TRABAJO ESPECIAL DE GRADO

SOFTWARE DIDÁCTICO PARA LA ENSEÑANZA DE VOIP

Presentado ante la Ilustre
Universidad Central de Venezuela
por el Br. Palacios V., Otoniel A.
para optar al título de
Ingeniero Electricista

Caracas, 2006

TRABAJO ESPECIAL DE GRADO

SOFTWARE DIDÁCTICO PARA LA ENSEÑANZA DE VOIP

Tutor Académico: Ing. Luis Fernández.

Presentado ante la Ilustre
Universidad Central de Venezuela
Por el Br. Palacios V., Otoniel A.
para optar al título de
Ingeniero Electricista

Caracas, 2006

DEDICATORIA

A mi Madre por su apoyo incondicional.

Palacios V., Otoniel A.

SOFTWARE DIDÁCTICO PARA LA ENSEÑANZA DE VOIP

Tutor Académico: Ing. Luis Fernández. Tesis. Caracas. U.C.V. Facultad de Ingeniería. Escuela de Ingeniería Eléctrica. Ingeniero Electricista. Opción: Comunicaciones. 2006. 89h. +anexos.

Palabras claves: Voz sobre IP, VoIP, Recomendación UIT-T H.323, Recomendación UIT-T Q.931, Recomendación UIT-T H.225, Protocolo de Iniciación de Sesiones, SIP, Especificación RFC 3261.

Resumen. Actualmente Voz sobre IP (VoIP) se ha convertido en la principal alternativa al servicio de telefonía convencional. Los protocolos H.323 y SIP son los más usados hoy en día en VoIP. Por esta razón, en este trabajo se desarrolla una aplicación computacional del tipo analizador de protocolo que muestra el proceso de señalización para el establecimiento de llamadas en las recomendaciones H.323 y SIP. Esta aplicación es construida en el lenguaje de programación Java. Para desarrollar esta aplicación fue necesario crear los bloques de programa interfaz de usuario, captura de trama, decodificación de tramas, filtrado de tramas. También se propone una guía de práctica de laboratorio que se usará en conjunto con esta aplicación en el estudio de los protocolos H.323 y SIP.

ÍNDICE GENERAL

Pág.

CAPÍTULO III

RECOMENDACIÓN UIT-T H.323

LISTA DE FIGURAS

FIGURAS Pág.

LISTA DE TABLAS

TABLAS Pág.

ACRÓNIMOS

ACF	Admisión Confirmada
ARJ	Admisión Rechazada
ARP	Protocolo de Resolución de Direcciones
ARQ	Solicitud de Admisión
ASN.1	Notación de Sintaxis Abstracta número 1
CID	Identificador de Conferencia
CODEC	Codificador-Decodificador
CRV	Valor de Referencia de Llamada
DCF	Desligamiento Confirmado
DRJ	Desligamiento Rechazado
DRQ	Solicitud de Desligamiento
HTTP	Protocolo de Transferencia de HiperTexto
IA5	Alfabeto Internacional númer 5
IETF	Grupo de Trabajo en Ingeniería de Internet
IP	Protocolo de Internet
ISDN	Red de Servicios Integrados
ITU	Unión Internacional de Telecomunicaciones
MC	Controlador Multipunto
MCU	Unididad de Control Multipunto
MP	Procesador Multipunto
PER	Reglas de Codificación de Empaquetamiento
PSTN	Red Pública de Telefonía Conmutada
QoS	Calidad de Servicio
RAS	Registro, Admisiones y Situación
RTCP	Protocolo de Control en Tiempo Real
RTP	Protocolo de Transmisión en Tiempo Real
SDP	Protocolo de Descripción de Sesiones

SIP	Protocolo de Iniciación de Sesiones
TCP	Protocolo de Control de Transmisión
TPKT	Paquete de Transporte
TSAP	Punto de Acceso al Servicio de Transporte
UA	Agente de Usuario
UAC	Agente de Usuario Cliente
UAS	Agente de Usuario Servidor
UDP	Protocolo de Datagrama de Usuarios
URI	Identificador Uniforme de Recursos
UTF-8	Formato de Transformación Unicode de 8 bits
VOCODEC	Codificador-Decodificador de Voz
VoIP	Voz sobre IP

INTRODUCCIÓN

El protocolo de Internet (IP) es hoy en día uno de los elementos más importantes en la tecnología de la información. En términos prácticos, se trata de un protocolo que sirve como base para otros que permiten compartir archivos, impresoras, correo electrónico y mensajería instantánea, entre otros servicios. Todos estas aplicaciones han revolucionado la manera en que trabajan y se comunican las personas. Dentro de estas aplicaciones se han incorporado aquellas que hacen posible la transmisión de voz y video a través de IP.

El crecimiento y la fuerte implantación de las redes IP, el desarrollo de técnicas avanzadas de digitalización de voz, mecanismos de control y asignación de prioridad al tráfico, protocolos de transmisión en tiempo real, así como el estudio de nuevos estándares que garanticen la calidad de servicio en redes IP, han creado un entorno donde es posible transmitir telefonía sobre IP. En los últimos años el interés por las redes de paquetes IP como soporte de tráfico multimedia ha experimentado notable crecimiento. En tal sentido, tanto la ITU (Unión Internacional de Telecomunicaciones) como el IETF (Grupo de Trabajo en Ingeniería de Internet) han estado desarrollando arquitecturas y protocolos para sistemas multimedia sobre IP. Dentro de estos se encuentran la recomendación H.323 desarrollado por la ITU y el estándar SIP (RFC 3261) creado por al IETF, los cuales son los más usados hoy en día en VoIP.

Por consiguiente, se hace cada vez más evidente la necesidad de conocer el funcionamiento de estas dos tecnologías. Y la mejor forma de entender cualquier tecnología es a través de una correcta adquisición de conceptos teóricos, acompañada por el trabajo práctico en el laboratorio que permitan verificar y poner en práctica ese conocimiento teórico. Por ello, el objetivo general de este trabajo es la implementación de una aplicación del tipo analizador de protocolos que sirva para la realización de una práctica de laboratorio. En esta práctica, el estudiante deberá

estudiar el proceso de establecimiento de una llamada en los protocolos H.323 y SIP. Por lo tanto, la aplicación debe ser capaz de capturar, decodificar y mostrar al usuario los mensajes intercambiados en la señalización de una llamada H.323 o SIP.

En este sentido, este trabajo primero que nada describe de forma general la tecnología voz sobre IP en el capítulo I.I. Luego, en el capítulo III se explica el proceso para establecer una llamada en la recomendación H.323. En este proceso se utilizan mensajes Q.931 bajo restricciones de la recomendación H.225. Por lo tanto, para la implementación de la aplicación se debe conocer como se codifican estos mensajes. El formato de estos mensajes se describe en el capítulo IV. En cuanto a SIP, en el capítulo V se explica la forma en que se establece una llamada en este estándar. Finalmente, en el capítulo VI se describe la estructura y funcionamiento de la Aplicación Analizador de Protocolos que se obtuvo como resultado de este trabajo especial de grado.

Sin embargo, para poder decodificas estos mensajes, tanto los de H.323 como los de SIP, fue necesario decodificar una serie de cabeceras que encapsulan y permiten el correcto transporte de estos mensajes. Estas cabeceras son las definidas en los protocolos 802.3, IPv4, UDP, TCP y la cabecera TPKT definida en el estándar RFC 1006. La estructura de estas cabeceras se explican en los anexos 1, 2, 3, 4 y 5. En el anexo 6 se encuentra la guía para la práctica de laboratorio. En el anexo 7 se encuentra el manual de uso de la aplicación Analizador de Protocolos, en el anexo 8 el manual de instalación y en el anexo 9 el código de este programa.

CAPÍTULO I

OBJETIVOS DE LA TESIS

1.1 Planteamiento del problema

En el mundo actual, en el que se ha hecho indispensable los servicios del creciente campo de las telecomunicaciones, el Protocolo de Internet (IP) representa una base fundamental para un gran número de aplicaciones de comunicaciones. Dentro de estas se encuentran aquellas que hacen posible la transmisión de voz y video a través de IP. Los cambios que IP ha propiciado en este ámbito, permiten a las redes de datos ofrecer nuevos y mejores servicios en telefonía, dentro de los cuales resalta VoIP. Hoy en día VoIP se ha convertido en la principal alternativa del servicio de telefonía convencional. Históricamente, la tecnología H.323 para VoIP ha sido ampliamente soportado por muchos fabricantes comerciales y se usa en todo el mundo. Por otro lado, el estándar SIP es un reciente protocolo abierto, con una creciente popularidad, gran integración con Internet y un establecimiento de llamadas más rápido que en H.323. Por estas razones, se hace evidente la necesidad de la enseñanza de estas dos tecnologías de VoIP.

Por otra parte, en cualquier ámbito y en cualquier parte del mundo, la enseñanza es inútil sin un correspondiente adiestramiento práctico.. Para lograr el verdadero y duradero aprendizaje, el ser humano necesita primero, ver una verificación real y práctica de que lo que se le está enseñando es cierto y segundo, poner en práctica de alguna forma y al menos en un pequeño número de veces lo que se le está enseñando. De no ser así, es muy probable que el aprendizaje sea poco y que se olvide rápidamente. Por consiguiente, surge la necesidad de crear una herramienta, tal como un analizador de protocolo, que sirva para que los estudiantes verifiquen y pongan en práctica los detalles técnicos de los protocolos H.323 y SIP.

1.2 Objetivo general de la tesis

Desarrollar una aplicación tipo analizador de protocolo que permita la enseñanza de las tecnologías VoIP.

1.3 Objetivos específicos

a) Desarrollar un analizador de protocolos apropiado para el estudio de los protocolos H.323 y SIP de VoIP, con un criterio docente.

b) Desarrollar una práctica de laboratorio basada en esta aplicación.

1.4 Metodología

La realización de este trabajo especial de grado contempla las siguientes fases:

Fase 1. Estudio y aprendizaje del lenguaje de programación Java.

Fase 2. Investigación sobre VoIP en general.

Fase 3. Investigación sobre el protocolo H.323.

Fase 4. Investigación sobre el protocolo SIP.

Fase 5. Desarrollo y prueba del módulo Captador de tramas.

Fase 6. Desarrollo y prueba de la Interfaz de Usuario.

Fase 7. Desarrollo y prueba del módulo Decodificador 802.3.

Fase 8. Desarrollo y prueba del módulo Decodificador IPv4.

Fase 9. Desarrollo y prueba del módulo Decodificador TCP.

Fase 10. Desarrollo y prueba del módulo Decodificador UDP.

Fase 11. Desarrollo y prueba del módulo Decodificador TPKT.

Fase 12. Desarrollo y prueba del módulo Filtro de Tramas.

Fase 13. Desarrollo y prueba del módulo Decodificador Q.931.

Fase 14. Desarrollo de un prototipo de práctica docente.

Fase 15. Redacción del informe final.

CAPÍTULO II

VOZ SOBRE IP (VoIP)

2.1 Definición

Voz sobre IP (VoIP) es la tecnología que permite la transmisión de fragmentos auditivos a través de redes basadas en el Protocolo de Internet (IP).

2.2 Historia

La primera aplicación real de VoIP fue introducida en 1995 por una compañía israelí llamada VocalTec [11]. La aplicación permitía transmitir voz entre dos computadores personales. La idea consistía en comprimir la señal de voz y trasladarla a paquetes IP para su transmisión sobre Internet. Esta primera generación sufría de retardos debidos a la congestión, desconexiones, incompatibilidad y baja calidad debido a la perdida y a la recepción fuera de orden de los paquetes.

Sin embargo, VoIP se desarrolló rápidamente. El avance más importante fue la creación de las pasarelas que actúan como una interfaz entre redes IP y PSTN. Posteriormente fueron desarrollados los controladores de acceso y las unidades de control multipunto que permitieron un control más avanzado sobre las comunicaciones IP. En los siguientes años, VoIP se desarrolló gradualmente alcanzando el punto, alrededor de 1998, en el que algunas compañías ofrecían servicios desde computadores personales a teléfonos convencionales y tiempo después desde teléfonos convencionales a teléfonos convencionales. La gradual introducción de servicios de Ethernet banda ancha permitió una mayor claridad y una menor latencia en las llamadas, aunque todavía se presentaban inconvenientes para establecer conexiones entre Internet y las redes PSTN [12].

Un avance importante para VoIP llegó cuando compañías fabricantes de hardware tales como Cisco, System y Nortel comenzaron a producir equipos para VoIP, los cuales eran más confortables y podían realizar las funciones que eran manejadas por computadores. Esto propició a que las grandes compañías pudieran implementar VoIP sobre sus redes IP y a que proveedores de telefonía a larga distancia comenzaran a encaminar algunas de sus llamadas a través de Internet [12].

A partir del año 2000, el uso de VoIP se expandió dramáticamente. Existen diferentes estándares técnicos para la transmisión y conmutación de paquetes VoIP, y cada uno es soportado por al menos uno de los más grandes fabricantes [12]. Sin embargo, no ha emergido un claro merecedor del rol de estándar universal. Por otra parte, mientras muchas compañías adoptan VoIP para minimizar costos tanto de llamadas de larga distancia como de infraestructura, los servicios de VoIP también han sido extendidos a los usuarios residenciales. En resumen, en sólo unos pocos años VoIP pasó de ser un pequeño ensayo a ser la principal alternativa del servicio de telefonía convencional.

2.3 Funcionamiento

Las funciones básicas que debe realizar un sistema VoIP son:

- Señalización
- Servicios de base de datos
- Codificación y decodificación de la voz
- Empaquetamiento de la voz y enrutamiento de los paquetes

2.3.1 Señalización

La señalización en VoIP activa y coordina los diversos componentes para completar una llamada. La señalización se lleva a cabo por el intercambio de

mensajes datagramas entre los componentes. El formato de estos mensajes es cubierto por ciertos estándares. Independientemente del protocolo usado, estos mensajes son críticos y requieren un tratamiento especial para garantizar su entrega.

2.3.2 Servicios de base de datos

Los servicios de base de datos son una forma de localizar un terminal y traducir su identificación a una dirección IP y un número de puerto. Una base de datos de control almacena estas identificaciones y sus traducciones para un posible uso futuro. Otra función importante es la generación de reportes de tarifación.

2.3.3 Codificación y decodificación de la voz

Para poder transmitir la voz a través de una red IP es necesario transformar la señal de voz en información digital, lo cual se logra a través de un codificador. De forma inversa, también es necesario un decodificador para convertir esta información digital en una señal auditiva. La unión de estos dos traductores suele llamarse CODEC (codificador-decodificador), y en el caso particular de la voz, VOCODEC (voz codificador-decodificador). Un VOCODEC aprovecha las características estadísticas naturales de la voz para lograr codificaciones y decodificaciones más eficientes. Además de convertir la señal de voz analógica en digital, un VOCODEC comprime este flujo de datos para aprovechar mejor el ancho de banda, realiza funciones de cancelación de eco y de supresión de silencio.

2.3.4 Empaquetamiento de la voz y enrutamiento de los paquetes

La información digital obtenida de la codificación de la voz debe ser insertada en un paquete de datos usando un protocolo en tiempo real, generalmente, RTP sobre UDP sobre IP.

UDP (Protocolo de Datagrama de Usuario) es un protocolo a nivel de transporte basado en el intercambio de datagramas. Permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión. El propio datagrama incorpora suficiente información de direccionamiento en su cabecera. No tiene un mecanismo de confirmación, ni de control de flujo, por lo que los paquetes pueden adelantarse unos a otros; y tampoco verifica si ha llegado correctamente, ya que no hay confirmación de entrega o de recepción. Se usa en la transmisión de audio y vídeo en tiempo real, donde no es posible realizar retransmisiones por los estrictos requisitos de retardo que se tiene en estos casos y también porque no sería imperceptible en el receptor la pérdida de un paquete aislado de voz.

UDP no realiza ninguna función para manejar la pérdida de paquetes o para asegurar el orden de entrega de los mismos, esta función recae sobre RTP. RTP (Protocolo de Transporte en Tiempo Real) es un protocolo de nivel de aplicación utilizado para la transmisión de información en tiempo real, como por ejemplo audio y video en una video-conferencia. Las principales funciones que proporciona RTP son: indicación del tipo de servicio multimedia, señalización de la actividad de voz, identificación del emisor, sincronización de los paquetes del flujo, detección de pérdidas, segmentación y reensamblado de objetos multimedia y opciones de cifrado(seguridad).

Finalmente, para obtener la mejor calidad de servicio, la infraestructura IP debería tratar de forma diferente a los tráficos de voz y de datos. La red IP debería ser capaz de diferenciar estos dos tipos de datos y darle mayor prioridad al tráfico de voz.

2.4 Calidad de servicio

La calidad de servicio (QoS) es un conjunto de requisitos del servicio que debe cumplir la red en el transporte de un flujo. QoS se refiere a la capacidad que debe poseer una red para proporcionar un servicio aceptable al tráfico seleccionado

sobre alguna tecnología, en este caso, VoIP. Una red debe prestar cierto nivel de calidad de servicio para un nivel de tráfico garantizando un conjunto especificado de parámetros, los cuales son entre otros: el retardo, la variación del retardo o fluctuación (jitter) y la pérdida de paquetes.

Si de algo adolece todavía VoIP, es de la baja calidad de servicio. Estos problemas son muchas veces inherentes a la red utilizada (por ejemplo, Internet). Por consiguiente, es necesario la definición de cada uno de los problemas que se presentan en VoIP con el fin de encontrarles sus causas y sus posibles soluciones. Para identificar estos problemas, se definen parámetros como latencia, fluctuación (jitter), eco y pérdida de paquetes.

La latencia se define técnicamente en VoIP como el tiempo promedio que tarda un paquete en llegar desde la fuente al destino. También se le llama retardo. Es un problema general de las redes de telecomunicación. Por ejemplo, la latencia en los enlaces vía satélite es muy elevada por las distancias que debe recorrer la información. Es un problema frecuente en enlaces lentos o congestionados. No hay una solución que se pueda implementar de manera sencilla. Muchas veces depende de los equipos por los que pasan los paquetes, es decir, de la red misma. Se puede intentar reservar un ancho de banda de origen a destino o señalizar los paquetes con valores de tipo de servicio para intentar que los equipos sepan que se trata de tráfico en tiempo real y lo traten con mayor prioridad, sin embargo, actualmente no suelen ser medidas muy eficaces ya que no se dispone del control de la red.

La fluctuación (jitter) se define técnicamente como la variación en el tiempo de llegada de los paquetes, causada por congestión de red, perdida de sincronización o por las diferentes rutas seguidas por los paquetes para llegar al destino. El jitter es un efecto de las redes de datos no orientadas a conexión y basadas en conmutación de paquetes. Como la información se discretiza en paquetes, cada uno de los paquetes puede seguir una ruta distinta para llegar al destino. En general, es un problema

frecuente en enlaces lentos o congestionados. La solución más ampliamente adoptada es la utilización del jitter buffer [13]. El jitter buffer consiste básicamente en asignar una pequeña cola o almacén para ir recibiendo los paquetes y sirviéndolos con un pequeño retraso. Si algún paquete no está en el buffer (se perdió o no ha llegado todavía), cuando sea necesario se descarta. Normalmente en los teléfonos IP (hardware y software) se pueden modificar los buffers. Un aumento del buffer implica menos jitter y menos pérdida de paquetes, pero también implica más retraso. Una disminución implica menos retardo pero más jitter y más pérdida de paquetes.

El eco se define como una reflexión retardada de la señal acústica original. El eco es fenómeno que se produce por un retorno de la señal que se escucha por los altavoces y se cuela de nuevo por el micrófono. El eco es especialmente molesto cuanto mayor es el retardo y cuanto mayor es su intensidad, con lo cual se convierte en un problema en VoIP puesto que los retardos suelen ser mayores que en la red de telefonía tradicional. Una posible solución sería la implementación de supresores de eco. Esto consiste en evitar la señal emitida sea devuelta convirtiendo por momentos la línea full-duplex en una línea half-duplex de tal manera que si se detecta comunicación en un sentido se impide la comunicación en sentido contrario [14]. El tiempo de conmutación de los supresores de eco es muy pequeño. Impide una comunicación full-duplex plena. Otra solución sería el uso de canceladores de eco. La cancelación de eco es un sistema por el cual el dispositivo emisor guarda la información que envía en memoria y es capaz de detectar en la señal de vuelta la misma información (tal vez atenuada y con ruido). El dispositivo filtra esa información y cancela esas componentes de la voz. Esta solución requiere mayor tiempo de procesamiento.

Las pérdidas de paquetes es un problema que se produce porque en VoIP al perderse un paquete, este no se reenvía. También se produce por descartes de paquetes que no llegan a tiempo al receptor. Sin embargo la voz es bastante previsible y si se pierden paquetes aislados se puede recomponer la voz de una manera bastante

óptima. El problema es mayor cuando se producen pérdidas de paquetes en ráfagas. Pero la pérdida de paquetes máxima es muy dependiente del CODEC que se utiliza. Cuanto mayor sea la compresión del CODEC más pernicioso es el efecto de la pérdida de paquetes. Para disminuir la pérdida de paquetes una técnica muy eficaz en redes con congestión o de baja velocidad es no transmitir los silencios. Gran parte de las conversaciones están llenas de momentos de silencio. Si solo se transmite cuando haya información audible se libera bastante los enlaces y se evita fenómenos de congestión.

CAPÍTULO III

RECOMENDACIÓN UIT-T H.323 [9]

La recomendación H.323 es una recomendación del UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) que tiene como título "Sistemas de comunicaciones multimedia basados en paquetes". H.323 expone los requisitos técnicos de los sistemas de comunicaciones multimedios en aquellas situaciones en las que la red de transporte subyacente es una red por paquetes que puede no garantizar la calidad de servicio. Esta Recomendación describe los componentes de un sistema H.323, lo que incluye terminales, pasarelas, controladores de acceso, controladores multipunto, procesadores multipunto y unidades de control multipunto. También describe los mensajes y procedimientos de control para comunicar estos componentes. H.323 no puede considerarse como un protocolo completo en sí mismo, sino más bien se trata de una especificación que define el modo de interactuar de varios protocolos entre sí.

3.1 Componentes

Como se mencionó anteriormente, la recomendación H.323, entre otras cosas, describe componentes pertenecientes a su estructura, tales como: terminales, pasarelas, controladores de acceso, controladores multipunto, procesadores multipunto y unidades de control multipunto.

3.1.1 Terminal

Un terminal H.323 es un punto extremo de la red que facilita las comunicaciones en tiempo real y en los dos sentidos con otro terminal, pasarela o unidad de control multipunto H.323. Esta comunicación consta de control,

indicaciones, audio, imágenes de vídeo en color y en movimiento y/o datos entre los dos terminales. Un terminal puede proporcionar sólo señales vocales, señales vocales y datos, señales vocales y vídeo o señales vocales, datos y vídeo.

3.1.2 Pasarela

Una pasarela H.323 es un punto extremo en la red que proporciona comunicaciones en ambos sentidos en tiempo real entre terminales H.323 de la red por paquetes y otros terminales UIT en una red con conmutación de circuitos, o con otra pasarela H.323. La pasarela proporcionará la conversión adecuada entre formatos de transmisión y entre procedimientos de comunicaciones. La pasarela llevará a cabo además el establecimiento y la liberación de la llamada en el lado red y en el lado RCC (red de conmutación de circuitos).

3.1.3 Guardián de puerta

Es una entidad H.323 de la red que facilita la traducción de direcciones y controla el acceso a la red para terminales, pasarelas y MCU H.323. El guardián de puerta puede prestar también otros servicios a los terminales, las pasarelas y las MCU, tales como la gestión de anchura de banda y la localización de pasarelas. Es opcional en un sistema H.323.

3.1.4 Unidad de control multipunto

La unidad de control multipunto (MCU) es un punto extremo de la red que permite que tres o más terminales y pasarelas participen en una conferencia multipunto. También puede conectar dos terminales en una conferencia punto a punto que más tarde pasa a ser una conferencia multipunto. La MCU consta de dos partes: un controlador multipunto obligatorio y procesadores multipunto opcionales. En el caso más sencillo, una MCU puede estar constituida solamente por un MC, sin

procesadores multipunto. Una MCU puede también ser incluida en una conferencia por el guardián de puerta sin ser explícitamente llamada por uno de los puntos extremos.

3.1.5 Controlador multipunto

El controlador multipunto (MC) es una entidad H.323 de la red que permite el control de tres o más terminales que participan en una conferencia multipunto. También puede conectar dos terminales en una conferencia punto a punto que más tarde pasa a ser una conferencia multipunto. El MC proporciona la capacidad de negociación con todos los terminales para conseguir niveles comunes de comunicación. También puede controlar recursos de conferencia, por ejemplo quién multidifunde vídeo. El MC no efectúa el mezclado o la conmutación de audio, vídeo ni datos.

3.1.6 Procesador multipunto

El procesador multipunto (MP) es una entidad H.323 de la red que permite el procesamiento centralizado de los trenes de audio, vídeo y/o datos en una conferencia multipunto. El MP recibe trenes de audio, vídeo y/o datos de los puntos extremos que participan en una conferencia multipunto centralizada o híbrida, procesa estos trenes de medios y los devuelve a los puntos extremos. Proporciona el mezclado, la conmutación u otro tipo de procesamiento de los trenes de medios bajo control del MC (controlador multipunto). El MP puede procesar un solo tren o múltiples trenes de medios, dependiendo del tipo de conferencia soportada.

3.1.7 Arquitectura típica

La siguiente figura muestra un arquitectura típica de una red H.323.

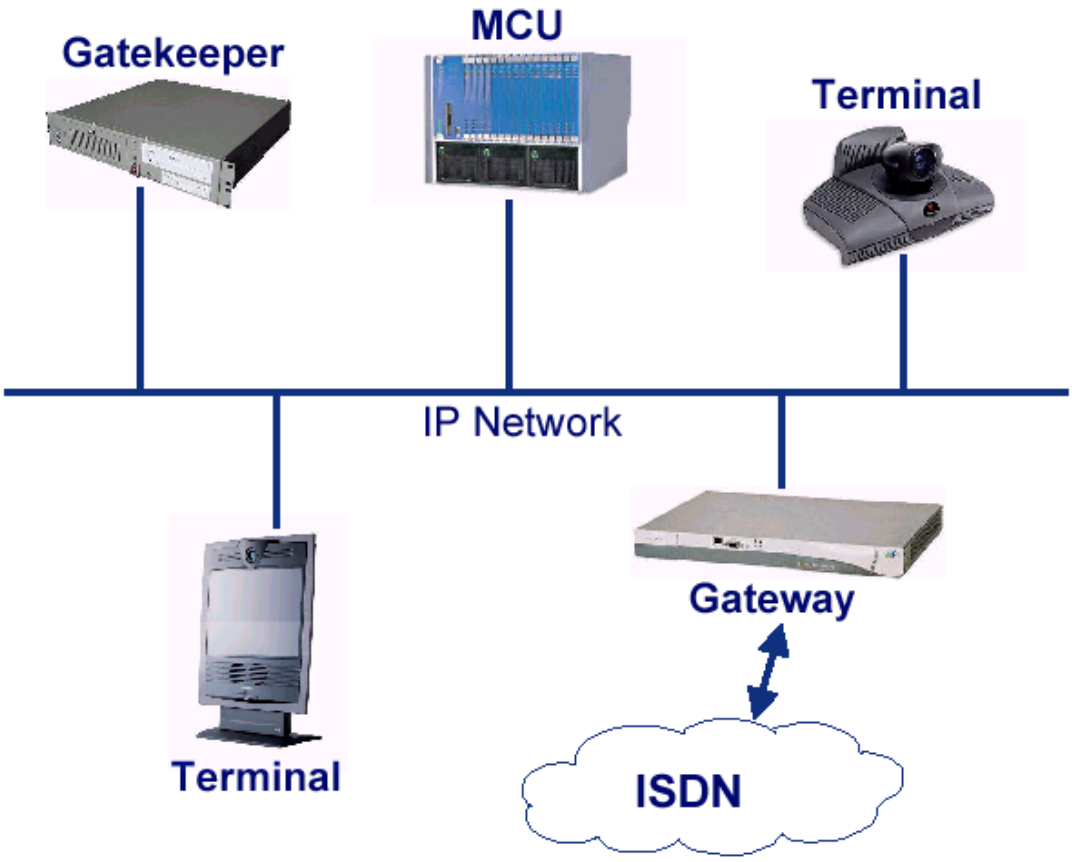


Figura 1. Arquitectura típica de una red H.323 [15].

3.2 Protocolos

H.323 no puede considerarse como un protocolo completo en sí mismo, sino más bien se trata de una especificación que define el modo de interactuar de varios protocolos entre sí. En la siguiente figura se muestra la pila de los protocolos más relevantes en H.323.

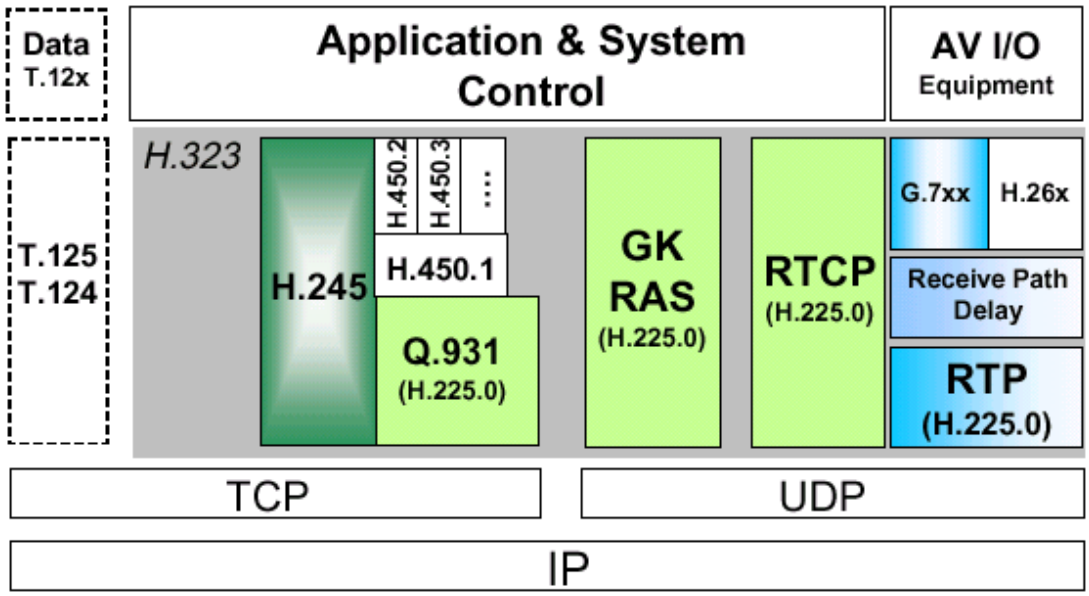


Figura 2. Pila de protocolos de H.323 [16].

3.2.1 H.225

H.225 es un protocolo de señalización de llamada y empaquetamiento de trenes de medios para sistemas de comunicación multimedios por paquetes. Esta recomendación describe los métodos por los que se asocian, codifican y empaquetan las señales de audio, vídeo, datos y control para su transporte entre equipos H.323 por una red de paquetes. Sus funciones pueden dividirse en tres partes:

- Registro, admisiones y situación (RAS)
- Señalización de la llamada
- Empaquetamiento

3.2.1.1 Registro, admisiones y situación (RAS)

La función de señalización RAS utiliza mensajes H.225.0 para llevar a cabo los procedimientos de registro, admisiones, cambios de anchura de banda, situación y

desligamiento entre puntos extremos y controladores de acceso. El canal de señalización RAS es independiente del canal de señalización de llamada y del canal de control H.245. En los entornos de red que no tienen un guardián de puerta, no se utiliza el canal de señalización RAS. En los entornos de red que sí tienen un guardián de puerta, el canal de señalización RAS se abre entre el punto extremo y el guardián de puerta. El canal de señalización RAS se abre antes de que se establezca cualquier otro canal entre puntos extremos H.323 [9].

La recomendación H.225 define una sintaxis de mensajes a través de un árbol ASN.1. Estos mensajes son codificados según la regla PER de ASN.1.

3.2.1.2 Señalización de la llamada

La función de señalización de llamada consiste en los mensajes y procedimientos utilizados para establecer una llamada, pedir cambios de anchura de banda de la llamada, obtener el estado de los puntos extremos de la llamada y desconectar la llamada. Para ello se utilizan mensajes codificados según la recomendación Q.931 y bajo las restricciones y modificaciones establecidas por la recomendación H.225. La recomendación H.225 indica cuales de los tipos de mensajes y elementos de información definidos en Q.931 serán usados en H.323, además de imponer algunas modificaciones en la estructura de ciertos elementos de información.

El canal de señalización de llamada es independiente del canal de RAS y del canal de control H.245. El canal de señalización de llamada se abre antes del establecimiento del canal H.245 y de cualquier otro canal lógico entre puntos extremos H.323. En los sistemas que no tienen un guardián de puerta, el canal de señalización de llamada se abre entre los dos puntos extremos que participan en la llamada. En los sistemas que sí tienen un guardián de puerta, el canal de señalización

de llamada se abre entre el punto extremo y el guardián de puerta o entre los propios puntos extremos, según decida el guardián de puerta.

3.2.1.3 Empaquetamiento

Esta función de H.225 formatea los trenes de vídeo, audio, datos y control transmitidos en mensajes de salida hacia la interfaz de la red y recupera los trenes de vídeo, audio, datos y control recibidos de los mensajes que han sido introducidos desde la interfaz de la red. Además, lleva a cabo la alineación de trama lógica, la numeración secuencial, la detección de errores y la corrección de los mismos según conviene a cada tipo de medio. H.225 hace uso del protocolo en tiempo real/protocolo de control en tiempo real (RTP/RTCP, real-time transport protocol/real-time transport control protocol) para la empaquetamiento y sincronización de medios. También controla aspectos como el número de canales lógicos y los limites de la velocidad binaria de los canales lógicos.

3.2.2 H.245

Esta recomendación es un protocolo de control para comunicaciones multimedios. Especifica la sintaxis y la semántica de los mensajes de información de terminal así como los procedimientos para utilizarlos en la negociación al comienzo de la comunicación o durante ésta. Se definen procedimientos para llevar los mensajes de control de extremo a extremo que rigen el funcionamiento de la entidad H.323, incluyendo el intercambio de capacidades, apertura y cierre de canales lógicos, peticiones de modo preferido, mensajes de control de flujo e instrucciones e indicaciones generales. Esta recomendación define una sintaxis de mensajes a través de un árbol ASN.1. Estos mensajes son codificados según la regla PER de ASN.1.

3.2.3 Otros protocolos

- H.450: Describe los servicios suplementarios (transferencias de llamada, sostener llamadas, llamadas en espera, identificación de nombres, etc.
- H.235: Describe la seguridad de H.323 (autenticación, cifrado, etc.).
- T.120: H.323 hace uso de este protocolo para el empaquetamiento y envío de datos, por ejemplo en la transferencia de archivos o en la transferencia segura de datos en tiempo real.
- RTP: (Protocolo de Transporte en Tiempo Real) Es un protocolo de nivel de aplicación utilizado para la transmisión de información en tiempo real, como por ejemplo audio y video en una video-conferencia.
- RTCP: (Protocolo de Control de Transporte en Tiempo Real) Provee control sobre el flujo de datos RTP. La principal función de RTCP es suministrar información sobre la calidad de servicio provisto por RTP.

3.3 Procedimientos para establecer una llamada

3.3.1 Direcciones

3.3.1.1 Dirección de red

Es la dirección de capa de red de una entidad H.323 definida por el protocolo de capa de red (entre redes) en uso (por ejemplo, una dirección IP). Cada una de las entidades H.323 deberá tener por lo menos una dirección de red. Dicha dirección identifica de manera exclusiva la entidad H.323 en la red.

3.3.1.2 Identificador de punto de acceso al servicio de capa de transporte (Identificador TSAP)

El punto de acceso al servicio de transporte (TSAP) es el punto donde una entidad de un nivel superior al de transporte accede al servicio de transporte provisto por una entidad del nivel de transporte. Un identificador de TSAP proporciona una identificación correspondiente a un protocolo de interconexión de redes (por ejemplo, dirección IP) particular y una identificación correspondiente a los protocolos de transporte conexos (por ejemplo, números de puerto de TCP y UDP). Este identificador tiene la garantía de ser único en el contexto del protocolo correspondiente. Los identificadores TSAP permiten la multiplexación de varios canales que comparten la misma dirección de red. Es utilizado para multiplexar varias conexiones de transporte del mismo tipo en una sola entidad H.323 con todas las conexiones de transporte que comparten la misma dirección de red. Los identificadores TSAP pueden ser asignados estáticamente por alguna autoridad internacional o bien ser asignados dinámicamente durante el establecimiento de una comunicación. Por cada dirección de red, cada una de las entidades H.323 puede tener varios identificadores TSAP [9].

3.3.1.3 Dirección alias

Un punto extremo puede tener también una o más direcciones alias asociadas al mismo. Una dirección alias puede representar el punto extremo o puede representar conferencias que el punto extremo está acogiendo. Las direcciones alias proporcionan un método alternativo de direccionamiento del punto extremo. Dichas direcciones incluyen direcciones de dígitos marcados (dialedDigits) o de números de parte (partyNumber), incluyendo números telefónicos privados y números E.164 públicos, identidades H.323 (cadenas alfanuméricas que representan nombres, direcciones similares a las del correo electrónico, etc.) y cualesquiera otras direcciones definidas en la Rec. UIT-T H.225.0. Las direcciones alias deberán ser

únicas dentro de una zona. Los controladores de acceso, los MC y los MP no tendrán direcciones alias.

3.3.2 Canal de registro, admisiones y situación (Canal RAS)

El canal de RAS se empleará para transportar mensajes utilizados en los procesos de descubrimiento de guardián de puerta y de registro de punto extremo que asocian una dirección alias de punto extremo con su dirección de transporte de canal de señalización de llamada. En el establecimiento y finalización de una llamada, se usa el canal RAS para enviar mensajes de admisión y de desligamiento. El canal de RAS deberá ser un canal no fiable. Como los mensajes RAS se transmiten por un canal no fiable, H.225.0 recomienda plazos y cuentas de reintento para diversos mensajes [9].

3.3.2.1 Mensajes de admisión de terminal a guardián de puerta

En una llamada en la que interviene un guardián de puerta, antes entrar en la señalización de la llamada, un punto extremo debe solicitar acceso al guardián de puerta a través de un mensaje de petición de admisión (ARQ, Admision Request). El mensaje ARQ solicita que a un punto extremo le sea permitido el acceso a la red de paquetes por el guardián de puerta, el cual concede la petición con un mensaje de confirmación de admisión (ACF, Admision Confirm) o la deniega con un mensaje de rechazo de admisión (ARJ, Admision Reject).

3.3.2.2 Mensajes de desligamiento

Al finalizar una llamada, cada punto extremo transmitirá un mensaje de petición de desligamiento (DRQ, disengage request) a su guardián de puerta. El guardián de puerta responderá con un mensaje de confirmación de desligamiento (DCF, disengage confirm). Si se envía de un terminal a un guardián de puerta, el

DRQ (petición de desligamiento) informa al guardián de puerta que un punto extremo está siendo abandonado. Si se envía de un guardián de puerta a un punto extremo, el DRQ obliga a una llamada a ser abandonada; dicha petición no será rehusada. El DRQ no se envía directamente entre puntos extremos. El mensaje rechazo de desligamiento DRJ (disengage reject) es enviado por el guardián de puerta si el punto extremo no está registrado.

3.3.3 Canal de señalización de llamada

El canal de señalización de llamada se empleará para transportar mensajes de control de llamada H.225. El canal de señalización será un canal fiable. En redes que no disponen de un guardián de puerta, los mensajes de señalización de llamada se pasan directamente entre los puntos extremos llamante y llamado utilizando las direcciones de transporte de señalización de llamada. En dichas redes, se supone que el punto extremo llamante conoce la dirección de transporte de señalización de llamada del punto extremo llamado y, por tanto, puede comunicarse directamente.

En las redes que disponen de un guardián de puerta, el intercambio de mensajes de admisión inicial tiene lugar entre el punto extremo llamante y el guardián de puerta utilizando la dirección de transporte de canal de RAS del guardián de puerta. Durante el intercambio de mensajes de admisión inicial, el guardián de puerta indica en el mensaje ACF si la señalización de llamada se envía directamente al otro punto extremo o se encamina a través del guardián de puerta. Los mensajes de señalización de llamada se envían bien a la dirección de transporte de señalización de llamada del punto extremo o bien a la dirección de transporte de señalización de llamada del guardián de puerta. El canal de señalización de llamada puede transportar diversas llamadas concurrentes, utilizando el valor de referencia de llamada para asociar el mensaje con la llamada. La Recomendación UIT-T Rec. H.225 especifica los mensajes Q.931 obligatorios que se utilizan para señalización de llamada en H.323.

Los mensajes de señalización de llamada se pueden transferir según dos métodos. El primero de ellos es el de señalización de llamada encaminada por el guardián de puerta (ver figura 3). En este método, los mensajes de señalización de llamada se encaminan a través del guardián de puerta entre los puntos extremos. El segundo método es el de señalización de llamada de puntos extremos directa (ver figura 4). En este método, los mensajes de señalización de llamada se pasan directamente entre los puntos extremos. La elección del método a utilizar corre a cargo del guardián de puerta. Ambos métodos utilizan las mismas clases de conexiones para los mismos fines y los mismos mensajes. Los mensajes de admisión son intercambiados en canales RAS con el guardián de puerta, seguidos de un intercambio de mensajes de señalización de llamada en un canal de señalización de llamada. Esto a su vez va seguido del establecimiento del canal de control H.245. Las acciones del guardián de puerta en respuesta a los mensajes de admisión determinan qué modelo de llamada se utiliza, lo cual no está sometido al control del punto extremo, aunque el punto extremo puede especificar una preferencia.

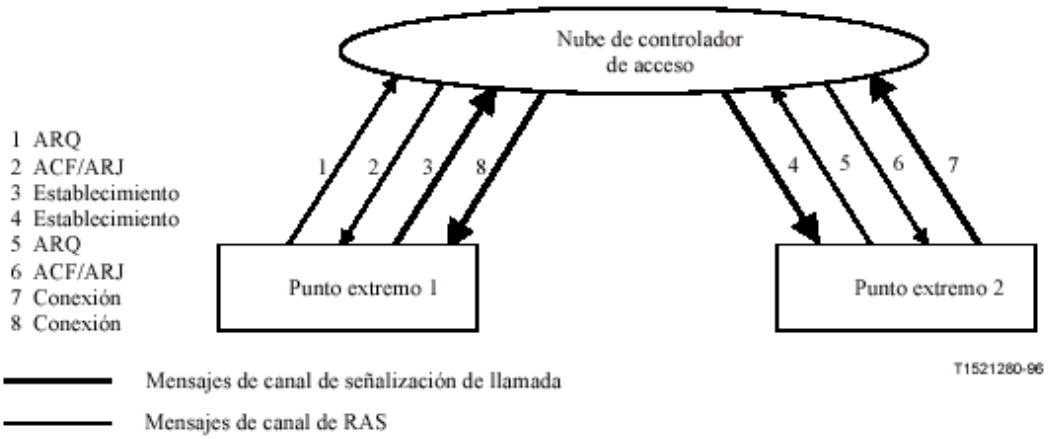


Figura 3. Señalización de llamada encaminada por guardián de puerta [9]

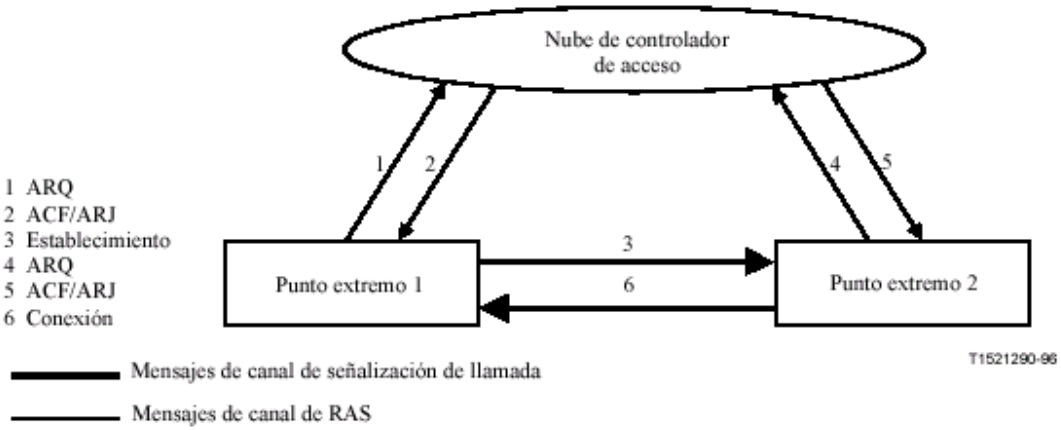


Figura 4. Señalización de llamada de punto extremo directa [9]

Cuando se utiliza la señalización de llamada encaminada por guardián de puerta, se dispone de dos métodos para encaminar el canal de control H.245. En el primero de ellos, el canal de control H.245 se establece directamente entre los puntos extremos (figura 5). Este método queda en estudio. En el segundo método, el canal de control H.245 es encaminado entre los puntos extremos a través del guardián de puerta (figura 6). Este método permite al guardián de puerta reencaminar el canal de control H.245 a un MC cuando una conferencia pasa de conferencia punto a punto a conferencia multipunto. El guardián de puerta realiza esta elección. Cuando se utiliza la señalización de llamada de punto extremo directa, el canal de control H.245 sólo puede ser conectado directamente entre los puntos extremos.

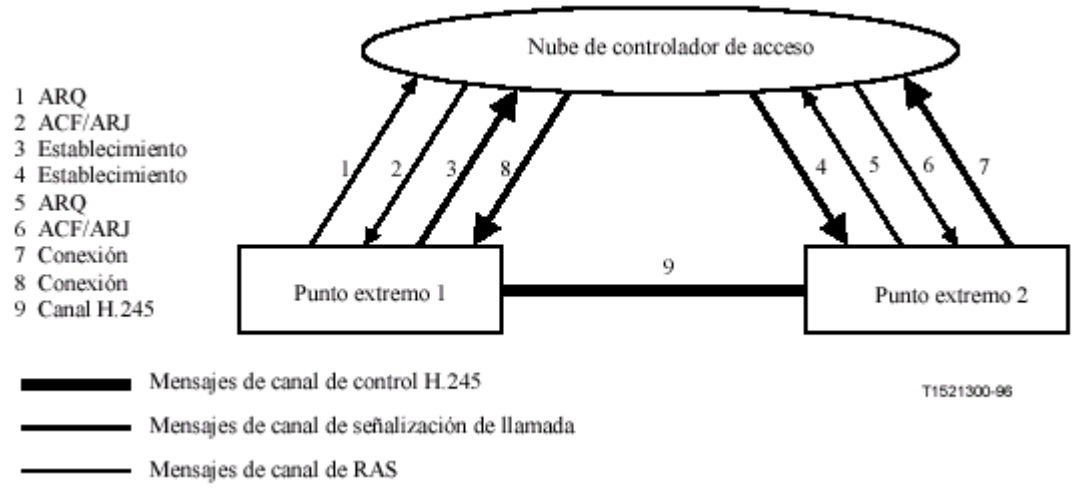


Figura 5. Conexión de canal de control H.245 directa entre puntos extremos [9]

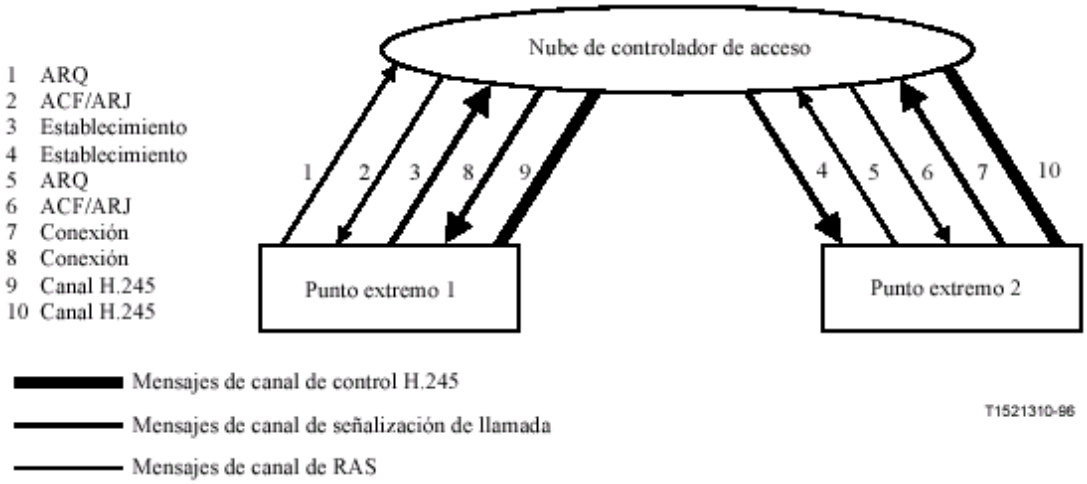


Figura 6. Control H.245 encaminado por guardián de puerta [9]

3.3.4 Valor de referencia de llamada

Todos los mensajes de señalización de llamada y RAS contienen un valor de referencia de llamada (CRV, call reference value). Hay un CRV para el canal de señalización de llamada y un CRV independiente para el canal RAS. Un CRV se utiliza para asociar los mensajes de señalización de llamada. Este CRV se utilizará en todos los mensajes de señalización de llamada entre dos entidades (punto extremo a

guardián de puerta, punto extremo a punto extremo, etc.) relacionadas con la misma llamada. Un segundo CRV se utiliza para asociar los mensajes RAS. Este CRV se utilizará en todos los mensajes RAS entre dos entidades relacionadas con la misma llamada. Se utilizarán nuevos CRV para nuevas llamadas. Una segunda llamada procedente de un punto extremo para invitar a otro punto extremo a la misma conferencia utilizará nuevos CRV. El CRV no es lo mismo que el ID de llamada o el ID de conferencia (CID, conference ID). El CRV asocia mensajes de señalización de llamada o RAS entre dos entidades dentro de la misma llamada, el ID de llamada asocia todos los mensajes entre todas las entidades dentro de la misma llamada, y el CID asocia todos los mensajes entre todas las entidades dentro de todas las llamadas en la misma conferencia. La referencia de llamada global, con el valor numérico 0, se utiliza para hacer referencia a todas las llamadas en el canal de señalización de llamada o en el canal RAS [9].

3.3.5 Identificador de llamada (ID)

El identificador de llamada es un valor distinto de cero globalmente único creado por el punto extremo llamante y que se intercambia en varios mensajes H.225.0. El ID de llamada identifica la llamada con la que está asociado el mensaje. Se utiliza para asociar todos los mensajes RAS y de señalización de llamada relacionados con la misma llamada. A diferencia del CRV, el ID de llamada no cambia dentro de una llamada. Todos los mensajes del punto extremo llamante a su guardián de puerta, del punto extremo llamante al punto extremo llamado, y del punto extremo llamado a su guardián de puerta relativos a la misma llamada contendrán el mismo ID de llamada.

3.3.6 Identificador de conferencia (CID)

El ID de conferencia (CID) es un valor único distinto de cero creado por el punto extremo llamante y que se intercambia en diversos mensajes H.225.0. El CID

identifica la conferencia con la cual está asociado el mensaje. Por tanto, los mensajes procedentes de todos los puntos extremos dentro de la misma conferencia tendrán el mismo CID.

3.4 Procedimientos de señalización de la llamada

La provisión de la comunicación se efectúa siguiendo las siguientes fases:

- Fase A: Establecimiento de la comunicación
- Fase B: Comunicación inicial e intercambio de capacidad
- Fase C: Establecimiento de comunicación audiovisual
- Fase D: Servicios de la llamada
- Fase E: Terminación de la llamada

3.4.1 Fase A: Establecimiento de la comunicación

El establecimiento de la comunicación se efectúa utilizando los mensajes de control de llamada definidos en la Rec. UIT-T H.225.0. La recomendación H.323 establece la secuencia de mensajes RAS H.225 y señalización de llamada H.225 para varios casos posibles.

El caso más simple es el establecimiento de llamada básica en la que ninguno de los puntos extremos está registrado en algún guardián de puerta (figura 7). Los dos puntos extremos comunican directamente. El punto extremo 1 (punto extremo llamante) envía el mensaje establecimiento (1) al identificador TSAP de canal de señalización de llamada conocido del punto extremo 2. Este mensaje es enviado por una entidad H.323 llamante para indicar su deseo de establecer una conexión hacia la entidad llamada. Luego el usuario llamado envía el mensaje llamada en curso (2) para indicar que se ha iniciado el establecimiento de llamada solicitado y que no se aceptará ninguna información más de establecimiento de

llamada. Seguidamente, el mensaje aviso (3) es enviado por el usuario llamado para indicar que se ha iniciado el aviso del usuario llamado, en términos de hoy día, "el teléfono está sonando". Finalmente, el punto extremo 2 responde con el mensaje conexión (4) para indicar aceptación de la llamada por la entidad llamada. Este mensaje contiene una dirección de transporte de canal de control H.245 para su utilización en la señalización H.245.

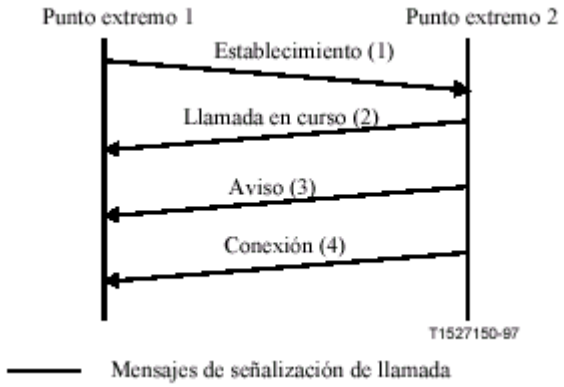


Figura 7. Establecimiento de llamada básica, sin controladores de acceso [9]

En el caso en que ambos puntos extremos están registrados en el mismo guardián de puerta y el guardián de puerta ha optado por encaminar la señalización de la llamada (figura 8), el punto extremo 1 (punto extremo llamante) inicia el intercambio ARQ (1)/ACF (2) con el guardián de puerta. El guardián de puerta devolverá una dirección de transporte de canal de señalización de llamada de él mismo en la ACF. El punto extremo 1 envía entonces el mensaje establecimiento (3) utilizando esa dirección de transporte. El guardián de puerta envía a continuación el mensaje establecimiento (4) al punto extremo 2. Si el punto extremo 2 desea aceptar la llamada, inicia un intercambio ARQ (6)/ACF (7) con el guardián de puerta. Es posible que el punto extremo 2 reciba un ARJ (7), en cuyo caso envía el mensaje liberación completa al guardián de puerta. El punto extremo 2 responde con el mensaje conexión (9) que contiene una dirección de transporte de canal de control H.245 para su utilización en la señalización H.245. El guardián de puerta envía al

punto extremo 1 el mensaje conexión (10) que puede contener la dirección de transporte de canal de control H.245 del punto extremo 2 o una dirección de transporte de canal de control H.245 de guardián de puerta, dependiendo de si el guardián de puerta decide encaminar o no el canal de control H.245.

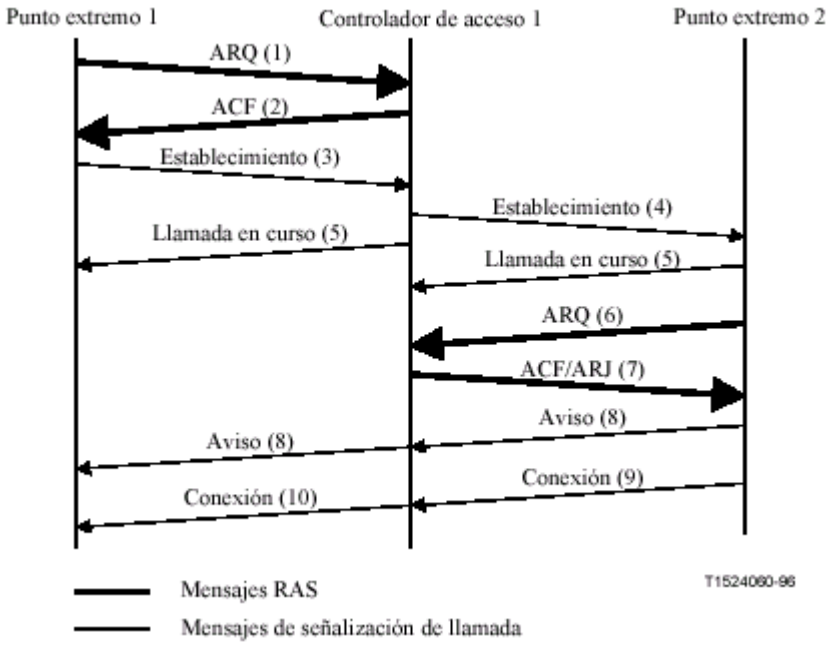


Figura 8. Ambos puntos extremos registrados, el mismo guardián de puerta. Señalización de llamada encaminada por el guardián de puerta [9].

La recomendación H.323 describe el proceso de establecimiento de la comunicación para otros casos posibles.

3.4.2 Fase B: Comunicación inicial e intercambio de capacidad

Una vez que ambos lados han intercambiado los mensajes de establecimiento de llamada de la fase A, los puntos extremos, si proyectan emplear H.245, establecerán el canal de control H.245. Se utilizan los procedimientos de la Rec. UIT-T H.245 en el canal de control H.245 para el intercambio de capacidad y la apertura de canales de medios.

Para conservar recursos, sincronizar la señalización y control de llamada y reducir el tiempo de establecimiento de la llamada, puede ser conveniente transmitir mensajes H.245 dentro del canal de señalización de llamada Q.931 en lugar de establecer un canal H.245 separado. Este proceso, conocido como "encapsulado" o "tunelización" de mensajes H.245, se efectúa copiando un mensaje H.245 codificado como una cadena de octetos. Cuando la tunelización está activa, pueden encapsularse uno o más mensajes H.245 en cualquier mensaje Q.931. Si se está utilizando la tunelización y no es necesaria la transmisión de un mensaje Q.931 en el instante en que debe transmitirse un mensaje H.245, se enviará entonces un mensaje facilidad con motivo (reason) colocado en información transportada (transportedInformation) [9].

3.4.3 Fase C: Establecimiento de comunicación audiovisual

Después del intercambio de capacidades y la determinación de principal-subordinado, se utilizarán los procedimientos de la Rec. UIT-T H.245 para abrir canales lógicos para los diversos trenes de información. Los trenes de audio y vídeo, que se transmiten por los canales lógicos establecidos en la Rec. H.245, se transportan en identificadores TSAP dinámicos utilizando un protocolo no fiable. Las comunicaciones de datos, que se transmiten por los canales lógicos establecidos en H.245, se transportan utilizando un protocolo fiable.

3.4.4 Fase D: Servicios de la llamada

Una vez abiertos los canales lógicos para los diversos trenes de información, los puntos extremos o el guardián de puerta pueden invocar ciertos servicios, como por ejemplo: cambios de anchura de banda, ampliación a conferencia y servicios suplementarios.

3.4.5 Fase E: Terminación de la llamada

La terminación de la llamada se realizará mediante el procedimiento siguiente:

- Interrumpir la transmisión de vídeo al final de una imagen completa y a continuación cerrar todos los canales lógicos de vídeo.
- Interrumpir la transmisión de datos y a continuación cerrar todos los canales lógicos de datos.
- Interrumpir la transmisión de audio y a continuación cerrar todos los canales lógicos de audio.
- Transmitir el mensaje instrucción finalizar sesión (endSessionCommand) H.245 por el canal de control H.245, indicando al extremo distante que desea desconectarse de la llamada, interrumpiendo entonces la transmisión de mensajes H.245.
- Esperar recibir el mensaje endSessionCommand del otro punto extremo y cerrar el canal de control H.245.
- Si el canal de señalización de llamada está abierto, se enviará un mensaje liberación completa y se cerrará el canal de señalización.
- En las redes que no contienen un guardián de puerta, después de los pasos anteriores se termina la llamada, no se requiere ninguna acción ulterior. En las redes que contienen un guardián de puerta, no es preciso que el guardián de puerta esté al corriente de la liberación de anchura de banda. Después de ejecutar los pasos anteriores, cada punto extremo transmitirá un mensaje de petición de desligamiento (DRQ, disengage request) H.225.0 a su guardián de puerta. El guardián de puerta responderá con un mensaje de confirmación de desligamiento (DCF, disengage confirm). Después de enviar el mensaje DRQ, los puntos extremos no enviarán más mensajes IRR no solicitados al guardián de puerta. La recomendación H.323 define otros casos. En este punto la llamada está terminada.

CAPÍTULO IV

FORMATO DE MENSAJES Q.931 [10] Y RESTRICCIONES DEL PROTOCOLO H.225 [7]

Los mensajes Q.931 son utilizados en el establecimiento de una llamada H.323. El protocolo Q.931 describe el formato general de estos mensajes y la codificación de los elementos de información que están contenidos dentro de estos mensajes. El protocolo Q.931 fue creado para redes ISDN y su título es "Especificación de la capa 3 de la interfaz usuario-red de la red digital de servicios integrados para el control de la llamada básica". Por lo tanto, el protocolo H.225 impone restricciones a estos mensajes y a los elementos de información para ser usados en H.323. En la siguiente tabla se muestra el formato de un mensaje Q.931.

<table>
<tr><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>Bit/Octeto</th></tr>
<tr><td colspan="8">Discriminador de protocolo</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="4">Longitud del valor de la referencia de llamada (en octetos)</td><td>2</td></tr>
<tr><td colspan="8">Valor de la referencia de llamada</td><td>3</td></tr>
<tr><td>0</td><td colspan="7">Tipo de mensaje</td><td>4</td></tr>
<tr><td colspan="8">Otros elementos de información, según se requieran</td><td>etc.</td></tr>
</table>

Tabla 1. Formato de un mensaje Q.931 [10].

Los elementos de información discriminador de protocolo, longitud del valor de la referencia de llamada, valor de la referencia de llamada y tipo de mensaje son comunes a todos los mensajes y están siempre presentes, mientras que los otros elementos de información son específicos de cada mensaje.

4.1 Discriminador de protocolo

El discriminador de protocolo es la primera parte de cada mensaje. La finalidad del discriminador de protocolo es distinguir mensajes para el control de la llamada usuario-red de otros mensajes. En mensajes Q.931 debe contener la secuencia de bits “0 0 0 0 1 0 0 0” (8 en base 10).

4.2 Valor de la referencia de llamada

La finalidad de la referencia de llamada es identificar a qué llamada se aplica un mensaje particular. Se codifica como muestra la tabla 2. La longitud del valor de la referencia de llamada se indica en el octeto 1, bits 1 a 4. La longitud máxima por defecto del elemento de información referencia de llamada es tres octetos.

<table>
<tr><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>Bit/Octeto</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="4">Longitud del valor de la referencia
de llamada (en octetos)</td><td>1</td></tr>
<tr><td>Bandera</td><td colspan="7"></td><td>2</td></tr>
<tr><td colspan="8">Valor de la referencia de llamada</td><td>etc.</td></tr>
</table>

Tabla 2. Elemento de información Referencia de Llamada [10].

El elemento de información referencia de llamada incluye el valor y la bandera de la referencia de llamada. Los valores de la referencia de llamada se asignan, para una llamada, en el punto extremo que origina la llamada. Estos valores son únicos solamente para el lado origen en una llamada determinada. El valor de la referencia de llamada se asigna al comienzo de una llamada y permanece fijo

mientras dura la llamada. Cuando termina una llamada, el valor de la referencia de llamada asociado puede reasignarse a otra llamada. La bandera de la referencia de llamada puede tomar les valores “0” ó “1”. La bandera de la referencia de llamada se utiliza para identificar el extremo del enlace lógico que ha originado la llamada. El lado de origen pone siempre la bandera de la referencia de llamada a “0”. El lado de destino pone siempre la bandera de la referencia de llamada a “1”. Por consiguiente, la bandera de la referencia de llamada identifica quien asignó el valor de la referencia de llamada para esta llamada, y su única finalidad es resolver las tentativas simultáneas de asignar un mismo valor de referencia de llamada.

4.3 Tipo de mensaje

La finalidad del tipo de mensaje es identificar la función del mensaje que se envía. Es la tercera parte de cada mensaje y se codifica como se muestra en la tabla 3. El bit 8 se reserva para posible uso futuro como bit de ampliación.

8	7	6	5	4	3	2	1	Bit/Octeto
0	Tipo de mensaje							1

Tabla 3. Elemento de información Tipo de mensaje. [10]

No todos los tipos de mensajes definidos en Q.931 son usados en H.323. El protocolo H.225 indica cuales tipos de mensajes se usan en H.323. Algunos de estos tipos de mensaje se muestran en la siguiente tabla.

Bits	Tipo de mensaje
8 7 6 5 4 3 2 1	
0 0 0 - - - - -	Mensaje de establecimiento de la llamada:
0 0 0 0 1	– AVISO
0 0 0 1 0	– LLAMADA EN CURSO
0 0 1 1 1	– CONEXIÓN
0 0 1 0 1	– ESTABLECIMIENTO
0 1 1 0 1	– ACUSE DE ESTABLECIMIENTO
0 1 0 - - - - -	Mensajes de liberación de llamada:
1 1 0 1 0	– LIBERACIÓN COMPLETA

Tabla 4. Tipos de mensajes.

4.4 Otros elementos de información

A parte de los elementos de información comunes, Q.931 define otros que, dependiendo del tipo de mensaje, pueden aparecer o no en el mensaje. Los elementos de información se clasifican de la siguiente forma:

- Elementos de información de un solo octeto: Son elementos de información con una longitud de un Octeto (ver tablas 5 y 6). El bit más significativo es igual a 1. Existen dos tipos de elementos de información de un solo octeto:
    - Tipo 1: Los elementos de tipo 1 proporcionan una identificación de elemento de información en las posiciones de bit 7, 6, 5 tal como muestra la tabla 5. Estos bits deben ser diferentes a la secuencia “010” que identifica al tipo 2. Los bits restantes poseen el contenido del elemento de información.
    - Tipo 2: Se identifican por el valor "010" en las posiciones de bit 7, 6, 5, tal como muestra la tabla 6.
- Elementos de información de longitud variable: Son elementos de información cuyo contenido puede contener más de un octeto. Por lo general su estructura es igual a la que se muestra en la tabla 7. El bit más significativo del primer octeto es igual a 0.

8	7	6	5	4	3	2	1	Bit/Octeto
1	Identificador del elemento de información			Contenido del elemento de información				1

Tabla 5. Elemento de información de un solo octeto tipo 1 [10].

8	7	6	5	4	3	2	1	Bit/Octeto
1	Identificador del elemento de información							1

Tabla 6. Elemento de información de un solo octeto tipo 2 [10].

8	7	6	5	4	3	2	1	Bit/Octeto
0	Identificador del elemento de información							1
Longitud del contenido del elemento de información (octetos)								2
Contenido del elemento de información								3 etc.

Tabla 7. Elemento de información de longitud variable [10]

.

