narrow | 2.15-24.6 | 30-100 | 20 | 3.5-4 (estimado) | No |

* *Duración de un paquete:*

Se han mostrado valores típicos utilizados normalmente. Se puede aumentar o disminuir el intervalo muestreado, impactando esto sobre el retardo del paquete, la eficiencia del códec, etc. Evidentemente, el tamaño del paquete variará proporcionalmente.

* *Puntuación MOS:*

*Mean Opinion Score*, Puntuación de Opinión Media, escala para medir la percepción subjetiva de la calidad de voz tras pasar por éstos códecs, va desde 1 (mala) hasta 5 (excelente). En este caso es la MOS medida en condiciones ideales, sin pérdida de paquetes ni distorsión adicional.

### 3.2.2. Vídeo

H.261 es un estándar de vídeo publicado por la ITU (International Telecom Union) en 1990. Fue diseñado para tasas de bit múltiplos de 64. Se diseñó con líneas ISDN en mente, de ahí este valor. Es un híbrido de predicción entre tramas, codificación transformada y compensación de movimiento. Funciona entre 40 kbps y 2 Mbps. Soporta dos resoluciones distintas, QCIF (Quarter Common Interchange format, 176x144) y CIF (Common Interchange format, 352x288).

H.263 es una versión mejorada de H.261. Fue diseñado para bajas velocidades, pero se amplió a grandes rangos de tasas de bit. Se supone que reemplazará a H.261 en muchas aplicaciones. Soporta cinco resoluciones distintas, QCIF, CIF, SQCIF (mitad de resolución que QCIF), 4CIF y 16CIF (cuatro y dieciséis veces la resolución de CIF, respectivamente). Estas altas resoluciones implican que puede incluso competir con los estándares MPEG.

Ambos algoritmos son los más utilizados (de hecho, son casi los únicos que se utilizan) en las aplicaciones que requieren vídeo, hay que hacer notar que el uso de H.263 a resolución CIF o superior implica que un terminal H.323 ha de ser capaz de proporcionar H.261 CIF. En otras palabras, si un terminal H.323 implementa el códec H.263 ha de implementar al menos el códec H.261 a la misma resolución, por razones de compatibilidad.

CU30 es un algoritmo patentado, desarrollado por el DISCOVER Lab de la Universidad de Cornell, que soporta 30 tramas por segundo en canales de baja velocidad. No está soportado por Netmeeting, aunque sí por algunas aplicaciones libres.

La Tabla 3-2 muestra, de forma resumida, las características más sobresalientes de estos códecs de vídeo.

**Tabla 3-2. Tabla de códecs de vídeo**

| **** Nombre*** | **** Tasa binaria (kbps)*** | **** Imágenes por segundo*** | **** Resoluciones soportadas*** | **** Patentado*** |
|---|---|---|---|---|
| H.261 | P * 64 (P=1..30) (Valor típico 384) | 29.97/mpi, (mpi=1..4) | CIF (352x288), QCIF (176x144) | No |
| H.263 | P * 0.1 (P=1..192) | 29.97/mpi, (mpi=1..32) | CIF (352x288), QCIF (176x144), SQCIF (128x96), 4CIF (704x576), 16CIF (1408x1152) | Dudoso |
| CU30 | Variable | 30 | Variable | Sí |

* *Tasa binaria:*

P es sencillamente un factor de escala, la tasa de estos códecs es múltiplo de 64 kbps.

* *Imágenes por segundo:*

mpi (minimum picture interval) es el mínimo intervalo entre fotogramas, evidentemente, cuanto menor sea mayor será la tas de imágenes por segundo.

## 3.3. RTP/RTCP

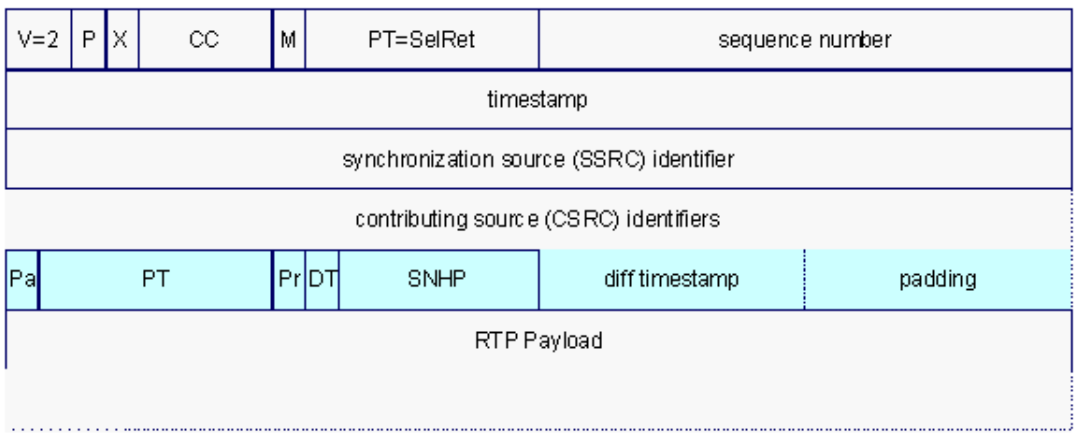
Para aplicaciones como la telefonía IP, con flujos de datos en tiempo real, se desarrollaron los protocolos RTP/RTCP (Real Time Protocol/Real Time Control Protocol) , definidos en la [SCF2003]. Son capaces de proporcionar calidad relativamente aceptable a los flujos de datos de tiempo real mediante mecanismos como control de datagramas descartados en la red, o control del correcto orden de reensamblaje de los mensajes, por citar dos de los más comunes. Son protocolos de nivel de aplicación y, evidentemente, por sus necesidades de tiempo real, los mensajes RTP/RTCP van encapsulados sobre datagramas UDP, lo que garantiza que los procesados de protocolos de nivel inferior añaden el mínimo retardo, amén de no necesitar establecer una conexión, como ocurriría si se utilizara TCP, por ejemplo.

A nivel general podemos hablar de una sesión RTP, en la cual dos o más participantes

intercambian flujos de datos entre sí, utilizando un par de puertos para cada tipo de media (el par se corresponde con el canal RTP, que utiliza un número de puerto par, y su correspondiente RTCP, que utiliza el puerto consecutivo, impar). A cada uno de los participantes que envía datos se le llama fuente (*synchronization source*), y se le identifica unívocamente dentro de la sesión mediante un número de 32 bits escogido aleatoriamente; aunque la probabilidad de que dos fuentes escojan el mismo número es muy baja, en la propia RFC se contempla esta posibilidad y se proporcionan los mecanismos para evitarlo. El puerto RTP se utilizará entonces para el envío del flujo, y el RTCP en general se usará para proporcionar un mecanismo de control dando principalmente realimentación a la fuente.

A continuación vamos a ver las capacidades de este protocolo. En este sentido, el protocolo RTP proporciona una serie de mecanismos para cumplir su función, incluidos en su cabecera:

**Figura 3-2. Cabecera de un paquete RTP**




- Proporciona un campo *Payload Type* (PT) que permite identificar el formato del

tipo de datos incluido en el paquete de datos de RTP, bien asociando nosotros mismos los valores del PT a ciertos formatos, o bien siguiendo la [CAS2003], donde se proporciona una lista de formatos y sus valores asociados, de forma análoga a como ocurre con los *well-known ports* de TCP.

En RTP queda a discreción del implementador tanto dar el significado y los valores del PT, como decidir sobre el formato de los datos del cuerpo (*payload*) del paquete. Esto nos lleva a preguntarnos qué ocurre con los flujos multiplexados en RTP. En principio el protocolo define un elemento, llamado mezclador (*mixer*), cuya función es, al recibir datos de distintas fuentes, combinarlos para formar un único flujo (por ejemplo juntando todas las voces de distintos participantes o componiendo un vídeo con la imagen que proporciona cada uno), pero esto no es realmente una multiplexación, es más bien una combinación de flujos que puede tener su utilidad en ciertos escenarios. En este sentido, y concretando un poco más, el mezclador se encargaría de decodificar las señales que le llegan, combinarlas y recodificarlas en una sola, lo cual implica que aunque las fuentes podrían estar utilizando distintos códecs, la señal combinada que sale del mezclador sería una y estaría codificada con otro códec (no necesariamente distinto de todos los anteriores). La fuente del flujo sería ahora el mezclador, y las fuentes contribuyentes (*contributing sources*, son los identificadores de fuente de todos los que contribuyen con sus datos) serían todas aquellas que han proporcionado datos al mezclador. Esto nos lleva a ver que, en efecto, el campo PT de cada paquete hace referencia tal y como se pretendía a la codificación usada en los datos del mismo, y que aunque es posible teóricamente introducir diferentes codificaciones en un mismo paquete RTP (por ejemplo, audio y vídeo a la vez), dado que nosotros tenemos el control del formato del paquete, en la práctica esto iría contra el propósito del campo PT, ya que no podríamos identificar todos los elementos del paquete (podríamos si redefiniéramos totalmente el campo PT, claro), y además introduciríamos más problemas en los cuales no vamos a entrar, que están descritos en la RFC. En resumen, una implementación de RTP bien hecha no debería combinar en un paquete flujos codificados de distinta forma, la única multiplexación admitida en este caso sería la ya vista al hablar del mezclador, y por tanto el campo PT define totalmente el contenido, tanto en tipo como en formato, del paquete RTP.

- Marca los paquetes de datos con un número de secuencia (*sequence number*), teniendo en cuenta que éstos viajan sobre UDP y que pueden llegar desordenados o incluso no llegar, esto proporciona un mecanismo que permite ordenar los

paquetes en el destino y saber cuáles y cuántos faltan.

- Implementa una marca de tiempo (*timestamp*) a partir del reloj de muestreo del emisor, que permite al receptor detectar error de *jitter* (variación de la latencia) y tomar las medidas adecuadas.

Por su parte, el protocolo RTCP se encarga del control y monitorización del flujo de datos RTP, mediante la transmisión de paquetes de control periódicamente a todos los participantes. Éstos tienen las siguientes funciones:

- La función principal es de realimentación, de esta forma es posible conocer dónde se están dando fallos, si éstos son globales o locales, etc. Además estos datos pueden ser utlizados como base para realizar codificaciones adaptativas, mejorando la calidad de servicio ofrecida (aunque recordemos que RTP/RTCP no garantiza ninguna calidad de servicio determinada).

  Para poder ejercer la función de realimentación correctamente, los paquetes RTCP se generan a partir de datos estadísticos obtenidos mediante la observación de los paquetes RTP, como por ejemplo, el número de paquetes perdidos, el porcentaje de paquetes desordenados, o el *jitter* medio entre llegadas.

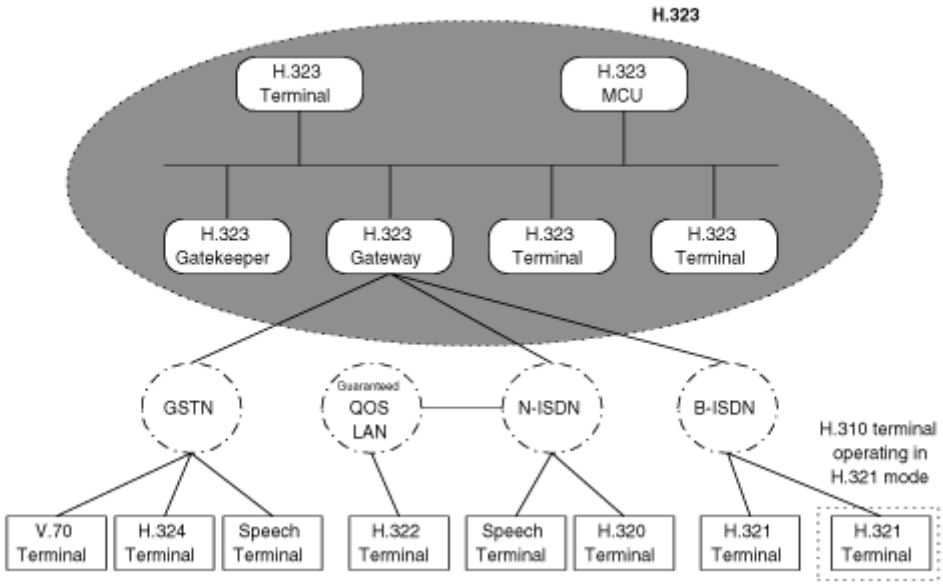
  Conviene resaltar que queda a discreción de la aplicación la interpretación y las medidas a tomar a partir de los datos recolectados mediante RTCP.

- Más funciones de RTCP incluyen dar nombres canónicos a los usuarios por si cambian su identificador de fuente, controlar el flujo de paquetes RTCP en función de el número de usuarios de la sesión (por motivos de escalabilidad), y por último, proporcionar información de control sobre la sesión, por ejemplo sobre qué usuarios hay presentes, para uso de las aplicaciones involucradas.

## 3.4. Norma H.323

La norma H.323 fue creada por el ITU-T, y en realidad es una *norma paraguas*, es decir, no es un protocolo en sí mismo, sino una familia de protocolos que interactúan entre sí para proveer un servicio, en este caso telefonía o servicios de videoconferencia IP. Como tal no nos vamos a centrar en desglosar los distintos protocolos que la componen, sino que a medida que éstos nos resulten relevantes en nuestro estudio, los iremos incluyendo proporcionando los detalles necesarios para entenderlos.

**Figura 3-3. Visión general de una red H.323**




## 3.4.1. Red básica H.323

En su forma más sencilla, una red H.323 consta únicamente de terminales (al menos dos), entendiéndose por terminal un cliente que implementa el protocolo H.323, bien de forma hardware (el caso de un teléfono IP, por ejemplo), bien como aplicación de software (también llamados *softphones*).

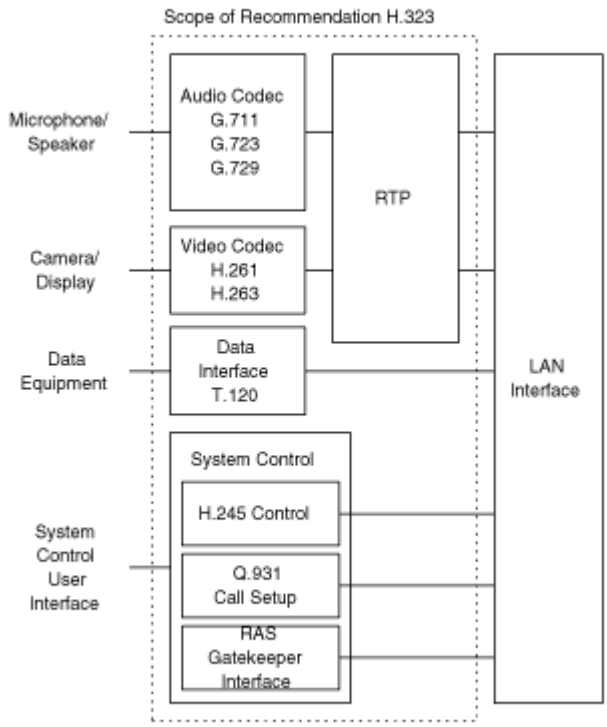
Cada cliente ha de implementar parte de la familia de protocolos H.323, en concreto los siguientes protocolos son necesarios:

- El protocolo H.225.0, variante evolucionada del Q.931, que se encarga del establecimiento y señalización de la llamada. Es de notar que, inicialmente, las implementaciones H.323 utilizaban el protocolo Q.931 directamente para esta tarea; con el tiempo y atendiendo a las necesidades específicas de VoIP, se modificó para ajustarse a las nuevas necesidades, dando como resultado el H.225.0, que es compatible en sus funciones más básicas con el anterior, aunque por supuesto se recomienda que las nuevas implementaciones utilicen el más

moderno de los dos.

- Mención especial merece el protocolo RAS (Registration/Admission/Status), que se utiliza para la comunicación de un terminal H.323 con otro elemento, llamado *GateKeeper*, que será introducido en la siguiente sección. Baste por ahora con saber que es una entidad encargada entre otras cosas de gestionar a un cierto número de terminales. Dicho protocolo es parte de H.225.0, y ha de implementarse obligatoriamente en un cliente, dado que si existe un GateKeeper en la red, hay obligación de usarlo.
- El protocolo H.245, que se encarga del control de uso del canal y de la negociación de capacidades entre los terminales.
- Por supuesto, también ha de implementar la pareja de protocolos RTP/RTCP, para el envío de información (vídeo y audio) en tiempo real.

**Figura 3-4. Terminal H.323**



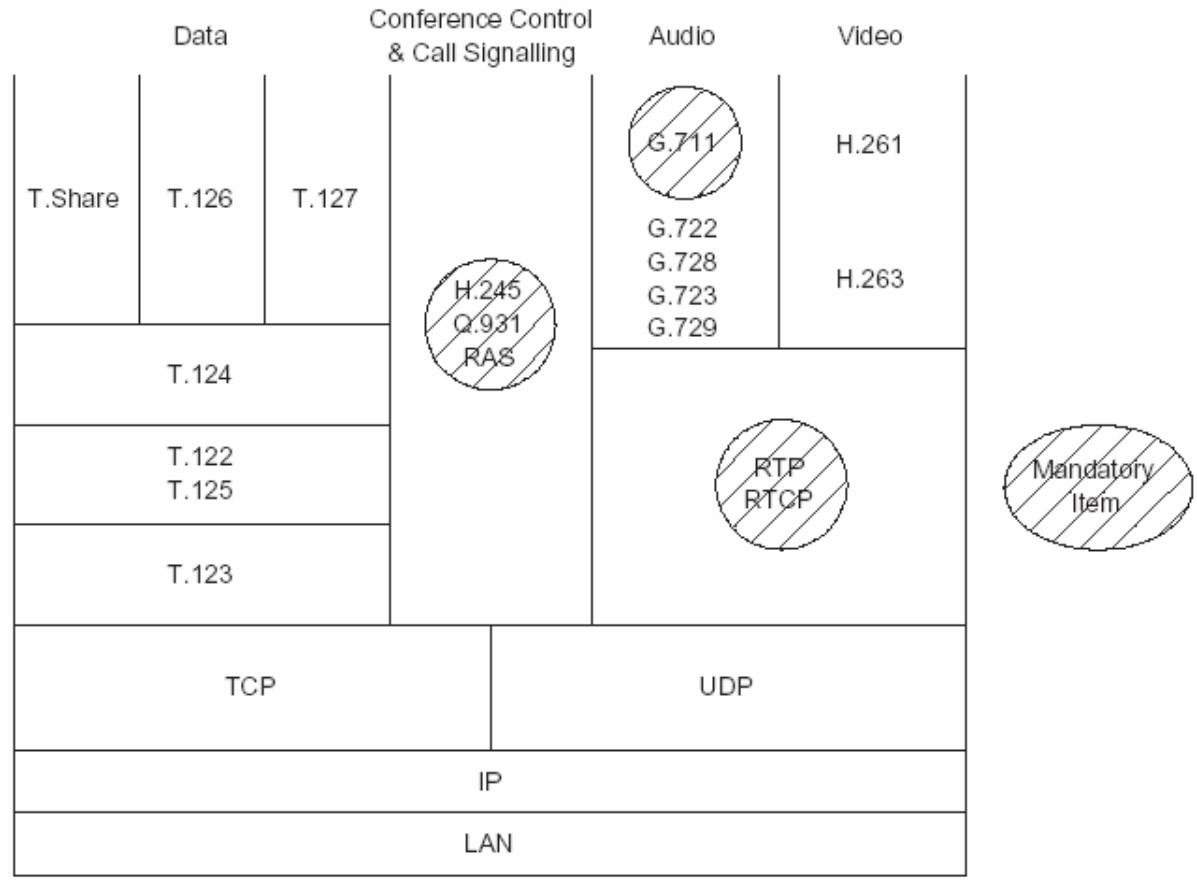
Además de todas estas condiciones, un cliente puede también implementar otras capacidades, como el envío de datos mediante el protocolo T.120 u otras, como soporte para multiconferencia, por ejemplo.

Centrémonos, pues, en las capacidades que ha de tener la red subyacente para proveer de un servicio H.323. En este sencillo escenario, en el que sólo intervienen los terminales, el único requisito que se pide al nivel de transporte es poder enviar y recibir datagramas UDP, no habiendo ningún requisito especial en el nivel de red. No obstante, más adelante tendremos en cuenta el uso de *proxies*, *firewalls*, y/o sistemas de NAT/PAT (Network y Port Address Translation), elementos no transparentes a la red, que pueden impedir la comunicación H.323 entre extremos, por los motivos que en su momento veremos.

Cabe destacar a este respecto lo ya mencionado: se trata de una red *best-effort*, por tanto no se garantiza la calidad de servicio, aunque la familia de protocolos H.323 y los protocolos RTP/RTCP intenten maximizar los recursos de la misma y dar el mejor servicio posible.

Pasemos entonces a analizar una comunicación H.323. Esto nos va a servir entre otras cosas, además de profundizar en el funcionamiento del protocolo, para detectar con qué elementos podemos jugar a la hora de intentar garantizar una cierta calidad de servicio, objetivo último de la red que se pretende desplegar.

**Figura 3-5. Torre de protocolos H.323**




Para iniciar una comunicación H.323, se establece una conexión de control, la que lleva los mensajes de establecimiento y señalización H.225.0. Lo habitual para esto es utilizar el puerto TCP 1720, aunque a continuación veremos algunas variantes. Se seleccionan entonces dinámicamente los puertos a usar, escogidos de entre el rango de 1024-65535, por un lado el canal de control, con el protocolo H.245, en un puerto TCP, y por otro los canales de audio y/o vídeo, asignando los puertos necesarios para RTP y RTCP, pero éstos son UDP (se necesitan una pareja de puertos en cada sentido, para cada flujo de datos, siendo x el puerto RTP, par, y x+1 su correspondiente RTCP, impar).

Alternativamente, existe la opción de encapsular los mensajes del protocolo H.245 sobre el canal H.225.0, ahorrando de esta forma el establecimiento de una conexión más. Y si además se utiliza un puerto UDP en vez de establecer una conexión TCP para el canal H.225.0, tenemos lo que se conoce como procedimiento *Fast-Connect*, que es el método más rápido que existe para establecer una conexión H.323, igualando en velocidad de establecimiento al protocolo SIP.

Resumiendo, no es necesario hacer nada en especial para habilitar las conferencias H.323 sobre una red TCP/IP, salvo disponer de los terminales preparados para efectuar y recibir llamadas. Sólo en el caso de tener elementos no transparentes habrá que tomar medidas para que éstos no sean un obstáculo. Eso sí, la calidad de servicio no está ni mucho menos garantizada, aunque H.323 apoyándose en RTP/RTCP permite adaptar el uso del canal al ancho de banda disponible, mediante la renegociación de *códecs*, por ejemplo.

### 3.4.2. Red H.323 con GateKeeper

El siguiente paso consiste en tener una red H.323 con GateKeeper.

Un GateKeeper es una entidad H.323 que se encarga, necesariamente, de las siguientes funciones:

- Traducción de direcciones: de aliases H.323 a sus correspondientes direcciones reales.

  H.323 soporta numerosas formas de identificación de usuario, soporta varios tipos de direccionamiento (DNS, ENUM, TRIP), de forma que un usuario se puede registrar en el GateKeeper dejando como identificador un nombre escogido por él (o por el GateKeeper) con su número de teléfono, por ejemplo, y otro puede hacerlo dejando un URL similar a una dirección de correo electrónico. El GateKeeper se encarga pues de la función de traducir el nombre del usuario que un llamante quiere localizar a su correspondiente dirección real.

- Control de admisión: permite o no registrarse a los clientes de la LAN en función de varios criterios.

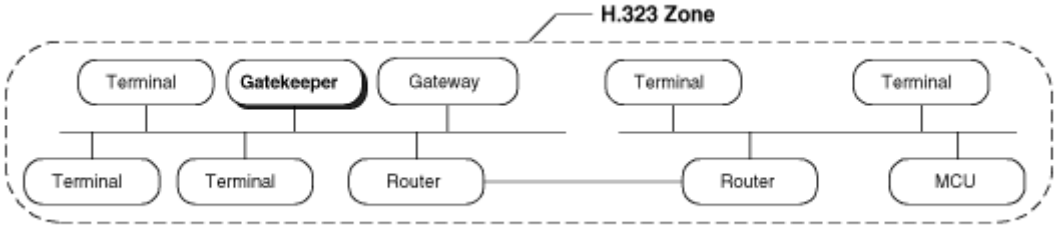
  En principio H.323 fue diseñado con redes de área local en mente, extendiéndose su ámbito más adelante a redes de área extensa. Aún así, la idea original de situar un GateKeeper en cada LAN persiste, aunque no sea necesariamente así. Un cliente puede registrarse en cualquier GateKeeper que se lo permita, éste mantiene un directorio con la dirección real del cliente y el alias H.323 que éste está utlizando. Los criterios de admisión pueden ser desde permitir registrarse a cierto tipo de direcciones hasta por ejemplo admitir un número máximo de clientes registrados.

- Control de ancho de banda: para la gestión del ancho de banda disponible.

