UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T J.121

(02/2002)

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

SERIE J: REDES DE CABLE Y TRANSMISIÓN DE PROGRAMAS RADIOFÓNICOS Y TELEVISIVOS, Y DE OTRAS SEÑALES MULTIMEDIOS

Sistemas interactivos para distribución de televisión digital

Protocolo de control de calidad para la distribución por la Web

Recomendación UIT-T J.121

RECOMENDACIONES UIT-T DE LA SERIE J

REDES DE CABLE Y TRANSMISIÓN DE PROGRAMAS RADIOFÓNICOS Y TELEVISIVOS, Y DE OTRAS SEÑALES MULTIMEDIOS

Recomendaciones generales	J.1–J.9
Especificaciones generales para transmisiones radiofónicas analógicas	J.10–J.19
Características de funcionamiento de los circuitos radiofónicos	J.20–J.29
Equipos y líneas utilizados para circuitos radiofónicos analógicos	J.30–J.39
Codificadores digitales para señales radiofónicas analógicas	J.40–J.49
Transmisión digital de señales radiofónicas	J.50–J.59
Circuitos para transmisiones de televisión analógica	J.60–J.69
Transmisiones de televisión analógica por líneas metálicas e interconexión con radioenlaces	J.70–J.79
Transmisión digital de señales de televisión	J.80–J.89
Servicios digitales auxiliares para transmisiones de televisión	J.90–J.99
Requisitos operacionales y métodos para transmisiones de televisión	J.100–J.109
Sistemas interactivos para distribución de televisión digital	**J.110–J.129**
Transporte de señales MPEG-2 por redes de transmisión de paquetes	J.130–J.139
Mediciones de la calidad de servicio	J.140–J.149
Distribución de televisión digital por redes locales de abonados	J.150–J.159
IPCablecom	J.160–J.179
Varios	J.180–J.199
Aplicación para televisión digital interactiva	J.200–J.209

Para más información, véase la Lista de Recomendaciones del UIT-T.

Recomendación UIT-T J.121

Protocolo de control de calidad para la distribución por la Web

Resumen

En esta Recomendación se definen los protocolos entre un servidor y un cliente para distribuir programas sonoros y de televisión en base a la Recomendación J.120, es decir, llevar a cabo distribución por la Web, por una red general que emplea el protocolo Internet (IP*)*, red que no tiene garantizada la calidad de servicio (QoS) y en la cual pueden producirse errores de datos o pérdida de paquetes. El uso de estos protocolos permite mejorar la calidad.

Se utiliza un esquema en el que se envía un informe de calidad del cliente al servidor, y éste realiza distribuciones óptimas de datos según dicho informe.

En esta Recomendación se definen clases de parámetros que el cliente debe informar al servidor, y un protocolo de transmisión que se utiliza para enviar el informe del cliente al servidor. Esta Recomendación define además la transmisión de parámetros de soporte que facilitan al servidor el análisis del informe del cliente.

Orígenes

La Recomendación UIT-T J.121, preparada por la Comisión de Estudio 9 (2001-2004) del UIT-T, fue aprobada por el procedimiento de la Resolución 1 de la AMNT el 13 de febrero de 2002.

PREFACIO

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Asamblea Mundial de Normalización de las Telecomunicaciones (AMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución 1 de la AMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

PROPIEDAD INTELECTUAL

La UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT no ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB.

ÍNDICE

Recomendación UIT-T J.121

Protocolo de control de calidad para la distribución por la Web

1 Alcance

La respuesta del servidor después de recibir el informe del cliente está fuera del alcance de esta Recomendación.

2 Referencias

Las siguientes Recomendaciones del UIT-T y otras referencias contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y otras referencias son objeto de revisiones por lo que se preconiza que los usuarios de esta Recomendación investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y otras referencias citadas a continuación. Se publica periódicamente una lista de las Recomendaciones UIT-T actualmente vigentes.

2.1 Referencias normativas

– 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 J.120 (2000), *Distribución de programas de radiodifusión sonora y de televisión por la red del protocolo Internet*.

2.2 Referencias informativas

[1] IETF RFC 1889 (1996), *RTP*: *A Transport Protocol for Real-Time Applications.*

3 Términos y definiciones

En esta Recomendación se definen los términos siguientes.

3.1 protocolo de transporte en tiempo real (RTP, *real-time transport protocol*): Protocolo de transporte para aplicaciones en tiempo real, definido en H.225.0.

3.2 protocolo de control de transporte en tiempo real (RTCP, *RTP control protocol*): Protocolo de control de paquetes RTP, definido en H.225.0.

3.3 distribución por la Web (webcasting): La distribución por la Web se define en la Rec. UIT-T J.120. Distribución de programas de radiodifusión sonora y de televisión por la red de protocolo Internet.

4 Abreviaturas y convenios

4.1 Abreviaturas

En esta Recomendación se utilizan las siguientes siglas.

IP Protocolo Internet (*Internet protocol*)

RTCP Protocolo de control de transporte en tiempo real (*RTP control protocol*)

RTP Protocolo de transporte en tiempo real (*real-time transport protocol*)

TCP Protocolo de control de transmisión (*transmission control protocol*)

UDP Protocolo de datagrama de usuario (*user datagram protocol*)

4.2 Convenios

Al implementar esta Recomendación, se tendrá en cuenta que la obligatoriedad de la especificación se expresa mediante el verbo modal "DEBER" (verbo modal inglés *must)* o un verbo en tiempo futuro con valor imperativo, por ejemplo "*expirará*" (verbo modal inglés *shall*) o el adjetivo "OBLIGATORIO" (*required*).

A continuación, se indican otras expresiones que se aplican a determinados requisitos con significado de obligación o posibilidad.

"DEBER" (*MUST*)	Este verbo (u otros con significado de obligación, como "tener que/de", "haber que/de") o un verbo en tiempo futuro con valor imperativo o el adjetivo OBLIGATORIO indican que se tiene la obligación de hacer lo que expresa la especificación.
"NO DEBER" (*MUST NOT*)	La negación indica que se prohíbe hacer lo que expresa la especificación.
"DEBERÍA" (*SHOULD*)	El modo condicional de estos verbos, u otros verbos con significado de conveniencia (aconsejar, recomendar, ser conveniente) o el adjetivo RECOMENDADO *(recommended*) indica que puede haber motivos fundados para que en determinadas circunstancias no se haga cierta cosa, pero que antes de hacer algo diferente, es preciso entender todas las consecuencias y sopesar el caso.
"NO DEBERÍA" (*SHOULD NOT*)	La negación indica la posibilidad de que haya motivos fundados para que en determinadas circunstancias la acción sea aceptable e incluso útil, pero que antes de realizarla es preciso entender todas las consecuencias y sopesar el caso.
"PODER" (*MAY*)	Este u otros verbos que indican posibilidad o probabilidad (deber de,) o el adjetivo OPCIONAL (*optional*) se refieren a la libertad de elegir. Un proveedor puede incluir un elemento porque el mercado lo exige o porque mejora el producto, mientras que otro puede optar por no hacerlo.

5 Definición del sistema

En la figura 1 se ilustra el sistema en cuestión y su flujo de señales entre el servidor y los clientes. En este esquema, el informe del emisor (SR, *sender report*) y el informe del receptor (RR, *receiver report*) de RTCP se utilizan para intercambiar información entre el servidor y el cliente. El RTCP se fundamenta en la transmisión periódica de paquetes de control a todos los participantes en la sesión por el mismo mecanismo de distribución que se aplica a los paquetes de datos de los medios. El protocolo subyacente proporcionará la multiplexación de los paquetes de datos y de control, por ejemplo utilizando números de puerto separados con UDP.