No todos los elementos de información definidos en el protocolo Q.931 son usados en H.323. El protocolo H.225 enumera cuales elementos de información deben ser usados en H.323. En la tabla 8 se muestran algunos de ellos con sus respectivos identificadores. Los elementos de información de un solo octeto pueden aparecer en cualquier posición del mensaje. Sin embargo, existe un orden particular de aparición de cada elemento de información de longitud variable en un mensaje. Los valores de código del identificador de elemento de información para los formatos de longitud variable se asignan en orden numérico ascendente, de acuerdo con el orden real de aparición de cada elemento de información en un mensaje. Por ejemplo, un elemento de información visualización siempre debe aparecer después del elemento de información capacidad de portador. Esto permite al equipo receptor

detectar la presencia o ausencia de un elemento de información particular sin explorar todo el mensaje.

Identificador	Elemento de Información
Bits 8 7 6 5 4 3 2 1	
1 : : : - - - -	Elementos de información de un solo octeto:
0 1 0 0 0 0 1	Envío completo
0 : : : : : : :	Elementos de información de longitud variable:
0 0 0 0 1 0 0	Capacidad portadora
0 0 0 1 0 0 0	Causa
0 1 0 1 0 0 0	Visualización
1 1 1 1 1 1 0	Usuario a usuario

Tabla 8. Elementos de información.

El segundo octeto de un elemento de información de longitud variable indica la longitud total del contenido de ese elemento de información empezando en el octeto 3, es decir, no toma en cuenta el identificador no ni el propio octeto longitud. En el elemento de información usuario a usuario, el protocolo H.225 impone la restricción de utilizar dos octetos para la longitud en vez de uno. La estructura del contenido depende del propio elemento de información.

En algunos casos se aplica un mecanismo de agrupación y ampliación en el contenido de los elementos de información de longitud variable. En este mecanismo la primera cifra del número de octeto identifica a un octeto o a un grupo de octetos (ver tabla 9). Cada grupo de octetos es una entidad autocontenida. Un grupo de octetos se forma utilizando un mecanismo de ampliación. El mecanismo de ampliación consiste en ampliar un octeto (N) en el octeto o los octetos siguientes (Na, Nb, etc.) utilizando el bit 8 de cada octeto como bit de ampliación. El valor "0" de este bit indica que el grupo continúa en el octeto siguiente. El valor "1" de este bit indica que ese octeto es el último del grupo.

8	7 6 5 4 3 2 1	Bit/Octeto
ext. 0		N
ext. 0		Na
ext. 0		Nb
ext. 0		Nc
ext. 1		Nd
ext. 1		N+1
ext. 1		N+2

Tabla 9. Mecanismo de agrupación y ampliación.

4.5 Mensaje Establecimiento

Este mensaje será enviado por una entidad H.323 llamante para indicar su deseo de establecer una conexión hacia la entidad llamada. El siguiente cuadro indica los elementos de información que pueden o deben estar presentes en el mensaje establecimiento. La letra M significa obligatorio (mandatory), O opcional, CM condicionalmente obligatorio (conditionally mandatory). Las subdirecciones de las partes llamantes y llamadas se necesitan en algunos casos de llamadas con una red con conmutación de circuitos; no deberían utilizarse para llamadas sólo dentro de la red de paquetes.

Elemento de información	**Situación H.225.0**
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M

Elemento de información	Situación H.225.0
Envío completo	O
Capacidad portadora	M
Facilidad ampliada	O
Facilidad	O
Indicador de notificación	O
Visualización	O
Facilidad de teclado	O
Señal	O
Número de la parte llamante	O
Subdirección de la parte llamante	CM
Número de la parte llamada	O
Subdirección de la parte llamada	CM
Usuario a usuario	M

Tabla 10. Mensaje establecimiento.

4.6 Mensaje Llamada en curso

Este mensaje puede ser enviado por el usuario llamado para indicar que se ha iniciado el establecimiento de llamada solicitado y que no se aceptará ninguna información más de establecimiento de llamada. El siguiente cuadro indica los elementos de información que pueden o deben estar presentes en este mensaje.

Elemento de información	Situación H.225.0
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M
Capacidad portadora	O
Facilidad ampliada	O
Facilidad	O
Indicador de progresión	O
Indicador de notificación	O
Visualización	O

Elemento de información	Situación H.225.0
Usuario a usuario	M

Tabla 11. Mensaje llamada en curso.

4.7 Mensaje Aviso

Este mensaje puede ser enviado por el usuario llamado para indicar que se ha iniciado el aviso del usuario llamado. En términos de hoy día, "el teléfono está sonando". El siguiente cuadro indica los elementos de información que pueden o deben estar presentes en este mensaje.

Elemento de información	Situación H.225.0
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M
Capacidad portadora	O
Facilidad ampliada	O
Facilidad	O
Indicador de progresión	O
Indicador de notificación	O
Visualización	O
Señal	O
Usuario a usuario	M

Tabla 12. Mensaje Aviso.

4.8 Mensaje Conexión

Este mensaje será enviado por la entidad llamada a la entidad llamante (guardián de puerta, pasarela o terminal llamante) para indicar aceptación de la llamada por la entidad llamada. El siguiente cuadro indica los elementos de información que pueden o deben estar presentes en este mensaje.

Elemento de información	Situación H.225.0
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M
Capacidad portadora	O
Facilidad ampliada	O
Facilidad	O
Indicador de progresión	O
Indicador de notificación	O
Visualización	O
Fecha/hora	O
Usuario a usuario	M

Tabla 13. Mensaje Conexión.

4.9 Mensaje Liberación Completa

Este mensaje será enviado por un terminal para indicar liberación de la llamada si el canal de señalización de llamada está abierto. Después, el valor de referencia de llamada está disponible para su reutilización. El siguiente cuadro indica los elementos de información que pueden o deben estar presentes en este mensaje.

Elemento de información	Situación H.225.0
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M
Causa	CM
Facilidad	O
Indicador de notificación	O
Visualización	O
Señal	O
Usuario a usuario	M

Tabla 14. Mensaje Liberación Completa.

El elemento de información Causa estará presente si no se envía el elemento ReleaseCompleteReason dentro del contenido dentro del elemento de información usuario a usuario.

4.10 Elemento de información capacidad portadora

El elemento de información capacidad portadora tiene por objeto especificar el servicio solicitado. Se codifica como muestra la siguiente tabla.

8	7	6	5	4	3	2	1	Bit/Octeto
	Identificador del elemento de información capacidad portadora							
0	0	0	0	0	1	0	0	1
Longitud del contenido de capacidad portadora								2
ext. 1	Norma de codificación		Capacidad de transferencia de información					3
ext. 1	Modo de transferencia		Velocidad de transferencia de información					4
ext. 1	Multiplicador de velocidad							4.1*
ext. 0/1	Identificador de capa 1		Protocolo de capa 1 de información del usuario					5*
	0	1						
ext. 0/1	Sinc./ asínc	Negoc.	Velocidad de usuario					5a*
ext. 0/1	Velocidad intermedia		NIC en Tx	NIC en Rx	Control flujo en Tx	Control flujo en Rx	Reserva 0	5b*
ext. 0/1	Encabeza-miento/no encabeza-miento	Soporte de multitrama	Modo	Negoc. LLI	Asignador/ asignado	Negoc. dentro/ fuera de banda	Reserva 0	5b*
ext. 0/1	Número de bits de parada		Número de bits de datos		Paridad			5c*
ext. 1	Modo dúplex	Tipo de módem						5d*
ext. 1	Identificador de capa 2		Protocolo de capa 2 de información del usuario					6*
	1	0						
ext. 0	Identificador de capa 3		Protocolo de capa 3 de información del usuario					7*
	1	1						
ext. 0	Reserva			Información adicional de protocolo de capa 3 (bits más significativos)				7a*
	0	0	0					
ext. 1	Reserva			Información adicional de protocolo de capa 3 (bits más significativos)				7b*
	0	0	0					

Tabla 15. Elemento de información capacidad portadora [10].

El protocolo H.225 impone las siguientes restricciones:

- Capacidad de transferencia de información (octeto N.º 3):
  - El bit de extensión (bit 8) se pondrá a '1'.
  - La norma de codificación (bits 6, 7) se pondrá a '00' indicando 'UIT-T'.
  - Capacidad de transferencia de información (bits 0-5): Las llamadas que se originan en un punto extremo H.323 utilizarán este campo para indicar su deseo de efectuar una llamada audiovisual. Por tanto, el campo se pondrá a 'información digital sin restricciones', es decir, '01000' o a 'información digital restringida' es decir '01001'. Si ha de efectuarse una llamada sólo vocal, el terminal H.323 pondrá la capacidad de transferencia de información a 'conversación' (es decir '00000') o a 'audio a 3,1 kHz' (es decir '10000').
- Bit de extensión para el octeto N.º 4 (bit 8): Se pondrá a '0' si la velocidad de transferencia de información se pone a 'multivelocidad'; se pondrá a '1' en otro caso.
- Modo de transferencia – octeto abreviado N.º 4 (bits 6, 7): Especificará 'modo circuito', valor '00'.
- Velocidad de transferencia de información (bits 5 a 1): Se codificará siguiendo el cuadro 4-6/Q.931, salvo que el valor '00000' (para el modo paquete) no se permite a menos que la pasarela se conecte a una red de paquetes. Las opciones son las siguientes:

Bits 5 4 3 2 1	
0 0 0 0 0	Este código se utilizará para llamadas en modo paquete.
1 0 0 0 0	64 kbit/s
1 0 0 0 1	2 × 64 kbit/s
1 0 0 1 1	384 kbit/s
1 0 1 0 1	1536 kbit/s

1 0 1 1 1 1920 kbit/s

1 1 0 0 0 Multivelocidad (velocidad básica de 64 kbit/s)

Los demás valores están reservados.

- Multiplicador de velocidad – octeto N.º 4.1: Estará presente si la velocidad de transferencia de información se pone a 'multivelocidad.
- Protocolo de capa 1(capa 1 de ISDN) – octeto N.º 5
  - El bit de extensión (bit 8) se pondrá a '1'.
  - Los bits 6 y 7 indicarán el identificador de capa 1 (capa 1 de ISDN), es decir, '01'.
  - Los bits 1 a 5 indicarán el protocolo de capa 1.
  - Los valores permitidos son G.711 (ley A '00011' y ley μ '00010') para indicar una llamada sólo voz y H.221 y H.242 ('00101') para indicar una llamada videotelefónica H.323.
  - Los octetos N.º 5a, 5b, 5c, 5d no estarán presentes.
- Identificador de protocolo de capa 2 (capa 2 de ISDN) – octeto N.º 6: No estará presente.
- Identificador de protocolo de capa 3 (capa 3 de ISDN) – octeto N.º 7: No estará presente.

4.11 Elemento de información Causa

Indica la razón por la cual una llamada fue rechazada o desconectada. El contenido y el uso del elemento de información causa se definen en la Recomendación Q.850. La siguiente tabla muestra su codificación.

8	7	6	5	4	3	2	1	Bit/Octeto
	Identificador del elemento de información causa							
0	0	0	0	1	0	0	0	1
	Longitud del contenido de causa							2
0/1 ext	Norma de codificación		0 Reservado	Ubicación				3
1 ext	Recomendación							3a
1 ext	Valor Causa							4
Diagnostico (si existe)								5

Tabla 16. Elemento de información Causa.

- Norma de Codificación - octeto N° 3 (bits 7,6)

  Bits

  7 6

  0 0 Codificación estandarizada CCITT

  0 1 ISO/IEC

  1 0 Estándard Nacional

  1 1 Norma específica de la ubicación identificada

- Ubicación - octeto N° 3 (bits 4 a 1)

  Bits

  4 3 2 1

  0 0 0 0 Usuario

  0 0 0 1 Red privada que da servicio al usuario local

  0 0 1 0 Red pública que da servicio al usuario local

  0 0 1 1 Red de tránsito

  0 1 0 0 Red pública que da servicio al usuario distante

  0 1 0 1 Red privada que da servicio al usuario distante

0 1 1 1 Red internacional

1 0 1 0 Red que se extiende más allá del punto de interfuncionamiento

Los demás valores están reservados

- Recomendación - octeto N° 3a

  Bits

  7 6 5 4 3 2 1

  0 0 0 0 0 0 0 Q.931

  0 0 0 0 0 1 1 X.21

  0 0 0 0 1 0 0 X.25

  0 0 0 0 1 0 1 Redes públicas móviles terrestres Q.1031/Q.1051

  Si este octeto es omitido se asume la recomendación Q.931.

- Valor Causa - octeto N° 4: Este valor indica la causa de la terminación de la llamada. Estos son algunos de sus posibles valores

  Liberación normal de la llamada

  Usuario ocupado

  No hay respuesta del usuario

  No hay respuesta del usuario

  Abonado ausente

  Llamada rechazada

  Red fuera de servicio

  Fallo temporal

  Capacidad portadora no implementada

  Contenido de elemento de información no válido

- Diagnostico - octeto N° 5: Dependiendo de la causa, se puede incluir un octeto de diagnostico en donde se proporcionan detalles de la causa de la terminación de la llamada.

4.12 Elemento de información Visualización

La finalidad del elemento de información visualización es suministrar información que puede ser visualizada por el usuario. La información contenida en este elemento se codifica en caracteres del Alfabeto Internacional N.° 5. El elemento de información visualización se codifica como se muestra en la tabla 17.

8	7	6	5	4	3	2	1	Bit/Octeto
0	Identificador del elemento de información visualización 0	1	0	1	0	0	0	1
Longitud del contenido de visualización								2
0	Información de visualización (caracteres del IA5)							3 etc.

Tabla 17. Elemento de información Visualización.

4.13 Elemento de información Usuario a usuario

El elemento de información usuario a usuario será utilizado por todas las entidades H.323 para transportar información relacionada con H.323 decodificada en PER de la sintaxis ASN.1. El elemento de información usuario a usuario se codifica como se muestra en la tabla 18.

8	7	6	5	4	3	2	1	Bit/Octeto
	Identificador del elemento de información usuario a usuario							
0	1	1	1	1	1	1	0	1
Longitud del contenido de usuario a usuario								2
Discriminador de protocolo								3
Información de usuario								4 etc.

Tabla 18. Elemento de información Usuario a usuario.

El protocolo H.225 impone las siguientes restricciones:

- Longitud de contenido de usuario a usuario: Serán 2 octetos en vez de 1.
- Discriminador de protocolo: Indicará información de usuario codificada ('00000101') X.208 y X.209 (ASN.1). (Esto se toma de la revisión 1993 de la Recomendación Q.931, que hace referencia a las anteriores revisiones de ASN.1. Las referencias correctas a ASN.1 son la Recomendación X.680 (sintaxis) y la Recomendación X.691 (PER)).
- Información de usuario: Contendrá una estructura ASN.1 que, además de la información pertinente H.323, incluya los datos de usuario efectivos.

CAPÍTULO V

PROTOCOLO DE INICIACIÓN DE SESIONES (SIP) [6]

El Protocolo de Iniciación de Sesiones es un protocolo de control a nivel de aplicación que permite la creación, modificación y terminación de sesiones multimedia con uno o más participantes. Estas sesiones incluyen llamadas telefónicas por Internet, distribuciones multimedia y conferencia multimedia.

Fue desarrollado por el Grupo de Trabajo en Ingeniería de Internet (IETF). Por lo tanto, SIP se caracteriza porque sus promotores tienen sus raíces en la comunidad IP y no en la industria de las telecomunicaciones. El primer borrador de SIP apareció en febrero de 1996 y el segundo en diciembre de ese mismo año. La primera versión SIP propuesta como estándar fue definida en la recomendación RFC2543 (SIP/1.0) y fue publicada en febrero de 1999. En Junio de 2002 la versión RFC2543 fue aclarada por la versión RFC3261 (SIP/2.0). En Noviembre del año 2000, SIP fue aceptado como el protocolo de señalización de 3GPP (Proyecto Sociedad de 3° Generación) y elemento permanente de la arquitectura IMS (Subsistema Multimedia IP) de IBM.

SIP no es un sistema de comunicaciones integrado verticalmente. SIP es más bien un componente que puede ser usado con otros protocolos para construir una arquitectura multimedia completa. Generalmente, esta arquitectura incluye protocolos tales como el Protocolo de Transporte en Tiempo Real (RTP) para transportar datos en tiempo real y proveer información de calidad de servicio, el Protocolo de Flujo en Tiempo Real (RTSP) para controlar la entrega de flujo multimedia, el Protocolo de Control de Pasarela Multimedia (MEGACO) para controlar las pasarelas hacia la Red Publica de Telefonía Conmutada (PSTN) y el Protocolo de Descripción de Sesiones (SDP) para describir las sesiones multimedia. Por lo tanto, SIP debería ser usado en

conjunto con otros protocolos para proveer servicios completos al usuario. Sin embargo, la funcionalidad básica y operación de SIP no depende de ninguno de estos protocolos. SIP no proporciona servicios. Más bien SIP provee bases que pueden ser usadas para implementar diferentes servicios.

5.1 Definiciones

A continuación se mencionan algunos términos que se definen en SIP.

- Sesión: Una sesión multimedia es un conjunto de emisores y receptores multimedia y los flujos de datos que fluyen de emisores a receptores.
- Mensaje: Es la data enviada entre elementos SIP como parte del protocolo. Los mensajes SIP son solicitudes o respuestas.
- Solicitud: Es un mensaje SIP enviado desde un cliente a un servidor con el propósito de invocar una operación particular.
- Método: Es la función principal que una solicitud desea invocar en un servidor. El método es transportado en propio el mensaje de solicitud. Ejemplos de método son “INVITE” y “BYE”.
- Respuesta: Es un mensaje SIP enviado desde un servidor hacia un cliente para indicar el estado de una solicitud enviada desde el cliente hacia el servidor.
- Cliente: Es un elemento de red que envía solicitudes SIP y recibe respuestas SIP. Los clientes pueden o no interactuar directamente con un usuario humano. Los agentes de usuario clientes y servidores proxy son ejemplos de clientes.
- Servidor: Es un elemento de red que recibe solicitudes para prestar algún servicio y envía de vuelta respuestas a esas solicitudes. Ejemplos de servidores son servidores proxy, agente de usuarios servidores, servidores de redirección y servidores de registro.
- Agente de usuario cliente (UAC): Es una entidad lógica que crea y envía una nueva solicitud. El papel de UAC permanece sólo en la duración de la

transacción. En otras palabras, si un componente de software inicia una solicitud, actúa como UAC en la duración de la transacción. Si más tarde recibe una solicitud, asume el rol de agente de usuario.

- Agente de usuario servidor (UAS): Es una entidad lógica que genera una respuesta a una solicitud SIP. La respuesta acepta, rechaza, o redirecciona la solicitud. Este rol permanece sólo en la duración de la transacción. En otras palabras, si un componente de software responde una solicitud, actúa como un UAS mientras dure la transacción. Si más tarde genera una solicitud, asume el rol de agente usuario cliente.
- Agente de usuario (UA): Es una entidad lógica que puede actuar tanto como un agente de usuario cliente y como un agente de usuario servidor.
- Transacción: Una transacción ocurre entre un cliente y un servidor y comprende todos los mensajes desde la primera solicitud enviada desde el cliente al servidor, hasta una respuesta final enviada del servidor al cliente. Si la solicitud es "INVITE" y la respuesta final no es una respuesta 2xx, la transacción también incluye una solicitud "ACK" a esa respuesta. La solicitud ACK para una respuesta 2xx a una solicitud INVITE es una transacción separada.
- Respuesta final: Es una respuesta que termina una transacción al contrario de las respuestas provisionales. Todas las respuestas 2xx, 3xx, 4xx, 5xx y 6xx son respuestas finales.
- Respuesta provisional: Es una respuesta usada por el servidor para indicar progreso, pero que no termina la transacción. Las respuestas 1xx son provisionales, las otras son consideradas finales.
- Diálogo: Es una relación entre dos agentes de usuario que persiste por algún tiempo. Un diálogo es establecido por un mensaje SIP, tal como una respuesta 2xx a una solicitud "INVITE". Un diálogo es identificado por un identificador de llamada, una etiqueta local y una etiqueta remota.
- Servidor proxy: Es una entidad intermedia que actúa tanto como servidor como cliente con el propósito de hacer solicitudes en nombre de otros clientes.

Un servidor proxy principalmente juega el papel de enrutador, lo cual significa que su trabajo es asegurar que una solicitud sea enviada a una entidad más cercana al usuario destino. Los servidores proxy también son útiles para imponer políticas (por ejemplo, asegurar que un usuario tenga permitido hacer llamadas). Un servidor proxy interpreta, y, si es necesario, rescribe partes específicas de un mensaje de solicitud antes de pasarlo.

- Identificador uniforme de recursos (URI): Es una cadena compacta de caracteres usada para identificar un recurso abstracto o físico.
- Servidor de registro: Es un servidor que acepta solicitudes REGISTER y coloca la información recibida en esas solicitudes en el servicio de localización del dominio que maneja.

5.2 Indicador uniforme de recursos (URI)

Un URI SIP o SIPS identifica un recurso de comunicación. Un URI SIPS específica que los recursos deben ser contactados de forma segura. Esto significa que se debe usar comunicación segura para alcanzar al usuario. El mecanismo especifico de seguridad depende de la política del dominio. Los esquemas “sip:” y “sips:” se rigen por el estándar RFC 2396. Estos esquemas usan una forma similar al “mailto” URL. La forma general de un URI SIP es:

sip:usuario:contraseña@host:puerto;parámetros-uri?cabeceras

El formato de un URI SIPS es el mismo excepto que usa “sips” en vez de “sip”. Estos elementos tienen los siguientes significados:

- usuario: Es el identificador de un recurso particular en el host que está siendo direccionado. En este contexto el término “host” frecuentemente se refiere a un dominio. La información de usuario de un URI consiste de este campo usuario, el campo contraseña y el símbolo @. La información de usuario de

un URI es opcional y puede estar ausente cuando el host destino no tiene una noción de usuario o cuando el host en sí mismo es el recurso que está siendo identificado.

- contraseña: Es una contraseña asociada con el usuario. Su uso dentro de un URI no es recomendado por razones de seguridad.
- host: Es el host que provee el recurso SIP. Contiene un nombre de dominio o una dirección numérica Ipv4 o Ipv6.
- puerto: Es el número del puerto donde la solicitud será enviada.
- parámetros-uri: Son parámetros que afectan una solicitud construida desde el URI. Estos parámetros son separados por punto y coma. Los parámetros URI toman la forma nombre-de-parámetro = valor-del-parámetro.
- cabeceras: Campos de cabecera que serán incluidos en una solicitud construida desde el URI.

El uso de los componentes de un URI depende del contexto. La siguiente tabla resume el uso de los componentes de los URI SIP y SIPS basados en el contexto en donde aparecen. La letra "M" significa obligatorio, la "O" opcional, el símbolo "-" significa no permitido y "--" indica que el elemento no tiene valor por defecto. La segunda columna indica los valores por defecto si un elemento opcional no está presente.

Comp. de un URI	**Valor por defecto**	**URI-de-Solicitud**	**"To"**	**"From"**	**"Contact" en Registro o Redireccón**	**"Contact","Record-Route","Route" en Diálogos**
usuario	--	O	O	O	O	O
contraseña	--	O	O	O	O	O
host	--	M	M	M	M	M
puerto	5060 para	O	-	-	O	O

	sip: y 5061 para sips:					
parámetros-de-usuarios	IP	O	O	O	O	O
cabeceras	--	-	-	-	O	-

Tabla 19. Uso y valores por defecto de los componentes de un URI [6].

5.3 Mensajes SIP

SIP es un protocolo basado en texto y usa el conjunto de caracteres UTF-8. Un mensaje SIP es una solicitud desde un cliente a un servidor o una respuesta desde un servidor a un cliente. Tanto las solicitudes como las respuestas usan el formato básico RFC 2822 (Formato de Mensaje de Internet). Ambos tipos consisten de una línea de inicio, una o más campos de cabecera, una línea vacía indicando el fin de los campos de cabecera y un cuerpo de mensaje opcional:

Línea de inicio
Cabecera del mensaje
Línea Vacía
Cuerpo del mensaje (opcional)

Tabla 20. Estructura de un mensaje SIP.

La línea vacía debe estar presente incluso si no existe el cuerpo de mensaje. La línea de inicio puede ser una línea de solicitud o una línea de estado dependiendo de si el mensaje es una solicitud o una respuesta.

Excepto por la diferencia en el conjunto de caracteres, la sintaxis de muchos de los mensajes y campos de cabeceras SIP es idéntica a la de HTTP/1.1. Por lo tanto, en el protocolo SIP se hacen muchas referencias al estándar HTTP/1.1 (RFC 2616). Sin embargo, SIP no es una extensión de HTTP.

5.3.1 Solicitudes

Las solicitudes SIP se distinguen por tener una Línea-de-Solicitud en la línea de inicio. Una Línea-de-Solicitud contiene un nombre de método, un URI-de-solicitud, y la versión del protocolo separados por un espacio simple.

Línea-de-Solicitud = Método URI-de-Solicitud Versión-SIP

- Método: Es la función principal que una solicitud desea invocar en un servidor. La especificación RFC 3261 define seis métodos:
  - "REGISTER" para registro de información de contacto.
  - "INVITE" para iniciar una sesión.
  - "ACK" para indicar el reconocimiento de recepción de una respuesta.
  - "CANCEL" para cancelar sesiones.
  - "BYE" para terminar sesiones.
  - "OPTIONS" para servicios de indagación de capacidades.
- URI-de-Solicitud: Indica el usuario o servicio al cual esta solicitud esta siendo dirigida.
- Versión-SIP: Tanto los mensajes de solicitud como los de respuesta incluyen la versión de SIP en uso. Para ser consistente con el estándar RFC 3261, las aplicaciones que envíen mensajes deben incluir "SIP/2.0" en Versión-SIP.

5.3.2 Respuestas

Las respuestas SIP se diferencian de solicitudes por tener una Línea-de-Estado como su línea de inicio. Una Línea-de-Estado consiste de la versión del protocolo seguido por un número Código-de-Estado y su frase textual asociada, con cada elemento separado por un espacio simple.

Línea-de-Estado = Versión-SIP Código-de-Estado Frase

- Código-de-Estado: Es un código entero de 3 dígitos que indica el resultado de un intento de entender y satisfacer una solicitud. El primer dígito del Código-de-Estado define la clase de respuesta. Cualquier respuesta con un código de estado entre 100 y 199 es calificada como una respuesta 1xx, una respuesta con código de estado entre 200 y 299 como una respuesta 2xx, y así sucesivamente. SIP/2.0 permite seis valores para el primer dígito:
    - 1xx – Provisional: Solicitud recibida, se continúa a procesar la solicitud.
    - 2xx – Satisfactorio: La acción fue recibida, entendida y aceptada satisfactoriamente.
    - 3xx – Redirección: Otras acciones deben ser tomadas para completar la solicitud.
    - 4xx – Error del cliente: La solicitud contiene mala sintaxis o no puede ser ejecutada en este servidor.
    - 5xx – Error del servidor: El servidor falló en ejecutar una solicitud aparentemente válida.
    - 6xx – Falla general: La solicitud no puede ser ejecutada por ningún servidor.
- Frase: Tiene la intención de dar una breve descripción textual al Código-de-Estado. La Frase está destinada para el usuario humano. Un cliente no está obligado a examinar o mostrarla. A pesar de que SIP sugiere valores

específicos de Frase, las implementaciones pueden escoger otros textos, por ejemplo, en el algún idioma en particular.

En la siguiente tabla se muestran las respuestas SIP con sus códigos y descripciones correspondientes.

Clase	Código	Descripción
1xx (Provisional)	100	Intentando
	180	Repicando
	181	La llamada está siendo transferida
	182	En cola
	183	Progreso de la sesión
2xx (Satisfactorio)	200	Satisfactorio (OK)
3xx (Redirección)	300	Opciones múltiples
	301	Movido permanentemente
	302	Movido temporalmente
	305	Usar servidor proxy
	380	Servicios alternativos
4xx (Error del cliente)	400	Mala solicitud
	401	Desautorizado
	402	Pago requerido
	403	Olvidado

	404	No encontrado
	405	Método no permitido
	406	No aceptable
	407	Autenticación requerida por el servidor proxy
	408	Tiempo de expiración de solicitud
	410	Ausente
	413	Entidad de la solicitud muy larga
	414	URI-de-la-Solicitud muy largo
	415	Tipo de medio no soportado
	416	Esquema URI no soportado
	420	Mala extensión
	421	Extensión requerida
	423	Intervalo muy breve
	480	Temporalmente indisponible
	481	Llamada o transacción no existe
	482	Lazo detectado
	483	Demasiados saltos
	484	Dirección incompleta
	485	Ambiguo
	486	Ocupado aquí
	487	Solicitud terminada

	488	No aceptable aquí
	491	Solicitud pendiente
	493	Indescifrable
5xx (Error del servidor)	500	Error interno del servidor
	501	No implementado
	502	Mala pasarela
	503	Servicio indisponible
	504	Tiempo de expiración del servidor
	505	Versión no soportada
	513	Mensaje muy largo
6xx (Falla global)	600	Ocupado en todas partes
	603	Declinar
	604	No existe en ninguna parte
	606	No aceptable

Tabla 21. Respuestas SIP

5.3.3 Campos de cabecera

Cada campo de cabecera consiste de un nombre del campo seguido por dos puntos (":") y el valor del campo:

Nombre-del-campo: valor-del-campo

El orden relativo de los campos de cabecera con diferentes nombres de campo no es significativo. Sin embargo, es recomendado que los campos de cabecera que son necesitados en el procesamiento de los servidores proxy (por ejemplo: Via, Route, Record-Route, Proxy-Require, Max-Forwards y Proxy-Authorization) aparezcan en el tope superior del mensaje facilitando su rápido análisis. El orden relativo de las filas de campos de cabecera con el mismo nombre de campo si es importante.

El formato de valor-del-campo es definido por nombre-del-campo. Muchos campos de cabecera existentes adherirán a la forma general un valor seguido por una secuencia de parámetros. Un parámetro consisten en un nombre y un valor separados por punto y coma:

nombre-del-campo: valor-del-campo *(;nombre-del-parámetro=valor-del-parámetro)

Aunque se puede adjuntar un número arbitrario de parámetros, un nombre de parámetro no debe aparecer más de una vez

5.3.4 Cuerpo del mensaje

Las mensajes pueden contener cuerpos de mensaje. La interpretación del cuerpo depende del método solicitado. El tipo de medio Internet del cuerpo del mensaje puede ser especificado por el campo de cabecera “Content-Type”. Si el cuerpo está sujeto a alguna codificación, tal como alguna compresión, entonces esto debe ser indicado por el campo de cabecera “Content-Encoding”, en otro caso, “Content-Encoding” debe ser omitido. La longitud del cuerpo en Octetos es provista por el campo de cabecera Content-Length [6].

5.4 Comportamiento general de los agentes de usuario

Los procedimientos de los UAC (agentes de usuario clientes) y los UAS (agentes de usuario servidores) dependen principalmente de dos factores. Primero, depende de si la solicitud o respuesta está o no dentro del diálogo, y segundo, depende del método de la solicitud. En esta sección se describe las reglas independientes del método que rigen el procesamiento por parte de los UAC y UAS de las solicitudes que están fuera del diálogo. Esto incluye, por supuesto, las solicitudes que establecen el diálogo.

5.4.1 Comportamiento de los agentes de usuario clientes

Ejemplos de solicitudes enviadas fuera de diálogo incluyen una solicitud "INVITE" para establecer una sesión y una solicitud "OPTIONS" para indagar capacidades.

5.4.1.1 Generación de solicitudes

Una solicitud SIP válida formulada por un UAC debe, como mínimo, contener los siguientes campos de cabecera: "To", "From", "CSeq", "Call-ID", "Max-Forwards", y "Via". Todos estos campos de cabecera son obligatorios en todas las solicitudes SIP y son los bloques fundamentales de un mensaje SIP. A continuación se describen estos campos de cabecera junto al URI-de-Solicitud de la línea de solicitud.

- URI-de-Solicitud: La URI-de-Solicitud inicial debería ser el valor de URI en el campo To. Una de las excepciones a esta regla es el método REGISTER.
- "To": El campo de cabecera "To" específica el recipiente lógico deseado de la solicitud, o la dirección de registro del usuario o el recurso destino de la solicitud. El campo de cabecera "To" contiene un URI y permite un nombre

para visualización. Una solicitud fuera de diálogo no debe contener un parámetro “tag” (etiqueta) para el campo “To”. El parámetro “tag” en el campo “To” de una solicitud es uno de los identificadores del diálogo. Puesto que un diálogo no ha sido establecido, no existe una etiqueta.

- “From”: Indica la identidad lógica del iniciador de la solicitud, posiblemente la dirección de registro del usuario. Al igual que el campo “To”, contiene un URI y opcionalmente un nombre para visualización. El campo de cabecera “From” puede contener un nuevo parámetro “tag” (etiqueta) elegido por el UAC.
- “Call-ID”: Actúa como un único identificador para agrupar una serie de mensajes. Debe ser el mismo para todos las solicitudes y respuestas enviadas por cualquier UA en un diálogo. En una nueva solicitud creada por un UAC fuera de diálogo, el campo “Call-ID” debe ser seleccionado por el UAC como un identificador globalmente único. Todos los UA deben tener un medio para garantizar que el campo “Call-ID” que ellos producen no será inadvertidamente generado por otro UA. Cuando una solicitud es reenviada después de alguna repuesta de falla, esa solicitud reenviada no es considerada una nueva solicitud, y por lo tanto, no necesita un nuevo “Call-ID”. Por otra parte, debería ser el mismo en cada registro de un UA.
- “CSeq”: Sirve para identificar y ordenar transacciones. Contiene un número de secuencia y un método. Para solicitudes fuera de diálogo que no sean de registro, el valor del número de secuencia es arbitrario. El método debe concordar con el de la solicitud. El valor del número de secuencia debe ser expresable como un entero sin signo de 32 bits y debe ser menor que 2^{31}.
- “Max-Forwards”: Sirve para limitar el número de saltos que una solicitud puede transitar en el camino a su destino. Consiste en un entero que es disminuido en uno en cada salto. Si el valor de “Max-Forwards” llega a ser 0 antes que la solicitud alcance su destino, la solicitud será rechazada con una respuesta de error 483 (Demasiados Saltos). Un UAC debe insertar un campo “Max-Forwards” en cada solicitud que él origina con un valor que debería ser

70. Este número fue escogido lo suficientemente grande para garantizar que una solicitud no sea borrada en cualquier red SIP sin lazos, pero no tan grande como para consumir los recursos de los servidores proxy cuando ocurre un lazo.

- Via: Indica el transporte usado para la transacción e identifica la ubicación donde la respuesta debe ser enviada. Cuando un UAC crea una solicitud debe insertar un campo "Via". Contiene el nombre y versión del protocolo que deben ser SIP y 2.0 respectivamente. El campo "Via" también debe contener un parámetro llamado "branch". Este parámetro es usado para identificar la transacción creada por la solicitud y también es usado por los servidores proxy para detectar lazos. El parámetro "branch" debe ser único para todas las solicitudes enviadas por el UA. La excepción a esta regla son la solicitud "CANCEL" y la solicitud "ACK" para una respuesta no 2xx. Una solicitud "CANCEL" tendrá el mismo valor del parámetro "branch" que el de la solicitud que está cancelando. Una solicitud "ACK" para una respuesta no 2xx también tendrá el mismo parámetro "branch" que la solicitud INVITE cuya respuesta reconoce. El identificador "branch" insertado por un elemento que obedezca la especificación RFC 3261 siempre debe comenzar con los caracteres "z9hG4bK".
- Contact: Provee un URI que puede ser usado para contactar esa instancia específica del UA para solicitudes subsiguientes. El campo "Contact" debe estar presente en una solicitud que puede resultar en el establecimiento de un diálogo y contener exactamente un URI.
- Componentes de mensaje adicionales: Después que una solicitud ha sido creada y los campos de cabecera descritos anteriormente han sido construidos correctamente, cualquier campo de cabecera opcional puede ser añadidos, como son los campos de cabecera específicos del método.

5.4.1.2 Procesamiento de respuestas

La mayor parte del procesamiento de respuesta es específica del método. Sin embargo, existen algunos comportamientos generales independientes del método que se describen en la especificación RFC 3261. Por ejemplo, si en la respuesta está presente más de un campo "Via", el UAC debería descartar el mensaje. La presencia de un campo "Via" adicional que precede el creador de la solicitud sugiere que el mensaje fue sacado de su ruta o que posiblemente fue alterado. Si el UAC recibe una respuesta 401 (Desautorizado) o 407 (Autenticación requerida por el servidor proxy), el UAC debería reintentar la solicitud pero ahora con la información de autenticación. Para ello, se debe añadir el campo "Proxy-Authenticate", el cual contiene información de autenticación. En este caso, la respuesta se envía nuevamente constituyendo una nueva transacción y debería tener el mismo valor de "Call-ID", "To" y "From" de la solicitud previa, pero el valor de "CSeq" debería contener un nuevo número de secuencia que igual al valor de la solicitud anterior incrementado en uno.

5.4.2 Comportamiento de los agentes de usuario servidores

Cuando una solicitud fuera de diálogo es recibida por un UAS, hay un conjunto de normas de procedimientos que se deben seguir, independientemente del método. Una vez que una solicitud es autenticada, el UAS debe inspeccionar el método de la solicitud, y revisar si lo soporta o no y, en caso de que no lo soporte, generar una respuesta 405 (Método no permitido). Luego el UAS debe inspeccionar la cabecera. Si el UAS no entiende un campo de cabecera en una solicitud, debe ignorar ese campo de cabecera y continuar procesando el mensaje. Un UAS debe ignorar cualquier campo de cabecera defectuoso que no sea necesario para procesar el mensaje. Seguidamente, el UAS debe examinar el cuerpo del mensaje. Si existe algún cuerpo cuyo tipo o codificación sea desconocido y si el cuerpo del mensaje no es opcional, el UAS debe rechazar la solicitud con una respuesta 415 (Tipo de medio no

soportado). Asumiendo que todas las revisiones anteriores fueron satisfactorias, el UAS debe ejecutar el procesamiento específico del método. Una vez procesada la solicitud, el UAS procede a formular la respuesta.

5.4.2.1 Generación de respuestas

Un UAS no debería enviar una respuesta provisional para una solicitud que no sea “INVITE”. Al contrario, un UAS debería generar lo más pronto posible una respuesta final a una solicitud que no sea “INVITE”. Los campos “From”, “Call-ID” y “Cseq” de la respuesta deben ser iguales a los de la solicitud. Los valores del campo “Via” de la respuesta deben ser iguales a los de la solicitud y mantener el mismo orden.

Si una solicitud contenía una etiqueta (“tag”) en el campo “To”, el campo “To” en la respuesta debe ser igual al de la solicitud. Sin embargo, si el campo “To” en la solicitud no contiene una etiqueta, el URI en el campo “To” en la respuesta debe ser igual al URI en el campo “To” de la solicitud; adicionalmente, el UAS debe añadir una etiqueta al campo “To” en la respuesta (con la excepción de la respuesta 100 (“Trying”), en la cual una etiqueta puede estar presente o no). Esto sirve para identificar el UAS que está respondiendo, resultando posiblemente en un componente de un identificador de diálogo. La misma etiqueta debe ser usada para todas las respuestas a esa solicitud, tanto final como provisional (otra vez exceptuando la respuesta 100 (“Trying”)).

5.5 Registro

Si un usuario quiere iniciar una sesión con otro usuario, SIP debe descubrir la ubicación actual en el que se encuentra el destino. Este proceso de descubrimiento es frecuentemente realizado por elementos de red SIP tales como servidores proxy y servidores de redirección los cuales son responsables de recibir una solicitud,

determinar donde enviarla en base al conocimiento de localización del usuario y enviarla allí. Para hacer esto, los elementos de red SIP consultan un servicio abstracto de localización, el cual provee registros de direcciones para un dominio particular. Hay muchas formas por las que el contenido del servicio de localización pueda ser establecido. Una forma es administrativamente, en donde se conoce a priori los miembros que pertenecen al dominio. Sin embargo, SIP provee un mecanismo para crear estos registros de forma explicita. Este mecanismo es conocido como registro.

El registro implica el envío de una solicitud REGISTER a un tipo especial de UAS conocido como servidor de registro. Un servidor de registro es un UAS que responde a solicitudes REGISTER y mantiene una lista de registros que es accesible a los servidores proxy y a los servidores de redirección dentro de su dominio administrativo. Un servidor de registro lee y escribe mapas basado en el contenido de las solicitudes REGISTER. Este servicio de localización es frecuentemente consultado por un servidor proxy que es responsable del enrutamiento de solicitudes para ese dominio. Los registros y servidores proxy son funciones lógicas que pueden ser desempeñadas por un único dispositivo en la red.

Las solicitudes REGISTER añaden, remueven y consultan registros. Una solicitud de registro no establece un diálogo. Una solicitud de registro puede añadir una nueva registro entre una dirección de registro y uno o más direcciones de contacto. El registro en nombre de una dirección de registro particular puede ser realizado por una tercera parte debidamente autorizada. Un cliente también puede remover registros previas o consultarlas para determinar cuales registros están actualmente en curso para una dirección de registro. Los siguientes campos de cabecera, excepto “Contact”, deben ser incluidos en una solicitud “REGISTER”. Un campo “Contact” puede ser incluido.

- “Request-URI”: Indica el nombre del dominio del servicio de localización al cual se refiere el registro. Los componentes “informacióndeusuario” y “@” del URI no deben estar presente (por ejemplo “sip:dominio.com).
- “To”: Contiene la dirección de registro cuyo registro, valga la redundancia, va a ser creado, consultado o modificado. El campo “To” y el campo “Request-URI” difieren típicamente en que el primero contiene el nombre de usuario (por ejemplo: “sip:usuario@dominio.com”).
- “From”: Contiene la dirección de registro de la persona responsable del registro. El valor es el mismo que el del campo “To” al menos que sea una solicitud de registro a través de una tercera persona.
- “Call-ID”: Todos los registros desde un UAC deberían usar el mismo valor “Call-ID” para los registros enviados a un servidor de registro en particular. Si el mismo cliente usara diferentes valores de “Call-ID”, un servidor de registro no podría detectar si una solicitud de registro retrasada pueda haber llegado fuera de orden (ver “CSeq” a continuación).
- “CSeq”: El valor “CSeq” garantiza el correcto ordenamiento de las solicitudes “REGISTER”. Un UA debe incrementar en uno el valor “CSeq” por cada solicitud de registro con el mismo “Call-ID”.
- “Contact”: Las solicitudes pueden contener un campo “Contact” con cero o más valores que contienen direcciones de registros. Los UA no deben enviar un nuevo registro (esto es, uno que contiene nuevos valores de “Contact”, lo que es diferente a una retransmisión) hasta que haya recibido una respuesta final desde el servidor de registro o hasta que la solicitud “REGISTER” previa haya agotado su tiempo de espera. El parámetro “expires” indica cuanto tiempo el UA desea que la registro sea válida. El valor es un número que indica segundos. Si este valor no es suministrado, se usa en cambio el valor de la cabecera “Expires”.

Si el registro es satisfactorio, el servidor de registro debe enviar una respuesta 200 (“OK”). Esta respuesta debe contener valores del campo “Contact”

enumerando todas los registros actuales. Cada contacto debe tener un parámetro "expires" indicando su intervalo de expiración escogido por el servidor de registro. Los registros expirarán al menos que sean refrescados, pero pueden ser removidos explícitamente. Un UA solicita la remoción inmediata de una registro especificando un intervalo de expiración de "0" para la dirección de contacto en la solicitud "REGISTER". Cada UA es responsable de refrescar los registros que ha establecido previamente. Un UA no debería refrescar registros establecidas por otros UA. Para refrescar cada una de sus registros, el UA envía una solicitud REGISTER por cada uno de sus registros antes de que su intervalo de expiración haya culminado.

5.6 Diálogos

Un concepto clave para los agentes de usuario es el diálogo. Un diálogo representa una relación SIP punto a punto entre dos agentes de usuario que prevalece por algún tiempo. El diálogo facilita la secuencia de mensajes entre los agentes de usuario y el correcto enrutamiento entre ellos. Representa un contexto para interpretar mensajes SIP. El diálogo es identificado dentro de cada UA con una identificador (ID) de diálogo, el cual consiste de un valor "Call-ID", una etiqueta local y una etiqueta remota. El identificador de diálogo en cada UA involucrado en el diálogo no es el mismo. Específicamente, la etiqueta local en un UA es idéntica a la etiqueta remota en el otro UA y viceversa. Las etiquetas facilitan la generación de un identificador de diálogo único.

Las reglas de establecimiento del ID de diálogo de un mensaje dependen de si el elemento SIP es un UAC o un UAS. Para un UAC, el valor "Call-ID" del diálogo es el "Call-ID" del mensaje, la etiqueta remota es la etiqueta ("tag") en el campo "To" del mensaje y la etiqueta local es la del campo "From". Para un UAS, el valor "Call-ID" del ID del diálogo es el "Call-ID" del mensaje, la etiqueta remota es la etiqueta ("tag") del campo "From" del mensaje y la etiqueta local es la etiqueta ("tag") del campo "To".

Un diálogo contiene ciertas piezas de estado necesarias para la transmisión dentro del diálogo. Este estado consiste del ID del diálogo, un número de secuencia local (usado para ordenar solicitudes desde el UA al UA opuesto), una número de secuencia remoto (usado para ordenar solicitudes desde el UA opuesto al UA), un URI local, un URI remoto, destino remoto, una bandera booleana llamada "secure" y un conjunto de ruta, el cual es una lista ordenad de URIs. El conjunto de ruta es la lista de servidores que necesitan ser atravesados para enviar una solicitud al UA opuesto. Todos estos parámetros son usados para construir solicitudes y analizar respuestas dentro del diálogo. Un diálogo puede estar también en el estado "temprano", el cual ocurre cuando es creado con una respuesta provisional, y luego es transformado a un estado "confirmado" cuando una llega respuesta final 2xx. Si en un diálogo temprano, se recibe otras respuestas, o si no se recibe ninguna respuesta, el dialogo temprano termina.

Los diálogos son creados a través de la generación de respuestas que no sean de fallos para solicitudes con métodos específicos. En la especificación RFC 3261, sólo las respuestas 2xx y las respuestas 101-199 con una etiqueta en el campo "To", donde la solicitud fue "INVITE", establecerán un diálogo. Una vez que el UAS responde una solicitud con una respuesta que establece un diálogo, el UAS construye el estado del diálogo. En el UAC, el estado se construye cuando se recibe esta respuesta. Este estado debe ser mantenido durante la duración del diálogo. En este estado se almacenan una serie de información que será utilizada en la construcción y manejos de solicitudes y respuestas que pertenezcan al diálogo. Un ejemplo de la información que se almacena es el conjunto de ruta especificado en la cabecera "Record-Route" de la solicitud, la cual es la lista de servidores que necesitan ser atravesados para enviar una solicitud al UA opuesto y que debe ser almacenado en el estado del diálogo.

Cuando un UAS responde a una solicitud con una respuesta que establece un diálogo (tal como una respuesta 2xx a una solicitud “INVITE”), el UAS debe copiar todos los campos Record-Route de la solicitud dentro de la respuesta y manteniendo el mismo orden. Debe añadir una campo “Contact” a la respuesta. El campo “Contact” contiene una dirección donde el UAS le gustaría ser contactado en subsiguientes solicitudes dentro del diálogo (lo cual incluye la solicitud “ACK” para una respuesta 2xx en el caso de la solicitud “INVITE”).

Seguidamente el UAS construye el estado del diálogo. Este estado debe ser mantenido durante la duración del diálogo. El conjunto de ruta es construido con los campos Record-Route manteniendo el mismo orden. El destino remoto es puesto igual al URI del campo “Contact” de la solicitud. El numero de secuencia remota se coloca igual a al valor del campo “CSeq” de la solicitud. El número de secuencia local debe estar vacío. El valor de “Call-ID” de la solicitud será el identificador de llamada del ID del diálogo. La etiqueta local del ID del diálogo debe ser igual a la etiqueta “tag” del campo “To” en la solicitud, y la etiqueta remota debe ser igual a la etiqueta “tag” del campo “From”. El URI remoto debe ser puesto igual al URI del campo “From”, y el URI local igual al URI en el campo “To”.

Por otra parte, cuando el UAC recibe una respuesta que establece un diálogo (tal como una respuesta 2xx a una solicitud “INVITE”), construye el estado del diálogo, el cual debe ser mantenido mientras dure el diálogo. El conjunto de ruta debe ser construido a partir de los campos Record-Route de la respuesta pero en orden inverso. El destino remoto es puesto igual al URI del campo “Contact” de la solicitud. El número de secuencia local debe ser puesto igual al valor de “CSeq” de la solicitud. El número de secuencia remota debe estar vacío (éste es establecido cuando el UA remoto envíe una solicitud dentro del diálogo). El componente identificador de llamada del ID del diálogo es colocado igual al valor “Call-ID” de la solicitud. La etiqueta local debe ser igual a la etiqueta “tag” del campo “From” en la solicitud, mientras que la etiqueta remota debe ser igual a la etiqueta “tag” del campo “To” en

la respuesta. El URI remoto debe ser puesto igual al URI del campo “To” y el URI local igual al URI del campo “From”.

Una solicitud dentro de un diálogo es construida usando muchos de los componentes del estado del diálogo almacenado. El URI en el campo “To” de la solicitud debe ser el URI remoto del diálogo. La etiqueta “tag” en el campo “To” debe ser la etiqueta remota del diálogo. El URI del campo “From” debe ser el URI local del estado del diálogo. La etiqueta “tag” en el campo “From” debe ser la etiqueta local del diálogo. El valor “Call-ID” de la solicitud debe ser el “Call-ID” del diálogo. Las solicitudes dentro del diálogo contienen números “CSeq” que deben ser incrementados en uno (excepto en las solicitudes “ACK” y “CANCEL”, que contienen un “CSeq” igual a la solicitud siendo reconocida o cancelada). Por lo tanto, si el número de secuencia local no está vacío, debe ser incrementado en uno y este valor debe ser colocado en el campo “CSeq” de la solicitud. Si está vacío, el valor inicial de “CSeq” debe ser creado usando las indicaciones de la especificación RFC 3261. El método del campo “Cseq” debe ser el mismo método de la solicitud. El UAC usa el destino remoto y el conjunto de ruta para construir el URI-de-la-Solicitud y los campos “Route” de la solicitud, respectivamente. Un AUC debería incluir el mismo campo “Contact” usado en las solicitudes anteriores. El resto de la solicitud es construido como indica la sección 5.4.1.1 (como una solicitud fuera de diálogo).

La construcción de repuestas en el diálogo se guía por el procedimiento en 5.4.2.1.

El mecanismo para terminar diálogos confirmados es específico del método. En la especificación RFC 3261, la solicitud con el método “BYE” termina una sesión y el diálogo asociada con ella. Esta solicitud es una solicitud dentro del diálogo y debe ser construida como tal.

5.7 Inicio de sesiones

Cuando un agente de usuario cliente decide iniciar una sesión (por ejemplo, audio, video o un juego), formula una solicitud “INVITE”. La solicitud INVITE pide a un servidor establecer una sesión. Esta solicitud puede ser reenviada por servidores proxy, eventualmente llegando a uno o más UAS que pueden aceptar la invitación. Estos UAS frecuentemente necesitarán preguntar al usuario humano si acepta la invitación. Después de algún tiempo, estos UAS pueden aceptar la invitación (lo que significa que la sesión es establecida) enviando una respuesta 2xx. Si la invitación no es aceptada, se envía una respuesta 3xx, 4xx, 5xx o 6xx, dependiendo de la razón del rechazo. Antes de enviar una respuesta final, el UAS también puede enviar respuestas provisionales (1xx) para avisar al UAC del progreso de contactar al usuario llamado, por ejemplo una respuesta 100 (Intentando) para indicar que la solicitud ha sido recibida y que se están tomando acciones o una respuesta 180 (Repicando) para indicar que el teléfono esta sonando.

Después de recibir una o más repuestas provisionales, el UAC esperará por una o más respuesta. Debido a la prolongada cantidad de tiempo que puede tomar recibir una respuesta final a la solicitud “INVITE” (ya que depende de cuando el usuario llamado ejecuta una acción), el mecanismo de fiabilidad para una transacción “INVITE” difiere de otras solicitudes. Esto quiere decir que el tiempo de expiración de una solicitud “INVITE” es mucho mayor que los de las demás solicitudes. Una vez que recibe una respuesta final, el UAC necesita enviar una solicitud “ACK” por cada respuesta final que recibe.

El procedimiento para enviar esta solicitud “ACK” depende del tipo de respuesta. En el caso de respuestas 2xx, el UAC debe generar una solicitud “ACK” por cada respuesta 2xx recibida. Los campos de esta solicitud ACK son construidos en la misma forma que se hace para una solicitud dentro de un diálogo con la excepción del campo “CSeq”. El número de secuencia del campo “CSeq” debe ser el

mismo que el de la solicitud “INVITE” que se reconoce, pero el método de “CSeq” debe ser “ACK”.

Una respuesta 2xx para una solicitud “INVITE” establece una sesión, y también crea un diálogo entre el UA que envió la solicitud “INVITE” y el UA que genera la respuesta 2xx. Por lo tanto, cuando se reciben múltiples respuestas 2xx de diferentes UA remotos (debido a una solicitud “INVITE” ramificada), cada respuesta 2xx establece un diálogo. Todos estos diálogos son parte de la llamada. Puesto que la solicitud “INVITE” inicial representa una solicitud fuera de diálogo, su construcción sigue los procedimientos de la Sección 4.4.1.1. Un UAC también puede añadir otros campos de cabecera, como por ejemplo, “User-Agent” el cual provee el nombre y versión del agente de usuario cliente.

El UAC puede decidir añadir un cuerpo de mensaje a la solicitud “INVITE”. Si este cuerpo es usado para describir sesiones, se debe hacer uso de la especificación RFC 2327 (Protocolo de Descripción de Sesiones (SDP)). SIP usa un modelo oferta/respuesta descrito en RFC 3264, donde un UA envía una descripción de sesión, llamada oferta, que contiene una descripción propuesta de la sesión. La oferta indica los medios de comunicación deseados (audio, video, juegos), parámetros de esos medios (por ejemplo tipos de CODEC) y direcciones para recibir medios. El otro UA responde con otra descripción de la sesión, llamada respuesta, la cual indica cuales medios de comunicación son aceptados, los parámetros que aplican para esos medios y direcciones para recibir medios. En la especificación RFC 3261, las ofertas y las respuestas sólo pueden aparecer en solicitudes “INVITE”, en sus respuestas y en las solicitudes “ACK”. Hay dos posibilidades para este modelo oferta/respuesta: la oferta esta en la solicitud “INVITE” y la respuesta en el mensaje 2xx, o la oferta está en la respuesta 2xx y la respuesta en la solicitud “ACK”. Si la solicitud “INVITE” no contiene una descripción de sesión, entonces el UAC ha solicitado que el UAS provea la oferta de la sesión en la respuesta 2xx.

5.8 Finalización de sesiones

La solicitud “BYE” es usada para terminar una sesión específica o un intento de sesión. Cuando una solicitud “BYE” es recibida sobre un diálogo, cualquier sesión asociada con ese diálogo debería terminar. Un UA no debe enviar una solicitud “BYE” fuera de dialogo. Los UA llamantes pueden enviar una solicitud “BYE” para diálogos confirmados o para diálogos tempranos, y los UA llamados pueden enviar una solicitud “BYE” sobre diálogos confirmados pero no sobre diálogos tempranos. Sin embargo, un UA llamado no debe enviar una solicitud “BYE” sobre un diálogo confirmado hasta que haya recibido una solicitud “ACK” para la respuesta 2xx o hasta que el tiempo de transacción del servidor culmine.

Una solicitud “BYE” es construida como se haría con cualquier otra solicitud dentro de diálogo. El UAS que recibe la solicitud “BYE” verifica si concuerda con algún diálogo existente. Si no concuerda con ningún diálogo existente, el UAS debería generar una respuesta 481 (Llamada/Transacción no existe). Si existe el diálogo, el UAS debería terminar la sesión siempre que no se trate de sesiones multicast. Si se trata de una sesión multicast, el UAS puede decidir si terminar o no su participación. Independientemente de si termina su participación o no, el UAS debe generar una respuesta 2xx para la solicitud BYE.

5.9 Comportamiento de los servidores proxy

Los servidores proxy son elementos que encaminan las solicitudes SIP hacia los agentes de usuarios servidores y las respuestas SIP hacia los agentes de usuario clientes. Una solicitud puede atravesar varios servidores proxy en su camino hacia un UAS. Cada uno de ellos va a tomar una decisión de enrutamiento, modificando la solicitud antes de enviarla al siguiente elemento. Las respuestas a esa solicitud serán encaminadas a través del mismo conjunto de servidores proxy que atravesó la solicitud en orden inverso.

Cuando un servidor proxy recibe una solicitud, debe inspeccionar el URI-de-la-Solicitud en la línea de inicio. Si este URI indica un dominio que no es responsabilidad de este servidor proxy, éste no debe cambiar el URI-de-la-Solicitud y debe proceder a pasar la solicitud al siguiente elemento. Si el dominio sí es responsabilidad del servidor proxy, éste remplaza el URI-de-la-Solicitud con el resultado obtenido de algún servicio de localización.

Si este servidor proxy desea permanecer en el camino de futuras solicitudes (asumiendo que la solicitud actual creará un diálogo), debe insertar un campo "Record-Route" dentro de la solicitud antes de cualquier campo "Record-Route" existente. Cuando el UAS devuelva una respuesta a esta solicitud, ésta debe contener una réplica de estos campos "Record-Route". Los campos "Record-Route" van a servir en el UAC y el UAS para construir el estado del diálogo. Dentro de estos estados se almacenará esta lista de campos "Record-Route" para así establecer una ruta dentro del diálogo.

Cuando se envíen futuras solicitudes dentro del diálogo, estás podrán contener una lista de campos "Route" que contienen los mismos valores de los campos "Record-Route" almacenados anteriormente. Si están presentes, estos campos "Route" son utilizados por los servidores proxy para encaminar una solicitud. El servidor proxy inspeccionará el campo "Route" que se encuentra en la parte superior. Si este campo "Route" indica este servidor proxy, éste remueve este campo. Luego el servidor proxy reenvía la solicitud al punto indicado en el nuevo campo "Route" en la parte superior o al URI-de-la-solicitud si no hay campos "Route" presentes.

Por otra parte, el servidor proxy debe insertar un campo Via dentro de la solicitud antes de cualquier campo Via existente. Este campo Via indica la ubicación del servidor proxy y debe tener un parámetro branch. Las respuestas se encaminan por el mismo trayecto recorrido por las solicitudes gracias a los campos Via. Cuando

un servidor proxy recibe una respuesta, remueve el campo Via que se encuentra en la parte superior y reenvía la respuesta al punto indicado por el siguiente campo Via.

5.10 Procedimiento para establecer una llamada típica

En resumen, en la siguiente figura se muestra el procedimiento para establecer y finalizar una llamada entre dos usuarios del mismo dominio a través de un servidor proxy que presta servicio en ese dominio. Las flechas azules indican solicitudes mientras que las rojas indican respuestas. En este caso, el servidor proxy decidió no estar en el camino que recorren las solicitudes y sus respuestas dentro del diálogo, al no añadir un campo “Record-Route” a la solicitud “INVITE” inicial. Por esta razón, la solicitud “ACK”, la solicitud “BYE” y su respuesta 200 “OK” no pasan a través de este servidor proxy.

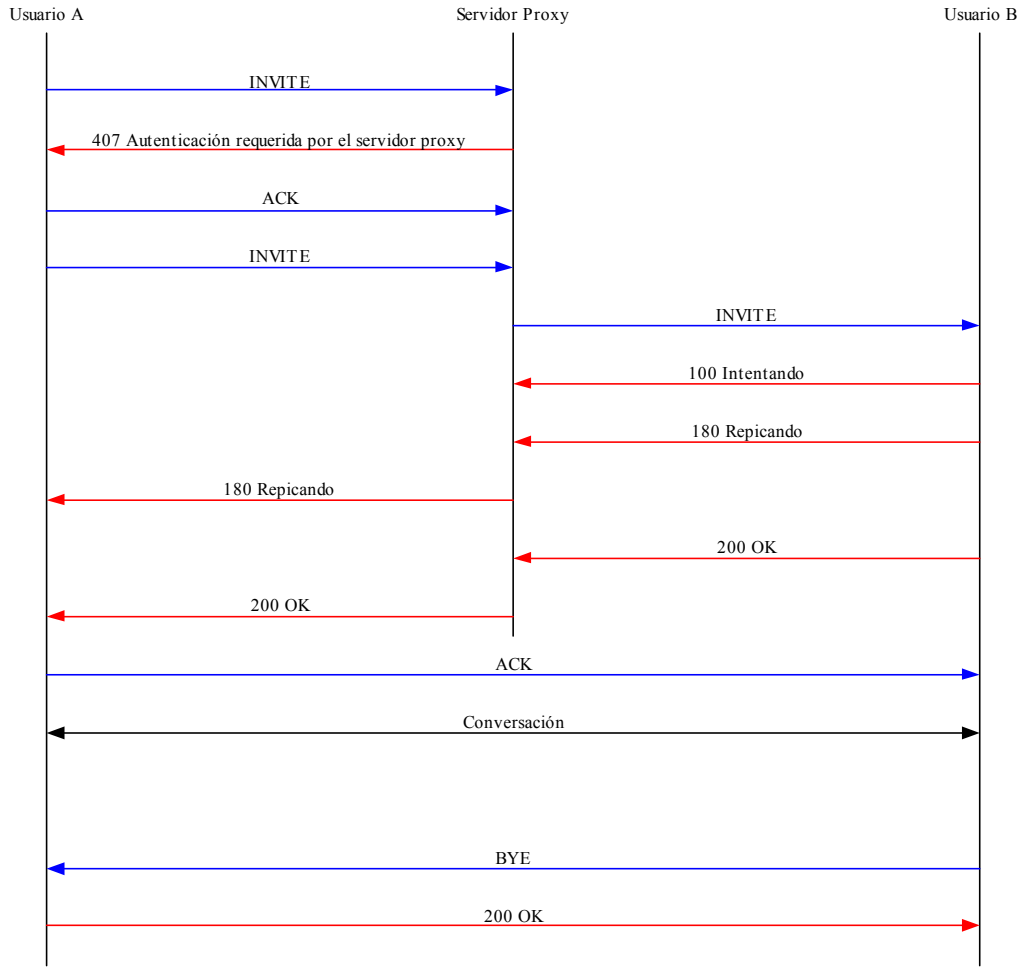


Figura 9. Procedimiento para establecer y finalizar una llamada.

La solicitud “INVITE” (1) es una solicitud que puede originar la creación de un diálogo (de hecho en RFC 3261, es la única que lo puede hacer). Sin embargo, puesto que no se ha establecido uno, esta es una solicitud fuera de diálogo y su construcción se explica en 5.4.1.1 (Generación de solicitudes). También es una solicitud que inicia una sesión y por lo tanto, también se siguen las indicaciones en 5.7 (Inicio de sesiones). En este caso, el servidor proxy solicita información de autenticación por medio de la respuesta 407 (2), la cual se construye como se indica en 5.4.2.1 (Generación de respuestas). Luego el usuario A envía una solicitud “ACK” (3) para indicar el reconocimiento de esta respuesta. Esta solicitud también es una

solicitud fuera de diálogo y se construye según 5.4.1.1. Seguidamente el usuario A procede a enviar nuevamente la solicitud "INVITE" (4), pero esta vez con la información de autenticación solicitada. Una vez más, esta es una solicitud fuera de diálogo y por lo tanto, se construye como indica la sección 5.4.1.1. El servidor proxy recibe esta solicitud, la maneja tal como se explica en 5.9 (Comportamiento de los servidores proxy) y la encamina hacia el usuario B (5). El usuario B responde con una respuesta 100 (Intentando) (6) para indicar al servidor proxy que se están tomando acciones para establecer la sesión. Esta respuesta se diferencia de otras respuestas provisionales en que nunca es reenviada por un servidor proxy. Esta respuesta se construye como se explica en 5.4.2.1. Luego el usuario B envía una respuesta 180 (Repicando) (7) para indicar al usuario A que se está alertando al usuario B. Esta repuesta también se construyen como se explica en 5.4.2.1 y como se indica en la sección 5.6 (Diálogos), porque es una respuesta que crea un diálogo temprano y se debe almacenar el estado de este diálogo. Esta respuesta es encaminada por el servidor proxy según lo especificado en 5.9 (8). Al llegar esta respuesta al usuario A, se almacena el estado del diálogo según 5.6.

Una vez que el usuario B decide aceptar la sesión, envía una respuesta 200 ("OK") (9) al usuario A. Esta respuesta debe ser construida según se explica en 5.4.2.1, en 5.6 (ya que convierte el diálogo temprano en un diálogo confirmado) y en 5.7 (Inicio de Sesiones, porque inicia una sesión). Esta respuesta es encaminada por el servidor proxy hacia el usuario A según 5.9. Al llegar esta respuesta al usuario A, el diálogo queda completamente establecido. Por consiguiente, cualquier solicitud que se envíe a partir de este momento es una solicitud dentro de diálogo. Esto incluye a la solicitud "ACK" (11), la cual es enviada para indicar que se reconoce la respuesta 200, y se debe construir como indica 5.6. Después que se envía esta solicitud se establece la sesión (12). Para finalizar la sesión y el diálogo, se envía una solicitud BYE. Esta solicitud también está dentro del diálogo y se construye según 5.6.

CAPÍTULO VI

APLICACIÓN “ANALIZADOR DE PROTOCOLOS”

La aplicación “Analizador de Protocolos” fue desarrollada con el objetivo de capturar, decodificar y presentar las tramas que contienen los mensajes necesarios para establecer una llamada en los protocolos SIP y H.323. En el caso particular de H.323, esta aplicación debe mostrar los mensajes Q.931 utilizados en la señalización de la llamada, mientras que en el caso de SIP, la aplicación debe mostrar los mensajes decodificados en formato UTF-8. Para lograr esto, la aplicación debe decodificar las cabeceras utilizadas para encapsular estos. Estas cabeceras son las definidas en los protocolos 802.3, Ipv4, UDP, TCP y la cabecera TPKT definida en el estándar RFC 1006.

En la figura 10 se muestra la estructura de esta aplicación. Cuando la interfaz gráfica de usuario lo ordena, el bloque captador comienza la captura de tramas. Luego cada trama es enviada al bloque decodificador. Este bloque está dividido en seis niveles que corresponden a los diversos protocolos que describen la estructura de la trama. Cuando el bloque decodificador recibe una trama, crea y formatea una serie de parámetros de control. Cada vez que la trama atraviesa cada uno de los niveles de decodificación, se leen y modifican estos parámetros para luego ser pasados al siguiente nivel.