El GateKeeper puede denegar a un cliente el establecer una conferencia si estima que todo el ancho de banda de una conexión está siendo utilizado, por ejemplo, esto puede servir para ayudar a mantener una cierta calidad de servicio.

- Gestión de zona: el GateKeeper ofrece sus servicios a los elementos H.323 que se hayan registrado en su zona de control.

**Figura 3-6. Zona de un GateKeeper**




La *zona* de un GateKeeper se define sin más como el conjunto de entidades H.323 que se hayan registrado en el mismo. Aunque teóricamente hay uno por zona, como se desprende de la propia definición de zona, se pueden establecer GateKeepers alternativos para mayor robustez frente a fallos y balance de carga.

Además, el GateKeeper puede asumir otras funciones opcionales, como autorización de usuarios, gestión de conferencias y control de señalización de llamada, entre otras.

El uso de un Gatekeeper en la red H.323 es opcional, pero si hay uno, ha de ser utilizado. Es decir, cuando un cliente H.323 se pone en funcionamiento, automáticamente busca un GateKeeper de la forma en que veremos más adelante. Si no lo encuentra, no se registra con él y además de no recibir ninguno de los servicios asociados al mismo, ha de proporcionar direcciones reales para la red donde está ubicado, o hacer uso de algún mecanismo alternativo para resolver direcciones ficticias, como por ejemplo directorios ILS o similares. Si lo encuentra, entonces está obligado a registrarse con él (a menos que el GateKeeper lo impida, evidentemente) y puede hacer uso de sus capacidades.

¿Qué se ha de tener en cuenta en la red al añadir este nuevo elemento? Automáticamente los clientes intentan descubrir si hay un Gatekeeper disponible mediante el protocolo RAS mencionado anteriormente, enviando mensajes a los puertos UDP 1719 (para unicast) o al 1718 (para multicast), aunque por compatibilidad hacia atrás el puerto 1718 también debe aceptar mensajes unicast. Éstos son los puertos donde el GateKeeper debería estar escuchando las peticiones

de los clientes, y como a priori la dirección del GateKeeper es desconocida, el mensaje RAS se envía a toda la red. El uso de mensajes multicast es la principal variación en la red, si se desea usar multicast habrá que habilitarlo en el nivel IP, aunque es totalmente opcional, como acabamos de ver.

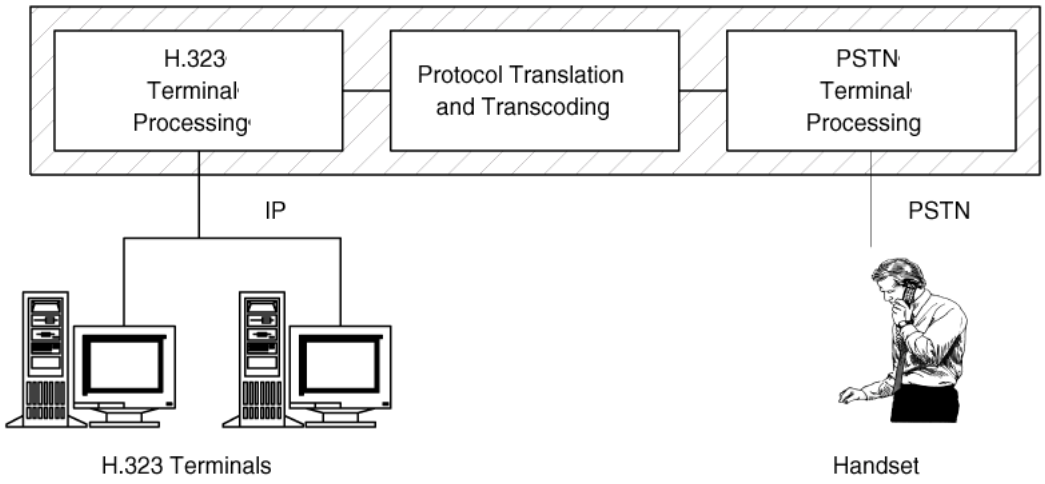
## 3.4.3. Elementos avanzados

En una red H.323 pueden coexistir más elementos, desde otras entidades H.323 hasta elementos ajenos a la norma que pueden dificultar su buen funcionamiento, es por ello que vamos a analizar los más comunes viendo qué riesgos presentan y qué posibles soluciones podemos adoptar. Conviene tener en mente que algunas de las entidades H.323 pueden encontrarse en el mismo sistema o incluso ser parte de la misma aplicación, como por ejemplo un GateKeeper que además haga las tareas de Proxy.

### 3.4.3.1. Pasarelas a la Red Telefónica Conmutada (RTBC)

En H.323 estos elementos son conocidos como Gateways H.323, y se encuentran conectados por una parte a nuestra red, y por otra parte a una red ajena, como por ejemplo la de telefonía básica, encargándose de traducir entre formatos y procedimientos de comunicación, códecs, establecimiento y liberación de llamada, etc.

**Figura 3-7. Pasarela a la RTBC**




Desde el punto de vista de nuestra red son tratados como un terminal más, eso sí, el GateKeeper ha de tener en su directorio la información de que las llamadas con destino en la RTB a la que está conectado el Gateway han de ser dirigidas al mismo.

### 3.4.3.2. Intermediarios H.323

Conocidos con el nombre de Proxy H.323, son elementos que se encargan de cursar el tráfico H.323 a través de ellos, estableciendo con el otro extremo su propia conexión, actuando de intermediario de toda la comunicación. De esta forma el otro extremo no interactúa directamente con el terminal del usuario que se encuentra detrás del Proxy.

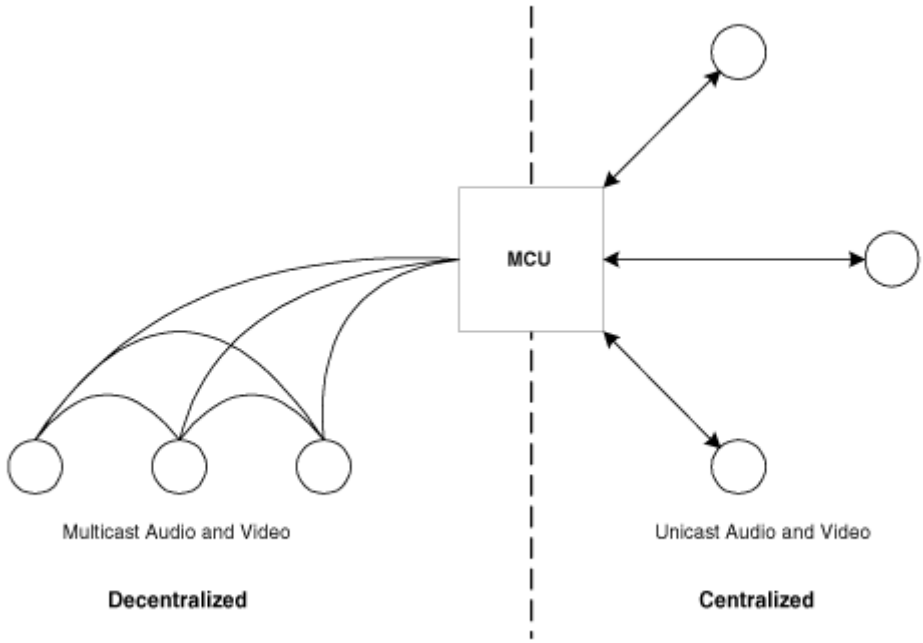
Cabe señalar que la función principal de un Proxy H.323 es dotar de un mecanismo adicional de seguridad a la red, como tal, pues, no se suele encontrar por sí solo, sino como parte de un GateKeeper o de un Firewall.

### 3.4.3.3. Unidades de Control Multipunto

Las Multipoint Control Units (MCU) dan soporte a conferencias entre tres o más terminales H.323. Una MCU consiste en al menos un Controlador Multipunto (Multipoint Controller, MC), requerido, y cero o más Procesadores Multipunto

(Multipoint Processors, MP). La función del MC consiste en manejar las negociaciones H.245 entre todos los extremos para acordar el uso de códecs comunes de audio y vídeo, así como en controlar recursos, determinando si alguno de los flujos va a ser multicast. El MP se encarga de la mezcla, alternancia, y procesado de los flujos de audio y vídeo (y datos, si es necesario).

**Figura 3-8. Unidad de Control Multipunto**




### 3.4.3.4. Firewalls

Los Firewalls, al ser elementos no transparentes de la red, presentan serios problemas a las comunicaciones H.323. Como ya hemos visto, aunque usualmente los puertos utilizados por H.323 son bastante conocidos, y muchos clientes permiten hacer tunelado H.245, en general necesitamos utilizar puertos en el rango 1024-65535 (tanto UDP como TCP), negociados dinámicamente, lo cual provoca problemas bastante serios a la hora de atravesar un Firewall en el camino.

Se necesita, por tanto, que el firewall tenga un módulo capaz de detectar H.323 y permitir el paso de los canales RTP y RTCP negociados, así como el del H.245 si fuese necesario. Nótese que al usarse un canal distinto en cada sentido de audio y/o

vídeo, si el firewall no es consciente del uso de H.323, independientemente de quién inicie la llamada, al menos un sentido de la comunicación no podrá ser establecido, desde fuera a dentro del Firewall, normalmente.

#### 3.4.3.5. Mecanismos de NAT/PAT

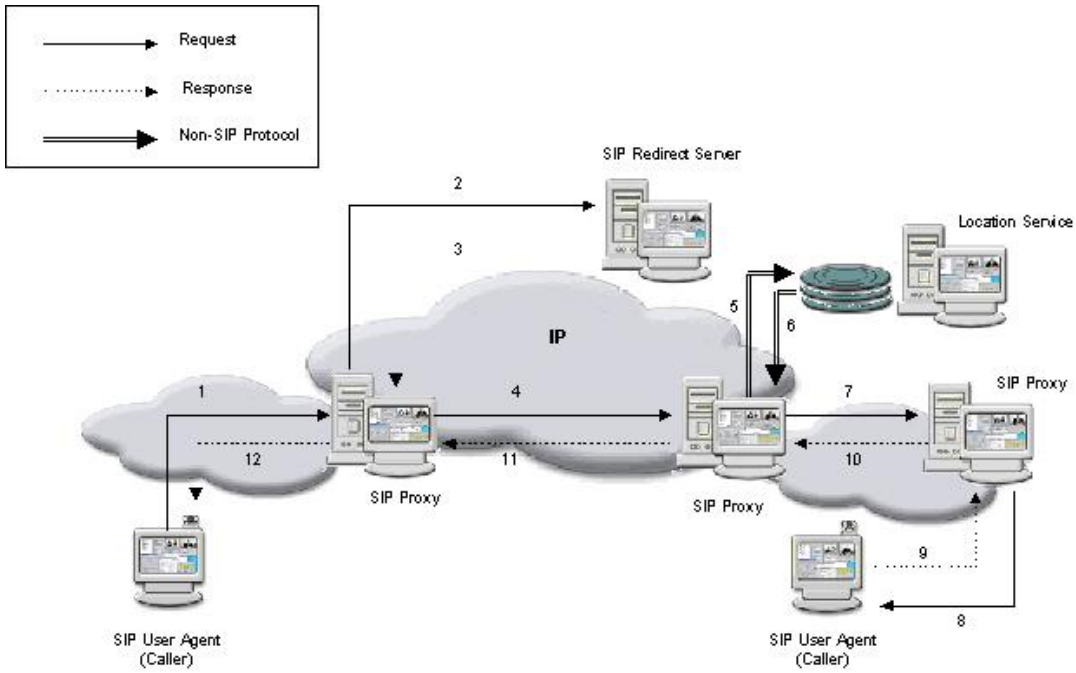
Los dispositivos que realizan NAT (Network Address Translation) o PAT (Port Address Translation), en principio tendrían un problema similar. Al intentar establecer una llamada desde un terminal que se encuentre detrás del NAT/PAT, éste se encargará de sustituir su dirección/puerto por otros adecuados a su caso. Dejando de lado el hecho de la traducción de direcciones, los propios mensajes de H.323 contienen la dirección del terminal, que suele ser una dirección privada, que pasa transparentemente a través del NAT. Aunque el mensaje llegue a su destino (por ejemplo, un GateKeeper fuera del ámbito del NAT), éste intentaría contestar a una dirección privada, es decir, carente de sentido más que dentro de su propia red. Esto es sólo un ejemplo de la complejidad del problema, que de nuevo se evita dotando al NAT de un módulo consciente de H.323 que se ocupe de mirar los mensajes H.323 y realizar las traducciones pertinentes. En este sentido cabe destacar que algunas aplicaciones proporcionan la opción de realizar esta tarea directamente, de forma que no es necesario que el NAT sea consciente del uso de H.323. En cualquier caso, el mapeado entre direcciones públicas y privadas que realiza el NAT no siempre permitirá las llamadas simultáneas de varias o todas las máquinas de la red interna, y la situación habrá de estudiarse caso por caso.

## 3.5. Protocolo SIP

SIP (Session Initiation Protocol) es un protocolo de sesión diseñado por el IETF, definido inicialmente en la RFC 2543, y más adelante redefinido en la RFC 3261. Como su propio nombre indica, es un protocolo cuyo objetivo consiste en proporcionar el mecanismo para establecer, modificar, y abandonar una sesión entre dos terminales, así como poder localizar y registrar usuarios, quedando fuera de su ámbito el tipo de datos a transmitir en la sesión o la forma de transporte de los mismos. A este efecto va acompañado de otro protocolo, llamado Session Description Protocol (SDP), especificado en la RFC 2327, cuya tarea consiste

precisamente en describir los detalles de la llamada, desde los códecs a utilizar para los distintos tipos de datos, como audio o vídeo, hasta los puertos especificados para ello.

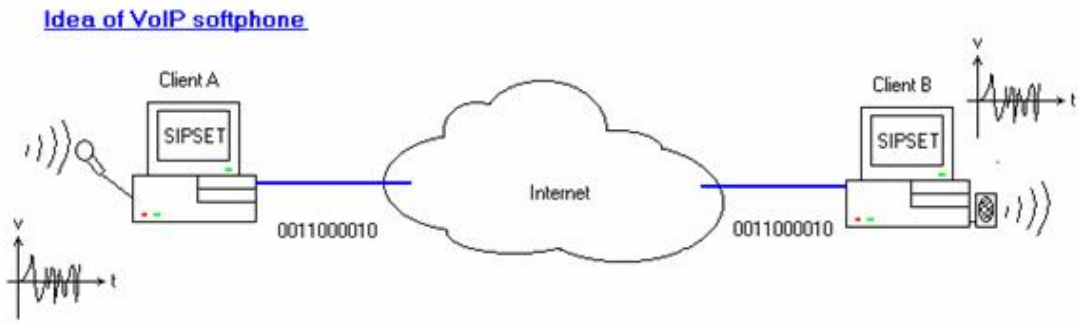
**Figura 3-9. Visión general de una red SIP**




## 3.5.1. Llamada básica en SIP

Al igual que ocurre con H.323, tan sólo son necesarios a priori los terminales SIP para establecer una comunicación. Sin embargo, SIP tan sólo se encarga en este caso del inicio de la sesión, dejando en manos de otros protocolos el resto de detalles de la llamada. En general, será el protocolo SDP quien se encargue de estos detalles, aunque también se pueden codificar como MIME o de la forma en que se quiera. En cualquier caso, estos detalles no forman parte del protocolo SIP y se incluyen como cuerpo del mensaje SIP de intento de establecimiento de la conexión.

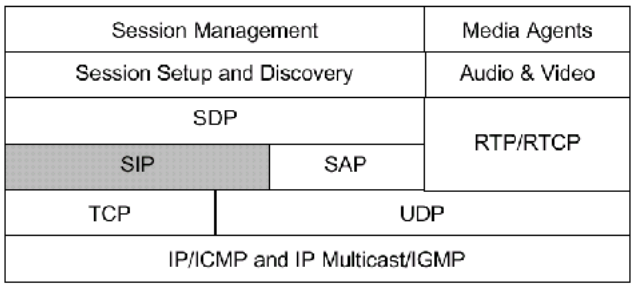
**Figura 3-10. SIP entre terminales**




Cada terminal SIP (UA, User Agent, Agente de Usuario) se compone, pues, de una parte cliente y de una parte servidor, en función de quién esté iniciando la sesión. Normalmente el puerto que utiliza SIP es el 5060, tanto en TCP como en UDP, pero en los mensajes SDP contenidos en la petición se proporciona información sobre los puertos UDP a usar para RTP/RTCP, un par de puertos por sentido y tipo de datos, al igual que ocurría con H.323. De nuevo estos puertos son escogidos dinámicamente en el rango 1024-65535.

Por tanto, para establecer una conferencia SIP tenemos los mismos requisitos en la red que al establecer su equivalente en H.323, disponer del protocolo de transporte TCP ó UDP, y del protocolo de red IP. En resumen, una red TCP/IP normal y corriente. De nuevo tenemos las mismas condiciones acerca de la calidad de servicio que teníamos presentes con H.323.

**Figura 3-11. Torre de protocolos SIP**



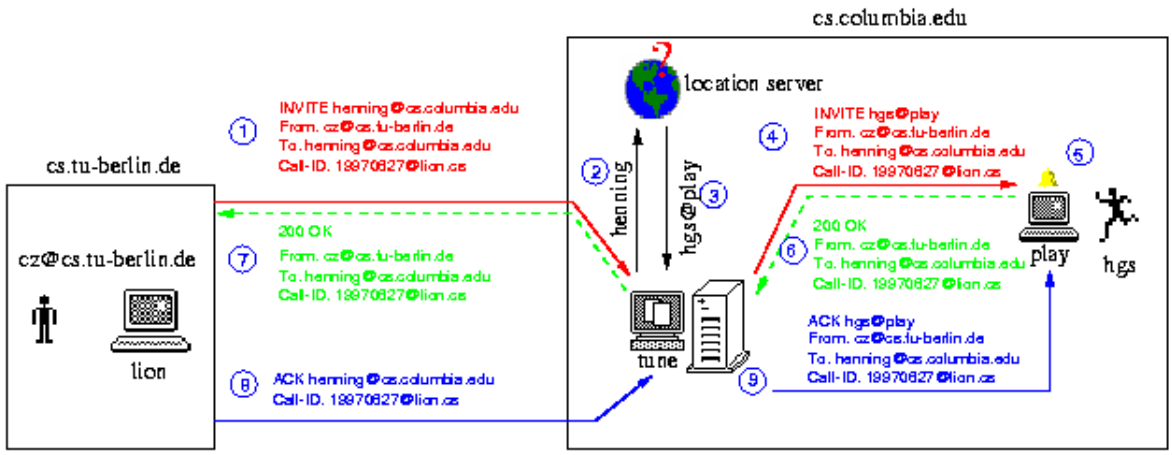
## 3.5.2. Elementos adicionales de la arquitectura SIP

Pasamos a introducir un nuevo elemento de la arquitectura SIP, el servidor SIP. En general un servidor SIP se implementa de forma que cumple tres funciones, puede hacer de intermediario (SIP Proxy Server), redirigir una llamada (SIP Redirect Server) e incluso registrar a un usuario en una base de datos para actualizar su localización (SIP Registrar). Un servidor SIP puede implementar de estas funciones las que quiera, dependiendo de cuánto queramos que haga.

### 3.5.2.1. Llamada con intermediario

En el caso de utilizar el servidor como Proxy, el agente de usuario llamante se pone en contacto con él, especificando con quién quiere establecer la sesión, y será el Proxy el encargado de resolver la dirección SIP proporcionada (mediante consulta a un directorio, por ejemplo) y ponerse en contacto con el agente de usuario llamado actuando de parte del llamante. Cuando la sesión se ha negociado con éxito, los agentes de usuario se ponen directamente en contacto, ya mediante el protocolo RTP para la transmisión de audio y vídeo, aunque esto no es estrictamente necesario y el Proxy podría actuar de intermediario incluso para la videoconferencia. Esto puede tener su interés en los casos en los que los agentes de usuario estén situados detrás de firewalls, por ejemplo, ya que simplificaría la tarea de éstos grandemente.

**Figura 3-12. Conferencia SIP con un solo Proxy**



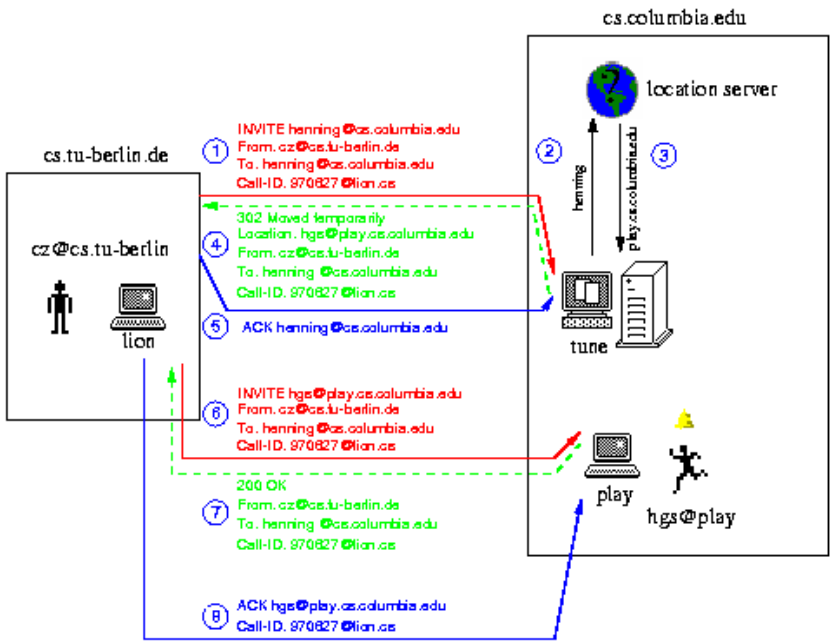

La estructura habitual en SIP se conoce como "el trapezoide SIP", debido a la

representación gráfica típica de la misma, y se basa en que el agente de usuario se pone en contacto con su Proxy (situados habitualmente en la misma red) y éste a su vez contacta con el Proxy de la red del agente de usuario llamado, una vez establecida la sesión lo habitual es que la conferencia sea directamente entre los agentes de usuario.

### 3.5.2.2. Llamada con redirección

A veces, un servidor no actúa de Proxy, sino que sencillamente redirige la llamada. El agente de usuario se pone en contacto con el servidor, indicándole con quién quiere establecer la sesión, y el Redirect Server cumple la función de averiguar la dirección o direcciones donde se puede encontrar a la persona llamada, pero sin ponerse en contacto con su agente de usuario, al contrario de lo que ocurría con el Proxy. En vez de eso le proporciona al llamante esta información, para que éste se ponga en contacto con el llamado.

**Figura 3-13. Redirección en SIP**




### 3.5.2.3. Registro de localización

La última entidad SIP es el Registrar, cuya función, como su nombre indica, es

registrar a los usuarios que lo soliciten, almacenando en un directorio su identidad o alias de SIP y la dirección o direcciones donde pueden ser localizados. Este directorio será posteriormente consultado por los elementos anteriores para resolver la localización de los usuarios llamados.

### 3.5.3. Elementos avanzados

Al igual que ocurría con H.323, en una arquitectura SIP nos encontramos con más elementos de los que hemos mencionado, y a efectos de nuestro estudio sobre la calidad de servicio, y sobre las trabas que nos encontramos en la red por elementos no transparentes, es aplicable lo mismo que mencionamos en H.323 con respecto a firewalls y mecanismos de NAT/PAT: de nuevo necesitaríamos un elemento capaz de mirar en el contenido de los mensajes SIP, y que reemplace las direcciones y/o puertos de forma que se mapeen correctamente las vertientes públicas y privadas de la comunicación. Ese elemento aplicado a firewalls deberá entender el protocolo SIP y sus contenidos, para permitir el paso por los puertos negociados de las comunicaciones, la complejidad de esto puede tener su relevancia, dado que implica que este módulo ha de saber interpretar SDP también, o el protocolo o formato con que se decida anunciar las capacidades de la sesión.

Por último mencionar que este elemento capaz de permitir el paso a través de firewalls, NAT y PAT suele recibir el nombre de Application Level Gateway (Pasarela de Nivel de Aplicación).

## 3.6. Comparación entre H.323 y SIP

Vamos ahora a entrar a comparar las dos aproximaciones, pero en vez de hacerlo de forma general, nos vamos a centrar en los puntos que atañen a nuestro estudio, básicamente, aquellas diferencias entre ambos que puedan dar pie a diferencias en la calidad de servicio, o más concretamente, a la resolución del problema que planteamos de poder garantizar esta calidad. Para ello hemos estudiado las características de ambos, deduciendo luego cual de los dos, si alguno, ofrece mejores condiciones para el trato de la calidad de servicio. En la tabla se muestran las diferencias más relevantes entre ambas aproximaciones.

