Transmisión

Alexandre Ribelles García

PID_00198490

Universitat Oberta de Catalunya

Índice

Introducción

Una vez codificada digitalmente la información y comprimida de la mejor manera posible en función de su contenido (en general, una compresión sin pérdidas para datos puros, y con pérdidas o sin ellas para audio y vídeo), es imprescindible definir mecanismos para su transmisión.

La relación entre codificación y transmisión es intensa, pues se han definido mecanismos de codificación específicos para combatir las limitaciones en la transmisión, como por ejemplo los formatos multimedia de distribución que ya conocéis. Es una de las formas en que la "marea" de la convergencia digital ha podido englobar también el contenido audiovisual en su proceso de unificación de la información digital sobre los mismos medios digitales de distribución. En la primera parte de este módulo nos centraremos en los problemas presentes en la distribución audiovisual sobre redes de paquetes como las redes IP.

El vídeo, sin duda debido a sus necesidades específicas de volumen de datos y cadencia, ha sido de las últimas informaciones en integrarse, pero una vez dado el paso, está presente en todas partes, en nuestro día a día, como un servicio más. Así pues, el proceso de integración ha contado con dos elementos paralelos: los estándares de formato digital de vídeo y la implementación de sistemas digitales avanzados de transmisión.

La transferencia de vídeo y audio digital en tiempo real, o *streaming*, es básicamente un servicio de valor añadido cuya calidad está adaptada a la capacidad y tipo de receptor cliente (teléfono móvil, tableta, puntos de información, etc.) gracias a estándares más o menos abiertos que posibilitan dar ese servicio al vuelo. Veremos los modos de distribución, los mecanismos de red que lo hacen posible y los protocolos de red que se utilizan habitualmente. Acabaremos el módulo viendo un ejemplo completo de implementación de transmisión en tiempo real.

1. Planteamiento básico

En una red IP como Internet, pensada para la transmisión de datos, las señales de vídeo no son fáciles de transportar, básicamente por dos razones: la primera es que no se trata de una red pensada para vídeo, el tráfico por la red del vídeo y audio comprimidos se efectúa mediante paquetes de datos genéricos (paquetes IP), como todos los otros tipos de datos (textos, imágenes, etc.) que fluyen por la Red. Estos paquetes de datos circulan por las mismas redes y son encaminados por los mismos centros de distribución (*routers* o encaminadores) que cualquier otro dato, y a veces se desordenan o se pierden. Además, el flujo de datos es muy alto, casi constante, por lo que necesita gran cantidad de paquetes para asegurar la reproducción con la cadencia necesaria: por ejemplo, un vídeo de YouTube de un minuto puede exigir una entrega de unos mil paquetes sin perder el orden y con una cadencia fija.

Sin duda, el vídeo por Internet (*streaming*) y el vídeo por *broadcast* (difusión amplia) –por ejemplo, la TDT– plantean diferencias críticas en la distribución que obligan a retos técnicos en parte superados. Sin embargo, más allá de la tecnología, se da una revolución más profunda y de consecuencias más intensas: el usuario elige lo que desea ver desde el punto en que desea verlo, en contraposición a ver lo que le ofrecen tal como indica la programación estática del canal.

Esta nueva capacidad de decisión por parte del usuario final obliga a replantear la atracción del producto y a crear nuevos mercados en los que aprovechar la interacción. Esta revolución ya está en marcha, y argumentos tales como la falta de calidad de imagen o de tratamiento profesional asegurado de los contenidos de la Red van perdiendo fuerza año tras año, al incorporarse profesionales preparados para trabajar en este medio como su entorno natural.

2. Codificación y transmisión

Los dos entornos habituales de aplicación de los códecs son la transmisión de audio/vídeo en tiempo real[1] y el almacenamiento. Sin embargo, en cada uno de estos dos casos debe seleccionarse una codificación adecuada a las condiciones en que se realiza.

[(1)]En inglés, *streaming*.

1) Almacenamiento

En los sistemas de codificación orientados a almacenamiento, el codificador tiene acceso a todo el material, por lo que puede hacer previsiones más fiables y conseguir tasas altas de compresión a la vez que mantiene la calidad audiovisual. El tiempo que tarde en generar el fichero final no es un factor crítico, por lo que los requerimientos de capacidad de cálculo y necesidad de memoria no son demasiado elevados. Ello posibilita que puedan existir codificadores de software dignos para cumplir esta misión e incluso funcionales en equipos informáticos domésticos.

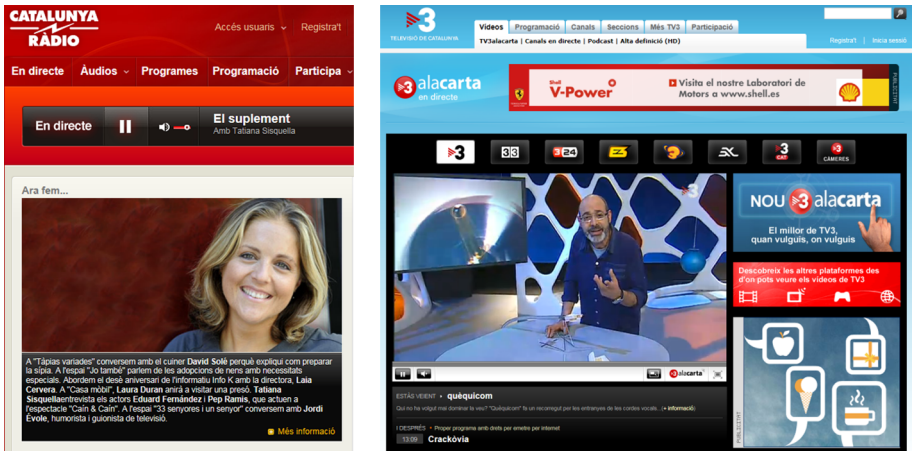
2) Transmisión en tiempo real

En los sistemas de transmisión en tiempo real como Internet Radio o IPTV (televisión basada en protocolo IP), el material de que se dispone para efectuar la codificación va llegando a la misma velocidad que se genera y debe codificarse a esa misma velocidad, por lo que se necesita un equipo de alta disponibilidad, fiabilidad, gran capacidad de memoria y excelente capacidad de cálculo. La mayoría son equipos de hardware dedicado de alto valor ubicados en la empresa que proporciona el servicio: la mayoría de las webs de compañías *broadcast* actuales de radio y televisión son un ejemplo.