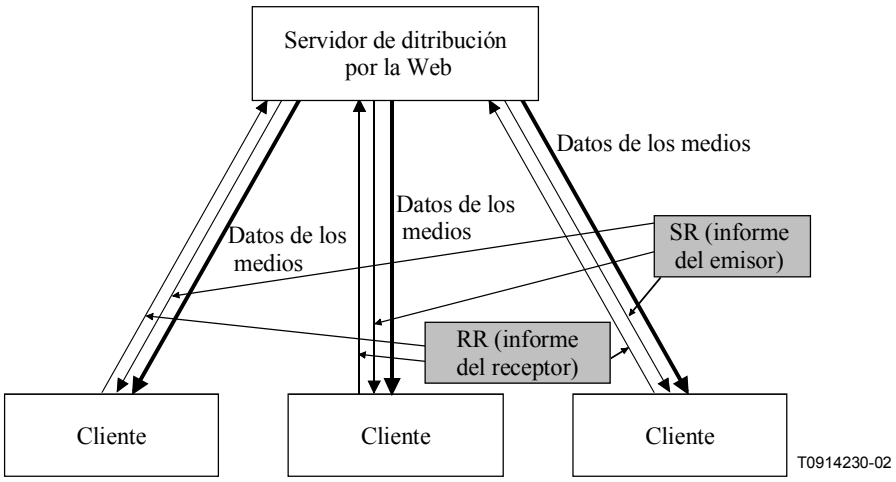


Figura 1/J.121 – Diagrama a bloques del esquema de control de calidad para la distribución por la Web

5.1 Procedimiento seguido por el servidor

5.1.1 Transmisión de la información de soporte de mediciones

En primer término, el servidor transmite información de soporte de mediciones que incluye un tiempo de referencia e información relativa a los paquetes transmitidos al cliente. El cliente se sirve de esta información de mediciones para hacer su informe y el servidor para analizarlo. Aunque no se define en esta Recomendación el intervalo de transmisión para esta información, en la cláusula 7 se describe el intervalo de transmisión recomendado. Esta información se transmite en un paquete SR (informe del emisor) definido en 6.1. El paquete SR se envía con un número de puerto independiente del de los datos de los medios (vídeo/audio). El número de puerto lo proporciona el proceso de iniciación de la sesión del protocolo J.120, y es el número que sigue al número de puerto RTP de los datos de los medios (es decir, este número más uno).

5.1.2 Recepción del informe del cliente

Es necesario que el servidor siempre esté en un estado que permita recibir el paquete del informe del cliente.

5.1.3 Distribución de los datos de los medios

El servidor debe llevar a cabo una distribución óptima de los datos de los medios (audio/vídeo) según los resultados del análisis del paquete del informe del cliente. El método que efectivamente se utilice para controlar la distribución es una cuestión de implementación y está fuera del alcance de esta Recomendación.

5.2 Procedimiento seguido por el cliente

5.2.1 Supervisión de la calidad de transmisión

El cliente observa en qué estado se reciben los datos de los medios y mide la calidad de transmisión. Los parámetros que se deben medir se describen en 6.2. Para hacer la medición, el cliente calculará estos parámetros con referencia a la precitada información de soporte de las mediciones transmitida desde el servidor.

5.2.2 Transmisión del informe del cliente al servidor

La calidad de transmisión medida en el cliente se transmite al servidor como un informe del cliente. Este informe se transmite como un paquete RR (informe del receptor) definido en 6.2. Se envía este paquete usando el número de puerto asignado al inicio de la sesión del protocolo J.120.

6 Definición de un paquete de informe

6.1 Informe del emisor (SR)

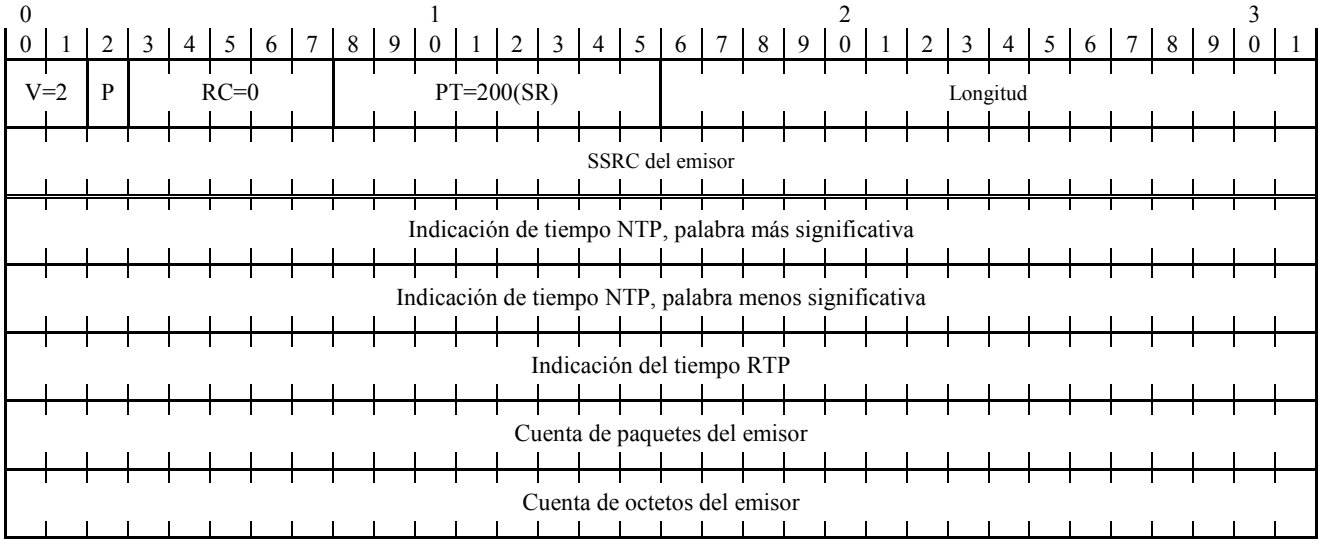


Figura 2/J.121 – Formato de paquete del informe del emisor (SR)

El paquete del informe del emisor consta de dos secciones. La primera sección, el encabezamiento, tiene 8 octetos de longitud. Los campos tienen el siguiente significado:

versión (V, *version*): 2 bits. Identifica la versión de RTP, que es la misma en los paquetes RTCP como en los paquetes de datos RTP. La versión definida por esta Recomendación es dos (2).

relleno (P, *padding*): 1 bit. Si el bit de relleno está puesto a 1, este paquete RTCP contiene al final algunos octetos de relleno adicionales que no son parte de la información de control. El último octeto del relleno es una cuenta de la cantidad de octetos que deben ser pasados por alto. Algunos algoritmos de criptación con tamaños de bloque fijos pueden requerir relleno. En un paquete RTCP compuesto, sólo debe requerirse relleno en el último paquete individual porque el paquete compuesto es criptado como un todo.

cuenta de informes de recepción (RC, *reception report count*): 5 bits. El número de bloques de informe de recepción incluidos en este paquete. El valor es siempre cero.

tipo de paquete (PT, *packet type*): 8 bits. Contiene la constante 200 para identificarlo como un paquete SR de RTCP.

longitud: 16 bits. La longitud de este paquete RTCP en palabras de 32 bits menos uno, incluyendo el encabezamiento y eventual relleno. (El desplazamiento por la resta de una unidad hace que cero sea una longitud válida y evita la posibilidad de un bucle infinito durante la exploración de un paquete RTCP compuesto, mientras que el conteo de palabras de 32 bits evita una comprobación de validez para un múltiplo de 4.)

fuente de sincronización (SSRC, *synchronization source*): 32 bits. El identificador de la fuente de sincronización para el originador de este paquete SR.