**Tabla 3-3. Comparación general entre H.323 y SIP**

| H.323 | SIP |
|---|---|
| Se diseña con requerimientos de conferencia multimedia en mente, siguiendo una estructura unificada que pretende dar el mismo nivel de robustez e interoperabilidad que la RTB. | Se diseña de forma modular y su propósito es establecer una sesión genérica entre dos puntos, sin centrarse específicamente en una conferencia multimedia de vídeo y audio. |
| Estandarización de control de llamada y servicios suplementarios perfectamente definida y compatible con los servicios de telefonía tradicional. | Falta de madurez en cuanto a dicha estandarización, el IETF está trabajando sobre ello para dotar de un sistema de servicios suplementarios robusto a SIP, que incluya interoperabilidad con la telefonía tradicional, así como con H.323. |
| Provee mejor funcionalidad e interoperabilidad con respecto a servicios suplementarios. Esto es porque se separa perfectamente la parte de servicios suplementarios de la parte de control de la llamada básica. | La llamada básica en SIP se extiende incluyendo elementos como servicios suplementarios, lo que puede dar lugar a problemas de interoperabilidad y compatibilidad entre distintas plataformas. |
| H.323 está diseñado para dar soporte a VoIP, su soporte para otras aplicaciones es muy limitado, y ha sido añadido posteriormente. | SIP permite que un amplio rango de aplicaciones funcionen bajo el mismo protocolo, donde VoIP es sólo una posibilidad más. Provee de mecanismos mucho más efectivos para manejar sesiones que no sean VoIP. |
| Los clientes H.323 han de implementar toda la funcionalidad de H.323, incluso si parte de la misma no es requerida en un momento dado. | Los clientes SIP son ligeros, dado que sólo han de implementar el protocolo SIP y aquellas cosas que sean necesarias para la aplicación, pudiendo ignorar las que no lo sean. |

De cualquier forma, en un futuro parece que se tiende a una convergencia SIP-H.323, o al menos a una coexistencia donde ninguno suplantará al otro, sino que serán utilizados en función de las necesidades de cada escenario específico. Lo que si cobra especial relevancia es el diseño de pasarelas SIP-H.323 para la interoperabilidad entre esta redes.

En cuanto al problema de la calidad de servicio, las cosas están bastante equilibradas, H.323 tiene más mecanismos que SIP para dar una mejor calidad de servicio, pero al estar hablando de una red IP, de nuevo, no hay garantías acerca de la misma, y al depender tanto H.323 como SIP del protocolo RTP/RTCP, la capacidad de controlar mediante una aplicación, por ejemplo, la calidad de servicio ofrecida es prácticamente la misma en ambos casos.

# 3.7. Implementaciones

En este apartado vamos a revisar la disponiblilidad de productos de telefonía IP basados en H.323 o en SIP que puedan ser interesantes para nuestra red inalámbrica rural. A pesar de que H323 está muy maduro, SIP empieza a imponerse como una solución más sencilla y atractiva.

## 3.7.1. Productos software libres

Los productos de software libre están basados generalmente en unas pocas bibliotecas que dictan sus características de bajo nivel a las aplicaciones que se montan sobre ellas.

### 3.7.1.1. H.323

Respecto a H.323 destaca el proyecto OpenH323 (http://www.openh323.org/), liderado por Quicknet Technologies, empresa que se dedica a servicios de telefonía sobre Internet. Esta biblioteca multiplataforma (Windows, Linux, BSD) implementa la versión 4 del protocolo, implementa o soporta por software los códecs de audio GSM-06.10, LPC-10 ley $\mu$, G.711 leyes A y $\mu$ y Speex, y soporta G.723.1, G.728 y G.729 con hardware adecuado (Quicknet Phone Cards, por ejemplo). También soporta vídeo (H.261 y, parcialmente, CU30).

Sobre OpenH323 se han construido terminales, como OpenPhone y OhPhone (http://www.openh323.org/), OpenPhone es la versión para Windows, OhPhone para Linux y BSD. Pero el más importante, conocido y con intefaz más amigable, es GnomeMeeting (http://www.gnomemeeting.org), sólo para Linux. Ver el apartado exclusivo de Gnomemeeting más adelante.

Además existen otros productos derivados de esta biblioteca, el GateKeeper OpenH323GK (http://www.gnugk.org), por ejemplo, uno de los más avanzados con múltiples bondades, como autenticación y contabilidad por RADIUS, capacidad de penetrar de forma correcta a través de dispositivos NAT, funcionalidad de Proxy H.323, incluso enrutando el tráfico de tiempo real a través suyo, soportando QoS al añadir el valor del campo de calidad de servicio a los paquetes RTP/RTCP, Además puede aparentar ser un terminal o un gateway, lo que permite construir jerarquías.

También son interesantes PSTNGW, que es una sencilla pasarela entre H.323 y la red telefónica, con funcionalidad muy básica, pero efectiva (requiere, por supuesto, hardware adecuado, en este caso de QuickNet) y OpenMCU que permite conferencia simultánea de hasta 4 interlocutores.

#### 3.7.1.2. SIP

Respecto a SIP destaca el proyecto GNU Open SIP (http://www.gnu.org/software/osip/), hecho en el lenguaje de programación C, con lo que sólo depende de las bibliotecas del mismo, ocupa poco y se puede usar para implementar tanto softphones, como servidores proxy, como usarlo de forma empotrada. Provee únicamente de un API para los parsers de SIP, SDP, y la biblioteca de manejo de transacciones SIP, según lo definido en la RFC 3261. Entre las aplicaciones creadas a partir de oSIP encontramos entre otras un softphone, Linphone (http://www.linphone.org/), con soporte para varios códecs incluyendo Speex, para las plataformas FreeBSD y Linux, o Partysip (http://www.nongnu.org/partysip/), un servidor Proxy desarrollado por WellX Telecom, con soporte para Windows, Linux y FreeBSD, que implementa además las funciones de servidor Registrar y Redirect. Por último existe una nueva iniciativa, eXosip (http://savannah.nongnu.org/projects/exosip/) (eXtended osip library) que pretende proveer de un API de nivel superior que facilite el desarrollo de aplicaciones para el usuario.

Por otra parte existe otra biblioteca SIP llamada Vovida (http://www.vovida.org/), que presenta soporte completo para la RFC 2543, y en la cual se está trabajando hacia la compatibilidad total con la RFC 3261. Vovida es una iniciativa liderada por Cisco que soporta múltiples aplicaciones (http://www.vovida.org/applications/), desde softphones como SIPSet, utilidades para teléfonos Cisco (SIPTiger), o incluso proyectos completos como el Vovida Open Communications Applications Library (VOCAL), que incluye servidores SIP de múltiples tipos, así como traductores de

protocolo de SIP a H.323 y de SIP a MGCP.

El NIST (http://dns.antd.nist.gov/proj/iptel/) (Instituto de Normalización Norteamericano) también tiene una biblioteca en Java abierta con la que se han hecho terminales como BonePhone (http://www.iptel.org/products/bonephone/home.html).

Al margen de estas tres grandes bibliotecas, existen productos de menor difusión basados en otras, pero que pueden presentar un cierto interés al tratarse de iniciativas de software libre, es el caso del softphone SIP KPhone (http://www.wirlab.net/kphone/), parte del popular entorno gráfico para Linux KDE (http://www.kde.org/), que permite realizar videoconferencia usando varios códecs, tanto de audio (G.711 ley $\mu$, GSM, iLBC) como de vídeo (H.261, H.263). O como el servidor SIP Express Router (http://www.iptel.org/ser/), que hace las funciones de servidor Proxy, Registrar y Redirect, disponible para las plataformas Linux, Solaris, FreeBSD, y Windows.

#### 3.7.1.3. Otros

Existen proyectos de hardware abierto como TuxScreen (http://www.tuxscreen.net/) y TuxScreen SIP (http://www.iptel-now.de/HOWTO/TUX_SIP/tux_sip.html), que usan los elementos software anteriores sobre Linux empotrado, y que se han llegado a fabricar en pequeñas cantidades, pero no de una forma industrial, aceptable para nuestro proyecto. Hay, si embargo, al menos un fabricante que comercializa teléfonos IP SIP y H.323 basados en Linux, Snom (http://www.snom.com), soportando únicamente GSM 6.10 como codec eficiente no patentado. Seguramente podría añadírseles un codec Speex, ya que Snom proporciona los manejadores de su hardware, aunque no la aplicación de telefonía.

Aunque diseñada su propio protocolo de Voz sobre IP (IAX), Asterisk (http://www.asterisk.org) es una centralita de fuente abierta que soporta gran cantidad de hardware, e interopera con H.323, SIP y MGCP. Para IAX tiene su propio softphone: Gnophone (http://www.gnophone.com).

Relacionado con Asterisk, aunque probablemente menos desarrollado, está el servidor de telefonía de GNU, Bayonne (http://www.gnu.org/software/bayonne/), recientemente integrado con OpenH323.

### 3.7.1.4. Terminales software propietarios

Existen numerosos terminales software propietarios, de los que nos interesan sobretodo los de entorno Windows, ya que es la plataforma no libre que se usa en EHAS. De ellos hay dos que se obtienen gratuitamente de Microsoft. Otros están muchas veces ligados a proveedores de telefonía IP en internet, y pueden ser utilizables o no en un entorno abierto.

NetMeeting (http://www.microsoft.com/windows/netmeeting/) es el producto H.323 [1] de videoconferencia más difundido. Sin embargo parece que Microsoft no está muy interesado en mantener este producto, favoreciendo el de mensajería instantánea, MSN Messenger (http://messenger.msn.com/), que debidamente configurado funciona como terminal SIP.

### 3.7.1.5. Terminales hardware

Existen numerosísimos tipos de teléfonos IP conectables a ethernet, siendo los más difundidos los de Cisco (http://www.cisco.com/en/US/products/hw/phones/ps379/), que muchas veces se utilizan como patrón de interoperabilidad, aunque en algunos modelos requieren *pasarelas de telefonía* propietarias. Es común encontrarlos multiprotocolo, difundiéndose cada vez más los que soportan SIP. Pero también es muy común encontrarlos que sólo interoperan entre sí. Empiezan también a difundirse versiones inalámbricas, ya sea Bluetooth (BlooPhone (http://test.bloophone.com/Products.asp)), que no nos interesan, como WiFi (WiFiPhone (http://www.wifiphone.org/products.html)), que podrían utilizarse a nivel de aldea, si es cubierta con un router o un punto de acceso inalámbrico, lo que puede ser intersante para interconsulta. También existe un mercado importante de Videoteléfonos (Innomedia (http://www.innomedia.com/videophone/videophone.htm), Bosslan (http://www.bosslan.com/BOSSVTEL.htm)), aunque en principio no está previsto su uso en EHAS, por su elevado precio.

Una alternativa interesante a los teléfionos IP son las pasarelas a teléfono ordinario (FXS, como el CISCO ATA-186 (http://www.cisco.com/univercd/cc/td/doc/pcat/ata186.htm)), que permiten conectar uno o más teléfonos ordinarios, robustos y muy económicos, a la red IP. A veces están integradas en conmutadores o routers (Bosslan (http://www.bosslan.com/BOSSTELBOX1S.htm)), ofreciendo una alternativa muy económica.

## 3.7.2. GateKeepers, servidores SIP y pasarelas

Ya hemos visto productos de software libre que tienen capacidad suficiente para ejercer de servidores SIP o Gatekeepers. Existen soluciones software propietarias que funcionan en servidores Windows o Unix propietarios, que no vamos a considerar, ya que EHAS no utiliza servidores propietarios. No obstante, podrían utilizarse estos componentes cuando vienen integrados en un dispositivo autónomo económico, normalmente una pasarela a telefonía y/o un router (Bosslan PBX (http://www.bosslan.com/BOSSTELPBX2.htm), Cisco Multimedia Conference Manager (http://www.cisco.com/warp/public/cc/pd/iosw/ioft/mucvmn/prodlit/ipmcm_ds.htm), Tenor GateKeeper (http://www.quintum.com/main/voip_e_tenor_gatekeeper.html)).

Existe la posibilidad de construir pasarelas con tarjetas más o menos inteligentes que se conectan a los ordenadores. Esta solución, aunque posiblemente algo más barata que la de utilizar productos autónomos cerrados, puede tener problemas de mantenibilidad (manejadores para el sistema operativo, etc). No obstante, pueden hacerse pequeñas pasarelas con el software del proyecto OpenH323 y tarjetas QuickNet (http://www.linuxjack.com/), patrocinadores del proyecto. Más soporte de tarjetas tiene Asterisk (http://www.asterisk.org), por ejemplo, aunque en estos momentos desconocemos sus capacidades H323 y SIP.

## 3.7.3. Estudio de las aplicaciones H.323 más conocidas

Por su gran difusión vamos a describir en detalle las aplicaciones Netmeeting y Gnomemeeting, centrándonos en los puertos utilizados por ambas y en cómo marcan los paquetes IP que contienen los datos en tiempo real. Como ya hemos mencionado, esto nos va a servir para realizar nuestro estudio de calidad de servicio.

### 3.7.3.1. Netmeeting

Aplicación de Microsoft que soporta videoconferencia H.323 y transferencia de datos mediante el protocolo T.120. Usa los siguientes puertos:

- Puerto TCP 389: Para el servicio de ILS (Internet Locator Server).
- Puerto TCP 522: Para el servicio de ULS (User Location Server).

- Puerto TCP 1503: Para la transferencia de datos mediante el protocolo T.120, incluye mensajería de texto.
- Puerto TCP 1720: Para el establecimiento de llamada de H.323, mediante el protocolo H.225.0.
- Puerto TCP 1731: Para control de la llamada de audio.
- Puerto TCP dinámico (1024-65535): Control de canal mediante H.245. Es de notar que Netmeeting no soporta tunelado de H.245 sobre H.225.0.
- Puertos UDP dinámicos (1024-65535): Para el flujo de datos en sí, mediante RTP/RTCP, tanto audio como vídeo.

Como se puede observar, realmente el único tráfico que queremos priorizar es el del protocolo RTP/RTCP, que es el que tiene el requisito de tiempo real, conocer el resto de los puertos usados por la aplicación puede ser útil para poder atravesar firewalls o dispositivos NAT/PAT, pero no tienen ninguna necesidad de ser tratados de forma especial. Por tanto nos fijamos en los puertos usados por RTP y descubrimos que son impredecibles. Afortunadamente, examinando los paquetes que el Netmeeting envía por la red vemos que marca el octeto de tipo de servicio de la cabecera IP de la siguiente forma:

- Paquetes de audio: El octeto toma el valor hexadecimal 0xa0 (en binario 1010 0000).
- Paquetes de vídeo: El octeto toma el valor hexadecimal 0x60 (en binario 0110 0000).

De esta forma vemos que es posible diferenciar entre audio y vídeo si utlizamos Netmeeting, esta información nos será de utilidad al analizar qué debemos hacer para garantizar la calidad de servicio.

#### 3.7.3.2. Gnomemeeting

Aplicación de videoconferencia de software libre, ofrecida bajo la GPL (General Public License). Utiliza los siguientes puertos:

- Puerto TCP 1720: Para el establecimiento de llamada H.323 mediante H.225.0, soporta tunelado de H.245, así que no es necesario establecer otra conexión para

éste.

- Puertos TCP en el rango 30000-30010: Si no se utiliza tunelado de H.245, el control de canal estará en este rango de puertos. Es necesario habilitarlo si se pretende establecer una comunicación con aplicaciones que no soporten tunelado H.245, como por ejemplo, el arriba mencionado Netmeeting.
- Puertos UDP en el rango 5000-5003: Para envío y recepción de los paquetes RTP/RTCP de audio y vídeo.
- Puertos UDP en el rango 5010-5013: Para comunicarse con un GateKeeper, sólo se utilizan si nos registramos con uno, evidentemente.

Una de las grandes ventajas que ofrece el Gnomemeeting es la posibilidad de configurar todos y cada uno de estos puertos y rangos de puertos de la forma que queramos, poniendo los valores que consideremos adecuados o necesarios.

En cuanto a cómo marca el Gnomemeeting sus paquetes de audio y vídeo, lo hace igual para ambos. Gnomemeeting está basado en la biblioteca OpenH323, que por defecto marca el campo de tipo de servicio de IP con el valor hexadecimal 0x10 (0001 0000 en binario), y por tanto otorga el mismo valor a sus paquetes, no obstante, al tratarse de código abierto sería muy sencillo modificar la aplicación para que otorgue distintos valores al audio y al vídeo, de hecho su propio autor ha expresado su intención de hacer este parámetro también configurable en futuras versiones.

Vemos en este ejemplo, pues, que podemos controlar qué puertos utiliza la aplicación y cómo marca los paquetes, de nuevo esta información nos puede resultar muy útil más adelante.

## 3.8. Calidad de servicio en VoIP

La gestión de la calidad de servicio en IP con servicios diferenciados se basa en la capacidad de todos los routers de la red para, observando el campo TOS de la cabecera IP u otros campos, ser capaces de clasificar y marcar el tráfico para darle un trato diferenciado Esto ha de ocurrir en todos los nodos, si se desea que esta técnica sea efectiva.

En la red que estamos diseñando, el trato de este octeto de la cabecera IP es crucial para garantizar la calidad de servicio, puesto que nosotros pretendemos aplicar

esa gestión a nuestro caso concreto: la videoconferencia sobre IP. Para ello vamos primero a ver sucintamente qué mecanismos tenemos en IP para controlar la QoS, aplicándolo después al caso que nos ocupa.

Nuestro objetivo final va a ser, por tanto, disponer de una red donde el tráfico de voz tenga máxima prioridad, garantizando así un servicio de telefonía, dejando en un segundo plano el tráfico de vídeo, para dar un servicio de videoconferencia supeditado al de voz. Por último, podremos hacer uso del resto de recursos de la red siempre y cuando estos no interfieran con las (video)conferencias.

### 3.8.1. Calidad de servicio en IP

Históricamente, la calidad de servicio se define en la cabecera IP en el segundo octeto, llamado TOS (Type of Service), según lo convenido en las RFC 791 y RFC 1349. Más adelante se redefinió su significado pasando a llamarse octeto DS (Differentiated Services) según la RFC 2474, que deja obsoletas a las anteriores. Por tanto, tenemos la opción de utilizar este octeto de una de tres formas posibles:

- Siguiendo las RFC 791 y RFC 1349, que aunque obsoletas, están todavía en uso en muchas aplicaciones actuales.
- Adheriéndonos a la RFC 2474, asegurando la interoperabilidad con otras redes.
- Definiendo por cuenta propia el significado de este octeto, controlando los parámetros de QoS nosotros.

Tomemos la opción que tomemos, hemos de saber cómo identificar los paquetes que llegan al nivel IP desde un nivel superior procedentes de una videoconferencia, bien sea sólo audio o bien incluyendo el vídeo. Es decir, al construir el datagrama IP su octeto que define el tipo de servicio (bien según las RFC antiguas, bien según la nueva, o siguiendo un esquema distinto, para el caso que nos ocupa nos es irrelevante) ha de marcarse en función de lo que venga del nivel superior. Para ello vamos a ver qué recibe el nivel IP desde arriba. En cualquier caso, a efectos de este estudio, llamaremos a este octeto de IP "octeto QoS" o "campo QoS", para que quede patente que el marcado de este octeto es independiente del esquema de marcado que se siga.

### 3.8.2. Aplicación a nuestro caso

Tanto H.323 como SIP, así como el resto de protocolos de videoconferencia mínimamente importantes y eficaces, siguen el mismo esquema: a nivel de aplicación todos usan el protocolo RTP/RTCP para el transporte de audio y/o vídeo. Los paquetes RTP se encapsulan en paquetes de transporte UDP, que son los que le llegan al nivel IP. El problema estriba en que el nivel IP pueda saber que lo que hay en el nivel superior es un paquete de audio o de vídeo, y les asigne la prioridad adecuada. Nos encontramos pues con varias posibilidades:

- La solución trivial, consiste en no hacer nada y esperar que el ancho de banda y la capacidad de procesamiento de la red sean suficientes para bregar con el tráfico de tiempo real.
- IP es consciente del protocolo de nivel superior, TCP ó UDP, por tanto se pueden marcar los paquetes UDP con mayor prioridad. Esto no permite diferenciar entre audio y vídeo y además puede dar prioridad a datagramas que no pertenezcan a una videoconferencia aunque usen UDP.
- Sabiendo que tenemos un paquete UDP entre manos, la tercera opción consiste en intentar averiguar si dentro hay un mensaje RTP/RTCP, o incluso asumir que es así a menos que se demuestre lo contrario. Esto tiene como consecuencia que la interfaz con el nivel IP deba examinar el contenido del paquete UDP, lo que se puede lograr de diversas maneras, y si el contenido es RTP (o asumimos que lo es) se puede deducir, gracias a la existencia del campo fijo PT (Payload Type) de la cabecera RTP, si el contenido es audio o vídeo y marcarlo a tal efecto en el campo de tipo de servicio de IP.
- Otra posible opción, más controvertida, consistiría en hacer la discriminación de los paquetes por puertos, es decir, espiar el contenido de las sesiones H.323 o SIP, averiguar qué puertos se negocian para audio y vídeo, y dar prioridad al tráfico por esos puertos, sería un mecanismo similar al necesario para poder dar servicio H.323 con NAT y/o firewalls.
- Por último cabe la posibilidad de utilizar un sistema de redes privadas virtuales dando prioridad a los paquetes que salgan hacia la red privada (es decir, un túnel con mayor prioridad), o alternativamente usar el protocolo de reserva RSVP para garantizar la calidad de servicio.

Teniendo en cuenta todas estas posibilidades básicas, el esquema propuesto

consiste en analizar varias de estas posibilidades, siguiendo un orden que permita minimizar el tiempo de procesado de cada paquete, de forma que al final, con un margen de error mínimo, demos la prioridad deseada a los paquetes de nuestra videoconferencia. El esquema consistiría, pues, a nivel IP en realizar una serie de pasos que nos permitan deducir el contenido del paquete y marcar el campo de QoS de IP de forma consecuente, a continuación se enumeran los mismos, comentando sus pros y sus contras:

1. Las aplicaciones (o terminales hardware) suelen indicar el valor del campo ToS, tal y como vimos al analizar el Netmeeting y el Gnomemeeting. De esta forma sólo hemos de fijarnos en el valor que la aplicación pretende imponer al octeto y deducir inmediatamente si se trata de un paquete de videoconferencia. Hay que hacer notar dos cosas en esta aproximación, primera, que no todas las aplicaciones marcan este octeto y casi todas lo hacen de forma distinta, luego habría que tener en cuenta al desplegar la red qué aplicaciones se pretenden usar, y segunda, tal y como hemos visto no todas las aplicaciones diferencian entre audio y vídeo, con lo cual nuestro análisis en ése caso no terminaría aquí, sino que deberíamos poder distinguir entre ambos. En cuanto a lo segundo, en principio no debería haber problema, ya que podemos entrar a analizar el campo PT de la cabecera RTP, como veremos más adelante, pero en cuanto a lo primero, nos encontramos con que si nos regimos por ello, estaremos haciendo nuestra red hasta cierto punto dependiente de las aplicaciones escogidas, limitándonos a efectos prácticos. Por último, conviene también recordar que otras aplicaciones podrían marcar sus contenidos como urgentes en la misma forma y recibir el mismo trato diferenciado, si no somos cuidadosos.

2. Evidentemente, los paquetes RTP van encapsulados sobre UDP, que a su vez va encapsulado sobre IP. Dado que todos los paquetes de videoconferencia van sobre UDP, nos basta con examinar el paquete IP y ver el campo que indica en su cabecera el tipo de protocolo que va dentro y si es UDP, darle prioridad. El problema con esta aproximación es que puede haber paquetes que lleguen al nivel IP desde el nivel superior encapsulados en UDP pero que no tengan que ver con la videoconferencia, como por ejemplo relativos a NFS u otras aplicaciones. Aún así, se estima que la proporción de éstos en una red típica es muy pequeña en comparación al número de paquetes de tiempo real pertenecientes a la videoconferencia.

3. Una posibilidad llegados a este punto es, sabiendo que el paquete es UDP,

considerar de voz y/o de vídeo los paquetes comprendidos en un cierto rango de tamaños, a determinar tras analizar los distintos códecs posibles, por ejemplo, los paquetes de audio suelen ocupar entre 40 y 160 bytes, con lo cual paquetes que se salgan de esta escala los podemos descartar del marcado.

4. Combinando las aproximaciones anteriores, y recordando lo que ocurría en el caso del Gnomemeeting, por ejemplo, también sabemos que hay aplicaciones que únicamente utilizan un cierto rango de puertos UDP, con lo que llegados a este punto sabríamos con cuasi total certeza que si, por ejemplo, hay un paquete UDP destinado al puerto 5000 de una cierta máquina, este paquete es de tiempo real. De nuevo el problema que tenemos es que el sistema de diferenciado depende de las aplicaciones usadas.

5. Habiendo comprobado que se trata de un paquete UDP y/o que su puerto de destino se corresponde al utilizado por una aplicación de VoIP, y/o que el paquete está marcado con una cierta prioridad, el siguiente paso consistiría en asumir que se trata de un paquete de tiempo real y tratarlo como tal. Para ello asumimos que se trata de un paquete RTP y realizamos una sencilla comprobación para asegurarnos no de que realmente lo es, sino para descartar aquellos que hayamos marcado como prioritarios hasta ahora erróneamente. Para ello comprobamos los campos fijos y aquellos cuyo valor podemos deducir de la cabecera RTP, y si coinciden con los que deberían ser, asumimos que el paquete es RTP de forma definitiva y examinamos su campo PT, que lleva la codificación del tipo de carga del paquete según lo especificado en la RFC 3551. De esta forma marcamos el paquete con una cierta prioridad para el audio y otra prioridad para el vídeo.