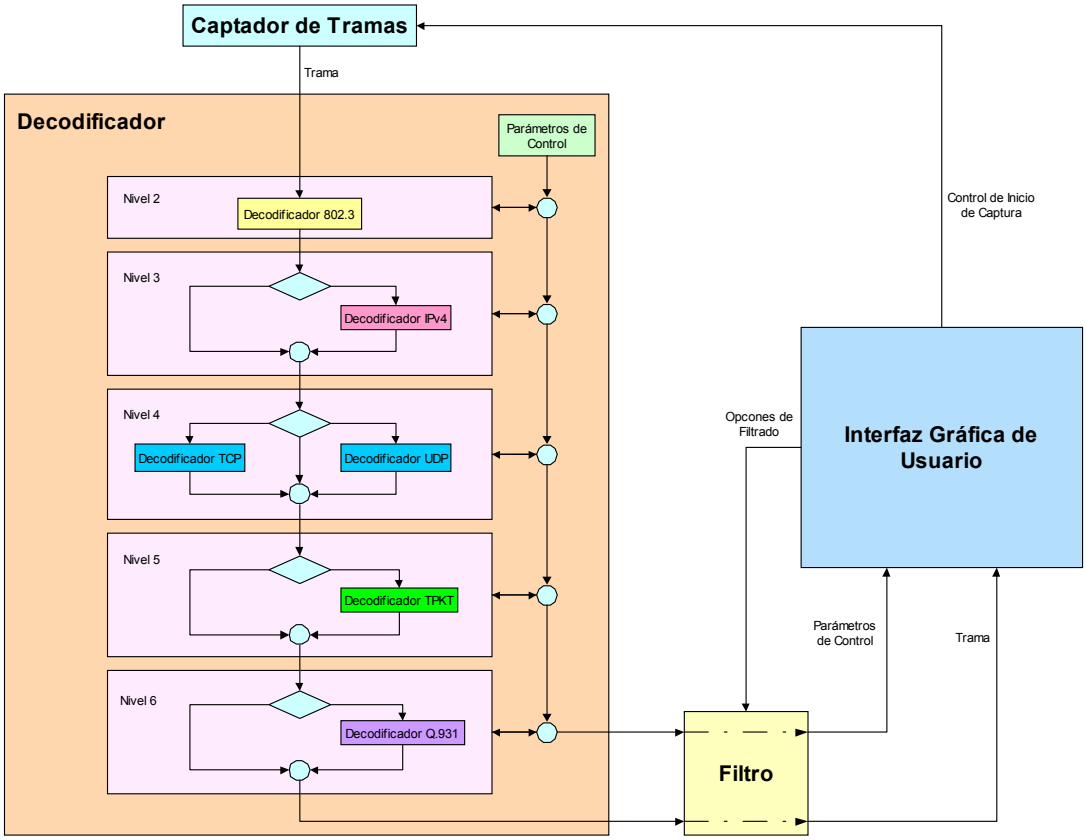


Figura 10. Estructura de la aplicación Analizador de Protocolos

A continuación se describen estos parámetros de control:

- Descripción de la trama: Es un arreglo bidimensional de cadenas de caracteres que sirve para almacenar las descripciones obtenidas cuando los niveles decodifican las tramas. Cuando un nivel obtiene alguna información al decodificar una trama, añade una columna a este arreglo. La primera fila de esta columna es una bandera para indicar el nivel que introdujo esta columna al arreglo. La segunda fila contiene un título que describe la información, por ejemplo, “Dirección IP Origen”. La tercera fila contiene el valor de esa información, por ejemplo, una dirección IP. La cuarta fila indica el espacio que ocupa esta información en la trama, por ejemplo, “4 Octetos”. Las últimas

tres filas serán mostradas al usuario en la interfaz gráfica, mientras que la primera se usa como control y es transparente al usuario.

- Cursores: Son dos números que indican la posición en la trama del siguiente octeto a leer y la posición de la siguiente columna a escribir en el arreglo descripción.
- Protocolos: Es un arreglo unidimensional de cadenas de caracteres usado para indicar los protocolos usados en cada nivel para decodificar la trama. Se usa para mostrar al usuario los protocolos y para filtrar las tramas.
- Control: Es un arreglo unidimensional de cadenas de caracteres que contienen parámetros que definen el origen y el destino de la trama. Contiene las direcciones IP y los puertos de transporte tanto del origen como del destino. Es usado para filtrar las tramas.

Una vez que la trama ha sido codificada, debe pasar por el filtro, en donde se comparan los parámetros de control con las opciones de filtrado que indica la interfaz de usuario, para determinar si la trama debe ser mostrada o no al usuario. Este bloque puede filtrar las tramas en función de las direcciones IP, los puertos de transporte o de los protocolos, según lo especifique el usuario. Finalmente, si cumple con las especificaciones de filtrado, la trama y los parámetros de control son enviados a la interfaz gráfica de usuario.

En esta interfaz, se almacena toda esta información para permitir al usuario analizar cualquier trama captada. En este bloque se muestra en diversas formas la información de la trama. Es posible observar todos los parámetros contenido en una trama tanto en forma de bloques como en formato de texto. Posee vistas correspondientes a cada uno de los niveles de codificación, en donde se puede observar los parámetros de sólo un nivel para todas las tramas. Si la trama no es decodificada completamente por el bloque decodificador, la interfaz gráfica muestra el resto de la trama en forma de texto decodificado en el formato UTF-8, el cual es usado en los mensajes SIP. Esto quiere decir que en el caso de SIP, el bloque

decodificador analizará todos las cabeceras que encapsulan al mensaje SIP y la interfaz gráfica mostrará el mensaje. En el anexo 7 titulado “Manual de uso de la aplicación Analizador de Protocolos” se explican los detalles de esta interfaz gráfica de usuario.

En los anexos 1, 2, 3, 4 y 5 se describen las estructuras de las cabeceras 802.3, IPv4, UDP, TCP y TPKT, respectivamente, que fueron la base para la implementación de los decodificadores de estas cabeceras. En capítulo IV se describe el formato de los mensajes Q.931 y las restricciones que impone sobre estos el protocolo H.225.

Para probar estos decodificadores, se implementó un bloque adicional que imprimía en un archivo de texto los valores numéricos en base 10 de la secuencia de octetos que componen cada trama. Se realizaron varias capturas y se decodificaron las tramas de forma manual. Se verificó que los resultados de estas decodificaciones manuales coincidieran con los resultados arrojados por los decodificadores de la aplicación. También se capturaron tramas con parámetros conocidos y se verificaron estos parámetros en los resultados de la aplicación. Por ejemplo, se capturaron tramas intercambiadas entre dos computadores conocidos y se verificaron las direcciones físicas e IP, tramas de protocolos IPv4 y ARP para confirmar los campos tipo/longitud de la cabecera 802.3, tramas con mensajes HTTP y SIP para verificar los puertos de transporte. Se verificaron las consistencias entre los campos longitud de las cabeceras IPv4 y TPKT y la longitud de la trama capturada. También se verificaron parámetros fijos definidos en los protocolos, por ejemplo, el campo versión en la cabecera IPv4, valores reservados (que generalmente deben ser llevados a ceros lógicos) y los discriminadores de protocolos en las cabeceras TPKT y en los mensajes Q.931.

En el caso de los mensajes Q.931, se verificó que la secuencia de los mensajes de establecimiento de una llamada fuera la esperada y se verificaron las

restricciones impuestas por el mismo protocolo Q.931 y por el protocolo H.225. De la misma forma, en el protocolo SIP también se verificó que la secuencia de los mensajes en el establecimiento de una llamada y la estructura de los mensajes fueran las esperadas.

CONCLUSIONES

La realización de este trabajo especial de grado trajo como fruto una aplicación capaz de capturar los mensajes intercambiados al establecer una llamada usando la recomendación UIT-T H.323 o el estándar SIP (RFC 3261). Esta aplicación servirá como herramienta para el estudio práctico de estas tecnologías al permitir al estudiante verificar en el laboratorio los procedimientos y mensajes utilizados para establecer una llamada. También permitirá aclarar de una forma práctica cualquier duda o inquietud que pueda surgir en la enseñanza teórica o en el estudio directo de estos protocolos.

Esta aplicación va acompañada de una guía para una práctica de laboratorio. Está práctica fue elaborada para que el estudiante obtenga una noción inicial sobre los detalles del establecimiento de una llamada en estos protocolos. En el diseño de este laboratorio se tomaron en cuenta el tiempo disponible para la realización de la práctica (tres horas), el tiempo necesario para el estudio previo a la práctica y los equipos disponibles (tres computadores personales interconectados en una red). En la implementación de la aplicación Analizador de Protocolos se consideraron algunos criterios para facilitar la realización de este laboratorio.

Sin embargo, esta aplicación no sólo se limita al entorno de los protocolos H.323 y SIP. Para poder mostrar los mensajes de estas tecnologías, la aplicación debe decodificar una serie de cabeceras que encapsulan y permiten la correcta transmisión de estos mensajes. Estas cabeceras son las definidas en los estándares 802.3, IPv4, UDP, TCP y la cabecera TPKT definida en el estándar RFC 1006. Esto permite que esta aplicación pueda ser usada en el estudio de estos estándares. Además, la aplicación Analizador de Protocolos fue diseñada con una estructura tal que facilite la incorporación de otros nuevos decodificadores.

RECOMENDACIONES

En el establecimiento de una llamada H.323 están involucrados tres canales de comunicación. El primer canal que se establece es el canal RAS (Registro, Admisiones y Situación), luego se establece el canal de señalización de llamada y finalmente se establece el canal de control de la llamada. Tal como era el objetivo inicial, la aplicación Analizador de Protocolos sólo permite visualizar los mensajes intercambiados en el canal de señalización. Sin embargo, también es importante comprender los mensajes intercambiados en los canales RAS y de control. Por lo tanto, se aconseja la implementación de una aplicación o la extensión del programa Analizador de Protocolos, con el fin de observar estos mensajes.

En el caso de SIP se recomienda la incorporación de los conceptos involucrados en el protocolo SDP (Protocolo de Descripción de Sesiones) en el estudio del laboratorio.

Por otra parte, se recomienda la implementación de una red con un número mayor de equipos que permita el estudio de las otras funcionalidades que ofrecen los protocolos SIP y H.323.

Sin embargo, todo esto implicaría un mayor tiempo para la realización de la práctica. También implicaría un mayor tiempo necesario para que el estudiante haga una preparación previa al laboratorio. Esto resultaría en una división de la misma en dos prácticas de laboratorio, una para H.323 y otra para SIP. No obstante, la decisión de hacer estas ampliaciones depende de si en verdad se requiere preparar a los profesionales a un nivel de profundidad mayor.

REFERENCIAS BIBLIOGRÁFICAS

[1] Estándar IEEE 802.3 (2005), Método de acceso múltiple con detección de portadora y detección de colisiones (CSMA/CD) y especificaciones de capa física.

[2] Estándar RFC 768 (1980), Protocolo de datagramas de usuario

[3] Estándar RFC 791 (1981), Protocolo de Internet.

[4] Estándar RFC 793 (1981), Protocolo de control de transmisión.

[5] Estándar RFC 1006 (1987), Servicios de transporte ISO sobre el tope de TCP.

[6] Estándar RFC 3261 (2002), SIP: Protocolo de iniciación de sesiones.

[7] Recomendación UIT-T H.225.0 (2000), Protocolos de señalización de llamada y paquetización de trenes de medios para sistemas de comunicación multimedios por paquetes.

[8] Recomendación UIT-T H.245 (2000), Protocolo de control para comunicación multimedios.

[9] Recomendación UIT-T H.323 (2000), Sistemas de comunicación multimedios basados en paquetes.

[10] Recomendación UIT-T Q.931 (1998), Especificación de la capa 3 de la interfaz usuario-red de la red digital de servicios integrados para el control de la llamada básica.

[11] History of Voice Over IP. <http://www2.rad.com/networks/2001/voip/history.htm>

[12] History of VoIP. <http://www.voipreview.org/news.details.aspx?nid=51>

[13] VoIP Foro – QoS – jitter – Causas, Soluciones y valores recomendados. <http://www.voipforo.com/QoS/QoS_Jitter.php>

[14] VoIP Foro – QoS – eco – Causas, Soluciones y valores recomendados. <http://www.voipforo.com/QoS/QoS_Eco.php>

[15] C. Schlatter. Basic Architecture of H.323, 2003. (figura, presentación)

[16] Josef Glasmann, Wolfgang Kellerer, Harald Müller. Service Architectures in H.323 and SIP – A Comparison, Germany, 2001. (pila)

BIBLIOGRAFÍA

Estándar RFC 768 (1980), Protocolo de datagramas de usuario

Estándar RFC 791 (1981), Protocolo de Internet.

Estándar RFC 793 (1981), Protocolo de control de transmisión.

Estándar RFC 1006 (1987), Servicios de transporte ISO sobre el tope de TCP.

Estándar RFC 3261 (2002), SIP: Protocolo de iniciación de sesiones.

Recomendación UIT-T H.225.0 (2000), Protocolos de señalización de llamada y paquetización de trenes de medios para sistemas de comunicación multimedios por paquetes.

Recomendación UIT-T H.245 (2000), Protocolo de control para comunicación multimedios.

Recomendación UIT-T H.323 (2000), Sistemas de comunicación multimedios basados en paquetes.

Recomendación UIT-T Q.931 (1998), Especificación de la capa 3 de la interfaz usuario-red de la red digital de servicios integrados para el control de la llamada básica.

[ANEXO 1]

[Trama 802.3]

En la siguiente tabla se muestra el formato de una trama 802.3

Campo	Longitud
Preámbulo	7 Octetos
Delimitador de inicio de trama (SFD)	1 Octeto
Dirección destino	6 Octetos
Dirección origen	6 Octetos
Longitud/Tipo	2 Octetos
Datos y Relleno (PAD)	48-1500 Octetos
Secuencia de verificación de trama (FCS)	4 Octetos
Extensión	-

Tabla 22. Formato de una trama 802.3

- Preámbulo: Es un campo de 7 octetos usado para permitir que el circuito de señalización física alcance una sincronización en el tiempo con la trama recibida. Cada octeto consiste de la secuencia alternada "10101010".
- Delimitador de inicio de trama (SFD): Es un octeto con la secuencia "10101011" que indica el comienzo de la trama.
- Dirección destino: Especifica la estación a la que se envía la trama.
- Dirección origen: Especifica la estación que envía la trama.
- Longitud/Tipo: Si el valor de este campo es menor o igual que el valor maxValidFrame que se especifica en la sección 4.2.7.1 del estándar 802.3, entonces indica el numero de octetos de los datos contenidos en la trama (el valor de maxValidFrame depende de la implementación, sin embargo, para todas las implementaciones de 802.3 existentes, este valor es igual 1500). Si el valor de este campo es mayor o igual que 1536, entonces indica la naturaleza del protocolo cliente (tipo).
- Datos y PAD: Contiene los datos a ser transportados. Se requiere un tamaño de la trama mínimo para un correcto acceso al medio con detección de

portadora y detección de colisiones (CSMA/CD). Por lo tanto, si es necesario, el campo datos es extendido con un relleno (pad en Inglés).

- Secuencia de verificación de trama: Contiene un valor de verificación de redundancia cíclica (CRC) de cuatro octetos. Se calcula en función del contenido de la dirección destino, la dirección origen, los datos y el PAD.
- Extensión: Es una secuencia de bits de extensión que se utiliza para extender la trama y así cumplir ciertas restricciones de tiempo en velocidades de operación de 1000 Mb/s.

[ANEXO 2]

[Cabecera IPv4]

La siguiente tabla muestra el formato de una cabecera IPv4.

0								10								20								30							
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Versión				IHL				Tipo de servicio								Longitud total															
Identificación																Banderas			Posición del fragmento											
Tiempo de vida								Protocolo								Suma de control de cabecera															
Dirección origen																															
Dirección destino																															
Opciones (si las hay)																												Relleno			
Datos																															

Tabla 23. Formato de la cabecera IP.

- Versión: Indica el formato de la cabecera de Internet. Para IPv4, este valor es 4.
- IHL (Longitud de la cabecera de Internet): Es la longitud de la cabecera de internet en palabras de 32 bits (4 octetos), y por consiguinte indica el comienzo de los datos. El valor mínimo es 5.
- Tipo de servicio: Provee una indicación de los parámetros abstractos de la calidad de servicio deseada. Es usado para especificar el tratamiento de datagramas durante su transmisión a través del sistem Internet. Se codifica de la siguiente forma:
    - Precedencia (bits 18,19,20): Puede tomar los siguientes valores: Control de red, Control entre redes, CRITICO/ECP, Muy urgente, Urgente, Inmediato y Rutina.
    - Retardo (bit 21): 0 indica un retardo normal y 1 indica un bajo retardo.
    - Rendimiento (bit 22): 0 indica un rendimiento normal y 1 indica alto rendimiento.
    - Fiabilidad (bit 23): 0 indica fiabilidad y 1 indica alta fiabilidad.
    - Reservado para uso futuro (bits 24, 25).
- Longitud total: Indica la longitud del datagrama, medida en octetos, incluyendo la cabecera de internet y los datos.
- Identificación: Es un valor de identificación asignado por el emisor para ayudar en el ensamblaje de los fragmentos del datagrama.
- Banderas: Son diversos indicadores de control:
    - Reservado (bit 26): Debe ser 0.
    - No fragmentar (bit 27): o indica que puede fragmentarse y 1 indica no fragmentar.

    - Más fragmentos (bit 28): 0 indica que es el último fragmento y 1 indica que hay más fragmentos.
- Posición del fragmento: Este campo indica a que parte del datagrama pertenece este fragmento. La posición del fragmento se mide en unidades de 8 octetos (64 bits). El primer fragmento tiene posición 0.
- Tiempo de vida: Este campo indica el tiempo máximo que el datagrama tiene permitido permanecer en el sistema internet. Si este campo contiene el valor cero, entonces el datagrama debe ser destruído. Este campo es modificado durante el procesamiento de la cabecera internet. El tiempo es medido en segundos, pero como todo módulo que procese un datagrama debe decrementar el TTL (Time To Live: Tiempo de Vida) al menos en uno, incluso si procesa el datagrama en menos de un segundo, se debe pensar en el TTL sólo como un límite superior del tiempo durante el cual un datagrama puede existir. La intención es hacer que los datagramas imposibles de entregar sean descartados, y limitar el máximo periodo de vida de un datagrama.
- Protocolo: Este campo indica el protocolo del siguiente nivel usado en la parte de datos del datagrama internet.
- Suma de Control de Cabecera: Es una suma de control sólo para la cabecera. Puesto que le cabecera cambia (por ejemplo, el tiempo de vida), es recalculado en cada punto donde la cabecera es procesada.
- Dirección origen: Indica la dirección del origen.
- Dirección destin: Indica la dirección del destino.
- Opciones: Las opciones pueden o no aparecer en los datagramas. Por lo general no aparecen. El estándar RFC 791 especifica el formato de las opciones.
- Relleno: El Valor de Relleno se usa para asegurar que la cabecera internet ocupa un multiplo de 32 bits. El valor de relleno es cero.

[ANEXO 3]

[Cabecera UDP]

La siguiente tabla muestra el formato de la cabecera UDP.

0										10										20										30	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	3	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Puerto origen																Puerto destino															
Longitud																Suma de control															
Datos																															

Tabla 24. Formato de la cabecera UDP.

- Puerto origen: Es opcional. cuando tiene sentido, indica el puerto del proceso emisor, y puede que se asuma que ése sea el puerto al cual la respuesta debería ser dirigida en ausencia de otra información. Si no se utiliza, se inserta un valor cero.
- Puerto destino: tiene significado dentro del contexto de una dirección de destino en un entorno internet particular.
- Longitud: representa la longitud en octetos de este datagrama de usuario, incluyendo la cabecera y los datos (esto implica que el valor mínimo del campo Longitud es ocho).
- Suma de control: Suma de comprobación de errores del mensaje. Para su cálculo se utiliza una pseudo-cabecera construida con información de la cabecera IP, la cabecera UDP y los datos.

[ANEXO 4]

[Cabecera TCP]

En la siguiente tabla se muestra el formato de la cabecera TCP.

<table>
<tr><td colspan="10">0</td><td colspan="10">10</td><td colspan="10">20</td><td colspan="2">30</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>3</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="16">Puerto origen</td><td colspan="16">Puerto destino</td></tr>
<tr><td colspan="32">Número de secuencia</td></tr>
<tr><td colspan="32">Número de acuse de recibo</td></tr>
<tr><td colspan="4">Posición de los datos</td><td colspan="6">Reservado</td><td colspan="6">Bits de control</td><td colspan="16">Ventana</td></tr>
<tr><td colspan="16">Suma de verificación</td><td colspan="16">Puntero de urgencia</td></tr>
<tr><td colspan="24">Opciones (si las hay)</td><td colspan="8">Relleno</td></tr>
</table>

Tabla 25. Formato de la cabecera TCP.

- Puerto origen: El número del puerto de origen.
- Puerto destino: El número del puerto de destino.
- Número de secuencia: Es el número de secuencia del primer octeto de datos de este segmento (excepto cuando el indicador SYN esté puesto a uno). Si SYN está puesto a uno, entonces representa el número de secuencia original ISN (Initial Sequence Number) y el primer octeto de datos es ISN+1.
- Número de acuse de recibo: Si el bit de control ACK está puesto a uno, este campo contiene el valor del siguiente número de secuencia que el emisor del segmento espera recibir. Una vez que una conexión queda establecida, este número se envía siempre.
- Posición de los datos: El número de palabras de 32 bits que ocupa la cabecera de TCP. Este número indica dónde comienzan los datos. La cabecera de TCP (incluso una que lleve opciones) es siempre un número entero de palabras de 32 bits.
- Reservado: Reservado para uso futuro. Debe valer 0.
- Bits de control: Son 6 bits con los siguientes significados:
    - URG (bit 10): Hace significativo el campo "Puntero urgente"
    - ACK (bit 11): Hace significativo el campo "Número de acuse de recibo"
    - PSH (bit 12): Función de "Entregar datos inmediatamente" (“Push”)
    - RST (bit 13): Reiniciar la conexión (“Reset”)
    - SYN (bit 14): Sincronizar los números de secuencia ('Synchronize').
    - FIN (bit 15): Últimos datos del emisor.

- Ventana: El número de octetos de datos, a contar a partir del número indicado en el campo de "Número de acuse de recibo", que el emisor de este segmento está dispuesto a aceptar.
- Suma de control: Suma de comprobación de errores del segmento actual. La suma de control también incluye una pseudocabecera de 96 bits prefijada imaginariamente a la cabecera TCP. Esta pseudocabecera contiene la dirección de origen, la dirección de destino, el protocolo, y la longitud del segmento de TCP.
- Puntero urgente: Este campo indica el valor actual del puntero urgente como un desplazamiento positivo desde el número de secuencia de este segmento. El puntero urgente apunta al número de secuencia del octeto al que seguirán los datos urgentes. Este campo es interpretado únicamente si el bit de control URG está establecido a uno.
- Opciones: Es opcional.
- Relleno: El relleno de la cabecera de TCP se utiliza para asegurar que la cabecera de TCP finaliza, y que los datos comienzan, en una posición múltiplo de 32 bits. El relleno está compuesto de ceros.

[ANEXO 5]

[Cabecera TPKT]

En la siguiente tabla se muestra el formato de la cabecera TPKT.

0										10					
0	1	2	3	4	5	6	7	8	9	0	1	2	3	3	5
Versión										Reservado					
Longitud del paquete															
Datos															

Tabla 26. Cabecera TPKT.

- Versión: Este campo es siempre 3.
- Longitud del paquete: Contiene la longitud en octetos del paquete completo incluyendo la cabecera

[ANEXO 6]

[Práctica de Laboratorio]

PRÁCTICA DE LABORATORIO

VOZ SOBRE IP (VoIP)

1. Objetivo

El objetivo de esta práctica es observar y verificar los procedimientos en el establecimiento de una llamada en los protocolos H.323 y SIP.

2. Voz sobre IP (VoIP)

Voz sobre IP (VoIP) es la tecnología que permite la transmisión de fragmentos auditivos a través de redes basadas en el Protocolo de Internet (IP).

Su funcionamiento consiste en convertir la señal voz en una señal digital. Luego esta señal digital es colocada dentro de paquetes IP para ser transmitido en la red hacia el destino. De forma inversa, cuando estos paquetes llegan al destino, se les extrae esta señal digital y se convierte en una señal analógica. Sin embargo, más allá de transmitir la voz, es necesario el intercambio de una serie de mensajes de señalización para establecer, controlar y finalizar la llamada. Actualmente existen varios estándares que describen estos mensajes y procedimientos de señalización.

El crecimiento y la fuerte implantación de las redes IP unido con el desarrollo de técnicas avanzadas de digitalización de voz, mecanismos de control, mecanismos asignación de prioridad al tráfico y protocolos de transmisión en tiempo real han creado un entorno donde es posible transmitir telefonía sobre IP. En sólo unos pocos años VoIP pasó a ser la principal alternativa al servicio de telefonía convencional. En tal sentido, tanto la ITU (Unión Internacional de Telecomunicaciones) como el IETF (Grupo de Trabajo en Ingeniería de Internet) han estado desarrollando arquitecturas y protocolos para sistemas multimedia sobre IP. Dentro de estos se encuentran la recomendación H.323 desarrollado por la ITU y el estándar SIP (RFC 3261) creado por al IETF, los cuales son los más usados hoy en día en VoIP.

3. Recomendación UIT-T H.323

La recomendación H.323 es una recomendación del UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) que tiene como título “Sistemas de comunicaciones multimedia basados en paquetes”. H.323 expone los requisitos técnicos de los sistemas de comunicaciones multimedios en aquellas situaciones en las que la red de transporte subyacente es una red por paquetes que puede no

garantizar la calidad de servicio. Esta Recomendación describe los componentes de un sistema H.323, lo que incluye terminales, pasarelas, controladores de acceso, controladores multipunto, procesadores multipunto y unidades de control multipunto. También describe los mensajes y procedimientos de control para comunicar estos componentes. H.323 no puede considerarse como un protocolo completo en sí mismo, sino más bien se trata de una especificación que define el modo de interactuar de varios protocolos entre sí.

3.1 Componentes

Dentro de los componentes que se definen en H.323 se encuentran:

- Terminal: Es un punto extremo de la red que facilita las comunicaciones en tiempo real y en los dos sentidos con otro terminal, pasarela o unidad de control multipunto H.323. Esta comunicación consta de control, indicaciones, audio, imágenes de vídeo en color y en movimiento y/o datos entre los dos terminales.
- Guardián de puerta (Gatekeeper): Es una entidad H.323 de la red que facilita la traducción de direcciones y controla el acceso a la red para terminales, pasarelas y MCU H.323. El guardián de puerta puede prestar también otros servicios a los terminales, las pasarelas y las MCU, tales como la gestión de anchura de banda y la localización de pasarelas. Es opcional en un sistema H.323.

3.2 Protocolos

H.323 no puede considerarse como un protocolo completo en sí mismo, sino más bien se trata de una especificación que define el modo de interactuar de varios protocolos entre sí. En la siguiente figura se muestra la pila de los protocolos más relevantes en H.323.

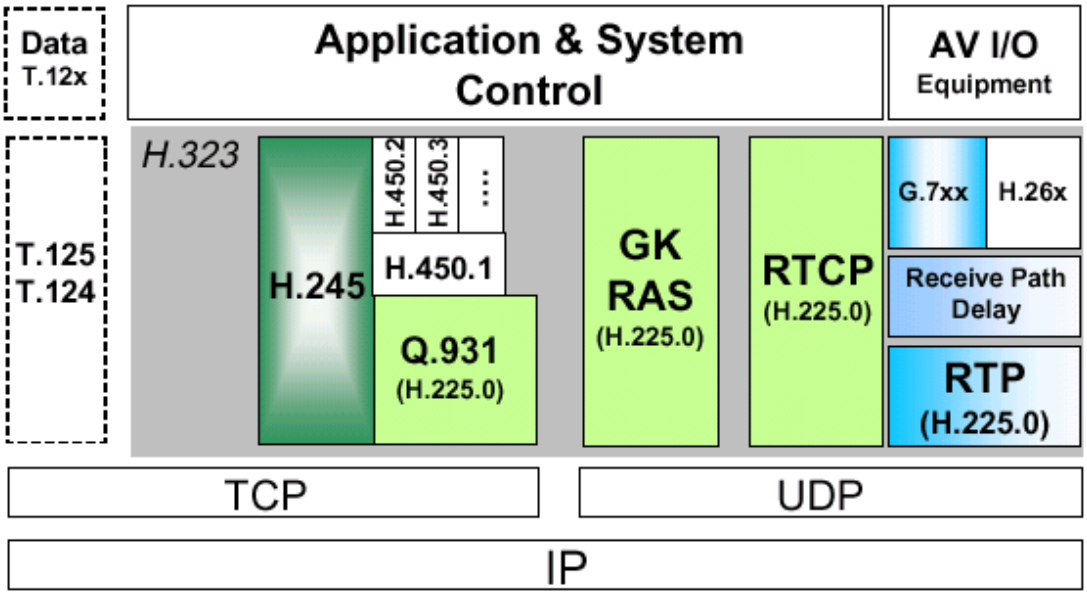


Figura 1. Pila de protocolos de H.323.

- H.225: Es un protocolo de señalización de llamada y empaquetamiento de trenes de medios para sistemas de comunicación multimedios por paquetes. Sus funciones pueden dividirse en tres partes:
    - Registro, admisiones y situación (RAS): Utiliza mensajes H.225.0 para llevar a cabo los procedimientos de registro, admisiones, cambios de anchura de banda, situación y desligamiento entre puntos extremos y controladores de acceso. La recomendación H.225 define una sintaxis de mensajes a través de un árbol ASN.1. Estos mensajes son codificados según la regla PER de ASN.1.
    - Señalización de la llamada: La función de señalización de llamada consiste en los mensajes y procedimientos utilizados para establecer una llamada, pedir cambios de anchura de banda de la llamada, obtener el estado de los puntos extremos de la llamada y desconectar la llamada. Para ello se utilizan mensajes codificados según la recomendación Q.931 y bajo las restricciones y modificaciones establecidas por la recomendación H.225. La recomendación H.225 indica cuales de los tipos de mensajes y elementos de información definidos en Q.931 serán usados en H.323, además de imponer algunas modificaciones en la estructura de ciertos elementos de información.
    - Empaquetamiento: Esta función de H.225 formatea los trenes de vídeo, audio, datos y control transmitidos en mensajes de salida hacia la interfaz de la red y recupera los trenes de vídeo, audio, datos y control recibidos de los mensajes que han sido introducidos desde la interfaz de la red. Además, lleva a cabo la alineación de trama lógica, la numeración secuencial, la detección de errores y la corrección de los

mismos según conviene a cada tipo de medio. H.225 hace uso del protocolo en tiempo real/protocolo de control en tiempo real (RTP/RTCP, real-time transport protocol/real-time transport control protocol) para la empaquetamiento y sincronización de medios.

- H.245: Esta recomendación es un protocolo de control para comunicaciones multimedios. Se definen procedimientos para llevar los mensajes de control de extremo a extremo que rigen el funcionamiento de la entidad H.323, incluyendo el intercambio de capacidades, apertura y cierre de canales lógicos, peticiones de modo preferido, mensajes de control de flujo e instrucciones e indicaciones generales. Esta recomendación define una sintaxis de mensajes a través de un árbol ASN.1. Estos mensajes son codificados según la regla PER de ASN.1.

3.3 Procedimientos para establecer y finalizar una llamada

A continuación se describen los procedimientos para establecer una llamada H.323 en un caso típico. Este caso el terminal A desea llamar al terminal B, y estos dos terminales están dentro del entorno de un guardián de puerta (gatekeeper) quien decide encaminar la señalización de la llamada.

En el establecimiento de una llamada H.323 están involucrados tres canales: RAS, señalización y control (figura 2).

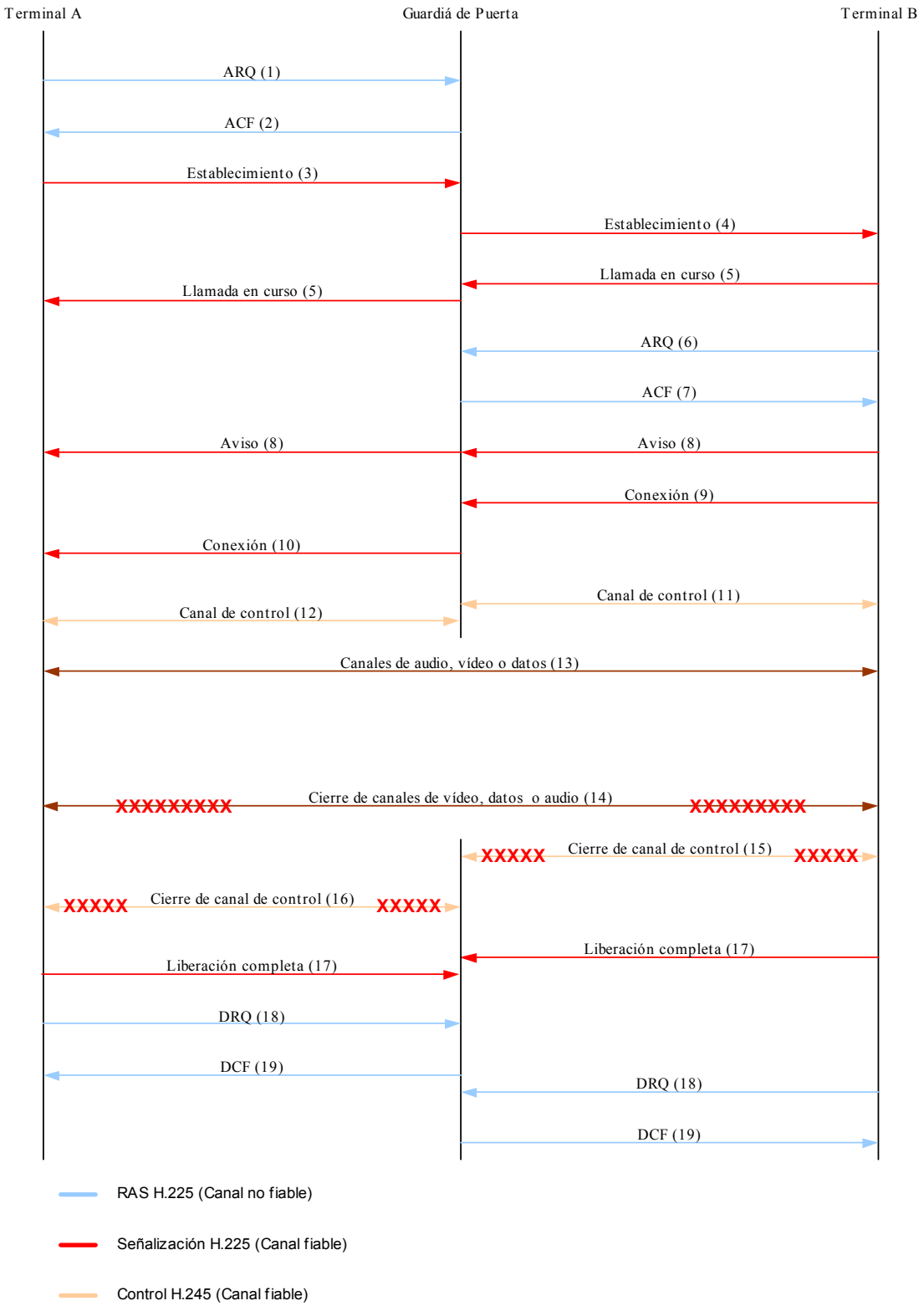


Figura 2. Establecimiento y finalización de una llamada H.323.

El canal de señalización RAS se abre antes de que se establezca cualquier otro canal. A través de este canal, el terminal A debe solicitar acceso a la red de paquetes por el guardián de puerta. Esto se hace por medio de un mensaje de petición de admisión (ARQ, Admision Request) (1). El guardián de puerta concede la petición con un mensaje de confirmación de admisión (ACF, Admision Confirm) (2). Estos mensajes son definidos en la sintaxis ASN.1 de H.225 y codificados en PER (Packed Encoding Rules). El guardián de puerta indica en el mensaje ACF (2) si la señalización de llamada se envía directamente al otro punto extremo o si se encamina a través del guardián de puerta. En este caso, la señalización se encaminará a través del guardián de puerta. En el mensaje ACF el guardián de puerta también indica una dirección de transporte para la señalización de la llamada.

El terminal A comienza el envío de mensajes de señalización utilizando la dirección de transporte indicada por el mensaje ACF. El terminal A envía un mensaje "Establecimiento" (3) para indicar su deseo de establecer una conexión hacia el terminal B. Luego el guardián de puerta reenvía este mensaje hacia el terminal B (4), el cual responde con un mensaje "Llamada en curso" (5) para indicar al guardián de puerta que se ha iniciado el establecimiento de la llamada solicitado y que no se aceptará más ninguna información de establecimiento de llamada. De la misma forma, el guardián de puerta envía un mensaje "Llamada en curso" (5) al terminal A. El terminal B debe responder a este mensaje al terminal A y por consiguiente, solicita el acceso al guardián de puerta por medio de un mensaje ARQ (6). Esta solicitud es confirmada con un mensaje ACF (7). A partir de este momento el terminal B puede enviar mensajes de señalización hacia el terminal A a través del guardián de puerta. Seguidamente, el mensaje "Aviso" (8) es enviado por el terminal B y encaminado por el guardián de puerta hacia el terminal A, para indicar que se ha iniciado el aviso al usuario B, en términos de hoy día, "el teléfono está sonando". Finalmente, una vez que el usuario B responde la llamada, el terminal B envía al guardián de puerta el mensaje "Conexión" (9) para indicar aceptación de la llamada. Este mensaje contiene una dirección de transporte del terminal B para su utilización en el canal de control H.245 entre el guardián de puerta y el terminal B. El guardián de puerta reenvía al terminal A el mensaje "Conexión" (10), pero con una dirección de transporte de él para su utilización en el canal de control H.245 entre el terminal A y el guardián de puerta.

Una vez que ambos lados han intercambiado los mensajes de establecimiento de llamada de la fase A, los puntos extremos, si proyectan emplear H.245, establecerán el canal de control H.245 (11 y 12). Se utilizan los procedimientos de la Rec. UIT-T H.245 en el canal de control H.245 para el intercambio de capacidad y la apertura de canales de medios. En este canal se intercambian mensajes de la sintaxis ASN.1 definida en H.245 y codificados en PER (Packed Encoding Rules). Después del intercambio de capacidades y la determinación de principal-subordinado, se utilizarán los procedimientos de la recomendación H.245 para abrir canales lógicos para los diversos trenes de información (13). Los trenes de audio y vídeo, que se

transmiten por los canales lógicos establecidos en H.245, utilizarán un protocolo no fiable. Las comunicaciones de datos, que se transmiten por los canales lógicos establecidos en H.245, se transportan utilizando un protocolo fiable.

Al finalizar la llamada, lo primero que se debe hacer es interrumpir las transmisiones de vídeo, datos y audio (14). Luego se deben cerrar los canales de control H.245 (15 y 16). En la figura 2 se supone que el terminal B es quien finaliza la llamada y por consiguiente, es quien cierra primero el canal de control. Después se enviarán mensajes "Liberación completa" (17) y se cerrarán los canales de señalización. Finalmente, cada punto extremo transmitirá en el canal RAS un mensaje de petición de desligamiento (DRQ, Disengage Request) al guardián de puerta (18). El guardián de puerta responderá con un mensaje de confirmación de desligamiento (DCF, Disengage Confirm) (19).

3.4 Estructura de los mensajes de señalización

Los mensajes Q.931 son utilizados en la señalización de una llamada H.323. El protocolo Q.931 describe el formato general de estos mensajes y la codificación de los elementos de información que están contenidos dentro de estos mensajes. El protocolo Q.931 fue creado para redes ISDN y su título es "Especificación de la capa 3 de la interfaz usuario-red de la red digital de servicios integrados para el control de la llamada básica". Por lo tanto, el protocolo H.225 impone restricciones a estos mensajes y a los elementos de información para ser usados en H.323. En la siguiente tabla se muestra el formato de un mensaje Q.931.

<table>
<tr><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>Bit/Octeto</th></tr>
<tr><td colspan="8">Discriminador de protocolo</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="4">Longitud del valor de la referencia de llamada (en octetos)</td><td>2</td></tr>
<tr><td colspan="8">Valor de la referencia de llamada</td><td>3</td></tr>
<tr><td>0</td><td colspan="7">Tipo de mensaje</td><td>4</td></tr>
<tr><td colspan="8">Otros elementos de información, según se requieran</td><td>etc.</td></tr>
</table>

Tabla 1. Formato de un mensaje Q.931.

Los elementos de información discriminador de protocolo, longitud del valor de la referencia de llamada, valor de la referencia de llamada y tipo de mensaje son comunes a todos los mensajes y están siempre presentes, mientras que los otros elementos de información son específicos de cada mensaje.

3.4.1 Discriminador de protocolo

El discriminador de protocolo es la primera parte de cada mensaje. La finalidad del discriminador de protocolo es distinguir mensajes para el control de la llamada usuario-red de otros mensajes. En mensajes Q.931 debe contener la secuencia de bits "00001000" (8 en base 10).

3.4.2 Valor de la referencia de llamada

La finalidad de la referencia de llamada es identificar a qué llamada se aplica un mensaje particular. Se codifica como muestra la tabla 2. La longitud del valor de la referencia de llamada se indica en el octeto 1, bits 1 a 4.

8	7	6	5	4	3	2	1	Bit/Octeto
0	0	0	0	Longitud del valor de la referencia de llamada (en octetos)				1
Bandera								2
Valor de la referencia de llamada								etc.

Tabla 2. Elemento de información Referencia de Llamada [10].

El elemento de información referencia de llamada incluye el valor y la bandera de la referencia de llamada. Los valores de la referencia de llamada se asignan, para una llamada, en el punto extremo que origina la llamada. Estos valores son únicos solamente para el lado origen en una llamada determinada. El valor de la referencia de llamada se asigna al comienzo de una llamada y permanece fijo mientras dura la llamada. Cuando termina una llamada, el valor de la referencia de llamada asociado puede reasignarse a otra llamada. La bandera de la referencia de llamada se utiliza para identificar el extremo del enlace lógico que ha originado la llamada. El lado de origen pone siempre la bandera de la referencia de llamada a "0". El lado de destino pone siempre la bandera de la referencia de llamada a "1". Por consiguiente, la bandera de la referencia de llamada identifica quien asignó el valor de la referencia de llamada para esta llamada, y su única finalidad es resolver las tentativas simultáneas de asignar un mismo valor de referencia de llamada.

3.4.3 Tipo de mensaje

La finalidad del tipo de mensaje es identificar la función del mensaje que se envía. Es la tercera parte de cada mensaje y se codifica como se muestra en la tabla 3. El bit 8 se reserva para un posible uso futuro como bit de ampliación.

8	7	6	5	4	3	2	1	Bit/Octeto
0	Tipo de mensaje							1

Tabla 3. Elemento de información Tipo de mensaje. [10]

No todos los tipos de mensajes definidos en Q.931 son usados en H.323. El protocolo H.225 indica cuales tipos de mensajes se usan en H.323. Algunos de estos tipos de mensaje se muestran en la siguiente tabla.

Bits	Tipo de mensaje
8 7 6 5 4 3 2 1	
0 0 0 - - - - -	Mensaje de establecimiento de la llamada:
0 0 0 0 1	– AVISO
0 0 0 1 0	– LLAMADA EN CURSO
0 0 1 1 1	– CONEXIÓN
0 0 1 0 1	– ESTABLECIMIENTO
0 1 1 0 1	– ACUSE DE ESTABLECIMIENTO
0 1 0 - - - - -	Mensajes de liberación de llamada:
1 1 0 1 0	– LIBERACIÓN COMPLETA

Tabla 4. Tipos de mensajes.

3.4.4 Otros elementos de información

A parte de los elementos de información comunes, Q.931 define otros que, dependiendo del tipo de mensaje, pueden aparecer o no. Los elementos de información se clasifican de la siguiente forma:

- Elementos de información de un solo octeto: Son elementos de información con una longitud de un Octeto (tablas 5). El bit más significativo es igual a 1.
- Elementos de información de longitud variable: Son elementos de información cuyo contenido puede contener más de un octeto. Por lo general su estructura es igual a la que se muestra en la tabla 6. El bit más significativo del primer octeto es igual a 0.

8	7	6	5	4	3	2	1	Bit/Octeto
1	Identificador del elemento de información							1

Tabla 5. Elemento de información de un solo octeto.

8	7	6	5	4	3	2	1	Bit/Octeto
0	Identificador del elemento de información							1
Longitud del contenido del elemento de información (octetos)								2
Contenido del elemento de información								3 etc.

Tabla 6. Elemento de información de longitud variable [10]

.

No todos los elementos de información definidos en el protocolo Q.931 son usados en H.323. El protocolo H.225 enumera cuales elementos de información deben ser usados en H.323. En la tabla 7 se muestran algunos de ellos con sus respectivos identificadores. Los elementos de información de un solo octeto pueden aparecer en cualquier posición del mensaje. Sin embargo, existe un orden particular de aparición de cada elemento de información de longitud variable. Los valores de código del identificador de elemento de información para los formatos de longitud variable se asignan en orden numérico ascendente, de acuerdo con el orden real de aparición de cada elemento de información en un mensaje. Por ejemplo, un elemento de información visualización siempre debe aparecer después del elemento de información capacidad de portador. Esto permite al equipo receptor detectar la

presencia o ausencia de un elemento de información particular sin explorar todo el mensaje.

Identificador	Elemento de Información
Bits 8 7 6 5 4 3 2 1	
1 : : : - - - -	Elementos de información de un solo octeto:
0 1 0 0 0 0 1	Envío completo
0 : : : : : : :	Elementos de información de longitud variable:
0 0 0 0 1 0 0	Capacidad portadora
0 0 0 1 0 0 0	Causa
0 1 0 1 0 0 0	Visualización
1 1 1 1 1 1 0	Usuario a usuario

Tabla 7. Elementos de información.

El segundo octeto de un elemento de información de longitud variable indica la longitud total del contenido de ese elemento de información empezando en el octeto 3, es decir, no toma en cuenta el identificador ni el propio octeto longitud. En el elemento de información usuario a usuario, el protocolo H.225 impone la restricción de utilizar dos octetos para la longitud en vez de uno. La estructura del contenido depende del propio elemento de información.

En algunos casos se aplica un mecanismo de agrupación y ampliación en el contenido de los elementos de información de longitud variable. En este mecanismo la primera cifra del número de octeto identifica a un octeto o a un grupo de octetos (ver tabla 8). Cada grupo de octetos es una entidad autocontenida. Un grupo de octetos se forma utilizando un mecanismo de ampliación. El mecanismo de ampliación consiste en ampliar un octeto (N) en el octeto o los octetos siguientes (Na, Nb, etc.) utilizando el bit 8 de cada octeto como bit de ampliación. El valor "0" de este bit indica que el grupo continúa en el octeto siguiente. El valor "1" de este bit indica que ese octeto es el último del grupo.

8	7	6	5	4	3	2	1	Bit/Octeto
ext. 0								N
ext. 0								Na
ext. 0								Nb
ext. 0								Nc
ext. 1								Nd
ext. 1								N+1
ext. 1								N+2

Tabla 8. Mecanismo de agrupación y ampliación.

3.4.5 Mensajes Establecimiento, Llamada en curso, Aviso, Conexión y Liberación completa

Los siguientes cuadro indican los elementos de información que pueden o deben estar presentes en los mensajes Establecimiento, Llamada en curso, Aviso,

Conexión y Liberación completa. La letra M significa obligatorio (mandatory), O opcional, CM condicionalmente obligatorio (conditionally mandatory).

Elemento de información	Situación H.225.0
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M
Envío completo	O
Capacidad portadora	M
Facilidad ampliada	O
Facilidad	O
Indicador de notificación	O
Visualización	O
Facilidad de teclado	O
Señal	O
Número de la parte llamante	O
Subdirección de la parte llamante	CM (llamadas con una red de conmutación de circuitos)
Número de la parte llamada	O
Subdirección de la parte llamada	CM (llamadas con una red de conmutación de circuitos)
Usuario a usuario	M

Tabla 9. Mensaje establecimiento.

Elemento de información	Situación H.225.0
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M
Capacidad portadora	O
Facilidad ampliada	O
Facilidad	O
Indicador de progresión	O
Indicador de notificación	O
Visualización	O
Usuario a usuario	M

Tabla 10. Mensaje llamada en curso.

Elemento de información	Situación H.225.0
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M
Capacidad portadora	O
Facilidad ampliada	O
Facilidad	O
Indicador de progresión	O
Indicador de notificación	O
Visualización	O
Señal	O
Usuario a usuario	M

Tabla 11. Mensaje Aviso.

Elemento de información	Situación H.225.0
Discriminador de protocolo	M

Elemento de información	Situación H.225.0
Referencia de llamada	M
Tipo de mensaje	M
Capacidad portadora	O
Facilidad ampliada	O
Facilidad	O
Indicador de progresión	O
Indicador de notificación	O
Visualización	O
Fecha/hora	O
Usuario a usuario	M

Tabla 12. Mensaje Conexión.

Elemento de información	Situación H.225.0
Discriminador de protocolo	M
Referencia de llamada	M
Tipo de mensaje	M
Causa	CM (si no se envía el elemento ReleaseCompleteReason dentro del elemento de información usuario a usuario
Facilidad	O
Indicador de notificación	O
Visualización	O
Señal	O
Usuario a usuario	M

Tabla 13. Mensaje Liberación Completa.

3.4.6 Elemento de información capacidad portadora

El elemento de información capacidad portadora tiene por objeto especificar el servicio solicitado. Se codifica como muestra la siguiente tabla.

8	7	6	5	4	3	2	1	Bit/Octeto
	Identificador del elemento de información capacidad portadora							
0	0	0	0	0	1	0	0	1
Longitud del contenido de capacidad portadora								2
ext. 1	Norma de codificación		Capacidad de transferencia de información					3
ext. 1	Modo de transferencia		Velocidad de transferencia de información					4
ext. 1	Multiplicador de velocidad							4.1*
ext. 0/1	Identificador de capa 1 0 1		Protocolo de capa 1 de información del usuario					5*
ext. 0/1	Sinc./ asínc	Negoc.	Velocidad de usuario					5a*
ext. 0/1	Velocidad intermedia		NIC en Tx	NIC en Rx	Control flujo en Tx	Control flujo en Rx	Reserva 0	5b*
ext. 0/1	Encabeza-miento/no encabeza-miento	Soporte de multitrama	Modo	Negoc. LLI	Asignador/ asignado	Negoc. dentro/ fuera de banda	Reserva 0	5b*
ext. 0/1	Número de bits de parada		Número de bits de datos		Paridad			5c*
ext. 1	Modo dúplex	Tipo de módem						5d*
ext. 1	Identificador de capa 2 1 0		Protocolo de capa 2 de información del usuario					6*
ext. 0	Identificador de capa 3 1 1		Protocolo de capa 3 de información del usuario					7*
ext. 0	Reserva 0 0 0			Información adicional de protocolo de capa 3 (bits más significativos)				7a*
ext. 1	Reserva 0 0 0			Información adicional de protocolo de capa 3 (bits más significativos)				7b*

Tabla 14. Elemento de información capacidad portadora [10].

El protocolo H.225 impone las siguientes restricciones:

- Capacidad de transferencia de información (octeto N.º 3):
  - El bit de extensión (bit 8) se pondrá a '1'.
  - La norma de codificación (bits 6, 7) se pondrá a '00' indicando 'UIT-T'.
  - Capacidad de transferencia de información (bits 0-5): Las llamadas que se originan en un punto extremo H.323 utilizarán este campo para indicar su deseo de efectuar una llamada audiovisual. Por tanto, el campo se pondrá a 'información digital sin restricciones', es decir, '01000' o a 'información digital restringida' es decir '01001'. Si ha de efectuarse una llamada sólo vocal, el terminal H.323 pondrá la capacidad de transferencia de información a 'conversación' (es decir '00000') o a 'audio a 3,1 kHz' (es decir '10000').

- Bit de extensión para el octeto N.º 4 (bit 8): Se pondrá a '0' si la velocidad de transferencia de información se pone a 'multivelocidad'; se pondrá a '1' en otro caso.
- Modo de transferencia – octeto abreviado N.º 4 (bits 6, 7): Especificará 'modo circuito', valor '00'.
- Velocidad de transferencia de información (bits 5 a 1): No se permite el valor '00000' (para el modo paquete) a menos que una pasarela se conecte a una red de paquetes. Las opciones son las siguientes:

Bits 5 4 3 2 1	
0 0 0 0 0	Este código se utilizará para llamadas en modo paquete.
1 0 0 0 0	64 kbit/s
1 0 0 0 1	2 × 64 kbit/s
1 0 0 1 1	384 kbit/s
1 0 1 0 1	1536 kbit/s
1 0 1 1 1	1920 kbit/s
1 1 0 0 0	Multivelocidad (velocidad básica de 64 kbit/s)

Los demás valores están reservados.

- Multiplicador de velocidad – octeto N.º 4.1: Estará presente si la velocidad de transferencia de información se pone a multivelocidad.
- Protocolo de capa 1(capa 1 de ISDN) – octeto N.º 5
  - El bit de extensión (bit 8) se pondrá a '1'.
  - Los bits 6 y 7 indicarán el identificador de capa 1 (capa 1 de ISDN), es decir, '01'.
  - Los bits 1 a 5 indicarán el protocolo de capa 1.
  - Los valores permitidos son G.711 (ley A '00011' y ley μ '00010') para indicar una llamada de sólo voz y H.221 y H.242 ('00101') para indicar una llamada videotelefónica H.323.
  - Los octetos N.º 5a, 5b, 5c, 5d no estarán presentes.
- Identificador de protocolo de capa 2 (capa 2 de ISDN) – octeto N.º 6: No estará presente.
- Identificador de protocolo de capa 3 (capa 3 de ISDN) – octeto N.º 7: No estará presente.

3.4.7 Elemento de información Causa

Indica la razón por la cual una llamada fue rechazada o desconectada. El contenido y el uso del elemento de información causa se definen en la Recomendación Q.850. La siguiente tabla muestra su codificación.

8	7	6	5	4	3	2	1	Bit/Octeto
	Identificador del elemento de información causa							
0	0	0	0	1	0	0	0	1
	Longitud del contenido de causa							2
0/1 ext	Norma de codificación		0 Reservado	Ubicación				3
1 ext	Recomendación							3a
1 ext	Valor Causa							4
Diagnostico (si existe)								5

Tabla 15. Elemento de información Causa.

- Norma de Codificación - octeto N° 3 (bits 7,6)
  Bits
  7 6
  0 0 Codificación estandarizada CCITT
  0 1 ISO/IEC
  1 0 Estándar Nacional
  1 1 Norma específica de la ubicación identificada
- Ubicación - octeto N° 3 (bits 4 a 1)
  Bits
  4 3 2 1
  0 0 0 0 Usuario
  0 0 0 1 Red privada que da servicio al usuario local
  0 0 1 0 Red pública que da servicio al usuario local
  0 0 1 1 Red de tránsito
  0 1 0 0 Red pública que da servicio al usuario distante
  0 1 0 1 Red privada que da servicio al usuario distante
  0 1 1 1 Red internacional
  1 0 1 0 Red que se extiende más allá del punto de interfuncionamiento
  Los demás valores están reservados
- Recomendación - octeto N° 3a
  Bits
  7 6 5 4 3 2 1
  0 0 0 0 0 0 0 Q.931
  0 0 0 0 0 1 1 X.21
  0 0 0 0 1 0 0 X.25
  0 0 0 0 1 0 1 Redes públicas móviles terrestres Q.1031/Q.1051
  Si este octeto es omitido se asume la recomendación Q.931.
- Valor Causa - octeto N° 4: Este valor indica la causa de la terminación de la llamada. Estos son algunos de sus posibles valores: Liberación normal de la llamada, Usuario ocupado, No hay respuesta del usuario, No hay respuesta del usuario, Abonado ausente
  Llamada rechazada, Red fuera de servicio, Fallo temporal, Capacidad portadora no implementada, Contenido de elemento de información no válido.

- Diagnostico - octeto N° 5: Dependiendo de la causa, se puede incluir un octeto de diagnostico en donde se proporcionan detalles de la causa de la terminación de la llamada.

3.4.8 Elemento de información Visualización

La finalidad del elemento de información visualización es suministrar información que puede ser visualizada por el usuario. La información contenida en este elemento se codifica en caracteres del Alfabeto Internacional N.° 5. El elemento de información visualización se codifica como se muestra en la tabla 16.

8	7	6	5	4	3	2	1	Bit/Octeto
0	Identificador del elemento de información visualización							1
	0	1	0	1	0	0	0	
Longitud del contenido de visualización								2
0	Información de visualización (caracteres del IA5)							3 etc.

Tabla 16. Elemento de información Visualización.

3.4.9 Elemento de información Usuario a usuario

El elemento de información usuario a usuario será utilizado por todas las entidades H.323 para transportar información relacionada con H.323 y decodificada en PER de la sintaxis ASN.1. El elemento de información usuario a usuario se codifica como se muestra en la tabla 17.

8	7	6	5	4	3	2	1	Bit/Octeto
0	Identificador del elemento de información usuario a usuario							1
	1	1	1	1	1	1	0	
Longitud del contenido de usuario a usuario								2
Discriminador de protocolo								3
Información de usuario								4 etc.

Tabla 17. Elemento de información Usuario a usuario.

El protocolo H.225 impone las siguientes restricciones:

- Longitud de contenido de usuario a usuario: Serán 2 octetos en vez de 1.
- Discriminador de protocolo: Indicará información de usuario codificada ('00000101') X.208 y X.209 (ASN.1). (Esto se toma de la revisión 1993 de la Recomendación Q.931, que hace referencia a las anteriores revisiones de ASN.1. Las referencias correctas a ASN.1 son la Recomendación X.680 (sintaxis) y la Recomendación X.691 (PER)).
- Información de usuario: Contendrá una estructura ASN.1 que contiene información pertinente H.323.

4. Protocolo de Iniciación de Sesiones (SIP)

El Protocolo de Iniciación de Sesiones es un protocolo de control a nivel de aplicación que permite la creación, modificación y terminación de sesiones multimedia con uno o más participantes. Estas sesiones incluyen llamadas telefónicas por Internet, distribuciones multimedia y conferencias multimedia.

Fue desarrollado por el Grupo de Trabajo en Ingeniería de Internet (IETF). Por lo tanto, SIP se caracteriza porque sus promotores tienen sus raíces en la comunidad IP y no en la industria de las telecomunicaciones. La primera versión SIP propuesta como estándar fue definida en la recomendación RFC2543 (SIP/1.0) y fue publicada en febrero de 1999. En Junio de 2002 surgió la segunda versión RFC3261 (SIP/2.0). SIP no es un sistema de comunicaciones integrado verticalmente. SIP es más bien un componente que puede ser usado con otros protocolos para construir una arquitectura multimedia completa. Sin embargo, la funcionalidad básica y operación de SIP no depende de ninguno de estos protocolos. SIP no proporciona servicios, más bien SIP provee bases que pueden ser usadas para implementar diferentes servicios.

4.1 Definiciones

A continuación se mencionan algunos términos que se definen en SIP.

- Sesión: Una sesión multimedia es un conjunto de emisores y receptores multimedia y los flujos de datos que fluyen de emisores a receptores.
- Mensaje: Es la data enviada entre elementos SIP como parte del protocolo. Los mensajes SIP son solicitudes o respuestas.
- Solicitud: Es un mensaje SIP enviado desde un cliente a un servidor con el propósito de invocar una operación particular.
- Método: Es la función principal que una solicitud desea invocar en un servidor. El método es transportado en propio el mensaje de solicitud. Ejemplos de método son “INVITE” y “BYE”.
- Respuesta: Es un mensaje SIP enviado desde un servidor hacia un cliente para indicar el estado de una solicitud enviada desde el cliente hacia el servidor.
- Cliente: Es un elemento de red que envía solicitudes SIP y recibe respuestas SIP. Los clientes pueden o no interactuar directamente con un usuario humano. Los agentes de usuario clientes y servidores proxy son ejemplos de clientes.
- Servidor: Es un elemento de red que recibe solicitudes para prestar algún servicio y envía de vuelta respuestas a esas solicitudes. Ejemplos de servidores son servidores proxy, agente de usuarios servidores, servidores de redirección y servidores de registro.
- Agente de usuario cliente (UAC): Es una entidad lógica que crea y envía una nueva solicitud. El papel de UAC permanece sólo en la duración de la transacción. En otras palabras, si un componente de software inicia una

solicitud, actúa como UAC en la duración de la transacción. Si más tarde recibe una solicitud, asume el rol de agente de usuario.

- Agente de usuario servidor: Es una entidad lógica que genera una respuesta a una solicitud SIP. La respuesta acepta, rechaza, o redirecciona la solicitud. Este rol permanece sólo en la duración de la transacción. En otras palabras, si un componente de software responde una solicitud, actúa como un UAS mientras dure la transacción. Si más tarde genera una solicitud, asume el rol de agente usuario cliente.
- Agente de usuario (UA): Es una entidad lógica que puede actuar tanto como un agente de usuario cliente y como un agente de usuario servidor.
- Transacción: Una transacción ocurre entre un cliente y un servidor y comprende todos los mensajes desde la primera solicitud enviada desde el cliente al servidor, hasta una respuesta final enviada del servidor al cliente. Si la solicitud es "INVITE" y la respuesta final no es una respuesta 2xx, la transacción también incluye una solicitud "ACK" a esa respuesta. La solicitud ACK para una respuesta 2xx a una solicitud INVITE es una transacción separada.
- Respuesta final: Es una respuesta que termina una transacción al contrario de las respuestas provisionales. Todas las respuestas 2xx, 3xx, 4xx, 5xx y 6xx son respuestas finales.
- Respuesta provisional: Es una respuesta usada por el servidor para indicar progreso, pero que no termina la transacción. Las respuestas 1xx son provisionales, las otras son consideradas finales.
- Diálogo: Es una relación entre dos agentes de usuario que persiste por algún tiempo. Un diálogo es establecido por un mensaje SIP, tal como una respuesta 2xx a una solicitud "INVITE". Un diálogo es identificado por un identificador de llamada, una etiqueta local y una etiqueta remota.
- Servidor proxy: Es una entidad intermedia que actúa tanto como servidor como cliente con el propósito de hacer solicitudes en nombre de otros clientes. Un servidor proxy principalmente juega el papel de enrutador, lo cual significa que su trabajo es asegurar que una solicitud sea enviada a una entidad más cercana al usuario destino. Los servidores proxy también son útiles para imponer políticas (por ejemplo, asegurar que un usuario tenga permitido hacer llamadas). Un servidor proxy interpreta, y, si es necesario, rescribe partes específicas de un mensaje de solicitud antes de pasarlo.
- Identificador uniforme de recursos (URI): Es una cadena compacta de caracteres usada para identificar un recurso abstracto o físico.

4.2 Indicador uniforme de recursos (URI)

Un URI SIP identifica un recurso de comunicación. El esquema "sip:" se rige por el estándar RFC 2396. Este esquema usa una forma similar al "mailto" URL. La forma general de un URI SIP es:

sip:usuario:contraseña@host:puerto;parámetros-uri?cabeceras

El uso de los componentes de un URI depende del contexto. Estos elementos tienen los siguientes significados:

- usuario: Es el identificador de un recurso particular en el host que está siendo direccionado. En este contexto el término "host" frecuentemente se refiere a un dominio. La información de usuario de un URI consiste de este campo usuario, el campo contraseña y el símbolo @. La información de usuario de un URI es opcional y puede estar ausente cuando el host destino no tiene una noción de usuario o cuando el host en sí mismo es el recurso que está siendo identificado.
- contraseña: Es una contraseña asociada con el usuario. Su uso dentro de un URI no es recomendado por razones de seguridad.
- host: Es el host que provee el recurso SIP. Contiene un nombre de dominio o una dirección numérica IPv4 o IPv6.
- puerto: Es el número del puerto donde la solicitud será enviada. El valor por defecto para un URI SIP es 5060 y para un URI SIPS es 5061.
- parámetros-uri: Son parámetros que afectan una solicitud construida desde el URI. Estos parámetros son separados por punto y coma. Los parámetros URI toman la forma nombre-de-parámetro = valor-del-parámetro.
- cabeceras: Campos de cabecera que serán incluidos en una solicitud construida desde el URI.

4.3 Mensajes SIP

SIP es un protocolo basado en texto y usa el conjunto de caracteres UTF-8 (8-bit Unicode Transformation Format). Excepto por la diferencia en el conjunto de caracteres, la sintaxis de muchos de los mensajes y campos de cabeceras SIP es idéntica a HTTP/1.1 (RFC 2616). Sin embargo, SIP no es una extensión de HTTP. Un mensaje SIP es una solicitud desde un cliente a un servidor o una respuesta desde un servidor a un cliente. Tanto las solicitudes como las respuestas usan el formato básico RFC 2822 (Formato de Mensaje de Internet). Ambos poseen la siguiente estructura:

Línea de inicio
Cabecera del mensaje
Línea Vacía
Cuerpo del mensaje (opcional)

Tabla 18. Estructura de un mensaje SIP.

La línea vacía debe estar presente incluso si no existe el cuerpo de mensaje. La línea de inicio puede ser una línea de solicitud o una línea de estado dependiendo de si el mensaje es una solicitud o una respuesta, respectivamente.

4.3.1 Solicitudes

Las solicitudes SIP se distinguen por tener una Línea-de-Solicitud en la línea de inicio. Una Línea-de-Solicitud contiene un nombre de método, un URI-de-solicitud, y la versión del protocolo separados por un espacio simple:

Línea-de-Solicitud = Método URI-de-Solicitud Versión-SIP

- Método: Es la función principal que una solicitud desea invocar en un servidor. La especificación RFC 3261 define seis métodos:
  - "REGISTER" para registro de información de contacto.
  - "INVITE" para iniciar una sesión.
  - "ACK" para indicar el reconocimiento de recepción de una respuesta.
  - "CANCEL" para cancelar sesiones.
  - "BYE" para terminar sesiones.
  - "OPTIONS" para servicios de indagación de capacidades.
- URI-de-Solicitud: Indica el usuario o servicio al cual esta solicitud esta siendo dirigida.
- Versión-SIP: Tanto los mensajes de solicitud como los de respuesta incluyen la versión de SIP en uso. Para ser consistente con el estándar RFC 3261, las aplicaciones que envíen mensajes deben incluir "SIP/2.0" en Versión-SIP.

4.3.2 Respuestas

Las respuestas SIP se diferencian de solicitudes por tener una Línea-de-Estado como su línea de inicio. Una Línea-de-Estado consiste de la versión del protocolo seguido por un número Código-de-Estado y su frase textual asociada, con cada elemento separado por un espacio simple:

Línea-de-Estado = Versión-SIP Código-de-Estado Frase

- Código-de-Estado: Es un código entero de 3 dígitos que indica el resultado de un intento de entender y satisfacer una solicitud. El primer dígito del Código-de-Estado define la clase de respuesta. Cualquier respuesta con un código de estado entre 100 y 199 es calificada como una respuesta 1xx, una respuesta con código de estado entre 200 y 299 como una respuesta 2xx, y así sucesivamente. SIP/2.0 permite seis valores para el primer dígito:
  - 1xx – Provisional: Solicitud recibida, se continúa a procesar la solicitud.

    - 2xx – Satisfactorio: La acción fue recibida, entendida y aceptada satisfactoriamente.
    - 3xx – Redirección: Otras acciones deben ser tomadas para completar la solicitud.
    - 4xx – Error del cliente: La solicitud contiene mala sintaxis o no puede ser ejecutada en este servidor.
    - 5xx – Error del servidor: El servidor falló en ejecutar una solicitud aparentemente válida.
    - 6xx – Falla general: La solicitud no puede ser ejecutada por ningún servidor.
- Frase: Tiene la intención de dar una breve descripción textual al Código-de-Estado. La Frase está destinada para el usuario humano. Un cliente no está obligado a examinar o mostrarla. A pesar de que SIP sugiere valores específicos de Frase, las implementaciones pueden escoger otros textos, por ejemplo, en el algún idioma en particular.

En la siguiente tabla se muestran algunas respuestas SIP con sus códigos y descripciones correspondientes.