Ilustración 1. Equipo codificador HD para TDT o IPTV

En otros servicios de transmisión en tiempo real menos ambiciosos (como por ejemplo la videoconferencia), en los que la calidad visual no es relevante y la cadencia de las imágenes no se asegura, los requisitos del codificador se relajan y pueden existir incluso en software (FaceTime, Messenger, etc.).

Ilustración 2. Otros servicios de transmisión en tiempo real menos ambiciosos



La mayoría de las empresas de radio y televisión emiten también parte de su programación por Internet en paralelo a su emisión por el medio habitual.

3) El caso intermedio: vídeo bajo demanda

Acceder a un vídeo de YouTube (vídeo bajo demanda[2]) parece situarse entre ambos casos, pues supone visualizar en tiempo real un material almacenado previamente en un servidor remoto. Lo crucial en este caso es garantizar la cadencia de recepción, y eso se ha previsto antes, en la generación del material, que se crea en un formato audiovisual fácilmente transportable y que puede comenzar a visualizarse desde la llegada de los primeros datos al cliente.

(2)En inglés, *video on demand*.

Esta previsión del formato en su almacenamiento hace que se suela incluir este caso entre los sistemas de transmisión en tiempo real, y así lo haremos a partir de este momento, aunque con puntualizaciones. Por ello, servicios de vídeo como Netflix, Hulu, Google TV o la mayoría de los servicios a la carta son sistemas de transmisión en tiempo real.

4) El receptor

El receptor (tanto por lo que respecta a la transmisión en tiempo real como a la lectura de ficheros almacenados) posee un descodificador que no necesita grandes prestaciones, ya que los sistemas de codificación son los que realizan el trabajo de análisis y generan la información mínima necesaria para reconstruir el material audiovisual. Así, existe software doméstico capaz tanto de reproducir un Blu-ray en alta definición como de visualizar una conferencia emitida por Internet al otro lado del mundo.

Ilustración 3. Reproductores de software de propósito general

VLC, QuickTime Player o Windows Media Player, entre otros, son ejemplos de reproductores de software de propósito general ejecutables en entornos domésticos de pocas prestaciones.

3. La problemática de transmisión de audio y vídeo sobre red IP

La transmisión de información audiovisual en tiempo real por una red IP, cuya infraestructura y protocolos de red están optimizados para el transporte de pequeños paquetes de datos genéricos, obliga a enfrentarse a una serie de problemas, que se detallan a continuación:

1) La tasa de transmisión de datos (ancho de banda necesario y disponible).

2) La cadencia del flujo frente a las variaciones del retardo.

3) La pérdida de datos.

3.1. La tasa de transmisión de datos (ancho de banda necesario y disponible)

En primer lugar, es deseable tener en mente el orden del volumen de datos que se requieren en un flujo audiovisual para detectar uno de los problemas a los que se enfrenta la transmisión.

Comencemos considerando el caso de un mero flujo de datos de audio, y a continuación el caso de un flujo de vídeo.

La señal de audio digital sigue el formato AES/EBU, con cada muestra de 16 bits y a velocidades de 32.000, 44.100 y 48.000 muestras según el ancho de banda analógico que se desee representar (hasta 16 kHz, calidad FM; hasta 22 kHz, calidad CD, y hasta 24 kHz, calidad de estudio profesional, respectivamente). Así, una señal estéreo digital AES supone un flujo de datos no comprimidos de:

$$\text{Flujo de audio} = 44.100 \text{ muestras} \times 16 \frac{\text{bits}}{\text{muestra}} \times 2 \text{ canales} = 1{,}41 \text{ Mbps}$$

Si además contamos con que el estándar AES reserva 32 bits para transportar cada muestra y otros datos asociados (como el código de tiempo), resulta:

$$\text{Flujo de transporte de audio} = \text{Flujo de audio} \times 2 = 2{,}82 \text{ Mbps}$$

Eso supondría casi una conexión de 3 Mbps solo para reproducir el audio estéreo a través de una red. El caso de audio multicanal o envolvente multiplica por tres o por seis esta cifra. Evidentemente, se comprime sin perder gran calidad para poder distribuir audio por red.

Ejercicio 1

Instalaos el programa de reproducción VLC. Con este programa, abrid un fichero de audio y, mientras está en reproducción, acceded a *Herramientas – Información multimedia*. En la solapa *Detalles del códec* se indica el formato en el que está comprimido el audio, y en la solapa *Estadística* se presentará una serie de datos entre los que destacan *Tasa de bits de entrada*, es decir, los kilobits por segundo que el programa va leyendo del fichero de audio, y *Tasa de bits de contenido*, que indica cuántos de los kilobits por segundo leídos realmente son para audio (los demás son de control, calidad, sincronía, etc.).

Entrada/lectura	
Tamaño de datos del medio	1504 KiB
Tasa de bits de entrada	128 kb/s
Tamaño de datos demuxados	1458 KiB
Tasa de bits del contenido	125 kb/s
Descartados (corruptos)	0

Comparad la tasa de bits de contenido con el flujo de audio (no el flujo de transporte de audio) que debería tener en formato AES para dar una idea de la compresión alcanzada con el formato de audio del fichero y de la pérdida de la calidad de sonido que se pueda detectar.

Probadlo con diferentes formatos (MP3, WMA, WAV, etc.).

Ejercicio 2

Hagamos lo mismo con material audiovisual, por ejemplo una película almacenada en un DVD o Blu-ray doméstico. Para ello, con el menú *Medio – Abrir disco*, seleccionad la unidad de disco y reproducid una película. Acceded en cualquier momento al menú *Herramientas – Información multimedia – Detalles del códec*, donde, en el apartado *Vídeo*, se indicarán el formato de codificación (MPEG-2 en DVD, H.264/AVC en Blu-ray), el ancho y el alto de la imagen en píxeles y los fotogramas por segundo. Es fácil calcular, pues, que el flujo de datos de lectura en bits por segundo debería ser, sin compresión, de:

$$\text{Flujo de datos} = \text{ancho(píxeles)} \times \text{alto(píxeles)} \times 8\frac{\text{bits}}{\text{color}} \times 3 \text{ colores} \times \frac{\text{fotogramas}}{\text{segundo}}$$

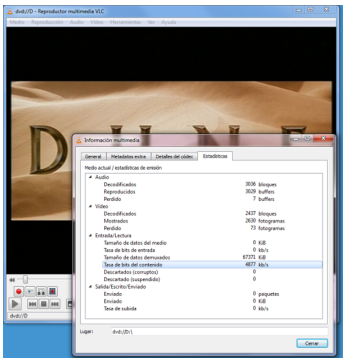


Ilustración 4. Tasa de bits informada por el programa VLC

Una cifra que se acerca a los 240 Mbps para material de calidad estándar y a más de 1 Gbps en calidad de alta definición. Ambos valores resultarían en muy difícil transmisión en las redes actuales para un solo flujo audiovisual.