Una posibilidad adicional, no desdeñable a priori, consiste en, viendo que existen soluciones en el mercado que permiten tanto a (algunos) GateKeepers de H.323 y Proxies de SIP actuar de intermediarios enrutando todo el tráfico a su través, incluyendo el de tiempo real, utilizar esos intermediarios para marcar el campo de calidad de servicio de todos los paquetes UDP (recordemos que no podemos hacer una mayor discriminación a nivel IP para ver cuáles son RTP) que sepamos que pertenecen a una videoconferencia. Para saber si es tráfico de videoconferencia una solución muy sencilla consistiría en establecer una interfaz exclusiva con una IP que únicamente se utilice para la aplicación H.323 o SIP, de esta forma, sabemos que todo el tráfico perteneciente a la misma es parte de una videoconferencia, y procederíamos a marcar los paquetes UDP de la misma asumiendo que son RTP según el valor del campo PT de su cabecera, diferenciando entre audio y vídeo. Veamos los principales

problemas con los que nos podemos encontrar:

1. Al hacer que el tráfico de tiempo real pase a través de un intermediario, podemos estar añadiendo un retardo adicional significativo, que puede o no estar compensado con la posterior priorización de los paquetes de tiempo real. Es razonable suponer que el retardo es más o menos el mismo para todos los paquetes, no se produciría degradación de la voz por jitter o efectos similares, pero sí se apreciaría un retardo común. Sería necesario determinar empíricamente la validez de esta solución en este sentido.
2. El intermediario tendría una carga mucho mayor de lo habitual, esto se puede solventar teniéndolo en cuenta antes y dimensionando el sistema adecuadamente, por ejemplo añadiendo más GateKeepers y Proxies SIP, y realizando balance de carga entre ellos.
3. Por último, recalcar que estamos priorizando *todo* el tráfico UDP que atraviesa el interfaz, y en particular y de forma errónea, los paquetes UDP utilizados tanto por SIP como por H.323 para establecer y gestionar sus sesiones. Afortunadamente, esto redunda en cualquier caso en nuestro beneficio, ya que el hecho de priorizar éste tráfico, aunque sea por error, únicamente ayuda al tránsito en la red de los paquetes asociados a la videoconferencia.

Por fin, llegados a este punto deberíamos tener al menos tres tipos de paquetes en la red, los de prioridad normal, los de prioridad máxima, que serían los de audio, y los de prioridad intermedia, que serían los de vídeo. El siguiente paso consiste, pues, en establecer la política de colas en los routers de la red de forma que prioriticen el audio, luego el vídeo, y luego el resto de paquetes.

# Notas

1. Posiblemente versión 1, ya que no soporta procedimientos introducidos en la versión 2 como Fast-Connect.

# Capítulo 4. Batería de pruebas

Queremos que nuestro sistema de VoIP priorice la videoconferencia frente al transporte de datos, y dentro de la videoconferencia en sí, a la voz frente al vídeo. Para ello nos hemos centrado en ver de qué manera podemos diferenciar los paquetes de voz y de vídeo de tiempo real del resto, y cómo darles la prioridad adecuada. Lo que vamos a hacer ahora consiste en realizar pruebas de laboratorio para corroborar que las medidas tomadas dan lugar a una situación en la cual la calidad de la videoconferencia se puede garantizar, con un margen de error mínimo.

## 4.1. Diseño de las pruebas

En primer lugar, vamos a detallar las pruebas a realizar y la plataforma que emplearemos. Para ello, realizaremos las pruebas sobre una red inalámbrica. Aunque la idea original consistía en realizar las pruebas en dos entornos, en primer lugar una red ethernet y acto seguido una red inalámbrica, para así obtener conclusiones sobre si es distinto el comportamiento en ambos casos, se desestimó el hacerlo debido a la dificultad presentada originalmente para hacer funcionar correctamente el conformado de tráfico en la red ethernet, problema que más adelante se solucionó, pero que dejaba poco margen de tiempo para realizar las pruebas en ambas plataformas. Los ordenadores quedarán conectados entre sí en cascada, dado que se trata de simular las topologías típicas a las que nos enfrentaremos en el mundo real.

Los paquetes irán marcados según su prioridad en el campo de tipo de servicio IP, con códigos únicos escogidos para hacer estas pruebas:

- Paquetes de audio: el campo será 0xa0 (1010 0000)
- Paquetes de vídeo: el campo será 0x60 (0110 0000)
- Resto de paquetes: el campo será otro cualquiera

El motivo de escoger estos valores es el poder utilizar luego, de forma directa, los ejemplos de conformado de tráfico que iremos viendo en aplicaciones reales. Así, el cliente de VoIP Netmeeting, presente en casi todas las versiones de Windows, verá sus paquetes priorizados, dado que seguimos su mismo esquema de marcado. Por otra parte, el cliente de software libre Gnomemeeting, aunque por defecto

no proporciona valores de prioridad diferentes para audio y vídeo, permite la modificación de los valores del TOS mediante unas sencillas modificaciones de su código (que utiliza la biblioteca OpenH323, también software libre). Por último, los teléfonos IP que se utilizarán para las pruebas subjetivas (en las que sólo se medirá la calidad del audio) permiten el marcado de los paquetes con el TOS deseado, luego no nos encontramos con ningún problema por ese lado.

Los criterios para decidir si un paquete es audio o vídeo no entran en juego en estas pruebas, ya que directamente se suministrarán a la red con su tipo de servicio fijado en función de lo que queramos evaluar, excepto teóricamente en el caso de las pruebas subjetivas, en las que es la propia aplicación la que debe fijar esos valores. Como en este caso trataremos con *hardphones*, los configuraremos de forma que pongan el TOS adecuado.

Para la realización de las pruebas hemos de tener en cuenta que queremos dar prioridad a ciertos tipos de tráfico UDP, para ello hemos de recordar algunos conceptos básicos sobre redes TCP/IP, a saber, que a priori el tráfico en una red así es tipo best-effort, y que si bien el algoritmo que utiliza TCP para evitar la congestión y hacer un reparto justo del tráfico es válido siempre y cuando sólo intervengan otros flujos TCP, cuando entra un flujo UDP en escena, es el propio TCP el que, por la propia naturaleza del mismo, cede ancho de banda ante el flujo UDP. Por tanto sería inútil tratar de comparar un flujo UDP con otro TCP, puesto que ya sabemos lo que va a pasar, el flujo TCP se acomodará ante uno o más flujos UDP. Teniendo en cuenta que el algoritmo usado por TCP es de arranque lento, y que va incrementando su caudal hasta que recibe notificación de que se empiezan a perder paquetes, momento en el cual decrementa el caudal y vuelve a empezar con la secuencia de aumentarlo, un flujo TCP tendrá un cierto rizado que será muy apreciable sobre todo en el momento en el que interviene un flujo TCP y otro flujo distinto, produciéndose un transitorio que sí convendrá investigar, puesto que puede afectar negativamente a nuestro flujo UDP. Otra comparación que deberemos hacer será entre flujos UDP con y sin priorización, viendo qué efectos provoca el conformado de tráfico sobre ellos.

Los códecs que se van a simular en las pruebas pueden ser cualesquiera, la propia batería de pruebas nos permitirá establecer el ancho de banda utilizado tanto en audio como en vídeo. En principio asumiremos flujos agregados tanto de audio como de vídeo, es decir, simularemos varias conexiones a la vez mediante un solo flujo con un caudal igual al de la suma de las diversas conexiones a simular, esto tiene bastante sentido ya que priorizaremos de la misma forma todos los flujos con el mismo

TOS. Cada una de las pruebas tendrá una cierta duración, que también podremos especificar a la hora de realizarlas.

Las pruebas a realizar serán, por tanto, las siguientes:

1. Medida del ancho de banda total disponible en la red.

   Se medirá cuál es la capacidad de la red mediante el establecimiento de una conexión TCP durante un cierto tiempo, a efectos meramente referenciales. Lo ubicamos en su propio apartado aunque en el fondo esto se hará más adelante como parte de los dos siguientes apartados.

2. Tráfico de tiempo real entre los PCs 1 y 4. Se medirán el ancho de banda utilizado, los retardos, el jitter y los paquetes descartados en la red, en varias situaciones:

   - Tráfico sólo de datos (TCP).

     Flujo de datos agregado durante el tiempo estipulado. Nótese que de aquí en adelante, cuando hablemos de flujos de datos agregados, nos estaremos refiriendo a una conexión TCP que usaremos para simular varias, teniendo en cuenta que el algoritmo TCP proporciona el mismo ancho de banda a las distintas conexiones, nos referimos pues aquí a uno o varios flujos indistintamente, lo que nos interesará es ver su comportamiento frente a los flujos UDP.

   - Introducimos tráfico de voz.

     Al flujo de datos anterior lo perturbamos con un flujo de voz durante el tiempo estipulado.

   - Introducimos tráfico de vídeo.

     Hacemos lo propio con un flujo agregado de vídeo, debería respetar el audio dado que le vamos a dar menor prioridad.

   - Por último, introducimos tráfico UDP sin marcar.

     Igual que en el caso anterior pero introduciendo ahora tráfico UDP sin marcas de TOS, debería coger únicamente el ancho de banda que no utilicen los tráficos UDP anteriores.

   El tráfico TCP seguirá activo mientras los demás van haciendo aparición, veremos su evolución e interacción durante ese tiempo.

3. La prueba subjetiva consistirá en realizar las medidas en un escenario concreto: con la misma topología en cascada añadiendo a nuestra red de cuatro nodos un terminal en cada extremo, en nuestro caso utilizaremos teléfonos de VoIP mientras alteramos las propiedades de conformado de tráfico de los nodos intermedios. La medida consistirá en la percepción subjetiva de las personas que se encuentran en los extremos, tanto de la calidad de voz como del retardo de la misma y cualquier fenómeno que parezca significativo al oído humano.

## 4.2. Herramientas

Una vez definidas las pruebas que se van a realizar, es el momento de escoger las herramientas a utilizar para realizarlas.

En primer lugar necesitamos una herramienta (o herramientas) para inyectar tráfico en la red, que sea capaz de proporcionar distintas tasas de tráfico, variable o no, y que nos permita modelarlo para representar tanto el de tiempo real como el tráfico habitual, así como marcar los paquetes de forma adecuada. Quizá no baste con una sola herramienta sino que haga falta una combinación de ellas. Por otra parte necesitamos utilidades para medir los parámetros a considerar, a saber: retardo y jitter de los paquetes de tiempo real y número y tipo de paquetes descartados en la red.

Tras un análisis de herramientas de software libre dedicadas a este tipo de menesteres, hemos optado por una en concreto que nos proporciona casi todos estos datos: iperf (http://dast.nlanr.net/), que nos permite inyectar tráfico en la red y realiza las medidas pertinentes, incluyendo el jitter y el número total de descartes. Nos permite también modelar tráficos de distintos anchos de banda, de tipo TCP y UDP, y hacer pruebas durante un cierto tiempo o en función de un volumen de datos. Además, está disponible tanto para Linux como para Windows, lo que nos permite usarla en todas las pruebas, caso de utilizar un entorno Windows.

Para medir los retardos tendremos que valernos de otra herramienta, en este caso utilizaremos el programa echoping (http://echoping.sourceforge.net/), que permite medir los retardos en una red utilizando los puertos de echo y/o discard, servicios que tendremos que activar en nuestros nodos. Con él mediremos los retardos de paquetes de voz al transitar por la red.

En ambos casos los resultados de salida serán formateados con las herramientas

sed y awk para presentarlos en forma gráfica con el programa xgraph (http://www.isi.edu/nsnam/xgraph/index.html).

Para las pruebas contaremos con cuatro PCs portátiles, que aunque inicialmente iban a ser configurados con la distribución WISP-Dist (http://leaf.sourceforge.net/mod.php?mod=userpage&menu=908&page_id=27), inspirada en Debian, que proporciona soporte para interfaces inalámbricas, al final se ha decidido por una instalación normal de Debian GNU/Linux. Los motivos principales para descartar WISP-Dist han sido la relativa dificultad de actualización del kernel de esta distribución, así como la falta de soporte del kernel por defecto para las disciplinas de colas y tratamiento de la calidad de servicio que vamos a emplear. Tras la instalación de Debian GNU/Linux en los cuatro ordenadores, se añadirá a cada uno de ellos un script que permita configurar la disciplina de colas, por cada tipo de esquema de calidad de servicio que se pretende estudiar.

## 4.3. Implementación y automatización

Una vez escogidas las herramientas, las máquinas donde se van a efectuar las mismas y la batería de pruebas a realizar, es hora de enumerar minuciosamente todos los detalles:

### 4.3.1. Banco de pruebas

Dispondremos de un banco de pruebas realizado con portátiles conectados entre sí mediante tarjetas inalámbricas. A todos ellos se les aplicará un *script* para configurar la disciplina de colas, probaremos con distintas configuraciones para establecer una comparación que nos lleve a tomar una decisión razonable. Una de las cosas que se pretende conocer con la realización de estas pruebas es el diferente impacto que distintas configuraciones de conformado de tráfico van a tener sobre el resultado final, encontrando así la política que más se ajuste a lo que queremos encontrar. Sin embargo, en casi todos los casos, el kernel de cada máquina realizará clasificación del tráfico según su tipo de servicio IP dando prioridad máxima a los paquetes de audio, media a los de vídeo, y mínima al resto, si bien habrá variaciones en cada uno de estos casos.

Por otra parte, y para simplificar el trabajo, editaremos el fichero **/etc/hosts** de cada una de las máquinas para trabajar con nombres en vez de con direcciones

IP, considerando de nuevo que es lo suficientemente sencillo como para hacerlo a mano, sin la necesidad de utilizar un servidor de nombres:

```
127.0.0.1    localhost
10.0.0.1     nodo1
10.0.0.2     nodo2
10.0.0.3     nodo3
10.0.0.4     nodo4
```

Los cuatro ordenadores portátiles, a los cuales llamaremos nodoX (con X entre 1 y 4) sobre los que haremos las pruebas tienen las siguientes características:

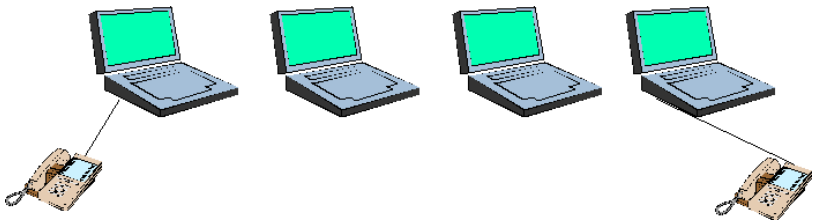
- *nodo1*: Procesador Pentium MMX 233 MHz con 112 MB de RAM. Una tarjeta de red inalámbrica con dirección 10.0.0.1, en la interfaz de red **wlan0**. Para las pruebas con teléfonos, una tarjeta de red ethernet con dirección 10.0.1.1, de nombre **eth0**.
- *nodo2*: Procesador Pentium MMX 233 MHz con 32 MB de RAM. Una tarjeta de red inalámbrica con dirección 10.0.0.2, en la interfaz de red **eth0**.
- *nodo3*: Procesador Pentium MMX 233 MHz con 48 MB de RAM. Una tarjeta de red inalámbrica con dirección 10.0.0.3, en la interfaz de red **wlan0**.
- *nodo4*: Procesador Pentium MMX 233 MHz con 32 MB de RAM. Una tarjeta de red inalámbrica con dirección 10.0.0.4, en la interfaz de red **wlan0**. Para las pruebas con teléfonos, una tarjeta de red ethernet con dirección 10.0.2.1, de nombre **eth0**.

Los cuatro tienen instalado Debian GNU/Linux con kernel de la serie 2.6, así como la versión 1.7.0 de iperf. La interfaz de red que configuraremos en cada uno será el dispositivo inalámbrico, cuya denominación depende del controlador utilizado, así, si usamos el controlador host_ap, los nombres de dispositivo serán del tipo **wlan0**.

Los teléfonos IP tendrán las direcciones 10.0.1.100 y 10.0.2.200 para el *teléfono1* y el *teléfono2*, respectivamente.

Los ordenadores quedarán conectados entre sí utilizando el modo *ad-hoc*, como pretendemos lograr la menor interferencia posible entre ellos y que cada nodo sólo vea al siguiente, se han colocado en línea, a distancias tales que su alcance sólo llegue a los adyacentes en la medida de lo posible, en las tarjetas con antena externa se ha suprimido ésta, para conseguir tal efecto. Los enlaces inalámbricos se fijarán a 1 Mb/s, estando la potencia recibida por los mismos en torno a los -78 dBm, esta información se obtiene mediante la herramienta de gestión inalámbrica *iwconfig*.

**Figura 4-1. Disposición del banco de pruebas**



#### 4.3.1.1. Banco inalámbrico básico

Los PCs pertenecerán a la red 10.0.0.0/24, y sus tablas de rutas quedarán configuradas de modo que cada nodo tenga su dirección IP y crearemos rutas a host, dado que la problemática del enrutamiento no es objeto ni parte de estas pruebas, sino que se estudia aparte. Por tanto tendremos para cada nodo las siguientes rutas, teniendo en cuenta que las configuraremos a partir de la configuración de red por defecto en cada una de las máquinas:

- nodo1: Todos sus paquetes se enviarán al nodo2. Para configurar este comportamiento:

```
#!/bin/bash

# rutas_nodo1.sh

ip route del 10.0.0.0/24
ip route add 10.0.0.2 dev wlan0
ip route add 10.0.0.0/24 via 10.0.0.2

# fin rutas_nodo1.sh
```

- nodo2: Si el destino es el nodo1, los paquetes se enviarán directamente a él, en cualquier otro caso se envían al nodo3:

```
#!/bin/bash

# rutas_nodo2.sh

ip route del 10.0.0.0/24
ip route add 10.0.0.1 dev eth0
ip route add 10.0.0.3 dev eth0
ip route add 10.0.0.0/24 via 10.0.0.3

# fin rutas_nodo2.sh
```

Es de notar que en el nodo2 la interfaz inalámbrica no es wlan0 sino eth0, esto es así porque en el resto de nodos se está utilizando el driver host_ap para manejar la tarjeta, no así en el nodo2, en el cual se usa el driver de orinoco. Esto se debe a que la tarjeta utilizada en nodo2 es incompatible con el driver host_ap.

- nodo3: Si el destino es el nodo4, los paquetes se enviarán directamente a él, en cualquier otro caso se envían al nodo2:

```
#!/bin/bash

# rutas_nodo3.sh

ip route del 10.0.0.0/24
ip route add 10.0.0.2 dev wlan0
ip route add 10.0.0.4 dev wlan0
ip route add 10.0.0.0/24 via 10.0.0.2

# fin rutas_nodo3.sh
```

- nodo4: Todos sus paquetes se enviarán al nodo3:

```
#!/bin/bash

# rutas_nodo4.sh

ip route del 10.0.0.0/24
ip route add 10.0.0.3 dev wlan0
ip route add 10.0.0.0/24 via 10.0.0.3

# fin rutas_nodo4.sh
```

#### 4.3.1.2. Banco inalámbrico con teléfonos VoIP

La idea aquí será el tener dos redes adicionales: la 10.0.1.0/24, a la que pertenece uno de los teléfonos IP, y la red 10.0.2.0/24, a la que pertenece el otro. Los ordenadores nodo1 y nodo4 dispondrán de sendas tarjetas ethernet que los conectarán directamente con las redes mencionadas, respectivamente, lo que añadirá rutas por defecto a las mismas. Para ello tendremos que modificar las tablas de rutas de los nodos, de la siguiente forma:

- nodo1: Todos sus paquetes se enviarán al nodo2, excepto los que vayan al teléfono1, que está en la red 10.0.1.0/24, directamente conectada a nodo1 por su interfaz eth0. Para configurar este comportamiento:

```
#!/bin/bash

# wrutas_nodo1.sh
# no hace falta añadir las rutas al teléfono IP porque está directamente
# conectado a la misma red que eth0, automáticamente al levantar la interfaz
# tendremos la conectividad asegurada

ip route del 10.0.0.0/24
ip route add 10.0.0.2 dev wlan0
ip route add 10.0.0.0/24 via 10.0.0.2
ip route add 10.0.2.0/24 via 10.0.0.2

# fin wrutas_nodo1.sh
```

- nodo2: Si el destino es el nodo1 o el teléfono1, los paquetes se enviarán directamente a nodo1, en cualquier otro caso se envían al nodo3:

```
#!/bin/bash

# wrutas_nodo2.sh

ip route del 10.0.0.0/24
ip route add 10.0.0.1 dev eth0
ip route add 10.0.0.3 dev eth0
ip route add 10.0.0.0/24 via 10.0.0.3
ip route add 10.0.1.0/24 via 10.0.0.1
ip route add 10.0.2.0/24 via 10.0.0.3

# fin wrutas_nodo2.sh
```

- nodo3: Si el destino es el nodo4 o el teléfono2, los paquetes se enviarán directamente a nodo4, en cualquier otro caso se envían al nodo2:

```
#!/bin/bash

# wrutas_nodo3.sh

ip route del 10.0.0.0/24
ip route add 10.0.0.2 dev wlan0
ip route add 10.0.0.4 dev wlan0
ip route add 10.0.0.0/24 via 10.0.0.2
ip route add 10.0.1.0/24 via 10.0.0.2
ip route add 10.0.2.0/24 via 10.0.0.4

# fin wrutas_nodo3.sh
```

- nodo4: Todos sus paquetes se enviarán al nodo3, excepto los que vayan al teléfono2, que está en la red 10.0.2.0/24, directamente conectada a nodo4 por su interfaz eth0:

```
#!/bin/bash
```

```
# wrutas_nodo4.sh
# no hace falta añadir las rutas al teléfono IP porque está directamente
# conectado a la misma red que eth0, automáticamente al levantar la interfaz
# tendremos la conectividad asegurada

ip route del 10.0.0.0/24
ip route add 10.0.0.3 dev wlan0
ip route add 10.0.0.0/24 via 10.0.0.3
ip route add 10.0.1.0/24 via 10.0.0.3

# fin wrutas_nodo4.sh
```

## 4.3.2. Pruebas con iperf

A continuación se detallan las pruebas exactas a realizar sobre el banco inalámbrico, tanto objetivas como subjetivas.

### 4.3.2.1. Pruebas objetivas

1. Medida del ancho de banda total disponible en la red.

   Se medirá cuál es la capacidad de la red mediante el establecimiento de una conexión TCP durante un cierto tiempo.

2. Tráfico de tiempo real entre los PCs 1 y 4. Se medirán los retardos, el jitter y los paquetes descartados en la red, en varias situaciones:

   - Tráfico sólo de datos (TCP).

     Flujo de datos agregado durante el tiempo estipulado.

   - Introducimos tráfico de voz.

     Al flujo de datos anterior lo perturbamos con un flujo de voz de un cierto ancho de banda durante el tiempo estipulado.

   - Introducimos tráfico de vídeo.

     Hacemos lo propio con un flujo agregado de vídeo, debería respetar el audio dado que le vamos a dar menor prioridad.

   - Por último, introducimos tráfico UDP sin marcar.

Igual que en el caso anterior pero introduciendo ahora tráfico UDP sin marcas de TOS, debería coger únicamente el ancho de banda que no utilicen los tráficos UDP anteriores.