Clase	Código	Descripción
1xx (Provisional)	100	Intentando
	180	Repicando
2xx (Satisfactorio)	200	Satisfactorio (OK)
3xx (Redirección)	301	Movido permanentemente
	302	Movido temporalmente
4xx (Error del cliente)	400	Mala solicitud
	401	Desautorizado
	404	No encontrado
	405	Método no permitido
	407	Autenticación requerida por el servidor proxy
	483	Demasiados saltos
	486	Ocupado aquí
5xx (Error del servidor)	500	Error interno del servidor
	501	No implementado
	505	Versión no soportada
6xx (Falla global)	600	Ocupado en todas partes
	604	No existe en ninguna parte

Tabla 19. Respuestas SIP

4.3.3 Cabecera

La cabecera del mensaje está compuesta de una serie de campos. La presencia de un campo en particular depende del tipo de mensaje. Cada campo de cabecera consiste de un nombre del campo seguido por dos puntos (":") y el valor del campo:

Nombre-del-campo: valor-del-campo

El orden relativo de los campos de cabecera con diferentes nombres de campo no es significativo. Sin embargo, es recomendado que los campos de cabecera que son necesitados en el procesamiento de los servidores proxy (por ejemplo: Via, Route, Record-Route, Proxy-Require, Max-Forwards y Proxy-Authorization) aparezcan en el tope superior del mensaje facilitando su rápido análisis. El orden relativo de las filas de campos de cabecera con el mismo nombre de campo si es importante.

El formato de valor-del-campo es definido por nombre-del-campo. Muchos campos de cabecera existentes adherirán a la forma general un valor seguido por una secuencia de parámetros. Un parámetro consiste en un nombre y un valor separados por punto y coma:

nombre-del-campo: valor-del-campo *(;nombre-del-parámetro=valor-del-parámetro)

Aunque se puede adjuntar un número arbitrario de parámetros, un nombre de parámetro no debe aparecer más de una vez

4.3.4 Cuerpo del mensaje

Los mensajes pueden contener cuerpos de mensaje. La interpretación del cuerpo depende del método solicitado. El tipo de medio Internet del cuerpo del mensaje puede ser especificado por el campo de cabecera "Content-Type". Si el cuerpo está sujeto a alguna codificación, tal como alguna compresión, entonces esto debe ser indicado por el campo de cabecera "Content-Encoding", en otro caso, "Content-Encoding" debe ser omitido. La longitud del cuerpo en Octetos es provista por el campo de cabecera Content-Length.

4.4 Establecimiento y finalización de una llamada SIP

La siguiente figura muestra los procesos de establecimiento y finalización de una llamada en SIP. Se trata de un caso típico que consiste en una llamada entre dos usuarios del mismo dominio a través de un servidor proxy que presta servicio en ese dominio. Las flechas azules indican solicitudes mientras que las rojas indican respuestas.

La solicitud “INVITE” (1) es una solicitud que puede originar la creación de un diálogo (de hecho en RFC 3261, es la única que lo puede hacer). Sin embargo, puesto que no se ha establecido uno, esta es una solicitud fuera de diálogo y su construcción se explica en 4.5.1.1 (Generación de solicitudes). También es una solicitud que inicia una sesión y por lo tanto, también se siguen las indicaciones en 4.7 (Inicio de sesiones). En este caso, el servidor proxy solicita información de autenticación por medio de la respuesta 407 (2), la cual se construye como se indica en 4.5.2.1 (Generación de respuestas). Luego el usuario A envía una solicitud “ACK” (3) para indicar el reconocimiento de esta respuesta. Esta solicitud también es una solicitud fuera de diálogo y se construye según 4.5.1.1. Seguidamente el usuario A procede a enviar nuevamente la solicitud “INVITE” (4), pero esta vez con la información de autenticación solicitada. Una vez más, esta es una solicitud fuera de diálogo y por lo tanto, se construye como indica la sección 4.5.1.1. El servidor proxy recibe esta solicitud, la maneja tal como se explica en 4.9 (Comportamiento de los servidores proxy) y la encamina hacia el usuario B (5). El usuario B responde con una respuesta 100 (Intentando) (6) para indicar al servidor proxy que se están tomando acciones para establecer la sesión. Esta respuesta se diferencia de otras respuestas provisionales en que nunca es reenviada por un servidor proxy. Esta respuesta se construye como se explica en 4.5.2.1. Luego el usuario B envía una respuesta 180 (Repicando) (7) para indicar al usuario A que se está alertando al usuario B. Esta repuesta también se construyen como se explica en 4.5.2.1 y como se indica en la sección 4.6 (Diálogos), porque es una respuesta que crea un diálogo temprano y se debe almacenar el estado de este diálogo. Esta respuesta es encaminada por el servidor proxy según lo especificado en 4.9 (8). Al llegar esta respuesta al usuario A, se almacena el estado del diálogo según 4.6.

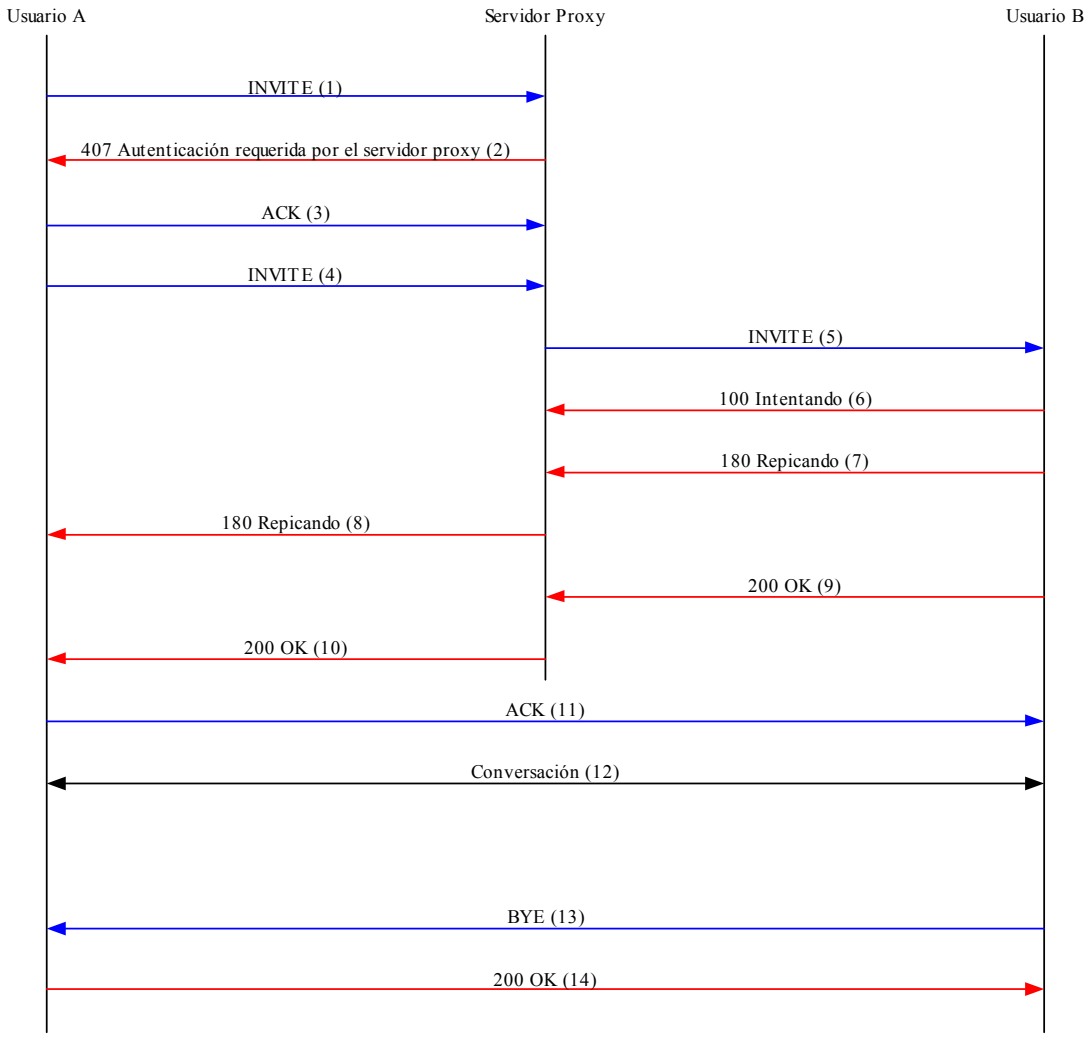


Figura 3. Establecimiento y finalización de una llamada SIP.

Una vez que el usuario B decide aceptar la sesión, envía una respuesta 200 ("OK") (9) al usuario A. Esta respuesta debe ser construida según se explica en 4.5.2.1, en 4.6 (ya que convierte el diálogo temprano en un diálogo confirmado) y en 4.7 (Inicio de Sesiones, porque inicia una sesión). Esta respuesta es encaminada por el servidor proxy hacia el usuario A según 4.9. Al llegar esta respuesta al usuario A, el diálogo queda completamente establecido. Por consiguiente, cualquier solicitud que se envíe a partir de este momento es una solicitud dentro de diálogo. Esto incluye a la solicitud "ACK" (11), la cual es enviada para indicar que se reconoce la respuesta 200, y se debe construir como indica 4.6. Después que se envía esta solicitud se establece la sesión (12). Para finalizar la sesión y el diálogo, se envía una solicitud BYE. Esta solicitud también está dentro del diálogo y se construye según 4.6.

4.5 Comportamiento general de los agentes de usuario

Cuando un agente de usuario envía una solicitud, inicia una transacción y se comportará como un agente de usuario cliente mientras dure esta transacción. El agente de usuario que responda a esta solicitud se comportará como un agente de usuario servidor. Los procedimientos de los UAC (agentes de usuario clientes) y los UAS (agentes de usuario servidores) dependen principalmente de dos factores. Primero, depende de si la solicitud o respuesta está o no dentro del diálogo, y segundo, depende del método de la solicitud. En esta sección se describe las reglas independientes del método que rigen el procesamiento por parte de los UAC y UAS de las solicitudes que están fuera del diálogo. Esto incluye, por supuesto, las solicitudes que establecen el diálogo. En RFC 3261 la única solicitud que puede originar la creación de un dialogo es la solicitud "INVITE".

4.5.1 Comportamiento de los agentes de usuario clientes

4.5.1.1 Generación de solicitudes

Una solicitud SIP válida formulada por un UAC debe, como mínimo, contener los siguientes campos de cabecera: "To", "From", "CSeq", "Call-ID", "Max-Forwards", y "Via". Todos estos campos de cabecera son obligatorios en todas las solicitudes SIP y son los bloques fundamentales de un mensaje SIP. A continuación se describen estos campos de cabecera junto al URI-de-Solicitud de la línea de solicitud.

- URI-de-Solicitud: La URI-de-Solicitud inicial debería ser el valor de URI en el campo To.
- "To": El campo de cabecera "To" específica el recipiente lógico deseado de la solicitud, o la dirección de registro del usuario o el recurso destino de la solicitud. El campo de cabecera "To" contiene un URI y permite un nombre para visualización. Una solicitud fuera de diálogo no debe contener un parámetro "tag" (etiqueta) para el campo "To". El parámetro "tag" en el campo "To" de una solicitud es uno de los identificadores del diálogo. Puesto que un diálogo no ha sido establecido, no existe una etiqueta.
- "From": Indica la identidad lógica del iniciador de la solicitud. Al igual que el campo "To", contiene un URI y opcionalmente un nombre para visualización. El campo de cabecera "From" puede contener un nuevo parámetro "tag" (etiqueta) elegido por el UAC.
- "Call-ID": Actúa como un único identificador para agrupar una serie de mensajes. Debe ser el mismo para todos las solicitudes y respuestas enviadas por cualquier UA en un diálogo. Cuando una solicitud es reenviada después de alguna repuesta de falla, esa solicitud reenviada no es considerada una nueva solicitud, y por lo tanto, no necesita un nuevo "Call-ID".

- "CSeq": Sirve para identificar y ordenar transacciones. Contiene un número de secuencia y un método. Para solicitudes fuera de diálogo que no sean de registro, el valor del número de secuencia es arbitrario. El método debe concordar con el de la solicitud. El valor del número de secuencia debe ser expresable como un entero sin signo de 32 bits y debe ser menor que 2^{31}.
- "Max-Forwards": Sirve para limitar el número de saltos que una solicitud puede transitar en el camino a su destino. Consiste en un entero que es disminuido en uno en cada salto. Si el valor de "Max-Forwards" llega a ser 0 antes que la solicitud alcance su destino, la solicitud será rechazada con una respuesta de error 483 (Demasiados Saltos). Un UAC debe insertar un campo "Max-Forwards" en cada solicitud que él origina con un valor que debería ser 70. Este número fue escogido lo suficientemente grande para garantizar que una solicitud no sea borrada en cualquier red SIP sin lazos, pero no tan grande como para consumir los recursos de los servidores proxy cuando ocurre un lazo.
- Via: Indica el transporte usado para la transacción e identifica la ubicación donde la respuesta debe ser enviada. Cuando un UAC crea una solicitud debe insertar un campo "Via". Contiene el nombre y versión del protocolo que deben ser SIP y 2.0 respectivamente. El campo "Via" también debe contener un parámetro llamado "branch". Este parámetro es usado para identificar la transacción creada por la solicitud y también es usado por los servidores proxy para detectar lazos. El parámetro "branch" debe ser único para todas las solicitudes enviadas por el UA. La excepción a esta regla son la solicitud "CANCEL" y la solicitud "ACK" para una respuesta no 2xx. Una solicitud "CANCEL" tendrá el mismo valor del parámetro "branch" que el de la solicitud que está cancelando. Una solicitud "ACK" para una respuesta no 2xx también tendrá el mismo parámetro "branch" que la solicitud INVITE cuya respuesta reconoce. El identificador "branch" insertado por un elemento que obedezca la especificación RFC 3261 siempre debe comenzar con los caracteres "z9hG4bK".
- Contact: Provee un URI que puede ser usado para contactar esa instancia específica del UA para solicitudes subsiguientes. El campo "Contact" debe estar presente en una solicitud que puede resultar en el establecimiento de un diálogo y contener exactamente un URI.
- Componentes de mensaje adicionales: Después que una solicitud ha sido creada y los campos de cabecera descritos anteriormente han sido construidos correctamente, cualquier campo de cabecera opcional puede ser añadidos, como son los campos de cabecera específicos del método.

4.5.1.2 Procesamiento de respuestas

La mayor parte del procesamiento de respuesta es específica del método. Sin embargo, existen algunos comportamientos generales independientes del método que se describen en la especificación RFC 3261. Por ejemplo, si en la respuesta está

presente más de un campo “Via”, el UAC debería descartar el mensaje. La presencia de un campo “Via” adicional que precede el creador de la solicitud sugiere que el mensaje fue sacado de su ruta o que posiblemente fue alterado. Si el UAC recibe una respuesta 401 (Desautorizado) o 407 (Autenticación requerida por el servidor proxy), el UAC debería reintentar la solicitud pero ahora con la información de autenticación. Para ello, se debe añadir el campo “Proxy-Authenticate”, el cual contiene información de autenticación. En este caso, la respuesta se envía nuevamente constituyendo una nueva transacción y debería tener el mismo valor de “Call-ID”, “To” y “From” de la solicitud previa, pero el valor de “CSeq” debería contener un nuevo número de secuencia que igual al valor de la solicitud anterior incrementado en uno.

4.5.2 Comportamiento de los agentes de usuario servidores

Cuando una solicitud fuera de diálogo es recibida por un UAS, hay un conjunto de normas de procedimientos que se deben seguir, independientemente del método. Una vez que una solicitud es autenticada, el UAS debe inspeccionar el método de la solicitud, y revisar si lo soporta o no y, en caso de que no lo soporte, generar una respuesta 405 (Método no permitido). Luego el UAS debe inspeccionar la cabecera. Si el UAS no entiende un campo de cabecera en una solicitud, debe ignorar ese campo de cabecera y continuar procesando el mensaje. Un UAS debe ignorar cualquier campo de cabecera defectuoso que no sea necesario para procesar el mensaje. Seguidamente, el UAS debe examinar el cuerpo del mensaje. Si existe algún cuerpo cuyo tipo o codificación sea desconocido y si el cuerpo del mensaje no es opcional, el UAS debe rechazar la solicitud con una respuesta 415 (Tipo de medio no soportado). Asumiendo que todas las revisiones anteriores fueron satisfactorias, el UAS debe ejecutar el procesamiento específico del método. Una vez procesada la solicitud, el UAS procede a formular la respuesta.

4.5.2.1 Generación de la respuesta

Un UAS no debería enviar una respuesta provisional para una solicitud que no sea “INVITE”. Al contrario, un UAS debería generar lo más pronto posible una respuesta final a una solicitud que no sea “INVITE”. Los campos “From”, “Call-ID” y “Cseq” de la respuesta deben ser iguales a los de la solicitud. Los valores del campo “Via” de la respuesta deben ser iguales a los de la solicitud y mantener el mismo orden.

Si una solicitud contenía una etiqueta (“tag”) en el campo “To”, el campo “To” en la respuesta debe ser igual al de la solicitud. Sin embargo, si el campo “To” en la solicitud no contiene una etiqueta, el URI en el campo “To” en la respuesta debe ser igual al URI en el campo “To” de la solicitud; adicionalmente, el UAS debe añadir una etiqueta al campo “To” en la respuesta (con la excepción de la respuesta 100 (“Trying”), en la cual una etiqueta puede estar presente o no). Esto sirve para identificar el UAS que está respondiendo, resultando posiblemente en un componente

de un identificador de diálogo. La misma etiqueta debe ser usada para todas las respuestas a esa solicitud, tanto final como provisional (otra vez exceptuando la respuesta 100 (“Trying”)).

4.6 Diálogos

Un concepto clave para los agentes de usuario es el diálogo. Un diálogo representa una relación SIP punto a punto entre dos agentes de usuario que prevalece por algún tiempo. El diálogo facilita la secuencia de mensajes entre los agentes de usuario y el correcto enrutamiento entre ellos. Representa un contexto para interpretar mensajes SIP. El diálogo es identificado dentro de cada UA con una identificador (ID) de diálogo, el cual consiste de un valor “Call-ID”, una etiqueta local y una etiqueta remota. El identificador de diálogo en cada UA involucrado en el diálogo no es el mismo. Específicamente, la etiqueta local en un UA es idéntica a la etiqueta remota en el otro UA y viceversa. Las etiquetas facilitan la generación de un identificador de diálogo único.

Las reglas de establecimiento del ID de diálogo de un mensaje dependen de si el elemento SIP es un UAC o un UAS. Para un UAC, el valor “Call-ID” del diálogo es el “Call-ID” del mensaje, la etiqueta remota es la etiqueta (“tag”) en el campo “To” del mensaje y la etiqueta local es la del campo “From”. Para un UAS, el valor “Call-ID” del ID del diálogo es el “Call-ID” del mensaje, la etiqueta remota es la etiqueta (“tag”) del campo “From” del mensaje y la etiqueta local es la etiqueta (“tag”) del campo “To”.

Un diálogo contiene ciertas piezas de estado necesarias para la transmisión dentro del diálogo. Este estado consiste del ID del diálogo, un número de secuencia local (usado para ordenar solicitudes desde el UA al UA opuesto), una número de secuencia remoto (usado para ordenar solicitudes desde el UA opuesto al UA), un URI local, un URI remoto, destino remoto, una bandera booleana llamada “secure” y un conjunto de ruta, el cual es una lista ordenad de URIs. El conjunto de ruta es la lista de servidores que necesitan ser atravesados para enviar una solicitud al UA opuesto. Todos estos parámetros son usados para construir solicitudes y analizar respuestas dentro del diálogo. Un diálogo puede estar también en el estado “temprano”, el cual ocurre cuando es creado con una respuesta provisional, y luego es transformado a un estado “confirmado” cuando una llega respuesta final 2xx. Si en un diálogo temprano, se recibe otras respuestas, o si no se recibe ninguna respuesta, el dialogo temprano termina.

Los diálogos son creados a través de la generación de respuestas que no sean de fallos para solicitudes con métodos específicos. En la especificación RFC 3261, sólo las respuestas 2xx y las respuestas 101-199 con una etiqueta en el campo “To”, donde la solicitud fue “INVITE”, establecerán un diálogo. Una vez que el UAS responde una solicitud con una respuesta que establece un diálogo, el UAS construye el estado del diálogo. En el UAC, el estado se construye cuando se recibe esta

respuesta. Este estado debe ser mantenido durante la duración del diálogo. En este estado se almacenan una serie de información que será utilizada en la construcción y manejos de solicitudes y respuestas que pertenezcan al diálogo. Un ejemplo de la información que se almacena es el conjunto de ruta especificado en la cabecera "Record-Route" de la solicitud, la cual es la lista de servidores que necesitan ser atravesados para enviar una solicitud al UA opuesto y que debe ser almacenado en el estado del diálogo.

Cuando un UAS responde a una solicitud con una respuesta que establece un diálogo (tal como una respuesta 2xx a una solicitud "INVITE"), el UAS debe copiar todos los campos Record-Route de la solicitud dentro de la respuesta y manteniendo el mismo orden. Debe añadir una campo "Contact" a la respuesta. El campo "Contact" contiene una dirección donde el UAS le gustaría ser contactado en subsiguientes solicitudes dentro del diálogo (lo cual incluye la solicitud "ACK" para una respuesta 2xx en el caso de la solicitud "INVITE").

Seguidamente el UAS construye el estado del diálogo. Este estado debe ser mantenido durante la duración del diálogo. El conjunto de ruta es construido con los campos Record-Route manteniendo el mismo orden. El destino remoto es puesto igual al URI del campo "Contact" de la solicitud. El numero de secuencia remota se coloca igual a al valor del campo "CSeq" de la solicitud. El número de secuencia local debe estar vacío. El valor de "Call-ID" de la solicitud será el identificador de llamada del ID del diálogo. La etiqueta local del ID del diálogo debe ser igual a la etiqueta "tag" del campo "To" en la solicitud, y la etiqueta remota debe ser igual a la etiqueta "tag" del campo "From". El URI remoto debe ser puesto igual al URI del campo "From", y el URI local igual al URI en el campo "To".

Por otra parte, cuando el UAC recibe una respuesta que establece un diálogo (tal como una respuesta 2xx a una solicitud "INVITE"), construye el estado del diálogo, el cual debe ser mantenido mientras dure el diálogo. El conjunto de ruta debe ser construido a partir de los campos Record-Route de la respuesta pero en orden inverso. El destino remoto es puesto igual al URI del campo "Contact" de la solicitud. El número de secuencia local debe ser puesto igual al valor de "CSeq" de la solicitud. El número de secuencia remota debe estar vacío (éste es establecido cuando el UA remoto envíe una solicitud dentro del diálogo). El componente identificador de llamada del ID del diálogo es colocado igual al valor "Call-ID" de la solicitud. La etiqueta local debe ser igual a la etiqueta "tag" del campo "From" en la solicitud, mientras que la etiqueta remota debe ser igual a la etiqueta "tag" del campo "To" en la respuesta. El URI remoto debe ser puesto igual al URI del campo "To" y el URI local igual al URI del campo "From".

Una solicitud dentro de un diálogo es construida usando mucho de los componentes del estado del diálogo almacenado. El URI en el campo "To" de la solicitud debe ser el URI remoto del diálogo. La etiqueta "tag" en el campo "To" debe ser la etiqueta remota del diálogo. El URI del campo "From" debe ser el URI

local del estado del diálogo. La etiqueta "tag" en el campo "From" debe ser la etiqueta local del diálogo. El valor "Call-ID" de la solicitud debe ser el "Call-ID" del diálogo. Las solicitudes dentro del diálogo contienen números "CSeq" que deben ser incrementados en uno (excepto en las solicitudes "ACK" y "CANCEL", que contienen un "CSeq" igual a la solicitud siendo reconocida o cancelada). Por lo tanto, si el número de secuencia local no está vacío, debe ser incrementado en uno y este valor debe ser colocado en el campo "CSeq" de la solicitud. Si está vacío, el valor inicial de "CSeq" debe ser creado usando las indicaciones de la especificación RFC 3261. El método del campo "Cseq" debe ser el mismo método de la solicitud. El UAC usa el destino remoto y el conjunto de ruta para construir el URI-de-la-Solicitud y los campos "Route" de la solicitud, respectivamente. Un AUC debería incluir el mismo campo "Contact" usado en las solicitudes anteriores. El resto de la solicitud es construido como indica la sección 4.5.1.1 (como una solicitud fuera de diálogo, para los campos "Via" y "Max-Forwards").

La construcción de repuestas en el diálogo se guía por el procedimiento en 4.5.2.1.

El mecanismo para terminar diálogos confirmados es específico del método. En la especificación RFC 3261, la solicitud con el método "BYE" termina una sesión y el diálogo asociada con ella. Esta solicitud es una solicitud dentro del diálogo y debe ser construida como tal.

4.7 Inicio de sesiones

Cuando un agente de usuario cliente decide iniciar una sesión (por ejemplo, audio, video o un juego), formula una solicitud "INVITE". La solicitud INVITE pide a un servidor establecer una sesión. Esta solicitud puede ser reenviada por servidores proxy, eventualmente llegando a uno o más UAS que pueden aceptar la invitación. Estos UAS frecuentemente necesitarán preguntar al usuario humano si acepta la invitación. Después de algún tiempo, estos UAS pueden aceptar la invitación (lo que significa que la sesión es establecida) enviando una respuesta 2xx. Si la invitación no es aceptada, se envía una respuesta 3xx, 4xx, 5xx o 6xx, dependiendo de la razón del rechazo. Antes de enviar una respuesta final, el UAS también puede enviar respuestas provisionales (1xx) para avisar al UAC del progreso de contactar al usuario llamado, por ejemplo una respuesta 100 (Intentando) para indicar que la solicitud ha sido recibida y que se están tomando acciones o una respuesta 180 (Repicando) para indicar que el teléfono esta sonando.

Después de recibir una o más repuestas provisionales, el UAC esperará por una o más respuesta. Debido a la prolongada cantidad de tiempo que puede tomar recibir una respuesta final a la solicitud "INVITE" (ya que depende de cuando el usuario llamado ejecuta una acción), el mecanismo de fiabilidad para una transacción "INVITE" difiere de otras solicitudes. Esto quiere decir que el tiempo de expiración

de una solicitud “INVITE” es mucho mayor que los de las demás solicitudes. Una vez que recibe una respuesta final, el UAC necesita enviar una solicitud “ACK” por cada respuesta final que recibe.

El procedimiento para enviar esta solicitud “ACK” depende del tipo de respuesta. En el caso de respuestas 2xx, el UAC debe generar una solicitud “ACK” por cada respuesta 2xx recibida. Los campos de esta solicitud ACK son construidos en la misma forma que se hace para una solicitud dentro de un diálogo con la excepción del campo “CSeq”. El número de secuencia del campo “CSeq” debe ser el mismo que el de la solicitud “INVITE” que se reconoce, pero el método de “CSeq” debe ser “ACK”.

Una respuesta 2xx para una solicitud “INVITE” establece una sesión, y también crea un diálogo entre el UA que envió la solicitud “INVITE” y el UA que genera la respuesta 2xx. Puesto que la solicitud “INVITE” inicial representa una solicitud fuera de diálogo, su construcción sigue los procedimientos de la Sección 4.5.1.1. Un UAC también puede añadir otros campos de cabecera, como por ejemplo, “User-Agent” el cual provee el nombre y versión del agente de usuario cliente.

El UAC puede decidir añadir un cuerpo de mensaje a la solicitud “INVITE”. Si este cuerpo es usado para describir sesiones, se debe hacer uso de la especificación RFC 2327 (Protocolo de Descripción de Sesiones (SDP)). SIP usa un modelo oferta/respuesta descrito en RFC 3264, donde un UA envía una descripción de sesión, llamada oferta, que contiene una descripción propuesta de la sesión. La oferta indica los medios de comunicación deseados (audio, video, juegos), parámetros de esos medios (por ejemplo tipos de CODEC) y direcciones para recibir medios. El otro UA responde con otra descripción de la sesión, llamada respuesta, la cual indica cuales medios de comunicación son aceptados, los parámetros que aplican para esos medios y direcciones para recibir medios. En la especificación RFC 3261, las ofertas y las respuestas sólo pueden aparecer en solicitudes “INVITE”, en sus respuestas y en las solicitudes “ACK”. Hay dos posibilidades para este modelo oferta/respuesta: la oferta esta en la solicitud “INVITE” y la respuesta en el mensaje 2xx, o la oferta está en la respuesta 2xx y la respuesta en la solicitud “ACK”. Si la solicitud “INVITE” no contiene una descripción de sesión, entonces el UAC ha solicitado que el UAS provea la oferta de la sesión en la respuesta 2xx.

4.8 Finalización de sesiones

La solicitud “BYE” es usada para terminar una sesión específica o un intento de sesión. Cuando una solicitud “BYE” es recibida sobre un diálogo, cualquier sesión asociada con ese diálogo debería terminar.

Una solicitud “BYE” es construida como se haría con cualquier otra solicitud dentro de diálogo. El UAS que recibe la solicitud “BYE” verifica si concuerda con algún diálogo existente. Si no concuerda con ningún diálogo existente, el UAS

debería generar una respuesta 481 (Llamada/Transacción no existe). Si existe el diálogo, el UAS debería terminar la sesión siempre que no se trate de sesiones multicast. Si se trata de una sesión multicast, el UAS puede decidir si terminar o no su participación. Independientemente de si termina su participación o no, el UAS debe generar una respuesta 2xx para la solicitud BYE.

4.9 Comportamiento de los servidores proxy

Los servidores proxy son elementos que encaminan las solicitudes SIP hacia los agentes de usuarios servidores y las respuestas SIP hacia los agentes de usuario clientes. Una solicitud puede atravesar varios servidores proxy en su camino hacia un UAS. Cada uno de ellos va a tomar una decisión de enrutamiento, modificando la solicitud antes de enviarla al siguiente elemento. Las respuestas a esa solicitud serán encaminadas a través del mismo conjunto de servidores proxy que atravesó la solicitud en orden inverso.

Cuando un servidor proxy recibe una solicitud, debe inspeccionar el URI-de-la-Solicitud en la línea de inicio. Si este URI indica un dominio que no es responsabilidad de este servidor proxy, éste no debe cambiar el URI-de-la-Solicitud y debe proceder a pasar la solicitud al siguiente elemento. Si el dominio sí es responsabilidad del servidor proxy, éste remplaza el URI-de-la-Solicitud con el resultado obtenido de algún servicio de localización.

Si este servidor proxy desea permanecer en el camino de futuras solicitudes (asumiendo que la solicitud actual creará un diálogo), debe insertar un campo “Record-Route” dentro de la solicitud antes de cualquier campo “Record-Route” existente. Cuando el UAS devuelva una respuesta a esta solicitud, ésta debe contener una réplica de estos campos “Record-Route”. Los campos “Record-Route” van a servir en el UAC y el UAS para construir el estado del diálogo. Dentro de estos estados se almacenará esta lista de campos “Record-Route” para así establecer una ruta dentro del diálogo.

Cuando se envíen futuras solicitudes dentro del diálogo, estás podrán contener una lista de campos “Route” que contienen los mismos valores de los campos “Record-Route” almacenados anteriormente. Si están presentes, estos campos “Route” son utilizados por los servidores proxy para encaminar una solicitud. El servidor proxy inspeccionará el campo “Route” que se encuentra en la parte superior. Si este campo “Route” indica este servidor proxy, éste remueve este campo. Luego el servidor proxy reenvía la solicitud al punto indicado en el nuevo campo “Route” en la parte superior o al URI-de-la-solicitud si no hay campos “Route” presentes.

Por otra parte, el servidor proxy debe insertar un campo Via dentro de la solicitud antes de cualquier campo Via existente. Este campo Via indica la ubicación del servidor proxy y debe tener un parámetro branch. Las respuestas se encaminan por el mismo trayecto recorrido por las solicitudes gracias a los campos Via. Cuando

un servidor proxy recibe una respuesta, remueve el campo Via que se encuentra en la parte superior y reenvía la respuesta al punto indicado por el siguiente campo Via.

5. Aplicación Analizador de Protocolos

Es una aplicación que sirve para capturar, decodificar y visualizar las tramas que contienen los mensajes de señalización H.323 y los mensajes SIP. La siguiente figura muestra los diversos controles que posee esta aplicación.

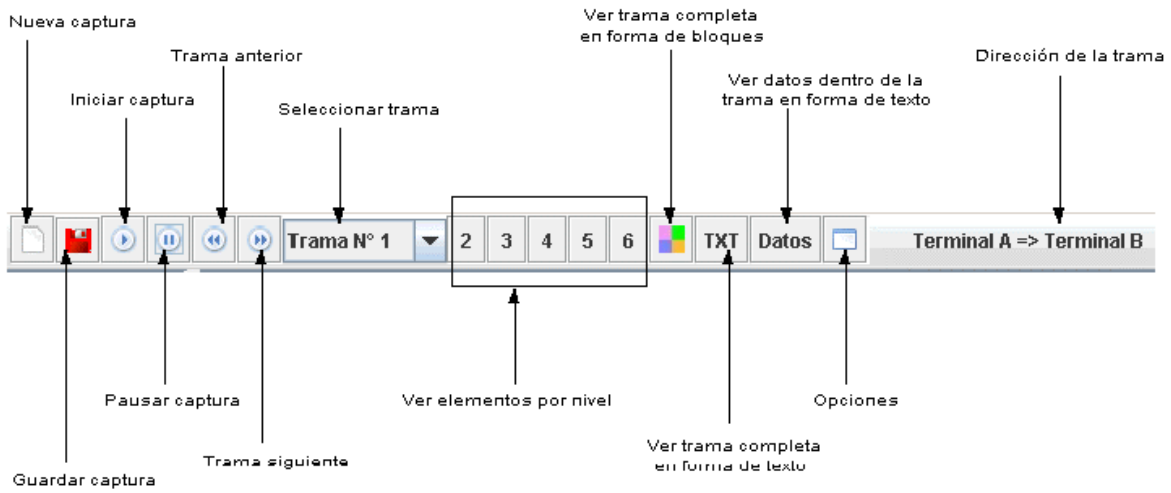


Figura 4. Barra de herramientas.

A continuación se explica cada una de estas funciones:

Este botón borra todas las tramas capturadas para comenzar una nueva captura.

Guarda la captura en un archivo de texto.

Comienza captura de tramas.

Pausa la captura de tramas.

Trama N° 1 Permiten escoger la trama a visualizar. Estos controles están habilitados sólo cuando se visualiza una trama completa.

2 3 4 5 6 Estos botones permiten visualizar en forma de bloques los componentes de todas las tramas según el nivel en que fueron decodificados:

- Nivel 2: Componentes de las tramas 802.3.
- Nivel 3: Cabeceras IPv4.

- Nivel 4: Cabeceras UDP y TCP.
- Nivel 5: Cabeceras TPKT.
- Nivel 6: Mensajes Q.931.

La siguiente figura muestra la visualización obtenida en el nivel 4.

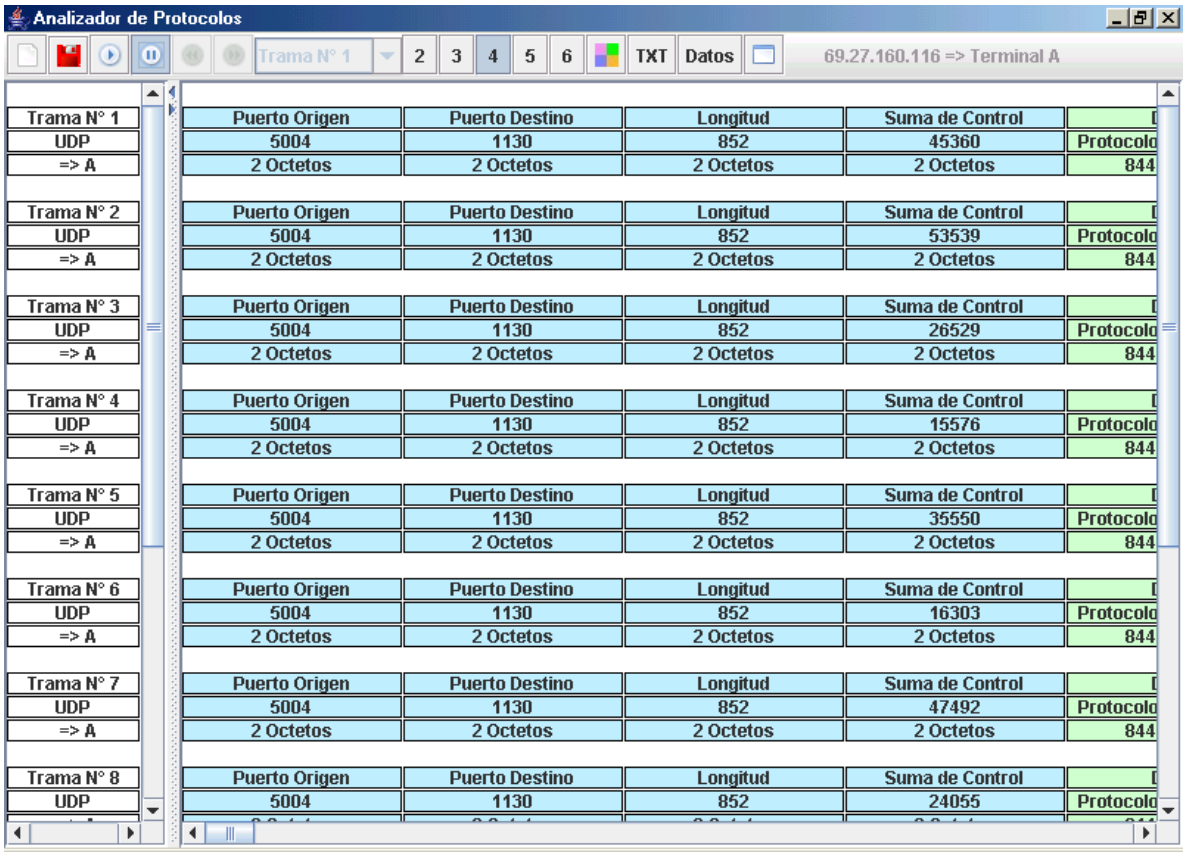


Figura 5. Visualización obtenida en el nivel 4.

Al hacer doble clic en alguna de las tramas, la pantalla mostrará una visión dedicada a sólo esa trama.

Permite visualizar en forma de bloques una trama completa, es decir, con los componentes decodificados en todos los niveles, tal como muestra la siguiente figura:

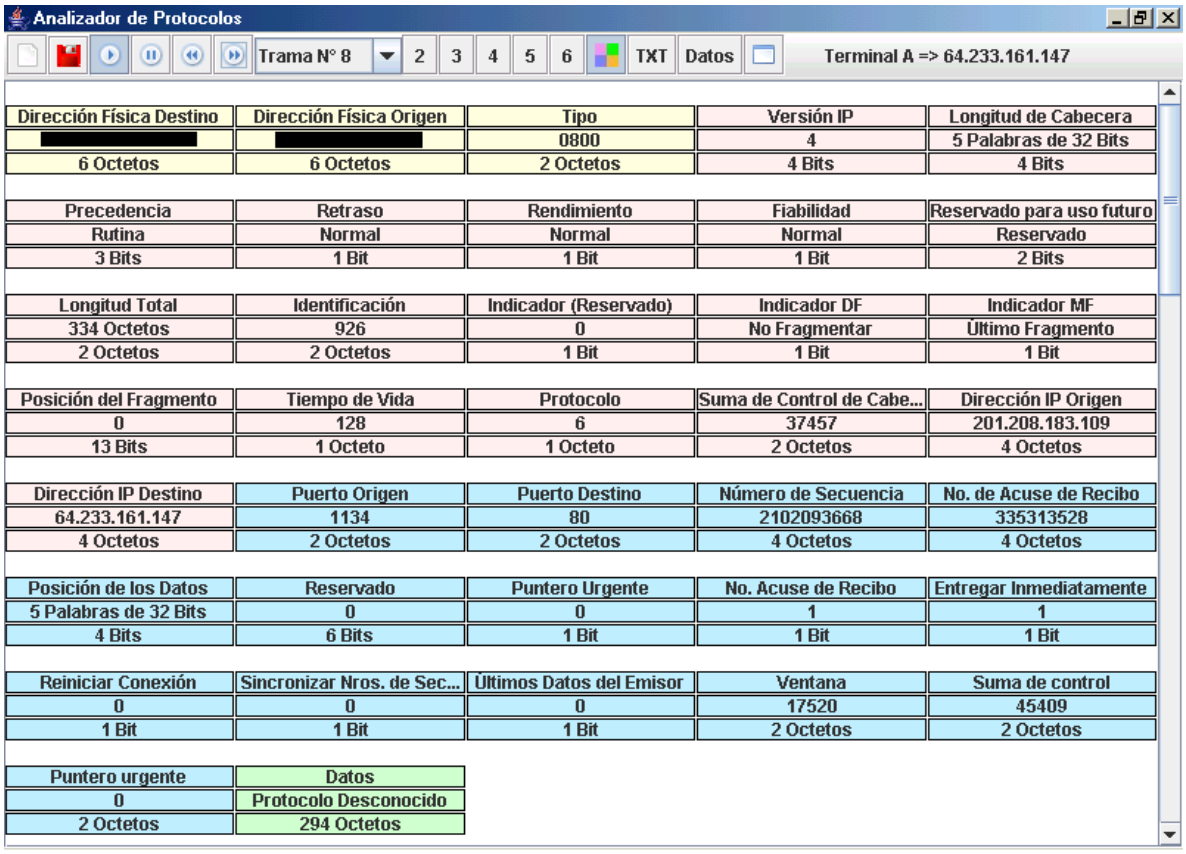


Figura 6. Visualización de una trama completa en forma de bloques.

TXT Permite visualizar en forma de texto una trama completa, es decir, con los componentes decodificados en todos los niveles. Es una visualización de los mismos componentes obtenidos con el botón , pero en forma de texto:

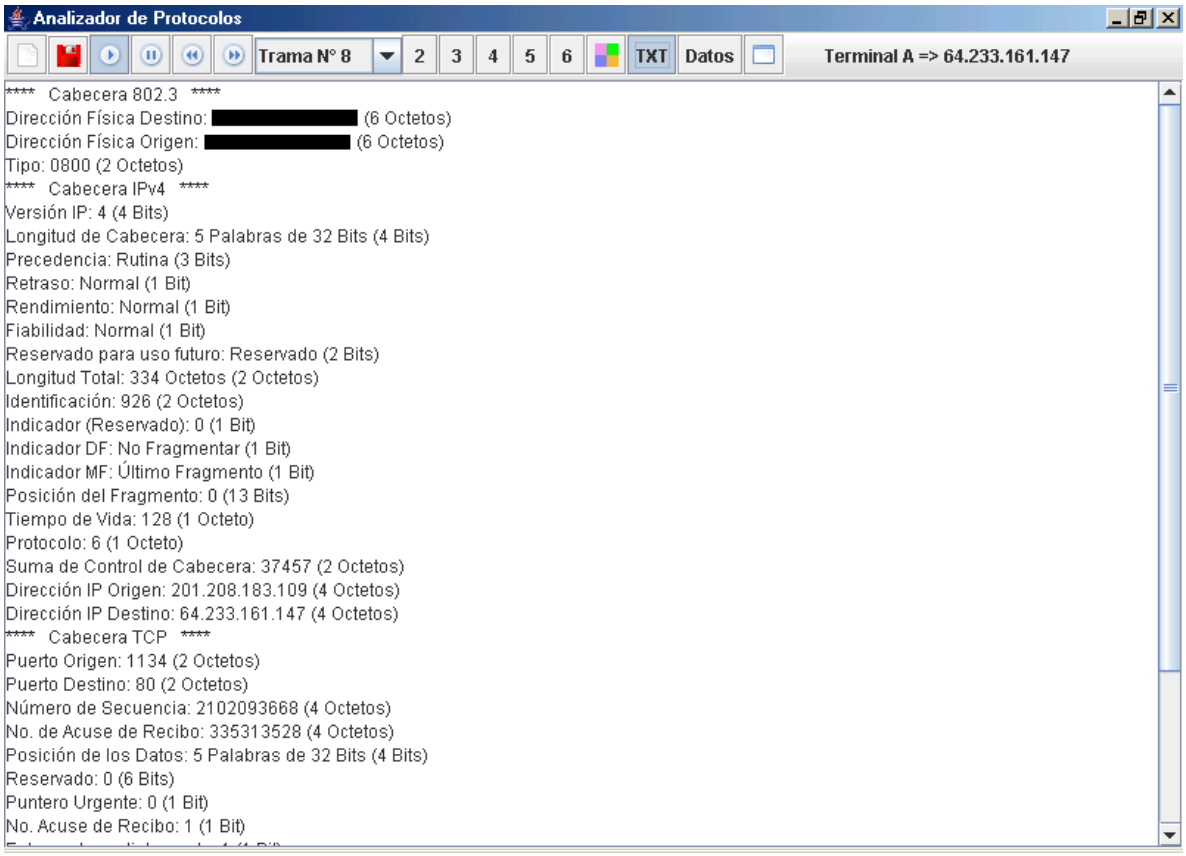


Figura 7. Visualización de una trama completa en forma de texto.

Si la trama no es completamente codificada por los niveles del decodificador, el resto de información es mostrado al presionar este botón en el formato de texto UTF-8. Esto quiere decir que si la trama no contiene un mensaje Q.931, el analizador decodificará todas las cabeceras y mostrará los datos en esta vista. Esta herramienta es útil para visualizar los mensajes en una llamada SIP. En esta modalidad, se dispone de un panel doble para poder comparar un mensaje con otro:

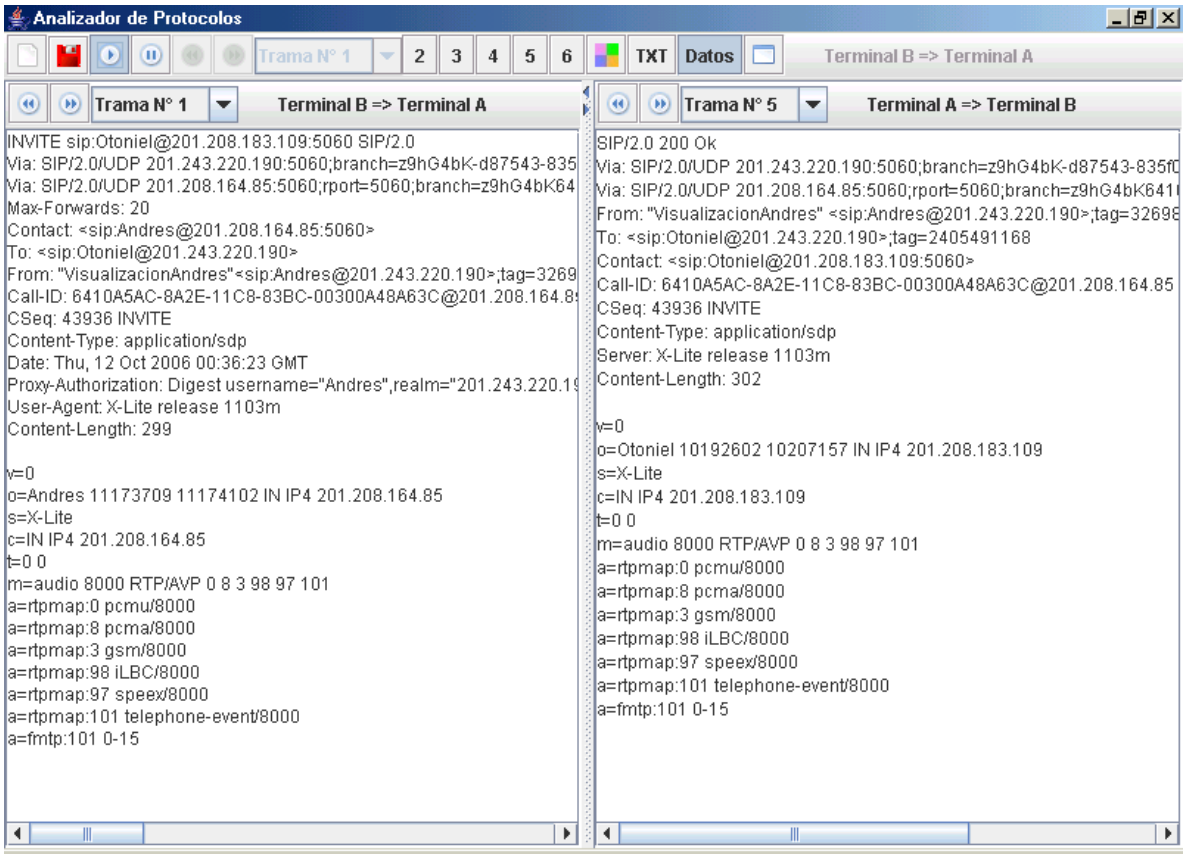


Figura 8. Visualización de los datos en el formato de texto UTF-8.

Abre un el diálogo de opciones que se muestra en la figura 9.

Este diálogo permite especificar el filtro de tramas. Los campos “Direcciones IP” servirán para identificar al destino o al origen de las tramas como terminal A, B o C. Si se habilita la casilla “Filtrar Direcciones IP”, se filtrarán las tramas que posean al menos una de las direcciones IP que se hayan introducido en el panel “Direcciones IP”. De la misma forma, si se habilita la casilla “Filtrar Puertos”, se filtrarán las tramas que posean al menos uno de los puertos que se hayan introducido el panel “Puertos”. Si se activa la casilla “Filtrar Protocolos” y la casilla “Señalización H.323”, se dejarán pasar las tramas que posean mensajes Q.931. En caso de seleccionar más de una opción de filtrado, el filtro se comportará como un AND lógico, es decir, se deben cumplir todas las condiciones seleccionadas para mostrar la trama. La casilla modo monitor y la interfaz de red no deben ser modificadas.

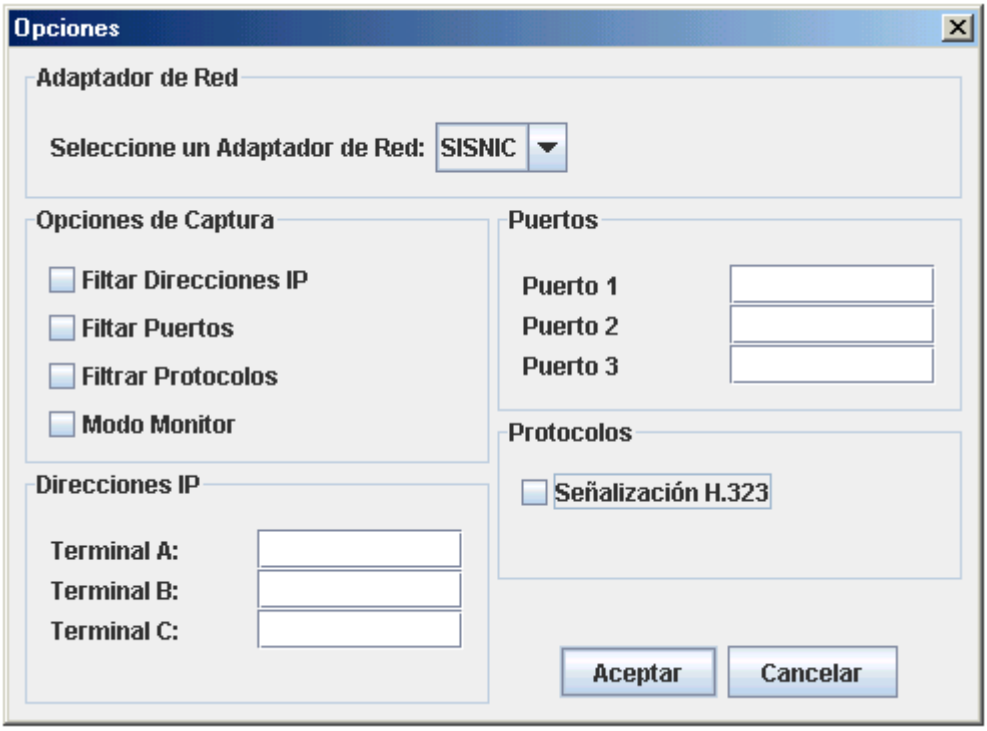


Figura 9. Diálogo de opciones.

6. Trabajo práctico de laboratorio

6.1. Equipos y aplicaciones necesarios

- Tres computadores personales interconectados en una red local
- Aplicación Softphone para H.323
- Aplicación Gatekeeper (Guardián de Puerta) para H.323
- Aplicación Softphone para SIP
- Aplicación Servidor Proxy para SIP
- Aplicación Analizador de Protocolos

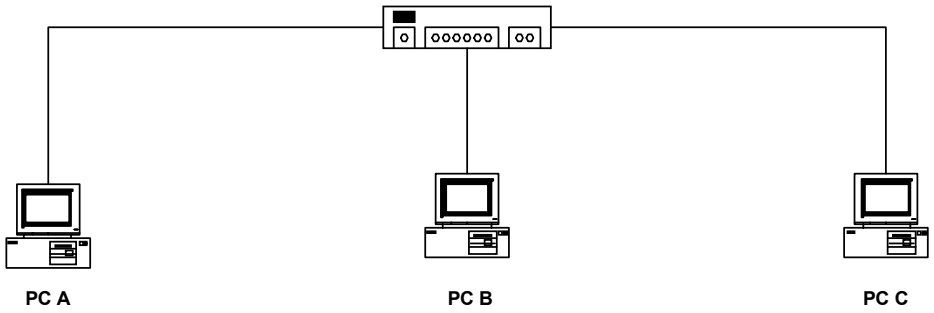


Figura 10. Equipos.

6.2. Procedimientos

6.2.1. Llamada H.323

1. Comience la aplicación Gatekeeper en el PC B.
2. Abra los Spftphone H.323 en los PCs A y C.
3. Inicie el Analizador de Protocolos en el PC B
4. En las opciones del programa Analizador de Protocolos (), active sólo las casillas "Filtrar Protocolos" y "Señalización H.323" y desactive todas las demás casillas.
5. Comience una captura.
6. En el PC A realice una llamada al PC C. Para ello, introduzca TerminalC en la aplicación Softphone.
7. Finalice la llamada.
8. Detenga la captura.
9. Utilice los botones 6, y TXT para analizar los mensajes Q.931 utilizados en la señalización de la llamada.
10. Si lo desea, guarde la captura en un archivo de texto con el botón .
11. Cierre las aplicaciones Gatekeeper y Softphone H.323.

6.2.2. Llamada SIP

1. Comience la aplicación Proxy en el PC B.
2. Abra los Spftphone SIP en los PCs A y C.
3. En las opciones del programa Analizador de Protocolos (), desactive las casillas "Filtrar Protocolos" y "Señalización H.323", active sólo la casilla "Filtrar Puertos" y coloque el valor 5060 en el campo "Puerto 1".
4. Borre la captura anterior y comience una nueva en el PC B.
5. En el PC A realice una llamada al PC C. Para ello, introduzca UsuarioC en la aplicación Softphone.
6. Para poder observar la finalización de la llamada, inicie el analizador de protocolos en el PC A.
7. En las opciones del programa Analizador de Protocolos () del PC A, active la casilla "Filtrar Puertos" y coloque el valor 5060 en el campo "Puerto 1".
8. Comience una captura en el PC A.
9. Finalice la llamada.
10. Detenga las capturas.
11. Utilice el botón Datos para analizar y comparar los mensajes SIP.
12. Si lo desea guarde las capturas en archivos de texto con el botón .
13. Cierre las aplicaciones Gatekeeper, Softphone SIP y Analizador de Protocolos.

7. Conclusiones

8. Bibliografía

Estándar RFC 3261 (2002), SIP: Protocolo de iniciación de sesiones.

Recomendación UIT-T H.225.0 (2000), Protocolos de señalización de llamada y paquetización de trenes de medios para sistemas de comunicación multimedios por paquetes.

Recomendación UIT-T H.245 (2000), Protocolo de control para comunicación multimedios.

Recomendación UIT-T H.323 (2000), Sistemas de comunicación multimedios basados en paquetes.

Recomendación UIT-T Q.931 (1998), Especificación de la capa 3 de la interfaz usuario-red de la red digital de servicios integrados para el control de la llamada básica.

[ANEXO 7]

[Manual de uso de la aplicación Analizador de Protocolos]

1. Funcionamiento.

En la figura 1 se muestra la estructura de esta aplicación. Cuando la interfaz gráfica de usuario lo ordena, el bloque captador comienza la captura de tramas. Luego cada trama es enviada al bloque decodificador. Este bloque está dividido en seis niveles que corresponden a los diversos protocolos que describen la estructura de la trama.

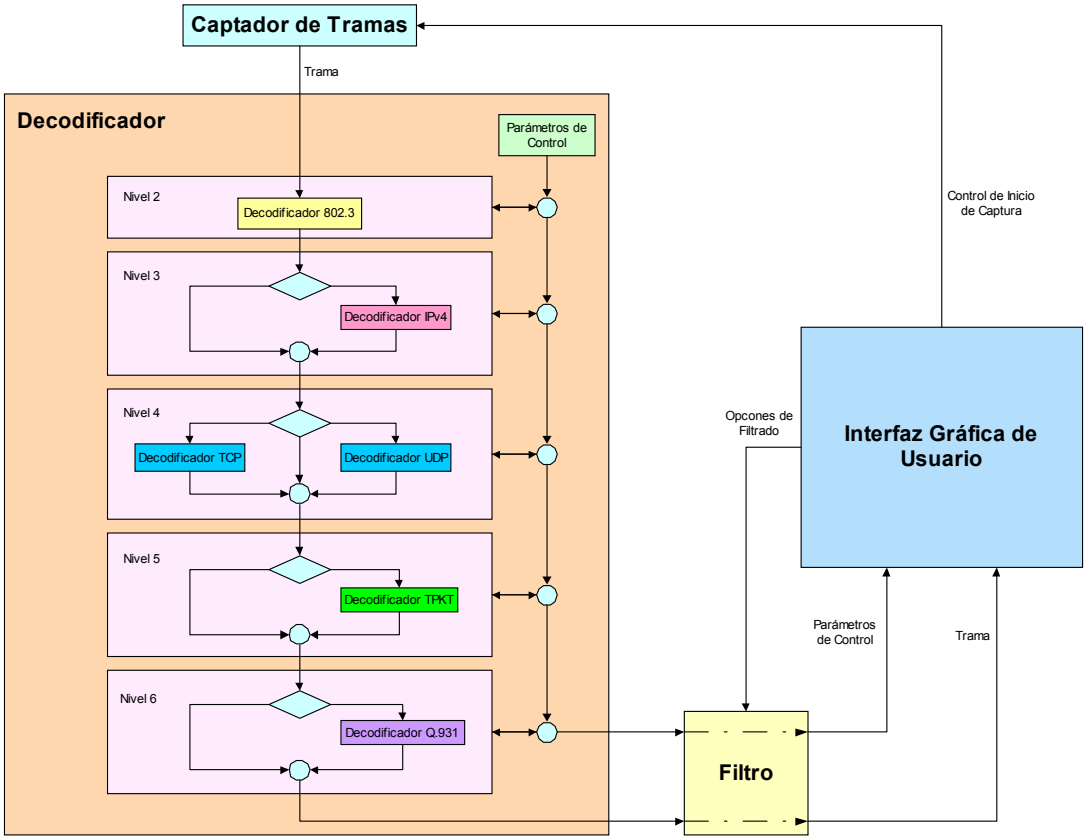


Figura 4. Estructura de la aplicación.

Una vez que la trama ha sido codificada, debe pasar por el filtro, en donde se comparan los parámetros de control con las opciones de filtrado que indica la interfaz de usuario, para determinar si la trama debe ser mostrada o no al usuario. Este bloque puede filtrar las tramas en función de las direcciones IP, los puertos de transporte o de los protocolos, según lo especifique el usuario. Finalmente, si cumple con las especificaciones de filtrado, la trama y los parámetros de control son enviados a la

interfaz gráfica de usuario. En esta interfaz, se almacena toda esta información para permitir al usuario analizar cualquier trama captada.

2. Iniciación de la aplicación

Para iniciar la aplicación se debe realizar una de las siguientes acciones:

- Ejecutar el comando:
  C:\"Archivos de programa"\Java\jdk1.5.0_06\bin\java Analizador
- Ejecutar el archivo:
  C:\"Archivos de programa"\Java\jdk1.5.0_06\AnalizadordeProtocolos.jar o alguno de sus accesos directos.

3. Interfaz de usuario

La siguiente figura muestra los diversos controles que posee la interfaz de usuario.

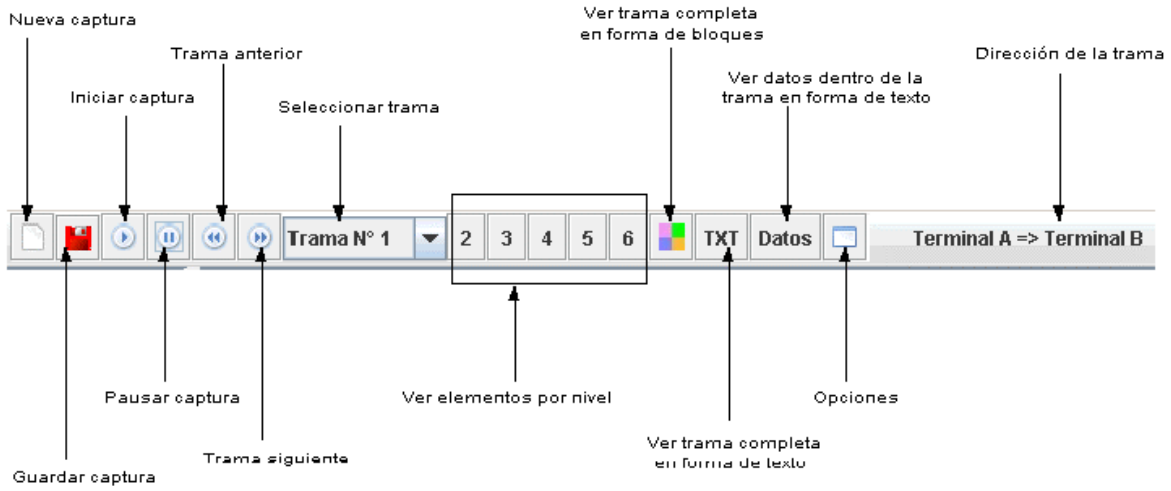


Figura 5. Barra de herramientas.

A continuación se explica cada una de estas funciones:

Este botón borra todas las tramas capturadas para comenzar una nueva captura.

Guarda la captura en un archivo de texto. Abre el siguiente diálogo para indicar el archivo:

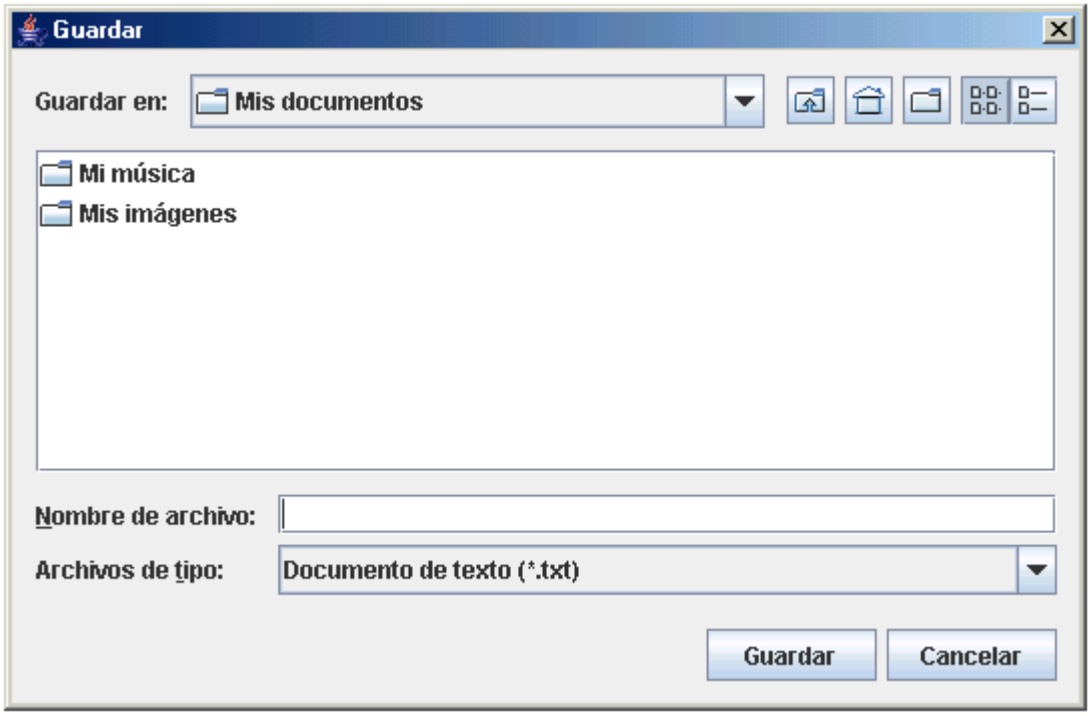


Figura 6. Diálogo para guardar captura.

Comienza captura de tramas.

Pausa la captura de tramas.

Permiten escoger la trama a visualizar. Estos controles están habilitados sólo cuando se visualiza una trama completa.

Estos botones permiten visualizar en forma de bloques los componentes de todas las tramas según el nivel en que fueron decodificados:

- Nivel 2: Componentes de las tramas 802.3.
- Nivel 3: Cabeceras IPv4.
- Nivel 4: Cabeceras UDP y TCP.
- Nivel 5: Cabeceras TPKT.
- Nivel 6: Mensajes Q.931.

La siguiente figura muestra la visualización obtenida en el nivel 4.

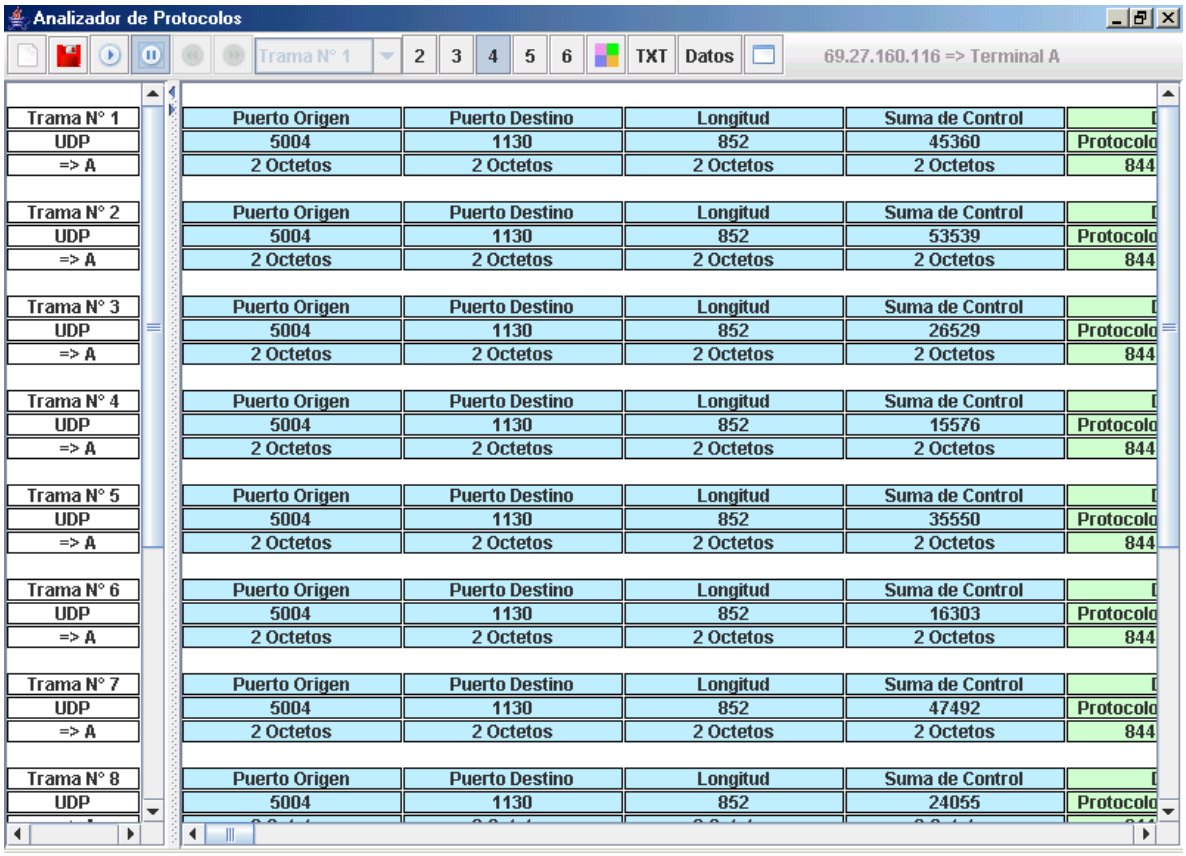


Figura 7. Visualización obtenida en el nivel 4.