Ahora, en la solapa *Estadística*, podéis ver la *Tasa de bits del contenido*, que son los bits por segundo necesarios para reproducir correctamente el fichero comprimido:

Entrada/lectura	
Tamaño de datos del medio	9079 KiB
Tasa de bits de entrada	973 kb/s
Tamaño de datos demuxados	6801 KiB
Tasa de bits del contenido	1030 kb/s
Descartados (corruptos)	0

La compresión que se ha alcanzado, por lo general muy alta, es calculable, como ya sabéis:

$$\text{Factor de compresión} = \frac{\text{Flujo de datos sin comprimir (bps)}}{\text{Flujo de datos comprimido (bps)}}$$

Ejercicio 3

Realizamos la misma operación pero en este caso con un fichero de vídeo que se haya descargado de Internet (WMV, MP4, AVI, etc.). Para visualizarlo con VLC, seleccionad el menú *Medio – Abrir archivo*, y recoged los datos que presenta. Calculad el factor de compresión y comparadlo con el del caso anterior. ¿Por qué hay tal diferencia?

Reflexión

Si en vez de DVD se utiliza Blu-ray, comparadlo con un fichero de vídeo que sea en alta definición, o la comparativa no tendría sentido.

En segundo lugar, no solo es reseñable el volumen de transmisión de datos, sino que existen variaciones en la tasa de bits del contenido.

Ello es debido a la tasa de bits variable de la codificación del vídeo (VBR) típica de DVD y Blu-ray, muy interesante porque mantiene la calidad de la codificación y es ideal cuando el objetivo es el almacenamiento de los datos, ya que reduce las necesidades de espacio, pero que genera inquietud en la transmisión: ¿hay que dimensionar el canal para la tasa máxima, o se tiene que dimensionar para una tasa promedio, en cuyo caso hay que aceptar pérdidas de datos?

Existen varias soluciones, que explicaremos de forma escueta, pues esta cuestión se sale de los objetivos de la asignatura:

- Poner una memoria intermedia[3] entre la salida del codificador y la entrada de la red, de manera que reduzca las variaciones del flujo VBR y lo acerque a un seudoflujo estable. Sin embargo, aumenta el retardo global de la señal.

(3)En inglés, *buffer*.

- Sumar diferentes flujos VBR antes de inyectarlos a la red, ya que la suma de flujos variables siempre supone un flujo más estable. Esta solución solo tiene sentido si todos los flujos parten del mismo punto de la red y se dirigen también al mismo punto.

Como tercer y último aspecto, destacamos que el ancho de banda disponible en una red IP también suele ser variable.

En estas redes no puede hacerse una reserva de ancho de banda, sino que es dinámica. Si el flujo de datos supera el ancho de banda, la red se congestionará, perderá datos y la calidad de recepción disminuirá. Si el flujo es inferior al ancho de banda, malgastamos ancho de banda que podría dedicarse a otras transmisiones.

Existen varias soluciones para este problema:

- que el codificador emisor genere un flujo de datos[4] de salida en función del ancho de banda libre en la red en cada momento;

(4)En inglés, *stream*.

- que el descodificador receptor elija un nivel de calidad del flujo de datos que recibe. Esto solo es posible si el flujo de datos está codificado por capas o es escalable (por ejemplo, MPEG-4).

3.2. La cadencia del flujo frente a las variaciones del retardo

Toda red genera un retardo entre el emisor y el receptor. Sin embargo, en las redes IP este retardo es variable ya que, como se verá a continuación, el camino de los datos a través de dichas redes puede ir cambiando en plena transmisión.

Tanto audio como vídeo son informaciones que requieren una cadencia constante de envío y recepción de datos para poder reconstruir el efecto de movimiento en vídeo y la inteligibilidad en audio: cuando comienza la reproducción, se ha de mantener el flujo presentando una imagen y una porción de audio cada cierto tiempo. Y si el caso es una comunicación audiovisual en ambos sentidos, o interactiva, la situación es más complicada.

Así, un retardo de 150 ms en el audio no es detectado por el oído humano, y por encima de 400 ms lo vuelve ininteligible. En vídeo, el retardo supone una congelación de la imagen o de parte de esta. Y retardos diferentes entre audio y vídeo son notables si el vídeo antecede al audio en 100 ms, o lo sigue más allá de los 20 ms.

Todas estas variaciones del retardo[5] obligan a añadir inteligencia y memoria en los equipos, especialmente en el equipo receptor, que incrementan su coste: se almacenan los datos en el receptor en una cola de memoria antes de reproducirlos en el usuario, con lo que no se perciben variaciones en la llegada de los datos, aunque se genera un retardo total proporcional al tamaño de la cola. A pesar de todo, los datos que llegan demasiado tarde (es decir, más allá del momento en que deberían haberse mostrado) son descartados y suponen errores.

(5)En inglés, *jitter*.

3.3. La pérdida de datos

Dado que toda red es susceptible de pérdida de datos (de paquetes por caída de un nodo de la red o de un enlace entre nodos, o corrupción del contenido del paquete por ráfagas de ruido), existen mecanismos para proteger los datos ante estas situaciones en la medida de lo posible.

Básicamente, existen cuatro técnicas de control de errores, agrupadas en dos familias:

1) Control de errores por codificación de canal. Por ejemplo:

- **Retransmisión de paquetes** (**ARQ**[6]): el receptor detecta los paquetes faltantes en función de algún criterio (orden, importancia) y los reclama para que sean retransmitidos.

(6)ARQ es la sigla de la expresión inglesa *automatic repat request*.

(7)FEC es la sigla de *forward error correction*.

- **Técnicas de FEC**[7]: el receptor o cualquiera de los nodos que cruza en su camino es capaz de detectar la mayoría de los bits corruptos que posee el paquete y reconstruirlo correctamente.

En el caso del mecanismo ARQ, gracias a un contador que marca de manera única y secuencial cada paquete en emisión, el receptor detecta los que faltan y los reclama. Los paquetes reclamados se reenvían siempre y cuando se siga algún criterio: el criterio más simple es que se estime que puedan llegar al receptor antes del instante en que han de presentarse al destinatario. Otra posibilidad es reclamar los paquetes perdidos no por su orden, sino según su importancia en la descodificación del material audiovisual.