La segunda sección, la información del emisor, tiene 20 octetos de longitud y está presente en todos los paquetes de informe del emisor. Contiene un resumen de las transmisiones de datos desde este emisor. Los campos tienen el siguiente significado:

indicación de tiempo NTP: 64 bits. Indica la hora de referencia de envío de este informe de modo que pueda usarse en combinación con las indicaciones de tiempo devueltas en los informes de recepción de otros receptores para medir el tiempo de propagación de ida y vuelta para esos receptores. Los receptores deben considerar que la exactitud de medición de la indicación de tiempo

puede estar limitada a mucho menos que la correspondiente a la resolución de la indicación de tiempo del protocolo de tiempo de la red (NTP). No se indica la incertidumbre de medición de la indicación de tiempo ya que puede desconocerse. Un emisor que puede llevar el tiempo transcurrido pero no tiene idea del tiempo de referencia, puede emplear en su lugar el tiempo transcurrido desde que entró en la sesión. Se supone que este tiempo es menor a 68 años, de manera que el bit superior será cero. Se puede utilizar el reloj de muestreo para estimar el tiempo de referencia transcurrido. Un emisor que no tiene idea del tiempo de referencia o del tiempo transcurrido puede fijar la indicación de tiempo NTP a cero.

indicación de tiempo RTP: 32 bits. Corresponde al mismo tiempo que la indicación de tiempo NTP (antes mencionada), pero se expresa en las mismas unidades y con el mismo desplazamiento aleatorio que las indicaciones de tiempo RTP en los paquetes de datos. Se puede utilizar esta correspondencia para la sincronización intramedios e intermedios de las fuentes cuyas indicaciones de tiempo NTP están sincronizadas, y puede ser utilizada por los receptores independientes de los medios para estimar la frecuencia de reloj RTP nominal. Obsérvese que en la mayoría de los casos esta indicación de tiempo no será igual a la indicación de tiempo RTP en cualquier paquete de datos adyacente. Más bien, se calcula a partir de la indicación de tiempo NTP correspondiente utilizando la relación entre el contador de indicaciones de tiempo RTP y el tiempo real que se mantiene por la comprobación periódica del tiempo de referencia en un instante de muestreo.

cuenta de paquetes del emisor: 32 bits. Número total de paquetes de datos RTP transmitidos por el emisor desde el comienzo de la transmisión hasta el momento en que se genera este paquete SR. La cuenta se reinicia si el emisor cambia su identificador SSRC.

cuenta de octetos del emisor: 32 bits. Número total de octetos de cabida útil (es decir sin incluir encabezamiento ni relleno) transmitido por el emisor en paquetes de datos RTP desde el comienzo de la transmisión hasta el momento en que se genera este paquete SR. La cuenta se reinicia si el emisor cambia su identificador SSRC. Se puede utilizar este campo para estimar la tasa promedio de datos de cabida útil.

6.2 Informe del receptor (RR)

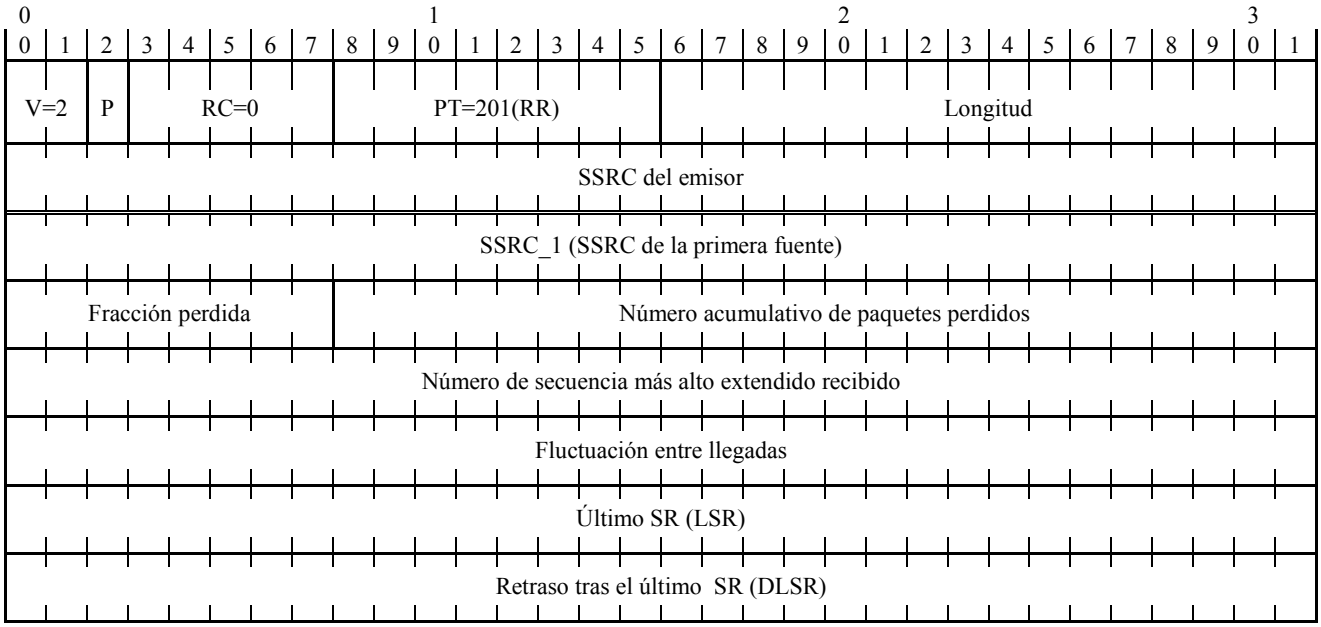


Figura 3/J.121 – Formato de paquete del informe del receptor (RR)

El paquete del informe del receptor consta de dos secciones. La primera sección, el encabezamiento con una longitud de 8 octetos, es la misma que la del paquete SR excepto que el campo tipo de paquete contiene la constante 201. Los campos se definen en 6.1.

La segunda sección contiene cero o más bloques de informe de recepción según el número de otras fuentes percibidas por este receptor desde el último informe. Cada bloque de informe de recepción aporta estadísticos sobre la recepción de paquetes RTP desde una única fuente de sincronización. Los receptores no traspasan estadísticos cuando una fuente cambia su identificador SSRC debido a una colisión. Estos estadísticos son:

SSRC_n (identificador de la fuente): 32 bits. Identificador SSRC de la fuente a la cual pertenece la información en este bloque de informe de recepción.