Al hacer doble clic en alguna de las tramas, la pantalla mostrará una visión dedicada a sólo esa trama.

Permite visualizar en forma de bloques una trama completa, es decir, con los componentes decodificados en todos los niveles, tal como muestra la siguiente figura:

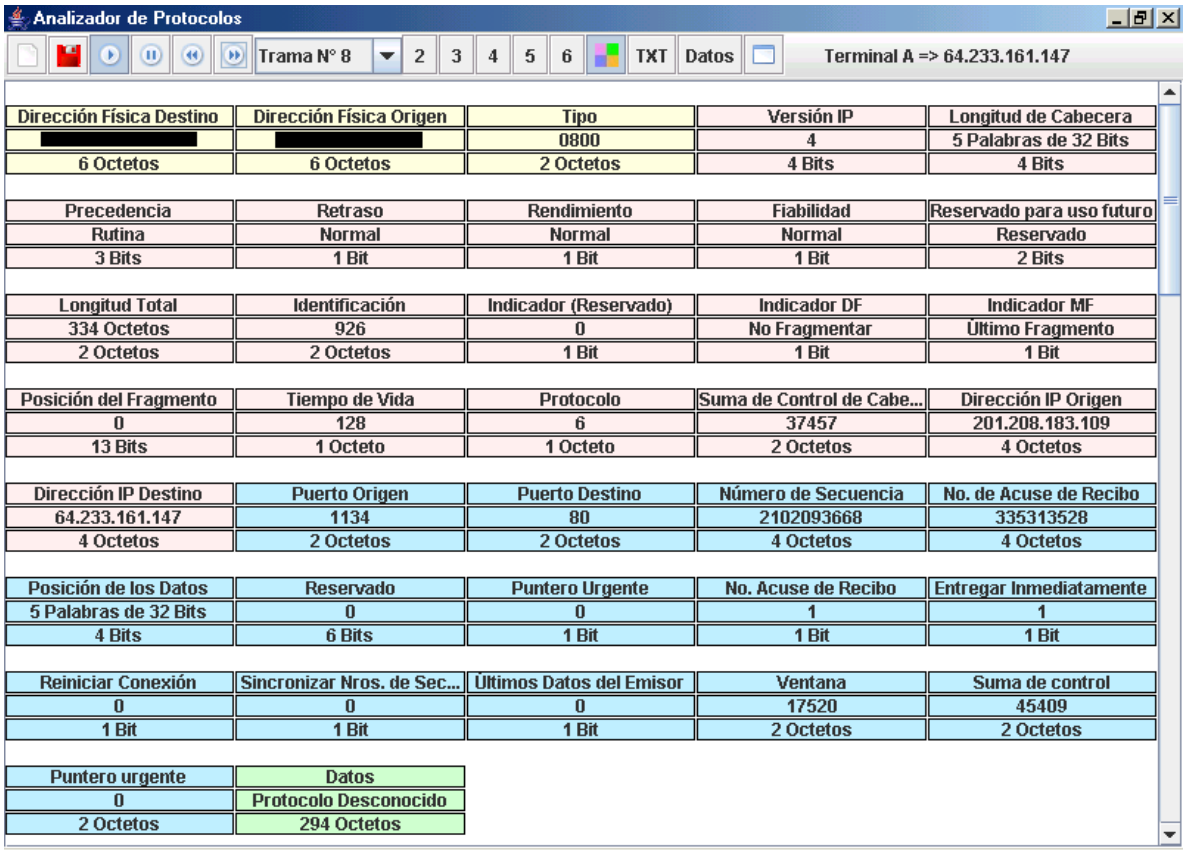


Figura 8. Visualización de una trama completa en forma de bloques.

TXT Permite visualizar en forma de texto una trama completa, es decir, con los componentes decodificados en todos los niveles. Es una visualización de los mismos componentes obtenidos con el botón , pero en forma de texto:

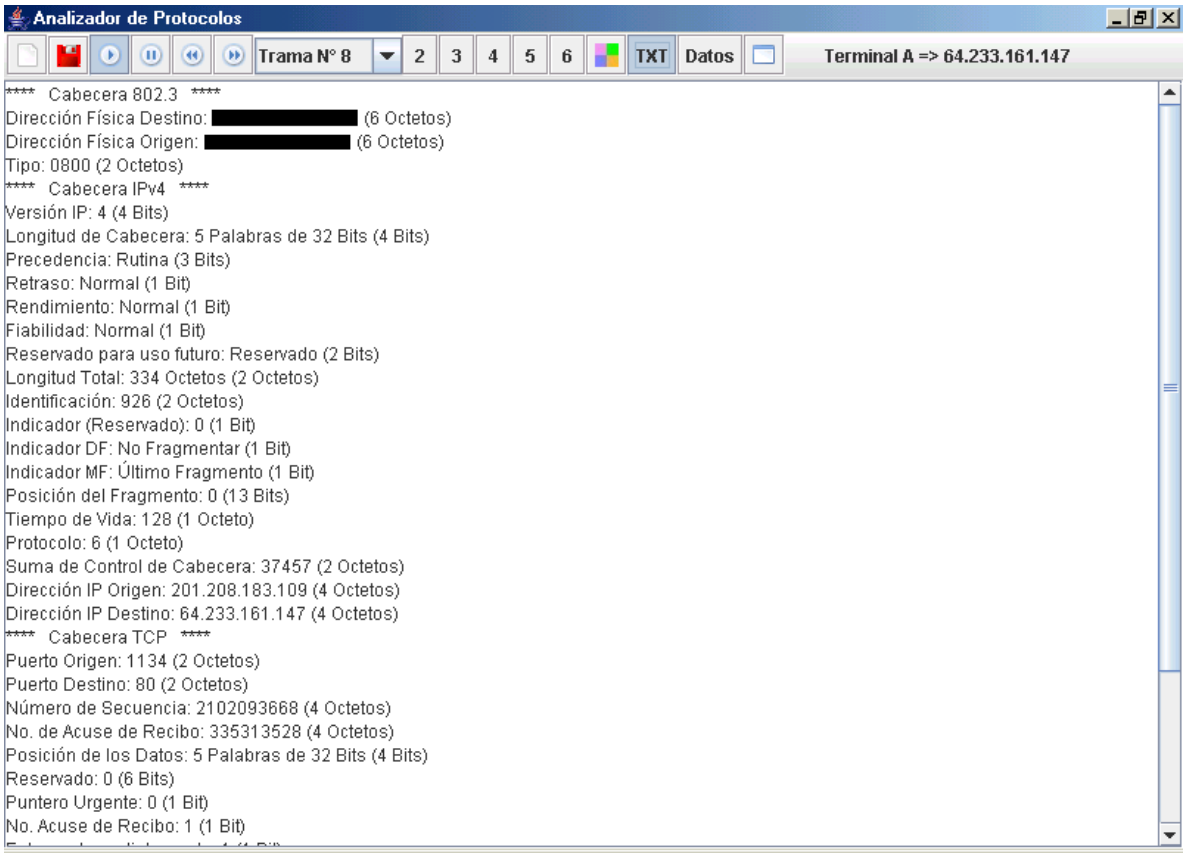


Figura 9. Visualización de una trama completa en forma de texto.

Si la trama no es completamente codificada por los niveles del decodificador, el resto de información es mostrado al presionar este botón en el formato de texto UTF-8. Esto quiere decir que si la trama no contiene un mensaje Q.931, el analizador decodificará todas las cabeceras y mostrará los datos en esta vista. Esta herramienta es útil para visualizar los mensajes en una llamada SIP. En esta modalidad, se dispone de un panel doble para poder comparar un mensaje con otro:

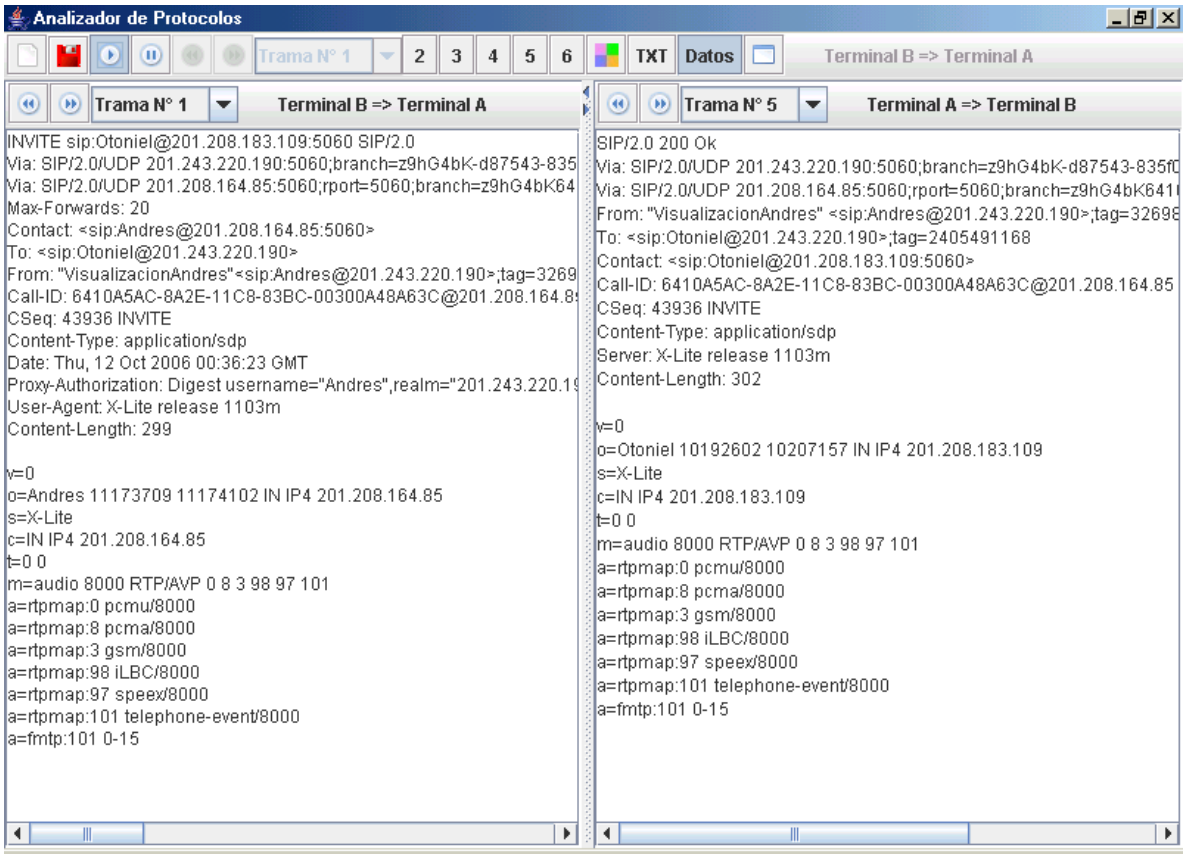


Figura 10. Visualización de los datos en el formato de texto UTF-8.

Abre un el diálogo de opciones que se muestra en la figura 8:

La primera vez que se ejecuta el programa Analizador de Protocolos en un ordenador, aparece este diálogo de opciones para que el usuario especifique por primera vez la configuración del programa, la cual será guardada en el ordenador. Este diálogo permite seleccionar la interfaz de red en la que se hará la captura. También permite especificar el filtro de tramas. Los campos “Direcciones IP” servirán para identificar al destino o al origen de las tramas como terminal A, B o C.

Si se habilita la casilla “Filtrar Direcciones IP”, se filtrarán las tramas que posean al menos una de las direcciones IP que se hayan introducido en el panel “Direcciones IP”. De la misma forma, si se habilita la casilla “Filtrar Puertos”, se filtrarán las tramas que posean al menos uno de los puertos que se hayan introducido el panel “Puertos”. Si se activa la casilla “Filtrar Protocolos” y la casilla “Señalización H.323”, se dejarán pasar las tramas que posean mensajes Q.931. En caso de seleccionar más de una opción de filtrado, el filtro se comportará como un AND lógico, es decir, se deben cumplir todas las condiciones seleccionadas para mostrar la trama. Finalmente, la casilla modo monitor permite habilitar la interfaz de red en modo monitor. Si se modifica está casilla o si se cambia la interfaz de red, el

programa debe ser reiniciado para que estos cambios tengan efecto. Si se modifican las opciones de filtrado, el programa no necesita ser reiniciado.

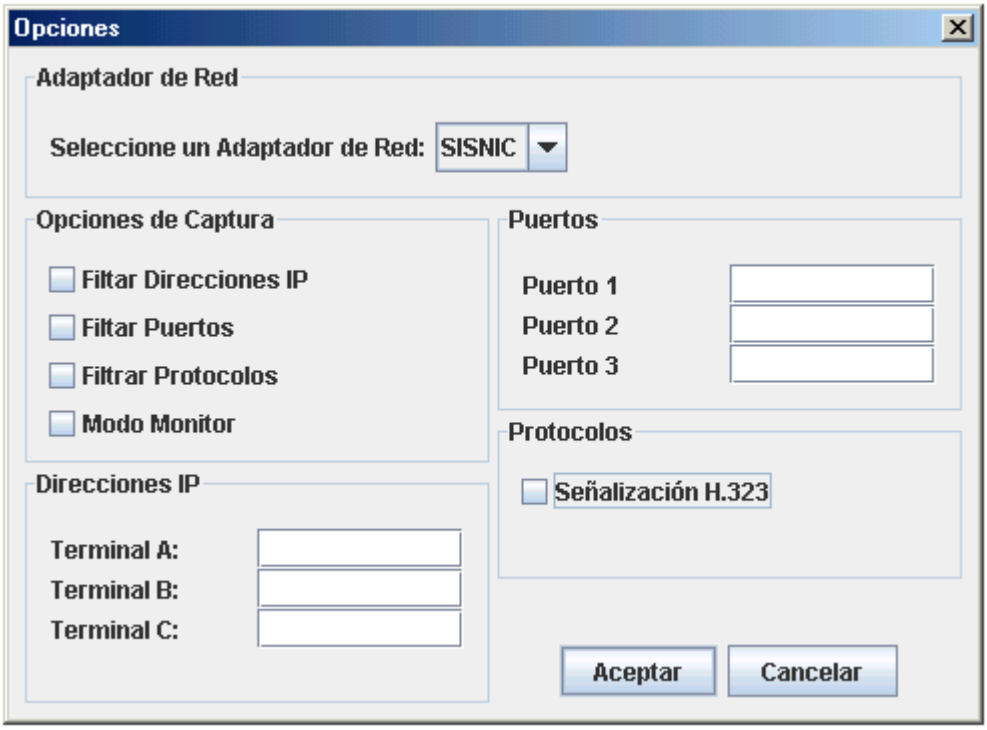


Figura 11. Diálogo de opciones.

[ANEXO 8]

[Manual de instalación de la aplicación Analizador de Protocolos]

1. Inserte el disco compacto que contiene la aplicación Analizador de Protocolos.
2. Abra la carpeta Windows dentro del disco compacto.
3. Instale el programa jdk-1_5_0_06-windows-i586-p.exe
4. Instale el programa WinPcap_3_1.exe
5. Copie todo el contenido dentro de la carpeta Analizador_de_Protocolos en el directorio C:\"Archivos de programa"\Java\jdk1.5.0_06\bin
6. Para ejecutar el programa haga alguna de las siguientes opciones:
    - Ejecute el archivo C:\"Archivos de programa"\Java\jdk1.5.0_06\bin\Analizador_de_Protocolos.jar
    - Cree un acceso directo al archivo C:\"Archivos de programa"\Java\jdk1.5.0_06\bin\Analizador_de_Protocolos.jar, coloque el acceso directo en el lugar deseado y ejecútelo.
    - Ejecute el comando: C:\"Archivos de programa"\Java\jdk1.5.0_06\bin\java

[ANEXO 9]

[Código de la aplicación Analizador de Protocolos]

1. Clase Analizador:

```
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.util.Date;

import net.sourceforge.jpcap.capture.*;
import net.sourceforge.jpcap.net.*;

public class Analizador{

	static int contador=0;
	static int contador2=0;
	static int contador3=0;
	static int aux;
	static boolean f;
	static String sdireccion;
	static int tramasrecibidas=0;
	static int[] cursores=new int[4];
	static String[] protocolos=new String[7];
	static String[] control=new String[5];
	static byte[] b;
	static boolean detener=false;

	static int dispositivoseleccionado;
	static String[] dispositivos;

	static File directorio=new File("c:\\"+File.separator+"Analizador"+File.separator+"Temporales");
	static File archivo;
	static String[] listadearchivos;
	static FileInputStream entrada;
	static ObjectInputStream entradaserializada;
	static FileOutputStream salida;
	static FileWriter salida3;
	static Date fecha;
	static ObjectOutputStream salidaserializada;
	static File archivo2;
	static FileInputStream entrada2;
	static FileOutputStream salida2;

	static boolean primeravez;
	static String dispositivousado;
	static boolean filtrarip;
	static boolean filtrarpuertos;
	static boolean filtrarprotocolo;
	static boolean modomonitor;
	static String ipa;
	static String ipb;
	static String ipc;
	static String puerto1;
	static String puerto2;
	static String puerto3;
	static boolean q931;

	static JFrame marco = new JFrame("Analizador de Protocolos");
	static JToolBar barradeherramientas = new JToolBar();
	static JToolBar barradeherramientas2 = new JToolBar();
```

```
static JToolBar barradeherramientas3 = new JToolBar();
static JDialog dialogoopciones=new JDialog(marco,"Opciones",true);
static JFileChooser electordearchivo=new JFileChooser();
static JPanel panel1=new JPanel();
static JPanel panel2=new JPanel();
static JPanel panel3=new JPanel();
static JPanel panel4=new JPanel();
static JPanel panel5=new JPanel();
static JPanel panel6=new JPanel();
static JPanel panel7=new JPanel();
static JPanel panel9=new JPanel();
static JPanel panel10=new JPanel();
static JPanel panel12=new JPanel();
static JPanel panel13=new JPanel();
static JPanel panel15=new JPanel();
static JPanel panel16=new JPanel();
static JLabel etiqueta1=new JLabel("Seleccione un Adaptador de Red:");
static JLabel etiqueta2=new JLabel("Terminal A:");
static JLabel etiqueta3=new JLabel("Terminal B:");
static JLabel etiqueta4=new JLabel("Terminal C:");
static JLabel etiquetapuerto1=new JLabel("Puerto 1");
static JLabel etiquetapuerto2=new JLabel("Puerto 2");
static JLabel etiquetapuerto3=new JLabel("Puerto 3");
static JCheckBox casillafiltrarprotocolo=new JCheckBox("Filtrar Protocolos");

static JCheckBox casillafiltrarip=new JCheckBox("Filtar Direcciones IP");
static JCheckBox casillafiltrarpuertos=new JCheckBox("Filtar Puertos");
static JCheckBox casillamodomonitor=new JCheckBox("Modo Monitor");

static JCheckBox casillaq931=new JCheckBox("Señalización H.323");

//static JCheckBox casillasip=new JCheckBox("SIP");

static JTextField cuadrodetexto2=new JTextField(9);
static JTextField cuadrodetexto3=new JTextField(9);
static JTextField cuadrodetexto4=new JTextField(9);
static JTextField cuadrodetextopuerto1=new JTextField(9);
static JTextField cuadrodetextopuerto2=new JTextField(9);
static JTextField cuadrodetextopuerto3=new JTextField(9);
static JComboBox listadispositivos;
static GridBagLayout gridbag=new GridBagLayout();
static GridBagConstraints constraints=new GridBagConstraints();
static JButton botonaceptar=new JButton("Aceptar");
static JButton botoncancelar=new JButton("Cancelar");

/*static ImageIcon icononuevo = new ImageIcon("nuevo.png");
static ImageIcon iconoabrir = new ImageIcon("abrir.gif");
static ImageIcon iconoguardar = new ImageIcon("guardar.gif");
static ImageIcon iconoescuchar = new ImageIcon("escuchar.png");
static ImageIcon iconotramasiguiente = new ImageIcon("tramasiguiente.png");
static ImageIcon iconotramaanterior = new ImageIcon("tramaanterior.png");
static ImageIcon iconodetener = new ImageIcon("detener.png");
static ImageIcon iconoopciones = new ImageIcon("opciones.png");*/

static JButton botonnuevo= new JButton(Utilidades.icononuevo);
//static JToggleButton botonabrir= new JToggleButton(Utilidades.iconoabrir);
static JButton botonguardar= new JButton(Utilidades.iconoguardar);
static JToggleButton botonescuchar= new JToggleButton(Utilidades.iconoescuchar);
static JToggleButton botondetener= new JToggleButton(Utilidades.iconodetener);
static JButton botontramaanterior= new JButton(Utilidades.iconotramaanterior);
static JButton botontramaanterior2= new JButton(Utilidades.iconotramaanterior);
static JButton botontramaanterior3= new JButton(Utilidades.iconotramaanterior);
//static JLabel tramanumero=new JLabel(" Trama N° 0 ");
static JComboBox caja=new JComboBox();
static JComboBox caja2=new JComboBox();
static JComboBox caja3=new JComboBox();
static JLabel direccion=new JLabel("");
```

```
static JLabel direccion2=new JLabel("");
static JLabel direccion3=new JLabel("");
static int numerotrama;
static int numerotrama3;
static JButton botontramasiguiente= new JButton(Utilidades.iconotramasiguiente);
static JButton botontramasiguiente2= new JButton(Utilidades.iconotramasiguiente);
static JButton botontramasiguiente3= new JButton(Utilidades.iconotramasiguiente);
static JToggleButton botoncapa2= new JToggleButton(" 2 ");
static JToggleButton botoncapa3= new JToggleButton(" 3 ");
static JToggleButton botoncapa4= new JToggleButton(" 4 ");
static JToggleButton botoncapa5= new JToggleButton(" 5 ");
static JToggleButton botoncapa6= new JToggleButton(" 6 ");
static JToggleButton botontramaportrama= new JToggleButton(Utilidades.iconotramaportrama);
static JToggleButton botontexto= new JToggleButton("TXT");
static JToggleButton botondatos= new JToggleButton("Datos");
static JButton botonopciones= new JButton(Utilidades.iconoopciones);

//static JSeparator separador1=new JSeparator(JSeparator.VERTICAL);
//static JSeparator separador2=new JSeparator(JSeparator.VERTICAL);
//static JSeparator separador3=new JSeparator(JSeparator.VERTICAL);

static final int NUMERODECOLUMNAS=80;
static final int NUMERODECOLUMNAS2=150;
static PlantillaModeloTabla modelotablatodaslascapas=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablacapa2=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablacapa3=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablacapa4=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablacapa5=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablacapa6=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablatrama=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablaindice2=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablaindice3=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablaindice4=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablaindice5=new PlantillaModeloTabla();
static PlantillaModeloTabla modelotablaindice6=new PlantillaModeloTabla();
static JTable tablacapa2=new JTable(modelotablacapa2);
static JTable tablacapa3=new JTable(modelotablacapa3);
static JTable tablacapa4=new JTable(modelotablacapa4);
static JTable tablacapa5=new JTable(modelotablacapa5);
static JTable tablacapa6=new JTable(modelotablacapa6);
static JTable tablatrama=new JTable(modelotablatrama);
static JTable tablaindice2=new JTable(modelotablaindice2);
static JTable tablaindice3=new JTable(modelotablaindice3);
static JTable tablaindice4=new JTable(modelotablaindice4);
static JTable tablaindice5=new JTable(modelotablaindice5);
static JTable tablaindice6=new JTable(modelotablaindice6);

static JPanel tarjetas = new JPanel();
static CardLayout cl=new CardLayout();;
static JPanel panel8=new JPanel();
static CardLayout cl2=new CardLayout();;
static CardLayout cl3=new CardLayout();
static JScrollPane paneldesplazable=new JScrollPane(tarjetas,
ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
static JScrollPane paneldesplazable2=new JScrollPane(tablatrama,
ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
static JTextArea areadetexto=new JTextArea();
static JScrollPane paneldesplazable3=new JScrollPane(areadetexto,
ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);

static JTextArea areadetexto2=new JTextArea();
static JScrollPane paneldesplazable5=new JScrollPane(areadetexto2,
ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
```

```
static JTextArea areadetexto3=new JTextArea();
static JScrollPane paneldesplazable6=new JScrollPane(areadetexto3,
ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
static JSplitPane panel14=new JSplitPane(JSplitPane.HORIZONTAL_SPLIT,
panel15,panel16);
static JScrollPane paneldesplazable4=new JScrollPane(panel13,
ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS);
static JSplitPane panel11=new JSplitPane(JSplitPane.HORIZONTAL_SPLIT,
paneldesplazable4,paneldesplazable);

static Object[] FILAENBLANCO={null};
static String[][] informaciondetrama;
static JLabel[][] tramacolor;

static PacketCapture captador=new PacketCapture();

static plantillaoyentedemouse oyentedemouse =new plantillaoyentedemouse();
static int ultimavista;

static Runnable actualizar = new Runnable() {
public void run() {

        informaciondetrama[1][1]="Trama N° "+tramasrecibidas;

        if(ipa.equals(control[1])){
                sdireccion="A => ";
        }else if(ipb.equals(control[1])){
                sdireccion="B => ";
                informaciondetrama[3][1]=sdireccion;
        }else if(ipc.equals(control[1])){
                sdireccion="C => ";
                informaciondetrama[3][1]=sdireccion;
        }else{
                sdireccion="  => ";
                informaciondetrama[3][1]=sdireccion;
        }
        if(ipa.equals(control[2])){
                sdireccion=sdireccion+"A";
        }else if(ipb.equals(control[2])){
                sdireccion=sdireccion+"B";
                informaciondetrama[3][1]=sdireccion;
        }else if(ipc.equals(control[2])){
                sdireccion=sdireccion+"C";
                informaciondetrama[3][1]=sdireccion;
        }else{
                sdireccion=sdireccion+" ";
                informaciondetrama[3][1]=sdireccion;
        }

        tramacolor=Utilidades.Colorear(informaciondetrama);

        modelotablaindice2.addRow(new JLabel[] {new Etiqueta1("Trama N° "+tramasrecibidas)});
        if(protocolos[2]!="desconocido"){
                modelotablaindice2.addRow(new JLabel[] {new Etiqueta1(protocolos[2])});
        }else{
                modelotablaindice2.addRow(new JLabel[] {new Etiqueta1("")});
        }
        modelotablaindice2.addRow(new JLabel[] {new Etiqueta1(sdireccion)});
        modelotablaindice2.addRow(FILAENBLANCO);

        modelotablaindice3.addRow(new JLabel[] {new Etiqueta1("Trama N° "+tramasrecibidas)});
        if(protocolos[3]!="desconocido"){
                modelotablaindice3.addRow(new JLabel[] {new Etiqueta1(protocolos[3])});
        }else{
                modelotablaindice3.addRow(new JLabel[] {new Etiqueta1("")});
```

```
}
modelotablaindice3.addRow(new JLabel[] {new Etiqueta1(sdireccion)});
modelotablaindice3.addRow(FILAENBLANCO);

modelotablaindice4.addRow(new JLabel[] {new Etiqueta1("Trama N° "+tramasrecibidas)});
if(protocolos[4]!="desconocido"){
	modelotablaindice4.addRow(new JLabel[] {new Etiqueta1(protocolos[4])});
}else{
	modelotablaindice4.addRow(new JLabel[] {new Etiqueta1("")});
}
modelotablaindice4.addRow(new JLabel[] {new Etiqueta1(sdireccion)});
modelotablaindice4.addRow(FILAENBLANCO);

modelotablaindice5.addRow(new JLabel[] {new Etiqueta1("Trama N° "+tramasrecibidas)});
if(protocolos[5]!="desconocido"){
	modelotablaindice5.addRow(new JLabel[] {new Etiqueta1(protocolos[5])});
}else{
	modelotablaindice5.addRow(new JLabel[] {new Etiqueta1("")});
}
modelotablaindice5.addRow(new JLabel[] {new Etiqueta1(sdireccion)});
modelotablaindice5.addRow(FILAENBLANCO);

modelotablaindice6.addRow(new JLabel[] {new Etiqueta1("Trama N° "+tramasrecibidas)});
if(protocolos[6]!="desconocido"){
	modelotablaindice6.addRow(new JLabel[] {new Etiqueta1(protocolos[6])});
}else{
	modelotablaindice6.addRow(new JLabel[] {new Etiqueta1("")});
}
modelotablaindice6.addRow(new JLabel[] {new Etiqueta1(sdireccion)});
modelotablaindice6.addRow(FILAENBLANCO);

contador=0;
modelotablacapa2.addRow(FILAENBLANCO);
modelotablacapa2.addRow(FILAENBLANCO);
modelotablacapa2.addRow(FILAENBLANCO);
for(int i=2;i<informaciondetrama[0].length;i++){
	if(informaciondetrama[0][i]=="capa2"||informaciondetrama[0][i]=="t"
		||informaciondetrama[0][i]=="datos2"
		||informaciondetrama[0][i]=="desconocido2"){
		modelotablacapa2.setValueAt(tramacolor[0][i],
			modelotablacapa2.getRowCount()-3,contador);
		modelotablacapa2.setValueAt(tramacolor[1][i],
			modelotablacapa2.getRowCount()-2,contador);
		modelotablacapa2.setValueAt(tramacolor[2][i],
			modelotablacapa2.getRowCount()-1,contador);
		contador++;
	}
}
modelotablacapa2.addRow(FILAENBLANCO);

contador=0;
modelotablacapa3.addRow(FILAENBLANCO);
modelotablacapa3.addRow(FILAENBLANCO);
modelotablacapa3.addRow(FILAENBLANCO);
for(int i=2;i<informaciondetrama[0].length;i++){
	if(informaciondetrama[0][i]=="capa3"||informaciondetrama[0][i]=="t"
		||informaciondetrama[0][i]=="datos3"
		||informaciondetrama[0][i]=="desconocido3"){
		modelotablacapa3.setValueAt(tramacolor[0][i],
			modelotablacapa3.getRowCount()-3,contador);
		modelotablacapa3.setValueAt(tramacolor[1][i],
			modelotablacapa3.getRowCount()-2,contador);
		modelotablacapa3.setValueAt(tramacolor[2][i],
			modelotablacapa3.getRowCount()-1,contador);
		contador++;
	}
}
```

```
modelotablacapa3.addRow(FILAENBLANCO);

contador=0;
modelotablacapa4.addRow(FILAENBLANCO);
modelotablacapa4.addRow(FILAENBLANCO);
modelotablacapa4.addRow(FILAENBLANCO);
for(int i=2;i<informaciondetrama[0].length;i++){
        if(informaciondetrama[0][i]=="capa4"||informaciondetrama[0][i]=="t"
                ||informaciondetrama[0][i]=="datos4"
                ||informaciondetrama[0][i]=="desconocido4"){
                modelotablacapa4.setValueAt(tramacolor[0][i],
                        modelotablacapa4.getRowCount()-3,contador);
                modelotablacapa4.setValueAt(tramacolor[1][i],
                        modelotablacapa4.getRowCount()-2,contador);
                modelotablacapa4.setValueAt(tramacolor[2][i],
                        modelotablacapa4.getRowCount()-1,contador);
                contador++;
        }
}
modelotablacapa4.addRow(FILAENBLANCO);

contador=0;
modelotablacapa5.addRow(FILAENBLANCO);
modelotablacapa5.addRow(FILAENBLANCO);
modelotablacapa5.addRow(FILAENBLANCO);
for(int i=2;i<informaciondetrama[0].length;i++){
        if(informaciondetrama[0][i]=="capa5"||informaciondetrama[0][i]=="t"
                ||informaciondetrama[0][i]=="datos5"
                ||informaciondetrama[0][i]=="desconocido5"){
                modelotablacapa5.setValueAt(tramacolor[0][i],
                        modelotablacapa5.getRowCount()-3,contador);
                modelotablacapa5.setValueAt(tramacolor[1][i],
                        modelotablacapa5.getRowCount()-2,contador);
                modelotablacapa5.setValueAt(tramacolor[2][i],
                        modelotablacapa5.getRowCount()-1,contador);
                contador++;
        }
}
modelotablacapa5.addRow(FILAENBLANCO);

contador=0;
modelotablacapa6.addRow(FILAENBLANCO);
modelotablacapa6.addRow(FILAENBLANCO);
modelotablacapa6.addRow(FILAENBLANCO);
for(int i=2;i<informaciondetrama[0].length;i++){
        if(informaciondetrama[0][i]=="capa6"||informaciondetrama[0][i]=="t"
                ||informaciondetrama[0][i]=="datos6"
                ||informaciondetrama[0][i]=="desconocido6"){
                modelotablacapa6.setValueAt(tramacolor[0][i],
                        modelotablacapa6.getRowCount()-3,contador);
                modelotablacapa6.setValueAt(tramacolor[1][i],
                        modelotablacapa6.getRowCount()-2,contador);
                modelotablacapa6.setValueAt(tramacolor[2][i],
                        modelotablacapa6.getRowCount()-1,contador);
                contador++;
        }
}
modelotablacapa6.addRow(FILAENBLANCO);

contador=0;
modelotablatodaslascapas.addRow(FILAENBLANCO);
modelotablatodaslascapas.addRow(FILAENBLANCO);
modelotablatodaslascapas.addRow(FILAENBLANCO);
for(int i=1;i<informaciondetrama[0].length;i++){
        if(informaciondetrama[0][i]=="capa2"||informaciondetrama[0][i]=="capa3"
        ||informaciondetrama[0][i]=="capa4"||informaciondetrama[0][i]=="capa5"
        ||informaciondetrama[0][i]=="capa6"){
```

```
                                        modelotablatodaslascapas.setValueAt(tramacolor[0][i],
                                                modelotablatodaslascapas.getRowCount()-3,contador);
                                        modelotablatodaslascapas.setValueAt(tramacolor[1][i],
                                                modelotablatodaslascapas.getRowCount()-2,contador);
                                        modelotablatodaslascapas.setValueAt(tramacolor[2][i],
                                                modelotablatodaslascapas.getRowCount()-1,contador);
                                        contador++;
                                }
                        }
                        if(informaciondetrama[0][0]=="desconocido2"||informaciondetrama[0][0]=="desconocido3"
                                ||informaciondetrama[0][0]=="desconocido4"||informaciondetrama[0][0]=="desconocido5"
                                ||informaciondetrama[0][0]=="desconocido6"){
                                        modelotablatodaslascapas.setValueAt(tramacolor[0][0],
                                                modelotablatodaslascapas.getRowCount()-3,contador);
                                        modelotablatodaslascapas.setValueAt(tramacolor[1][0],
                                                modelotablatodaslascapas.getRowCount()-2,contador);
                                        modelotablatodaslascapas.setValueAt(tramacolor[2][0],
                                                modelotablatodaslascapas.getRowCount()-1,contador);
                                        contador++;
                                }
                        caja.addItem("Trama N° "+tramasrecibidas);
                        caja2.addItem("Trama N° "+tramasrecibidas);
                        caja3.addItem("Trama N° "+tramasrecibidas);
        }
};

        public static void creartablas(){

                        tablacapa2.setShowGrid(false);
                        tablacapa3.setShowGrid(false);
                        tablacapa4.setShowGrid(false);
                        tablacapa5.setShowGrid(false);
                        tablacapa6.setShowGrid(false);
                        tablatrama.setShowGrid(false);
                        tablaindice2.setShowGrid(false);
                        tablaindice3.setShowGrid(false);
                        tablaindice4.setShowGrid(false);
                        tablaindice5.setShowGrid(false);
                        tablaindice6.setShowGrid(false);

                        tablacapa2.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablacapa3.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablacapa4.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablacapa5.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablacapa6.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablatrama.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablaindice2.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablaindice3.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablaindice4.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablaindice5.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());
                        tablaindice6.setDefaultRenderer(JLabel.class,new MostradordeEtiqueta());

                        for(int i=0;i<NUMERODECOLUMNAS;i++){
                                modelotablacapa2.addColumn("");
                                modelotablacapa3.addColumn("");
                                modelotablacapa4.addColumn("");
                                modelotablacapa5.addColumn("");
                                modelotablacapa6.addColumn("");
                        }

                        modelotablaindice2.addColumn("");
                        modelotablaindice3.addColumn("");
                        modelotablaindice4.addColumn("");
                        modelotablaindice5.addColumn("");
                        modelotablaindice6.addColumn("");

                        for(int i=0;i<5;i++){
```

```
                        modelotablatrama.addColumn("");
                }

                for(int i=0;i<NUMERODECOLUMNAS2;i++){
                        modelotablatodaslascapas.addColumn("");
                }

                for(int i=0;i<NUMERODECOLUMNAS;i++){
                        (tablacapa2.getColumnModel().getColumn(i)).setPreferredWidth(150);
                        (tablacapa3.getColumnModel().getColumn(i)).setPreferredWidth(150);
                        (tablacapa4.getColumnModel().getColumn(i)).setPreferredWidth(150);
                        (tablacapa5.getColumnModel().getColumn(i)).setPreferredWidth(150);
                        (tablacapa6.getColumnModel().getColumn(i)).setPreferredWidth(150);
                }

                (tablaindice2.getColumnModel().getColumn(0)).setPreferredWidth(90);
                (tablaindice3.getColumnModel().getColumn(0)).setPreferredWidth(90);
                (tablaindice4.getColumnModel().getColumn(0)).setPreferredWidth(90);
                (tablaindice5.getColumnModel().getColumn(0)).setPreferredWidth(90);
                (tablaindice6.getColumnModel().getColumn(0)).setPreferredWidth(90);

                modelotablacapa2.addRow(FILAENBLANCO);
                modelotablacapa3.addRow(FILAENBLANCO);
                modelotablacapa4.addRow(FILAENBLANCO);
                modelotablacapa5.addRow(FILAENBLANCO);
                modelotablacapa6.addRow(FILAENBLANCO);
                modelotablatrama.addRow(FILAENBLANCO);
                modelotablaindice2.addRow(FILAENBLANCO);
                modelotablaindice3.addRow(FILAENBLANCO);
                modelotablaindice4.addRow(FILAENBLANCO);
                modelotablaindice5.addRow(FILAENBLANCO);
                modelotablaindice6.addRow(FILAENBLANCO);
                for(int i=0;i<(NUMERODECOLUMNAS2*4/5);i++){
                        modelotablatrama.addRow(FILAENBLANCO);
                }
                tablacapa2.addMouseListener(oyentedemouse);
                tablacapa3.addMouseListener(oyentedemouse);
                tablacapa4.addMouseListener(oyentedemouse);
                tablacapa5.addMouseListener(oyentedemouse);
                tablacapa6.addMouseListener(oyentedemouse);
                tablaindice2.addMouseListener(oyentedemouse);
                tablaindice3.addMouseListener(oyentedemouse);
                tablaindice4.addMouseListener(oyentedemouse);
                tablaindice5.addMouseListener(oyentedemouse);
                tablaindice6.addMouseListener(oyentedemouse);
        }

        static void crearymostrargui(){

                botonaceptar.addActionListener(new ActionListener(){
                        public void actionPerformed(ActionEvent event) {
                                if(!primeravez&&(dispositivoseleccionado!=listadispositivos.getSelectedIndex()
                                        ||modomonitor!=casillamodomonitor.isSelected())){
                                        JOptionPane.showMessageDialog(null,"Ha modificado la configuración
de captura. Para que este cambio \ntenga efecto debe reiniciar el programa.\n",
                                                        "Debe reiniciar el
programa.",JOptionPane.WARNING_MESSAGE);
                                }
                                dispositivoseleccionado=listadispositivos.getSelectedIndex();
                                dispositivousado=dispositivos[dispositivoseleccionado];
                                filtrarip=casillafiltrarip.isSelected();
                                filtrarpuertos=casillafiltrarpuertos.isSelected();
                                filtrarprotocolo=casillafiltrarprotocolo.isSelected();
                                modomonitor=casillamodomonitor.isSelected();
                                ipa=cuadrodetexto2.getText();
                                ipb=cuadrodetexto3.getText();
                                ipc=cuadrodetexto4.getText();
```

```
                                puerto1=cuadrodetextopuerto1.getText();
                                puerto2=cuadrodetextopuerto2.getText();
                                puerto3=cuadrodetextopuerto3.getText();
                                q931=casillaq931.isSelected();
                                //sip=casillasip.isSelected();
                                try{
                                        escribiropciones();
                                }catch(Exception e) {
                                        e.printStackTrace();
                                }
                                dialogoopciones.dispose();
                                detener=false;
                        }
                });

                botoncancelar.addActionListener(new ActionListener(){
                        public void actionPerformed(ActionEvent event){
                                dialogoopciones.dispose();
                                actualizardialogo();
                                detener=false;
                        }
                });


                (paneldesplazable.getVerticalScrollBar()).addAdjustmentListener(new AdjustmentListener(){
                        public void adjustmentValueChanged(AdjustmentEvent event){

(paneldesplazable4.getVerticalScrollBar()).setValue((paneldesplazable.getVerticalScrollBar()).getValue());
                        }
                });

                (paneldesplazable4.getVerticalScrollBar()).addAdjustmentListener(new AdjustmentListener(){
                        public void adjustmentValueChanged(AdjustmentEvent event){

(paneldesplazable.getVerticalScrollBar()).setValue((paneldesplazable4.getVerticalScrollBar()).getValue());
                        }
                });

                panel11.setOneTouchExpandable(true);
                panel14.setOneTouchExpandable(true);

                panel1.setBorder(BorderFactory.createEmptyBorder(5,5,5,5));
                try{
                        dispositivos = PacketCapture.lookupDevices();
                }catch(Exception e) {
                        e.printStackTrace();
                }
                listadispositivos = new JComboBox(dispositivos);
                panel2.setLayout(new FlowLayout(FlowLayout.LEFT));
                panel2.add(etiqueta1);
                panel2.add(listadispositivos);
                panel2.setBorder(BorderFactory.createCompoundBorder(
                        BorderFactory.createTitledBorder("Adaptador de Red"),
                        BorderFactory.createEmptyBorder(5,5,5,5)));
                panel1.setLayout(gridbag);
                constraints.fill=GridBagConstraints.HORIZONTAL;
                constraints.anchor = GridBagConstraints.NORTH;
                constraints.gridx=0;
                constraints.gridy=0;
                constraints.gridwidth=2;
                constraints.gridheight=2;
                gridbag.setConstraints(panel2,constraints);
                panel1.add(panel2);

                panel3.setLayout(new GridLayout(4,1));
                panel3.setBorder(BorderFactory.createCompoundBorder(
                        BorderFactory.createTitledBorder("Opciones de Captura"),
```

```
            BorderFactory.createEmptyBorder(5,5,5,5)));
panel3.add(casillafiltrarip);
panel3.add(casillafiltrarpuertos);
panel3.add(casillafiltrarprotocolo);
panel3.add(casillamodomonitor);
constraints.fill=GridBagConstraints.BOTH;
constraints.anchor = GridBagConstraints.NORTH;
constraints.gridx=0;
constraints.gridy=2;
constraints.gridwidth=1;
constraints.gridheight=4;
gridbag.setConstraints(panel3,constraints);
panel1.add(panel3);

panel4.setLayout(new GridLayout(3,2));
panel4.add(etiqueta2);
panel4.add(cuadrodetexto2);
panel4.add(etiqueta3);
panel4.add(cuadrodetexto3);
panel4.add(etiqueta4);
panel4.add(cuadrodetexto4);
panel7.setBorder(BorderFactory.createCompoundBorder(
            BorderFactory.createTitledBorder("Direcciones IP"),
            BorderFactory.createEmptyBorder(5,5,5,5)));
panel7.setLayout(new FlowLayout(FlowLayout.LEFT));
panel7.add(panel4);
constraints.fill=GridBagConstraints.BOTH;
constraints.anchor = GridBagConstraints.NORTH;
constraints.gridx=0;
constraints.gridy=6;
constraints.gridwidth=1;
constraints.gridheight=3;
gridbag.setConstraints(panel7,constraints);
panel1.add(panel7);

panel9.setLayout(new GridLayout(3,2));
panel9.add(etiquetapuerto1);
panel9.add(cuadrodetextopuerto1);
panel9.add(etiquetapuerto2);
panel9.add(cuadrodetextopuerto2);
panel9.add(etiquetapuerto3);
panel9.add(cuadrodetextopuerto3);
panel10.setBorder(BorderFactory.createCompoundBorder(
            BorderFactory.createTitledBorder("Puertos"),
            BorderFactory.createEmptyBorder(5,5,5,5)));
panel10.setLayout(new FlowLayout(FlowLayout.LEFT));
panel10.add(panel9);
constraints.fill=GridBagConstraints.HORIZONTAL;
constraints.anchor = GridBagConstraints.NORTH;
constraints.gridx=1;
constraints.gridy=2;
constraints.gridheight=3;
constraints.gridwidth=1;
gridbag.setConstraints(panel10,constraints);
panel1.add(panel10);

panel5.setLayout(new GridLayout(2,1));
panel5.setBorder(BorderFactory.createCompoundBorder(
            BorderFactory.createTitledBorder("Protocolos"),
            BorderFactory.createEmptyBorder(5,5,5,5)));
panel5.add(casillaq931);
//panel5.add(casillasip);
constraints.fill=GridBagConstraints.HORIZONTAL;
constraints.anchor = GridBagConstraints.NORTH;
constraints.gridx=1;
constraints.gridy=5;
constraints.gridwidth=1;
```

```
constraints.gridheight=2;
gridbag.setConstraints(panel5,constraints);
panel1.add(panel5);

panel6.add(botonaceptar);
panel6.add(botoncancelar);
constraints.gridx=1;
constraints.gridy=7;
constraints.fill=GridBagConstraints.BOTH;
constraints.anchor = GridBagConstraints.CENTER;
constraints.gridwidth=1;
constraints.gridheight=2;
gridbag.setConstraints(panel6,constraints);
panel1.add(panel6);

dialogoopciones.getContentPane().add(panel1);
dialogoopciones.addWindowListener(new WindowAdapter(){
        public void windowClosing(WindowEvent e) {
                actualizardialogo();
                detener=false;
        }
});
dialogoopciones.pack();
dialogoopciones.setLocationRelativeTo(null);
dialogoopciones.setResizable(false);

botonnuevo.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent event){
                renovar();
        }
});

electordearchivo.setFileFilter(new javax.swing.filechooser.FileFilter(){
                public String getDescription() {
            return "Documento de texto (*.txt)";
          }
          public boolean accept(File arch){
                return true;
          }
        });

botonguardar.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent event){
                detener=true;

                if(electordearchivo.showSaveDialog(marco)==JFileChooser.APPROVE_OPTION){
                        if(electordearchivo.getFileFilter().getDescription()=="Documento de
texto (*.txt)"){
                                archivo=new
File(electordearchivo.getSelectedFile().getPath()+".txt");
                        }else{
                                archivo=electordearchivo.getSelectedFile();
                        }
                        try{
                                salida3= new FileWriter(archivo);
                                fecha=new Date();
                                salida3.write(fecha.toString());
                                salida3.write("\r\n");
                                salida3.write("\r\n");
                                for(int i=1;i<=tramasrecibidas;i++){
                                        salida3.write("Trama N° "+i);
                                        salida3.write("      ");
                                        salida3.write(direcciónde trama(i));
                                        salida3.write("\r\n");
                                        salida3.write(textodetrama3(i));
                                        salida3.write("\r\n");
                                        salida3.write("\r\n");
```

```
                                        }
                                        salida3.close();
                                }catch(Exception e){
                                        e.printStackTrace();
                                }
                        }
                        detener=false;
                }
});

botonescuchar.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        if(botonescuchar.isSelected()){
                                botondetener.setSelected(false);
                        }else{
                                botonescuchar.setSelected(true);
                        }
                }
});

botondetener.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        if (botondetener.isSelected()){
                                botonescuchar.setSelected(false);
                        }else{
                                botondetener.setSelected(true);
                        }
                }
});

botoncapa2.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        if (botoncapa2.isSelected()){
                                botontramaanterior.setEnabled(false);
                                caja.setEnabled(false);
                                direccion.setEnabled(false);
                                direccion.setText("");
                                botontramasiguiente.setEnabled(false);
                                botoncapa3.setSelected(false);
                                botoncapa4.setSelected(false);
                                botoncapa5.setSelected(false);
                                botoncapa6.setSelected(false);
                                botontramaportrama.setSelected(false);
                                botontexto.setSelected(false);

                                botondatos.setSelected(false);
                                cl.show(tarjetas,"tablacapa2");
                                cl3.show(panel13,"tablaindice2");
                                cl2.show(panel8,"tablascapas");
                        }else{
                                botoncapa2.setSelected(true);
                        }
                }
});

botoncapa3.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        if (botoncapa3.isSelected()){
                                botontramaanterior.setEnabled(false);
                                caja.setEnabled(false);
                                direccion.setEnabled(false);
                                direccion.setText("");
                                botontramasiguiente.setEnabled(false);
                                botoncapa2.setSelected(false);
                                botoncapa4.setSelected(false);
                                botoncapa5.setSelected(false);
                                botoncapa6.setSelected(false);
```

```
                                botontramaportrama.setSelected(false);
                                botontexto.setSelected(false);
                                botondatos.setSelected(false);
                                cl.show(tarjetas,"tablacapa3");
                                cl3.show(panel13,"tablaindice3");
                                cl2.show(panel8,"tablascapas");
                        }else{
                                botoncapa3.setSelected(true);
                        }
                }
});

botoncapa4.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        if (botoncapa4.isSelected()){
                                botontramaanterior.setEnabled(false);
                                caja.setEnabled(false);
                                direccion.setEnabled(false);
                                direccion.setText("");
                                botontramasiguiente.setEnabled(false);
                                botoncapa2.setSelected(false);
                                botoncapa3.setSelected(false);
                                botoncapa5.setSelected(false);
                                botoncapa6.setSelected(false);
                                botontramaportrama.setSelected(false);
                                botontexto.setSelected(false);
                                botondatos.setSelected(false);
                                cl.show(tarjetas,"tablacapa4");
                                cl3.show(panel13,"tablaindice4");
                                cl2.show(panel8,"tablascapas");
                        }else{
                                botoncapa4.setSelected(true);
                        }
                }
});

botoncapa5.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        if (botoncapa5.isSelected()){
                                botontramaanterior.setEnabled(false);
                                caja.setEnabled(false);
                                direccion.setEnabled(false);
                                direccion.setText("");
                                botontramasiguiente.setEnabled(false);
                                botoncapa2.setSelected(false);
                                botoncapa3.setSelected(false);
                                botoncapa4.setSelected(false);
                                botoncapa6.setSelected(false);
                                botontramaportrama.setSelected(false);
                                botontexto.setSelected(false);
                                botondatos.setSelected(false);
                                cl.show(tarjetas,"tablacapa5");
                                cl3.show(panel13,"tablaindice5");
                                cl2.show(panel8,"tablascapas");
                        }else{
                                botoncapa5.setSelected(true);
                        }
                }
});

botoncapa6.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        if (botoncapa6.isSelected()){
                                botontramaanterior.setEnabled(false);
                                caja.setEnabled(false);
                                direccion.setEnabled(false);
                                direccion.setText("");
```

```
                botontramasiguiente.setEnabled(false);
                botoncapa2.setSelected(false);
                botoncapa3.setSelected(false);
                botoncapa4.setSelected(false);
                botoncapa5.setSelected(false);
                botontramaportrama.setSelected(false);
                botontexto.setSelected(false);
                botondatos.setSelected(false);
                cl.show(tarjetas,"tablacapa6");
                cl3.show(panel13,"tablaindice6");
                cl2.show(panel8,"tablascapas");
            }else{
                botoncapa6.setSelected(true);
            }
        }
});

botontramaanterior.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent event){
            if (numerotrama>1){
                numerotrama--;
                caja.setSelectedIndex(numerotrama-1);
                direccion.setText(direcciónde trama(numerotrama));
                if(botontramaportrama.isSelected()){
                        mostrartrama();
                }else if(botontexto.isSelected()){
                        areadetexto3.setText(textodetrama(numerotrama));
                }
                areadetexto3.setCaretPosition(0);
            }
        }
});

botontramaanterior2.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent event){
            if (numerotrama>1){
                numerotrama--;
                caja2.setSelectedIndex(numerotrama-1);
                direccion2.setText(direcciónde trama(numerotrama));
                areadetexto.setText(textodetrama2(numerotrama));
                areadetexto.setCaretPosition(0);
            }
        }
});

botontramaanterior3.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent event){
            if (numerotrama3>1){
                numerotrama3--;
                caja3.setSelectedIndex(numerotrama3-1);
                direccion3.setText(direcciónde trama(numerotrama3));
                areadetexto2.setText(textodetrama2(numerotrama3));
                areadetexto2.setCaretPosition(0);
            }
        }
});

botontramasiguiente.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent event){
            if (numerotrama<tramasrecibidas){
                numerotrama++;
                caja.setSelectedIndex(numerotrama-1);
                direccion.setText(direcciónde trama(numerotrama));
                if(botontramaportrama.isSelected()){
                        mostrartrama();
                }else if(botontexto.isSelected()){
                        areadetexto3.setText(textodetrama(numerotrama));
```

```
                        }
                        areadetexto3.setCaretPosition(0);
                    }
                }
});

caja.addItemListener(new ItemListener(){
                public void itemStateChanged(ItemEvent e){
                    if (caja.getSelectedIndex()>=0){
                        numerotrama=caja.getSelectedIndex()+1;
                        direccion.setText(direcciónde trama(numerotrama));
                        if(botontramaportrama.isSelected()){
                            mostrartrama();
                        }else if(botontexto.isSelected()){
                            areadetexto3.setText(textodetrama(numerotrama));
                        }
                        areadetexto3.setCaretPosition(0);
                    }
                }
});

caja2.addItemListener(new ItemListener(){
                public void itemStateChanged(ItemEvent e){
                    if (caja2.getSelectedIndex()>=0){
                        numerotrama=caja2.getSelectedIndex()+1;
                        direccion2.setText(direcciónde trama(numerotrama));
                        areadetexto.setText(textodetrama2(numerotrama));
                        areadetexto.setCaretPosition(0);
                    }
                }
});

caja3.addItemListener(new ItemListener(){
                public void itemStateChanged(ItemEvent e){
                    if (caja3.getSelectedIndex()>=0){
                        numerotrama3=caja3.getSelectedIndex()+1;
                        direccion3.setText(direcciónde trama(numerotrama3));
                        areadetexto2.setText(textodetrama2(numerotrama3));
                        areadetexto2.setCaretPosition(0);
                    }
                }
});

botontramasiguiente2.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                    if (numerotrama<tramasrecibidas){
                        numerotrama++;
                        caja2.setSelectedIndex(numerotrama-1);
                        direccion2.setText(direcciónde trama(numerotrama));
                        areadetexto.setText(textodetrama2(numerotrama));
                        areadetexto.setCaretPosition(0);
                    }
                }
});

botontramasiguiente3.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                    if (numerotrama3<tramasrecibidas){
                        numerotrama3++;
                        caja3.setSelectedIndex(numerotrama3-1);
                        direccion3.setText(direcciónde trama(numerotrama3));
                        areadetexto2.setText(textodetrama2(numerotrama3));
                        areadetexto2.setCaretPosition(0);
                    }
                }
});
```

```
botontramaportrama.addActionListener(new ActionListener(){
	public void actionPerformed(ActionEvent event){
		ultimavista=1;
		if (botontramaportrama.isSelected()){
			if(tramasrecibidas>0&&numerotrama<1){
				numerotrama=1;
				caja.setSelectedIndex(numerotrama-1);
				direccion.setText(direcciónde trama(numerotrama));
				mostrartrama();
			}else if(tramasrecibidas>0){
				caja.setSelectedIndex(numerotrama-1);
				direccion.setText(direcciónde trama(numerotrama));
				mostrartrama();
			}
			botontramaanterior.setEnabled(true);
			direccion.setEnabled(true);
			caja.setEnabled(true);
			botontramasiguiente.setEnabled(true);
			botoncapa2.setSelected(false);
			botoncapa3.setSelected(false);
			botoncapa4.setSelected(false);
			botoncapa5.setSelected(false);
			botoncapa6.setSelected(false);
			botontexto.setSelected(false);
			botondatos.setSelected(false);
			cl2.show(panel8,"tablatrama");
		}else{
			botontramaportrama.setSelected(true);
		}
	}
});

botontexto.addActionListener(new ActionListener(){
	public void actionPerformed(ActionEvent event){
		ultimavista=2;
		if (botontexto.isSelected()){
			if(tramasrecibidas>0&&numerotrama<1){
				numerotrama=1;
				caja.setSelectedIndex(numerotrama-1);
				direccion.setText(direcciónde trama(numerotrama));
				try{
					mostrartexto();
				}catch(Exception e){
					e.printStackTrace();
				}
			}else if(tramasrecibidas>0){
				caja.setSelectedIndex(numerotrama-1);
				direccion.setText(direcciónde trama(numerotrama));
				try{
					mostrartexto();
				}catch(Exception e){
					e.printStackTrace();
				}
			}
			areadetexto3.setCaretPosition(0);
			botontramaanterior.setEnabled(true);
			caja.setEnabled(true);
			direccion.setEnabled(true);
			botontramasiguiente.setEnabled(true);
			botoncapa2.setSelected(false);
			botoncapa3.setSelected(false);
			botoncapa4.setSelected(false);
			botoncapa5.setSelected(false);
			botoncapa6.setSelected(false);
			botontramaportrama.setSelected(false);
			botondatos.setSelected(false);
			cl2.show(panel8,"areadetexto");
```

```
                        }else{
                                botontexto.setSelected(true);
                        }
                }
        });

        botondatos.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        ultimavista=3;
                        if (botondatos.isSelected()){
                                if(tramasrecibidas>0&&numerotrama<1&&numerotrama3<1){
                                        numerotrama=1;
                                        numerotrama3=1;
                                        caja2.setSelectedIndex(numerotrama-1);
                                        caja3.setSelectedIndex(numerotrama3-1);
                                        direccion2.setText(direcciónde trama(numerotrama));
                                        direccion3.setText(direcciónde trama(numerotrama3));
                                        try{
                                                areadetexto.setText(textodetrama2(numerotrama));

areadetexto2.setText(textodetrama2(numerotrama3));
                                        }catch(Exception e){
                                                e.printStackTrace();
                                        }
                                        areadetexto.setCaretPosition(0);
                                        areadetexto2.setCaretPosition(0);
                                }else if(tramasrecibidas>0){
                                        caja2.setSelectedIndex(numerotrama-1);
                                        caja3.setSelectedIndex(numerotrama3-1);
                                        direccion2.setText(direcciónde trama(numerotrama));
                                        direccion3.setText(direcciónde trama(numerotrama3));
                                        try{
                                                areadetexto.setText(textodetrama2(numerotrama));

areadetexto2.setText(textodetrama2(numerotrama3));
                                        }catch(Exception e){
                                                e.printStackTrace();
                                        }
                                }
                                botontramaanterior.setEnabled(false);
                                //tramanumero.setEnabled(true);
                                caja.setEnabled(false);
                                direccion.setText("");
                                direccion.setEnabled(false);
                                botontramasiguiente.setEnabled(false);
                                botoncapa2.setSelected(false);
                                botoncapa3.setSelected(false);
                                botoncapa4.setSelected(false);
                                botoncapa5.setSelected(false);
                                botoncapa6.setSelected(false);
                                botontramaportrama.setSelected(false);
                                botontexto.setSelected(false);
                                cl2.show(panel8,"datos");
                        }else{
                                botondatos.setSelected(true);
                        }
                }
        });

        botonopciones.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent event){
                        detener=true;
                        dialogoopciones.show();
                }
        });

        botonnuevo.setToolTipText("Nueva captura");
```

```
botonguardar.setToolTipText("Guardar captura en archivo de texto");
botonescuchar.setToolTipText("Iniciar captura");
botondetener.setToolTipText("Pausar captura");
botontramaanterior.setToolTipText("Trama anterior");
botontramaanterior2.setToolTipText("Trama anterior");
botontramaanterior3.setToolTipText("Trama anterior");
botontramasiguiente.setToolTipText("Trama siguiente");
botontramasiguiente2.setToolTipText("Trama siguiente");
botontramasiguiente3.setToolTipText("Trama siguiente");
caja.setToolTipText("Selecionar trama");
caja2.setToolTipText("Selecionar trama");
caja3.setToolTipText("Selecionar trama");
botoncapa2.setToolTipText("Ver elementos del nivel 2");
botoncapa3.setToolTipText("Ver elementos del nivel 3");
botoncapa4.setToolTipText("Ver elementos del nivel 4");
botoncapa5.setToolTipText("Ver elementos del nivel 5");
botoncapa6.setToolTipText("Ver elementos del nivel 6");
botontramaportrama.setToolTipText("Ver trama completa en bloques");
botontexto.setToolTipText("Ver trama completa en texto");
botondatos.setToolTipText("Ver datos dentro de la trama en forma de texto");
botonopciones.setToolTipText("Opciones");

barradeherramientas.setFloatable(false);
barradeherramientas2.setFloatable(false);
barradeherramientas3.setFloatable(false);
barradeherramientas.add(botonnuevo);
//barradeherramientas.add(botonabrir);
barradeherramientas.add(botonguardar);
//barradeherramientas.add(separador1);
barradeherramientas.add(botonescuchar);
barradeherramientas.add(botondetener);
barradeherramientas.add(botontramaanterior);
barradeherramientas2.add(botontramaanterior2);
barradeherramientas3.add(botontramaanterior3);
barradeherramientas.add(botontramasiguiente);
barradeherramientas2.add(botontramasiguiente2);
barradeherramientas3.add(botontramasiguiente3);
barradeherramientas.add(caja);
barradeherramientas2.add(caja2);
barradeherramientas3.add(caja3);
//barradeherramientas.add(separador2);
barradeherramientas.add(botoncapa2);
barradeherramientas.add(botoncapa3);
barradeherramientas.add(botoncapa4);
barradeherramientas.add(botoncapa5);
barradeherramientas.add(botoncapa6);
barradeherramientas.add(botontramaportrama);
barradeherramientas.add(botontexto);
barradeherramientas.add(botondatos);
//barradeherramientas.add(separador3);
barradeherramientas.add(botonopciones);
barradeherramientas.add(direccion);
barradeherramientas2.add(direccion2);
barradeherramientas3.add(direccion3);
botoncapa2.setSelected(true);
botontramaanterior.setEnabled(false);
caja.setMaximumSize(new Dimension(100,caja.getPreferredSize().height));
caja2.setMaximumSize(new Dimension(100,caja2.getPreferredSize().height));
caja3.setMaximumSize(new Dimension(100,caja3.getPreferredSize().height));
direccion.setText("");
caja.setEnabled(false);
direccion.setEnabled(false);
botontramasiguiente.setEnabled(false);

panel15.setLayout(new BorderLayout());
panel15.add(barradeherramientas2,BorderLayout.NORTH);
panel15.add(paneldesplazable3,BorderLayout.CENTER);
```

```
                panel16.setLayout(new BorderLayout());
                panel16.add(barradeherramientas3,BorderLayout.NORTH);
                panel16.add(paneldesplazable5,BorderLayout.CENTER);
                panel14.setDividerLocation(750);
                areadetexto.setMinimumSize(new Dimension(0,0));
                areadetexto.setMaximumSize(new Dimension(1000,1000));
                areadetexto2.setMinimumSize(new Dimension(0,0));
                areadetexto2.setMaximumSize(new Dimension(1000,1000));

                creartablas();
                panel11.setDividerLocation(110);

                tarjetas.setLayout(cl);
                tarjetas.add("tablacapa2", tablacapa2);
                tarjetas.add("tablacapa3", tablacapa3);
                tarjetas.add("tablacapa4", tablacapa4);
                tarjetas.add("tablacapa5", tablacapa5);
                tarjetas.add("tablacapa6", tablacapa6);
cl.show(tarjetas,"tablacapa2");

panel13.setLayout(cl3);
                panel13.add("tablaindice2",tablaindice2);
                panel13.add("tablaindice3",tablaindice3);
                panel13.add("tablaindice4",tablaindice4);
                panel13.add("tablaindice5",tablaindice5);
                panel13.add("tablaindice6",tablaindice6);
                cl3.show(panel13,"tablaindice2");

panel8.setLayout(cl2);
                panel8.add("tablatrama", paneldesplazable2);
                panel8.add("tablascapas", panel11);
                panel8.add("areadetexto", paneldesplazable6);
                panel8.add("datos", panel14);
                cl2.show(panel8,"tablascapas");

                areadetexto.setEditable(false);
                areadetexto2.setEditable(false);
                areadetexto3.setEditable(false);

                marco.getContentPane().setLayout(new BorderLayout());
                marco.addWindowListener(new WindowAdapter(){
                        public void windowClosing(WindowEvent e) {
                                System.exit(0);}
                });
                marco.getContentPane().add(barradeherramientas, BorderLayout.NORTH);
                marco.getContentPane().add(panel8, BorderLayout.CENTER);
                marco.setBounds(100,75,600,450);
                marco.setExtendedState(JFrame.MAXIMIZED_BOTH);
marco.show();
        }

        public static boolean filtro(){
                f=true;
                if(filtrarprotocolo){
                        f=(q931&&(control[0]=="Q.931"));
                }
                if(f&&filtrarpuertos){
                        f=puerto1.equals(control[3])
                                ||puerto2.equals(control[3])
                                ||puerto3.equals(control[3])
                                ||puerto1.equals(control[4])
                                ||puerto2.equals(control[4])
                                ||puerto3.equals(control[4]);
                }
                if(f&&filtrarip){
                        f=ipa.equals(control[1])
                                ||ipb.equals(control[1])
```

```
                                        ||ipc.equals(control[1])
                                        ||ipa.equals(control[2])
                                        ||ipb.equals(control[2])
                                        ||ipc.equals(control[2]);
                        }
                        return f;
                }

                public static void iniciarcaptador() throws Exception{
                        try{
                                captador.open(dispositivousado,2000,modomonitor,3);
                                captador.addRawPacketListener(new RawPacketListener(){
                                        public void rawPacketArrived(RawPacket trama){
                                                if(botonescuchar.isSelected()&&(!detener)){
                                                        cursores[0]=0;
                                                        cursores[1]=trama.getData().length-1;
                                                        cursores[2]=0;
                                                        cursores[3]=NUMERODECOLUMNAS2-1;
                                                        protocolos=new String[7];
                                                        control=new String[5];

                informaciondetrama=Decodificador5.Decodificar(trama.getData(),cursores,protocolos,control);
                                                        if(filtro()){
                                                                try{
                                                                        tramasrecibidas++;
                                                                        if(cursores[0]<trama.getData().length){
                                                                                archivo2=new
File("c:\\"+File.separator+"Analizador"+File.separator+"Temporales"+File.separator+"datos"+tramasrecibidas);
                                                                                archivo2.deleteOnExit();
                                                                                salida2 = new
FileOutputStream(archivo2);

                salida2.write(trama.getData(),cursores[0],trama.getData().length-cursores[0]);
                                                                                salida2.close();
                                                                        }
                                                                        SwingUtilities.
invokeAndWait(actualizar);
                                                                }catch(Exception e) {
                                                                        e.printStackTrace();
                                                                }
                                                                }
                                                }
                                        }
                                });
                                captador.capture(-1);
                        }catch(Exception e) {
                                e.printStackTrace();
                        }

                }

                public static String direcciónde trama(int k){
                        if(modelotablatodaslascapas.getValueAt((k-1)*3+1,21)!=null){
                                if(ipa.equals(((JLabel)modelotablatodaslascapas.getValueAt((k-1)*3+1,19)).getText())){
                                        sdireccion="        Terminal A => ";
                                }else if(ipb.equals(((JLabel)modelotablatodaslascapas.getValueAt((k-1)*3+1,19)).getText())){
                                        sdireccion="        Terminal B => ";
                                }else if(ipc.equals(((JLabel)modelotablatodaslascapas.getValueAt((k-1)*3+1,19)).getText())){
                                        sdireccion="        Terminal C => ";
                                }else{
                                        sdireccion="        "+((JLabel)modelotablatodaslascapas.getValueAt((k-
1)*3+1,19)).getText()+" => ";
                                }
                                if(ipa.equals(((JLabel)modelotablatodaslascapas.getValueAt((k-1)*3+1,20)).getText())){
                                        sdireccion=sdireccion+"Terminal A ";
                                }else if(ipb.equals(((JLabel)modelotablatodaslascapas.getValueAt((k-1)*3+1,20)).getText())){
                                        sdireccion=sdireccion+"Terminal B ";
```

```
                        }else if(ipc.equals(((JLabel)modelotablatodaslascapas.getValueAt((k-1)*3+1,20)).getText())){
                                sdireccion=sdireccion+"Terminal C ";
                        }else{
                                sdireccion=sdireccion+((JLabel)modelotablatodaslascapas.getValueAt((k-
1)*3+1,20)).getText()+" ";
                        }
                }else{
                        sdireccion="";
                }
                return(sdireccion);
        }

        public static String textodetrama(int k){
                StringBuffer sb=new StringBuffer("");

                if(((JLabel)modelotablaindice2.getValueAt((k-1)*4+2,0)).getText()!=null
                        &&((JLabel)modelotablaindice2.getValueAt((k-1)*4+2,0)).getText()!=""){
                        sb.append("****   Cabecera "
                                +((JLabel)modelotablaindice2.getValueAt((k-1)*4+2,0)).getText()
                                +"   ****\r\n");
                        aux=0;
                        while(modelotablacapa2.getValueAt((k-1)*4+1,aux)!=null&&
                                        !"Datos".equals(((JLabel)modelotablacapa2.getValueAt((k-
1)*4+1,aux)).getText())){
                                sb.append(((JLabel)modelotablacapa2.getValueAt((k-1)*4+1,aux)).getText()
                                        +": "+((JLabel)modelotablacapa2.getValueAt((k-1)*4+2,aux)).getText()
                                        +" ("+((JLabel)modelotablacapa2.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                aux++;
                        }
                }

                if(((JLabel)modelotablaindice3.getValueAt((k-1)*4+2,0)).getText()!=null
                        &&((JLabel)modelotablaindice3.getValueAt((k-1)*4+2,0)).getText()!=""){
                        sb.append("****   Cabecera "
                                +((JLabel)modelotablaindice3.getValueAt((k-1)*4+2,0)).getText()
                                +"   ****\r\n");
                        aux=0;
                        while(modelotablacapa3.getValueAt((k-1)*4+1,aux)!=null&&
                                        !"Datos".equals(((JLabel)modelotablacapa3.getValueAt((k-
1)*4+1,aux)).getText())){
                                sb.append(((JLabel)modelotablacapa3.getValueAt((k-1)*4+1,aux)).getText()
                                        +": "+((JLabel)modelotablacapa3.getValueAt((k-1)*4+2,aux)).getText()
                                        +" ("+((JLabel)modelotablacapa3.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                aux++;
                        }
                }

                if(((JLabel)modelotablaindice4.getValueAt((k-1)*4+2,0)).getText()!=null
                        &&((JLabel)modelotablaindice4.getValueAt((k-1)*4+2,0)).getText()!=""){
                        sb.append("****   Cabecera "
                                +((JLabel)modelotablaindice4.getValueAt((k-1)*4+2,0)).getText()
                                +"   ****\r\n");
                        aux=0;
                        while(modelotablacapa4.getValueAt((k-1)*4+1,aux)!=null&&
                                        !"Datos".equals(((JLabel)modelotablacapa4.getValueAt((k-
1)*4+1,aux)).getText())){
                                sb.append(((JLabel)modelotablacapa4.getValueAt((k-1)*4+1,aux)).getText()
                                        +": "+((JLabel)modelotablacapa4.getValueAt((k-1)*4+2,aux)).getText()
                                        +" ("+((JLabel)modelotablacapa4.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                aux++;
                        }
                }

                if(((JLabel)modelotablaindice5.getValueAt((k-1)*4+2,0)).getText()!=null
```

```
                        &&((JLabel)modelotablaindice5.getValueAt((k-1)*4+2,0)).getText()!=""){
                        sb.append("****   Cabecera "
                                +((JLabel)modelotablaindice5.getValueAt((k-1)*4+2,0)).getText()
                                +"   ****\r\n");
                        aux=0;
                        while(modelotablacapa5.getValueAt((k-1)*4+1,aux)!=null&&
                                        !"Datos".equals(((JLabel)modelotablacapa5.getValueAt((k-
1)*4+1,aux)).getText())){
                                sb.append(((JLabel)modelotablacapa5.getValueAt((k-1)*4+1,aux)).getText()
                                        +": "+((JLabel)modelotablacapa5.getValueAt((k-1)*4+2,aux)).getText()
                                        +" ("+((JLabel)modelotablacapa5.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                aux++;
                        }
                }

                if(((JLabel)modelotablaindice6.getValueAt((k-1)*4+2,0)).getText()!=null
                        &&((JLabel)modelotablaindice6.getValueAt((k-1)*4+2,0)).getText()!=""){
                        sb.append("****   Datos "
                                +((JLabel)modelotablaindice6.getValueAt((k-1)*4+2,0)).getText()
                                +"   ****\r\n");
                        aux=0;
                        while(modelotablacapa6.getValueAt((k-1)*4+1,aux)!=null&&
                                        !"Datos".equals(((JLabel)modelotablacapa6.getValueAt((k-
1)*4+1,aux)).getText())){
                                sb.append(((JLabel)modelotablacapa6.getValueAt((k-1)*4+1,aux)).getText()
                                        +": "+((JLabel)modelotablacapa6.getValueAt((k-1)*4+2,aux)).getText()
                                        +" ("+((JLabel)modelotablacapa6.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                aux++;
                        }
                }

                try{
                        archivo2=new
File("c:\\"+File.separator+"Analizador"+File.separator+"Temporales"+File.separator+"datos"+k);
                        if(archivo2.exists()){
                                sb.append("****   Datos ("+archivo2.length()+" Octetos)   ****\r\n");
                        }
                }catch(Exception e){
                        e.printStackTrace();
                }

                return sb.toString();
        }

        public static String textodetrama2(int k){
                String ss=new String();
                try{
                        archivo2=new
File("c:\\"+File.separator+"Analizador"+File.separator+"Temporales"+File.separator+"datos"+k);
                        if(archivo2.exists()){
                                entrada2 = new FileInputStream(archivo2);
                                b=new byte[(int)archivo2.length()];
                                entrada2.read(b);
                                entrada2.close();
                                ss=new String(b,"UTF-8");
                        }
                }catch(Exception e){
                        e.printStackTrace();
                }
                return ss;
        }

        public static String textodetrama3(int k){
                StringBuffer sb=new StringBuffer("");
```

```
                    if(((JLabel)modelotablaindice2.getValueAt((k-1)*4+2,0)).getText()!=null
                            &&((JLabel)modelotablaindice2.getValueAt((k-1)*4+2,0)).getText()!=""){
                            sb.append("************************   Cabecera "
                                    +((JLabel)modelotablaindice2.getValueAt((k-1)*4+2,0)).getText()
                                    +"   ************************\r\n");
                            aux=0;
                            while(modelotablacapa2.getValueAt((k-1)*4+1,aux)!=null&&
                                            !"Datos".equals(((JLabel)modelotablacapa2.getValueAt((k-
1)*4+1,aux)).getText())){
                                    sb.append(((JLabel)modelotablacapa2.getValueAt((k-1)*4+1,aux)).getText()
                                            +": "+((JLabel)modelotablacapa2.getValueAt((k-1)*4+2,aux)).getText()
                                            +" ("+((JLabel)modelotablacapa2.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                    aux++;
                            }
                    }

                    if(((JLabel)modelotablaindice3.getValueAt((k-1)*4+2,0)).getText()!=null
                            &&((JLabel)modelotablaindice3.getValueAt((k-1)*4+2,0)).getText()!=""){
                            sb.append("************************   Cabecera "
                                    +((JLabel)modelotablaindice3.getValueAt((k-1)*4+2,0)).getText()
                                    +"   ************************\r\n");
                            aux=0;
                            while(modelotablacapa3.getValueAt((k-1)*4+1,aux)!=null&&
                                            !"Datos".equals(((JLabel)modelotablacapa3.getValueAt((k-
1)*4+1,aux)).getText())){
                                    sb.append(((JLabel)modelotablacapa3.getValueAt((k-1)*4+1,aux)).getText()
                                            +": "+((JLabel)modelotablacapa3.getValueAt((k-1)*4+2,aux)).getText()
                                            +" ("+((JLabel)modelotablacapa3.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                    aux++;
                            }
                    }

                    if(((JLabel)modelotablaindice4.getValueAt((k-1)*4+2,0)).getText()!=null
                            &&((JLabel)modelotablaindice4.getValueAt((k-1)*4+2,0)).getText()!=""){
                            sb.append("************************   Cabecera "
                                    +((JLabel)modelotablaindice4.getValueAt((k-1)*4+2,0)).getText()
                                    +"   ************************\r\n");
                            aux=0;
                            while(modelotablacapa4.getValueAt((k-1)*4+1,aux)!=null&&
                                            !"Datos".equals(((JLabel)modelotablacapa4.getValueAt((k-
1)*4+1,aux)).getText())){
                                    sb.append(((JLabel)modelotablacapa4.getValueAt((k-1)*4+1,aux)).getText()
                                            +": "+((JLabel)modelotablacapa4.getValueAt((k-1)*4+2,aux)).getText()
                                            +" ("+((JLabel)modelotablacapa4.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                    aux++;
                            }
                    }

                    if(((JLabel)modelotablaindice5.getValueAt((k-1)*4+2,0)).getText()!=null
                            &&((JLabel)modelotablaindice5.getValueAt((k-1)*4+2,0)).getText()!=""){
                            sb.append("************************   Cabecera "
                                    +((JLabel)modelotablaindice5.getValueAt((k-1)*4+2,0)).getText()
                                    +"   ************************\r\n");
                            aux=0;
                            while(modelotablacapa5.getValueAt((k-1)*4+1,aux)!=null&&
                                            !"Datos".equals(((JLabel)modelotablacapa5.getValueAt((k-
1)*4+1,aux)).getText())){
                                    sb.append(((JLabel)modelotablacapa5.getValueAt((k-1)*4+1,aux)).getText()
                                            +": "+((JLabel)modelotablacapa5.getValueAt((k-1)*4+2,aux)).getText()
                                            +" ("+((JLabel)modelotablacapa5.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                    aux++;
                            }
                    }
```

```
                    if(((JLabel)modelotablaindice6.getValueAt((k-1)*4+2,0)).getText()!=null
                            &&((JLabel)modelotablaindice6.getValueAt((k-1)*4+2,0)).getText()!=""){
                            sb.append("**************************   Datos "
                                    +((JLabel)modelotablaindice6.getValueAt((k-1)*4+2,0)).getText()
                                    +"   **************************\r\n");
                            aux=0;
                            while(modelotablacapa6.getValueAt((k-1)*4+1,aux)!=null&&
                                            !"Datos".equals(((JLabel)modelotablacapa6.getValueAt((k-
1)*4+1,aux)).getText())){
                                    sb.append(((JLabel)modelotablacapa6.getValueAt((k-1)*4+1,aux)).getText()
                                            +": "+((JLabel)modelotablacapa6.getValueAt((k-1)*4+2,aux)).getText()
                                            +" ("+((JLabel)modelotablacapa6.getValueAt((k-
1)*4+3,aux)).getText()+")\r\n");
                                    aux++;
                            }
                    }

                    try{
                            archivo2=new
File("c:\\"+File.separator+"Analizador"+File.separator+"Temporales"+File.separator+"datos"+k);
                            if(archivo2.exists()){
                                    sb.append("******************   Data decodificada en UTF-8
******************\r\n");
                                    entrada2 = new FileInputStream(archivo2);
                                    b=new byte[(int)archivo2.length()];
                                    entrada2.read(b);
                                    sb.append(new String(b,"UTF-8"));
                                    entrada2.close();
                            }
                    }catch(Exception e){
                            e.printStackTrace();
                    }
                    return sb.toString();
            }

            public static void mostrartexto() throws IOException{
                    areadetexto3.setText(textodetrama(numerotrama));
            }

            public static void mostrartrama(){
                    contador=0;
                    contador3=0;
                    for(int i=0;i<NUMERODECOLUMNAS2;i++){
                            modelotablatrama.setValueAt(modelotablatodaslascapas.getValueAt((numerotrama-1)*3+0,i)
                                    ,contador+1,contador3);
                            modelotablatrama.setValueAt(modelotablatodaslascapas.getValueAt((numerotrama-1)*3+1,i)
                                    ,contador+2,contador3);
                            modelotablatrama.setValueAt(modelotablatodaslascapas.getValueAt((numerotrama-1)*3+2,i)
                                    ,contador+3,contador3);
                            if(contador3<4){
                                    contador3++;
                            }else{
                                    contador3=0;
                                    contador=contador+4;
                            }
                    }
            }

            public static void leeropciones() throws IOException{

                    entrada = new FileInputStream(archivo);
                    entradaserializada = new ObjectInputStream(entrada);
                    try{
                            primeravez=(Boolean)entradaserializada.readObject();
                       dispositivoseleccionado=(Integer)entradaserializada.readObject();
                       filtrarip=(Boolean)entradaserializada.readObject();
```

```
                filtrarpuertos=(Boolean)entradaserializada.readObject();
                filtrarprotocolo=(Boolean)entradaserializada.readObject();
                modomonitor=(Boolean)entradaserializada.readObject();
                ipa=(String)entradaserializada.readObject();
                ipb=(String)entradaserializada.readObject();
                ipc=(String)entradaserializada.readObject();
                puerto1=(String)entradaserializada.readObject();
                puerto2=(String)entradaserializada.readObject();
                puerto3=(String)entradaserializada.readObject();
                q931=(Boolean)entradaserializada.readObject();
            }catch(Exception e){
                        e.printStackTrace();
            }
            entrada.close();
   entradaserializada.close();

}

public static void escribiropciones() throws IOException{

   salida = new FileOutputStream(archivo);
   salidaserializada = new ObjectOutputStream(salida);
   salidaserializada.writeObject(primeravez);
   salidaserializada.writeObject(dispositivoseleccionado);
   salidaserializada.writeObject(filtrarip);
   salidaserializada.writeObject(filtrarpuertos);
   salidaserializada.writeObject(filtrarprotocolo);
   salidaserializada.writeObject(modomonitor);
   salidaserializada.writeObject(ipa);
   salidaserializada.writeObject(ipb);
   salidaserializada.writeObject(ipc);
   salidaserializada.writeObject(puerto1);
   salidaserializada.writeObject(puerto2);
   salidaserializada.writeObject(puerto3);
   salidaserializada.writeObject(q931);
   salida.close();
   salidaserializada.close();
}

public static void actualizardialogo(){

            listadispositivos.setSelectedIndex(dispositivoseleccionado);
            casillafiltrarip.setSelected(filtrarip);
            casillafiltrarpuertos.setSelected(filtrarpuertos);
            casillafiltrarprotocolo.setSelected(filtrarprotocolo);
            casillamodomonitor.setSelected(modomonitor);
            cuadrodetexto2.setText(ipa);
            cuadrodetexto3.setText(ipb);
            cuadrodetexto4.setText(ipc);
            cuadrodetextopuerto1.setText(puerto1);
            cuadrodetextopuerto2.setText(puerto2);
            cuadrodetextopuerto3.setText(puerto3);
            casillaq931.setSelected(q931);
}

public static void renovar(){
            if(tramasrecibidas>0){
                        caja.removeAllItems();
                        caja2.removeAllItems();
                        caja3.removeAllItems();
                        detener=true;
                        aux=modelotablacapa2.getRowCount()-1;
                        for(int i=aux;i>0;i--){
                                    modelotablacapa2.removeRow(i);
                                    modelotablacapa3.removeRow(i);
                                    modelotablacapa4.removeRow(i);
                                    modelotablacapa5.removeRow(i);
```

```
                        modelotablacapa6.removeRow(i);
                        modelotablaindice2.removeRow(i);
                        modelotablaindice3.removeRow(i);
                        modelotablaindice4.removeRow(i);
                        modelotablaindice5.removeRow(i);
                        modelotablaindice6.removeRow(i);
                }
                aux=modelotablatodaslascapas.getRowCount()-1;
                for(int i=aux;i>=0;i--){
                        modelotablatodaslascapas.removeRow(i);
                }
                for(int i=modelotablatrama.getRowCount()-1;i>0;i--){
                        modelotablatrama.removeRow(i);
                }
                for(int i=0;i<(NUMERODECOLUMNAS2*4/5);i++){
                        modelotablatrama.addRow(FILAENBLANCO);
                }
                areadetexto3.setText("");
                areadetexto2.setText("");
                areadetexto.setText("");
                direccion.setText("");
                direccion2.setText("");
                direccion3.setText("");
                tramasrecibidas=0;
                numerotrama=0;
                numerotrama3=0;
                //tramanumero.setText(" Trama N° "+0+" ");
                listadearchivos=directorio.list();
                for(int i=0;i<listadearchivos.length;i++){
                        archivo2=new File("c:\\"+File.separator+"Analizador"+File.separator+"Temporales"
                                +File.separator+listadearchivos[i]);
                        archivo2.delete();
                }
                detener=false;
        }
}


public static void main(String[] args) throws IOException{

        numerotrama=0;
        numerotrama3=0;
        ultimavista=1;
        archivo = new File("c:\\"+File.separator+"Analizador"+File.separator+"opciones");
        if(!directorio.exists()){
                directorio.mkdirs();
        }
        //archivo.deleteOnExit();
        crearymostrargui();

        if(!archivo.exists()){
                dispositivoseleccionado=dispositivos.length-1;
                dispositivousado=dispositivos[dispositivoseleccionado];
                salida = new FileOutputStream(archivo);
    salidaserializada = new ObjectOutputStream(salida);
    salidaserializada.writeObject(true);
    salidaserializada.writeObject(dispositivoseleccionado);
    salidaserializada.writeObject(false);
    salidaserializada.writeObject(false);
    salidaserializada.writeObject(false);
    salidaserializada.writeObject(false);
    salidaserializada.writeObject("");
    salidaserializada.writeObject("");
    salidaserializada.writeObject("");
    salidaserializada.writeObject("");
    salidaserializada.writeObject("");
```

```
		salidaserializada.writeObject("");
		salidaserializada.writeObject(false);
		//salidaserializada.writeObject(false);
		salida.close();
		salidaserializada.close();
				leeropciones();
				actualizardialogo();
				dialogoopciones.show();
				primeravez=false;
				escribiropciones();
			}else{
				leeropciones();
				actualizardialogo();
				dispositivousado=dispositivos[dispositivoseleccionado];
			}
			try{
				iniciarcaptador();
			}catch(Exception e) {
				e.printStackTrace();
			}
}

public static class plantillaoyentedemouse implements MouseListener {

		public void mouseClicked(MouseEvent e){
			if(e.getClickCount()==2&&((JTable)e.getComponent()).getSelectedRow()>-1
				&&((JTable)e.getComponent()).getSelectedRow()<(tramasrecibidas*4)){
				numerotrama=(((JTable)e.getComponent()).getSelectedRow())/4+1;
				if(numerotrama3<1){
					numerotrama3=1;
				}
				if(ultimavista==1){
					botontramaportrama.setSelected(true);
					caja.setSelectedIndex(numerotrama-1);
					direccion.setText(direcciónde trama(numerotrama));
					mostrartrama();
					botontramaanterior.setEnabled(true);
					direccion.setEnabled(true);
					caja.setEnabled(true);
					botontramasiguiente.setEnabled(true);
					botoncapa2.setSelected(false);
					botoncapa3.setSelected(false);
					botoncapa4.setSelected(false);
					botoncapa5.setSelected(false);
					botoncapa6.setSelected(false);
					botontexto.setSelected(false);
					botondatos.setSelected(false);
					cl2.show(panel8,"tablatrama");
				}else if(ultimavista==2){
					botontexto.setSelected(true);
					caja.setSelectedIndex(numerotrama-1);
					direccion.setText(direcciónde trama(numerotrama));
					try{
						mostrartexto();
					}catch(Exception e2){
						e2.printStackTrace();
					}
					areadetexto3.setCaretPosition(0);
					botontramaanterior.setEnabled(true);
					caja.setEnabled(true);
					direccion.setEnabled(true);
					botontramasiguiente.setEnabled(true);
					botoncapa2.setSelected(false);
					botoncapa3.setSelected(false);
					botoncapa4.setSelected(false);
					botoncapa5.setSelected(false);
					botoncapa6.setSelected(false);
```

```
                                        botontramaportrama.setSelected(false);
                                        botondatos.setSelected(false);
                                        //botontodaslascapas.setSelected(false);
                                        cl2.show(panel8,"areadetexto");
                                }else if(ultimavista==3){
                                        botondatos.setSelected(true);
                                        caja2.setSelectedIndex(numerotrama-1);
                                        caja3.setSelectedIndex(numerotrama3-1);
                                        direccion2.setText(direcciónde trama(numerotrama));
                                        direccion3.setText(direcciónde trama(numerotrama3));
                                        areadetexto.setText(textodetrama2(numerotrama));
                                        areadetexto2.setText(textodetrama2(numerotrama3));
                                        areadetexto.setCaretPosition(0);
                                        areadetexto2.setCaretPosition(0);
                                        botontramaanterior.setEnabled(false);
                                        caja.setEnabled(false);
                                        direccion.setText("");
                                        direccion.setEnabled(false);
                                        botontramasiguiente.setEnabled(false);
                                        botoncapa2.setSelected(false);
                                        botoncapa3.setSelected(false);
                                        botoncapa4.setSelected(false);
                                        botoncapa5.setSelected(false);
                                        botoncapa6.setSelected(false);
                                        botontramaportrama.setSelected(false);
                                        botontexto.setSelected(false);
                                        cl2.show(panel8,"datos");
                                }
                        }
                        ((JTable)e.getComponent()).clearSelection();
                }
                public void mouseEntered(MouseEvent e){
                }
                public void mouseExited(MouseEvent e){
                }
                public void mousePressed(MouseEvent e){
                }
                public void mouseReleased(MouseEvent e){
                }

        }

}
```