Y es que hay bits más importantes que otros; por ejemplo, en la familia MPEG, los pertenecientes a las imágenes I siempre son más importantes que los de las imágenes P, y estos más que los de las imágenes B, que incluso pueden descartarse por completo. En la codificación escalable encontramos otro ejemplo: el flujo principal ha de tener más importancia que el primer flujo de mejora, y este más que el segundo flujo de mejora del contenido y subsiguientes.

Los controles de errores por codificación de canal pueden combinarse, de manera que se intenta recuperar el máximo número de paquetes corruptos mediante FEC, y aquellos que no se consiguen o que ni siquiera llegan a destino se retransmiten mediante ARQ. Las tablas resumen siguientes representan la idea de su aplicación combinada en la transmisión de un flujo MPEG:

	Imagen I	**Imagen P**	**Imagen B**
FEC	Máxima	Media	Mínima
ARQ	Máxima	Media	Descartable

Y en la transmisión de un flujo escalable:

	Flujo base	**F. mejora 1**	**F. mejora 2**
FEC	Máximo	Medio	Mínimo
ARQ	Máximo	Medio	Descartable

2) Control de errores por codificación de fuente. Por ejemplo:

- **Ocultamiento del error**[8]: se estima la información perdida para así poderla ocultar/disimular en la presentación. La misma correlación o redundancia que poseen el vídeo y el audio (y que tan bien ha servido para definir estrategias de codificación) ofrece ahora una manera sencilla de valorar esa información perdida.

(8)En inglés, *error concealment*.

- **Codificación de vídeo resistente a los errores**[9]: se aplican técnicas de codificación que aumenten la resistencia ante errores específicos, por ejemplo la codificación de Huffman, la codificación *run-lenght*, etc.

(9)En inglés, *error resilient video coding*.

Por lo general, los errores que afectan a este tipo de control suelen ser dos:

a) Pérdida de la sincronización del flujo de bits, de manera que el descodificador no sabe qué bits corresponden a qué parámetros. Es muy grave si se utiliza un código de longitud variable (como Huffman o *run-lenght*), ya que el error en un solo bit lleva al descodificador a suponer una longitud incorrecta de la palabra, con lo que interpreta también de manera incorrecta las siguientes hasta que se llegue a alguna resincronización.

b) Estados incorrectos y **propagación de errores**, que sobre todo afecta a aquellos descodificadores que trabajan sobre codificación predictiva (por ejemplo, DPCM sobre audio).

4. Transmisión de vídeo en una red IP

Las técnicas para la transmisión de vídeo en las redes IP se conocen como **mecanismos de *streaming***. El *streaming* es la transmisión en tiempo real (en vivo) de audio y vídeo sobre una red.

Antes de la aparición de las técnicas de *streaming*, las aplicaciones multimedia usaban Internet principalmente para realizar transferencias de archivos. Así, una vez que los contenidos eran descargados completamente, podían ser reproducidos. En la actualidad, utilizando *streaming*, es posible ir visualizando un contenido multimedia a medida que este es transferido.

Como hemos visto en los estándares digitales anteriores, los flujos de datos de audio y vídeo se empaquetan formando cada uno un flujo elemental (ES[10]), que además guarda información de sincronía entre ellos. A continuación se combinan (al menos uno de vídeo y otro de audio) en un flujo elemental empaquetado (PES[11]), lo que posiblita que se combinen más.

(10)ES es la sigla de *elementary stream*.

(11)PES es la sigla de *packetized elementary stream*.

A partir de este momento, podemos hacer dos cosas: almacenar este contenido en un contenedor para su reproducción o transmitirlo (en nuestro caso, por una red IP). En apartados anteriores se ha visto por encima cómo se acondicionan los flujos de datos para ser almacenados, ahora se verá cómo enviarlos por la red.

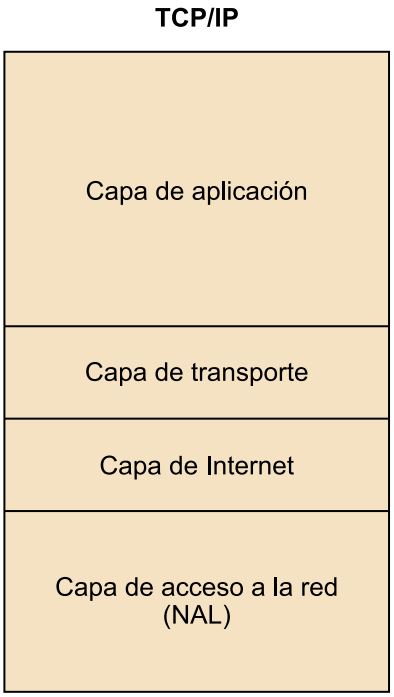
Existen múltiples técnicas de *streaming* según la realidad particular del sistema de vídeo digital. En este apartado se verán las más importantes en el contexto de las redes basadas en IP, como 3G, IPTV e Internet, y a continuación se hará una revisión de los protocolos de las redes IP y de los protocolos de *streaming*.

4.1. Las redes IP y sus protocolos

Una red puede analizarse a distintos niveles. Por ejemplo, se podría estudiar desde un punto de vista eléctrico, midiendo las tensiones de las señales que circulan y los tipos de conectores necesarios. Este punto de vista es el denominado **nivel físico** o **nivel de acceso a la red (NAL[12])**, más propio de investigadores e instaladores.

(12)NAL es la sigla inglesa de la expresión *nivel de acceso a la red*.

Ilustración 5. Arquitectura de red TCP/IP



Un segundo punto de vista sería el **nivel de enlace**, que estudia la identificación de las tarjetas de red y el protocolo de intercambio de bits entre estas. Este nivel también es más propio de desarrolladores y programadores que nuestro, ya que estamos situados en un nivel de uso más abstracto.

Los niveles que nos interesan son los dos siguientes: el **nivel de red**, en el que hablamos de datos que circulan por la red desde un equipo en dirección a otro equipo (o a otros), y el **nivel de transporte**, que consiste en los mecanismos para asegurar que los datos llegan correctamente y, si no lo hacen así, se pide su reenvío. Sobre esta cuarta y última capa ya tendríamos las aplicaciones informáticas que usan la Red. En este apartado solo se hace mención a estas dos últimas capas (red y transporte) de las redes IP, con el fin de entender los mecanismos de *streaming* presentados a continuación.

4.1.1. Protocolos de red (IP)

Las redes IP[13] son un territorio inestable y siempre cambiante de ruta desde un punto a otro. Para que un paquete de datos llegue a su destino, ha de tener indicado dicho destino en cada uno de los paquetes que conforman los datos que se envían.

(13)IP es la sigla de *Internet protocol*.