Fracción perdida: 8 bits. Fracción de los paquetes de datos RTP recibidos de la fuente SSRC_n perdidos desde que se envió el anterior paquete SR o RR, expresada como un número de coma fija con la coma binaria en el extremo izquierdo del campo. (Es equivalente a tomar la parte entera después de multiplicar la fracción perdida por 256.) Esta fracción se define como el número de paquetes perdidos dividido por el número de paquetes esperados, según se define en el siguiente párrafo. Si la pérdida es negativa debido a la existencia de duplicados, la fracción perdida se fija a cero. Obsérvese que un receptor no puede indicar si se perdieron algunos paquetes después del último recibido, ni que no se expedirá ningún bloque de informe de recepción para una fuente si se han perdido todos los paquetes de la misma enviados durante el último intervalo de informes.

Número acumulativo de paquetes perdidos: 24 bits. Número total de paquetes de datos RTP procedentes de la fuente SSRC_n que se han perdido desde el comienzo de la recepción. Este número se define como el número de paquetes esperados menos el número de paquetes realmente recibidos, donde el número de paquetes recibidos incluye todo paquete retrasado o duplicado. Por lo tanto los paquetes que llegan tarde no se cuentan como perdidos, y la pérdida puede ser negativa si hay duplicados. El número de paquetes esperados se define como el último número de secuencia extendido recibido, según se define más adelante, menos el número de secuencia inicial recibido. Esto puede calcularse como se indica en el apéndice III.

Número de secuencia más alto extendido recibido: 32 bits. Los 16 bits inferiores contienen el número de secuencia más alto recibido en un paquete de datos RTP procedente de la fuente SSRC_n y los 16 bits más significativos extienden ese número de secuencia con la correspondiente cuenta de los ciclos de los números de la secuencia, que puede mantenerse según el algoritmo del apéndice II. Obsérvese que diferentes receptores dentro de la misma sesión generarán diferentes extensiones del número de secuencia si sus tiempos de comienzo difieren significativamente.

Fluctuación entre llegadas: 32 bits. Estimación de la varianza estadística del tiempo entre llegadas de paquetes de datos RTP, medido en unidades de indicación de tiempo y expresado como un número entero sin signo. La fluctuación entre llegadas J se define como la desviación media (valor absoluto alisado) de la diferencia D en el espaciamiento de paquetes en el receptor en comparación con la obtenida en el emisor para un par de paquetes. Según se muestra en la ecuación más adelante, esto equivale a la diferencia en el "tiempo de tránsito relativo" para los dos paquetes; este tiempo es la diferencia entre la indicación de tiempo RTP de un paquete y el reloj del receptor en el momento de recepción, medida en las mismas unidades.

Si *Si* es la indicación de tiempo RTP del paquete *i*, y *Ri* es el tiempo de recepción en unidades de indicación de tiempo RTP para el paquete *i*, entonces para los dos paquetes *i* y *j*, *D* puede expresarse como:

$$D(i+j) = (Rj - Ri) - (Sj - Si) = (Rj - Sj) - (Ri - Si)$$

La fluctuación entre llegadas se calcula de manera continua conforme se recibe cada paquete de datos *i* desde la fuente SSRC_n, y se utiliza esta diferencia *D* para ese paquete y para el anterior paquete *i* – 1 en el orden de llegada (no necesariamente en secuencia), según la fórmula:

$$J = J + \frac{|D(i-1,i)| - J}{16}$$

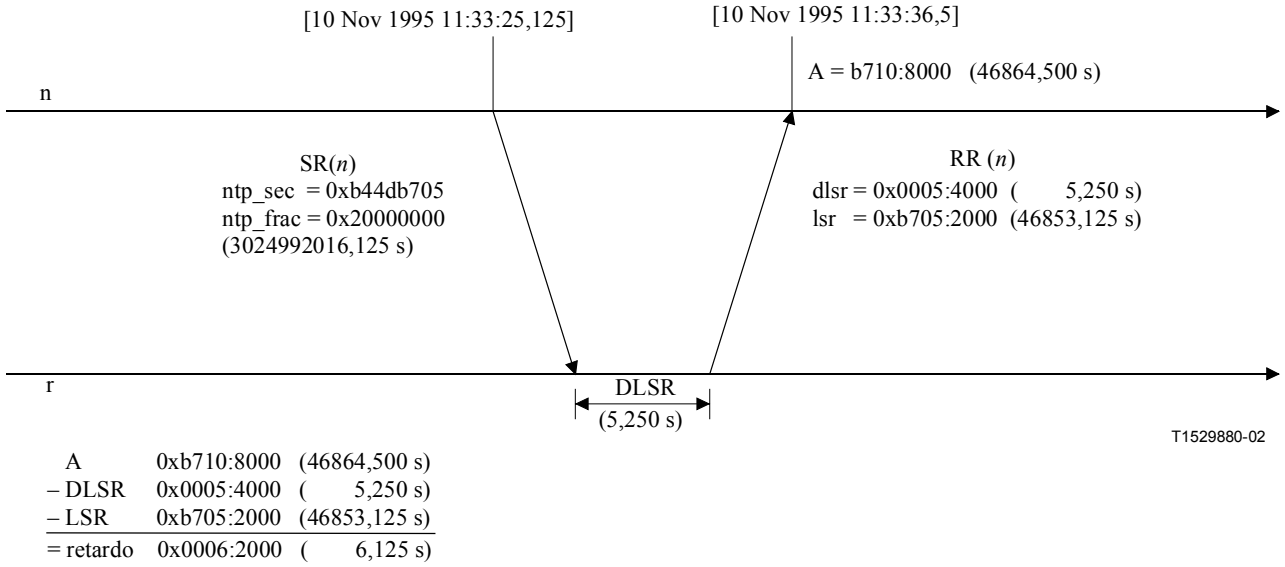
Siempre que se expide un informe de recepción, se muestrea el valor actual de *J*.

El cálculo de la fluctuación entre llegadas se prescribe en este caso pues permite que monitores independientes de los perfiles realicen interpretaciones válidas de informes provenientes de diferentes implementaciones. Este algoritmo es el óptimo estimador de primer nivel y el parámetro de ganancia 1/16 da una buena razón de reducción de ruido en tanto que mantiene una tasa razonable de convergencia. En el apéndice V se presenta un ejemplo de implementación.

Última indicación de tiempo SR (LSR, *last SR time-stamp*): 32 bits. Los 32 bits centrales de un total de 64 en la indicación de tiempo NTP (como se explica en A.4/H.225.0: orden de los octetos, alineación y formato de tiempo) recibidos como parte del paquete de informe del emisor (SR) RTCP más reciente procedente de la fuente SSRC_n. Si aún no se ha recibido ningún paquete SR, el campo se fija a cero.

Retraso tras el último SR (DLSR, *delay since last SR*): 32 bits. Retraso, expresado en unidades de 1/65536 segundos, entre la recepción del último paquete SR procedente de la fuente SSRC_n y el envío de este bloque de informe de recepción. Si aún no se ha recibido ningún paquete SR procedente de SSRC_n, el campo DLSR se fija a cero.

Sea SSRC_r el receptor que expide este informe de receptor. El SSRC_n fuente puede calcular el retraso de propagación de ida y vuelta al SSRC_r registrando el tiempo A en que se recibe este bloque de informe de recepción. Se calcula el tiempo total de viaje de ida y vuelta A – LSR usando el último campo de indicación de tiempo SR (LSR) y luego restando este campo de manera que el retraso de propagación de viaje de ida y vuelta sea (A – LSR – DLSR). Esto se ilustra en la figura 4. Se puede utilizar como una medida aproximada de la distancia a un grupo de receptores, aunque algunos enlaces tienen retrasos muy asimétricos.