2. Clase Decodificador5:

```
public class Decodificador5{

        static final int NUMERODECOLUMNAS2=150;
        static String protocolos[];

        public static String[][] Decodificar(byte[] b,int[] cursores,String[] protocolos,String[] control){

                int p[]=Utilidades.byteaInt(b);
                protocolos[2]="802.3";
                String s[][]=new String[4][NUMERODECOLUMNAS2];
                cursores[2]++;
                cursores[2]++;
                DecodificadorCapa2.DecodificarInicio(p,s,cursores,protocolos,control);
                DecodificadorCapa3.Decodificar(p,s,cursores,protocolos,control);
                DecodificadorCapa4.Decodificar(p,s,cursores,protocolos,control);
                DecodificadorCapa5.Decodificar(p,s,cursores,protocolos,control);
                DecodificadorCapa6.Decodificar(p,s,cursores,protocolos,control);
                DecodificadorCapa2.DecodificarFinal(p,s,cursores,protocolos,control);
                return s;
        }
}
```

3. Clase Decodificador802_3

```
public class Decodificador802_3{

	static String tipo;

	public static void DecodificarInicio(int p[],String s[][],int cursores[], String protocolos[],String[] control){

		tipo="Desconocido";
		s[0][cursores[2]]="capa2";
		s[1][cursores[2]]="Dirección Física Destino";
		s[2][cursores[2]]=Utilidades.IntaHex(p[cursores[0]])+":"+
								Utilidades.IntaHex(p[cursores[0]+1])+":"+
								Utilidades.IntaHex(p[cursores[0]+2])+":"+
								Utilidades.IntaHex(p[cursores[0]+3])+":"+
								Utilidades.IntaHex(p[cursores[0]+4])+":"+
								Utilidades.IntaHex(p[cursores[0]+5]);
		//s[2][cursores[2]]="--:--:--:--:--:--";
		cursores[0]=cursores[0]+6;
		s[3][cursores[2]]="6 Octetos";
		cursores[2]=cursores[2]+1;
		s[0][cursores[2]]="capa2";
		s[1][cursores[2]]="Dirección Física Origen";
		s[2][cursores[2]]=Utilidades.IntaHex(p[cursores[0]])+":"+
								Utilidades.IntaHex(p[cursores[0]+1])+":"+
								Utilidades.IntaHex(p[cursores[0]+2])+":"+
								Utilidades.IntaHex(p[cursores[0]+3])+":"+
								Utilidades.IntaHex(p[cursores[0]+4])+":"+
								Utilidades.IntaHex(p[cursores[0]+5]);
		//s[2][cursores[2]]="--:--:--:--:--:--";
		cursores[0]=cursores[0]+6;
		s[3][cursores[2]]="6 Octetos";
		cursores[2]=cursores[2]+1;

		int c=p[cursores[0]]*256+p[cursores[0]+1];
		s[0][cursores[2]]="capa2";
		if(c<=1500){
			s[1][cursores[2]]="Longitud";
			s[2][cursores[2]]=""+c;
		}else if(c>=1536){
			s[1][cursores[2]]="Tipo";
			tipo=Utilidades.IntaHex(p[cursores[0]])+Utilidades.IntaHex(p[cursores[0]+1]);
			s[2][cursores[2]]=tipo;
		}else{
			s[1][cursores[2]]="Desconocido";
			s[2][cursores[2]]="Desconocido";
		}
		cursores[0]=cursores[0]+2;
		s[3][cursores[2]]="2 Octetos";
		cursores[2]=cursores[2]+1;

		if(tipo.equals("0800")&&c>=1536){
			s[0][cursores[2]]="datos2";
			s[1][cursores[2]]="Datos";
			s[2][cursores[2]]="IPv4";
			protocolos[3]="IPv4";
		}else{
			s[0][cursores[2]]="desconocido2";
			s[1][cursores[2]]="Datos";
			s[2][cursores[2]]="Protocolo Desconocido";
			protocolos[3]="desconocido";
			s[0][0]="desconocido2";
			s[1][0]="Datos";
			s[2][0]="Protocolo Desconocido";
			s[3][0]=""+(p.length-cursores[0])+" Octetos";
		}
```

```
                s[3][cursores[2]]=""+(p.length-cursores[0])+" Octetos";
                cursores[2]=cursores[2]+1;
        }

                public static void DecodificarFinal(int p[],String s[][],int cursores[], String protocolos[],String[] control){

        }

}
```

4. Clase DecodificadorCapa2

```
public class DecodificadorCapa2{

        public static void DecodificarInicio(int p[],String s[][],int cursores[], String protocolos[],String[] control){

                if(protocolos[2]=="802.3"){
                        control[0]="802.3";
                        Decodificador802_3.DecodificarInicio(p,s,cursores,protocolos,control);
                }else{
                        protocolos[3]="desconocido";
                        protocolos[4]="desconocido";
                        protocolos[5]="desconocido";
                        protocolos[6]="desconocido";
                }
        }

        public static void DecodificarFinal(int p[],String s[][],int cursores[], String protocolos[],String[] control){
                if(protocolos[2]=="802.3"){
                        Decodificador802_3.DecodificarFinal(p,s,cursores,protocolos,control);
                }
        }
}
```

5. Clase DecodificadorCapa3

```
public class DecodificadorCapa3{

        public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){

                if(protocolos[3]=="IPv4"){
                        control[0]="IPv4";
                        DecodificadorIPv4.Decodificar(p,s,cursores,protocolos,control);
                }else{
                        protocolos[4]="desconocido";
                        protocolos[5]="desconocido";
                        protocolos[6]="desconocido";
                }
        }
}
```

6. Clase DecodificadorCapa4

```
public class DecodificadorCapa4{

        public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){

                if(protocolos[4]=="TCP"){
                        control[0]="TCP";
                        DecodificadorTCP.Decodificar(p,s,cursores,protocolos,control);
                }else if(protocolos[4]=="UDP"){
                        control[0]="UDP";
                        DecodificadorUDP.Decodificar(p,s,cursores,protocolos,control);
                }else{
                        protocolos[5]="desconocido";
                        protocolos[6]="desconocido";
                }
```

```
	}
}
```

7. Clase DecodificadorCapa5

```
public class DecodificadorCapa5{
	public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){
		if((p.length-cursores[0])>0&&protocolos[5]!="desconocido"){
			if(p[cursores[0]]==3){
				protocolos[5]="TPKT";
				control[0]="TPKT";
				DecodificadorTPKT.Decodificar(p,s,cursores,protocolos,control);
			}else{
				protocolos[5]="desconocido";
			}
		}
	}
}
```

8. Clase DecodificadorCapa6

```
public class DecodificadorCapa6{
	public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){

		if((p.length-cursores[0])>0&&protocolos[6]!="desconocido"){
			if(p[cursores[0]]==8){
				s[2][1]="Q.931";
				protocolos[6]="Q.931";
				control[0]="Q.931";
				s[0][0]="";
				s[1][0]="";
				s[2][0]="";
				s[3][0]="";
				try{
					DecodificadorQ_931.Decodificar(p,s,cursores,protocolos,control);
				}catch(Exception e) {
					e.printStackTrace();
				}
			}
		}
	}
}
```

9. Clase DecodificadorIPv4

```
public class DecodificadorIPv4{

	static int aux;
	static int aux2;
	static int longitudcabecera;
	static boolean finopciones;

	public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){

		s[0][cursores[2]]="capa3";
		s[1][cursores[2]]="Versión IP";
		s[2][cursores[2]]=""+(p[cursores[0]]>>4);
		s[3][cursores[2]]="4 Bits";
		cursores[2]=cursores[2]+1;

		s[0][cursores[2]]="capa3";
		s[1][cursores[2]]="Longitud de Cabecera";
		s[2][cursores[2]]=""+(p[cursores[0]]&15)+" Palabras de 32 Bits";
		longitudcabecera=(p[cursores[0]]&15)*4;
		cursores[0]=cursores[0]+1;
		s[3][cursores[2]]="4 Bits";
		cursores[2]=cursores[2]+1;
```

```
s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Precedencia";
aux=p[cursores[0]]>>5;
if(aux==7){
		s[2][cursores[2]]="Control de Red";
}else if(aux==6){
		s[2][cursores[2]]="Control Entre Redes";
}else if(aux==5){
		s[2][cursores[2]]="CRITICO/ECP";
}else if(aux==4){
		s[2][cursores[2]]="Muy urgente";
}else if(aux==3){
		s[2][cursores[2]]="Urgente";
}else if(aux==2){
		s[2][cursores[2]]="Inmediato";
}else if(aux==1){
		s[2][cursores[2]]="Prioridad";
}else if(aux==0){
		s[2][cursores[2]]="Rutina";
}
s[3][cursores[2]]="3 Bits";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Retraso";
if(((p[cursores[0]]&31)>>4)==0){
		s[2][cursores[2]]="Normal";
}else{
		s[2][cursores[2]]="Bajo";
}
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Rendimiento";
if(((p[cursores[0]]&15)>>3)==0){
		s[2][cursores[2]]="Normal";
}else{
		s[2][cursores[2]]="Alto";
}
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Fiabilidad";
if(((p[cursores[0]]&7)>>2)==0){
		s[2][cursores[2]]="Normal";
}else{
		s[2][cursores[2]]="Alta";
}
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Reservado para uso futuro";
s[2][cursores[2]]="Reservado";
s[3][cursores[2]]="2 Bits";
cursores[2]=cursores[2]+1;
cursores[0]=cursores[0]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Longitud Total";
aux=p[cursores[0]]*256;
cursores[0]=cursores[0]+1;
aux=aux+p[cursores[0]];
cursores[0]=cursores[0]+1;
```

```
s[2][cursores[2]]=""+aux+" Octetos";
s[3][cursores[2]]="2 Octetos";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Identificación";
aux=p[cursores[0]]*256;
cursores[0]=cursores[0]+1;
aux=aux+p[cursores[0]];
cursores[0]=cursores[0]+1;
s[2][cursores[2]]=""+aux;
s[3][cursores[2]]="2 Octetos";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Indicador (Reservado)";
if((p[cursores[0]]>>7)==0){
        s[2][cursores[2]]=""+0;
}else{
        s[2][cursores[2]]=""+1;
}
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Indicador DF";
if(((p[cursores[0]]&127)>>6)==0){
        s[2][cursores[2]]="Puede fragmentarse";
}else{
        s[2][cursores[2]]="No Fragmentar";
}
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Indicador MF";
if(((p[cursores[0]]&63)>>5)==0){
        s[2][cursores[2]]="Último Fragmento";
}else{
        s[2][cursores[2]]="Más Fragmentos";
}
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Posición del Fragmento";
aux=(p[cursores[0]]&31)*256;
cursores[0]=cursores[0]+1;
aux=aux+p[cursores[0]];
cursores[0]=cursores[0]+1;
s[2][cursores[2]]=""+aux;
s[3][cursores[2]]="13 Bits";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Tiempo de Vida";
s[2][cursores[2]]=""+p[cursores[0]];
cursores[0]=cursores[0]+1;
s[3][cursores[2]]="1 Octeto";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Protocolo";
s[2][cursores[2]]=""+p[cursores[0]];
if(p[cursores[0]]==6){
        protocolos[4]="TCP";
}else if(p[cursores[0]]==17){
```

```
            protocolos[4]="UDP";
}
cursores[0]=cursores[0]+1;
s[3][cursores[2]]="1 Octeto";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Suma de Control de Cabecera";
aux=p[cursores[0]]*256;
cursores[0]=cursores[0]+1;
aux=aux+p[cursores[0]];
cursores[0]=cursores[0]+1;
s[2][cursores[2]]=""+aux;
s[3][cursores[2]]="2 Octetos";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Dirección IP Origen";
control[1]=""+p[cursores[0]]+"."+p[cursores[0]+1]+"."+p[cursores[0]+2]+"."
            +p[cursores[0]+3];
s[2][cursores[2]]=control[1];
cursores[0]=cursores[0]+4;
s[3][cursores[2]]="4 Octetos";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa3";
s[1][cursores[2]]="Dirección IP Destino";
control[2]=""+p[cursores[0]]+"."+p[cursores[0]+1]+"."+p[cursores[0]+2]+"."
            +p[cursores[0]+3];
s[2][cursores[2]]=control[2];
cursores[0]=cursores[0]+4;
s[3][cursores[2]]="4 Octetos";
cursores[2]=cursores[2]+1;

finopciones=false;
for(int i=0;i<(longitudcabecera-20);i++){
            if(p[cursores[0]]==0){
                        s[0][cursores[2]]="capa3";
                        s[1][cursores[2]]="Tipo-Opción";
                        if(finopciones){
                                    s[2][cursores[2]]="Relleno Opciones(No Copiar)";
                        }else{
                                    s[2][cursores[2]]="Fin de  Opciones(No Copiar)";
                                    finopciones=true;
                        }
                        s[3][cursores[2]]="1 Octeto";
                        cursores[0]=cursores[0]+1;
                        cursores[2]=cursores[2]+1;
            }else if(p[cursores[0]]==128){
                        s[0][cursores[2]]="capa3";
                        s[1][cursores[2]]="Tipo-Opción";
                        if(finopciones){
                                    s[2][cursores[2]]="Relleno Opciones(Copiar)";
                        }else{
                                    s[2][cursores[2]]="Fin de  Opciones(Copiar)";
                                    finopciones=true;
                        }
                        s[3][cursores[2]]="1 Octeto";
                        cursores[0]=cursores[0]+1;
                        cursores[2]=cursores[2]+1;
            }else if(p[cursores[0]]==1){
                        s[0][cursores[2]]="capa3";
                        s[1][cursores[2]]="Tipo-Opción";
                        s[2][cursores[2]]="Sin Operación(No Copiar)";
                        s[3][cursores[2]]="1 Octeto";
                        cursores[0]=cursores[0]+1;
                        cursores[2]=cursores[2]+1;
```

```
}else if(p[cursores[0]]==129){
		s[0][cursores[2]]="capa3";
		s[1][cursores[2]]="Tipo-Opción";
		s[2][cursores[2]]="Sin Operación(Copiar)";
		s[3][cursores[2]]="1 Octeto";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;
}else if(p[cursores[0]]==130){
		s[0][cursores[2]]="capa3";
		s[1][cursores[2]]="Tipo-Opción";
		s[2][cursores[2]]="Seguridad(Copiar)";
		s[3][cursores[2]]="1 Octeto";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;

		s[0][cursores[2]]="capa3";
		s[1][cursores[2]]="Longitud-Opción";
		s[2][cursores[2]]=""+p[cursores[0]];
		s[3][cursores[2]]="1 Octeto";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;
		i++;

		s[0][cursores[2]]="capa3";
		s[1][cursores[2]]="Nivel de Seguridad";
		aux=p[cursores[0]]*256;
		cursores[0]=cursores[0]+1;
		i++;
		aux=aux+p[cursores[0]];
		if(aux==0){
			s[2][cursores[2]]="No clasificado";
		}else if(aux==61749){
			s[2][cursores[2]]="Confidencial";
		}else if(aux==30874){
			s[2][cursores[2]]="EFTO";
		}else if(aux==48205){
			s[2][cursores[2]]="MMMM";
		}else if(aux==24102){
			s[2][cursores[2]]="PROG";
		}else if(aux==44819){
			s[2][cursores[2]]="Restringido";
		}else if(aux==55176){
			s[2][cursores[2]]="Secreto";
		}else if(aux==65705){
			s[2][cursores[2]]="Alto Secreto";
		}else{
			s[2][cursores[2]]="Reservado para uso futuro";
		}
		s[3][cursores[2]]="2 Octeto";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;
		i++;

		s[0][cursores[2]]="capa3";
		s[1][cursores[2]]="Compartimentos";
		aux=p[cursores[0]]*256;
		cursores[0]=cursores[0]+1;
		i++;
		aux=aux+p[cursores[0]];
		s[2][cursores[2]]=""+aux;
		s[3][cursores[2]]="2 Octetos";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;
		i++;

		s[0][cursores[2]]="capa3";
		s[1][cursores[2]]="Manejo de Restricciones";
```

```
                s[2][cursores[2]]=Utilidades.IntaHex(p[cursores[0]])+
                        Utilidades.IntaHex(p[cursores[0]+1]);
                s[3][cursores[2]]="2 Octetos";
                cursores[0]=cursores[0]+2;
                cursores[2]=cursores[2]+1;
                i=i+2;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Código de Control de Transmisión";
                s[2][cursores[2]]=Utilidades.IntaHex(p[cursores[0]])
                        +Utilidades.IntaHex(p[cursores[0]+1])
                        +Utilidades.IntaHex(p[cursores[0]+2]);
                s[3][cursores[2]]="3 Octetos";
                cursores[0]=cursores[0]+3;
                cursores[2]=cursores[2]+1;
                i=i+3;

        }else if(p[cursores[0]]==131){
                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Tipo-Opción";
                s[2][cursores[2]]="Ruta de Origen No Estricta(Copiar)";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Longitud-Opción";
                aux=p[cursores[0]];
                s[2][cursores[2]]=""+p[cursores[0]];
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
                i++;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Puntero";
                s[2][cursores[2]]=""+p[cursores[0]];
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
                i++;

                for(int j=0;j<(aux-3);j++){
                        s[0][cursores[2]]="capa3";
                        s[1][cursores[2]]="Dirección IP de Ruta";
                        s[2][cursores[2]]=""+p[cursores[0]]+"."+p[cursores[0]+1]+"."
                        +p[cursores[0]+2]+"."+p[cursores[0]+3];
                        cursores[0]=cursores[0]+4;
                        s[3][cursores[2]]="4 Octetos";
                        cursores[2]=cursores[2]+1;
                        i=i+4;
                }
        }else if(p[cursores[0]]==137){
                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Tipo-Opción";
                s[2][cursores[2]]="Ruta de Origen Estricta(Copiar)";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Longitud-Opción";
                aux=p[cursores[0]];
                s[2][cursores[2]]=""+p[cursores[0]];
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
```

```
                i++;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Puntero";
                s[2][cursores[2]]=""+p[cursores[0]];
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
                i++;

                for(int j=0;j<(aux-3);j++){
                        s[0][cursores[2]]="capa3";
                        s[1][cursores[2]]="Dirección IPde Ruta";
                        s[2][cursores[2]]=""+p[cursores[0]]+"."+p[cursores[0]+1]+"."
                        +p[cursores[0]+2]+"."+p[cursores[0]+3];
                        cursores[0]=cursores[0]+4;
                        s[3][cursores[2]]="4 Octetos";
                        cursores[2]=cursores[2]+1;
                        i=i+4;
                }
        }else if(p[cursores[0]]==7){
                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Tipo-Opción";
                s[2][cursores[2]]="Registrar Ruta(No Copiar)";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Longitud-Opción";
                aux=p[cursores[0]];
                s[2][cursores[2]]=""+p[cursores[0]];
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
                i++;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Puntero";
                s[2][cursores[2]]=""+p[cursores[0]];
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
                i++;

                for(int j=0;j<(aux-3);j++){
                        s[0][cursores[2]]="capa3";
                        s[1][cursores[2]]="Dirección IP de Ruta";
                        s[2][cursores[2]]=""+p[cursores[0]]+"."+p[cursores[0]+1]+"."
                        +p[cursores[0]+2]+"."+p[cursores[0]+3];
                        cursores[0]=cursores[0]+4;
                        s[3][cursores[2]]="4 Octetos";
                        cursores[2]=cursores[2]+1;
                        i=i+4;
                }
        }else if(p[cursores[0]]==136){
                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Tipo-Opción";
                s[2][cursores[2]]="Identificador de Flujo(Copiar)";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Longitud-Opción";
                aux=p[cursores[0]];
                s[2][cursores[2]]=""+p[cursores[0]];
```

```
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
                i++;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="ID de Flujo";
                s[2][cursores[2]]=Utilidades.IntaHex(p[cursores[0]])+
                        Utilidades.IntaHex(p[cursores[0]+1]);
                s[3][cursores[2]]="2 Octetos";
                cursores[0]=cursores[0]+2;
                cursores[2]=cursores[2]+1;
                i=i+2;
        }else if(p[cursores[0]]==68){
                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Tipo-Opción";
                s[2][cursores[2]]="Marca de tiempo(No Copiar)";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Longitud-Opción";
                aux=p[cursores[0]];
                s[2][cursores[2]]=""+p[cursores[0]];
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
                i++;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Puntero";
                s[2][cursores[2]]=""+p[cursores[0]];
                s[3][cursores[2]]="1 Octeto";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;
                i++;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Desbordamiento";
                s[2][cursores[2]]=""+(p[cursores[0]]>>4);
                s[3][cursores[2]]="4 Bits";
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa3";
                s[1][cursores[2]]="Indicador";
                aux2=p[cursores[0]]&15;
                if(aux2==0){
                        s[1][cursores[2]]="Sólo marcas de tiempo";
                }else if(aux2==1){
                        s[1][cursores[2]]="Marcas de tiempo y Direcciones IP";
                }else if(aux2==3){
                        s[1][cursores[2]]="Dirección IP preespecificados";
                }
                s[3][cursores[2]]="4 Bits";
                cursores[2]=cursores[2]+1;
                cursores[0]=cursores[0]+1;
                i++;

                if(aux2==1){
                        for(int j=0;j<((aux-4)/2);j++){
                                s[0][cursores[2]]="capa3";
                                s[1][cursores[2]]="Dirección IP";
                                s[2][cursores[2]]=""+p[cursores[0]]+"."+p[cursores[0]+1]+"."
                                +p[cursores[0]+2]+"."+p[cursores[0]+3];
                                cursores[0]=cursores[0]+4;
                                s[3][cursores[2]]="4 Octetos";
```

```
                                                cursores[2]=cursores[2]+1;
                                                i=i+4;

                                                s[0][cursores[2]]="capa3";
                                                s[1][cursores[2]]="Marca de tiempo";
                                                if((p[cursores[0]]>>7)==0){
                                                        s[2][cursores[2]]=""+(p[cursores[0]]*256*256*256
                                                        +p[cursores[0]+1]*256*256+p[cursores[0]]*256
                                                        +p[cursores[0]])+"ms";
                                                }else{
                                                        s[2][cursores[2]]=""+((p[cursores[0]]-
128)*256*256*256
                                                        +p[cursores[0]+1]*256*256+p[cursores[0]]*256
                                                        +p[cursores[0]]);
                                                }
                                                cursores[0]=cursores[0]+4;
                                                s[3][cursores[2]]="4 Octetos";
                                                cursores[2]=cursores[2]+1;
                                                i=i+4;
                                        }
                                }else if(aux2==0||aux2==3){
                                        s[0][cursores[2]]="capa3";
                                        s[1][cursores[2]]="Marca de tiempo";
                                        if((p[cursores[0]]>>7)==0){
                                                s[2][cursores[2]]=""+(p[cursores[0]]*256*256*256
                                                +p[cursores[0]+1]*256*256+p[cursores[0]]*256
                                                +p[cursores[0]])+"ms";
                                        }else{
                                                s[2][cursores[2]]=""+((p[cursores[0]]-128)*256*256*256
                                                +p[cursores[0]+1]*256*256+p[cursores[0]]*256
                                                +p[cursores[0]]);
                                        }
                                        cursores[0]=cursores[0]+4;
                                        s[3][cursores[2]]="4 Octetos";
                                        cursores[2]=cursores[2]+1;
                                        i=i+4;
                                }
                        }
                }

                if(protocolos[4]=="TCP"){
                        s[0][cursores[2]]="datos3";
                        s[1][cursores[2]]="Datos";
                        s[2][cursores[2]]="TCP";
                }else if(protocolos[4]=="UDP"){
                        s[0][cursores[2]]="datos3";
                        s[1][cursores[2]]="Datos";
                        s[2][cursores[2]]="UDP";
                }else{
                        s[0][cursores[2]]="desconocido3";
                        s[1][cursores[2]]="Datos";
                        s[2][cursores[2]]="Protocolo Desconocido";
                        s[0][0]="desconocido3";
                        s[1][0]="Datos";
                        s[2][0]="Protocolo Desconocido";
                        s[3][0]=""+(p.length-cursores[0])+" Octetos";
                }
                s[3][cursores[2]]=""+(p.length-cursores[0])+" Octetos";
                cursores[2]=cursores[2]+1;

        }
}
```