El protocolo IP es un protocolo robusto que no espera que haya un camino asegurado y estable entre origen y destino, sino que los datos se dividen en paquetes que se envían de manera consecutiva y cada uno “navega” en una red llena de paquetes hasta alcanzar el destino que tienen marcado.

Dicho técnicamente, el protocolo IP es no orientado a conexión para la comunicación a través de una red de paquetes conmutados. Es el protocolo de Internet y de los sistemas de IPTV.

Estos paquetes son enviados con información extra que pueda garantizar su propia integridad, pero no proporciona ningún tipo de seguridad de que los datos realmente lleguen a su destino (de esto se encargará el protocolo superior, el protocolo de transporte).

4.1.2. Protocolos de transporte (UDP, TCP y SCTP)

Por encima de este "cartero" que es el protocolo IP, siempre hay un protocolo de transporte cuya responsabilidad es asegurar la transferencia sin errores de datos entre el emisor y el receptor. Sin embargo, ha de ser ágil, o su propia acción puede ralentizar la celeridad con que se reciben tales datos. Los bits de este protocolo van dentro del paquete de red.

Dependiendo del tipo de servicio que queramos, podemos elegir uno de los tres protocolos de transporte siguientes:

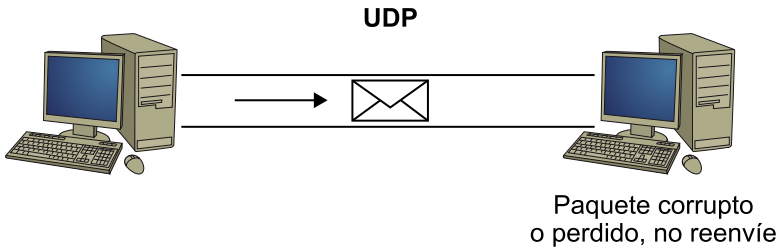
a) Protocolo UDP

UDP[14] es un protocolo ligero que no da garantías sobre la pérdida de paquetes ni la recepción por duplicado de paquetes. Solo cuenta con mecanismos de chequeo de datos, de modo que si hay que gestionar los errores en la transmisión, se deberá hacer en el lado del destinatario (en la aplicación que los reciba).

(14)UDP es la sigla de *user datagram protocol*.

Es el protocolo de transporte natural para *streaming* de vídeo y audio.

Ilustración 6. Protocolo UDP



En UDP, la pérdida, error o duplicación de paquetes no genera ninguna petición de reenvío por parte del destinatario.

La razón para que toda videoconferencia que se precie, toda sesión de Spotify o toda web pirata de series de TV no estrenadas en Europa utilicen este protocolo es que consume pocos bits del total de bits de datos y es rápido en inicializarse (pues no hace mucha gestión). Ciertamente, es menos seguro que otros,

pero en *streaming* se considera inútil una retransmisión de la información perdida, puesto que esta llegaría a destiempo (el *streaming* presenta restricciones de tiempo real).

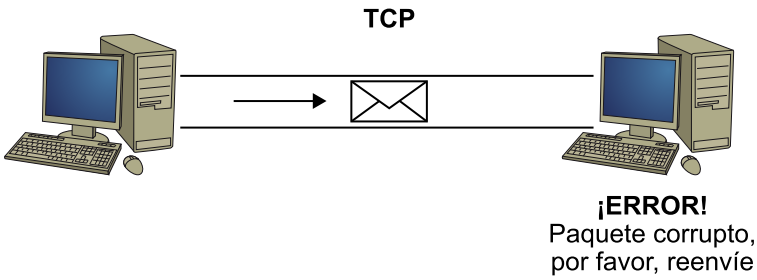
b) Protocolo TCP

El protocolo TCP[15] nos asegura que los paquetes lleguen, y que lo hagan en orden.

(15)TCP es la sigla de *transmission control protocol*.

Esto se consigue mediante un mecanismo en el cual, cuando no se recibe durante cierto tiempo la confirmación del paquete por parte del destinatario, dicho paquete se reenvía. Debido a este comportamiento, es posible detectar paquetes perdidos y pedir su retransmisión. En el caso de transmisión de *stream*, cuando se pierden paquetes, la retransmisión aumenta el retardo y el consumo del ancho de banda, lo que puede provocar que se vacíe el *buffer* de reproductor (y, por consiguiente, la interrupción de la reproducción del *stream*).

Ilustración 7. Protocolo TCP



En TCP se asegura la recepción correcta de todos y cada uno de los paquetes.

En redes de IPTV (Imagenio, entre otras), donde existe una red dedicada en exclusiva y que asegura gran calidad de servicio y no hay congestión, el mecanismo de *streaming* continúa siendo el tradicional basado en UDP. En cambio, en redes de uso general, donde no existen mecanismos para asegurar la calidad de servicio, como Internet, se podría optar por realizar el *streaming* sobre TCP. Sin embargo, este protocolo presenta dos problemas: en primer lugar, porque el tráfico de paquetes retransmitidos afecta a la cadencia y calidad del flujo de datos y, en segundo lugar, porque por el propio funcionamiento intrínseco del protocolo TCP, se generan oscilaciones de la tasa media de transmisión en detrimento periódico del *streaming*.

c) Protocolo SCTP

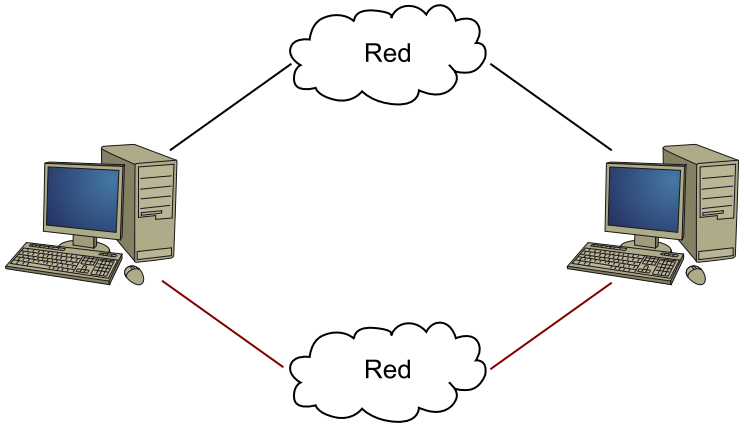
El protocolo de transporte SCTP[16] es una solución de compromiso entre los tradicionales UDP y TCP. A pesar de estar completamente estandarizado, ha tenido una lenta difusión. Tiene una potencial utilidad en el *streaming* de vídeo en redes inalámbricas y otros contextos con grandes pérdidas puntuales de datos.