El tráfico TCP seguirá activo mientras los demás van haciendo aparición, veremos su evolución e interacción durante ese tiempo. A continuación vemos el script de pruebas, detallando cada una de sus partes:

```
#!/bin/bash

# bateria_pruebas.sh
# Parámetros usados por el script:

# Nodos origen y destino de los datos
ORIGEN=$1
DESTINO=$2
# Anchos de banda equivalentes de los flujos
BW_AUDIO=$3
BW_VIDEO=$4
BW_RUIDO=$5
# Longitud media de los paquetes de los flujos
LEN_AUDIO=$6
LEN_VIDEO=$7
LEN_RUIDO=$8
# Directorio donde se guardarán los logs
DIRECTORIO=$9
```

Se puede observar que toma nueve parámetros, que podemos ir variando en función del efecto que deseemos conseguir: ORIGEN y DESTINO son los nodos donde se ejecutará iperf, como cliente y como servidor, respectivamente. Las variables BW_AUDIO, BW_VIDEO y BW_RUIDO almacenan los valores, en kilobits por segundo, del ancho de banda que simularemos para voz, vídeo y tráfico genérico UDP. Sus correspondientes LEN_AUDIO, LEN_VIDEO y LEN_RUIDO servirán para parametrizar los tamaños de los paquetes, basándonos en tamaños medios para cada caso. Por último DIRECTORIO es la variable que indica el directorio donde guardaremos el registro de las pruebas.

```
# Lanzamos los servidores
ssh $DESTINO "iperf -s -i 0.5" > /tmp/prueba-servidor-tcp.log &
ssh $DESTINO "iperf -s -u -i 0.5 -p 50001" > /tmp/prueba-servidor-udp-voz.log &
ssh $DESTINO "iperf -s -u -i 0.5 -p 50005" > /tmp/prueba-servidor-udp-video.log &
ssh $DESTINO "iperf -s -u -i 0.5 -p 50009" > /tmp/prueba-servidor-udp.log &
sleep 10
```

Establecemos los servidores en el nodo DESTINO, tendremos un servidor para tráfico TCP, y sendos servidores para voz, vídeo y tráfico UDP genérico,

escuchando en distintos puertos. Esto nos servirá para extraer de los registros los datos de cada flujo agregado independientemente de los demás.

```
# Conectamos cada 20 segundos un flujo distinto
echoping -n 20 -u -d $DESTINO > /tmp/prueba-latencia.log &
iperf -c $DESTINO -t 80 -i 0.5 > /tmp/prueba-cliente-tcp.log &
sleep 20

echoping -n 20 -u -d $DESTINO >> /tmp/prueba-latencia.log &
iperf -c $DESTINO -t 60 -i 0.5 -u -S 0xa0 -b $BW_AUDIO -l $LEN_AUDIO -p 50001 \
     > /tmp/prueba-cliente-voz.log &
sleep 20

echoping -n 20 -u -d $DESTINO >> /tmp/prueba-latencia.log &
iperf -c $DESTINO -t 40 -i 0.5 -u -S 0x60 -b $BW_VIDEO -l $LEN_VIDEO -p 50005 \
     > /tmp/prueba-cliente-video.log &
sleep 20

echoping -n 20 -u -d $DESTINO >> /tmp/prueba-latencia.log &
iperf -c $DESTINO -t 20 -i 0.5 -u -b $BW_RUIDO -l $LEN_RUIDO -p 50009 \
     > /tmp/prueba-cliente-udp.log &
sleep 20
```

Ahora lanzamos una conexión TCP, y cada 20 segundos añadimos un nuevo flujo: en el instante 20 entrará el flujo agregado de voz, en el instante 40, el de vídeo, y en el 60, el de tráfico UDP. Mientras tanto con echoping vamos midiendo la latencia que tiene un paquete UDP a lo largo de todo el proceso.

La idea es poder apreciar el efecto que la entrada de cada nuevo tipo de tráfico tiene sobre los anteriores, es decir, el efecto transitorio, para luego observar lo que ocurre en la situación estable. Esto tiene especial relevancia en el caso de la interacción TCP con UDP, para analizar cómo se afectan mutuamente.

```
# Eliminamos los servidores tras un tiempo de guarda
sleep 40
killall iperf
ssh $DESTINO "killall iperf"

# Recolectamos los resultados
mkdir $DIRECTORIO
mv /tmp/prueba-* $DIRECTORIO

# fin bateria_pruebas.sh
```

Por último eliminamos los servidores tras esperar un cierto tiempo para no interferir con las medidas, y guardamos los registros en el directorio escogido.

3. Estas pruebas se realizarán en diferentes casos de conformado de tráfico, desde utilizando exclusivamente la disciplina PRIO hasta usando DSMARK de la forma más eficiente.

#### 4.3.2.2. Pruebas subjetivas

La prueba subjetiva consistirá en realizar las medidas en un escenario concreto: con la misma topología en cascada añadiendo a nuestra red de cuatro nodos un terminal en cada extremo, en nuestro caso utilizaremos teléfonos de VoIP mientras alteramos las propiedades de conformado de tráfico de los nodos intermedios. La medida consistirá en la percepción subjetiva de las personas que se encuentran en los extremos, tanto de la calidad de voz como del retardo de la misma y cualquier fenómeno que parezca significativo al oído humano.

### 4.3.3. Casos de estudio

Con objeto de simplificar el estudio, nos vamos a centrar, tras un estudio preliminar, en ciertos casos de conformado de tráfico que pasaremos a analizar en profundidad al obtener los resultados. Los vamos a detallar aquí, así como los scripts que utilizaremos finalmente, nos quedaremos con los que proporcionen resultados más significativos.

Para ello previamente vamos a describir, brevemente, como funciona el procesado de paquetes en el núcleo de Linux. Por defecto, en Linux sólo hay una cola, de tipo pfifo_fast, donde se sirve a los paquetes a medida que llegan, clasificándolos en tres bandas en función de su campo TOS, de la forma tradicional en la que se clasifican los paquetes (marcados para mínimo retardo, mínimo coste, máxima fiabilidad, etc.). Nosotros vamos a redefinir la política de colas del núcleo de distintas formas, para analizar cada caso y ver cuál nos conviene más. Para ello, primero veamos una sucinta explicación de los términos que vamos a emplear:

En la parte de procesado de tráfico de Linux podemos tener una disciplina de cola (abreviadamente, qdisc), que puede ser "con clases" (classful, en inglés) o "sin clases" (classless). ¿Qué significa cada término? Una disciplina classful puede contener clases, que a su vez podemos asociar a otras disciplinas de colas, con o sin clases, todas las veces que queramos, de forma que podemos crear jerarquías de disciplinas de colas en las cuales tendremos, por ejemplo, una disciplina raíz con

tres clases en cada una de las cuales hay una cola, que puede ser con clases o sin ellas, si la primera es con clases puede tener a su vez dos clases con dos colas, etc. Evidentemente, una disciplina sin clases es aquella que no permite una mayor progresión en el árbol jerárquico, no podemos introducir nuevas colas dentro de ella. La disciplina pfifo_fast es un ejemplo de disciplina sin clases. Por otra parte necesitamos indicar de alguna forma a qué cola se ha de enviar cada paquete, para ello a cada disciplina le podremos asociar uno o más filtros que, en función de la cabecera o los contenidos del paquete, lo enviarán a la cola escogida. A continuación veamos un breve resumen de las disciplinas de colas que trataremos:

- pfifo_fast: como ya hemos dicho, es la disciplina por defecto, los paquetes se procesan en orden de llegada, y se pueden clasificar en una de tres bandas posibles en función del tipo de servicio de la cabecera IP. No se envían paquetes de una banda hasta que la(s) que tienen mayor prioridad está(n) vacía(s).
- PRIO: es una disciplina con clases, muy similar a la anterior, por defecto crea tres clases a las que podremos asociar sus correspondientes colas, la diferencia es que mediante el uso de filtros decidiremos a qué cola enviamos cada paquete, con lo cual las posibilidades de clasificación son mucho más amplias.
- HTB: la disciplina Hierarchical Token Bucket nos permite establecer proporciones de tráfico en cada una de las clases que asociemos a ella, indicaremos cuánto tráfico cursará cada una y la disciplina HTB se encargará de mantener la proporción, que será un caudal garantizado. Además tenemos la posibilidad de "tomar prestado" el caudal sobrante o incluso el de otras clases que no lo estén aprovechando en ése momento.
- DSMARK: En realidad, DSMARK no es exactamente una herramienta de conformado de tráfico, sino que es más bien una herramienta de clasificación, en función del campo DS de la cabecera IP. Crearemos una serie de clases, a las cuales se enviarán los paquetes tras examinar su cabecera IP, en cada clase tendremos asociada la disciplina correspondiente que se encargará de procesar los paquetes.
- SFQ: La disciplina Stochastic Fair Queueing, que es una disciplina sin clases, se encarga de dar a los diferentes flujos TCP o UDP un trato justo y equitativo en el uso del ancho de banda de la red. Lo utilizamos como ejemplo, realmente, dado que en las pruebas hemos escogido utilizar tráficos agregados y en realidad todos los paquetes de una categoría pertenecen al mismo flujo, si pertenecieran a

flujos distintos SFQ se encargaría de que todos recibieran un trato equitativo. Por supuesto sólo tiene sentido hablar de esto cuando el interfaz está lleno y hay una cola de espera.

- RED: La disciplina de detección aleatoria anticipada se encarga de prevenir la congestión mediante la previsión de congestiones, cuando parece que la red va a empezar a congestionarse, se encarga de marcar paquetes para su descarte y de esta forma avisar al emisor para que reduzca su tasa de envío, mucho antes de que se congestione la cola.

Esto es sólo una breve descripción de las técnicas que emplearemos, para una explicación más concienzuda y como referencia a más bibliografía sobre el tema, consultar el Linux Advanced Routing & Traffic Control HOWTO (http://lartc.org/howto/) y el Differentiated Service on Linux HOWTO (http://www.opalsoft.net/qos/DS.htm).

Veamos ahora nuestros casos de estudio:

1. Sin conformado de tráfico:

   El caso más evidente, y sobre el cual compararemos a los demás, es el caso en que no tocamos para nada el procesado de paquetes en cola y observamos lo que ocurre.

2. Con disciplina de colas PRIO:

   La disciplina de colas prio nos ofrece tres clases de diferente prioridad, a las cuales asociaremos las colas que deseemos. Sólo cuando no haya ningún paquete en una cola de mayor prioridad podremos transmitir el paquete de la cola actual.

```
#!/bin/bash

# colas_prio.sh

# El parámetro es el interfaz a configurar
IFACE=$1

tc qdisc add dev $IFACE handle 1:0 root dsmark indices 64 set_tc_index default_index 1
tc filter add dev $IFACE parent 1:0 protocol ip prio 4 tcindex mask 0xfc shift 2

tc qdisc add dev $IFACE handle 2: parent 1:1 prio

tc qdisc add dev $IFACE parent 2:1 sfq perturb 10
tc qdisc add dev $IFACE parent 2:2 sfq perturb 10
tc qdisc add dev $IFACE parent 2:3 red limit 64KB min 5KB max 15KB burst 8 avpkt 1000 \
     bandwidth 64Kbit probability 0.1
```

```
tc filter add dev $IFACE parent 2: protocol ip prio 1 handle 0x28 tcindex classid 2:1 \
     pass_on
tc filter add dev $IFACE parent 2: protocol ip prio 2 handle 0x18 tcindex classid 2:2 \
     pass_on
tc filter add dev $IFACE parent 2: protocol ip prio 3 u32 match ip tos 0x00 0x00 \
     flowid 2:3

# fin colas_prio.sh
```

En nuestro caso hemos realizado una pequeña combinación con la ayuda de la disciplina DSMARK, con objeto de mostrar la versatilidad del sistema. Tenemos pues una disciplina DSMARK, con 64 clases posibles, a una de ellas le asociamos la disciplina PRIO, con tres disciplinas de colas por defecto, en las dos primeras, las de mayor prioridad, asociamos sendas disciplinas de tipo SFQ, aquí será donde enviaremos el audio y el vídeo, a la tercera le asociaremos una RED, principalmente porque a ella irá a parar todo el tráfico TCP.

3. Con disciplina de colas HTB priorizando UDP:

   HTB es una disciplina jerárquica que mantiene las proporciones especificadas en cada una de sus ramas. Así le damos 3 Mbps al audio, 2 Mbps al vídeo, y 1 kbps al resto de tráfico (que puede tomar prestado del ancho de banda libre hasta 4 Mbps). Si la capacidad de nuestros enlaces es menor, cosa que ocurrirá, los anchos de banda asignados seguirán manteniendo las proporciones.

```
#!/bin/bash

# colas_htb_udp.sh

# El parámetro es el interfaz a configurar
IFACE=$1

tc qdisc add dev $IFACE root handle 1: htb default 30
tc class add dev $IFACE parent 1: classid 1:1 htb rate 4Mbit burst 15k

tc class add dev $IFACE parent 1:1 classid 1:10 htb rate 3Mbit burst 15k
tc class add dev $IFACE parent 1:1 classid 1:20 htb rate 2Mbit ceil 4mbit burst 15k
tc class add dev $IFACE parent 1:1 classid 1:30 htb rate 1kbit ceil 4mbit burst 15k

tc qdisc add dev $IFACE parent 1:10 handle 10: sfq perturb 10
tc qdisc add dev $IFACE parent 1:20 handle 20: sfq perturb 10
tc qdisc add dev $IFACE parent 1:30 handle 30: sfq perturb 10

U32="tc filter add dev $IFACE protocol ip parent 1:0 prio 1 u32"
$U32 match ip protocol 17 0xff flowid 1:10
$U32 match ip protocol 6 0xff flowid 1:20
```

```
# fin colas_htb_udp.sh
```

Vemos pues que disponemos de una disciplina HTB con tres clases, a las cuales asociaremos sendas colas de tipo SFQ. Cabe destacar que en este caso enviaremos todo el tráfico UDP a la cola con 3 Mbps, el resto del tráfico, es decir, el tráfico TCP, irá a la cola de 2 Mbps. No hacemos distinción entre audio y vídeo, aunque dejamos las colas preparadas para el siguiente caso, que diferirá únicamente en los filtros.

4. Con disciplina de colas HTB priorizando la voz:

   La diferencia ahora con el caso anterior estriba en que, en vez de separar el tráfico UDP del TCP, ahora a cada una de las clases con disciplina SFQ enviaremos el audio, el vídeo, y el resto del tráfico, es decir, separamos los distintos flujos UDP.

```
#!/bin/bash

# colas_htb_audio.sh

# El parámetro es el interfaz a configurar
IFACE=$1

tc qdisc add dev $IFACE root handle 1: htb default 30
tc class add dev $IFACE parent 1: classid 1:1 htb rate 4Mbit burst 15k

tc class add dev $IFACE parent 1:1 classid 1:10 htb rate 3Mbit burst 15k
tc class add dev $IFACE parent 1:1 classid 1:20 htb rate 2Mbit ceil 4mbit burst 15k
tc class add dev $IFACE parent 1:1 classid 1:30 htb rate 1kbit ceil 4mbit burst 15k

tc qdisc add dev $IFACE parent 1:10 handle 10: sfq perturb 10
tc qdisc add dev $IFACE parent 1:20 handle 20: sfq perturb 10
tc qdisc add dev $IFACE parent 1:30 handle 30: sfq perturb 10

U32="tc filter add dev $IFACE protocol ip parent 1:0 prio 1 u32"
$U32 match ip tos 0xa0 0xff flowid 1:10
$U32 match ip protocol 6 0xff flowid 1:20

# fin colas_htb_audio.sh
```

5. Con disciplina DSMARK con ancho de banda compartido:

   En este caso, permitiremos que los distintos tráficos tomen el ancho de banda sobrante de los demás.

```
#!/bin/bash

# colas_dsmark_compartidas.sh

# El parámetro es el interfaz a configurar
```

```
IFACE=$1

tc qdisc add dev $IFACE handle 1:0 root dsmark indices 64 set_tc_index default_index 1
tc filter add dev $IFACE parent 1:0 protocol ip prio 4 tcindex mask 0xfc shift 2
tc qdisc add dev $IFACE handle 2: parent 1:1 htb

tc class add dev $IFACE parent 2: classid 2:1 htb rate 500Kbit ceil 550Kbit
tc class add dev $IFACE parent 2:1 classid 2:10 htb rate 300Kbit ceil 500Kbit prio 1
tc class add dev $IFACE parent 2:1 classid 2:20 htb rate 100Kbit ceil 200Kbit prio 2
tc class add dev $IFACE parent 2:1 classid 2:40 htb rate 300Kbit ceil 500Kbit prio 3

tc qdisc add dev $IFACE parent 2:10 sfq perturb 10
tc qdisc add dev $IFACE parent 2:20 sfq perturb 10
tc qdisc add dev $IFACE parent 2:40 red limit 500KB min 5KB max 15KB burst 8 avpkt 1000 \
     bandwidth 300Kbit probability 0.1

tc filter add dev $IFACE parent 2: protocol ip prio 1 handle 0x28 tcindex \
     classid 2:10 pass_on
tc filter add dev $IFACE parent 2: protocol ip prio 1 handle 0x18 tcindex \
     classid 2:20 pass_on
tc filter add dev $IFACE parent 2: protocol ip prio 2 u32 match ip tos 0x00 0x00 \
     flowid 2:40

# fin colas_dsmark_compartidas.sh
```

Crearemos una disciplina DSMARK, dentro de la cual anidaremos una disciplina HTB con, de nuevo, tres subclases, en este caso las tres serán de tipo HTB, y cada una de ellas tendrá una sola subclase, las dos primeras con disciplina SFQ, y la última con RED, como vimos anteriormente. La diferencia principal con el caso de uso directo de HTB consistirá en los filtros, antes filtrábamos en función de parámetros de la cabecera IP, ahora filtraremos exclusivamente según el manejador asociado en DSMARK al campo DS de la cabecera IP.

6. Con disciplina DSMARK sin compartir ancho de banda:

   Ahora haremos algo muy parecido al caso anterior, con la diferencia de que ahora el único tipo de tráfico que puede tomar caudal prestado del resto es el de voz.

```
#!/bin/bash

# colas_dsmark_aisladas.sh

# El parámetro es el interfaz a configurar
IFACE=$1

tc qdisc add dev $IFACE handle 1:0 root dsmark indices 64 set_tc_index default_index 1
tc filter add dev $IFACE parent 1:0 protocol ip prio 4 tcindex mask 0xfc shift 2
tc qdisc add dev $IFACE handle 2: parent 1:1 htb
```

```
tc class add dev $IFACE parent 2: classid 2:1 htb rate 500Kbit ceil 550Kbit
tc class add dev $IFACE parent 2:1 classid 2:10 htb rate 300Kbit ceil 500Kbit prio 1
tc class add dev $IFACE parent 2:1 classid 2:20 htb rate 150Kbit prio 2
tc class add dev $IFACE parent 2:1 classid 2:40 htb rate 64Kbit prio 3

tc qdisc add dev $IFACE parent 2:10 sfq perturb 10
tc qdisc add dev $IFACE parent 2:20 sfq perturb 10
tc qdisc add dev $IFACE parent 2:40 red limit 64KB min 5KB max 15KB burst 8 avpkt 1000 \
     bandwidth 64Kbit probability 0.1

tc filter add dev $IFACE parent 2: protocol ip prio 1 handle 0x28 tcindex \
     classid 2:10 pass_on
tc filter add dev $IFACE parent 2: protocol ip prio 1 handle 0x18 tcindex \
     classid 2:20 pass_on
tc filter add dev $IFACE parent 2: protocol ip prio 2 u32 match ip tos 0x00 0x00 \
     flowid 2:40

# fin colas_dsmark_aisladas.sh
```

## 4.4. Resultados

Una vez realizadas las pruebas, con diversos tipos de conformado de tráfico y parámetros, vamos a tomar los resultados más significativos de los esquemas utilizados, analizándolos y viendo cuáles de ellos son los idóneos para dar prioridad a la voz.

### 4.4.1. Pruebas objetivas

Antes de empezar a hacer las pruebas, la primera medida consiste en ver cuál es el ancho de banda disponible en la red. Para ello, y teniendo en cuenta que los extremos de la red serán nodo1 y nodo4, con la ayuda de iperf vamos a medir el ancho de banda utilizado durante 10 segundos. Establecemos el servidor en nodo4:

```
nodo4:~# iperf -s
```

Realizamos la conexión desde nodo1, midiendo en intervalos de 1 segundo:

```
nodo1:~# iperf -c nodo4 -i 1
```

Obteniendo un resultado como éste:

```
------------------------------------------------------------
```

```
Client connecting to nodo4, TCP port 5001
TCP window size: 16.0 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.1 port 1029 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  72.0 KBytes   590 Kbits/sec
[  3]  1.0- 2.0 sec  72.0 KBytes   590 Kbits/sec
[  3]  2.0- 3.0 sec  64.0 KBytes   524 Kbits/sec
[  3]  3.0- 4.0 sec  64.0 KBytes   524 Kbits/sec
[  3]  4.0- 5.0 sec  56.0 KBytes   459 Kbits/sec
[  3]  5.0- 6.0 sec  80.0 KBytes   655 Kbits/sec
[  3]  6.0- 7.0 sec  48.0 KBytes   393 Kbits/sec
[  3]  7.0- 8.0 sec  72.0 KBytes   590 Kbits/sec
[  3]  8.0- 9.0 sec  72.0 KBytes   590 Kbits/sec
[  3]  9.0-10.0 sec  72.0 KBytes   590 Kbits/sec
[  3]  0.0-10.5 sec   680 KBytes   532 Kbits/sec
```

De donde se puede observar que el ancho de banda medio en el intervalo medido es de 532 Kbps, evidentemente, la medida varía cada vez, pero se mantiene en el rango observado en el ejemplo. A efectos prácticos, consideramos que la red nos proporciona un ancho de banda de 530-550 Kbps para realizar nuestras estimaciones, y así parece confirmarse empíricamente. Por otra parte observamos también que los enlaces, presumiblemente al tener poca potencia, no permiten tener un ancho de banda estable, sino que éste fluctúa ligeramente a lo largo del tiempo.

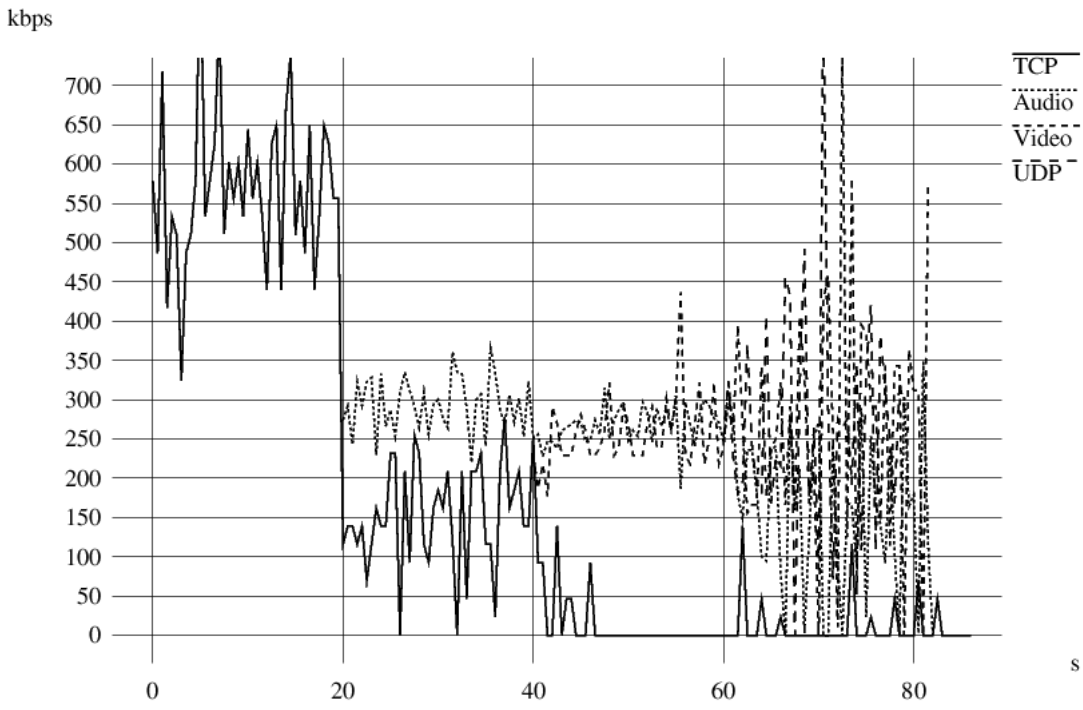
Se decide entonces realizar una serie de pruebas con las distintas disciplinas de colas, con diferentes índices de ocupación de la red, para ello el script de pruebas permite especificar el ancho de banda deseado para cada flujo UDP. Lo primero que observamos, como ya se ha comentado, es que el flujo TCP se adapta a la capacidad que queda libre de la red, con algún que otro pico de ancho de banda en los transitorios, que perturban los flujos UDP, su efecto se hace notar en las pruebas subjetivas. Por lo demás, podemos prescindir de él a efectos de ir aumentando progresivamente el caudal. En cuanto al resto de los tráficos, hasta que no alcanzan alrededor de los 550 Kbps agregados, no se nota demasiado el efecto del conformado, excepto sobre todo con HTB que se encarga de asignar proporciones entre caudales, lo comentaremos en su momento. Lo que a nosotros nos interesa realmente es ver el efecto que se produce en una red realmente saturada, y realizando progresivas pruebas, se decide tomar un ancho de banda de 300 Kbps para cada uno de los tres flujos UDP, suficiente para saturar la red y mostrar los efectos de las disciplinas empleadas sobre los distintos tipos de tráfico en condiciones adversas, pero sin llegar a extremos que no se darán en la práctica (tómese como idea, por ejemplo, el hecho de usar un códec tipo Speex, que puede usar tasas tan bajas como 5 Kbps, esto implicaría que tendríamos en la red 50

personas realizando una videoconferencia simultáneamente, situación bastante absurda).

#### 4.4.1.1. Prueba sin conformado de tráfico

Principalmente a efectos de referencia, la primera prueba realizada consiste en, sin tocar en absoluto la configuración de colas del núcleo de Linux, ver qué ocurre de forma natural con los distintos tráficos de nuestra prueba. Para ello, nada mejor que una imagen de lo que ocurre a lo largo del tiempo con los anchos de banda de los tráficos involucrados, representada en la figura.