A	0xb710:8000	(46864,500 s)
– DLSR	0x0005:4000	(5,250 s)
– LSR	0xb705:2000	(46853,125 s)
= retardo	0x0006:2000	(6,125 s)

Figura 4/J.121 – Ejemplo de cálculo del tiempo de ida y vuelta

7 Intervalo de transmisión RTCP

Las características del RTP permiten que una aplicación se adapte automáticamente al tamaño de las sesiones que comprendan de unos pocos a miles de participantes. Por ejemplo, en una audioconferencia el tráfico de datos es intrínsecamente autolimitativo porque sólo una o dos personas hablarán a la vez, por lo que, con distribución multidifusión, la velocidad de datos en cualquier enlace permanece relativamente constante con independencia del número de participantes. No obstante, el tráfico de control no es autolimitativo. Si los informes de recepción de cada

participante se enviaran a una velocidad constante, el tráfico de control aumentaría linealmente con el número de participantes. Por consiguiente, debe disminuirse la velocidad.

Para cada sesión, se supone que el tráfico de datos está sujeto a un límite global denominado la "anchura de banda de la sesión" que se divide entre los participantes. Es posible que sea la red la que reserve esta anchura de banda y la que imponga el límite, o podría simplemente compartirse en forma razonable. Puede elegirse la anchura de banda de la sesión con referencia a algún costo o a un conocimiento previo de la anchura de banda de la red disponible para la sesión. Es hasta cierto grado independiente de la codificación de los medios, pero la elección de la codificación puede verse limitada por la anchura de banda de la sesión. Se espera que el parámetro anchura de banda de la sesión sea asignado por una aplicación de gestión de las sesiones cuando se invoca una aplicación de medios, pero las aplicaciones de medios pueden fijar además un valor por defecto basado en la anchura de banda de datos de un solo emisor para la codificación seleccionada para la sesión. La aplicación también puede imponer los límites de anchura de banda con fundamento en las reglas de alcance de la multidifusión o en otros criterios.

Los cálculos de anchura de banda para el tráfico de control y de datos incluyen los protocolos de transporte de capa inferior y de red (por ejemplo UDP e IP) ya que eso es lo que el sistema de reserva de recursos necesitaría saber. También puede esperarse que la aplicación sepa cuáles de estos protocolos están en uso. No están incluidos en el cálculo los encabezamientos del nivel del enlace ya que el paquete se encapsulará con diferentes encabezamientos de nivel de enlace según se desplace.

El tráfico de control se debe limitar a una fracción pequeña y conocida de la anchura de banda de la sesión: pequeña para no afectar la función principal del protocolo de transporte, que es transportar los datos, y conocida para que se pueda incluir el tráfico de control en la especificación de anchura de banda referente a un protocolo de reserva de recursos, y para que cada participante pueda calcular independientemente su proporción. Se sugiere que la fracción de la anchura de banda de sesión atribuida a RTCP se fije en 5%. Si bien el valor de ésta y otras constantes no es crítico en el cálculo de los intervalos, todos los participantes en la sesión deben utilizar los mismos valores para que se calcule el mismo intervalo. Por consiguiente, estas constantes deben fijarse para un perfil particular.

El algoritmo descrito en el apéndice IV está diseñado para alcanzar las metas esbozadas anteriormente. Calcula el intervalo entre los envíos de paquetes RTCP compuestos para dividir la anchura de banda del tráfico de control disponible entre los participantes. Esto permite a una aplicación ofrecer una respuesta más rápida para las sesiones cortas en las que, por ejemplo, la identificación de todos los participantes es importante, no obstante lo cual se adapta automáticamente a sesiones extensas. El algoritmo tiene las siguientes características:

- Se atribuye colectivamente a los emisores al menos 1/4 de la anchura de banda del tráfico de control a fin de que en las sesiones con un gran número de receptores pero con un pequeño número de emisores, los nuevos participantes que se adhieran puedan recibir más rápidamente el CNAME para los sitios emisores.
- Se requiere que el intervalo calculado entre paquetes RTCP sea mayor que un mínimo de 5 segundos para evitar que ráfagas de paquetes RTCP excedan la anchura de banda permitida cuando el número de participantes es pequeño y el tráfico no está alisado según la ley de los grandes números.
- El intervalo entre paquetes RTCP varía aleatoriamente sobre la gama [0,5, 1,5] veces el intervalo calculado para evitar la sincronización involuntaria de todos los participantes. El primer paquete RTCP enviado después de la adhesión a una sesión también se retrasa con una variación aleatoria igual a la mitad del intervalo RTCP mínimo, en caso de que la aplicación comience en múltiples sitios simultáneamente, por ejemplo empiece por un anuncio de la sesión.

- Se hace una estimación dinámica del tamaño promedio del paquete RTCP compuesto, incluidos todos los paquetes recibidos y enviados, para adaptarse automáticamente a los cambios en la cantidad de información de control transportada.

Este algoritmo puede utilizarse para sesiones en las cuales todos los participantes están autorizados para transmitir. En ese caso, el parámetro anchura de banda de la sesión es el producto de la anchura de banda de cada emisor multiplicada por el número de participantes y la anchura de banda RTCP es el 5% de ese resultado.

Apéndice I

Algoritmos

Se proporcionan ejemplos de código escrito en el lenguaje de programación C para distintos aspectos de los algoritmos RTP del emisor y el receptor. Puede haber otros métodos de implementación que sean más rápidos en determinados entornos operativos o que tengan otras ventajas. Estas notas de implementación son exclusivamente informativas y tienen por objeto aclarar la especificación del RTP.