(16)SCTP es la sigla de *stream control transmission protocol*.

Este protocolo se basa en datagramas, pero a diferencia de UDP, dispone de un registro de números de secuencia con los que efectúa controles de paquetes fuera de orden y perdidos, que serán retransmitidos.

A diferencia de TCP, este protocolo permite diferentes direcciones IP dentro de una misma conexión (origen SCTP – destino SCTP). O sea, que cada uno de los extremos de conexión puede tener diferentes IP (manteniendo el puerto de conexión) y los paquetes serán enviados indistintamente por cualquier IP y a cualquier IP correspondiente a cada extremo SCTP.

Ilustración 8. Protocolo SCTP



SCTP posibilita diferentes direcciones IP por un mismo puerto.

El protocolo SCTP también soporta el tráfico sobre la misma conexión de diferentes tipos de paquetes manteniendo un número de secuencia para cada uno de ellos.

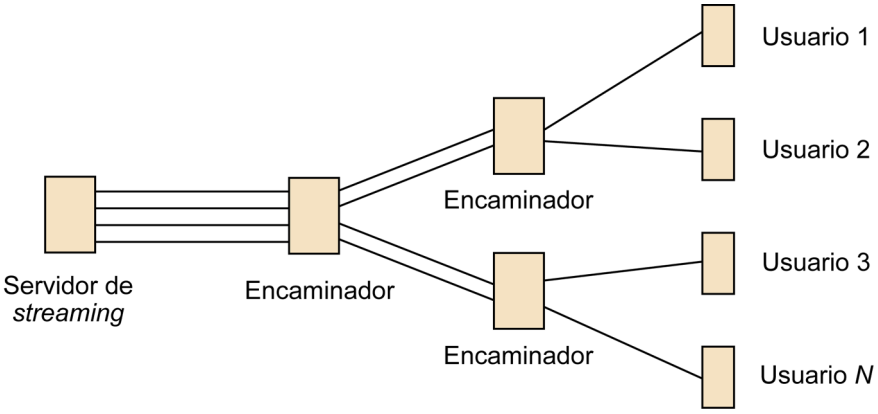
4.2. Técnicas de multidifusión sobre redes IP

En las redes IP existen distintas técnicas para la difusión de la información a varios destinatarios simultáneos:

- **Unidifusión**[17]: el envío de un punto a otro en una red se denomina unidifusión. Este mecanismo, habitualmente utilizado por todas las aplicaciones, tiene un número *n* máximo de usuarios limitado.

(17) En inglés, *unicast*.

Ilustración 9. Unidifusión

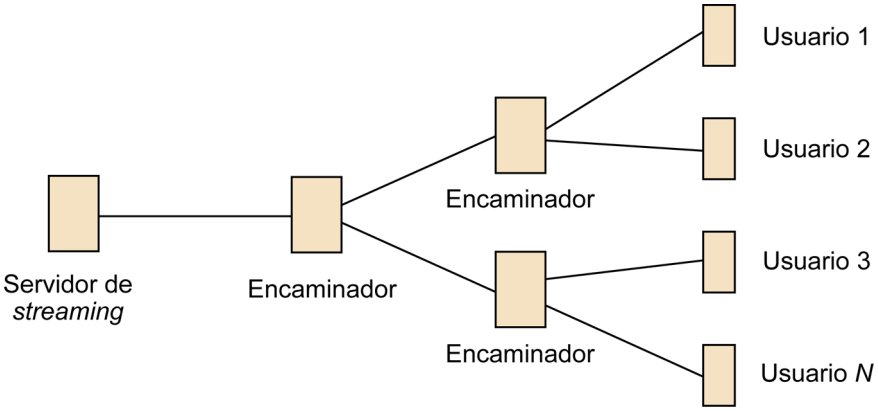


N usuarios visualizan simultáneamente un programa de televisión por Internet; en *unicast*, el servidor ha de enviar el flujo de datos para cada usuario.

- **Multidifusión**[18]: se trata del envío de la información a múltiples destinos a la vez con la ayuda de centros de difusión preparados que, si están enviando el *stream* a un usuario, al recibir otra petición, duplican los datos del *stream*. El servidor queda mucho más descargado, pero obliga a disponer de encaminadores capacitados para este protocolo.

(18)En inglés, *multicast*.

Ilustración 10. Multidifusión



En multidifusión, el servidor envía inicialmente un flujo de datos al primer usuario. Los demás reciben el suyo a través de reenvíos de los centros de difusión, que se percatan de poseer más de un usuario con la misma petición.

- ***Broadcast***: el envío a todos los nodos en una red se denomina *broadcast*. Lógicamente, solo es aplicable en contextos reducidos, por ejemplo para el descubrimiento de recursos en una red local (LAN[19]).

(19)LAN es la sigla inglesa de red de área local.

La técnica de multidifusión es de fácil descripción y de complicada implementación.

La **multidifusión** debe usar la estrategia más eficiente para el envío de los mensajes sobre cada usuario de la red (con el fin de disminuir el consumo de ancho de banda), utilizando cada enlace a lo sumo una vez para que cada paquete sea difundido y creando copias en los centros de difusión cuando los enlaces en los destinos se dividan. Por lo general, esto se logra mediante la utilización de **árboles de cubrimiento**, al surgir un protocolo para mantener dicho árbol a través de los centros de difusión.

Desde el punto de vista del emisor y los receptores, el proceso es sencillo: el emisor envía la información una sola vez a una dirección IP especial (de forma idéntica a una unidifusión); los receptores se suscriben al grupo de multidifusión asociado a esa IP especial y reciben una copia de todo lo enviado por el emisor. Por tanto, antes del envío de la información en multidifusión, deben establecerse una serie de parámetros.

Para poder recibirla, es necesario establecer el denominado grupo *multicast*. En IPv4, las IP especiales pertenecen al rango 224.0.0.0 a 239.255.255.255, y el protocolo utilizado para manejar y asociarse a los grupos de multidifusión se denomina *Internet group management protocol* (IGMP).

Dependiendo de la información, se elige el mecanismo ideal para su difusión.

En **IPTV**, la transmisión de canales en vivo habitualmente se realiza mediante multidifusión (*multicast*), puesto que se espera que tengan una numerosa audiencia (su envío individual a cada espectador (*unicast*) redundaría en un dimensionamiento inadmisible de la red).

En cambio, el **vídeo bajo demanda (VoD** [20]) es por naturaleza punto a punto, y por tanto se suele utilizar unidifusión. Pero esto no es una regla, incluso en el contexto de IPTV. Por ejemplo:

[20] VoD es la sigla de *video on demand*.