**Figura. Ancho de banda de los distintos flujos, sin conformado de tráfico**

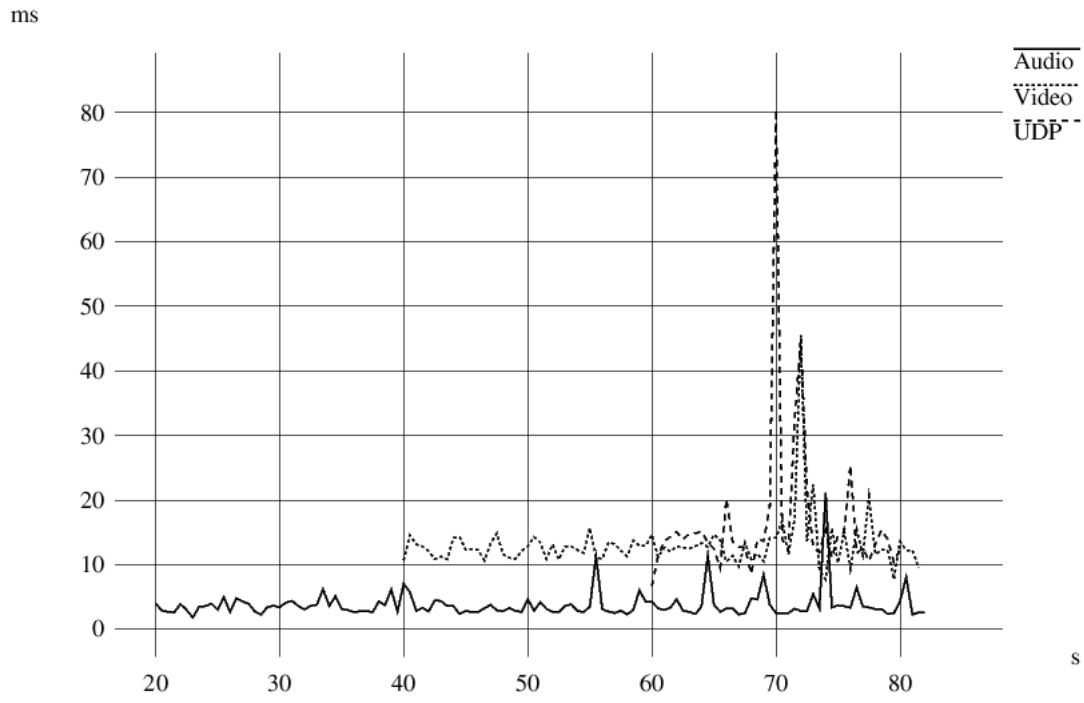



Como se puede apreciar en la figura, durante los primeros veinte segundos tenemos un flujo TCP con una media de unos 550-560 kbps, capacidad máxima de nuestra red inalámbrica. En el instante 20 entra el flujo de voz, a 300 kbps, realmente no se produce contienda entre ambos flujos; puesto que el TCP se adapta a la capacidad máxima que le ofrece el canal, el flujo de voz toma el caudal que le corresponde y

relega al TCP a un segundo plano, eso sí, el TCP periódicamente intenta coger más ancho de banda perturbando la transmisión de voz, aunque donde realmente se ve perturbada es frente a otros flujos UDP. A los 40 segundos entra el flujo de vídeo, que modifica a la baja el ancho de banda otorgado al de voz, se observa que ambos están por debajo de los 300 kbps, recordemos que la red no es capaz de otorgarnos tanta capacidad. Por otra parte, el flujo TCP se ve casi totalmente anulado frente a los dos caudales UDP, que además ven fuertemente afectada su estabilidad de uso de ancho de banda. Por último irrumpe un nuevo flujo UDP con un ancho de banda igual al de los anteriores, en total tenemos 900 kbps (esto sin contar con el caudal TCP) intentando propagarse por la red, el resultado es una gran inestabilidad en los tres flujos, con pérdidas de paquetes en la red y desórdenes en las llegadas de los mismos.

En cuanto al jitter, veamos lo que ocurre con los flujos de voz, vídeo y UDP:

**Figura 4-3. Jitter de los distintos flujos, sin conformado de tráfico**



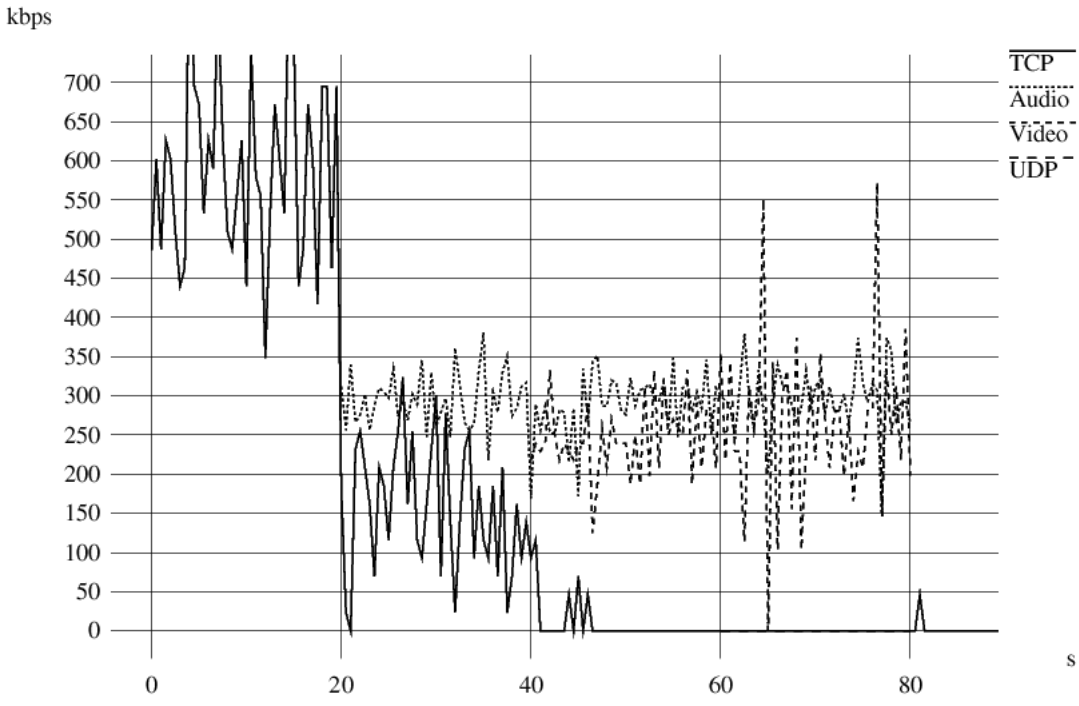

Curiosamente, el jitter de audio en condiciones normales permanece bastante continuo, incluso con la presencia de los otros dos tipos de tráfico UDP. Aprovechemos para recordar que el jitter es la variación del tiempo que transcurre entre la llegada de dos paquetes consecutivos con respecto a la situación ideal, así, cuanto más continuo y bajo sea, mejor. Observamos también que la introducción del

flujo de vídeo conlleva un jitter mayor y una pequeña perturbación de el de voz, pero no muy significativa, sin embargo, el flujo UDP sin marcar desestabiliza en mayor medida la voz, y en gran medida el vídeo, además de presentar un jitter bastante irregular. Esto se debe seguramente a los tamaños de paquetes que estamos manejando, los de voz son sensiblemente más pequeños que el resto, que serán más proclives a sufrir alto jitter al necesitarse más tiempo para su transmisión por una red que va paulatinamente saturándose.

#### 4.4.1.2. Prueba con disciplina de colas PRIO

Veamos que ocurre ahora al usar la disciplina PRIO, que se encarga sencillamente de crear tres clases, con tres niveles de prioridad, la voz siempre tendrá prioridad sobre el vídeo que a su vez la tendrá sobre los datos.

**Figura 4-4. Ancho de banda de los distintos flujos, con disciplina PRIO**

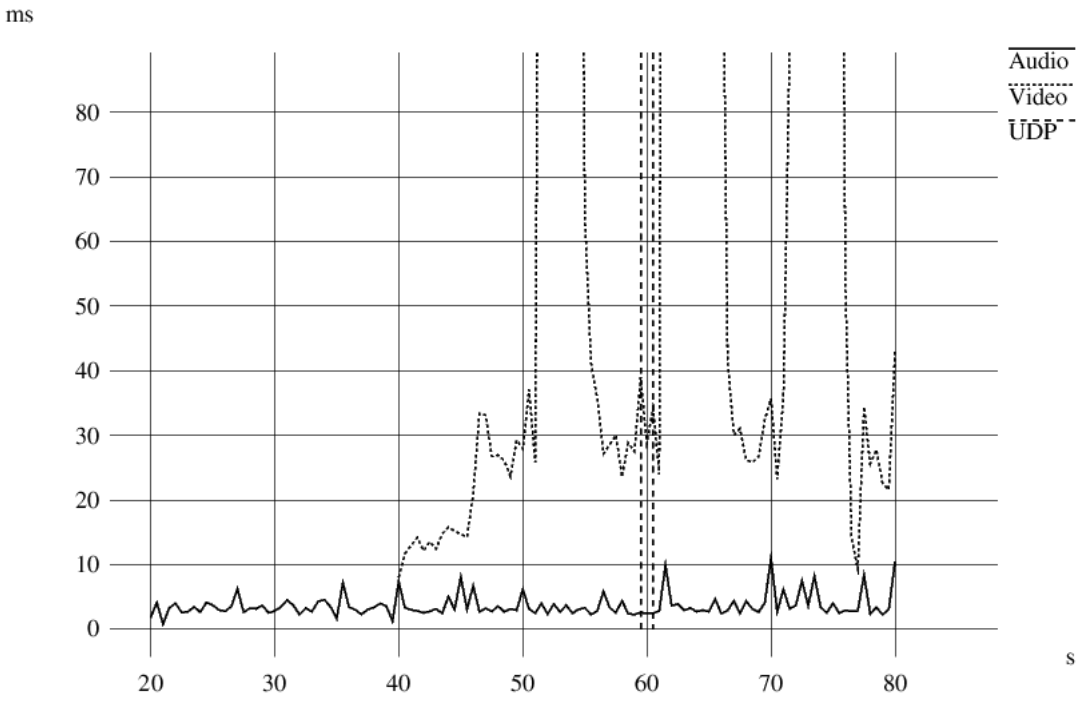



Como se puede observar el arranque es similar al del caso anterior, evidentemente, dado que hasta que no se sature la red no empezaremos a ver realmente los efectos de la disciplina escogida. Al entrar el audio el comportamiento es muy similar, de

nuevo, nos encontramos ante el caso en que el flujo TCP sucumbe ante el de voz UDP. Con el vídeo obtenemos el mismo comportamiento que antes, ambos flujos sumados son 600 kbps, luego se producen ciertas pérdidas de paquetes, de forma similar al caso anterior, teniendo en cuenta que se transmite todo el flujo de voz que tiene prioridad frente al vídeo. La diferencia principal con que nos encontramos es que en este caso el tráfico TCP se ve anulado por completo, y al introducir el flujo UDP sin marcar, dado que va a la tercera clase de tráfico, no encuentra hueco por donde enviarse, dado que todo el ancho de banda es usado por las otras dos clases, resultando en la anulación de este tipo de tráfico.

El jitter, por otra parte, presenta este aspecto:

**Figura 4-5. Jitter de los distintos flujos, con disciplina PRIO**



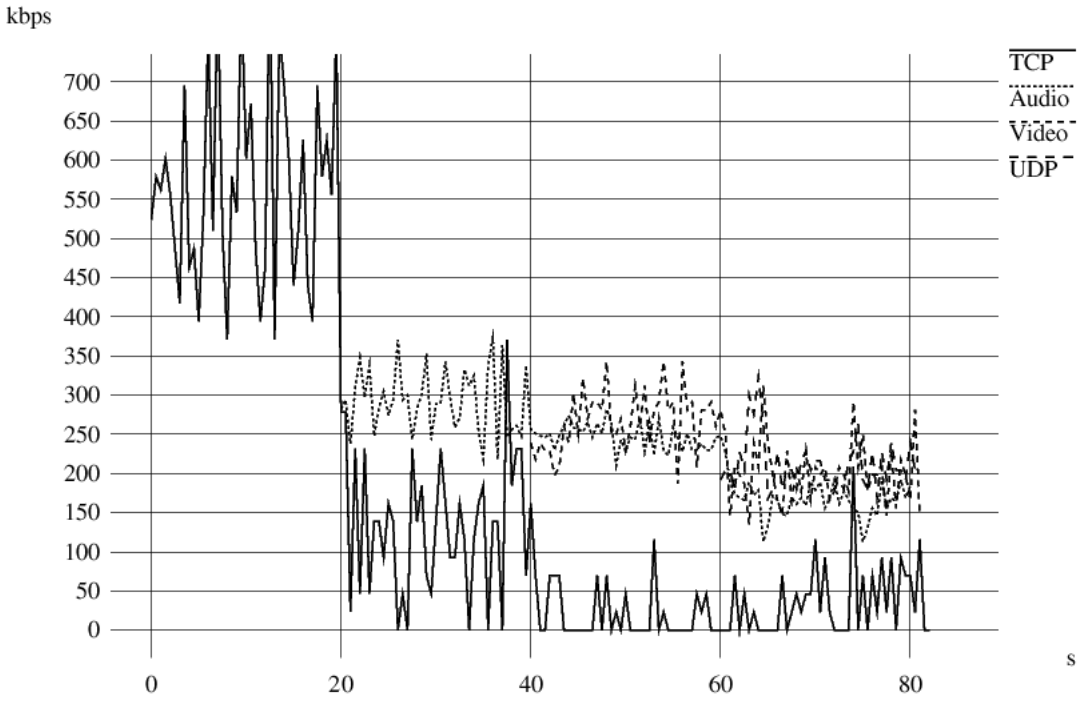

Se ve claramente que el esquema PRIO respeta totalmente las prioridades establecidas, y el resultado en el jitter es el que cabía esperar: poquísimo jitter para el flujo de voz, un jitter mayor para el de vídeo, incluso mayor que en el caso anterior dado que ahora priorizamos el audio frente al vídeo, y en el caso del flujo sin marcar, se observa que consigue introducir algunos paquetes cuando entra en escena, y es relegado como ya decíamos a un segundo plano, su jitter es , por tanto, ridículamente alto, por encima de un segundo, pero no es significativo, ya que no se

esperaba que este flujo tuviera relevancia en este contexto, al estar a merced del uso del ancho de banda de los otros, más prioritarios.

### 4.4.1.3. Prueba con disciplina de colas HTB priorizando UDP

Utilizando ahora la disciplina HTB, vamos a dar prioridad sencillamente al tráfico UDP sobre el TCP, sin diferenciar entre distintos tipos de tráfico UDP. Como se ve en la figura, los resultados se empiezan a aproximar más al efecto deseado:

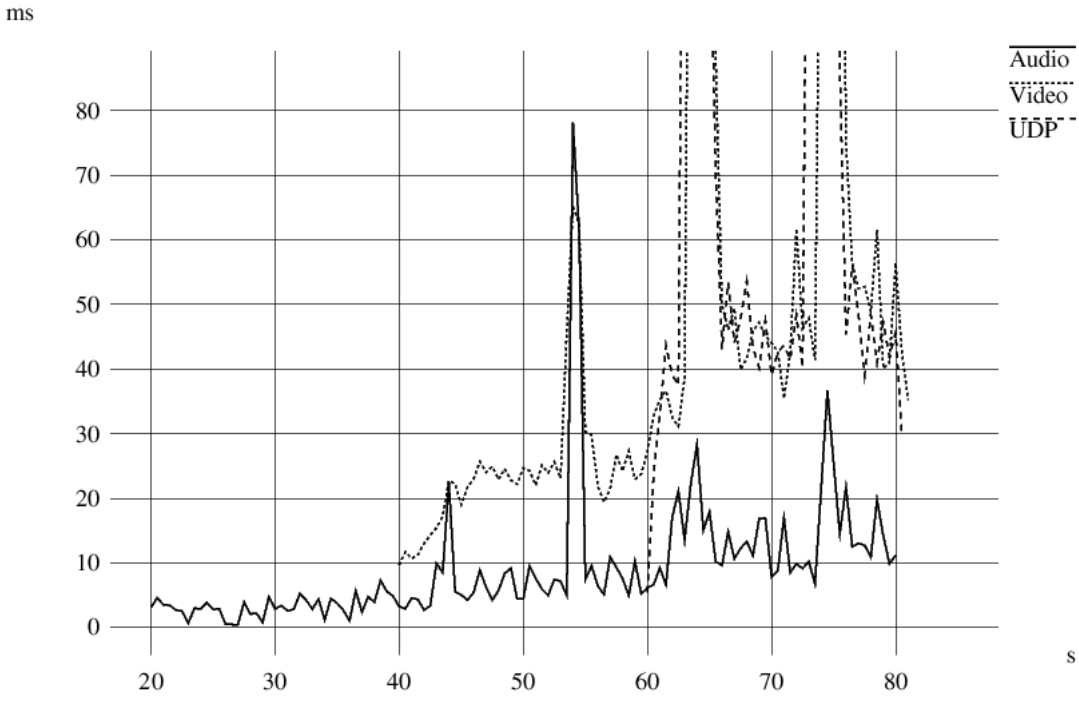
**Figura 4-6. Ancho de banda de los distintos flujos, con HTB priorizando el tráfico UDP**




En este caso cabe resaltar por un lado la mayor estabilidad del tráfico UDP, en líneas generales, y por otro el respeto a un mínimo flujo TCP, incluso en condiciones de alta carga. El ancho de banda se reparte entre los diferentes flujos UDP, que van usando menos ancho de banda a medida que entran nuevos contendientes, pero manteniendo en todo caso el equilibrio entre los tres. Vemos que hemos llegado a un esquema en

el que todo el tráfico es cursado, no como en los casos anteriores en los que algún tipo de tráfico perdía totalmente la posibilidad de ser enviado.

**Figura 4-7. Jitter de los distintos flujos, con HTB priorizando el tráfico UDP**

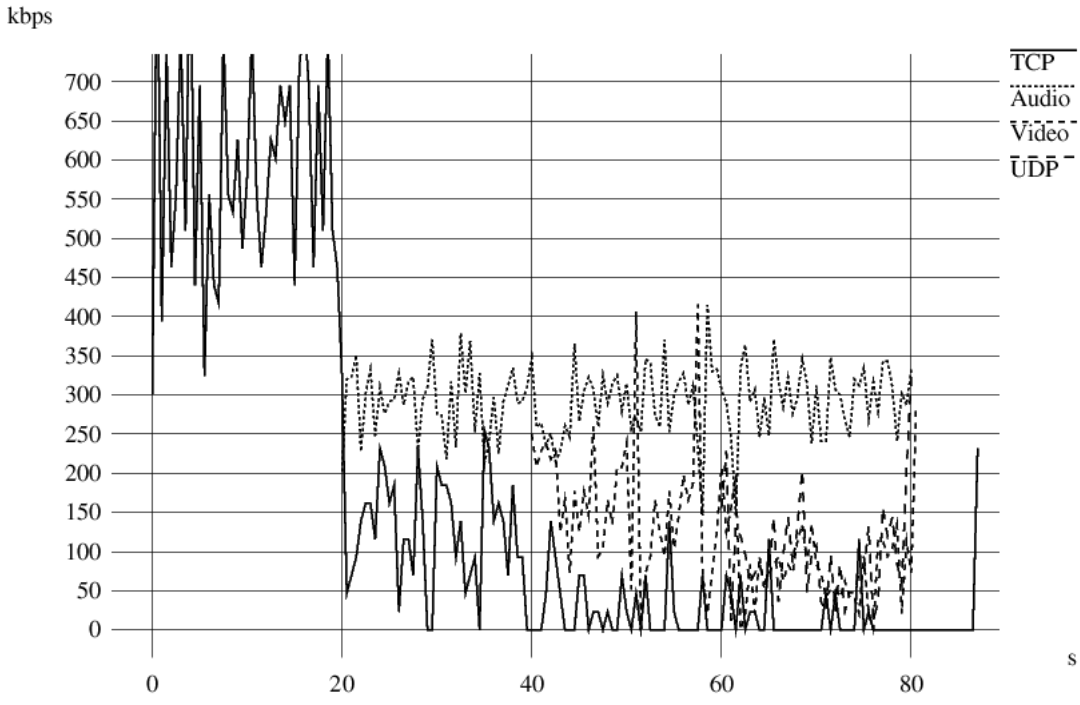



En cuanto al jitter, dado que no estamos priorizando los flujos UDP entre ellos, sino sólo frente al resto de tráficos, se produce de nuevo el fenómeno de que al ir introduciendo nuevo tráfico UDP el jitter va creciendo, de nuevo en función del tamaño de paquete. Curiosamente, en este caso el jitter es mucho mayor que sin priorizar tráfico, pero es que hay que tener en cuenta que, a diferencia de la situación anterior, ahora existe además un tráfico TCP que interfiere con el UDP, de forma que los paquetes se intercalan con los de TCP y aumenta la variación entre sus llegadas. De nuevo hay menos jitter con los paquetes de audio dado su reducido tamaño.

#### 4.4.1.4. Prueba con disciplina de colas HTB priorizando la voz

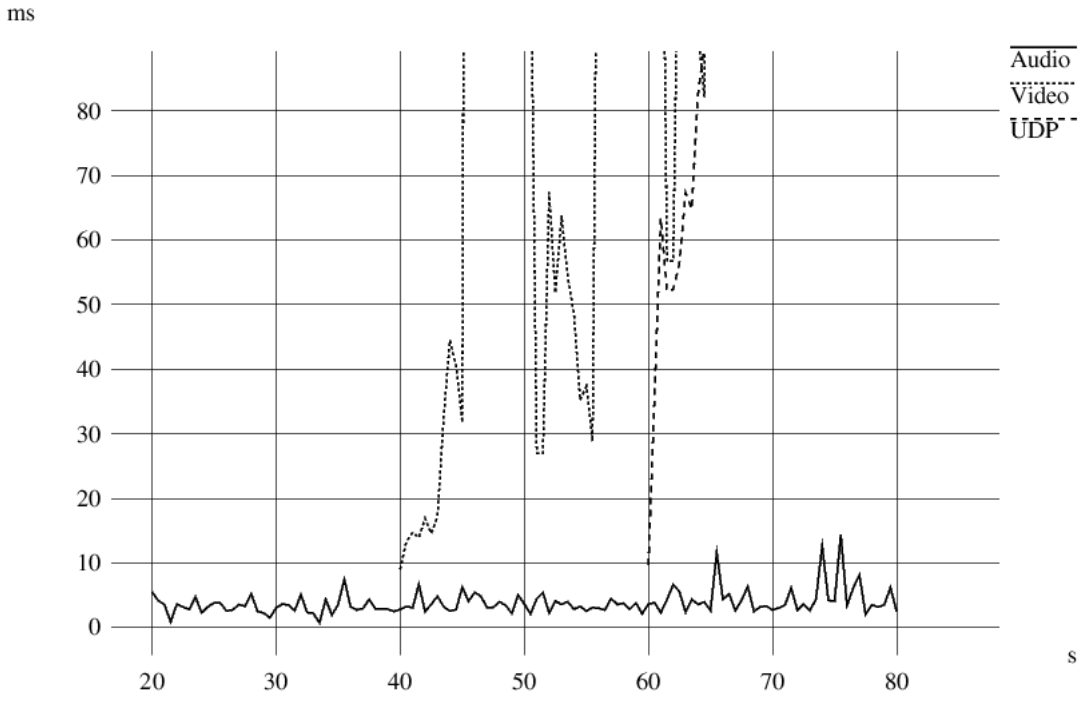
Ahora se dará mayor prioridad al tráfico de voz frente al resto, observemos los resultados:

**Figura 4-8. Ancho de banda de los distintos flujos, con HTB priorizando la voz**




Se aprecia claramente que la disciplina HTB nos permite dar a la voz una supremacía sobre el resto de contendientes, vemos que sin embargo el resto de tráficos (vídeo y UDP sin marcar) siguen siendo muy irregulares, más incluso que en el caso anterior. Podríamos dar diferentes prioridades y mejorar la calidad del servicio de vídeo, pero la siguiente disciplina, DSMARK, será mucho más potente, como ahora comprobaremos.