Se utilizan las siguientes definiciones para todos los ejemplos; por razones de claridad y brevedad, las definiciones de estructuras sólo son válidas para las arquitecturas de 32 bits en las que el octeto más significativo ocupa la primera posición (conocidas por "*big-endian*"). Se supone que los campos de bits están llenos completamente con el bit más significativo en primera posición, sin relleno adicional. Se requerirían modificaciones para construir una implementación portable.

```
/*
   * rtp.h  --  RTP header file (RFC XXXX)
   */
#include <sys/types.h>

/*
   * The type definitions below are valid for 32-bit architectures and
   * may have to be adjusted for 16- or 64-bit architectures.
   */
typedef unsigned char  u_int8;
typedef unsigned short u_int16;
typedef unsigned int   u_int32;
typedef short int16;

/*
   * Current protocol version.
   */
#define RTP_VERSION    2

#define RTP_SEQ_MOD (1<<16)
#define RTP_MAX_SDES 255               /* maximum text length for SDES */

typedef enum {
     RTCP_SR   = 200,
     RTCP_RR   = 201,
     RTCP_SDES = 202,
     RTCP_BYE  = 203,
     RTCP_APP  = 204
} rtcp_type_t;
```

```
typedef enum {
     RTCP_SDES_END   = 0,
     RTCP_SDES_CNAME = 1,
     RTCP_SDES_NAME  = 2,
     RTCP_SDES_EMAIL = 3,
     RTCP_SDES_PHONE = 4,
     RTCP_SDES_LOC   = 5,
     RTCP_SDES_TOOL  = 6,
     RTCP_SDES_NOTE  = 7,
     RTCP_SDES_PRIV  = 8
} rtcp_sdes_type_t;

/*
   * RTP data header
   */
typedef struct {
     unsigned int version:2;            /* protocol version */
     unsigned int p:1;                  /* padding flag */
     unsigned int x:1;                  /* header extension flag */
     unsigned int cc:4;                 /* CSRC count */
     unsigned int m:1;                  /* marker bit */
     unsigned int pt:7;                 /* payload type */
     u_int16 seq;                       /* sequence number */
     u_int32 ts;                        /* time-stamp */
     u_int32 ssrc;                      /* synchronization source */
     u_int32 csrc[1];                   /* optional CSRC list */
} rtp_hdr_t;

/*
   * RTCP common header word
   */
     typedef struct {
     unsigned int version:2;            /* protocol version */
     unsigned int p:1;                  /* padding flag */
     unsigned int count:5;              /* varies by packet type */
     unsigned int pt:8;                 /* RTCP packet type */
     u_int16 length;                    /* pkt len in words, w/o this word */
} rtcp_common_t;

/*
   * Big-endian mask for version, padding bit and packet type pair
   */
#define RTCP_VALID_MASK (0xc000 | 0x2000 | 0xfe)
#define RTCP_VALID_VALUE ((RTP_VERSION << 14) | RTCP_SR)

/*
   * Reception report block
   */
typedef struct {
     u_int32 ssrc;                      /* data source being reported */
     unsigned int fraction:8;           /* fraction lost since last SR/RR */
     int lost:24;                       /* cumul. no. pkts lost (signed!) */
     u_int32 last_seq;                  /* extended last seq. no. received */
     u_int32 jitter;                    /* inter-arrival jitter */
     u_int32 lsr;                       /* last SR packet from this source */
     u_int32 dlsr;                      /* delay since last SR packet */
} rtcp_rr_t;

/*
   * SDES item
   */
typedef struct {
     u_int8 type;                       /* type of item (rtcp_sdes_type_t) */
     u_int8 length;                     /* length of item (in octets) */
```

```
     char data[1];                        /* text, not null-terminated */
     } rtcp_sdes_item_t;

/*
   * One RTCP packet
   */
typedef struct {
     rtcp_common_t common;                /* common header */
     union {
       /* sender report (SR) */
       struct {
            u_int32 ssrc;                   /* sender generating this report */
            u_int32 ntp_sec;                /* NTP time-stamp */
            u_int32 ntp_frac;
            u_int32 rtp_ts;                 /* RTP time-stamp */
            u_int32 psent;                  /* packets sent */
            u_int32 osent;                  /* octets sent */
            rtcp_rr_t rr[1];                /* variable-length list */
       sr;

       /* reception report (RR) */
       struct {
            u_int32 ssrc;                /* receiver generating this report */
            rtcp_rr_t rr[1];             /* variable-length list */
       } rr;

       /* source description (SDES) */
       struct rtcp_sdes {
            u_int32 src;                 /* first SSRC/CSRC */
            p_sdes_item_t item[1];       /* list of SDES items */
       sdes;

       /* BYE */
       struct {
            u_int32 src[1];              /* list of sources */
            /* can't express trailing text for reason */
       } bye;
     } r;
} rtcp_t;

typedef struct rtcp_sdes rtcp_sdes_t;

/*
   * Per-source state information
   */
typedef struct {
     u_int16 max_seq;                     /* highest seq. number seen */
     u_int32 cycles;                      /* shifted count of seq. number cycles
*/
     u_int32 base_seq;                    /* base seq number */
     u_int32 bad_seq;                     /* last 'bad' seq number + 1 */
     u_int32 probation;                   /* sequ. packets till source is valid */
     u_int32 received;                    /* packets received */
     u_int32 expected_prior;              /* packet expected at last interval */
     u_int32 received_prior;              /* packet received at last interval */
     u_int32 transit;                     /* relative trans time for prev pkt */
     u_int32 jitter;                      /* estimated jitter */
     /* ... */
} source;
```

Apéndice II

Comprobaciones de la validez del encabezamiento RTP

Un receptor RTP debe comprobar la validez del encabezamiento RTP de los paquetes entrantes ya que podrían estar criptados o proceder de una aplicación diferente que resulta estar mal direccionada. De igual manera, si está activada la criptación, se necesita la comprobación de la validez del encabezamiento para verificar que los paquetes entrantes se han descriptado correctamente, aunque un fallo en la comprobación de la validez del encabezamiento (por ejemplo, tipo de cabida útil desconocido) no necesariamente indicará un fallo de la decripción. En paquetes de datos RTP de una fuente que se percibe por primera vez, sólo es posible efectuar comprobaciones de validez limitadas:

- El campo versión RTP debe ser 2.
- El tipo de cabida útil debe ser conocido, en particular no debe ser igual a SR ni a RR.
- Si el bit P está puesto a 1, el último octeto del paquete debe contener una cuenta válida de octetos, en particular, inferior a la longitud total del paquete menos el tamaño del encabezamiento.
- El bit X debe ser cero si el perfil no especifica que puede utilizarse el mecanismo de extensión del encabezamiento. De lo contrario, el campo longitud de la extensión debe ser inferior al tamaño total del paquete menos la longitud fija del encabezamiento y el relleno.
- La longitud del paquete debe ser coherente con la cuenta de CSRC (CC, *CSRC count*) y con el tipo de cabida útil (si las cabidas útiles tienen una longitud conocida).

Las tres últimas comprobaciones son algo complejas y no siempre posibles, lo que deja únicamente las dos primeras que totalizan sólo unos pocos bits. Si el identificador SSRC en el paquete ya se ha recibido antes, el paquete es probablemente válido y se puede hacer una validación adicional verificando si el número de secuencia está en la gama esperada. Si el identificador SSRC no se ha recibido anteriormente, se puede considerar que los paquetes de datos que llevan ese identificador no tienen validez hasta que un número pequeño de ellos lleguen con números de secuencia consecutivos.

La rutina `update_seq` que se muestra más adelante asegura que una fuente será declarada válida sólo después de que se hayan recibido paquetes `MIN_SEQUENTIAL` en secuencia. Valida además el número de secuencia `seq` de un paquete últimamente recibido y actualiza el estado de la secuencia para la fuente del paquete en la estructura a que apunta `s`.

Cuando se percibe una nueva fuente por primera vez, o sea, cuando su identificador SSRC no está en el cuadro y se atribuye el estado por fuente para tal fin, se debe fijar `s->probation` al número de paquetes secuenciales requeridos antes de declarar válida una fuente (parámetro `MIN_SEQUENTIAL`) y `s->max_seq` inicializado a `seq-1. s->probation` marca la fuente como aún no válida por lo que se puede descartar el estado después de un corto periodo de temporización en lugar de uno largo.

Después de que una fuente se considera válida, el número de secuencia se considerará válido si no está a más de `MAX_DROPOUT` delante de `s->max_seq` ni a más de `MAX_MISORDER` detrás. Si el nuevo número de secuencia está delante `max_seq` modulo la gama de números de secuencia RTP (16 bits), pero es menor que `max_seq`, se ha completado un ciclo y se incrementa la cuenta (desplazada) de los ciclos del número de secuencia. Se devuelve un valor de uno para indicar un número de secuencia válido.

En otro caso, se devuelve el valor cero para indicar que fracasó la validación, y se almacena el número de secuencia incorrecto. Si el siguiente paquete recibido lleva el número de secuencia superior que le sigue, se considera que es el comienzo válido de una nueva secuencia de paquetes causada probablemente por una interrupción prolongada o un rearranque de la fuente. Puesto que

quizá falten múltiples ciclos completos de números de secuencia, se reinician los estadísticos de pérdida de paquetes.