- Existen técnicas para reducir el tiempo de *zapping* entre los canales en vivo (enviados por multidifusión) que implican el envío unidifusión de una ráfaga inicial de gran ancho de banda para llenar rápidamente la memoria intermedia del receptor y reproducir enseguida tras el cambio de canal.

- Por otro lado, en los sistemas de VoD, cuando un contenido es muy solicitado bajo demanda, este puede ser reproducido en forma de nVoD[21], que implica el envío por multidifusión y una espera inicial de algunos minutos por parte del cliente para la transmisión multidifusión siguiente. Una técnica un poco más sofisticada implica el envío de una buena porción del VoD por unidifusión y el almacenamiento en la memoria intermedia de la transmisión multidifusión antigua más cercana en el tiempo. Esto evita la espera inicial del sistema nVoD pero exige a los receptores poseer

[21] nVoD se refiere a la expresión *near VoD*.

un espacio de memoria de almacenamiento mucho mayor. Este tipo de mecanismo mixto aún debe investigarse más.

Debe concluirse que, en los sistemas de vídeo digital especializados como IPTV, la utilización eficiente de las técnicas de difusión redundará en una mejor calidad de experiencia y aprovechamiento de los recursos de la red.

La situación en el contexto de **Internet** es muy distinta. En los comienzos de Internet, la multidifusión no se encontraba estandarizada y difícilmente era soportada por los sistemas operativos y el hardware dedicado de la época. Hoy, por razones de índole comercial, se mantiene esta restricción sobre Internet, por tanto no se soporta la multidifusión (y, por razones lógicas de escala, tampoco el *broadcast*). Una esperanza para el soporte de multidifusión en Internet reside en el protocolo IPv6, la nueva versión del protocolo IP, que lo soporta de forma nativa.

Ilustración 11. Unidifusión y multidifusión en Internet
Tanto las técnicas multidifusión como las unidifusión están restringidas actualmente en Internet.

Si Internet soportara multidifusión, los sistemas dedicados de IPTV presentarían una fuerte competencia, que sería una oferta de contenido global para un público global (por este motivo no resulta atractivo para los proveedores de conectividad ofrecer tales beneficios a los proveedores de servicios de Internet). Los actuales sistemas de vídeo en Internet realizan, según sus posibilidades, un "*multicast* en capa de aplicación" mediante el despliegue global de *data-centers* y la réplica inteligente de la información.

A pesar de ser técnicas de multidifusión más eficientes, el problema subsiste en el tramo más comprometido: la red de acceso, donde estas redes solo pueden hacer distribución unidifusión (YouTube, Google Video, Netflix, etc.).

4.3. Los protocolos de *streaming*

Existe una tercera categoría de protocolos diseñados específicamente para *streaming*, conocidos como protocolos de tiempo real, que intentan asegurar el envío de flujos de datos con una cadencia asegurada, tal como necesita el material audiovisual.

4.3.1. Protocolos de tiempo real (RTP, RTCP y RTSP)

Cuando hablamos al principio del módulo de los problemas inherentes a las variaciones del ancho de banda de las redes IP, una de las dos soluciones era ajustar la tasa de codificación a la capacidad real en cada momento de la red para que no se saturase y perdiese datos. Esta solución se basa en estos protocolos, mediante los cuales el receptor avisa al codificador de origen acerca del ancho de banda efectivo en cada instante.

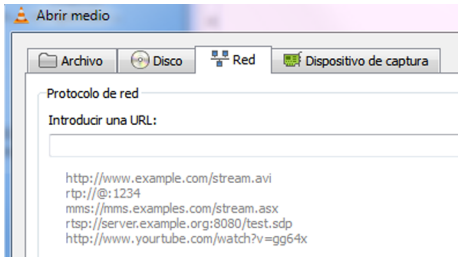
a) Protocolo RTP

RTP[22] es un protocolo de transporte desarrollado para *streaming*, por lo general montado sobre UDP. El transporte de RTP sobre TCP también está definido, aunque se utiliza menos porque presenta una latencia y sobrecarga mayores que UDP (lo que supone una desventaja para aplicaciones de tiempo real). Este incluye datos extras no presentes en TCP, como marcación de tiempo (guarda en cada paquete la hora en que se transmite a partir de un reloj de referencia, lo que facilita la sincronización) y número de secuencia (por si llega desordenado o si se pierde), lo que contribuye a facilitar el transporte en forma continua. También hay datos de control que permiten al servidor realizar el *streaming* a una tasa de flujo definida y correcta.

(22)RTP es la sigla de *real-time transport protocol*.

Lamentablemente, RTP no cuenta con información de sincronización ni con mecanismos nativos para la recuperación de fallos. Hay una serie de códecs estandarizados para ser enviados sobre RTP, varios dentro de la línea MPEG-2.

Ilustración 12. El programa Videolan



Al indicar en la aplicación la dirección de la fuente de *streaming*, hay que indicar el protocolo de tiempo real en que se transmite.

b) Protocolo RTCP

RTCP[23] es un protocolo usado en conjunción con RTP para la recepción de informes estadísticos. Permite, por ejemplo, la detección de fallos en el árbol de distribución de clientes multidifusión y del número de paquetes perdidos y estadísticas de fluctuaciones (*jitter*).

(23)RTCP es la sigla de *real-time transport control protocol*.

Recepción de informes estadísticos

Existen informes enviados por el emisor e informes enviados por el receptor. Por ejemplo, los del emisor son habitualmente el volumen de cuadros enviados o cantidad de bytes, y los informes del receptor presentan los cuadros perdidos o la tasa de cuadros entregados.

Una característica particular de este protocolo es que, para amortizar la sobrecarga[24] que supone consumir bits para el protocolo en vez de para datos, se pueden reunir varios mensajes RTCP y se pueden enviar en un mensaje RTCP compuesto. Los paquetes deben estar compuestos al menos por un mensaje de receptor o emisor y el nombre del participante, y se tienen que enviar de forma periódica sin llegar a consumir el cinco por ciento del ancho de banda de la sesión. Los paquetes RTCP se transportan sobre UDP.

(24)En inglés, *overhead*.

RTCP no permite encriptación, autenticación y autorización. Una actualización con estas funcionalidades es el estándar SRTCP[25].

[25]SRTCP es la sigla de *secure real-time transport protocol.*

c) Protocolo RTSP

[26]RTSP es la sigla de *real-time streaming protocol.*

RTSP[26] realiza control sobre datos multimedia de tiempo real y brinda la posibilidad de interactividad con el reproductor, de forma similar a un vídeo reproductor doméstico. El RTSP permite reproducir, pausar, adelantar, y más. También puede reaccionar a congestiones en la red y reducir el ancho de banda.