**Figura 4-9. Jitter de los distintos flujos, con HTB priorizando la voz**



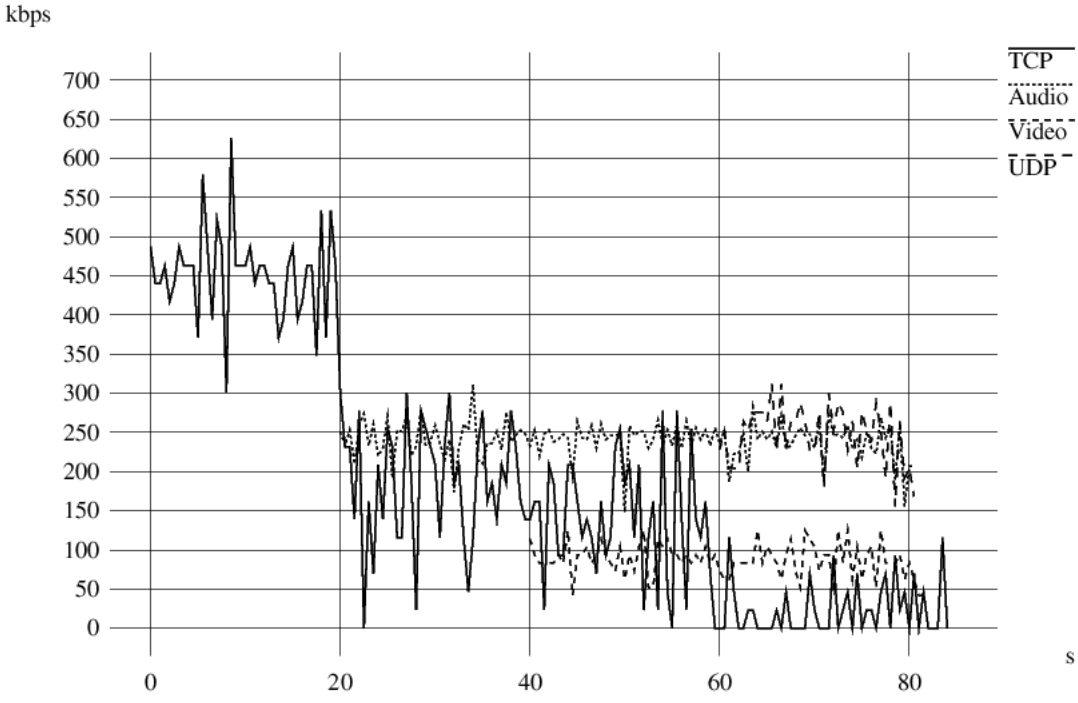

En este caso, el jitter de voz es muy regular y bajo, que es exactamente lo que estamos buscando. Sin embargo, si hablamos de los jitter de vídeo y UDP sin marcar, vemos que aumentan una barbaridad, porque ahora no tienen ningún tipo de prioridad frente al resto del tráfico, más bien al contrario. De hecho llegan a valores altos como un segundo y medio, o incluso dos en el caso del tráfico sin marcar.

### 4.4.1.5. Prueba con disciplina DSMARK con ancho de banda compartido

La disciplina más eficaz a la hora de separar los distintos flujos es la DSMARK, como queda claro en la siguiente imagen:

**Figura 4-10. Ancho de banda de los distintos flujos, con DSMARK**

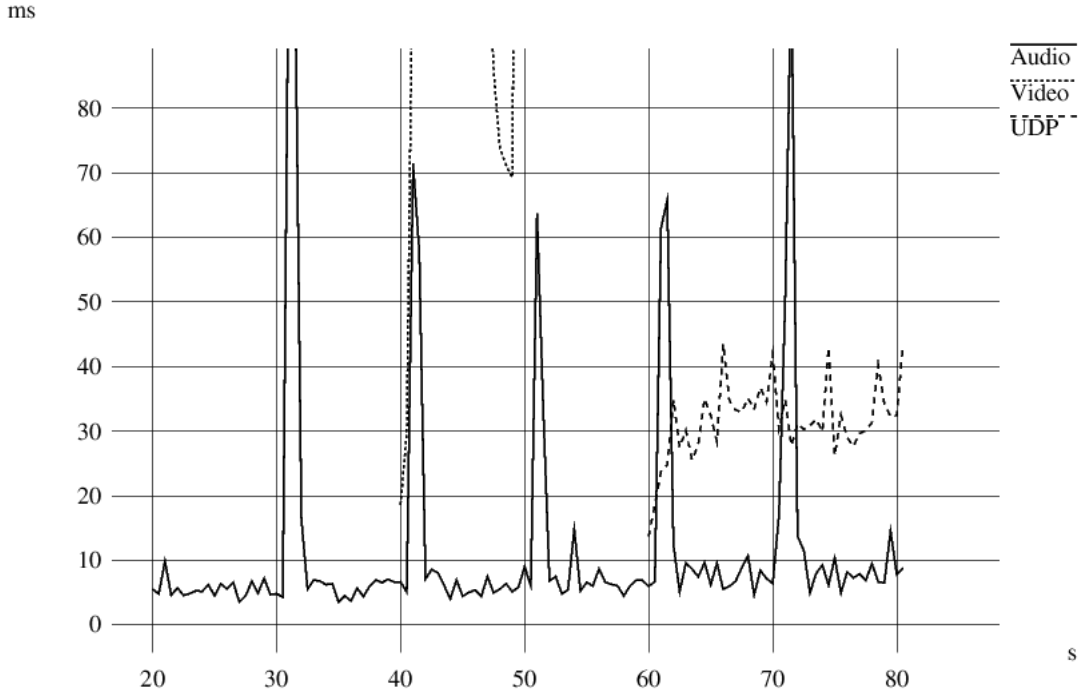
**compartiendo ancho de banda**




Comprobamos aquí como DSMARK con un filtrado adecuado es capaz de separar mucho mejor los tráficos, manteniendo una estabilidad envidiable. Cabe destacar que en este ejemplo los distintos flujos toman el ancho de banda que les sobra a los demás, de ahí que por ejemplo el flujo UDP alcance la misma capacidad que el de audio, entre otras cosas porque se ha dotado al tráfico sin marcar de una capacidad equivalente a la de la voz. El tráfico de vídeo, sin embargo, está dotado de una capacidad mucho menor, lo que se refleja a continuación al examinar el jitter.

**Figura 4-11. Jitter de los distintos flujos, con DSMARK compartiendo ancho de**

**banda**



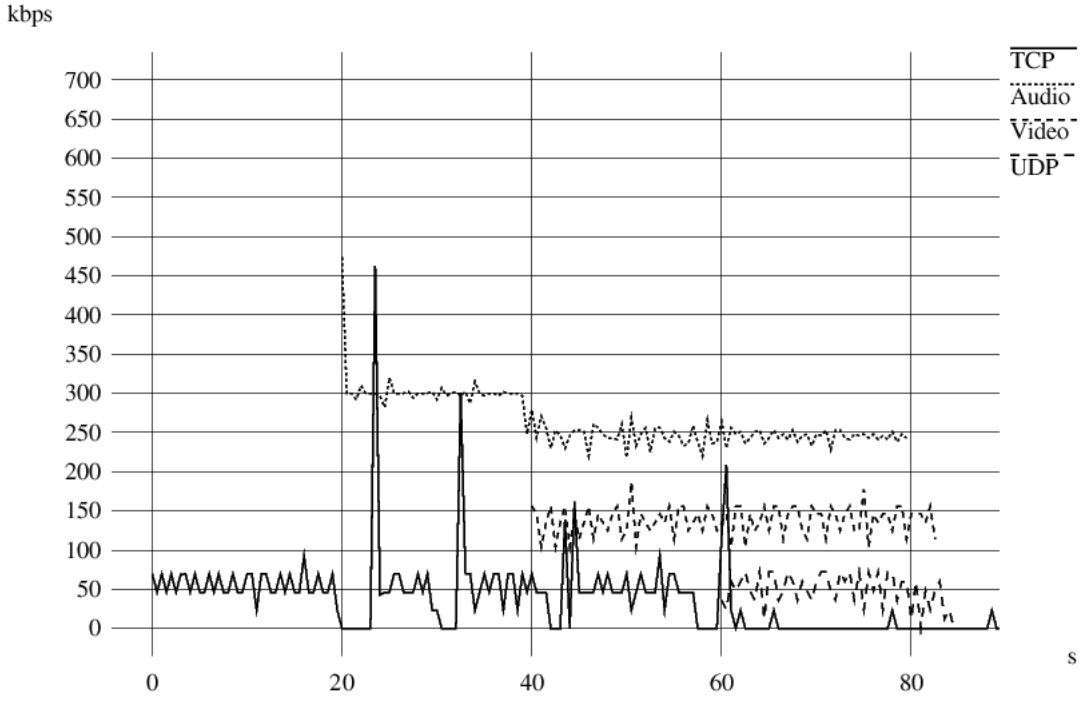

Vemos aquí que el jitter de voz ha aumentado con respecto a los casos anteriores, aunque los aumentos son en forma de picos, el jitter se mantiene el resto del tiempo muy estable. Veremos cómo afecta esto a las pruebas subjetivas. El jitter de vídeo crece por encima de lo representado aquí, principalmente por el poco ancho de banda que tiene el vídeo con respecto al resto de tráfico UDP. Se comprueba empíricamente, pues, que un mayor ancho de banda nos proporciona un mejor jitter para nuestra señal.

### 4.4.1.6. Prueba con disciplina DSMARK sin compartir ancho de banda

Como colofón de esta serie de pruebas presentamos el último esquema, quizá no tan flexible como el anterior, pero increíblemente efectivo a la hora de separar anchos de banda, como se puede apreciar:

**Figura 4-12. Ancho de banda de los distintos flujos, con DSMARK sin**

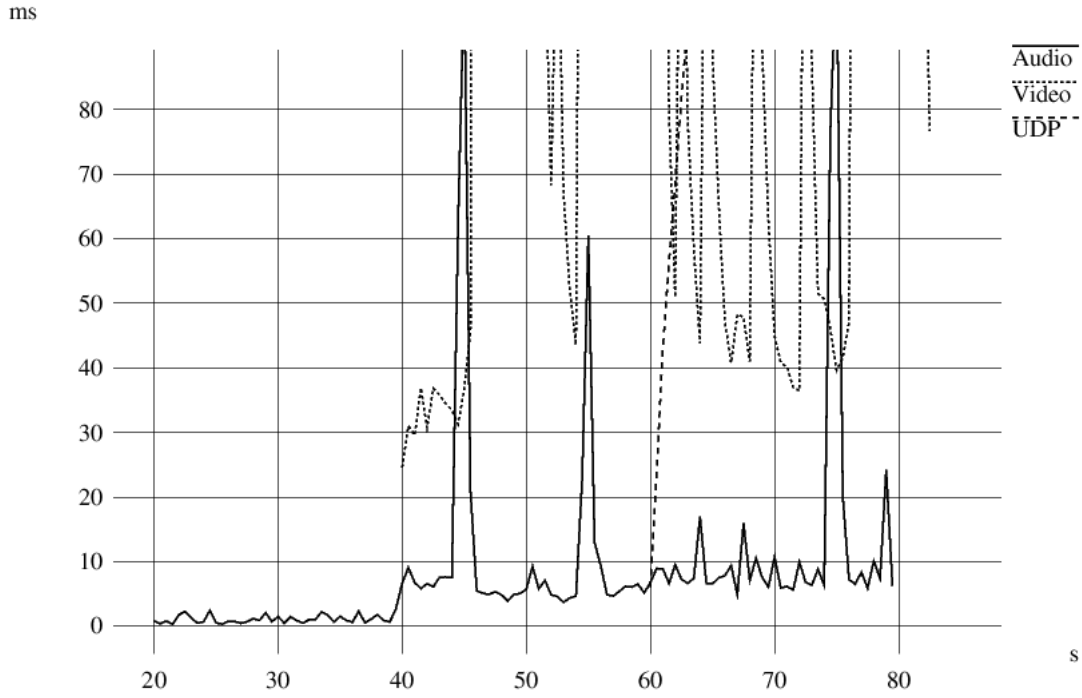
**compartir ancho de banda**




En este caso se ha optado por dotar de muy poco ancho de banda al tráfico sin marcar, que no puede tomar prestado del ancho de banda libre, y la mayor parte de la capacidad va destinada a los caudales de voz y vídeo. De esta forma, sacrificando algo de flexibilidad se puede garantizar prácticamente la disponibilidad del canal para la voz, objetivo último de esta serie de pruebas.

**Figura 4-13. Jitter de los distintos flujos, con DSMARK sin compartir ancho de**

**banda**




La voz empieza con un jitter despreciable, al introducir el vídeo, que no puede tomar ancho de banda prestado, vemos que su jitter es relativamente alto, y empiezan a aparecer picos en el jitter de voz, a la vez que éste aumenta. El resto de tráfico UDP presenta un jitter también relativamente alto, aunque sube un poco, no pasa de los 200 milisegundos, lo cual sigue sin ser mucho. En cualquier caso son resultados de jitter mucho mejores, en general, de los obtenidos hasta ahora.

## 4.4.2. Pruebas subjetivas

Una vez realizadas las pruebas anteriores, parece evidente que el mejor esquema, al menos desde un punto de vista teórico y objetivo, es el que utiliza la disciplina DSMARK, con las colas HTB repartiendo el tráfico de forma proporcional, por lo tanto, éstos son los esquemas que emplearemos para realizar las pruebas subjetivas, viendo primero lo que ocurre en situaciones de alta carga y saturación de la red sin emplear ningún tipo de clasificado o conformado de tráfico, y luego empleando la disciplina DSMARK para la clasificación y la HTB para el conformado de tráfico.

Veamos pues los puntos relevantes observados en las pruebas subjetivas en ambos

casos:

1. Prueba sin conformado de tráfico:
   - La calidad de voz en condiciones normales es buena.
   - Al introducir un flujo TCP mientras se habla, se aprecian perturbaciones en la línea en forma de *clicks* auditivos.
   - Al introducir flujos UDP la calidad permanece estable hasta que el caudal UDP sin contar con nuestra conversación se acerca a los 400 Kbps, momento a partir del cual la calidad de nuestra señal de voz se degrada muy significativamente.
   - En condiciones de alta carga, el hecho de medir los retardos de paquetes de datos con echoping hace que aparezcan también perturbaciones en forma de *clicks*, además de serios retardos, de entre 1 hasta 10 segundos, en los paquetes de datos.
   - En condiciones de alta carga se produce también un retardo apreciable en la voz, de unos 3-4 segundos.

2. Prueba con disciplina DSMARK con ancho de banda compartido:
   - La calidad de voz en condiciones normales es buena.
   - Al introducir un flujo TCP mientras se habla, no se aprecia ninguna perturbación en la línea.
   - Al introducir flujos UDP la calidad permanece estable, incluso con un flujo de 600 Kbps de UDP sin marcar, no se aprecia efecto en la conversación.
   - En condiciones de alta carga, el hecho de medir los retardos de paquetes de datos con echoping no produce perturbación alguna, los retardos en los paquetes de datos aumentan significativamente, incluso hasta los 22-23 segundos.
   - Se produce el mismo retardo en la voz en condiciones de alta carga.

3. Prueba con disciplina DSMARK sin compartir ancho de banda:

- La calidad de voz en condiciones normales es buena.
- Al introducir un flujo TCP mientras se habla, no se aprecia ninguna perturbación en la línea.
- Al introducir flujos UDP la calidad permanece estable, incluso con un flujo de 600 Kbps de UDP sin marcar, no se aprecia efecto en la conversación.
- En condiciones de alta carga, el hecho de medir los retardos de paquetes de datos con echoping no produce perturbación alguna, los retardos en los paquetes de datos aumentan significativamente, incluso hasta los 25 segundos.
- Se produce el mismo retardo en la voz en condiciones de alta carga.

Se ha podido apreciar, pues, que el hecho de introducir una política de colas cuidadosa hacia la voz mejora la calidad subjetiva de la conversación de voz. Cabe resaltar dos detalles; el primero, que en situaciones de alta carga el retardo de la voz es el mismo en ambos casos, tanto con conformado como sin él, cuando no debería, en principio, ser apreciable en los casos en los que garantizamos el caudal para la voz. En efecto, lo que ocurre es que los teléfonos IP realizan un buffering intensivo con objeto de sacrificar la inmediatez en aras de la calidad. La otra cosa a tener en cuenta es que, como se puede apreciar de los datos recopliados en las pruebas, no existe apenas diferencia entre permitir o no a los datos que no tienen la prioridad el tomar prestado del ancho de banda total, en todo caso se logra mejorar un poco, de 2 a 3 segundos, el retardo de los datos, lo cual no es particularmente impresionante, teniendo en cuenta que hablamos de un retardo de un orden de magnitud mayor.

## 4.5. Conclusiones

Tras realizar las pruebas, tanto objetivas como subjetivas, se llega a la conclusión de que el conformado de tráfico es la herramienta casi idónea para la priorización de la voz y el vídeo, si logramos controlar dos detalles:

- Que todos los nodos de la red apliquen la política de conformado de tráfico.
- Que las aplicaciones o teléfonos IP permitan marcar el campo de tipo se servicio de las cabeceras IP a voluntad.

Las pruebas realizadas no representan, ni mucho menos, la totalidad de las posibilidades de conformado de tráfico en Linux, y existe, como casi siempre, la posibilidad de lograr los mismos objetivos de forma diferente, por ejemplo, mediante el uso de una disciplina CBQ podríamos haber llegado a resultados muy similares a los ofrecidos con HTB. Estas pruebas representan más bien un muestrario y un análisis somero de las posibilidades del núcleo de Linux en cuanto al conformado de tráfico y más concretamente al tratamiento prioritario del tráfico de tiempo real. A este respecto cabe mencionar que, además de haber preseleccionado las disciplinas a utilizar y haber escogido sólo algunas, en un futuro aparecerán otras nuevas que pueden ser más apropiadas, tal es el caso de la disciplina CSZ (Clark-Shenker-Zang) que se encarga de priorizar el tráfico en tiempo real, pero cuyo soporte está actualmente en un estado inestable sin garantías de funcionamiento correcto.

Queda claro, pues, que la disciplina DSMARK permite un clasificado correcto del tráfico, compatible además con la política de servicios diferenciados definida en la RFC correspondiente, que permitiría incluso a la salida de nuestra red que se mantuviera una clasificación coherente de los paquetes enviados. Dentro de ello, el uso de la disciplina HTB para el conformado de tráfico resulta más que adecuado, obteniéndose muy buenos resultados sobre todo cuando se permite al tráfico de voz tomar prestado ancho de banda en detrimento de tráficos menos urgentes, incluso a costa de una ligera infrautilización de las capacidades del canal, en algunos casos.

Además, los resultados teóricos se ven avalados por las pruebas subjetivas, obteniéndose una calidad muy aceptable en las conversaciones de voz que no se ve perturbada por el resto de los tráficos, al contrario de lo que ocurría en una red sin conformado.

En cuanto al resto de parámetros, el jitter aumenta en ciertas circunstancias, pero no llega a ser significativo en casi ningún caso, con las contadas excepciones de cuando se dispara, que coincide con que el flujo correspondiente tiene muy baja o nula prioridad, lo cual es perfectamente coherente. Lo mismo ocurre con los retardos, el tráfico priorizado, evidentemente, presenta muy poco retardo en todos los casos, mientras que el que vamos dejando como secundario llega a presentar retardos muy fuertes con pérdidas de paquetes.

# Capítulo 5. Diseño

Se explican a continuación las decisiones de diseño tomadas hasta ahora, sin perjuicio de poder, a la hora de realizar la implementación efectiva, modificar estas ideas preliminares en función de la necesidad o conveniencia.

## 5.1. Los nodos

### 5.1.1. Hardware

Cada nodo de una red ad-hoc como la que se pretende desarrollar consta de todos los elementos para funcionar de manera autónoma:

1. Un ordenador empotrado de potencia suficiente para las tareas de encaminamiento, con algún slot de expansión en el que se pueda incorporar una tarjeta WiFi, el menor consumo posible, que tenga hardware de tipo *watchdog* para asegurar el re-arranque automático si el sistema deja de responder, que se pueda alimentar mediante tensión continua entre 11V y 13V para que pueda aceptar directamente la alimentación del subsistema solar, y que todo el material esté soportado por Linux. El almacenamiento del sistema será en algún tipo de dispositivo de bajo consumo como Compact Flash o flash rom.

2. Una distribución de Linux apropiada, que quepa en el espacio disponible, que sea fácil de administrar a distancia a través de la red, que cuente con todas las utilidades necesarias para la gestión avanzada del direccionamiento IP, el encaminamiento, el control de la congestión y el soporte de calidad de servicio, y que soporte correctamente todo el material. Un buen ejemplo de ésto sería Debian GNU/Linux, o alguna distribución empotrada basada en ella.

3. Una o varias radios WiFi con el máximo de potencia posible (200mW) y conectores para antenas externas, soportadas por Linux y a ser posible tanto en modo ad-hoc como en modos cliente y punto de acceso.

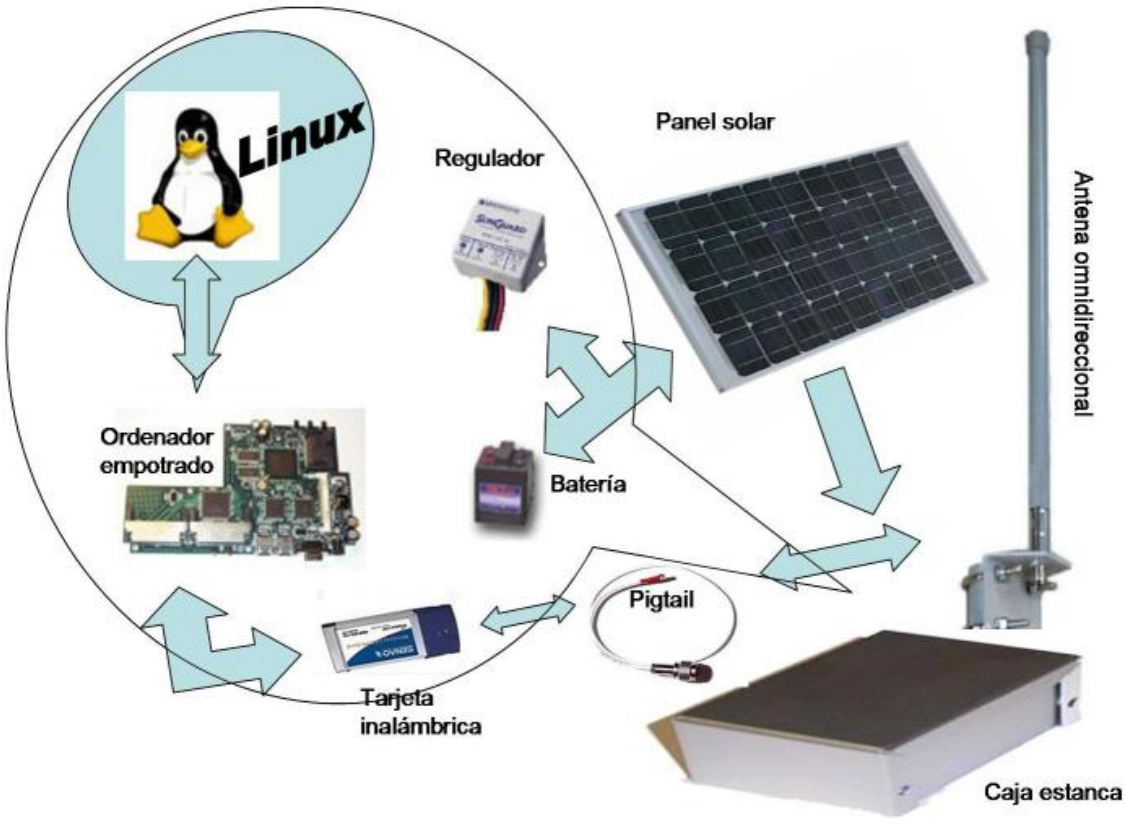
4. Un subsistema solar, constituido por panel(es), regulador, batería(s) y accesorios como el cableado, los conectores y otros elementos mecánicos de soporte y fijación.

5. Antenas externas y *pigtails* apropiados para conectarlas a las radios. A priori todo nodo tendrá al menos una antena omnidireccional, pero se podría considerar la posibilidad de poner antenas directivas si una determinada radio se instala para conectar con dispositivos en una dirección determinada.

6. Caja estanca ventilada para albergar todos los elementos, con puntos de fijación en la parte superior para los paneles solares y en los laterales para las antenas y para su fijación a un mástil o a un muro. El diseño de la caja deberá ser tal que el nodo una vez ensamblado sea una unidad robusta y fácil de transportar.

Se van a producir dos versiones de los equipos, una para pruebas, más versátil y con más facilidades para el desarrollo y la depuración, y una segunda para producción en la que se ha minimizado el consumo y se ha dimensionado el subsistema de alimentación para asegurar el funcionamiento continuo bajo condiciones de exposición solar típicas de zonas tropicales. La Figura 5-1 muestra los elementos de un nodo.

**Figura 5-1. Elementos de un nodo**



#### 5.1.1.1. Versión de pruebas

La primera versión se basa en un ordenador empotrado de arquitectura x86, con el que resulta fácil trabajar y que soporta directamente ciertas distribuciones software para encaminadores inalámbricos como son WISP-Dist y ROSE. El empotrado empleado es el Soekris-net4521, por sus características de bajo consumo (<7W), amplio rango de voltaje de alimentación aceptado, que permite adaptar directamente el subsistema solar, y slots PCMCIA y CF para insertar radios y almacenamiento masivo. A este ordenador se ha adaptado una radio Engenius Senao SL-2511-CD-Ext2, que ofrece numerosas ventajas sobre las demás: está soportada por el driver host_ap, que permite ponerla como punto de acceso si es necesario, tienen dos conectores MMCX para adaptar antenas externas y transmiten hasta 200mW, que es el máximo disponible en el mercado sin amplificadores. El sistema se completa con pigtails, antenas omnidireccionales de 12dBi, antenas direccionales de rejilla de 24dBi, y paneles solares, reguladores y baterías de Isofotón. El cálculo de los sistemas solares para esta fase de desarrollo se hizo pensando en un funcionamiento discontinuo (sólo durante las horas de luz) en la latitud de Madrid.