Se muestran los valores característicos para los parámetros, basados en un tiempo máximo fuera de orden de 2 segundos a 50 paquetes/segundo y una interrupción máxima de 1 minuto. El parámetro interrupción MAX_DROPOUT debe ser una pequeña fracción del espacio de números de secuencia de 16 bits para tener una probabilidad razonable de que los nuevos números de secuencia después de un rearranque no caerán en la gama aceptable para los números de secuencia desde antes del rearranque.

```
void init_seq(source *s, u_int16 seq)
{
     s->base_seq = seq - 1;
     s->max_seq = seq;
     s->bad_seq = RTP_SEQ_MOD + 1;
     s->cycles = 0;
     s->received = 0;
     s->received_prior = 0;
     s->expected_prior = 0;
     /* other initialization */
}

int update_seq(source *s, u_int16 seq)
{
     u_int16 udelta = seq - s->max_seq;
     const int MAX_DROPOUT = 3000;
     const int MAX_MISORDER = 100;
     const int MIN_SEQUENTIAL = 2;
     /*
     * Source is not valid until MIN_SEQUENTIAL packets with
     * sequential sequence numbers have been received.
     */
     if (s->probation) {
     /* packet is in sequence */
       if (seq == s->max_seq + 1) {
          s->probation--;
          s->max_seq = seq;
          if (s->probation == 0) {
           init_seq(s, seq);
           s->received++;
           return 1;
          }
       }
       else {
          s->probation = MIN_SEQUENTIAL - 1;
          s->max_seq = seq;
       }
       return 0;
     }
     else if (udelta < MAX_DROPOUT) {
       /* in order, with permissible gap */
       if (seq < s->max_seq) {
       /*
       * Sequence number wrapped - count another 64K cycle.
       */
          s->cycles += RTP_SEQ_MOD;
       }
       s->max_seq = seq;
     } else if (udelta <= RTP_SEQ_MOD - MAX_MISORDER) {
       /* the sequence number made a very large jump */
       if (seq == s->bad_seq) {
          /*
```

```
            * Two sequential packets -- assume that the other side
            * restarted without telling us so just re-sync
            * (i.e., pretend this was the first packet).
            */
            init_seq(s, seq);
        }
        else {
            s->bad_seq = (seq + 1) & (RTP_SEQ_MOD-1);
            return 0;
        }
    } else {
        /* duplicate or reordered packet */
    }
    s->received++;
    return 1;
}
```

La comprobación de la validez se puede reforzar exigiendo más de dos paquetes en secuencia. Las desventajas son que se descartará un mayor número de paquetes iniciales y que tasas altas de pérdida de paquetes podrían impedir la validación. No obstante, debido a que la validación del encabezamiento RTCP es relativamente fuerte, si se recibe un paquete RTCP desde una fuente antes de los paquetes de datos, se podría ajustar la cuenta de modo que sólo se exijan dos paquetes en secuencia. Si puede tolerarse la pérdida de datos inicial durante unos pocos segundos, una aplicación podría optar por descartar todos los paquetes de datos de una fuente hasta que se haya recibido de la misma un paquete RTCP válido.

En función de la aplicación y la codificación, el conocimiento adicional acerca del formato de la cabida útil se puede incluir en los algoritmos para hacer una validez más completa. Para tipos de cabida útil en los que el incremento de la indicación de tiempo es el mismo para todos los paquetes, los valores de las indicaciones de tiempo pueden predecirse a partir del paquete anterior recibido de la misma fuente utilizando la diferencia de número de secuencia (suponiendo que no ha cambiado el tipo de cabida útil).

Es posible una comprobación fuerte del tipo "trayecto rápido" ya que es muy probable que los cuatro primeros octetos en el encabezamiento de un paquete de datos RTP últimamente recibido sean exactamente los mismos que los del paquete anterior de la misma fuente de sincronización (SSRC), excepto que el número de secuencia habrá aumentado en uno.

De igual manera, se puede utilizar una memoria caché de un solo asiento para consultas de SSRC más rápidas en aplicaciones en las que los datos se reciben generalmente sólo de una fuente en cada momento.

Apéndice III

Determinación del número de paquetes RTP esperados y perdidos

Para calcular las tasas de pérdida de paquetes, es necesario saber el número de paquetes esperados y realmente recibidos de cada fuente, usando la información de estado por fuente definida en la fuente `struct` referenciada mediante el puntero `s` en el código que se muestra más adelante. El número de paquetes recibidos es sencillamente la cuenta de los paquetes según llegan, incluyendo cualquier paquete retrasado o duplicado. El número de paquetes esperados puede ser calculado por el receptor como la diferencia entre el número de secuencia más alto recibido (`s->max_seq`) y el primer número de secuencia recibido (`s->base_seq`). Como el número de secuencia es sólo 16 bits y se repite en ciclos, es necesario extender el número de secuencia más alto añadiendo la cuenta

(desplazada) de los ciclos de número de secuencia (`s->cycles`). La cuenta de los paquetes recibidos y la cuenta de los ciclos se mantienen por la rutina de comprobación de validez del encabezamiento RTP de conformidad con el apéndice II.

```
extended_max = s->cycles + s->max_seq;
expected = extended_max – s->base_seq + 1;
```

Se define el número de paquetes perdidos como el número de los paquetes esperados menos el número de los paquetes realmente recibidos:

```
lost = expected – s->received;
```

Puesto que este número se transporta en 24 bits, debe amarrarse a 0xffffff en lugar de reciclarse a cero.

La fracción de paquetes perdidos durante el último intervalo de informes (desde que se envió el anterior paquete SR o RR) se calcula a partir de las diferencias entre las cuentas de los paquetes esperados y recibidos a lo largo del intervalo, donde `expected_prior` y `received_prior` son los valores salvaguardados cuando se generó el anterior informe de recepción:

```
expected_interval = expected – s->expected_prior;
s->expected_prior = expected;
received_interval = s->received – s->received_prior;
s->received_prior = s->received;
lost_interval = expected_interval – received_interval;
if (expected_interval == 0 || lost_interval <= 0) fraction = 0;
else fraction = (lost_interval << 8) / expected_interval;
```

La fracción resultante es un número de coma fija de 8 bits con la coma binaria en el borde izquierdo.

Apéndice IV

Cálculo del intervalo de transmisión RTCP

La siguiente función devuelve el tiempo entre las transmisiones de los paquetes RTCP, medido en segundos. Debe invocarse después de enviar un paquete RTCP compuesto para calcular el tiempo de espera hasta que deba enviarse el próximo paquete. Debe invocarse también para calcular el retraso antes de enviar el primer paquete RTCP tras el arranque inicial en lugar de enviar el paquete de inmediato. Esto evita una posible ráfaga de paquetes RTCP si se inicia una aplicación en muchos sitios simultáneamente, por ejemplo como resultado de un anuncio de sesión.