10. Clase DecodificadorQ_931

```
public class DecodificadorQ_931{
```

```
static int aux;
static long aux2;
static int aux3;
static int longitud;
static int longitud2;
static int contador;
static boolean ext;

public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){

		if(cursores[0]<p.length){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Discriminador de Protocolo";
			s[2][cursores[2]]=""+p[cursores[0]];
			s[3][cursores[2]]="1 Octeto";
			cursores[0]++;
			cursores[2]++;
		}

		if(cursores[0]<p.length){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Reservado";
			s[2][cursores[2]]=""+(p[cursores[0]]>>7)+((p[cursores[0]]>>6)&1)

+((p[cursores[0]]>>5)&1)+((p[cursores[0]]>>4)&1);
			s[3][cursores[2]]="4 Bits";
			cursores[2]++;

			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Longitud Referencia de Llamada";
			aux=p[cursores[0]]&15;
			s[2][cursores[2]]=""+aux+" Octetos";
			s[3][cursores[2]]="4 Bits";
			cursores[0]++;
			cursores[2]++;
		}

		if(cursores[0]<p.length){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Bandera";
			if(p[cursores[0]]>>7==0){
				s[2][cursores[2]]="Desde el Originador";
			}else{
				s[2][cursores[2]]="Hacia el Originador";
			}
			s[3][cursores[2]]="1 Bit";
			cursores[2]++;

			if(aux>0){
				s[0][cursores[2]]="capa6";
				s[1][cursores[2]]="Referencia de Llamada";
				aux2=p[cursores[0]]&127;
				cursores[0]++;
				for(int i=1;i<aux;i++){
					aux2=aux2*256+p[cursores[0]];
					cursores[0]++;
				}
				s[2][cursores[2]]=""+aux2;
				s[3][cursores[2]]=""+aux+" Octetos - 1 Bit";
				cursores[2]++;
			}
		}

		if(cursores[0]<p.length){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Reservado";
			s[2][cursores[2]]=""+(p[cursores[0]]>>7);
```

```
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Tipo de Mensaje";
		if(p[cursores[0]]==0){
			s[2][cursores[2]]="Escape";
		}else if(p[cursores[0]]==1){
			s[2][cursores[2]]="Aviso";
		}else if(p[cursores[0]]==2){
			s[2][cursores[2]]="Llamada en curso";
		}else if(p[cursores[0]]==7){
			s[2][cursores[2]]="Conexión";
		}else if(p[cursores[0]]==15){
			s[2][cursores[2]]="Acuse de Conexión";
		}else if(p[cursores[0]]==3){
			s[2][cursores[2]]="Progreso";
		}else if(p[cursores[0]]==5){
			s[2][cursores[2]]="Establecimiento";
		}else if(p[cursores[0]]==13){
			s[2][cursores[2]]="Acuse de Establecimiento";
		}else if(p[cursores[0]]==38){
			s[2][cursores[2]]="Reanudación";
		}else if(p[cursores[0]]==46){
			s[2][cursores[2]]="Acuse de Reanudación";
		}else if(p[cursores[0]]==34){
			s[2][cursores[2]]="Rechazo de Reanudación";
		}else if(p[cursores[0]]==37){
			s[2][cursores[2]]="Suspensión";
		}else if(p[cursores[0]]==45){
			s[2][cursores[2]]="Acuse de Suspensión";
		}else if(p[cursores[0]]==33){
			s[2][cursores[2]]="Rechazo de Suspensión";
		}else if(p[cursores[0]]==32){
			s[2][cursores[2]]="Información de Usuario";
		}else if(p[cursores[0]]==69){
			s[2][cursores[2]]="Desconexión";
		}else if(p[cursores[0]]==77){
			s[2][cursores[2]]="Liberación";
		}else if(p[cursores[0]]==90){
			s[2][cursores[2]]="Liberación Completa";
		}else if(p[cursores[0]]==70){
			s[2][cursores[2]]="Rearranque";
		}else if(p[cursores[0]]==78){
			s[2][cursores[2]]="Acuse de Rearranque";
		}else if(p[cursores[0]]==96){
			s[2][cursores[2]]="Segmento";
		}else if(p[cursores[0]]==121){
			s[2][cursores[2]]="Control de Congestión";
		}else if(p[cursores[0]]==123){
			s[2][cursores[2]]="Información";
		}else if(p[cursores[0]]==110){
			s[2][cursores[2]]="Notificación";
		}else if(p[cursores[0]]==125){
			s[2][cursores[2]]="Estado";
		}else if(p[cursores[0]]==117){
			s[2][cursores[2]]="Indagación de Estado";
		}else{
			s[2][cursores[2]]=""+p[cursores[0]];
		}
		s[3][cursores[2]]="7 Bits";
		cursores[0]++;
		cursores[2]++;
}

while(cursores[0]<p.length){
```

```
s[0][cursores[2]]="capa6";
s[1][cursores[2]]="Tipo Elemento de Información";
aux=p[cursores[0]]>>7;
if(aux==0){
		s[2][cursores[2]]="Longitud Variable";
}else{
		s[2][cursores[2]]="Un Octeto";
}
s[3][cursores[2]]="1 Bit";
cursores[2]++;

if(aux==1){
		if(((p[cursores[0]]>>4)&7)==2){
			if(p[cursores[0]]==160){
				s[0][cursores[2]]="capa6";
				s[1][cursores[2]]="Identificador Elemento de información";
				s[2][cursores[2]]="Más datos";
				s[3][cursores[2]]="7 Bits";
				cursores[0]++;
				cursores[2]++;
			}else if(p[cursores[0]]==161){
				s[0][cursores[2]]="capa6";
				s[1][cursores[2]]="Identificador Elemento de información";
				s[2][cursores[2]]="Envío completo";
				s[3][cursores[2]]="7 Bits";
				cursores[0]++;
				cursores[2]++;
			}
			else{
				s[0][cursores[2]]="capa6";
				s[1][cursores[2]]="Identificador Elemento de información";
				s[2][cursores[2]]="Reservado";
				s[3][cursores[2]]="7 Bits";
				cursores[0]++;
				cursores[2]++;
			}
		}else if(((p[cursores[0]]>>4)&7)==0){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Identificador Elemento de información";
			s[2][cursores[2]]="Reservado";
			s[3][cursores[2]]="7 Bits";
			cursores[0]++;
			cursores[2]++;
		}else if(((p[cursores[0]]>>4)&7)==1){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Identificador Elemento de información";
			s[2][cursores[2]]="Cambio";
			s[3][cursores[2]]="3 Bits";
			cursores[2]++;

			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Enclavamiento";
			if(((p[cursores[0]]>>3)&1)==0){
				s[2][cursores[2]]="Con Enclavamiento";
				s[3][cursores[2]]="1 Bit";
				cursores[2]++;

				s[0][cursores[2]]="capa6";
				s[1][cursores[2]]="Nuevo Conjunto de Códigos";
				aux3=p[cursores[0]]&7;
				if(aux3==0){
					s[2][cursores[2]]="0: Q.931";
				}else if(aux3==4){
					s[2][cursores[2]]="4: ISO/CEI";
				}else if(aux3==5){
					s[2][cursores[2]]="5: Uso Nacional";
```

```
}else if(aux3==6){
        s[2][cursores[2]]="6: Red Local";
}else if(aux3==7){
        s[2][cursores[2]]="7: Usuario";
}else{
        s[2][cursores[2]]="Reservado";
}
s[3][cursores[2]]="3 Bits";
cursores[0]++;
cursores[2]++;

s[0][cursores[2]]="capa6";
s[1][cursores[2]]="Otros Elementos de Información";
s[2][cursores[2]]="Desconocidos";
s[3][cursores[2]]=""+(p.length-cursores[0])+" Octetos";
cursores[0]=p.length;
cursores[2]++;
}else{
s[2][cursores[2]]="Sin Enclavamiento";
s[3][cursores[2]]="1 Bit";
cursores[2]++;

s[0][cursores[2]]="capa6";
s[1][cursores[2]]="Nuevo Conjunto de Códigos";
aux3=p[cursores[0]]&7;
if(aux3==0){
        s[2][cursores[2]]="0: Q.931";
}else if(aux3==4){
        s[2][cursores[2]]="4: ISO/CEI";
}else if(aux3==5){
        s[2][cursores[2]]="5: Uso Nacional";
}else if(aux3==6){
        s[2][cursores[2]]="6: Red Local";
}else if(aux3==7){
        s[2][cursores[2]]="7: Usuario";
}else{
        s[2][cursores[2]]="Reservado";
}
s[3][cursores[2]]="3 Bits";
cursores[0]++;
cursores[2]++;

s[0][cursores[2]]="capa6";
s[1][cursores[2]]="Tipo Elemento de Información";
if(p[cursores[0]]>>7==0){
        s[2][cursores[2]]="Longitud Variable";
        s[3][cursores[2]]="1 Bit";
        cursores[2]++;

        s[0][cursores[2]]="capa6";
        s[1][cursores[2]]="Identificador Elemento de
información";
        s[2][cursores[2]]=""+p[cursores[0]];
        s[3][cursores[2]]="7 Bits";
        cursores[0]++;
        cursores[2]++;

        if(cursores[0]<p.length){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Longitud Elemento
de información";
                longitud=p[cursores[0]];
                s[2][cursores[2]]=""+longitud+"
Octetos";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]++;
                cursores[2]++;
```

```
                                                }

                                                if(cursores[0]<p.length&&longitud>0){
                                                        s[0][cursores[2]]="capa6";
                                                        s[1][cursores[2]]="Contenido Elemento
de información";
                                                        s[2][cursores[2]]="Desconocido";
                                                        s[3][cursores[2]]=""+longitud+"
Octeto";
                                                        cursores[2]++;
                                                        cursores[0]=cursores[0]+longitud;
                                                }

                                        }else{
                                                s[2][cursores[2]]="Un Octeto";
                                                s[3][cursores[2]]="1 Bit";
                                                cursores[2]++;

                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Identificador Elemento de
Información";
                                                s[2][cursores[2]]=""+(p[cursores[0]]&127);
                                                s[3][cursores[2]]="7 Bits";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }

                                }

                        }else if(((p[cursores[0]]>>4)&7)==3){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Identificador Elemento de información";
                                s[2][cursores[2]]="Nivel de Congestión";
                                s[3][cursores[2]]="3 Bits";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Nivel de Congestión";
                                if((p[cursores[0]]&15)==0){
                                        s[2][cursores[2]]="Receptor Preparado";
                                }else if((p[cursores[0]]&15)==15){
                                        s[2][cursores[2]]="Receptor No Preparado";
                                }else{
                                        s[2][cursores[2]]="Reservado";
                                }
                                s[3][cursores[2]]="4 Bits";
                                cursores[0]++;
                                cursores[2]++;
                        }else if(((p[cursores[0]]>>4)&7)==5){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Identificador Elemento de información";
                                s[2][cursores[2]]="Indicador de repetición";
                                s[3][cursores[2]]="3 Bits";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Indicación de repetición";
                                if((p[cursores[0]]&15)==2){
                                        s[2][cursores[2]]="Lista de prioridad para seleccionar una
posibilidad";
                                }else{
                                        s[2][cursores[2]]="Reservado";
                                }
                                s[3][cursores[2]]="4 Bits";
                                cursores[0]++;
                                cursores[2]++;
                        }else{
```

```
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Elemento de información tipo 2";
                                        s[2][cursores[2]]=""+p[cursores[0]];
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[0]++;
                                        cursores[2]++;
                                }
                        }else{
                                if(p[cursores[0]]==4){
                                        capacidadportadora(p,s,cursores);
                                }else if(p[cursores[0]]==16){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Identificador Elemento de información";
                                        s[2][cursores[2]]="Identidad de llamada";
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[0]++;
                                        cursores[2]++;

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Longitud Elemento de información";
                                                longitud=p[cursores[0]];
                                                s[2][cursores[2]]=""+longitud+" Octetos";
                                                s[3][cursores[2]]="1 Octeto";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Identidad de llamada";
                                                s[2][cursores[2]]="";
                                                for(int i=0;i<longitud;i++){
                                                        if(cursores[0]<p.length){

s[2][cursores[2]]=s[2][cursores[2]]+p[cursores[0]]+",";
                                                                cursores[0]++;
                                                        }
                                                }
                                                s[3][cursores[2]]=""+longitud+" Octetos";
                                                cursores[2]++;
                                        }
                                }else if(p[cursores[0]]==20){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Identificador Elemento de información";
                                        s[2][cursores[2]]="Estado de la llamada";
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[0]++;
                                        cursores[2]++;

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Longitud Elemento de información";
                                                longitud=p[cursores[0]];
                                                s[2][cursores[2]]=""+longitud+" Octetos";
                                                s[3][cursores[2]]="1 Octeto";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Norma de codificación";
                                                aux3=(p[cursores[0]]>>5)&3;
                                                if(aux3==0){
                                                        s[2][cursores[2]]="UIT-T";
                                                }else if(aux3==1){
```

```
                                        s[2][cursores[2]]="ISO/CEI";
                                }else if(aux3==2){
                                        s[2][cursores[2]]="Norma nacional";
                                }else if(aux3==3){
                                        s[2][cursores[2]]="Norma definida para la red";
                                }
                                s[3][cursores[2]]="2 Bits";
                                cursores[2]++;
                        }

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Valor del estado de la llamada";
                                aux3=p[cursores[0]]&63;
                                if(aux3==0){
                                        s[2][cursores[2]]="Nulo";
                                }else if(aux3==1){
                                        s[2][cursores[2]]="Llamada iniciada";
                                }else if(aux3==2){
                                        s[2][cursores[2]]="Envío solapado";
                                }else if(aux3==3){
                                        s[2][cursores[2]]="Llamada saliente en curso";
                                }else if(aux3==4){
                                        s[2][cursores[2]]="Llamada entregada";
                                }else if(aux3==6){
                                        s[2][cursores[2]]="Llamada presente";
                                }else if(aux3==7){
                                        s[2][cursores[2]]="Llamada recibida";
                                }else if(aux3==8){
                                        s[2][cursores[2]]="Petición de conexión";
                                }else if(aux3==9){
                                        s[2][cursores[2]]="Llamada entrante en curso";
                                }else if(aux3==10){
                                        s[2][cursores[2]]="Activo";
                                }else if(aux3==11){
                                        s[2][cursores[2]]="Petición de desconexión";
                                }else if(aux3==12){
                                        s[2][cursores[2]]="Indicación de desconexión";
                                }else if(aux3==15){
                                        s[2][cursores[2]]="Petición de suspensión";
                                }else if(aux3==17){
                                        s[2][cursores[2]]="Petición de reanudación";
                                }else if(aux3==19){
                                        s[2][cursores[2]]="Petición de liberación";
                                }else if(aux3==25){
                                        s[2][cursores[2]]="Recepción solapada";
                                }else if(aux3==61){
                                        s[2][cursores[2]]="Petición de rearranque
(global)";
                                }else if(aux3==62){
                                        s[2][cursores[2]]="Rearranque (global)";
                                }else{
                                        s[2][cursores[2]]="Reservado";
                                }
                                s[3][cursores[2]]="6 Bits";
                                cursores[0]++;
                                cursores[2]++;
                        }
                }else if(p[cursores[0]]==112){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Identificador Elemento de información";
                        s[2][cursores[2]]="Número de la parte llamada";
                        s[3][cursores[2]]="7 Bits";
                        cursores[0]++;
                        cursores[2]++;

                        if(cursores[0]<p.length){
```

```
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Longitud Elemento de información";
                longitud=p[cursores[0]];
                s[2][cursores[2]]=""+longitud+" Octetos";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]++;
                cursores[2]++;
        }

        if(cursores[0]<p.length){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Bit de Extensión";
                ext=(p[cursores[0]]>>7)==0;
                if(ext){
                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                }else{
                        s[2][cursores[2]]="Último Octeto";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Tipo de número";
                aux3=(p[cursores[0]]>>4)&7;
                if(aux3==0){
                        s[2][cursores[2]]="Desconocido";
                }else if(aux3==1){
                        s[2][cursores[2]]="Número internacional";
                }else if(aux3==2){
                        s[2][cursores[2]]="Número nacional";
                }else if(aux3==3){
                        s[2][cursores[2]]="Número específico de red";
                }else if(aux3==4){
                        s[2][cursores[2]]="Número de abonado";
                }else if(aux3==6){
                        s[2][cursores[2]]="Número abreviado";
                }else if(aux3==7){
                        s[2][cursores[2]]="Reservado para ampliación";
                }else{
                        s[2][cursores[2]]="Reservado";
                }
                s[3][cursores[2]]="3 Bits";
                cursores[2]++;


                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Plan de numeración";
                if(aux3==0||aux3==2||aux3==4||aux3==8){
                        aux3=p[cursores[0]]&15;
                        if(aux3==0){
                                s[2][cursores[2]]="Desconocido";
                        }else if(aux3==1){
                                s[2][cursores[2]]="RDSI/telefonía
(Recomendación E.164 [19])";
                        }else if(aux3==3){
                                s[2][cursores[2]]="Datos

(Recomendación X.121 [21])";
                        }else if(aux3==4){
                                s[2][cursores[2]]="Télex

(Recomendación F.69 [22])";
                        }else if(aux3==8){
                                s[2][cursores[2]]="Normalización
nacional";
                        }else if(aux3==9){
                                s[2][cursores[2]]="Red privada";
                        }else if(aux3==15){
```

```
                                                s[2][cursores[2]]="Reservado para
ampliación";
                                        }else{
                                                s[2][cursores[2]]="Reservado";
                                        }
                                }else{
                                        s[2][cursores[2]]="Desconocido";
                                }
                                s[3][cursores[2]]="4 Bits";
                                cursores[2]++;
                                cursores[0]++;
                        }

                        for(int i=0;i<(longitud-1);i++){
                                if(cursores[0]<p.length){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Bit de Extensión";
                                        ext=(p[cursores[0]]>>7)==0;
                                        if(ext){
                                                s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                        }else{
                                                s[2][cursores[2]]="Último Octeto";
                                        }
                                        s[3][cursores[2]]="1 Bit";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Cifras del número";
                                        s[2][cursores[2]]=""+(char)p[cursores[0]];
                                        cursores[0]++;
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[2]++;
                                }
                        }
                }else if(p[cursores[0]]==113){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Identificador Elemento de información";
                        s[2][cursores[2]]="Subdirección de la parte llamada";
                        s[3][cursores[2]]="7 Bits";
                        cursores[0]++;
                        cursores[2]++;

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Longitud Elemento de información";
                                longitud=p[cursores[0]];
                                s[2][cursores[2]]=""+longitud+" Octetos";
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]++;
                                cursores[2]++;
                        }

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Bit de Extensión";
                                ext=(p[cursores[0]]>>7)==0;
                                if(ext){
                                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                }else{
                                        s[2][cursores[2]]="Último Octeto";
                                }
                                s[3][cursores[2]]="1 Bit";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
```

```
                                                s[1][cursores[2]]="Tipo de subdirección";
                                                aux3=(p[cursores[0]]>>4)&7;
                                                if(aux3==0){
                                                        s[2][cursores[2]]="NSAP (Rec. UIT-T X.213 [23]
e ISO/CEI 8348 Add.2 [24])";
                                                }else if(aux3==2){
                                                        s[2][cursores[2]]="Especificado por el usuario";
                                                }else{
                                                        s[2][cursores[2]]="Reservado";
                                                }
                                                s[3][cursores[2]]="3 Bits";
                                                cursores[2]++;

                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Indicador par/impar";
                                                aux3=(p[cursores[0]]>>3)&1;
                                                if(aux3==0){
                                                        s[2][cursores[2]]="Número par de señales de
dirección";
                                                }else{
                                                        s[2][cursores[2]]="Número impar de señales de
dirección";
                                                }
                                                s[3][cursores[2]]="1 Bit";
                                                cursores[2]++;

                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Reserva";

        s[2][cursores[2]]=""+((p[cursores[0]]>>2)&1)+((p[cursores[0]]>>1)&1
                                                        +(p[cursores[0]]&1));
                                                s[3][cursores[2]]="3 Bits";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }
                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Información de subdirección";
                                                s[2][cursores[2]]="";
                                                for(int i=0;i<(longitud-1);i++){
                                                        if(cursores[0]<p.length){

        s[2][cursores[2]]=s[2][cursores[2]]+p[cursores[0]]+",";
                                                                        cursores[0]++;
                                                        }
                                                }
                                                s[3][cursores[2]]=""+longitud+" Octetos";
                                                cursores[2]++;
                                        }
                                }else if(p[cursores[0]]==108){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Identificador Elemento de información";
                                        s[2][cursores[2]]="Número de la parte llamante";
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[0]++;
                                        cursores[2]++;

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Longitud Elemento de información";
                                                longitud=p[cursores[0]];
                                                s[2][cursores[2]]=""+longitud+" Octetos";
                                                s[3][cursores[2]]="1 Octeto";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }
```

```
if(cursores[0]<p.length){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto
siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Tipo de número";
		aux3=(p[cursores[0]]>>4)&7;
		if(aux3==0){
			s[2][cursores[2]]="Desconocido";
		}else if(aux3==1){
			s[2][cursores[2]]="Número internacional";
		}else if(aux3==2){
			s[2][cursores[2]]="Número nacional";
		}else if(aux3==3){
			s[2][cursores[2]]="Número específico de red";
		}else if(aux3==4){
			s[2][cursores[2]]="Número de abonado";
		}else if(aux3==6){
			s[2][cursores[2]]="Número abreviado";
		}else if(aux3==7){
			s[2][cursores[2]]="Reservado para ampliación";
		}else{
			s[2][cursores[2]]="Reservado";
		}
		s[3][cursores[2]]="3 Bits";
		cursores[2]++;


		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Plan de numeración";
		if(aux3==0||aux3==2||aux3==4||aux3==8){
			aux3=p[cursores[0]]&15;
			if(aux3==0){
				s[2][cursores[2]]="Desconocido";
			}else if(aux3==1){
				s[2][cursores[2]]="RDSI/telefonía
(Recomendación E.164 [19])";
			}else if(aux3==3){
				s[2][cursores[2]]="Datos
(Recomendación X.121 [21])";
			}else if(aux3==4){
				s[2][cursores[2]]="Télex
(Recomendación F.69 [22])";
			}else if(aux3==8){
				s[2][cursores[2]]="Normalización
nacional";
			}else if(aux3==9){
				s[2][cursores[2]]="Red privada";
			}else if(aux3==15){
				s[2][cursores[2]]="Reservado para
ampliación";
			}else{
				s[2][cursores[2]]="Reservado";
			}
		}else{
			s[2][cursores[2]]="Desconocido";
		}
		s[3][cursores[2]]="4 Bits";
```

```
                                        cursores[2]++;
                                        cursores[0]++;
                                }

                                if(cursores[0]<p.length){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Bit de Extensión";
                                        ext=(p[cursores[0]]>>7)==0;
                                        if(ext){
                                                s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                        }else{
                                                s[2][cursores[2]]="Último Octeto";
                                        }
                                        s[3][cursores[2]]="1 Bit";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Indicador de presentación";
                                        aux3=(p[cursores[0]]>>5)&3;
                                        if(aux3==0){
                                                s[2][cursores[2]]="Presentación permitida";
                                        }else if(aux3==1){
                                                s[2][cursores[2]]="Presentación restringida";
                                        }else if(aux3==2){
                                                s[2][cursores[2]]="Número no disponible debido al
interfuncionamiento";
                                        }else if(aux3==3){
                                                s[2][cursores[2]]="Reserva";
                                        }
                                        s[3][cursores[2]]="3 Bits";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Reserva";

        s[2][cursores[2]]=""+((p[cursores[0]]>>4)&1)+((p[cursores[0]]>>3)&1)+((p[cursores[0]]>>2)&1);
                                        s[3][cursores[2]]="3 Bits";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Indicador de selección";
                                        aux3=(p[cursores[0]]>>5)&3;
                                        if(aux3==0){
                                                s[2][cursores[2]]="Proporcionado por el usuario,
no verificado";
                                        }else if(aux3==1){
                                                s[2][cursores[2]]="Proporcionado por el usuario,
verificado y aprobado";
                                        }else if(aux3==2){
                                                s[2][cursores[2]]="Proporcionado por el usuario,
verificado y rechazado";
                                        }else if(aux3==3){
                                                s[2][cursores[2]]="Proporcionado por la red";
                                        }
                                        s[3][cursores[2]]="3 Bits";
                                        cursores[2]++;
                                        cursores[0]++;
                                }

                                for(int i=0;i<(longitud-2);i++){
                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Bit de Extensión";
                                                ext=(p[cursores[0]]>>7)==0;
                                                if(ext){
```

```
                                                s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                        }else{
                                                s[2][cursores[2]]="Último Octeto";
                                        }
                                        s[3][cursores[2]]="1 Bit";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Cifras del número";
                                        s[2][cursores[2]]=""+(char)p[cursores[0]];
                                        cursores[0]++;
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[2]++;
                                }
                        }


                }else if(p[cursores[0]]==109){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Identificador Elemento de información";
                        s[2][cursores[2]]="Subdirección de la parte llamante";
                        s[3][cursores[2]]="7 Bits";
                        cursores[0]++;
                        cursores[2]++;

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Longitud Elemento de información";
                                longitud=p[cursores[0]];
                                s[2][cursores[2]]=""+longitud+" Octetos";
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]++;
                                cursores[2]++;
                        }

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Bit de Extensión";
                                ext=(p[cursores[0]]>>7)==0;
                                if(ext){
                                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                }else{
                                        s[2][cursores[2]]="Último Octeto";
                                }
                                s[3][cursores[2]]="1 Bit";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Tipo de subdirección";
                                aux3=(p[cursores[0]]>>4)&7;
                                if(aux3==0){
                                        s[2][cursores[2]]="NSAP (Rec. UIT-T X.213 [23]
e ISO/CEI 8348 Add.2 [24])";
                                }else if(aux3==2){
                                        s[2][cursores[2]]="Especificado por el usuario";
                                }else{
                                        s[2][cursores[2]]="Reservado";
                                }
                                s[3][cursores[2]]="3 Bits";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Indicador par/impar";
                                aux3=(p[cursores[0]]>>3)&1;
                                if(aux3==0){
```

```
                                                s[2][cursores[2]]="Número par de señales de
dirección";
                                        }else{
                                                s[2][cursores[2]]="Número impar de señales de
dirección";
                                        }
                                        s[3][cursores[2]]="1 Bit";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Reserva";

        s[2][cursores[2]]=""+((p[cursores[0]]>>2)&1)+((p[cursores[0]]>>1)&1
                                                +(p[cursores[0]]&1));
                                        s[3][cursores[2]]="3 Bits";
                                        cursores[0]++;
                                        cursores[2]++;
                                }

                                if(cursores[0]<p.length){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Información de subdirección";
                                        s[2][cursores[2]]="";
                                        for(int i=0;i<(longitud-1);i++){
                                                if(cursores[0]<p.length){

        s[2][cursores[2]]=s[2][cursores[2]]+p[cursores[0]]+",";
                                                        cursores[0]++;
                                                }
                                        }
                                        s[3][cursores[2]]=""+longitud+" Octetos";
                                        cursores[2]++;
                                }
                        }else if(p[cursores[0]]==8){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Identificador Elemento de información";
                                s[2][cursores[2]]="Causa";
                                s[3][cursores[2]]="7 Bits";
                                cursores[0]++;
                                cursores[2]++;

                                if(cursores[0]<p.length){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Longitud Elemento de información";
                                        longitud=p[cursores[0]];
                                        s[2][cursores[2]]=""+longitud+" Octetos";
                                        s[3][cursores[2]]="1 Octeto";
                                        cursores[0]++;
                                        cursores[2]++;
                                }

                                contador=0;
                                if(cursores[0]<p.length&&contador<longitud){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Bit de Extensión";
                                        ext=(p[cursores[0]]>>7)==0;
                                        if(ext){
                                                s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                        }else{
                                                s[2][cursores[2]]="Último Octeto";
                                        }
                                        s[3][cursores[2]]="1 Bit";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Norma de codificación";
```

```
                                        aux3=(p[cursores[0]]>>5)&3;
                                        if(aux3==0){
                                                s[2][cursores[2]]="Codificación estandarizada
CCITT";
                                        }else if(aux3==1){
                                                s[2][cursores[2]]="ISO/CEI";
                                        }else if(aux3==2){
                                                s[2][cursores[2]]="Norma nacional";
                                        }else if(aux3==3){
                                                s[2][cursores[2]]="Norma específica de la
ubicación identificada";
                                        }
                                        s[3][cursores[2]]="2 Bits";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Reserva";
                                        s[2][cursores[2]]=""+((p[cursores[0]]>>4)&1);
                                        s[3][cursores[2]]="1 Bit";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Ubicación";
                                        aux3=p[cursores[0]]&16;
                                        if(aux3==0){
                                                s[2][cursores[2]]="Usuario";
                                        }else if(aux3==1){
                                                s[2][cursores[2]]="Red privada que da servicio al
usuario local";
                                        }else if(aux3==2){
                                                s[2][cursores[2]]="Red pública que da servicio al
usuario local";
                                        }else if(aux3==3){
                                                s[2][cursores[2]]="Red de tránsito";
                                        }else if(aux3==4){
                                                s[2][cursores[2]]="Red pública que da servicio al
usuario distante";
                                        }else if(aux3==5){
                                                s[2][cursores[2]]="Red privada que da servicio al
usuario distante";
                                        }else if(aux3==7){
                                                s[2][cursores[2]]="Red internacional";
                                        }else if(aux3==10){
                                                s[2][cursores[2]]="Red que se extiende más allá
del punto de interfuncionamiento";
                                        }else{
                                                s[2][cursores[2]]="Reservado";
                                        }
                                        s[3][cursores[2]]="4 Bits";
                                        cursores[2]++;
                                        cursores[0]++;
                                        contador++;
                }

                if(ext&&cursores[0]<p.length&&contador<longitud){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Bit de Extensión";
                                        ext=(p[cursores[0]]>>7)==0;
                                        if(ext){
                                                s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                        }else{
                                                s[2][cursores[2]]="Último Octeto";
                                        }
                                        s[3][cursores[2]]="1 Bit";
                                        cursores[2]++;
```

```
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Recomendación";
                                        aux3=p[cursores[0]]&127;
                                        if(aux3==0){
                                                s[2][cursores[2]]="Q.931";
                                        }else if(aux3==3){
                                                s[2][cursores[2]]="X.21";
                                        }else if(aux3==4){
                                                s[2][cursores[2]]="X.25";
                                        }else if(aux3==5){
                                                s[2][cursores[2]]="Redes públicas móviles
terrestres Q.1031/Q.1051";
                                        }else{
                                                s[2][cursores[2]]="Reservado";
                                        }
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[2]++;
                                        cursores[0]++;
                                        contador++;
                                }

                                if(cursores[0]<p.length&&contador<longitud){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Bit de Extensión";
                                        ext=(p[cursores[0]]>>7)==0;
                                        if(ext){
                                                s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                        }else{
                                                s[2][cursores[2]]="Último Octeto";
                                        }
                                        s[3][cursores[2]]="1 Bit";
                                        cursores[2]++;

                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Valor causa";
                                        aux3=p[cursores[0]]&127;
                                        if(aux3==1){
                                                s[2][cursores[2]]="Número no atribuido (no
asignado)";
                                        }else if(aux3==2){
                                                s[2][cursores[2]]="No hay ruta hacia la red de
tránsito especificada";
                                        }else if(aux3==3){
                                                s[2][cursores[2]]="No hay ruta hacia el destino";
                                        }else if(aux3==4){
                                                s[2][cursores[2]]="Enviar tono especial de
información";
                                        }else if(aux3==5){
                                                s[2][cursores[2]]="Prefijo interurbano marcado
erróneamente";
                                        }else if(aux3==6){
                                                s[2][cursores[2]]="Canal inaceptable";
                                        }else if(aux3==7){
                                                s[2][cursores[2]]="Llamada concedida y en curso
de conexión por un canal establecido";
                                        }else if(aux3==8){
                                                s[2][cursores[2]]="Precedencia";
                                        }else if(aux3==9){
                                                s[2][cursores[2]]="Precedencia - circuito reservado
para reutilización";
                                        }else if(aux3==16){
                                                s[2][cursores[2]]="Liberación normal de la
llamada";
                                        }else if(aux3==17){
                                                s[2][cursores[2]]="Usuario ocupado";
                                        }else if(aux3==18){
```

```
                s[2][cursores[2]]="No hay respuesta del usuario";
        }else if(aux3==19){
                s[2][cursores[2]]="No hay respuesta del usuario";
        }else if(aux3==20){
                s[2][cursores[2]]="Abonado ausente";
        }else if(aux3==21){
                s[2][cursores[2]]="Llamada rechazada";
        }else if(aux3==22){
                s[2][cursores[2]]="Número cambiado";
        }else if(aux3==23){
                s[2][cursores[2]]="Redireccionamiento a nuevo
destino";
        }else if(aux3==25){
                s[2][cursores[2]]="Error de encaminamiento de
central";
        }else if(aux3==26){
                s[2][cursores[2]]="Liberación por usuario no
seleccionado";
        }else if(aux3==27){
                s[2][cursores[2]]="Destino fuera de servicio";
        }else if(aux3==28){
                s[2][cursores[2]]="Formato de número no válido
(dirección incompleta)";
        }else if(aux3==29){
                s[2][cursores[2]]="Facilidad rechazada";
        }else if(aux3==30){
                s[2][cursores[2]]="Respuesta a INDAGACIÓN DE
ESTADO";
        }else if(aux3==31){
                s[2][cursores[2]]="Normal, no especificado";
        }else if(aux3==34){
                s[2][cursores[2]]="No hay circuito/canal
disponible";
        }else if(aux3==38){
                s[2][cursores[2]]="Red fuera de servicio";
        }else if(aux3==39){
                s[2][cursores[2]]="Conexión trama modo
permanente fuera de servicio";
        }else if(aux3==40){
                s[2][cursores[2]]="Conexión trama modo
permanente operacional";
        }else if(aux3==41){
                s[2][cursores[2]]="Fallo temporal";
        }else if(aux3==42){
                s[2][cursores[2]]="Congestión en el equipo de
conmutación";
        }else if(aux3==43){
                s[2][cursores[2]]="Información de acceso
descartada";
        }else if(aux3==44){
                s[2][cursores[2]]="Circuito/canal solicitado no
disponible";
        }else if(aux3==46){
                s[2][cursores[2]]="Llamada con precedencia
bloqueada";
        }else if(aux3==47){
                s[2][cursores[2]]="Recurso no disponible, no
especificado";
        }else if(aux3==49){
                s[2][cursores[2]]="Calidad de servicio no
disponible";
        }else if(aux3==50){
                s[2][cursores[2]]="Facilidad solicitada no
abonada";
        }else if(aux3==53){
                s[2][cursores[2]]="Prohibición de llamadas
salientes dentro de un grupo cerrado de usuarios";
```

```
entrantes dentro de un grupo cerrado de usuarios ";

autorizada";

disponible actualmente";

de acceso de salida y en la clase de abonado asignadas";

no especificado";

implementada";

implementado";

implementada";

capacidad portadora de información digital restringida";

implementado";

no válido";



pero no está suspendida la identidad de tal llamada";




posee la identidad de llamada solicitada";

grupo cerrado de usuarios";



inexistente";

válida";

especificado";

información obligatorio";

no implementado";
```

```
}else if(aux3==55){
        s[2][cursores[2]]="Prohibición de llamadas
}else if(aux3==57){
        s[2][cursores[2]]="Capacidad portadora no
}else if(aux3==58){
        s[2][cursores[2]]="Capacidad portadora no
}else if(aux3==62){
        s[2][cursores[2]]="Inchoherencia en la información
}else if(aux3==63){
        s[2][cursores[2]]="Servicio u opción no disponible,
}else if(aux3==65){
        s[2][cursores[2]]="Capacidad portadora no
}else if(aux3==66){
        s[2][cursores[2]]="Tipo de canal no
}else if(aux3==69){
        s[2][cursores[2]]="Facilidad solicitada no
}else if(aux3==70){
        s[2][cursores[2]]="Solamente está disponible la
}else if(aux3==79){
        s[2][cursores[2]]="Servicio u opción no
}else if(aux3==81){
        s[2][cursores[2]]="Valor de referencia de llamada
}else if(aux3==82){
        s[2][cursores[2]]="El canal identificado no existe";
}else if(aux3==83){
        s[2][cursores[2]]="Existe una llamada suspendida,
}else if(aux3==84){
        s[2][cursores[2]]="Identidad de llamada en uso";
}else if(aux3==85){
        s[2][cursores[2]]="Ninguna llamada suspendida";
}else if(aux3==86){
        s[2][cursores[2]]="Se ha liberado una llamada que
}else if(aux3==87){
        s[2][cursores[2]]="El usuario no es miembro del
}else if(aux3==88){
        s[2][cursores[2]]="Destino incompatible";
}else if(aux3==90){
        s[2][cursores[2]]="Grupo cerrado de usuarios
}else if(aux3==91){
        s[2][cursores[2]]="Selección de red de tránsito no
}else if(aux3==95){
        s[2][cursores[2]]="Mensaje no válido, no
}else if(aux3==96){
        s[2][cursores[2]]="Falta de elemento de
}else if(aux3==97){
        s[2][cursores[2]]="Tipo de mensaje inexistente o
}else if(aux3==98){
```

```
                                                s[2][cursores[2]]="Mensaje incompatible con el
estado de la llamada o tipo de mensaje inexistente o no implementado";
                                        }else if(aux3==99){
                                                s[2][cursores[2]]="Elemento/parámetro de
información inexistente o no implementado";
                                        }else if(aux3==100){
                                                s[2][cursores[2]]="Contenido de elemento de
información no válido";
                                        }else if(aux3==101){
                                                s[2][cursores[2]]="Mensaje incompatible con el
estado de la llamada";
                                        }else if(aux3==102){
                                                s[2][cursores[2]]="Recuperación tras la expiración
del plazo del temporizador";
                                        }else if(aux3==103){
                                                s[2][cursores[2]]="Parámetro inexistente o no
implementado, transferido";
                                        }else if(aux3==110){
                                                s[2][cursores[2]]="Mensaje con parámetro no
reconocido descartado";
                                        }else if(aux3==111){
                                                s[2][cursores[2]]="Error de protocolo, no
especificado";
                                        }else if(aux3==127){
                                                s[2][cursores[2]]="Interfuncionamiento, no
especificado";
                                        }else {
                                                s[2][cursores[2]]="Desconocido";
                                        }
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[2]++;
                                        cursores[0]++;
                                        contador++;
                                }

                                if(cursores[0]<p.length&&contador<longitud){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Diagnóstico";
                                        s[2][cursores[2]]="";
                                        while(cursores[0]<p.length&&contador<longitud){

        s[2][cursores[2]]=s[2][cursores[2]]+p[cursores[0]]+",";
                                                cursores[0]++;
                                                contador++;
                                        }
                                        s[3][cursores[2]]=""+longitud+" Octetos";
                                        cursores[2]++;
                                }
                        }else if(p[cursores[0]]==24){
                                identificaciondelcanal(p,s,cursores);
                        }else if(p[cursores[0]]==41){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Identificador Elemento de información";
                                s[2][cursores[2]]="Fecha/hora";
                                s[3][cursores[2]]="7 Bits";
                                cursores[0]++;
                                cursores[2]++;

                                if(cursores[0]<p.length){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Longitud Elemento de información";
                                        longitud=p[cursores[0]];
                                        s[2][cursores[2]]=""+longitud+" Octetos";
                                        s[3][cursores[2]]="1 Octeto";
                                        cursores[0]++;
                                        cursores[2]++;
                                }
```

```
		contador=0;
		if(cursores[0]<p.length&&contador<longitud){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Año";
			s[2][cursores[2]]=""+p[cursores[0]];
			s[3][cursores[2]]="1 Octeto";
			cursores[0]++;
			cursores[2]++;
			contador++;
		}
		if(cursores[0]<p.length&&contador<longitud){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Mes";
			s[2][cursores[2]]=""+p[cursores[0]];
			s[3][cursores[2]]="1 Octeto";
			cursores[0]++;
			cursores[2]++;
			contador++;
		}
		if(cursores[0]<p.length&&contador<longitud){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Día";
			s[2][cursores[2]]=""+p[cursores[0]];
			s[3][cursores[2]]="1 Octeto";
			cursores[0]++;
			cursores[2]++;
			contador++;
		}
		if(cursores[0]<p.length&&contador<longitud){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Hora";
			s[2][cursores[2]]=""+p[cursores[0]];
			s[3][cursores[2]]="1 Octeto";
			cursores[0]++;
			cursores[2]++;
			contador++;
		}
		if(cursores[0]<p.length&&contador<longitud){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Minuto";
			s[2][cursores[2]]=""+p[cursores[0]];
			s[3][cursores[2]]="1 Octeto";
			cursores[0]++;
			cursores[2]++;
			contador++;
		}
		if(cursores[0]<p.length&&contador<longitud){
			s[0][cursores[2]]="capa6";
			s[1][cursores[2]]="Segundo";
			s[2][cursores[2]]=""+p[cursores[0]];
			s[3][cursores[2]]="1 Octeto";
			cursores[0]++;
			cursores[2]++;
			contador++;
		}

	}else if(p[cursores[0]]==40){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Identificador Elemento de información";
		s[2][cursores[2]]="Visualización";
		s[3][cursores[2]]="7 Bits";
		cursores[0]++;
		cursores[2]++;

		if(cursores[0]<p.length){
			s[0][cursores[2]]="capa6";
```

```
                        s[1][cursores[2]]="Longitud Elemento de información";
                        longitud=p[cursores[0]];
                        s[2][cursores[2]]=""+longitud+" Octetos";
                        s[3][cursores[2]]="1 Octeto";
                        cursores[0]++;
                        cursores[2]++;
                }

                for(int i=0;i<longitud;i++){
                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Bit de Extensión";
                                ext=(p[cursores[0]]>>7)==0;
                                if(ext){
                                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                }else{
                                        s[2][cursores[2]]="Último Octeto";
                                }
                                s[3][cursores[2]]="1 Bit";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Información de visualización";
                                s[2][cursores[2]]=""+(char)p[cursores[0]];
                                cursores[0]++;
                                s[3][cursores[2]]="7 Bits";
                                cursores[2]++;
                        }
                }
        }else if(p[cursores[0]]==125){
                compatibilidaddecapaalta(p,s,cursores);
        }else if(p[cursores[0]]==44){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Identificador Elemento de información";
                s[2][cursores[2]]="Facilidad de teclado";
                s[3][cursores[2]]="7 Bits";
                cursores[0]++;
                cursores[2]++;

                if(cursores[0]<p.length){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Longitud Elemento de información";
                        longitud=p[cursores[0]];
                        s[2][cursores[2]]=""+longitud+" Octetos";
                        s[3][cursores[2]]="1 Octeto";
                        cursores[0]++;
                        cursores[2]++;
                }

                for(int i=0;i<longitud;i++){
                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Bit de Extensión";
                                ext=(p[cursores[0]]>>7)==0;
                                if(ext){
                                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                }else{
                                        s[2][cursores[2]]="Último Octeto";
                                }
                                s[3][cursores[2]]="1 Bit";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Información facilidad de
teclado";
```

```
                                        s[2][cursores[2]]=""+(char)p[cursores[0]];
                                        cursores[0]++;
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[2]++;
                                }
                        }
                }else if(p[cursores[0]]==124){
                        s[1][cursores[2]]="Identificador Elemento de información";
                        s[2][cursores[2]]="Compatibilidad de capa baja";
                        s[3][cursores[2]]="7 Bits";
                        cursores[0]++;
                        cursores[2]++;

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Longitud Elemento de información";
                                longitud=p[cursores[0]];
                                s[2][cursores[2]]=""+longitud+" Octetos";
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]++;
                                cursores[2]++;
                        }

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Contenido Elemento de información";
                                s[2][cursores[2]]="Desconocido";
                                s[3][cursores[2]]=""+longitud+" Octetos";
                                for(int i=0;i<longitud;i++){
                                        cursores[0]++;
                                }
                                cursores[2]++;
                        }

                        //Pendiente

                }else if(p[cursores[0]]==32){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Identificador Elemento de información";
                        s[2][cursores[2]]="Facilidades específicas de la red";
                        s[3][cursores[2]]="7 Bits";
                        cursores[0]++;
                        cursores[2]++;

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Longitud Elemento de información";
                                longitud=p[cursores[0]];
                                s[2][cursores[2]]=""+longitud+" Octetos";
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]++;
                                cursores[2]++;
                        }

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Longitud de la identificación de la red";
                                longitud2=p[cursores[0]];
                                s[2][cursores[2]]=""+longitud2+" Octetos";
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]++;
                                cursores[2]++;
                        }

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Bit de Extensión";
```

```
                    ext=(p[cursores[0]]>>7)==0;
                    if(ext){
                            s[2][cursores[2]]="Continúa en el octeto
siguiente";
                    }else{
                            s[2][cursores[2]]="Último Octeto";
                    }
                    s[3][cursores[2]]="1 Bit";
                    cursores[2]++;

                    s[0][cursores[2]]="capa6";
                    s[1][cursores[2]]="Identificación de tipo de  red ";
                    aux3=(p[cursores[0]]>>4)&7;
                    if(aux3==0){
                            s[2][cursores[2]]="Especificado por el usuario";
                    }else if(aux3==2){
                            s[2][cursores[2]]="Identificación de la red nacional
";
                    }else if(aux3==3){
                            s[2][cursores[2]]="Identificación de la red
internacional";
                    }else{
                            s[2][cursores[2]]="Reservado";
                    }
                    s[3][cursores[2]]="3 Bits";
                    cursores[2]++;

                    s[0][cursores[2]]="capa6";
                    s[1][cursores[2]]="Plan de identificación de la red";
                    aux3=p[cursores[0]]&15;
                    if(aux3==0){
                            s[2][cursores[2]]="Desconocido";
                    }else if(aux3==1){
                            s[2][cursores[2]]="De la compañía portadora ";
                    }else if(aux3==3){
                            s[2][cursores[2]]="De red de datos
(Recomendación X.121 [21])";
                    }else{
                            s[2][cursores[2]]="Reservado";
                    }
                    s[3][cursores[2]]="3 Bits";
                    cursores[2]++;
                    cursores[0]++;
            }

            for(int i=0;i<longitud2;i++){
                    if(cursores[0]<p.length){
                            s[0][cursores[2]]="capa6";
                            s[1][cursores[2]]="Bit de Extensión";
                            ext=(p[cursores[0]]>>7)==0;
                            if(ext){
                                    s[2][cursores[2]]="Continúa en el octeto
siguiente";
                            }else{
                                    s[2][cursores[2]]="Último Octeto";
                            }
                            s[3][cursores[2]]="1 Bit";
                            cursores[2]++;

                            s[0][cursores[2]]="capa6";
                            s[1][cursores[2]]="Identificación de la red";
                            s[2][cursores[2]]=""+(char)p[cursores[0]];
                            cursores[0]++;
                            s[3][cursores[2]]="7 Bits";
                            cursores[2]++;
                    }
            }
```

```
                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Especificación de facilidad específica de la
red";
                                s[2][cursores[2]]=""+p[cursores[0]];
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]++;
                                cursores[2]++;
                        }
                }else if(p[cursores[0]]==39){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Identificador Elemento de información";
                        s[2][cursores[2]]="Indicador de notificación";
                        s[3][cursores[2]]="7 Bits";
                        cursores[0]++;
                        cursores[2]++;

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Longitud Elemento de información";
                                longitud=p[cursores[0]];
                                s[2][cursores[2]]=""+longitud+" Octetos";
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]++;
                                cursores[2]++;
                        }

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Bit de Extensión";
                                ext=(p[cursores[0]]>>7)==0;
                                if(ext){
                                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                }else{
                                        s[2][cursores[2]]="Último Octeto";
                                }
                                s[3][cursores[2]]="1 Bit";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Descripción de la notificación";
                                if(p[cursores[0]]==0){
                                        s[2][cursores[2]]="Suspendida por el usuario";
                                }else if(p[cursores[0]]==1){
                                        s[2][cursores[2]]="Reanudada por el usuario";
                                }else if(p[cursores[0]]==2){
                                        s[2][cursores[2]]="Cambio de servicio portador";
                                }else {
                                        s[2][cursores[2]]="Reservado";
                                }
                                cursores[0]++;
                                s[3][cursores[2]]="7 Bits";
                                cursores[2]++;
                        }
                }else if(p[cursores[0]]==30){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Identificador Elemento de información";
                        s[2][cursores[2]]="Indicador de progreso";
                        s[3][cursores[2]]="7 Bits";
                        cursores[0]++;
                        cursores[2]++;

                        if(cursores[0]<p.length){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Longitud Elemento de información";
```

```
                longitud=p[cursores[0]];
                s[2][cursores[2]]=""+longitud+" Octetos";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]++;
                cursores[2]++;
        }

        if(cursores[0]<p.length){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Bit de Extensión";
                ext=(p[cursores[0]]>>7)==0;
                if(ext){
                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                }else{
                        s[2][cursores[2]]="Último Octeto";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Norma de codificación ";
                aux3=(p[cursores[0]]>>5)&3;
                if(aux3==0){
                        s[2][cursores[2]]="UIT-T";
                }else if(aux3==1){
                        s[2][cursores[2]]="ISO/CEI";
                }else if(aux3==2){
                        s[2][cursores[2]]="Norma nacional";
                }else if(aux3==3){
                        s[2][cursores[2]]="Norma específica de la
ubicación identificada";
                }
                s[3][cursores[2]]="2 Bits";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Reserva";
                s[2][cursores[2]]=""+((p[cursores[0]]>>4)&1);
                s[3][cursores[2]]="2 Bits";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Localización";
                aux3=p[cursores[0]]&15;
                if(aux3==0){
                        s[2][cursores[2]]="Usuario";
                }else if(aux3==1){
                        s[2][cursores[2]]="Red privada que sirve al usuario
local";
                }else if(aux3==2){
                        s[2][cursores[2]]="Red pública que sirve al usuario
local";
                }else if(aux3==3){
                        s[2][cursores[2]]="Red de tránsito";
                }else if(aux3==4){
                        s[2][cursores[2]]="Red pública que sirve al usuario
distante";
                }else if(aux3==5){
                        s[2][cursores[2]]="Red privada que sirve al usuario
distante";
                }else if(aux3==10){
                        s[2][cursores[2]]="Red más allá del punto de
interfuncionamiento";
                }else{
                        s[2][cursores[2]]="Reservado";
                }
```

```
                                                s[3][cursores[2]]="4 Bits";
                                                cursores[2]++;
                                                cursores[0]++;
                                        }

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Bit de Extensión";
                                                ext=(p[cursores[0]]>>7)==0;
                                                if(ext){
                                                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                                }else{
                                                        s[2][cursores[2]]="Último Octeto";
                                                }
                                                s[3][cursores[2]]="1 Bit";
                                                cursores[2]++;

                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Descripción de progreso";
                                                if(p[cursores[0]]==1){
                                                        s[2][cursores[2]]="La llamada no es RDSI de
extremo a extremo";
                                                }else if(p[cursores[0]]==2){
                                                        s[2][cursores[2]]="La dirección de destino no es
RDSI";
                                                }else if(p[cursores[0]]==3){
                                                        s[2][cursores[2]]="La dirección de origen no es
RDSI";
                                                }else if(p[cursores[0]]==4){
                                                        s[2][cursores[2]]="La llamada ha retornado a la
RDSI";
                                                }else if(p[cursores[0]]==5){
                                                        s[2][cursores[2]]="Se ha producido
interfuncionamiento y ha resultado en un cambio del servicio de telecomunicación";
                                                }else if(p[cursores[0]]==8){
                                                        s[2][cursores[2]]="Se encuentra disponible una
información o secuencia adecuada dentro de banda";
                                                }else {
                                                        s[2][cursores[2]]="Reservado";
                                                }
                                                cursores[0]++;
                                                s[3][cursores[2]]="7 Bits";
                                                cursores[2]++;
                                        }
                                }else if(p[cursores[0]]==121){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Identificador Elemento de información";
                                        s[2][cursores[2]]="Indicador de rearranque";
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[0]++;
                                        cursores[2]++;

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Longitud Elemento de información";
                                                longitud=p[cursores[0]];
                                                s[2][cursores[2]]=""+longitud+" Octetos";
                                                s[3][cursores[2]]="1 Octeto";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Bit de Extensión";
                                                ext=(p[cursores[0]]>>7)==0;
```

```
                                                if(ext){
                                                        s[2][cursores[2]]="Continúa en el octeto
siguiente";
                                                }else{
                                                        s[2][cursores[2]]="Último Octeto";
                                                }
                                                s[3][cursores[2]]="1 Bit";
                                                cursores[2]++;

                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Relleno";

        s[2][cursores[2]]=""+((p[cursores[0]]>>6)&1)+((p[cursores[0]]>>5)&1)

        +((p[cursores[0]]>>4)&1)+((p[cursores[0]]>>3)&1);
                                                s[3][cursores[2]]="4 Bits";
                                                cursores[2]++;

                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Norma de codificación ";
                                                aux3=p[cursores[0]]&7;
                                                if(aux3==0){
                                                        s[2][cursores[2]]="Canales indicados";
                                                }else if(aux3==6){
                                                        s[2][cursores[2]]="Una sola interfaz";
                                                }else if(aux3==7){
                                                        s[2][cursores[2]]="Todas las interfaces";
                                                }else{
                                                        s[2][cursores[2]]="Reservado";
                                                }
                                                s[3][cursores[2]]="3 Bits";
                                                cursores[2]++;
                                                cursores[0]++;
                                        }

                                        for(int i=0;i<(longitud-1);i++){
                                                cursores[0]++;
                                        }
                                }else if(p[cursores[0]]==-2){
                                        //Mensaje segmentado
                                }else if(p[cursores[0]]==52){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Identificador Elemento de información";
                                        s[2][cursores[2]]="Señal";
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[0]++;
                                        cursores[2]++;

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Longitud Elemento de información";
                                                longitud=p[cursores[0]];
                                                s[2][cursores[2]]=""+longitud+" Octetos";
                                                s[3][cursores[2]]="1 Octeto";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Valor de señal";
                                                if(p[cursores[0]]==0){
                                                        s[2][cursores[2]]="Tono de invitación a marcar
presente";
                                                }else if(p[cursores[0]]==1){
                                                        s[2][cursores[2]]="Tono de corriente de llamada
presente";
```

```
                                        }else if(p[cursores[0]]==2){
                                                s[2][cursores[2]]="Tono de intervención presente";
                                        }else if(p[cursores[0]]==3){
                                                s[2][cursores[2]]="Tono de congestión de red
presente";
                                        }else if(p[cursores[0]]==4){
                                                s[2][cursores[2]]="Tono de ocupado presente";
                                        }else if(p[cursores[0]]==5){
                                                s[2][cursores[2]]="Tono de confirmación
presente";
                                        }else if(p[cursores[0]]==6){
                                                s[2][cursores[2]]="Tono de respuesta presente";
                                        }else if(p[cursores[0]]==7){
                                                s[2][cursores[2]]="Tono de llamada en espera
presente";
                                        }else if(p[cursores[0]]==8){
                                                s[2][cursores[2]]="Tono de aviso de descolgado
presente";
                                        }else if(p[cursores[0]]==9){
                                                s[2][cursores[2]]="Tono de apropiación presente";
                                        }else if(p[cursores[0]]==63){
                                                s[2][cursores[2]]="Tonos ausentes";
                                        }else if(p[cursores[0]]==64){
                                                s[2][cursores[2]]="Aviso presente - secuencia 0";
                                        }else if(p[cursores[0]]==65){
                                                s[2][cursores[2]]="Aviso presente - secuencia 1";
                                        }else if(p[cursores[0]]==66){
                                                s[2][cursores[2]]="Aviso presente - secuencia 2";
                                        }else if(p[cursores[0]]==67){
                                                s[2][cursores[2]]="Aviso presente - secuencia 3";
                                        }else if(p[cursores[0]]==68){
                                                s[2][cursores[2]]="Aviso presente - secuencia 4";
                                        }else if(p[cursores[0]]==69){
                                                s[2][cursores[2]]="Aviso presente - secuencia 5";
                                        }else if(p[cursores[0]]==70){
                                                s[2][cursores[2]]="Aviso presente - secuencia 6";
                                        }else if(p[cursores[0]]==71){
                                                s[2][cursores[2]]="Aviso presente - secuencia 7";
                                        }else if(p[cursores[0]]==79){
                                                s[2][cursores[2]]="Aviso ausente";
                                        }else {
                                                s[2][cursores[2]]="Reservado";
                                        }
                                        cursores[0]++;
                                        s[3][cursores[2]]="1 Octeto";
                                        cursores[2]++;
                                }
                        }else if(p[cursores[0]]==-1){
                                //Selección de red de tránsito
                        }else if(p[cursores[0]]==126){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Identificador Elemento de información";
                                s[2][cursores[2]]="Usuario a usuario";
                                s[3][cursores[2]]="7 Bits";
                                cursores[0]++;
                                cursores[2]++;

                                if(cursores[0]+1<p.length){
                                        s[0][cursores[2]]="capa6";
                                        s[1][cursores[2]]="Longitud Elemento de información";
                                        longitud=256*p[cursores[0]]+p[cursores[0]+1];
                                        s[2][cursores[2]]=""+longitud+" Octetos";
                                        s[3][cursores[2]]="2 Octeto";
                                        cursores[0]=cursores[0]+2;
                                        cursores[2]++;
                                }
```

```
                                        if(cursores[0]<p.length&&longitud>0){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Discriminador de protocolo";
                                                if(p[cursores[0]]==0){
                                                        s[2][cursores[2]]="Específico de usuario";
                                                }else if(p[cursores[0]]==1){
                                                        s[2][cursores[2]]="Capa alta OSI";
                                                }else if(p[cursores[0]]==2){
                                                        s[2][cursores[2]]="Recomendación X.244";
                                                }else if(p[cursores[0]]==3){
                                                        s[2][cursores[2]]="Reservado para la función de
convergencia de gestión del sistema";
                                                }else if(p[cursores[0]]==4){
                                                        s[2][cursores[2]]="Caracteres Alfabeto
Internacional N.° 5";
                                                }else if(p[cursores[0]]==5){
                                                        s[2][cursores[2]]="X.208 y X.209 (ASN.1)";
                                                }else if(p[cursores[0]]==7){
                                                        s[2][cursores[2]]="Adaptación de velocidad de
acuerdo con la Recomendación V.120";
                                                }else if(p[cursores[0]]==8){
                                                        s[2][cursores[2]]="Mensajes de control de llamada
usuario-red de la Recomendación Q.931/I.451";
                                                }else if(p[cursores[0]]>=16&&p[cursores[0]]<=63){
                                                        s[2][cursores[2]]="Reservado para otros protocolos
de capa de red o de capa 3";
                                                }else if(p[cursores[0]]>=64&&p[cursores[0]]<=79){
                                                        s[2][cursores[2]]="Uso nacional";
                                                }else if(p[cursores[0]]>=80&&p[cursores[0]]<=254){
                                                        s[2][cursores[2]]="Reservado para otros protocolos
de capa de red o de capa 3";
                                                }else{
                                                        s[2][cursores[2]]="Reservado";
                                                }
                                                s[3][cursores[2]]="1 Octeto";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }

                                        if(cursores[0]<p.length&&longitud>1){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Información de usuario";
                                                s[2][cursores[2]]="Desconocido";
                                                s[3][cursores[2]]=""+(longitud-1)+" Octetos";
                                                cursores[0]=cursores[0]+longitud-1;
                                                cursores[2]++;
                                        }
                                }else if(p[cursores[0]]==-1){
                                        //Grupo cerrado de usuarios
                                }else if(p[cursores[0]]==-1){
                                        //Retardo de tránsito de extremo a extremo
                                }else if(p[cursores[0]]==-1){
                                        //Velocidad de información
                                }else if(p[cursores[0]]==-1){
                                        //Parámetros binarios de la capa de paquete
                                }else if(p[cursores[0]]==-1){
                                        //Tamaño de la ventana de la capa de paquete
                                }else if(p[cursores[0]]==-1){
                                        //Tamaño de paquete
                                }else if(p[cursores[0]]==-1){
                                        //Número redireccionante
                                }else if(p[cursores[0]]==-1){
                                        //Indicación de cobro revertido
                                }else if(p[cursores[0]]==-1){
                                        //Selección e indicación de retardo de tránsito
                                }else{
                                        s[0][cursores[2]]="capa6";
```

```
                                        s[1][cursores[2]]="Identificador Elemento de información";
                                        s[2][cursores[2]]=""+p[cursores[0]];
                                        s[3][cursores[2]]="7 Bits";
                                        cursores[0]++;
                                        cursores[2]++;

                                        if(cursores[0]<p.length){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Longitud Elemento de información";
                                                longitud=p[cursores[0]];
                                                s[2][cursores[2]]=""+longitud+" Octetos";
                                                s[3][cursores[2]]="1 Octeto";
                                                cursores[0]++;
                                                cursores[2]++;
                                        }

                                        if(cursores[0]<p.length&&longitud>0){
                                                s[0][cursores[2]]="capa6";
                                                s[1][cursores[2]]="Contenido Elemento de información";
                                                s[2][cursores[2]]="Desconocido";
                                                s[3][cursores[2]]=""+longitud+" Octeto";
                                                cursores[2]++;
                                                cursores[0]=cursores[0]+longitud;
                                        }
                                }

                        }
                }
}

public static void capacidadportadora(int p[],String s[][],int cursores[]){
        s[0][cursores[2]]="capa6";
        s[1][cursores[2]]="Identificador Elemento de información";
        s[2][cursores[2]]="Capacidad portadora";
        s[3][cursores[2]]="7 Bits";
        cursores[0]++;
        cursores[2]++;

        if(cursores[0]<p.length){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Longitud Elemento de información";
                longitud=p[cursores[0]];
                s[2][cursores[2]]=""+longitud+" Octetos";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]++;
                cursores[2]++;
        }

        contador=0;
        if(cursores[0]<p.length&&contador<longitud){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Bit de Extensión";
                if((p[cursores[0]]>>7)==1){
                        s[2][cursores[2]]="Último Octeto";
                }else{
                        s[2][cursores[2]]="Continúa en el octeto siguiente";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Norma de Codificación";
                aux3=(p[cursores[0]]>>5)&3;
                if(aux3==0){
                        s[2][cursores[2]]="UIT-T";
                }else if(aux3==1){
                        s[2][cursores[2]]="ISO/CEI";
```

```
		}else if(aux3==2){
			s[2][cursores[2]]="Norma nacional";
		}else if(aux3==3){
			s[2][cursores[2]]="Norma definida para la red";
		}
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Capacidad de transferencia de información";
		aux3=p[cursores[0]]&31;
		if(aux3==0){
			s[2][cursores[2]]="Conversación";
		}else if(aux3==8){
			s[2][cursores[2]]="Información digital sin restricciones";
		}else if(aux3==9){
			s[2][cursores[2]]="Información digital restringida";
		}else if(aux3==16){
			s[2][cursores[2]]="Audio de 3,1 kHz";
		}else if(aux3==17){
			s[2][cursores[2]]="Inf. Dig. sin Rest. con tonos/anuncios";
		}else if(aux3==24){
			s[2][cursores[2]]="Vídeo";
		}else{
			s[2][cursores[2]]="Reservado";
		}
		s[3][cursores[2]]="5 Bits";
		cursores[0]++;
		cursores[2]++;
		contador++;
}

if(cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Modo de transferencia";
		if(((p[cursores[0]]>>5)&3)==0){
			s[2][cursores[2]]="Modo circuito";
		}else if(((p[cursores[0]]>>5)&3)==2){
			s[2][cursores[2]]="Modo paquete";
		}else{
			s[2][cursores[2]]="Reservado";
		}
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Velocidad de transferencia de información";
		aux3=p[cursores[0]]&31;
		if(aux3==0){
			s[2][cursores[2]]="Modo paquete";
		}else if(aux3==16){
			s[2][cursores[2]]="64 kbit/s";
		}else if(aux3==17){
			s[2][cursores[2]]="2 x 64 kbit/s";
		}else if(aux3==19){
			s[2][cursores[2]]="384 kbit/s";
```

```
		}else if(aux3==21){
			s[2][cursores[2]]="1536 kbit/s";
		}else if(aux3==23){
			s[2][cursores[2]]="1920 kbit/s";
		}else if(aux3==24){
			s[2][cursores[2]]="Multivelocidad (velocidad básica de 64 kbit/s)";
		}else{
			s[2][cursores[2]]="Reservado";
		}
		s[3][cursores[2]]="5 Bits";
		cursores[0]++;
		cursores[2]++;
		contador++;
}

if(ext&&cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Multiplicador de velocidad";
		s[3][cursores[2]]=""+p[cursores[0]];
		s[3][cursores[2]]="7 Bits";
		cursores[0]++;
		cursores[2]++;
		contador++;
}

if(cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Identificador de capa 1";
		s[2][cursores[2]]=""+((p[cursores[0]]>>6)&1)+((p[cursores[0]]>>5)&1);
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Protocolo de capa 1 de información de usuario";
		aux3=p[cursores[0]]&31;
		if(aux3==1){
			s[2][cursores[2]]="UIT-T Recs. V.110, I.460 y X.30";
		}else if(aux3==2){
			s[2][cursores[2]]="G.711 [10] ley u";
		}else if(aux3==3){
			s[2][cursores[2]]="G.711 ley A";
		}else if(aux3==4){
			s[2][cursores[2]]="G.721 [11] MICDA a 32 kbit/s y Recomendación I.460";
		}else if(aux3==5){
			s[2][cursores[2]]="H.221 y H.242";
```

```
                        }else if(aux3==6){
                                s[2][cursores[2]]="H.223 [92] y H.245";
                        }else if(aux3==7){
                                s[2][cursores[2]]="Adaptación de velocidad no normalizada por el UIT-T";
                        }else if(aux3==8){
                                s[2][cursores[2]]="Adaptación de velocidad normalizada por el UIT-T V.120";
                        }else if(aux3==9){
                                s[2][cursores[2]]="Adaptación de velocidad normalizada por el UIT-T X.31 [14]
con relleno de banderas HDLC";
                        }else{
                                s[2][cursores[2]]="Reservado";
                        }
                        s[3][cursores[2]]="5 Bits";
                        cursores[0]++;
                        cursores[2]++;
                        contador++;
                }

                if(ext&&cursores[0]<p.length&&contador<longitud){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Bit de Extensión";
                        ext=(p[cursores[0]]>>7)==0;
                        if(ext){
                                s[2][cursores[2]]="Continúa en el octeto siguiente";
                        }else{
                                s[2][cursores[2]]="Último Octeto";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Síncrono/asíncrono";
                        if(((p[cursores[0]]>>6)&1)==0){
                                s[2][cursores[2]]="Síncrono";
                        }else{
                                s[2][cursores[2]]="Asíncrono";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Negociación";
                        if(((p[cursores[0]]>>6)&1)==0){
                                s[2][cursores[2]]="Es posible";
                        }else{
                                s[2][cursores[2]]="No es posible";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Velocidad de usuario";
                        aux3=p[cursores[0]]&31;
                        if(aux3==0){
                                s[2][cursores[2]]="0";
                        }else if(aux3==1){
                                s[2][cursores[2]]="0,6 kbit/s Recomendación X.1";
                        }else if(aux3==2){
                                s[2][cursores[2]]="1,2 kbit/s";
                        }else if(aux3==3){
                                s[2][cursores[2]]="2,4 kbit/s Recomendación X.1";
                        }else if(aux3==4){
                                s[2][cursores[2]]="3,6 kbit/s";
                        }else if(aux3==5){
                                s[2][cursores[2]]="4,8 kbit/s Recomendación X.1";
                        }else if(aux3==6){
                                s[2][cursores[2]]="7,2 kbit/s";
```

```
		}else if(aux3==7){
			s[2][cursores[2]]="8 kbit/s Recomendación I.460";
		}else if(aux3==8){
			s[2][cursores[2]]="9,6 kbit/s Recomendación X.1";
		}else if(aux3==9){
			s[2][cursores[2]]="14,4 kbit/s";
		}else if(aux3==10){
			s[2][cursores[2]]="16 kbit/s Recomendación I.460";
		}else if(aux3==11){
			s[2][cursores[2]]="19,2 kbit/s";
		}else if(aux3==12){
			s[2][cursores[2]]="32 kbit/s Recomendación I.460";
		}else if(aux3==13){
			s[2][cursores[2]]="38,4 kbit/s Recomendación V.110";
		}else if(aux3==14){
			s[2][cursores[2]]="48 kbit/s Recomendación X.1";
		}else if(aux3==15){
			s[2][cursores[2]]="56 kbit/s";
		}else if(aux3==18){
			s[2][cursores[2]]="57,6 kbit/s Recomendación V.14 ampliada";
		}else if(aux3==19){
			s[2][cursores[2]]="28,8 kbit/s Recomendación V.110";
		}else if(aux3==20){
			s[2][cursores[2]]="24 kbit/s Recomendación V.110";
		}else if(aux3==21){
			s[2][cursores[2]]="0,1345 kbit/s Recomendación X.1";
		}else if(aux3==22){
			s[2][cursores[2]]="0,100 kbit/s Recomendación X.1";
		}else if(aux3==23){
			s[2][cursores[2]]="0,075/1,2 kbit/s Recomendación X.1";
		}else if(aux3==24){
			s[2][cursores[2]]="1,2/0,075 kbit/s Recomendación X.1";
		}else if(aux3==25){
			s[2][cursores[2]]="0,050 kbit/s Recomendación X.1";
		}else if(aux3==26){
			s[2][cursores[2]]="0,075 kbit/s Recomendación X.1";
		}else if(aux3==27){
			s[2][cursores[2]]="0,110 kbit/s Recomendación X.1";
		}else if(aux3==28){
			s[2][cursores[2]]="0,150 kbit/s Recomendación X.1";
		}else if(aux3==29){
			s[2][cursores[2]]="0,200 kbit/s Recomendación X.1";
		}else if(aux3==30){
			s[2][cursores[2]]="0,300 kbit/s Recomendación X.1";
		}else if(aux3==31){
			s[2][cursores[2]]="12 kbit/s";
		}else{
			s[2][cursores[2]]="Reservado";
		}
		s[3][cursores[2]]="5 Bits";
		cursores[0]++;
		cursores[2]++;
		contador++;
	}

	if(ext&&cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;
```

```
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Velocidad intermedia";
                        aux3=(p[cursores[0]]>>5)&3;
                        if(aux3==0){
                                s[2][cursores[2]]="No se utiliza";
                        }else if(aux3==1){
                                s[2][cursores[2]]="8 kbit/s";
                        }else if(aux3==2){
                                s[2][cursores[2]]="16 kbit/s";
                        }else if(aux3==3){
                                s[2][cursores[2]]="32 kbit/s";
                        }
                        s[3][cursores[2]]="2 Bits";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Reloj independiente de la red Tx";
                        if(((p[cursores[0]]>>4)&1)==0){
                                s[2][cursores[2]]="No se requiere para enviar datos con reloj independiente de la
red";
                        }else{
                                s[2][cursores[2]]="Requerido para enviar datos con reloj independiente de la red";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Reloj independiente de la red Rx";
                        if(((p[cursores[0]]>>3)&1)==0){
                                s[2][cursores[2]]="No puede aceptar datos con reloj independiente de la red";
                        }else{
                                s[2][cursores[2]]="Puede aceptar datos con reloj independiente de la red";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Control de flujo en transmisión";
                        if(((p[cursores[0]]>>2)&1)==0){
                                s[2][cursores[2]]="No requerido para enviar datos con mecanismo de control de
flujo";
                        }else{
                                s[2][cursores[2]]="Requerido para enviar datos con mecanismo de control de flujo";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Control de flujo en recepción";
                        if(((p[cursores[0]]>>1)&1)==0){
                                s[2][cursores[2]]="No puede aceptar datos con mecanismo de control";
                        }else{
                                s[2][cursores[2]]="Puede aceptar datos con mecanismo de control de flujo";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Reserva";
                        s[2][cursores[2]]=""+(p[cursores[0]]&1);
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;
                        cursores[0]++;
                        contador++;

                }
```

```
if(ext&&cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Adaptación de velocidad con encabezamiento/sin encabezamiento";
		if(((p[cursores[0]]>>6)&1)==0){
			s[2][cursores[2]]="No incluido";
		}else{
			s[2][cursores[2]]="Incluido";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Soporte del establecimiento de multitrama en el enlace de datos";
		if(((p[cursores[0]]>>5)&1)==0){
			s[2][cursores[2]]="No se admite el establecimiento de multitrama";
		}else{
			s[2][cursores[2]]="Se admite establecimiento de multitrama";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Modo de operación";
		if(((p[cursores[0]]>>4)&1)==0){
			s[2][cursores[2]]="Transparente a los bits";
		}else{
			s[2][cursores[2]]="Sensible al protocolo";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Negociación de identificador del enlace lógico";
		if(((p[cursores[0]]>>3)&1)==0){
			s[2][cursores[2]]="Valor por defecto, LLI = 256 únicamente";
		}else{
			s[2][cursores[2]]="Protocolo de negociación completo";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Asignador/Asignado";
		if(((p[cursores[0]]>>2)&1)==0){
			s[2][cursores[2]]="El originador del mensaje es el asignado por defecto";
		}else{
			s[2][cursores[2]]="El originador del mensaje es el asignador solamente";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Negociación dentro/fuera de banda";
		if(((p[cursores[0]]>>1)&1)==0){
			s[2][cursores[2]]="Mediante mensajes INFORMACIÓN DE USUARIO a través de
una conexión temporal de señalización";
```

```
		}else{
			s[2][cursores[2]]="Dentro de banda utilizando el enlace lógico cero";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Reserva";
		s[2][cursores[2]]=""+(p[cursores[0]]&1);
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;
		cursores[0]++;
		contador++;
}

if(ext&&cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Número de bits de parada";
		aux3=(p[cursores[0]]>>5)&3;
		if(aux3==0){
			s[2][cursores[2]]="No se utiliza";
		}else if(aux3==1){
			s[2][cursores[2]]="1 bit";
		}else if(aux3==2){
			s[2][cursores[2]]="1,5 bits";
		}else if(aux3==3){
			s[2][cursores[2]]="2 bits";
		}
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Número de bits de datos, excluyendo, si está presente, el bit de paridad";
		aux3=(p[cursores[0]]>>3)&3;
		if(((p[cursores[0]]>>3)&3)==0){
			s[2][cursores[2]]="No se utiliza";
		}else if(aux3==1){
			s[2][cursores[2]]="5 bits";
		}else if(aux3==2){
			s[2][cursores[2]]="7 bits";
		}else if(aux3==3){
			s[2][cursores[2]]="8 bits";
		}
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Información de paridad";
		aux3=p[cursores[0]]&7;
		if(aux3==0){
			s[2][cursores[2]]="Impar";
		}else if(aux3==2){
			s[2][cursores[2]]="Par";
		}else if(aux3==3){
			s[2][cursores[2]]="Ninguno";
		}else if(aux3==4){
```

```
                    s[2][cursores[2]]="Forzado a 0";
                }else if(aux3==5){
                    s[2][cursores[2]]="Forzado a 1";
                }else{
                    s[2][cursores[2]]="Reservado";
                }
                s[3][cursores[2]]="2 Bits";
                cursores[2]++;
                cursores[0]++;
                contador++;
}

if(ext&&cursores[0]<p.length&&contador<longitud){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Bit de Extensión";
                ext=(p[cursores[0]]>>7)==0;
                if(ext){
                    s[2][cursores[2]]="Continúa en el octeto siguiente";
                }else{
                    s[2][cursores[2]]="Último Octeto";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Modo dúplex";
                if(((p[cursores[0]]>>6)&1)==0){
                    s[2][cursores[2]]="Semidúplex";
                }else{
                    s[2][cursores[2]]="Dúplex";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Tipo de módem";
                aux3=p[cursores[0]]&63;
                if(aux3<6||(aux3>31&&aux3<48)){
                    s[2][cursores[2]]="Uso nacional";
                }else if(aux3==17){
                    s[2][cursores[2]]="Recomendación V.21";
                }else if(aux3==18){
                    s[2][cursores[2]]="Recomendación V.22";
                }else if(aux3==19){
                    s[2][cursores[2]]="Recomendación V.22";
                }else if(aux3==20){
                    s[2][cursores[2]]="Recomendación V.23";
                }else if(aux3==21){
                    s[2][cursores[2]]="Recomendación V.26";
                }else if(aux3==22){
                    s[2][cursores[2]]="Recomendación V.26 bis";
                }else if(aux3==23){
                    s[2][cursores[2]]="Recomendación V.26 ter";
                }else if(aux3==24){
                    s[2][cursores[2]]="RecomendaciónV.27 ";
                }else if(aux3==25){
                    s[2][cursores[2]]="Recomendación V.27 bis";
                }else if(aux3==26){
                    s[2][cursores[2]]="Recomendación V.27 ter";
                }else if(aux3==27){
                    s[2][cursores[2]]="Recomendación V.29";
                }else if(aux3==29){
                    s[2][cursores[2]]="Recomendación V.32";
                }else if(aux3==30){
                    s[2][cursores[2]]="Recomendación V.34";
                }else if(aux3>47){
                    s[2][cursores[2]]="Especificado por el usuario";
```

```
		}else{
			s[2][cursores[2]]="Reservado";
		}
		s[3][cursores[2]]="6 Bits";
		cursores[2]++;
		cursores[0]++;
		contador++;
	}

	if(cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Identificador de capa 2";
		s[2][cursores[2]]=""+((p[cursores[0]]>>6)&1)+((p[cursores[0]]>>5)&1);
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Protocolo de capa 2 de información de usuario";
		aux3=p[cursores[0]]&31;
		if(aux3==2){
			s[2][cursores[2]]="Q.921/I.441";
		}else if(aux3==6){
			s[2][cursores[2]]="Recomendación X.25 [5], capa enlace";
		}else if(aux3==12){
			s[2][cursores[2]]="Control de canal lógico LAN (ISO/CEI 8802-2)";
		}else{
			s[2][cursores[2]]="Reservado";
		}
		s[3][cursores[2]]="5 Bits";
		cursores[0]++;
		cursores[2]++;
		contador++;
	}

	if(cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Identificador de capa 3";
		s[2][cursores[2]]=""+((p[cursores[0]]>>6)&1)+((p[cursores[0]]>>5)&1);
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Protocolo de capa 3 de información de usuario";
		aux3=p[cursores[0]]&31;
		if(aux3==2){
```

```
                        s[2][cursores[2]]="Q.931";
                }else if(aux3==6){
                        s[2][cursores[2]]="X.25, capa paquete";
                }else if(aux3==11){
                        s[2][cursores[2]]="ISO/CEI TR 9577 ";
                }else{
                        s[2][cursores[2]]="Reservado";
                }
                s[3][cursores[2]]="5 Bits";
                cursores[0]++;
                cursores[2]++;
                contador++;
        }

        if(cursores[0]<p.length&&contador<longitud){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Bit de Extensión";
                ext=(p[cursores[0]]>>7)==0;
                if(ext){
                        s[2][cursores[2]]="Continúa en el octeto siguiente";
                }else{
                        s[2][cursores[2]]="Último Octeto";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Reserva";

s[2][cursores[2]]=""+((p[cursores[0]]>>6)&1)+((p[cursores[0]]>>5)&1)+((p[cursores[0]]>>4)&1);
                s[3][cursores[2]]="3 Bits";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Información adicional capa 3";
                if((p[cursores[0]]&15)==12){
                        s[2][cursores[2]]="Protocolo Internet (RFC 791) (ISO/CEI TR 9577 [82])";
                }else if((p[cursores[0]]&15)==15){
                        s[2][cursores[2]]="Protocolo punto a punto (RFC 1548)";
                }else{
                        s[2][cursores[2]]="Reservado";
                }
                s[3][cursores[2]]="5 Bits";
                cursores[0]++;
                cursores[2]++;
                contador++;
        }

        if(cursores[0]<p.length&&contador<longitud){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Bit de Extensión";
                ext=(p[cursores[0]]>>7)==0;
                if(ext){
                        s[2][cursores[2]]="Continúa en el octeto siguiente";
                }else{
                        s[2][cursores[2]]="Último Octeto";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Reserva";

s[2][cursores[2]]=""+((p[cursores[0]]>>6)&1)+((p[cursores[0]]>>5)&1)+((p[cursores[0]]>>4)&1);
                s[3][cursores[2]]="3 Bits";
                cursores[2]++;
```

```
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Información adicional capa 3";
                        if((p[cursores[0]]&15)==12){
                                s[2][cursores[2]]="Protocolo Internet (RFC 791) (ISO/CEI TR 9577 [82])";
                        }else if((p[cursores[0]]&15)==15){
                                s[2][cursores[2]]="Protocolo punto a punto (RFC 1548)";
                        }else{
                                s[2][cursores[2]]="Reservado";
                        }
                        s[3][cursores[2]]="5 Bits";
                        cursores[0]++;
                        cursores[2]++;
                        contador++;
                }

                for(int i=contador;i<longitud;i++){
                        cursores[0]++;
                        System.out.println("Completando");
                }
        }

        public static void identificaciondelcanal(int p[],String s[][],int cursores[]){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Identificador Elemento de información";
                s[2][cursores[2]]="Capacidad portadora";
                s[3][cursores[2]]="7 Bits";
                cursores[0]++;
                cursores[2]++;

                if(cursores[0]<p.length){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Longitud Elemento de información";
                        longitud=p[cursores[0]];
                        s[2][cursores[2]]=""+longitud+" Octetos";
                        s[3][cursores[2]]="1 Octeto";
                        cursores[0]++;
                        cursores[2]++;
                }

                contador=0;
                if(cursores[0]<p.length&&contador<longitud){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Bit de Extensión";
                        ext=(p[cursores[0]]>>7)==0;
                        if(ext){
                                s[2][cursores[2]]="Continúa en el octeto siguiente";
                        }else{
                                s[2][cursores[2]]="Último Octeto";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Identificador de interfaz presente";
                        if(((p[cursores[0]]>>6)&1)==0){
                                s[2][cursores[2]]="Interfaz identificada implícitamente";
                        }else{
                                s[2][cursores[2]]="Interfaz identificada explícitamente en uno o más octetos que
comienzan con el octeto 3.1";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Tipo de interfaz";
                        if(((p[cursores[0]]>>5)&1)==0){
                                s[2][cursores[2]]="Interfaz básica";
```

```
		}else{
			s[2][cursores[2]]="Otra interfaz";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Reserva";
		s[2][cursores[2]]=""+((p[cursores[0]]>>4)&1);
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Preferido/exclusivo";
		if(((p[cursores[0]]>>3)&1)==0){
			s[2][cursores[2]]="El canal indicado es preferido";
		}else{
			s[2][cursores[2]]="Exclusivo; sólo el canal indicado es aceptable";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Indicador de canal D";
		if(((p[cursores[0]]>>2)&1)==0){
			s[2][cursores[2]]="El canal identificado no es el canal D";
		}else{
			s[2][cursores[2]]="El canal identificado es el canal D";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Selección de canal de información";
		aux3=p[cursores[0]]&3;
		if(aux3==0){
			if(((p[cursores[0]]>>5)&1)==0){
				s[2][cursores[2]]="Ningún canal";
			}else{
				s[2][cursores[2]]="Ningún canal";
			}
		}else if(aux3==1){
			if(((p[cursores[0]]>>5)&1)==0){
				s[2][cursores[2]]="Canal B1";
			}else{
				s[2][cursores[2]]="Como se indica en los octetos siguientes";
			}
		}else if(aux3==2){
			if(((p[cursores[0]]>>5)&1)==0){
				s[2][cursores[2]]="Canal B2";
			}else{
				s[2][cursores[2]]="Reservado";
			}
		}else if(aux3==3){
			if(((p[cursores[0]]>>5)&1)==0){
				s[2][cursores[2]]="Cualquier canal";
			}else{
				s[2][cursores[2]]="Cualquier canal";
			}
		}
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;
		cursores[0]++;
		contador++;
}

while(cursores[0]<p.length&&ext&&contador<longitud){
```

```
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Identificador de interfaz";
		s[2][cursores[2]]=""+p[cursores[0]];
		cursores[0]++;
		s[3][cursores[2]]="7 Bits";
		cursores[2]++;
		contador++;
}

if(cursores[0]<p.length&&contador<longitud){
		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Bit de Extensión";
		ext=(p[cursores[0]]>>7)==0;
		if(ext){
			s[2][cursores[2]]="Continúa en el octeto siguiente";
		}else{
			s[2][cursores[2]]="Último Octeto";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Norma de codificación";
		aux3=(p[cursores[0]]>>5)&3;
		if(aux3==0){
			s[2][cursores[2]]="UIT-T";
		}else if(aux3==1){
			s[2][cursores[2]]="ISO/CEI";
		}else if(aux3==2){
			s[2][cursores[2]]="Norma nacional";
		}else if(aux3==3){
			s[2][cursores[2]]="Norma definida para la red";
		}
		s[3][cursores[2]]="2 Bits";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Número/mapa";
		if(((p[cursores[0]]>>4)&1)==0){
			s[2][cursores[2]]="Número en el siguiente octeto";
		}else{
			s[2][cursores[2]]="Mapa de intervalo en el siguiente o siguientes octetos";
		}
		s[3][cursores[2]]="1 Bit";
		cursores[2]++;

		s[0][cursores[2]]="capa6";
		s[1][cursores[2]]="Tipo de canal/tipo de elemento de mapa";
		aux3=p[cursores[0]]&15;
		if(aux3==3){
			s[2][cursores[2]]="Unidades de canal B";
		}else if(aux3==6){
			s[2][cursores[2]]="Unidades de canal H0";
		}else if(aux3==8){
			s[2][cursores[2]]="Unidades de canal H11";
		}else if(aux3==9){
```

```
                        s[2][cursores[2]]="Unidades de canal H12";
                }else{
                        s[2][cursores[2]]="Reservado";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;
                cursores[0]++;
                contador++;
        }

        while(cursores[0]<p.length&&contador<longitud){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Número de canal/Mapa de intervalos";
                s[2][cursores[2]]=""+(p[cursores[0]]>>7)+((p[cursores[0]]>>6)&1)
                        +((p[cursores[0]]>>5)&1)+((p[cursores[0]]>>4)&1)
                        +((p[cursores[0]]>>3)&1)+((p[cursores[0]]>>2)&1)
                        +((p[cursores[0]]>>1)&1)+(p[cursores[0]]&1);
                s[3][cursores[2]]="1 Octeto";
                cursores[2]++;
                cursores[0]++;
                contador++;
        }
}

public static void compatibilidaddecapaalta(int p[],String s[][],int cursores[]){
        s[0][cursores[2]]="capa6";
        s[1][cursores[2]]="Identificador Elemento de información";
        s[2][cursores[2]]="Compatibilidad de capa alta";
        s[3][cursores[2]]="7 Bits";
        cursores[0]++;
        cursores[2]++;

        if(cursores[0]<p.length){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Longitud Elemento de información";
                longitud=p[cursores[0]];
                s[2][cursores[2]]=""+longitud+" Octetos";
                s[3][cursores[2]]="1 Octeto";
                cursores[0]++;
                cursores[2]++;
        }

        contador=0;
        if(cursores[0]<p.length&&contador<longitud){
                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Bit de Extensión";
                if((p[cursores[0]]>>7)==1){
                        s[2][cursores[2]]="Último Octeto";
                }else{
                        s[2][cursores[2]]="Continúa en el octeto siguiente";
                }
                s[3][cursores[2]]="1 Bit";
                cursores[2]++;

                s[0][cursores[2]]="capa6";
                s[1][cursores[2]]="Norma de Codificación";
                aux3=(p[cursores[0]]>>5)&3;
                if(aux3==0){
                        s[2][cursores[2]]="UIT-T";
                }else if(aux3==1){
                        s[2][cursores[2]]="ISO/CEI";
                }else if(aux3==2){
                        s[2][cursores[2]]="Norma nacional";
                }else if(aux3==3){
                        s[2][cursores[2]]="Norma definida para la red";
                }
                s[3][cursores[2]]="2 Bits";
```

```
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Interpretación";
                                aux3=(p[cursores[0]]>>2)&7;
                                if(aux3==4){
                                        s[2][cursores[2]]="Primera (o única) identificación de las características de capa
alta";
                                }else{
                                        s[2][cursores[2]]="Reservado";
                                }
                                s[3][cursores[2]]="3 Bits";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Método de presentación del perfil de protocolo";
                                aux3=(p[cursores[0]]>>2)&7;
                                if(aux3==4){
                                        s[2][cursores[2]]="Perfil de protocolo de capa alta";
                                }else{
                                        s[2][cursores[2]]="Reservado";
                                }
                                s[3][cursores[2]]="3 Bits";
                                cursores[2]++;
                                cursores[0]++;
                                contador++;
                        }

                        if(cursores[0]<p.length&&contador<longitud){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Bit de Extensión";
                                if((p[cursores[0]]>>7)==1){
                                        s[2][cursores[2]]="Último Octeto";
                                }else{
                                        s[2][cursores[2]]="Continúa en el octeto siguiente";
                                }
                                s[3][cursores[2]]="1 Bit";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Identificación de las características de capa alta ";
                                aux3=p[cursores[0]]&127;
                                if(aux3==1){
                                        s[2][cursores[2]]="Telefonía";
                                }else if(aux3==4){
                                        s[2][cursores[2]]="Facsímil del grupo 2/3 (Recomendación F.182 [68])";
                                }else if(aux3==33){
                                        s[2][cursores[2]]="Facsímil del grupo 4 clase I (Recomendación F.184 [69])";
                                }else if(aux3==36){
                                        s[2][cursores[2]]="Servicio facsímil grupo 4, clases II y III (Recomendación
F.184)";
                                }else if(aux3==40){
                                        s[2][cursores[2]]="Se incluye punto de código sólo para los servicios sobre los que
existen Recomendaciones UIT-T";
                                }else if(aux3==49){
                                        s[2][cursores[2]]="Se incluye punto de código sólo para los servicios sobre los que
existen Recomendaciones UIT-T";
                                }else if(aux3==50){
                                        s[2][cursores[2]]="Videotex basado en sintaxis";
                                }else if(aux3==51){
                                        s[2][cursores[2]]="Interfuncionamiento videotex internacional entre centrales
pasarelas o unidades de interfuncionamiento";
                                }else if(aux3==53){
                                        s[2][cursores[2]]="Servicio télex (Recomendación F.60 [76])";
                                }else if(aux3==56){
                                        s[2][cursores[2]]="Sistemas de tratamiento de mensajes (MHS, message handling
systems)";
```

```
                        }else if(aux3==65){
                                s[2][cursores[2]]="Aplicación OSI ";
                        }else if(aux3==66){
                                s[2][cursores[2]]="Aplicación FTAM (ISO 8571)";
                        }else if(aux3==94){
                                s[2][cursores[2]]="Reservado para mantenimiento";
                        }else if(aux3==95){
                                s[2][cursores[2]]="Reservado para gestión";
                        }else if(aux3==96){
                                s[2][cursores[2]]="Videotelefonía";
                        }else if(aux3==97){
                                s[2][cursores[2]]="Videoconferencia Recomendaciones F.702 [94] y F.731 [97]
perfil 1b ";
                        }else if(aux3==98){
                                s[2][cursores[2]]="Conferencia audiográfica";
                        }else if(aux3>99&&aux3<103){
                                s[2][cursores[2]]="Reservados para los servicios audiovisuales";
                        }else if(aux3==104){
                                s[2][cursores[2]]="Servicios multimedios";
                        }else if(aux3>105&&aux3<111){
                                s[2][cursores[2]]="Reservados para los servicios audiovisuales";
                        }else if(aux3==127){
                                s[2][cursores[2]]="Reservado";
                        }else{
                                s[2][cursores[2]]="Reservado";
                        }
                        s[3][cursores[2]]="7 Bits";
                        cursores[2]++;
                        cursores[0]++;
                        contador++;
                }

                if(ext&&cursores[0]<p.length&&contador<longitud&&(aux3==94||aux3==95)){
                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Bit de Extensión";
                        if((p[cursores[0]]>>7)==1){
                                s[2][cursores[2]]="Último Octeto";
                        }else{
                                s[2][cursores[2]]="Continúa en el octeto siguiente";
                        }
                        s[3][cursores[2]]="1 Bit";
                        cursores[2]++;

                        s[0][cursores[2]]="capa6";
                        s[1][cursores[2]]="Para mantenimiento o gestión";
                        aux3=p[cursores[0]]&127;
                        if(aux3==1){
                                s[2][cursores[2]]="Telefonía";
                        }else if(aux3==4){
                                s[2][cursores[2]]="Facsímil del grupo 2/3 (Recomendación F.182 [68])";
                        }else if(aux3==33){
                                s[2][cursores[2]]="Facsímil del grupo 4 clase I (Recomendación F.184 [69])";
                        }else if(aux3==36){
                                s[2][cursores[2]]="Servicio facsímil grupo 4, clases II y III (Recomendación
F.184)";
                        }else if(aux3==40){
                                s[2][cursores[2]]="Se incluye punto de código sólo para los servicios sobre los que
existen Recomendaciones UIT-T";
                        }else if(aux3==49){
                                s[2][cursores[2]]="Se incluye punto de código sólo para los servicios sobre los que
existen Recomendaciones UIT-T";
                        }else if(aux3==50){
                                s[2][cursores[2]]="Videotex basado en sintaxis";
                        }else if(aux3==51){
                                s[2][cursores[2]]="Interfuncionamiento videotex internacional entre centrales
pasarelas o unidades de interfuncionamiento";
                        }else if(aux3==53){
```

```
                                        s[2][cursores[2]]="Servicio télex (Recomendación F.60 [76])";
                                }else if(aux3==56){
                                        s[2][cursores[2]]="Sistemas de tratamiento de mensajes (MHS, message handling
systems)";
                                }else if(aux3==65){
                                        s[2][cursores[2]]="Aplicación OSI ";
                                }else if(aux3==66){
                                        s[2][cursores[2]]="Aplicación FTAM (ISO 8571)";
                                }else if(aux3==94){
                                        s[2][cursores[2]]="No disponible para asignación";
                                }else if(aux3==95){
                                        s[2][cursores[2]]="No disponible para asignación";
                                }else if(aux3==96){
                                        s[2][cursores[2]]="Videotelefonía";
                                }else if(aux3==97){
                                        s[2][cursores[2]]="Videoconferencia Recomendaciones F.702 [94] y F.731 [97]
perfil 1b ";
                                }else if(aux3==98){
                                        s[2][cursores[2]]="Conferencia audiográfica";
                                }else if(aux3>99&&aux3<103){
                                        s[2][cursores[2]]="Reservados para los servicios audiovisuales";
                                }else if(aux3==104){
                                        s[2][cursores[2]]="Servicios multimedios";
                                }else if(aux3>105&&aux3<111){
                                        s[2][cursores[2]]="Reservados para los servicios audiovisuales";
                                }else if(aux3==127){
                                        s[2][cursores[2]]="Reservado";
                                }else{
                                        s[2][cursores[2]]="Reservado";
                                }
                                s[3][cursores[2]]="7 Bits";
                                cursores[2]++;
                                cursores[0]++;
                                contador++;
                        }else if(ext&&cursores[0]<p.length&&contador<longitud&&aux3==96){
                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Bit de Extensión";
                                if((p[cursores[0]]>>7)==1){
                                        s[2][cursores[2]]="Último Octeto";
                                }else{
                                        s[2][cursores[2]]="Continúa en el octeto siguiente";
                                }
                                s[3][cursores[2]]="1 Bit";
                                cursores[2]++;

                                s[0][cursores[2]]="capa6";
                                s[1][cursores[2]]="Para videotelefonía";
                                aux3=p[cursores[0]]&127;
                                if(aux3==1){
                                        s[2][cursores[2]]="Canal inicial de la Recomendación H.221";
                                }else if(aux3==2){
                                        s[2][cursores[2]]="Canal subsiguiente de la Recomendación H.221";
                                }else if(aux3==3){
                                        s[2][cursores[2]]="Canal inicial asociado con una llamada de audio de 3,1 kHz
activo o de conversación";
                                }else{
                                        s[2][cursores[2]]="Reservado";
                                }
                                s[3][cursores[2]]="7 Bits";
                                cursores[2]++;
                                cursores[0]++;
                                contador++;
                        }

                }
}
```