#### 5.1.1.2. Versión de campo

Para la versión definitiva se buscó la manera de optimizar el sistema de forma que se minimizara su coste, su mantenimiento y en lo posible también su tamaño. A diferencia de lo que puede suceder en sistemas con propósito comercial, los sistemas tecnológicos empleados en proyectos de desarrollo en países pobres tienen que tener en cuenta de forma prioritaria el coste, ya que las instalaciones que se hacen en un momento dado con una financiación puntual luego tienen que resultar sostenibles para las comunidades que se benefician de ellos y en las que recaerán los costes de mantenimiento y amortización.

Como los elementos críticos en cuanto al tamaño son los paneles solares, la cuestión se reduce en estudiar si el coste se reduce optando por sistemas que consuman menos y por lo tanto requieran un subsistema de alimentación más pequeño, o si por el contrario la reducción en el tamaño implica un aumento en el precio del ordenador empotrado que compensa la reducción en el subsistema solar. Con el mantenimiento de las instalaciones se tiene una paradoja; por una parte, el precio de la mano de obra en países pobres es inferior siempre que sea para tareas que no exijan una calificación muy alta, pero por otra parte la difícil accesibilidad a las instalaciones y la escasez de personal técnico hacen que merezca la pena hacer todo lo posible para que los

sistemas sean muy autónomos, su mantenimiento muy bajo, y que muchas tareas puedan ejecutarse a distancia.

De lo dicho en el párrafo anterior se deduce el interés por que los equipos definitivos consuman mucho menos y que los sistemas tengan en cuenta la autonomía y bajo mantenimiento. Por otra parte los ordenadores empotrados empleados tienen que mantener las mismas prestaciones, disponibilidad de interfaces para conexión de periféricos, etc. Después de un amplio estudio se ha considerado que la mejor alternativa son las PDAs Compaq iPAQ H3600 con extensión para adaptar PCMCIA. No parece necesario disponer de slots CF porque la flash interna de las PDA, de 16Mb, es suficiente para albergar el sistema software. Las PDAs están optimizadas para consumir poco ya que se trata de dispositivos móviles que funcionan la mayor parte del tiempo en batería. Por otra parte, hay distribuciones de Linux para iPAQ que tienen la mayor parte de la funcionalidad necesaria, y el software que aún no está disponible para plataformas ARM se puede producir con compilación cruzada. Lo lógico en realidad sería emplear placas como las que van dentro de las iPAQ y no el aparato entero, pero resultan ser más caras las placas solas que la PDA completa por razones de mercado.

#### 5.1.1.3. Subsistema solar

En cuanto al **subsistema de alimentación solar**, consta de cuatro clases de elementos: paneles solares, regulador de carga, baterías, y accesorios de conexionado y estructura. Sobre los últimos no se va a comentar nada.

El regulador de carga podría ser muy pequeño, pero los reguladores comerciales de uso habitual más pequeños son de 10 A. Después de haber comparado las especificaciones de varios, escogemos el ISOLER 10 de Isofotón (http://www.isofoton.es/espaniol/productos.htm), por sus protecciones eléctricas y la tropicalización de los circuitos. Este regulador es válido para las dos versiones del sistema inicialmente previstas. El número de paneles solares y/o su tamaño varía drásticamente entre los dos escenarios y en función del consumo del sistema; para el primer diseño, tomando un consumo medio de 5W (a verificar, pero resulta razonable teniendo en cuenta las especificaciones técnicas de las placas y el consumo previsible de los dispositivos accesorios) durante 8 horas de luz diarias y en Madrid (insolación diaria sobre superficie inclinada en el mes peor de unos 2kWh/m2) haría falta un panel de 22Wp como el Isofotón I22. Para el diseño definitivo, tomando la placa StrongARM de más bajo consumo de que se dispone

se tendría un consumo medio del equipo de aproximadamente 1W, pero debería estar disponible permanentemente (24h/día), en áreas tropicales cuyos datos de insolación pueden variar muchos de unos puntos a otros. Tomando como referencia Alto Amazonas, con 4270 Wh/m2 de media diaria en el mes peor, haría falta un panel de 6Wp, por lo que serviría el Isofotón I10 de 10Wp (casi exactamente del tamaño y de la forma de una hoja formato A3). En cuanto a baterías, en el escenario de pruebas haría falta una batería de un mínimo de 5Ah, y en el escenario definitivo suponiendo una autonomía de 4 días y no queriendo descargar la batería más del 50% para asegurar su durabilidad, haría falta una batería de más de 16 Ah.

### 5.1.2. Software

En cuanto al software, el sistema contiene un kernel de linux 2.4.20 con soporte para encaminamiento IP avanzado, filtrado y marcado de paquetes, etc. Además de unas mínimas herramientas y programas de línea de comando para poder trabajar mediante la consola de Linux, nos aseguramos de disponer de las utilidades necesarias para:

- Configurar las radios WiFi (driver hosta_p, prism2_param, iwconfig, iwpriv).
- Manejar las interfaces de red y las rutas, incluído el acceso a las colas (ip, ifconfig, route, tc) .
- Soporte de diferentes disciplinas de colas y planificadores implementados para Linux, así como de diffserv. Logging remoto, soporte SNMP.
- Gestión del hardware watchdog y gestión avanzada de energía. Protocolos de encaminamiento dinámico.
- Software de autoconfiguración de la dirección IP.

Después de probar varias distribuciones, la que demuestra ser más apropiada para este proyecto es WISP-Dist, una rama del proyecto LEAF. La pega es que está diseñada específicamente para procesadores x86 y hace falta producirla entera para ARM mediante compilación cruzada si se quiere que funcione en las PDAs.

### 5.1.3. Autoconfiguración de la red

Cuando un nodo cobra vida, lo primero que hace es cargar el sistema operativo. Hecho esto, ejecuta un procedimiento para autoconfigurar su dirección IP cuyo resultado es que se da a sí mismo una dirección 169.254.X.Y que es diferente de las de los otros nodos de la red a los que es capaz de localizar. Obviamente no todos los nodos hacen esta operación a la vez, y es bueno que así sea para evitar problemas de carrera. En el presente proyecto se va a emplear **zcip** para realizar la autoconfiguración de dirección IPv4 [CAG2003].

Una vez que un nodo tiene dirección IP, se arranca el software encargado del encaminamiento dinámico. El nodo buscará la ruta a otros y en particular al que haga de pasarela al exterior; a medida que todos vayan estando disponibles la red irá convergiendo a sus direcciones y rutas definitivas. Si en cualquier momento un nodo deja de estar disponible, el protocolo de encaminamiento dinámico debe reaccionar inmediatamente para que se use un camino alternativo en toda ruta que contara con ese nodo anteriormente. Se van a evaluar como protocolos de encaminamiento los de MobileMesh [MOB2003], AODV y OSPF; el primero tiene todo lo que se quiere, pero no es popular y cabe la duda de si se debe a fallos en su funcionamiento; el segundo es lo más extendido en redes mesh, pero mantiene una lógica de enlaces bi-estado que no se comporta bien en muchos casos [COU2002]; el tercero no es apropiado para redes mesh, pero dada la inmovilidad de los nodos resulta planteable y se dispone con igual facilidad de implementaciones software apropiadas para las plataformas hardware que van a emplearse.

### 5.1.4. Implementación de QoS en los nodos

Lo primero que cabe decidir en los nodos en relación a la calidad de servicio es si se van a utilizar infraestructuras 802.11b o redes ad-hoc. Teniendo en cuenta que el estándar no soporta calidad de servicio, se ha preferido pensar en las redes ad-hoc porque ofrecen dos ventajas obvias: resultan muy apropiadas para producir redes autoconfigurables y adaptables a cualquier topología, y pasan la lógica de conmutación a la capa IP, en la que sí que se soporta QoS.

Como primera medida para asegurar que la red no puede ser colapsada en condiciones normales, se van a emplear métodos de prevención de la congestión y de conformado de tráfico. Los mecanismos que se van a emplear forman parte del kernel de linux y son fáciles de usar disponiendo de las utilidades **ip**, **tc** e **iptables**.

Se trata de TBF y GRED, que nos van a permitir limitar el tráfico que puede entrar por cada nodo, de forma que se asegure que la red podrá conmutar varias comunicaciones simultáneas sin problemas.

Para implementar QoS mediante servicios diferenciados, un primer problema va a ser la identificación de nodos periféricos y nodos centrales. En este caso tiene sentido que todos los nodos de la red realicen las funciones de periféricos y centrales. Cabría poner en cuestión en este caso si tiene sentido optar por servicios diferenciados, o si más bien se puede realizar una solución propietaria pasada en una configuración ad-hoc de las disciplinas de colas con prioridades. La ventaja de usar servicios diferenciados es que nos van a permitir acotar la carga de procesador requerida para la clasificación de los paquetes, ya que cada nodo clasifica sólo los que llegan a la red por él, mientras que los que le llegan de otro nodo de la red ya están clasificados.

Cuando un nodo recibe un paquete, si se identifica como procedente de un cliente se procede al control de admisión, la clasificación y el marcado del DSCP, en cambio si procede de otro nodo de la red ad-hoc no se hace tal cosa. Todos los paquetes son examinados después para determinar el PHB (comportamiento por salto) que hay que aplicar, y según su DSCP se procede a su tratamiento apropiado en las colas de salida. El nodo que ejerce de pasarela a Internet procederá al remarcado del DSCP de todos los paquetes para asegurar un contenido coherente con la RFC 791. El resultado es que los paquetes que circulan por la red ad-hoc están marcados según sean de voz, de vídeo, de control o de tráficos elásticos. A los paquetes de voz se les asegura un mínimo retardo y una alta garantía de entrega. A los paquetes de vídeo se les da prioridad sobre los tráficos elásticos pero tendrán menos prioridad que el audio, de forma que en las sesiones de videoconferencia el audio se asegure y el vídeo se conmute lo mejor posible pero sin impedir por ello la conmutación de otros tráficos. Cuando se produce una alta ocupación en la banda de voz, cosa que no es previsible o que en el peor de los casos será infrecuente, el control de admisión prohíbe nuevas comunicaciones de voz hasta que termine alguna de las vigentes. Con el resto de los tráficos, no hay un control de admisión tan riguroso y además sí será habitual que se dé una alta ocupación. Los mecanismos implementados a nivel de colas y planificación hacen que los paquetes de voz no se vean afectados por la alta ocupación de la red debida a tráficos elásticos o de vídeo. Los tráficos elásticos en particular percibirán una red best-effort y tendrán que adaptarse a las prestaciones disponibles en cada momento.

La forma de implementar los servicios diferenciados está en relación directa con lo que se pretende diferenciar como tráficos:

- Clase EF, que va a corresponder a la voz sobre IP (se va a identificar pretendidamente por el tipo de protocolo, UDP, en conjunción al tamaño de los paquetes, entre 40 y 160 bytes, y al puerto, que debe ser mayor de 1024 en origen y en destino, distinguiéndolo así de ciertos servicios básicos como el DNS)
- Clase AF1, que va a corresponder al ssh, telnet, slogin, etc. (puertos destino correspondientes, tráfico TCP)
- Clase AF2, corresponderá al vídeo y otros tráficos UDP (todo UDP que no sea identificado como voz)
- Clase AF3, que será el de web (TCP destinado al puerto 80, 8080 y otros puertos típicos de proxies)
- Clase AF4, FTP y otros de transferencia de ficheros (TCP destinado a puertos de servidores de ficheros ftp, webdav, etc.)
- BE, para el resto

Como se ha indicado, todos los nodos deberán realizar la clasificación de paquetes, para lo que usarás sus colas ingress. Determinarán por el contenido del DSCP si se trata de paquetes ya recibidos en el dominio diferenciado o bien son recién llegados, y en este segundo caso se procederá a su clasificación y marcado.

Todos los paquetes pasarán en la salida por una disciplina DSMARK con varias subclases, asociadas a las seis clases de tráfico diferenciadas arriba, y a cada una corresponderá una cola apropiada. La EF, que se gestionará con una cola HTB, tendrá además garantizado un ancho de banda mínimo de un 30% de la banda disponible, que se medirá en laboratorio. El 45% de la banda disponible se repartirá entra las clases AF a las que se asignarán colas GRED, y un 25% se dejará para otros tráficos, que se conmutarán con la política "best effort", empleando para ello colas FIFO.

## 5.2. Los terminales

Los terminales, están normalmente ubicados en puestos de salud. Como dijimos en el análisis, pueden estar constituidos por una sóla máquina de bajo consumo que haga todas las funciones, o separar la computación de la telefonía, haciendo ésta más robusta, ya que es la más importante.

En EHAS, dada la mala experiencia con los ordenadores portátiles en zonas aisladas y malas condiciones ambientales y eléctricas, y a pesar de que éstos están diseñados para un consumo muy bajo, se ha optado por la tecnología Mini-ITX con procesadores VIA C3 o EDEN, que reune en una sóla placa de bajo consumo y precio un procesador relativamente potente (600MHz sin ventilador y a 1000 Mhz con ventilador), hardware para terminal multimedia, conexón ethernet, USB, etc. Este tipo de máquinas también se van a usar en servidores y pasarelas, incluso las de VHF y HF, ya que se puede realizar la modulación por software con ayuda de su tarjeta de sonido. Por ello, los dos tipos de soluciones que presentamos a continuación las usan.

### 5.2.1. Terminal único

Este terminal puede estar construido con una VIA EPIA MII6000, con procesador de 600 Mhz, sin ventilador, Compact Flash para el sistema operativo y adaptador CardBus (PCMCIA) donde se puede colocar una radio inalámbrica de potencia (Senao SL-2511-CD-Ext2 o similar, como las de los routers) sin antena exterior o con una pequeña que le permita alcanzar al router. Será alimentado con el mismo panel solar que todo el establecimiento. Como la impresora, la pantalla y otros elementos del establecimiento (lámparas, por ejemplo) funcionan con corriente alterna, se utilizará un inversor y una fuente ATX normal. La pantalla será de cristal líquido, para minimizar el consumo. Para almacenamiento local se utilizarán discos sólidos por USB.

Su software consistirá en un terminal sobre Linux (similar a thinstation (???) o pxes (http://pxes.sourceforge.net)) con una aplicación local de telefonía IP (GnomeMeeting o LinPhone). Esta configuración permitirá conectarse a servidores externos Linux (VNC) o Windows (RDP, ICA). Por supuesto se configurará el sistema de ahorro de energía para que el terminal no consuma practicamente nada, pero se despierte con llamadas de voz.

### 5.2.2. Computación separada

Si se opta por computación separada, la solución más simple será una pasarela con una toma FXS para un teléfono ordinario y un switch. Es conveniente que se disponga de algún codec de bajo ancho de banda no patentado, para poder interoperar con software libre, aunque sólo sea un GSM (por ejemplo, ni Cisco

(http://www.cisco.com/) ni Bosslan (http://www.bosslan.com/) lo tienen). No se ha encontrado ninguna que tenga interfaz inalámbrica, por lo que habrá que conectar un convertidor ethernet-wifi cliente a una de las bocas del switch y la computadora en otra.

La opción teléfono+switch aún no está descartada. Los teléfonos Snom pueden servir, teniendo además códec GSM.

La computadora será otra Mini-ITX más económica, como puede ser la VIA EPIA VE6000, de mitad de precio que la MII6000. En este caso tendrá disco duro donde se ubicará el sistema operativo elegido (Linux o Windows).

## 5.3. Las pasarelas

Las pasarelas están normalmente ubicadas en centros de salud, o en cualquier punto de la red que tenga salida telefónica al exterior, y/o acceso a Internet. Sus misiones son:

- Encaminar el tráfico a Internet, si hay conexión permanente. Si no hay conexión permanente y hay tarifa de llamada local, se levantará la conexión para servicios deteminados (correo de propia microrred de salud) y para usuarios autorizados (probablemente a través de un proxy autenticado). Si no hay conexión a Internet intermitente económica, sólo se cursará tráfico de correo por UUCP (directo o con transporte TCP). En todo caso esta función de encaminamiento estará asistida por proxies transparentes (al menos de web y correo), para optimizar el tráfico y para poder ajustar los parámetros de TCP de la red inalámbrica a sus características especiales.
- Encaminar llamadas de voz, haciendo la función de una centralita, con control de acceso y tarificación para las llamadas salientes. Como lo más probable es que haya una sola línea, deberían poder compartirse para ambas funciones.
- Hacer de servidor para otras máquinas del centro de salud, ejerciendo las mismas funciones que para los puestos de salud. En centros de salud muy pequeños podría pensarse en usarla como estación de trabajo. Eso lo hace más propensa a caidas por mal uso o fallas de software, por lo que no es aconsejable.

- En algunos casos la pasarela servirá puestos de salud que tengan solamente accesibilidad por VHF. En esos casos hará de pasarela de correo, pero no podrá cursar comunicaciones de voz fuera del establecimiento. Su realización es parte de otro proyecto del programa EHAS.

### 5.3.1. Pasarela única

Podrían realizarse con un PC cualquiera, normalmente uno de consumo bajo, por lo que, si es posible, recurriremos a una Mini-ITX, esta vez de gama baja (VIA EPIA 5000), cuyo bus PCI nos permitiría colocar una tarjeta de telefonía, si se opta por implementar la pasarela por software. La conexión inalámbrica y la telefónica pueden hacerse con adaptadores USB, extensores de bus PCI o usando la PCMCIA y/o la Compact Flash de una VIA EPIA MII6000. En lugares con poco tráfico, especialmente los que tienen conexión a Internet permanente, podría reemplazarse el disco duro por uno de estado sólido, mejorando la robustez del dispositivo.

### 5.3.2. VoIP separada

Más mantenible, sin embargo, puede ser utilizar una pasarela VoIP externa. En principio este dispositivo haría innecesario un servidor si hay conexión directa a Internet, bastando con el conversor ethernet-inalámbrico o un router inalámbrico para la salida al aire. No obstante, como se comentó más arriba, puede ser necesario situar un proxy que mejore el rendimiento de TCP y mantenga cachés.

En la mayoría de los centros bastará una pasarela con una línea FXS para un teléfono, y una FXO para la línea. Existen dispositivos convertidores FXO a FXS que permiten utilizar dispositivos con dos FXO, como el Cisco ATA-188[1]. Sin embargo allá donde se necesite compartir la línea para un modem, habrá que incorporar algún mecanismo de exclusión mutua.

## Notas

1. Este dispositivo no soporta códecs GSM.

# Capítulo 6. Presupuesto

## 6.1. Presupuesto de contratación

### 6.1.1. Coste de ejecución material

**Tabla 6-1. Coste de ejecución material**

| Equipos | Precio (Euros) | Duración (años) | Uso (años) | Total (Euros) |
|---|---|---|---|---|
| 2 Teléfonos IP | 150 unidad | 3 | 1 | 300 |
| 4 ordenadores portátiles con procesador Pentium 233 MHz | 150 unidad | 3 | 1 | 600 |
| 4 tarjetas inalámbricas PCMCIA | 30 unidad | 3 | 1 | 120 |
| 2 tarjetas ethernet PCMCIA | 20 unidad | 3 | 1 | 40 |
| 4 ordenadores con procesador Pentium 200 MHz | 100 unidad | 3 | 1 | 400 |

Coste total de los equipos: 1460 €

**Tabla 6-2. Fungibles**

| Elementos | Total (Euros) |
|---|---|
| Material informático y de oficina | 60,10 |
| Encuadernación | 108,80 |

Coste total de fungibles: 168,90 €

*Coste total de ejecución material: 1628,90 €*

### 6.1.2. Gastos generales

16% sobre la ejecución material para amortizaciones y cargas fiscales: 260,62 €

### 6.1.3. Beneficio industrial

6% sobre la ejecución material: 97,73 €

## 6.2. Honorarios

**Tabla 6-3. Coste de ejecución material**

| Personas | Tiempo (horas) | Coste por hora (Euros) | Total (Euros) |
|---|---|---|---|
| Ingeniero Junior | 300 | 12 | 3600 |
| Mecanógrafo | 75 | 10 | 750 |

Total honorarios: 4350 €

## 6.3. Presupuesto total

**Tabla 6-4. Presupuesto total**

| | |
|---|---|
| Total presupuesto de contratación | 1987,25 € |
| Total honorarios | 4350 € |
| SUBTOTAL | 6337,25 € |
| 16% IVA | 1013,96 € |
| TOTAL | 7351,21 € |

El presupuesto final de este proyecto asciende a **SIETE MIL TRESCIENTOS CINCUENTA Y UNO COMA VEINTIUNO** euros.

# Capítulo 7. Bibliografía

## Referencias


[HGP2000] *IP Telephony. Packet-based multimedia communications systems.*, Olivier Hersen, David Gurle, y Jean-Pierre Petit, 2000, Addison Wesley.

[MVS2002] Andrés Martínez, Valentín Villarroel, Joaquín Seoane, y Francisco del Pozo, *EHAS Program: Rural Telemedicine Systems for Primary Healthcare in Developing Countries*, 2002, Proceedings of the International Symposium on Technology and Society.

[CAG2003] *Dynamic Configuration of Link-Local IPv4 Addresses*, S. Cheshire, B. Aboba, y E. Guttman, Septiembre de 2003, IETF, http://www.ietf.org/internet-drafts/draft-ietf-zeroconf-ipv4-linklocal-10.txt.

[COU2002] *Performance of Multihop Wireless Networks: Shortest Path is Not Enough*, D.S.J. de Couto, Octubre de 2002, First Workshop on Hot Topics in Networking, Princeton, New Jersey.

[MOB2003] *MobileMesh: Providing Solutions For Mobile Adhoc Networking*, Octubre de 2003, Mitre Corporation, http://www.mitre.org/work/tech_transfer/mobilemesh/.

[RAD1999] *Linux - Advanced Networking Overview*, S. Radhakrishnan, Agosto de 1999, ITTC, University of Kansas, http://www.qos.ittc.ukans.edu/howto.pdf.

[MAL2003] *HostAP driver for Intersil Prism2/2.5/3*, J. Malinen, Noviembre de 2003, http://hostap.epitest.fi/.

[HUB2003] *Linux Advanced Routing & Traffic Control*, B. Hubert, Julio de 2003, http://www.lartc.org/.

[AND2003] *A Linux Wireless Access Point HOWTO*, S. Anderson, Junio de 2003, http://oob.freeshell.org/nzwireless/LWAP-HOWTO.html.

[DSL2001] *Differentiated Services on Linux*, Mayo de 2001, http://diffserv.sourceforge.net/.

[ASK1999] *Differentiated Services on Linux*, W. Almesberger, J.H. Salim, y A. Kuznetsov, Junio de 1999, http://www.almesberger.net/cv/papers/18270721.pdf.

[TOU2003] *Wireless LAN resources for Linux*, J. Tourrilhes, Marzo de 2003, http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Wireless.html.

[HAJ1998] *RFC2327: SDP: Session Description Protocol*, M. Handley y V. Jacobson, Abril de 1998, IETF.

[BBC1998] *RFC2475: An architecture for Differentiated Services*, S. Blake, D. Black, M. Carlson, E. Davies, Zh. Wang, y W. Weiss, 1998, IETF.

[NBB1998] *RFC2474: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, K. Nichols, S. Blake, F. Baker, y D. Black, 1998, IETF.

[HBW1999] *RFC2597: Assured Forwarding PHB Group*, J. Heinanen, W. Weiss, F. Baker, y J. Wroclawski, 1999, IETF.

[JNP1999] *RFC2598: Expedited Forwarding PHB*, V. Jacobson, K. Nichols, y K. Poduri, 1999, IETF.

[RSC2002] *RFC3261: SIP: Session Initiation Protocol*, J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, y E. Schooler, Junio de 2002, IETF.

[SCF2003] *RFC3550: RTP: A Transport Protocol for Real-Time Applications*, Henning Schulzrinne, Stephen L. Casner, Ron Frederick, y Van Jacobson, Julio de 2003, IETF.

[CAS2003] *RFC3551: RTP Profile for Audio and Video Conferences with Minimal Control*, Stephen L. Casner, Julio de 2003, IETF.

[ITU2001] *Draft H.323v4 (Including Editorial Corrections - February 2001)*, ITU Telecommunication Standardization Sector, Febrero de 2001, Editado por Paul

Jones, ITU Telecommunication Standardization Sector, http://ftp3.itu.int/av-arch/avc-site/till_0012/0011_Gen/H323v4-final_010206.zip.

[JON2003] *H.323 Protocol Overview*, Paul E. Jones, Febrero de 2003, http://www.packetizer.com/iptel/h323/papers/h323_protocol_overview.ppt.

[SCH2003] *Basic Architecture of H.323*, C. Schlatter, 2003, http://www.switch.ch/vconf/ws2003/h323_basics_handout.pdf.

[SIK2002] *Understanding SIP*, Dorgham Sisalem y Jiri Kuthan, 2000-2002, GMD Fokus, http://iptel.org/sip/.

[RAD2001] *SIP: Protocol Overview*, RadVision white paper, 2001, RadVision, http://www.radvision.com/papers/C1_What_is_SIP.html.