Los parámetros tienen el siguiente significado:

rtcp_bw: La anchura de banda RTCP proyectada, es decir, la anchura de banda total que será utilizada para los paquetes de RTCP por todos los miembros de esta sesión, en octetos por segundo. Esta debe ser un 5% del parámetro "anchura de banda de sesión" suministrado a la aplicación en el arranque.

senders (emisores): El número de emisores activos desde que se envió el último informe, y se conoce por construcción de informes de receptor para este paquete RTCP. Incluye al emisor en cuestión, si éste también emitió durante este intervalo.

members (miembros): El número estimado de los miembros de la sesión, incluido el miembro en cuestión. Aumenta según se detectan nuevos miembros de la sesión al recibirse paquetes RTP o RTCP, y disminuye según se retiran miembros de la sesión (mediante RTCP BYE) o cuando expira

el periodo de temporización para su estado (se recomienda un periodo de 30 minutos). En la primera llamada este parámetro debe tener el valor 1.

we_sent: Esta bandera tiene el valor true si se han enviado datos durante los dos últimos intervalos RTCP. Si la bandera tiene el valor true, el paquete RTCP compuesto que se acaba de enviar contenía un paquete SR.

packet_size: El tamaño del paquete RTCP compuesto que se acaba de enviar, en octetos, incluyendo la encapsulación de la red (por ejemplo, 28 octetos para UDP por IP).

avg_rtcp_size: Puntero al estimador de tamaño de paquete RTCP compuesto, inicializado y actualizado por esta función para el paquete que se acaba de enviar y también actualizado por una línea idéntica de código en la rutina de recepción RTCP para cada paquete RTCP recibido de otros participantes en la sesión.

initial: El valor de esta bandera es true para la primera llamada en el arranque inicial, a fin de calcular el tiempo de espera para el envío del primer informe.

```
#include <math.h>

     double rtcp_interval(int members,
     int senders,
     double rtcp_bw,
     int we_sent,
     int packet_size,
     int *avg_rtcp_size,
     int initial)
     {
     /*
     * Minimum time between RTCP packets from this site (in seconds).
     * This time prevents the reports from 'clumping' when sessions
     * are small and the law of large numbers isn't helping to smooth
     * out the traffic.  It also keeps the report interval from
     * becoming ridiculously small during transient outages like a
     * network partition.
     */
     double const RTCP_MIN_TIME = 5.;
     /*
     * Fraction of the RTCP bandwidth to be shared among active
     * senders.  (This fraction was chosen so that in a typical
     * session with one or two active senders, the computed report
     * time would be roughly equal to the minimum report time so that
     * we don't unnecessarily slow down receiver reports.) The
     * receiver fraction must be 1 - the sender fraction.
     */
     double const RTCP_SENDER_BW_FRACTION = 0.25;
     double const RTCP_RCVR_BW_FRACTION = (1-RTCP_SENDER_BW_FRACTION);
     /*
     * Gain (smoothing constant) for the low-pass filter that
     * estimates the average RTCP packet size (see Cadzow reference).
     */
     double const RTCP_SIZE_GAIN = (1./16.);

     double t; /* interval */
     double rtcp_min_time = RTCP_MIN_TIME;
     int n;   /* no. of members for computation */

     /*
     * Very first call at application start-up uses half the min
     * delay for quicker notification while still allowing some time
     * before reporting for randomization and to learn about other
```

```
* sources so the report interval will converge to the correct
* interval more quickly.  The average RTCP size is initialized
* to 128 octets which is conservative (it assumes everyone else
* is generating SRs instead of RRs: 20 IP + 8 UDP + 52 SR + 48
* SDES CNAME).
*/
if (initial) {
rtcp_min_time /= 2;
*avg_rtcp_size = 128;
}

/*
* If there were active senders, give them at least a minimum
* share of the RTCP bandwidth.  Otherwise all participants share
* the RTCP bandwidth equally.
*/
n = members;
if (senders > 0 && senders < members * RTCP_SENDER_BW_FRACTION) {
if (we_sent) {
rtcp_bw *= RTCP_SENDER_BW_FRACTION;
n = senders;
} else {
rtcp_bw *= RTCP_RCVR_BW_FRACTION;
n -= senders;
}
}

/*
* Update the average size estimate by the size of the report
* packet we just sent.
*/
*avg_rtcp_size += (packet_size - *avg_rtcp_size)*RTCP_SIZE_GAIN;

/*
* The effective number of sites times the average packet size is
* the total number of octets sent when each site sends a report.
* Dividing this by the effective bandwidth gives the time
* interval over which those packets must be sent in order to
* meet the bandwidth target, with a minimum enforced.  In that
* time interval we send one report so this time is also our
* average time between reports.
*/
t = (*avg_rtcp_size) * n / rtcp_bw;
if (t < rtcp_min_time) t = rtcp_min_time;

/*
* To avoid traffic bursts from unintended synchronization with
* other sites, we then pick our actual next report interval as a
* random number uniformly distributed between 0.5*t and 1.5*t.
*/
return t * (drand48() + 0.5);
}
```

Apéndice V

Estimación de la fluctuación entre llegadas

Los siguientes fragmentos de código a continuación implementan el algoritmo descrito en 6.2 para calcular una estimación de la varianza estadística del tiempo entre llegadas de datos RTP que se insertará en el campo fluctuación entre llegadas de los informes de recepción. Las entradas son `r->ts`, la indicación de tiempo del paquete entrante y la llegada, la hora (tiempo) actual en las mismas unidades. Aquí `s` apunta el estado para la fuente; `s->transit` contiene el tiempo relativo de tránsito para el paquete anterior y `s->jitter` contiene la fluctuación estimada. El campo fluctuación del informe de recepción se mide en unidades de indicación de tiempo y se expresa como un número entero sin signo, pero el valor estimado de la fluctuación se mantiene en un numero coma flotante. Conforme llega cada paquete de datos, se actualiza el valor estimado de la fluctuación:

```
int transit = arrival - r->ts;
int d = transit - s->transit;
s->transit = transit;
if (d < 0) d = -d;
s->jitter += (1./16.) * ((double)d - s->jitter);
```

Cuando se genera un bloque de informes de recepción (al cual apunta `rr`) para este miembro se retorna el valor estimado de la fluctuación actual:

```
rr->jitter = (u_int32) s->jitter;
```

Como otra posibilidad, el valor estimado de la fluctuación puede mantenerse como un número entero, pero escalado para reducir el error de redondeo. El cálculo es el mismo salvo en la última línea:

```
s->jitter += d - ((s->jitter + 8) >> 4);
```

En este caso, el valor estimado se muestrea para el informe de recepción de esta forma:

```
rr->jitter = s->jitter >> 4;
```

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie B	Medios de expresión: definiciones, símbolos, clasificación
Serie C	Estadísticas generales de telecomunicaciones
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	Sistemas audiovisuales y multimedios
Serie I	Red digital de servicios integrados
Serie J	**Redes de cable y transmisión de programas radiofónicos y televisivos, y de otras señales multimedios**
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	RGT y mantenimiento de redes: sistemas de transmisión, circuitos telefónicos, telegrafía, facsímil y circuitos arrendados internacionales
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de transmisión telefónica, instalaciones telefónicas y redes locales
Serie Q	Conmutación y señalización
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	Terminales para servicios de telemática
Serie U	Conmutación telegráfica
Serie V	Comunicación de datos por la red telefónica
Serie X	Redes de datos y comunicación entre sistemas abiertos
Serie Y	Infraestructura mundial de la información y aspectos del protocolo Internet
Serie Z	Lenguajes y aspectos generales de soporte lógico para sistemas de telecomunicación

Impreso en Suiza
Ginebra, 2002