11. Clase DecodificadorTCP

```
public class DecodificadorTCP{

        static int aux;
        static long aux2;
        static int longitudcabecera;
        static boolean finopciones;

        public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){

                s[0][cursores[2]]="capa4";
                s[1][cursores[2]]="Puerto Origen";
                aux=p[cursores[0]]*256;
                cursores[0]=cursores[0]+1;
                aux=aux+p[cursores[0]];
                control[3]=""+aux;
                s[2][cursores[2]]=control[3];
                s[3][cursores[2]]="2 Octetos";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa4";
                s[1][cursores[2]]="Puerto Destino";
                aux=p[cursores[0]]*256;
                cursores[0]=cursores[0]+1;
                aux=aux+p[cursores[0]];
                control[4]=""+aux;
                s[2][cursores[2]]=control[4];
                s[3][cursores[2]]="2 Octetos";
                cursores[0]=cursores[0]+1;
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa4";
                s[1][cursores[2]]="Número de Secuencia";
                aux2=p[cursores[0]];
                cursores[0]++;
                aux2=aux2*256+p[cursores[0]];
                cursores[0]++;
                aux2=aux2*256+p[cursores[0]];
                cursores[0]++;
                aux2=aux2*256+p[cursores[0]];
                cursores[0]++;
                s[2][cursores[2]]=""+aux2;
                s[3][cursores[2]]="4 Octetos";
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa4";
                s[1][cursores[2]]="No. de Acuse de Recibo";
                aux2=p[cursores[0]];
                cursores[0]++;
                aux2=aux2*256+p[cursores[0]];
                cursores[0]++;
                aux2=aux2*256+p[cursores[0]];
                cursores[0]++;
                aux2=aux2*256+p[cursores[0]];
                cursores[0]++;
                s[2][cursores[2]]=""+aux2;
                s[3][cursores[2]]="4 Octetos";
                cursores[2]=cursores[2]+1;

                s[0][cursores[2]]="capa4";
                s[1][cursores[2]]="Posición de los Datos";
                longitudcabecera=(p[cursores[0]]>>4);
                s[2][cursores[2]]=""+longitudcabecera+" Palabras de 32 Bits";
                s[3][cursores[2]]="4 Bits";
                cursores[2]=cursores[2]+1;
```

```
s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Reservado";
aux=(p[cursores[0]]&15);
cursores[0]++;
aux=aux*4+p[cursores[0]]>>6;
s[2][cursores[2]]=""+aux;
s[3][cursores[2]]="6 Bits";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Puntero Urgente";
s[2][cursores[2]]=""+((p[cursores[0]]>>5)&1);
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="No. Acuse de Recibo";
s[2][cursores[2]]=""+((p[cursores[0]]>>4)&1);
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Entregar Inmediatamente";
s[2][cursores[2]]=""+((p[cursores[0]]>>3)&1);
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Reiniciar Conexión";
s[2][cursores[2]]=""+((p[cursores[0]]>>2)&1);
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Sincronizar Nros. de Secuencia";
s[2][cursores[2]]=""+((p[cursores[0]]>>1)&1);
s[3][cursores[2]]="1 Bit";
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Últimos Datos del Emisor";
s[2][cursores[2]]=""+(p[cursores[0]]&1);
s[3][cursores[2]]="1 Bit";
cursores[0]++;
cursores[2]=cursores[2]+1;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Ventana";
s[2][cursores[2]]=""+(p[cursores[0]]*256+p[cursores[0]+1]);
s[3][cursores[2]]="2 Octetos";
cursores[0]=cursores[0]+2;
cursores[2]++;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Suma de control";
s[2][cursores[2]]=""+(p[cursores[0]]*256+p[cursores[0]+1]);
s[3][cursores[2]]="2 Octetos";
cursores[0]=cursores[0]+2;
cursores[2]++;

s[0][cursores[2]]="capa4";
s[1][cursores[2]]="Puntero urgente";
s[2][cursores[2]]=""+(p[cursores[0]]*256+p[cursores[0]+1]);
s[3][cursores[2]]="2 Octetos";
cursores[0]=cursores[0]+2;
cursores[2]++;
```

```
                finopciones=false;
                for(int i=0;i<(longitudcabecera-20);i++){
                        if(p[cursores[0]]==0){
                                s[0][cursores[2]]="capa4";
                                s[1][cursores[2]]="Tipo-Opción";
                                if(finopciones){
                                        s[2][cursores[2]]="Relleno Opciones";
                                }else{
                                        s[2][cursores[2]]="Fin de  Opciones";
                                        finopciones=true;
                                }
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]=cursores[0]+1;
                                cursores[2]=cursores[2]+1;
                        }else if(p[cursores[0]]==1){
                                s[0][cursores[2]]="capa3";
                                s[1][cursores[2]]="Tipo-Opción";
                                s[2][cursores[2]]="Sin Operación";
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]=cursores[0]+1;
                                cursores[2]=cursores[2]+1;
                        }else if(p[cursores[0]]==2){
                                s[0][cursores[2]]="capa4";
                                s[1][cursores[2]]="Tipo-Opción";
                                s[2][cursores[2]]="Máximo Tamaño de Segmento";
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]=cursores[0]+1;
                                cursores[2]=cursores[2]+1;

                                s[0][cursores[2]]="capa4";
                                s[1][cursores[2]]="Longitud-Opción";
                                s[2][cursores[2]]=""+p[cursores[0]];
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]=cursores[0]+1;
                                cursores[2]=cursores[2]+1;
                                i++;

                                s[0][cursores[2]]="capa4";
                                s[1][cursores[2]]="Máximo Tamaño de Segmento";
                                s[2][cursores[2]]=""+(p[cursores[0]]*256+p[cursores[0]+1]);
                                s[3][cursores[2]]="1 Octeto";
                                cursores[0]=cursores[0]+2;
                                cursores[2]=cursores[2]+1;
                                i=i+2;
                        }
                }

                if(cursores[0]<p.length){
                        s[0][cursores[2]]="desconocido4";
                        s[1][cursores[2]]="Datos";
                        s[2][cursores[2]]="Protocolo Desconocido";
                        s[3][cursores[2]]=""+(p.length-cursores[0])+" Octetos";
                        cursores[2]=cursores[2]+1;
                        s[0][0]="desconocido4";
                        s[1][0]="Datos";
                        s[2][0]="Protocolo Desconocido";
                        s[3][0]=""+(p.length-cursores[0])+" Octetos";
                }
        }
}
```

12. Clase DecodificadorTPKT

```
public class DecodificadorTPKT{

        static int aux;
```

```
	public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){

		s[0][cursores[2]]="capa5";
		s[1][cursores[2]]="Versión";
		s[2][cursores[2]]=""+p[cursores[0]];
		s[3][cursores[2]]="1 Octeto";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;

		s[0][cursores[2]]="capa5";
		s[1][cursores[2]]="Reservado";
		s[2][cursores[2]]=""+p[cursores[0]];
		s[3][cursores[2]]="1 Octeto";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;

		s[0][cursores[2]]="capa5";
		s[1][cursores[2]]="Longitud del Paquete";
		aux=p[cursores[0]]*256;
		cursores[0]=cursores[0]+1;
		aux=aux+p[cursores[0]];
		s[2][cursores[2]]=""+aux;
		s[3][cursores[2]]="2 Octetos";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;

		if(cursores[0]<p.length){
			s[0][cursores[2]]="desconocido5";
			s[1][cursores[2]]="Datos";
			s[2][cursores[2]]="Protocolo Desconocido";
			s[3][cursores[2]]=""+(p.length-cursores[0])+" Octetos";
			cursores[2]=cursores[2]+1;
			s[0][0]="desconocido5";
			s[1][0]="Datos";
			s[2][0]="Protocolo Desconocido";
			s[3][0]=""+(p.length-cursores[0])+" Octetos";
		}
	}
}
```

13. Clase DecodifcadorUDP

```
public class DecodificadorUDP{

	static int aux;

	public static void Decodificar(int p[],String s[][],int cursores[], String protocolos[],String[] control){

		s[0][cursores[2]]="capa4";
		s[1][cursores[2]]="Puerto Origen";
		aux=p[cursores[0]]*256;
		cursores[0]=cursores[0]+1;
		aux=aux+p[cursores[0]];
		control[3]=""+aux;
		s[2][cursores[2]]=control[3];
		s[3][cursores[2]]="2 Octetos";
		cursores[0]=cursores[0]+1;
		cursores[2]=cursores[2]+1;

		s[0][cursores[2]]="capa4";
		s[1][cursores[2]]="Puerto Destino";
		aux=p[cursores[0]]*256;
		cursores[0]=cursores[0]+1;
		aux=aux+p[cursores[0]];
		control[4]=""+aux;
		s[2][cursores[2]]=control[4];
```

```
                    s[3][cursores[2]]="2 Octetos";
                    cursores[0]=cursores[0]+1;
                    cursores[2]=cursores[2]+1;

                    s[0][cursores[2]]="capa4";
                    s[1][cursores[2]]="Longitud";
                    aux=p[cursores[0]]*256;
                    cursores[0]=cursores[0]+1;
                    aux=aux+p[cursores[0]];
                    s[2][cursores[2]]=""+aux;
                    s[3][cursores[2]]="2 Octetos";
                    cursores[0]=cursores[0]+1;
                    cursores[2]=cursores[2]+1;

                    s[0][cursores[2]]="capa4";
                    s[1][cursores[2]]="Suma de Control";
                    aux=p[cursores[0]]*256;
                    cursores[0]=cursores[0]+1;
                    aux=aux+p[cursores[0]];
                    s[2][cursores[2]]=""+aux;
                    s[3][cursores[2]]="2 Octetos";
                    cursores[0]=cursores[0]+1;
                    cursores[2]=cursores[2]+1;

                    if(cursores[0]<p.length){
                            s[0][cursores[2]]="desconocido4";
                            s[1][cursores[2]]="Datos";
                            s[2][cursores[2]]="Protocolo Desconocido";
                            s[3][cursores[2]]=""+(p.length-cursores[0])+" Octetos";
                            cursores[2]=cursores[2]+1;
                            s[0][0]="desconocido4";
                            s[1][0]="Datos";
                            s[2][0]="Protocolo Desconocido";
                            s[3][0]=""+(p.length-cursores[0])+" Octetos";
                    }

            }
}
```

14. Clase Etiqueta1

```
import javax.swing.*;
import java.awt.*;

public class Etiqueta1 extends JLabel{


        public Etiqueta1(String s){
                setText(s);
                setHorizontalAlignment(CENTER);
                setBorder(BorderFactory.createLineBorder(Color.black));
        }

}
```

15. Clase Etiqueta2

```
import javax.swing.*;
import java.awt.*;

public class Etiqueta2 extends JLabel{

        Color color= new Color(255,255,222);

        public Etiqueta2(String s){
                setText(s);
                setHorizontalAlignment(CENTER);
```

```
                setBackground(color);
                setOpaque(true);
                setBorder(BorderFactory.createLineBorder(Color.black));
        }

}
```

16. Clase Etiqueta3

```
import javax.swing.*;
import java.awt.*;

public class Etiqueta3 extends JLabel{

        Color color= new Color(255,237,237);

        public Etiqueta3(String s){
                setText(s);
                setHorizontalAlignment(CENTER);
                setBackground(color);
                setOpaque(true);
                setBorder(BorderFactory.createLineBorder(Color.black));
        }
}
```

17. Clase Etiqueta4

```
import javax.swing.*;
import java.awt.*;

public class Etiqueta4 extends JLabel{

        Color color= new Color(191,237,255);

        public Etiqueta4(String s){
                setText(s);
                setHorizontalAlignment(CENTER);
                setBackground(color);
                setOpaque(true);
                setBorder(BorderFactory.createLineBorder(Color.black));
        }
}
```

18. Clase Etiqueta5

```
import javax.swing.*;
import java.awt.*;

public class Etiqueta5 extends JLabel{

        Color color= new Color(204,255,204);

        public Etiqueta5(String s){
                setText(s);
                setHorizontalAlignment(CENTER);
                setBackground(color);
                setOpaque(true);
                setBorder(BorderFactory.createLineBorder(Color.black));
        }

}
```

19. Clase Etiqueta6

```
import javax.swing.*;
import java.awt.*;
```

```
public class Etiqueta6 extends JLabel{

	Color color= new Color(255,230,255);

	public Etiqueta6(String s){
		setText(s);
		setHorizontalAlignment(CENTER);
		setBackground(color);
		setOpaque(true);
		setBorder(BorderFactory.createLineBorder(Color.black));
	}

}
```

20. Clase MostradordeEtiquetas

```
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;

public class MostradordeEtiqueta extends JLabel implements TableCellRenderer {

   public MostradordeEtiqueta() {

   }

   public Component getTableCellRendererComponent(
                        JTable table, Object etiqueta,
                        boolean isSelected, boolean hasFocus,
                        int row, int column) {
        setBackground(new Color(153, 0, 153));
        return (JLabel)etiqueta;
   }
}
```

21. Clase PlantillaModeloTabla

```
import javax.swing.*;
import javax.swing.table.*;

public class PlantillaModeloTabla extends DefaultTableModel{

	public Class getColumnClass(int c) {
      return JLabel.class;
   }

   public boolean isCellEditable(int row, int col) {
      return false;
   }

}
```

22. Clase Utilidades

```
import java.math.BigInteger;
import javax.swing.*;

public class Utilidades {

	static byte[] dataimagennuevo={-119,80,78,71,13,10,26,10,0,0,0,13,73,72,68,82,0,0,0,16,0,0,0,16,8,4,0,0,0,-75,-
6,55,-22,0,0,0,4,103,65,77,65,0,0,-81,-56,55,5,-118,-
23,0,0,0,25,116,69,88,116,83,111,102,116,119,97,114,101,0,65,100,111,98,101,32,73,109,97,103,101,82,101,97,100,121,113,-
55,101,60,0,0,0,-72,73,68,65,84,40,-49,117,-111,91,14,-62,32,16,69,-89,-82,-61,38,-18,-127,-122,125,-102,-24,-121,27,-44,-8,-
118,-75,79,40,-44,-21,64,95,-42,-106,-100,-31,-121,57,97,46,25,2,-47,-106,4,-55,63,118,-76,1,57,92,9,-107,-102,-42,98,66,-31,-
80,31,20,87,-46,-76,5,74,-49,27,47,92,89,-71,-32,-40,43,68,17,73,-117,26,-102,79,-115,28,41,110,44,84,40,112,98,101,20,-76,-
89,-26,70,-122,-5,56,-118,-77,68,-67,96,-4,-123,-31,-23,110,-52,-125,95,57,-49,-123,-127,78,-55,56,-55,115,93,-24,-108,-118,-
77,100,33,-63,-94,-31,52,21,19,20,-116,15,-84,-42,4,-29,105,24,21,22,-102,-111,-64,47,38,126,-123,-49,60,69,79,75,-78,-
```

```
37,69,-94,75,-37,46,-37,58,39,-47,9,49,37,-117,117,59,4,-59,-96,47,50,-91,91,-16,-82,121,-74,79,0,0,0,0,73,69,78,68,-82,66,96,-
126};
		static byte[] dataimagenabrir={71,73,70,56,57,97,16,0,16,0,-9,0,0,-9,-8,-106,-10,-37,94,-12,-98,51,-10,-43,80,-9,-8,-
104,-12,-88,50,-22,-119,52,-11,-79,50,-11,-71,50,-11,-51,65,-12,-107,51,-11,-65,49,-11,-61,49,-11,-57,56,-9,-7,-103,-28,-
126,52,-26,-83,80,-14,-84,50,-37,119,52,-20,-58,100,-17,-113,51,-24,-87,54,-33,-90,96,-29,-74,107,-45,127,70,-14,-101,51,-26,-
65,112,-14,-26,-117,-9,-49,86,-23,-57,118,-29,-126,52,-15,-46,99,-24,-90,69,-20,-47,121,-12,-21,-118,-24,-107,57,-25,-123,52,-
40,-112,81,-36,-99,90,-24,-119,53,-17,-35,-122,-12,-18,-113,-15,-54,-125,-20,-45,125,-18,-62,84,-24,-97,51,-7,-40,118,-20,-
46,124,-13,-39,102,-24,-59,117,-27,-69,110,-30,-80,103,-27,-125,52,-21,-92,63,-10,-31,105,-9,-12,-113,-9,-17,-124,-9,-14,-118,-
10,-26,116,-10,-21,124,-9,-10,-109,-89,105,29,-59,-125,50,0,0,0,0,0,0,16,120,-13,111,0,120,-13,111,0,-2,0,0,0,111,-4,79,0,44,-
13,111,0,36,11,-9,-65,9,4,0,0,0,16,0,16,-80,35,0,16,120,-13,111,0,-2,0,0,0,111,-4,79,0,0,0,0,0,108,-74,1,0,15,0,-20,8,108,-
13,111,0,-12,97,2,0,-89,9,36,124,3,0,0,0,0,0,0,0,0,97,2,0,43,124,-57,8,12,0,0,0,-80,35,0,16,103,105,102,0,-20,-
126,115,123,0,0,0,0,2,0,-41,17,0,0,0,16,0,0,0,16,-16,-29,-10,-65,0,0,0,16,90,-29,-10,-65,0,0,0,16,0,0,0,16,0,0,0,16,111,-
4,79,0,84,-13,111,0,-48,6,-9,-65,0,0,0,16,1,0,0,16,2,0,0,0,48,-13,111,0,0,0,0,0,-2,0,0,0,111,-4,79,0,108,-74,1,0,0,0,0,16,-20,-
14,111,0,36,-13,111,0,0,0,0,0,-92,-13,111,0,32,27,-5,-65,0,-107,-10,-65,-1,-1,-1,-1,108,-13,111,0,84,13,-9,-
65,0,0,0,16,16,48,0,16,-108,-13,111,0,0,0,0,0,-76,-13,111,0,-4,12,-9,-65,0,0,0,16,16,48,0,16,-108,-13,111,0,0,0,0,0,0,0,0,0,-
2,0,0,0,111,-4,79,0,32,27,-5,-65,84,-102,1,16,122,91,72,0,0,0,0,0,98,107,-10,-65,0,0,0,0,-2,0,0,0,111,-4,79,0,-12,-
13,111,0,35,14,-9,-65,104,91,72,0,98,-98,1,16,18,0,0,0,118,91,72,0,-54,-48,71,0,111,-4,79,0,-8,-5,79,0,0,0,0,0,18,0,0,0,-60,-
13,111,0,-30,19,-10,-65,-64,-8,111,0,32,27,-5,-65,-80,-107,-10,-65,-1,-1,-1,-1,-8,-5,79,0,-85,-87,66,0,0,0,0,16,-1,-1,-1,-
1,104,91,72,0,-1,0,0,0,-63,-86,66,0,44,121,0,0,104,91,72,0,-1,0,0,0,111,-4,79,0,125,-12,111,0,-87,-36,65,0,104,-12,111,0,-
52,1,0,0,-8,-3,79,0,28,1,0,0,1,-91,-10,-65,0,0,77,0,20,-1,79,0,-52,1,0,0,0,0,0,0,12,0,77,0,0,0,77,0,-8,-
3,79,0,64,0,0,0,0,0,0,0,0,5,0,0,0,5,0,0,-112,-12,111,0,-79,-90,-10,-65,0,0,77,0,-56,-90,-10,-
65,0,0,77,0,65,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,-60,-64,68,0,24,-2,79,0,-96,-94,78,0,28,-2,79,0,-92,-9,111,0,-1,-1,-1,-1,-4,-
3,79,0,46,-63,68,0,24,-2,79,0,33,-7,4,1,0,0,63,0,44,0,0,0,0,16,0,16,0,64,8,-128,0,127,8,28,72,-80,-32,-64,12,13,124,-24,-64,-
111,-93,-95,-61,28,62,4,-10,16,64,-111,-94,-113,-117,62,6,104,28,16,81,-30,12,12,23,13,-118,28,-40,-93,-92,-63,30,8,16,40,-
44,-63,-78,37,-53,9,29,123,-88,0,64,-77,38,-128,20,26,44,-116,-36,-55,-77,103,-55,-110,6,12,72,-24,-15,35,-28,-113,30,5,-
110,94,-84,-63,34,-128,-45,-89,29,23,44,80,-72,-93,-86,-43,-86,29,-113,114,-48,-79,35,-57,-115,-81,96,111,100,61,10,-128,-
128,89,2,29,76,96,52,104,-76,-89,-37,31,1,1,0,59};
		static byte[] dataimagenguardar={71,73,70,56,57,97,16,0,15,0,-9,0,0,-1,-1,-1,-1,33,0,-1,0,0,-9,-9,-9,-9,33,0,-9,0,0,-
17,-17,-17,-17,24,0,-17,0,0,-25,-25,-25,-25,24,0,-25,8,0,-34,-34,-34,-34,24,0,-34,8,0,-42,-42,-42,-42,24,0,-42,8,0,-50,-50,-50,-
50,24,0,-50,16,0,-58,-58,-58,-58,16,0,-67,-67,-67,-67,16,0,-75,-75,-75,-75,24,0,-75,16,0,-83,-83,-83,-83,24,0,-83,16,0,-83,8,0,-
91,-91,-91,-91,24,0,-91,8,0,-91,0,0,-100,-100,-100,-100,33,0,-100,8,0,-108,-108,-108,-108,33,0,-108,8,0,-116,-116,-116,-
116,0,0,-124,-124,-124,-124,0,0,123,123,123,123,0,0,115,115,115,115,0,0,107,107,107,107,0,0,99,99,99,90,90,90,82,82,82,-
64,-64,-64,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,16,88,-13,111,0,88,-13,111,0,-2,0,0,0,111,-4,79,0,12,-
13,111,0,36,11,-9,-65,9,4,0,0,0,16,0,16,-80,35,0,16,88,-13,111,0,-2,0,0,0,111,-4,79,0,-39,61,-121,9,108,-74,1,0,0,0,0,0,76,-
13,111,0,40,-35,-36,-126,0,0,3,0,3,0,0,0,47,-117,-89,8,0,0,0,0,0,0,0,0,12,0,0,0,-80,35,0,16,103,105,102,0,-
89,9,36,124,0,0,0,0,0,0,0,0,0,0,0,16,0,0,0,16,-16,-29,-10,-65,0,0,0,16,90,-29,-10,-65,0,0,0,16,0,0,0,16,0,0,0,16,111,-4,79,0,52,-
13,111,0,-48,6,-9,-65,0,0,0,16,1,0,0,16,2,0,0,0,16,-13,111,0,0,0,0,0,-2,0,0,0,111,-4,79,0,108,-74,1,0,0,0,0,16,-52,-14,111,0,4,-
13,111,0,0,0,0,0,-124,-13,111,0,32,27,-5,-65,0,-107,-10,-65,-1,-1,-1,-1,76,-13,111,0,84,13,-9,-65,0,0,0,16,16,48,0,16,116,-
13,111,0,0,0,0,0,-108,-13,111,0,-4,12,-9,-65,0,0,0,16,16,48,0,16,116,-13,111,0,0,0,0,0,0,0,0,0,-2,0,0,0,111,-4,79,0,32,27,-5,-
65,84,-102,1,16,122,91,72,0,0,0,0,0,98,107,-10,-65,0,0,0,0,-2,0,0,0,111,-4,79,0,-44,-13,111,0,35,14,-9,-65,104,91,72,0,98,-
98,1,16,18,0,0,0,118,91,72,0,-54,-48,71,0,111,-4,79,0,-8,-5,79,0,0,0,0,0,18,0,0,0,-92,-13,111,0,-30,19,-10,-65,-96,-
8,111,0,32,27,-5,-65,-80,-107,-10,-65,-1,-1,-1,-1,-8,-5,79,0,-85,-87,66,0,0,0,0,16,-1,-1,-1,-1,104,91,72,0,-1,0,0,0,-63,-
86,66,0,44,121,0,0,104,91,72,0,-1,0,0,0,111,-4,79,0,93,-12,111,0,-87,-36,65,0,72,-12,111,0,-128,0,0,0,68,-1,79,0,28,1,0,0,1,-91,-
10,-65,0,0,77,0,96,0,80,0,-128,0,0,0,0,0,0,0,12,0,77,0,0,0,77,0,68,-1,79,0,64,0,0,0,0,0,0,0,0,5,0,0,0,5,0,0,112,-12,111,0,-79,-90,-
10,-65,0,0,77,0,-56,-90,-10,-65,0,0,77,0,65,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,-60,-64,68,0,102,-1,79,0,-96,-94,78,0,106,-
1,79,0,-122,-9,111,0,-1,-1,-1,-1,72,-1,79,0,46,-63,68,0,102,-1,79,0,33,-7,4,1,0,0,55,0,44,0,0,0,0,16,0,15,0,64,8,-
109,0,5,8,20,112,-29,70,-122,-125,8,51,-116,16,-80,80,96,0,2,10,26,64,-96,-128,1,-61,7,17,38,6,10,28,17,34,68,2,9,23,56,-
108,40,65,67,-93,-64,-120,19,44,96,-16,-128,49,-59,-118,23,38,71,56,112,-64,96,0,3,9,11,22,-88,48,-55,-77,-89,70,-
120,13,38,80,-36,112,-47,-60,10,-109,13,40,-88,100,105,34,69,-117,23,49,98,10,4,97,96,64,2,18,88,121,-114,-96,64,1,-64,-
57,11,92,93,104,29,49,2,-85,-39,-84,62,-45,-2,4,-86,20,-61,-122,-106,38,15,68,108,-5,-42,104,-117,-127,51,4,72,28,90,116,-
59,83,1,21,2,2,0,59};
		static byte[] dataimagenescuchar={-119,80,78,71,13,10,26,10,0,0,0,13,73,72,68,82,0,0,0,16,0,0,0,16,8,6,0,0,0,31,-
13,-1,97,0,0,0,4,103,65,77,65,0,0,-81,-56,55,5,-118,-
23,0,0,0,25,116,69,88,116,83,111,102,116,119,97,114,101,0,65,100,111,98,101,32,73,109,97,103,101,82,101,97,100,121,113,-
55,101,60,0,0,2,95,73,68,65,84,56,-53,-91,-109,77,111,18,81,20,-122,-5,3,-4,5,-4,14,88,-70,116,-61,-82,-102,46,-70,51,113,99,-
62,-58,-60,69,83,87,37,53,-43,69,99,10,97,44,81,76,-101,-42,6,42,17,107,-86,-74,-87,-111,90,-23,20,-118,105,44,31,-
91,12,31,2,22,-24,80,-104,-63,97,96,96,24,-54,113,-50,-123,25,-87,-70,-13,38,-17,102,-26,60,-17,-7,-72,-25,-114,1,-64,-40,-
88,110,44,-126,65,-107,73,-107,89,-43,-28,80,-26,-31,55,-61,-97,-15,-93,-32,53,85,-58,7,111,4,-117,-17,-32,-116,10,29,-89,-23,-
40,-55,41,23,-115,39,-72,-3,-93,36,-3,106,47,71,77,121,121,11,-58,96,-20,21,-125,33,124,-35,19,40,91,-93,-89,89,-122,-
83,9,32,74,93,80,46,-5,68,-115,-106,12,-91,10,15,-63,-93,56,-77,-76,-109,-75,98,-84,102,-94,25,24,17,-114,39,-65,-117,-19,-
114,2,120,36,-71,7,-68,-40,5,78,85,-77,-45,35,-33,68,73,-122,-3,112,68,116,109,-91,-48,-60,72,12,-80,47,44,27,51,107,112,85,-
112,-95,-14,83,-122,-73,-31,10,-80,117,25,-54,-11,14,-108,-8,-114,110,-78,-77,27,100,-18,-83,-79,-40,-114,1,13,76,62,-6,7,-
123,101,-113,-62,8,-34,122,24,-128,-115,-61,10,-127,-49,106,109,34,60,-55,-36,57,56,-68,33,10,89,52,48,7,-65,-91,104,-
20,25,75,69,80,-53,122,115,118,-113,-24,117,-120,37,112,-95,-38,6,-95,-91,-64,69,93,-126,-27,-11,119,52,-78,104,48,25,-119,-
99,112,56,-84,90,-93,75,96,-52,88,-28,-38,42,-4,-103,104,-30,81,8,-4,49,14,114,21,-119,24,-55,-54,37,-84,-71,-41,57,100,-119,-
63,113,52,78,12,-80,124,13,-58,64,2,-49,29,-64,-8,-116,-97,-4,-53,-78,-46,-80,-118,46,44,-81,-68,-44,13,-52,95,-66,38,-24,-70,-
40,-127,-122,-92,-24,-67,98,-32,-60,28,125,5,-50,-100,-73,-56,-51,-92,-117,117,112,-72,-36,122,11,38,-73,63,67,21,74,53,-24,-
```

```
9,-127,-128,-7,11,-119,-108,59,62,-13,-119,-64,8,-90,-53,45,72,-107,90,-48,-19,-11,97,-9,48,1,86,-25,7,125,-120,-122,-5,-98,-
86,-59,31,8,51,-100,48,-104,50,102,67,33,-84,-127,-55,82,-109,36,-56,-88,-39,-19,-49,86,-103,59,-50,-12,-32,26,-
75,69,114,108,68,-83,-101,31,3,34,-53,15,2,-21,77,69,-81,6,-105,9,51,-89,-117,60,-40,92,30,113,-38,-79,-11,123,-
111,70,87,121,-34,29,-74,-82,122,55,-103,80,36,3,5,86,32,51,-63,-98,19,121,14,-74,3,81,-104,119,44,49,83,-10,-9,127,-81,-14,-
24,99,-70,-5,34,111,-103,125,-66,77,61,89,92,-95,29,79,-99,-36,-126,-99,-30,30,-37,92,-12,-76,-51,71,-35,-90,-110,-1,126,76,-
1,-13,-100,127,1,29,120,21,87,-52,-62,-98,37,0,0,0,0,73,69,78,68,-82,66,96,-126};
        static byte[] dataimagentramasiguiente={-
119,80,78,71,13,10,26,10,0,0,0,13,73,72,68,82,0,0,0,16,0,0,0,16,8,6,0,0,0,31,-13,-1,97,0,0,0,4,103,65,77,65,0,0,-81,-56,55,5,-
118,-
23,0,0,0,25,116,69,88,116,83,111,102,116,119,97,114,101,0,65,100,111,98,101,32,73,109,97,103,101,82,101,97,100,121,113,-
55,101,60,0,0,2,114,73,68,65,84,56,-53,-91,83,75,111,18,81,20,-18,15,-16,23,-16,59,96,-23,-46,13,-69,86,107,-46,-99,-
119,27,19,54,38,46,-102,-70,-79,-109,-6,-24,-94,49,66,102,44,81,76,-101,-42,6,42,9,-102,-44,-38,-90,-90,84,-100,14,32,-90,-
79,80,-96,12,15,-127,-14,40,20,6,28,6,-122,-57,80,-114,115,46,15,-15,-79,-13,38,-33,-30,-34,-100,-17,59,-25,59,-25,-36,9,0,-
104,24,-57,-75,101,-48,-88,-48,-87,-48,-85,-104,25,64,63,120,-45,-4,25,63,78,-68,-94,66,123,-1,-83,104,112,-72,51,-116,-9,56,-
58,-99,-124,78,-123,64,48,44,28,30,69,-72,55,-82,36,51,107,-81,24,48,6,99,127,19,24,-112,-81,-38,-40,60,21,56,77,-16,-123,-
78,8,-110,-36,1,-27,-78,71,80,107,-76,33,87,-84,-128,-25,40,-56,-81,-20,37,40,-116,29,-118,12,5,-76,72,14,70,-66,75,-51,-
106,2,120,-28,118,23,42,82,7,4,21,-11,86,-105,-68,73,114,27,14,125,126,-55,-78,19,69,17,45,17,64,95,88,54,102,30,-110,-29,-
25,13,40,-2,104,67,-95,-38,71,52,-41,-128,92,-91,53,18,-39,59,-16,-16,119,55,10,104,71,-125,2,58,7,119,-58,96,-
39,120,74,98,27,110,60,98,9,9,-55,-7,106,11,-90,22,92,16,-54,72,-112,41,55,73,76,36,121,14,-76,-35,-53,32,23,5,-12,-
98,111,81,14,61,99,-87,72,-70,-2,-112,85,-31,-126,72,-82,14,89,-95,-87,10,124,34,8,-98,73,32,54,20,-72,-88,-54,-80,-70,-7,-
98,67,46,10,-52,-8,79,66,2,54,-85,92,-21,16,1,-52,-120,-72,-71,-24,-123,88,-66,1,83,-108,42,64,57,97,-6,-79,-101,-36,-37,-
54,37,108,88,55,5,-28,18,-127,-29,64,-112,8,96,-7,-24,21,-55,-45,79,-36,36,43,-118,78,-50,59,-119,-83,-55,-
7,125,114,23,27,29,88,93,123,61,18,-48,127,-2,26,-26,-86,82,11,106,-78,66,124,18,50,-43,39,39,-117,-14,-120,-116,9,112,50,-
79,108,21,104,-117,117,100,65,103,117,-58,-103,116,-82,12,-67,30,64,-70,-44,36,-27,34,57,81,-112,9,38,31,-20,-109,-
87,96,99,59,-35,30,28,124,9,3,101,-2,48,106,-94,-26,-98,-83,100,112,-78,62,94,16,-101,-93,73,32,17,-57,-119,-98,-111,-
116,13,-59,4,113,53,-69,-23,-59,58,127,-37,28,-21,-113,113,-72,72,-12,-69,0,-75,-11,-111,-107,10,-107,126,96,-75,-82,-
112,106,82,23,50,89,38,-52,28,-53,86,-64,104,-79,73,115,-12,-50,-81,69,26,95,-27,37,-85,-113,90,-73,111,-15,94,127,28,-46,5,-
111,-12,4,61,-121,83,2,-20,-78,1,88,-94,87,-8,89,-45,-10,-33,-85,60,-2,-103,-18,-68,74,25,22,94,-18,50,79,-105,-41,56,-6,-
71,89,120,102,98,-124,69,-93,-123,-101,51,58,-104,91,76,-28,-33,-97,-23,127,-66,-13,79,-29,62,8,95,-109,103,116,-
37,0,0,0,0,73,69,78,68,-82,66,96,-126};
        static byte[] dataimagentramaanterior={-
119,80,78,71,13,10,26,10,0,0,0,13,73,72,68,82,0,0,0,16,0,0,0,16,8,6,0,0,0,31,-13,-1,97,0,0,0,4,103,65,77,65,0,0,-81,-56,55,5,-
118,-
23,0,0,0,25,116,69,88,116,83,111,102,116,119,97,114,101,0,65,100,111,98,101,32,73,109,97,103,101,82,101,97,100,121,113,-
55,101,60,0,0,2,123,73,68,65,84,56,-53,-91,-109,77,111,18,81,20,-122,-5,3,-4,5,-4,14,88,-70,116,67,-94,-
15,35,106,81,23,38,110,76,-40,-104,-72,104,90,19,35,-87,49,-115,-59,90,8,99,-119,98,-38,-76,54,80,73,-48,4,-76,-75,-58,-
42,74,71,-112,-122,20,40,-91,12,31,2,22,40,-76,12,8,3,3,-61,80,-114,115,46,31,82,117,-25,77,-34,76,-26,-50,121,-97,115,-50,-
99,115,-121,0,96,104,80,103,102,64,38,73,33,73,41,73,-43,-107,-78,-69,39,-5,51,126,-48,120,74,-110,124,-20,77,89,109,-5,-
70,79,-71,125,81,122,103,119,-113,13,4,67,-20,-90,55,76,-65,-34,72,80,35,-42,-94,26,99,48,-10,4,-96,107,62,109,113,102,53,-
127,-67,56,-109,43,-108,-127,-29,-101,32,30,-73,-119,42,53,1,50,-7,34,-72,-68,65,102,118,53,-82,-63,-40,30,-92,7,-112,-
93,57,24,-2,-50,-43,27,34,-32,-30,-123,22,20,-71,38,-80,-110,-86,-115,22,-39,-29,120,1,54,61,126,-50,-76,28,65,-120,-100,0,-
80,47,44,27,51,-9,-52,71,101,1,-10,11,117,-56,-107,4,-94,108,-87,1,-119,60,-33,-121,-84,-82,-69,-104,59,-117,57,108,71,-
122,0,-123,-115,-2,65,97,-39,-125,102,-107,-42,123,-62,60,60,-71,69,-10,113,-123,19,7,96,-80,-70,41,-12,34,64,-23,-38,-114,-
48,-40,51,-106,-102,58,-86,-61,-91,-121,78,2,-56,20,27,16,-49,-15,112,-18,-98,-125,0,-16,91,-71,38,-62,97,-119,-121,-71,37,7,-
115,94,4,-88,-4,59,-69,44,30,86,-2,-89,0,23,-57,63,75,-38,32,6,-52,124,118,-52,46,-55,1,87,31,123,-56,59,86,33,-120,-57,-
80,104,94,98,-47,75,0,-66,64,-112,0,10,-107,38,92,-101,-14,-61,-123,7,-97,8,0,-127,-105,31,-47,112,-2,-2,7,-72,50,-31,38,-
43,116,-86,104,-62,-36,-4,-85,62,64,-7,101,43,68,-105,-72,6,84,120,17,-46,-84,-44,-1,19,-97,-44,-62,54,36,15,59,6,-
124,13,79,122,33,118,80,35,127,38,-102,46,-127,-63,100,-18,-73,-96,48,-81,-59,-88,84,-90,0,-19,54,16,3,-106,121,99,58,64,50,-
94,-80,-12,-21,79,-3,16,-55,-44,-96,-39,106,-61,-6,-73,16,104,-116,-17,-5,-121,40,-69,107,57,82,-81,57,61,12,91,-18,-100,50,-
102,48,59,102,-116,102,107,-60,-120,79,76,16,-109,-78,-21,-97,47,48,-73,-116,-47,-50,111,-20,13,-110,-31,109,64,99,-1,-24,-
28,114,-59,42,9,44,85,69,82,13,-126,112,-104,48,115,52,93,4,-99,-55,-62,-115,26,-106,127,15,-46,-32,40,107,-51,30,-51,-126,-
43,-50,-72,-3,49,72,-27,-54,-28,76,-80,-25,80,-110,-123,21,103,0,-76,-122,89,102,68,-1,-18,-17,81,30,-68,76,-73,95,38,-43,-
29,47,86,-88,-87,-103,121,-38,-16,-52,-56,78,-21,41,118,66,103,-94,71,117,54,-22,38,21,-2,-9,101,-6,-97,-21,-4,11,-52,-
124,9,60,-23,-52,58,46,0,0,0,0,73,69,78,68,-82,66,96,-126};
        static byte[] dataimagendetener={-119,80,78,71,13,10,26,10,0,0,0,13,73,72,68,82,0,0,0,16,0,0,0,16,8,6,0,0,0,31,-13,-
1,97,0,0,0,4,103,65,77,65,0,0,-81,-56,55,5,-118,-
23,0,0,0,25,116,69,88,116,83,111,102,116,119,97,114,101,0,65,100,111,98,101,32,73,109,97,103,101,82,101,97,100,121,113,-
55,101,60,0,0,2,99,73,68,65,84,56,-53,-91,-109,75,111,18,97,20,-122,-5,3,-4,5,-4,14,88,-70,116,-61,-50,-22,-90,59,-29,-35,-
80,49,113,-47,-44,-107,-92,-102,-72,104,-116,-112,-103,66,20,-45,-90,-75,-127,74,-126,38,85,-37,-44,72,91,-
23,8,98,26,11,29,40,83,64,-64,2,14,-123,97,28,6,-122,-53,80,-114,115,62,-104,17,47,59,39,121,23,-13,-27,-68,-49,121,-49,119,-
103,0,-128,-119,113,-99,115,-127,73,-109,69,-109,85,-45,-44,72,-42,-47,-102,-23,-49,-6,113,-29,25,77,-26,-69,47,37,91,-32,-
29,49,29,-39,79,51,7,-119,67,33,-50,38,-123,-35,-67,20,-13,98,39,71,79,-5,-21,54,-84,-63,-38,-33,0,35,-13,89,95,-88,108,-
113,31,102,57,-66,38,-127,-84,-12,64,61,29,16,53,90,93,40,85,-22,16,-34,99,-71,-123,-51,-84,29,107,117,-120,14,48,-93,-
103,77,125,-107,-37,29,21,-16,83,-70,125,-88,-53,61,16,52,53,59,125,-78,38,43,93,-40,-115,-58,100,-49,-6,17,66,-52,4,-
128,115,97,108,-20,-84,-101,-85,82,23,42,63,-70,-64,-117,67,-107,-59,14,-108,-22,29,3,-78,-71,21,-26,110,-81,-16,56,-114,9,1,-
106,0,-13,-115,-58,-40,-70,-7,-30,-125,16,-36,114,39,13,-13,-28,-20,14,92,119,37,72,26,-4,82,-71,-17,64,-
7,35,52,122,17,96,13,127,57,98,112,102,-116,-118,-90,11,-9,17,-112,32,93,-117,66,91,3,108,19,64,-83,-47,3,-87,-91,-62,-119,-
```

```
88,-64,-30,-22,107,6,-67,8,-104,-118,29,36,4,-36,44,44,64,0,118,-68,-95,1,-114,107,109,-94,73,-5,54,92,-101,103,73,58,-4,-17,-
86,-89,-80,-30,93,21,-48,75,0,-5,113,-106,0,-80,0,-69,-22,-111,-79,-72,80,109,-61,-7,123,65,-72,74,15,1,-8,47,-75,122,-80,-72,-
12,-36,0,88,63,124,78,50,-94,-36,-127,-122,-94,14,59,-110,-56,44,-28,79,20,-56,85,20,13,-16,-98,0,112,99,-15,100,-
46,69,17,40,-113,-41,24,-63,-30,13,102,-24,66,-87,6,-125,1,-112,14,24,23,-107,-27,21,34,52,95,-90,88,50,94,-81,63,-128,-
83,79,73,-80,-69,-33,26,-101,104,-70,-29,-85,-38,-126,-95,40,39,72,109,-29,36,116,-91,-53,45,-29,56,-47,-100,-47,-70,59,-
97,44,115,87,-36,-23,-31,49,-22,23,-119,122,21,-73,-81,-67,11,-55,124,-67,73,-110,-120,77,-107,-92,-63,49,-16,-8,-48,-100,46,-
42,-63,-31,-15,-55,51,-44,-6,-81,-117,52,126,-107,-25,-68,81,-5,-78,127,-115,-117,-60,50,80,-32,37,-78,39,56,115,50,47,-
64,70,40,14,115,-44,2,55,-19,124,-13,-9,85,30,127,76,55,-97,-27,109,-77,79,55,-24,71,-82,37,-122,-102,119,11,-113,-99,-76,-
16,-48,-31,97,102,28,1,-6,18,-99,-6,-9,99,-6,-97,-25,-4,19,25,63,12,115,113,34,-11,-106,0,0,0,0,73,69,78,68,-82,66,96,-126};
        static byte[] dataimagenopciones={-119,80,78,71,13,10,26,10,0,0,0,13,73,72,68,82,0,0,0,16,0,0,0,16,8,6,0,0,0,31,-
13,-1,97,0,0,0,4,103,65,77,65,0,0,-81,-56,55,5,-118,-
23,0,0,0,25,116,69,88,116,83,111,102,116,119,97,114,101,0,65,100,111,98,101,32,73,109,97,103,101,82,101,97,100,121,113,-
55,101,60,0,0,1,98,73,68,65,84,24,25,-91,-63,33,110,-108,81,24,-123,-31,-9,-36,123,33,77,26,-80,96,73,88,66,29,8,-74,-128,-
87,-63,98,8,10,-59,10,72,-64,-48,-92,73,23,65,5,-37,-87,36,21,-75,85,116,72,-6,-1,-33,57,-52,-99,48,6,55,-23,-13,40,9,15,-95,-
73,-97,126,124,124,-3,-26,-28,-37,-19,38,-57,9,96,19,-101,114,17,7,-57,-92,10,39,-92,-118,-78,121,-6,120,-71,-69,-70,-70,-7,-
4,-13,-5,-23,-59,56,121,117,114,-2,-24,-24,-72,61,59,-30,16,-57,119,-101,58,7,46,-58,102,73,-37,-36,-2,-26,80,127,-106,52,-
74,70,42,124,121,-9,-110,67,-67,-1,122,-51,52,18,51,-35,-81,69,-128,56,-108,-63,4,59,-108,77,-83,-95,-42,80,49,-
9,21,94,60,127,66,-71,-104,6,9,-109,-102,80,66,-102,-120,-126,44,90,19,82,67,42,-44,67,119,99,-84,97,-14,90,76,-93,92,76,13,-
120,4,-126,30,-80,-64,14,61,-48,-44,25,-126,82,-88,102,38,-105,-103,70,28,118,36,72,104,64,36,16,72,-112,10,22,68,64,19,-
35,-99,-55,46,-90,97,23,123,-110,-40,73,104,64,36,60,-60,-80,113,-96,1,22,59,-74,-103,-58,-70,44,-20,72,8,72,-126,36,-
62,86,66,99,-85,53,58,80,9,77,97,90,-105,-107,105,-92,-52,-44,-127,-80,37,65,-62,36,-119,61,3,93,34,54,83,108,-90,97,-
101,61,1,98,75,-30,127,-99,127,122,103,114,-62,52,110,126,93,95,126,56,91,78,93,-123,-53,84,21,-74,-87,101,-63,54,78,-120,-
117,84,112,76,108,108,-90,75,-74,-108,-124,-121,104,60,-48,95,-20,-104,-16,-2,4,59,-18,-18,0,0,0,0,73,69,78,68,-82,66,96,-
126};
        static byte[] dataimagentramaportrama={-1,-40,-1,-32,0,16,74,70,73,70,0,1,1,1,1,44,1,44,0,0,-1,-
37,0,67,0,8,6,6,7,6,5,8,7,7,7,9,9,8,10,12,20,13,12,11,11,12,25,18,19,15,20,29,26,31,30,29,26,28,28,32,36,46,39,32,34,44,35,28,
28,40,55,41,44,48,49,52,52,52,31,39,57,61,56,50,60,46,51,52,50,-1,-
37,0,67,1,9,9,9,12,11,12,24,13,13,24,50,33,28,33,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,
50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,50,-1,-64,0,17,8,0,16,0,16,3,1,34,0,2,17,1,3,17,1,-1,-
60,0,31,0,0,1,5,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,2,3,4,5,6,7,8,9,10,11,-1,-60,0,-
75,16,0,2,1,3,3,2,4,3,5,5,4,4,0,0,1,125,1,2,3,0,4,17,5,18,33,49,65,6,19,81,97,7,34,113,20,50,-127,-111,-95,8,35,66,-79,-
63,21,82,-47,-16,36,51,98,114,-
126,9,10,22,23,24,25,26,37,38,39,40,41,42,52,53,54,55,56,57,58,67,68,69,70,71,72,73,74,83,84,85,86,87,88,89,90,99,100,101,1
02,103,104,105,106,115,116,117,118,119,120,121,122,-125,-124,-123,-122,-121,-120,-119,-118,-110,-109,-108,-107,-106,-
105,-104,-103,-102,-94,-93,-92,-91,-90,-89,-88,-87,-86,-78,-77,-76,-75,-74,-73,-72,-71,-70,-62,-61,-60,-59,-58,-57,-56,-55,-54,-
46,-45,-44,-43,-42,-41,-40,-39,-38,-31,-30,-29,-28,-27,-26,-25,-24,-23,-22,-15,-14,-13,-12,-11,-10,-9,-8,-7,-6,-1,-
60,0,31,1,0,3,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,1,2,3,4,5,6,7,8,9,10,11,-1,-60,0,-
75,17,0,2,1,2,4,4,3,4,7,5,4,4,0,1,2,119,0,1,2,3,17,4,5,33,49,6,18,65,81,7,97,113,19,34,50,-127,8,20,66,-111,-95,-79,-
63,9,35,51,82,-16,21,98,114,-47,10,22,36,52,-31,37,-
15,23,24,25,26,38,39,40,41,42,53,54,55,56,57,58,67,68,69,70,71,72,73,74,83,84,85,86,87,88,89,90,99,100,101,102,103,104,105
,106,115,116,117,118,119,120,121,122,-126,-125,-124,-123,-122,-121,-120,-119,-118,-110,-109,-108,-107,-106,-105,-104,-
103,-102,-94,-93,-92,-91,-90,-89,-88,-87,-86,-78,-77,-76,-75,-74,-73,-72,-71,-70,-62,-61,-60,-59,-58,-57,-56,-55,-54,-46,-45,-44,-
43,-42,-41,-40,-39,-38,-30,-29,-28,-27,-26,-25,-24,-23,-22,-14,-13,-12,-11,-10,-9,-8,-7,-6,-1,-38,0,12,3,1,0,2,17,3,17,0,63,0,-18,-
13,92,-43,87,-82,-70,-72,56,-54,-121,-77,-10,90,-9,-3,12,51,44,-77,-5,115,-105,-34,-28,-28,-65,75,-34,-10,-13,93,-113,-1,-39};

        static ImageIcon icononuevo = new ImageIcon(dataimagennuevo);
        static ImageIcon iconoabrir = new ImageIcon(dataimagenabrir);
        static ImageIcon iconoguardar = new ImageIcon(dataimagenguardar);
        static ImageIcon iconoescuchar = new ImageIcon(dataimagenescuchar);
        static ImageIcon iconotramasiguiente = new ImageIcon(dataimagentramasiguiente);
        static ImageIcon iconotramaanterior = new ImageIcon(dataimagentramaanterior);
        static ImageIcon iconodetener = new ImageIcon(dataimagendetener);
        static ImageIcon iconoopciones = new ImageIcon(dataimagenopciones);
        static ImageIcon iconotramaportrama = new ImageIcon(dataimagentramaportrama);

        public static int[] byteaInt(byte[] c){
                int cint[]=new int[c.length];
                for(int i=0;i<c.length;i++){
                        if(c[i]>=0){
                                cint[i]=c[i];
                        }else{
                                cint[i]=c[i]+256;
                        }
                }
                return cint;
        }
```

```
public static String IntaHex(int b){

		String aux;
		StringBuffer aux2=new StringBuffer(2);
		int w;
		w=b>>4;
		if(w==0){
			aux2.append("0");
		}else if(w==1){
			aux2.append("1");
		}else if(w==2){
			aux2.append("2");
		}else if(w==3){
			aux2.append("3");
		}else if(w==4){
			aux2.append("4");
		}else if(w==5){
			aux2.append("5");
		}else if(w==6){
			aux2.append("6");
		}else if(w==7){
			aux2.append("7");
		}else if(w==8){
			aux2.append("8");
		}else if(w==9){
			aux2.append("9");
		}else if(w==10){
			aux2.append("A");
		}else if(w==11){
			aux2.append("B");
		}else if(w==12){
			aux2.append("C");
		}else if(w==13){
			aux2.append("D");
		}else if(w==14){
			aux2.append("E");
		}else if(w==15){
			aux2.append("F");
		}

		w=b&15;
		if(w==0){
			aux2.append('0');
		}else if(w==1){
			aux2.append('1');
		}else if(w==2){
			aux2.append('2');
		}else if(w==3){
			aux2.append('3');
		}else if(w==4){
			aux2.append('4');
		}else if(w==5){
			aux2.append('5');
		}else if(w==6){
			aux2.append('6');
		}else if(w==7){
			aux2.append('7');
		}else if(w==8){
			aux2.append('8');
		}else if(w==9){
			aux2.append('9');
		}else if(w==10){
			aux2.append('A');
		}else if(w==11){
			aux2.append('B');
		}else if(w==12){
			aux2.append('C');
```

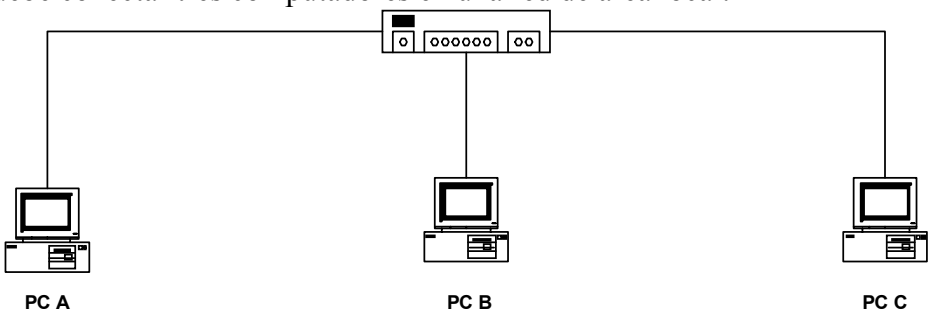
```
		}else if(w==13){
			aux2.append('D');
		}else if(w==14){
			aux2.append('E');
		}else if(w==15){
			aux2.append('F');
		}
		aux=aux2.toString();
		return aux;
}

public static JLabel[][] Colorear(String s[][]){
		JLabel e[][]=new JLabel[3][s[0].length];
		for(int i=0;i<s[0].length;i++){
			if(s[0][i]=="t"){
				if(s[1][i]!=null){
					e[0][i]=new Etiqueta1(s[1][i]);
				}
				if(s[2][i]!=null){
					e[1][i]=new Etiqueta1(s[2][i]);
				}
				if(s[3][i]!=null){
					e[2][i]=new Etiqueta1(s[3][i]);
				}
			}else if(s[0][i]=="capa2"){
				if(s[1][i]!=null){
					e[0][i]=new Etiqueta2(s[1][i]);
				}
				if(s[2][i]!=null){
					e[1][i]=new Etiqueta2(s[2][i]);
				}
				if(s[3][i]!=null){
					e[2][i]=new Etiqueta2(s[3][i]);
				}
			}else if(s[0][i]=="capa3"||s[0][i]=="datos2"||s[0][i]=="desconocido2"){
				if(s[1][i]!=null){
					e[0][i]=new Etiqueta3(s[1][i]);
				}
				if(s[2][i]!=null){
					e[1][i]=new Etiqueta3(s[2][i]);
				}
				if(s[3][i]!=null){
					e[2][i]=new Etiqueta3(s[3][i]);
				}
			}else if(s[0][i]=="capa4"||s[0][i]=="datos3"||s[0][i]=="desconocido3"){
				if(s[1][i]!=null){
					e[0][i]=new Etiqueta4(s[1][i]);
				}
				if(s[2][i]!=null){
					e[1][i]=new Etiqueta4(s[2][i]);
				}
				if(s[3][i]!=null){
					e[2][i]=new Etiqueta4(s[3][i]);
				}
			}else if(s[0][i]=="capa5"||s[0][i]=="datos4"||s[0][i]=="desconocido4"){
				if(s[1][i]!=null){
					e[0][i]=new Etiqueta5(s[1][i]);
				}
				if(s[2][i]!=null){
					e[1][i]=new Etiqueta5(s[2][i]);
				}
				if(s[3][i]!=null){
					e[2][i]=new Etiqueta5(s[3][i]);
				}
			}else if(s[0][i]=="capa6"||s[0][i]=="datos5"||s[0][i]=="desconocido5"){
				if(s[1][i]!=null){
					e[0][i]=new Etiqueta6(s[1][i]);
```

```
                                        }
                                        if(s[2][i]!=null){
                                                e[1][i]=new Etiqueta6(s[2][i]);
                                        }
                                        if(s[3][i]!=null){
                                                e[2][i]=new Etiqueta6(s[3][i]);
                                        }
                                }else{
                                        if(s[1][i]!=null){
                                                e[0][i]=new Etiqueta1(s[1][i]);
                                        }
                                        if(s[2][i]!=null){
                                                e[1][i]=new Etiqueta1(s[2][i]);
                                        }
                                        if(s[3][i]!=null){
                                                e[2][i]=new Etiqueta1(s[3][i]);
                                        }
                                }
                        }
                        return e;
                }
        }
```

[ANEXO 10]

[Manual de Instalación del Laboratorio]

1. Se debe conectar tres computadores en una red de área local:



2. En el computador B se debe instalar la aplicación "Gnu Gatekeeper" disponible en http://www.gnugk.org/h323download.html
    a. Una vez instalada, esta aplicación se abre automáticamente al iniciarse el sistema operativo.
    b. Puede ser reabierta en el acceso directo "Reload".
3. Instalar la aplicación "Openphone" en los computadores A y C. Los archivos necesarios para esta aplicación se encuentran en http://www.openh323.org/code.html
    a. Ejecutar el archivo "openphone.exe"
    b. Seleccionar el menú "Options"
    c. Seleccionar la opción "General"
    d. Colocar "TerminalA" y "TerminalC" en los campos "Username" de los computadores A y C respectivamente.
    e. Habilitar la casilla "Disable H.245 Tunneling". La tunelización H.245 consiste en enviar la información del canal de control (canal H.245) dentro de mensajes de señalización de llamada. Al habilitar esta opción se evita mensajes de señalización adicionales.
    f. Presionar el botón "Ok"
    g. Seleccionar el menú "Options"
    h. Seleccionar la opción "Gatekeeper"
    i. Habilitar la casilla "Use Gatekeeper"
    j. Habilitar la casilla "Discover Automatically"
    k. Presionar el botón "Ok"
4. Instalar la aplicación "Repro SIP Proxy" disponible en http://www.sipfoundry.org/repro/
    a. Para ejecutar la aplicación "Repro SIP Proxy" en modo Record-Route, ejecutar el comando: repro --record-route=sip:IPB, en donde IPB es la dirección IP del computador B.

b. Abrir el administrador del proxy por medio de la dirección http http://localhost:5080/index.html
c. Añadir un dominio en el menú “DOMAINS”. Este dominio debe ser la dirección IP del computador B.
d. Registrar los usuarios “UsuarioA” y “UsuarioC”. Para ello se debe acceder al menú “ADD USER”. El dominio debe ser el de la dirección IP del computador B.

5. Instalar en los computadores A y C el Softphone “X-Lite” disponible en http://www.xten.com/index.php?menu=Products&smenu=xlite
   a. Al iniciarse por primera vez esta aplicación, aparece el diálogo “SIP Accounts” (también es posible abrir este diálogo en la opción “SIP Accounts Settings” del menú.)
   b. Presionar el botón “Add” para añadir una cuenta
   c. Introducir el nombre de visualización en el campo “Display Name”
   d. Introducir “UsuarioA” y “UsuarioC” en el campo “User Name” de los computadores A y C respectivamente.
   e. Introducir la dirección IP del computador B en el campo “Domain”
   f. Presionar el botón “OK”