Las órdenes estandarizadas en RTSP son:

- *Describe*: con este comando se recibe una descripción del recurso, una lista de los flujos de datos necesarios.
- *Setup*: se indican los parámetros de configuración del flujo de datos.
- *Play*: ejecuta la reproducción del flujo de datos especificado.
- *Pause*: detiene la reproducción del flujo de datos indicado, reiniciable con la orden *Play*.
- *Teardown*: detiene y libera los recursos utilizados, y finaliza el flujo de datos.

El RTSP fue inspirado en el protocolo web (HTTP 1.1), pero con las mejoras de que se puede mantener el estado de la conexión (HTTP no mantiene estado) y de que ambos (cliente y servidor) pueden realizar pedidos. RTSP soporta RTP como protocolo de transporte.

Una de sus utilidades es brindar una forma inicial de escoger el canal de distribución óptimo hacia el cliente. Por ejemplo, algunos clientes pueden tener filtrados en su cortafuegos los paquetes UDP, por lo que el servidor de *streaming* debería proporcionar la posibilidad de escoger entre diferentes protocolos de transporte como UDP, TCP o UDP *multicast*.

4.3.2. Otros protocolos de *streaming*

a) Protocolo RDT

[27]RDT es la sigla de *real data transport*.

RDT[27] es un protocolo propietario para la transmisión de audio y vídeo, desarrollado por RealNetworks en 1995. Al igual que RTP, trabajaba en conjunto con RTSP para el control del *streaming*; dado que no parece que ofrezca mayores ventajas que su equivalente estándar RTP, su continuidad está en entredicho.

b) Protocolo HTTP

A pesar de ser diseñado con fines completamente distintos (es el protocolo por el que podemos navegar por web), el HTTP[28] se utiliza para el *streaming* de audio y vídeo por distintas razones. El principal motivo es que la mayoría de las conexiones a Internet se encuentran protegidas por un cortafuegos o un *proxy*, por lo que la forma más sencilla de hacer *streaming* de un contenido en ese contexto es usando HTTP.

(28)HTTP es la sigla de *hyper-text transfer protocol*.

Lógicamente, presenta una mayor sobrecarga[29] que la familia de protocolos de tiempo real (RTP, RTCP y RTSP), por lo que no está bien considerado en muchos contextos, por ejemplo en el grupo MPEG. Sin embargo, como ya se comentó, es una opción válida para Internet debido a la existencia de cortafuegos/ *proxies* y a que representa una forma simple de afrontar los errores por pérdidas en la red (mediante retransmisión).

(29)En inglés, *overhead*.

El *streaming* sobre HTTP no se encuentra muy estandarizado a pesar de, por ejemplo, ser el más utilizado por las radios en Internet.

Radios en Internet

Podéis ver un directorio de emisoras de radio por Internet en la página web Showtcast Radio Internet.

Actualmente, sobre HTTP se pueden transmitir varios tipos de formatos de vídeo. Además de ES crudos (como los aceptados por RTP), sobre HTTP se pueden enviar MPEG-TS, MPEG-PS, OGG y ASF, entre otros.

Recordad que RTP puede montarse sobre TCP (además de la forma tradicional sobre UDP), por lo que puede recuperar fallos en errores de la red.

c) Protocolo MMS (*Microsoft Media Services protocol*)

MMS[30] es un protocolo propietario para la transmisión de audio y vídeo. Desarrollado por Microsoft, fue abandonado oficialmente en el 2003 y dejó de proporcionársele mantenimiento en el 2008.

(30)MMS es la sigla de *Microsoft Media Services protocol*.

Operaba sobre TCP, UDP o HTTP, negociando cliente y servidor cuál elegir de los tres, o unívocamente el servidor según el estado de la red. Si el cliente no podía negociar una buena conexión utilizando MMS sobre UDP (abreviada, MMSU), entonces intentaba con MMS sobre TCP (abreviada MMST). Si eso fallaba, utilizaba una versión modificada de HTTP para establecer la conexión (abreviada, MMSH). A cada paso iba creciendo el *overhead* y aumentaba su ineficiencia.

4.4. El protocolo MPEG-2 TS

MPEG TS[31] es un protocolo que nos brinda un mecanismo para multiplexar (es decir, combinar en un solo flujo de datos) los flujos de audio, vídeo y datos, para así transmitirlos por una red. Se utiliza mucho en los sistemas de vídeo digital de televisión por cable y TDT.

(31)TS es la sigla de *transport stream*.

Como dijimos en el apartado sobre MPEG-2, un ES[32] es básicamente la salida del codificador, y pueden ser de dos tipos: los flujos de datos de vídeo (VES), y los de audio (AES). Para manejar los diferentes ES, estos se dividen en paquetes de diferente tamaño, según las características de la aplicación y del descodificador. El proceso de partición en paquetes del ES se llama empaquetado, y un ES empaquetado es un PES[33].

(32)ES es la sigla de *elementary stream*.

(33)PES es la sigla de *packetized elementary stream*.

Cuando se utiliza MPEG-TS, varios PES se pueden transmitir de forma conjunta en un mismo *stream* TS. Este proceso se denomina **multiplexado**.

En la mayoría de las aplicaciones se necesita audio y vídeo, y por tanto se requiere el multiplexado de al menos un PES de audio y un PES de vídeo. Los PES transmitidos dentro de un TS se agrupan en programas. Por lo general, un flujo de datos TS transmite varios canales de televisión en simultáneo; cada uno es un programa formado por al menos un PES de vídeo y un PES de audio. Además de los PES, un flujo de datos TS transmite información de control en formato de tablas; por ejemplo, en estas tablas se puede enviar información de la programación presente en el *stream*.

El MPEG-TS no brinda simplemente una forma adecuada de efectuar el multiplexado de los diferentes ES, sino que también ataca el problema de recrear el reloj de la fuente en cada uno de los receptores, para lograr así una correcta descodificación y sincronismo del audio y del vídeo.

4.4.1. Paquetes y tablas del TS

Los paquetes TS tienen un tamaño fijo de 188 bytes. Un flujo TS está compuesto por uno o más programas; para identificar qué PES pertenece a qué programa, el TS envía periódicamente tablas con esa información.

Dentro de la cabecera de los paquetes TS se encuentra un campo llamado PID[34], que identifica el contenido específico de ese paquete TS. El contenido de un paquete TS puede ser:

(34)PID es la sigla de *packet identifier*.

- parte de un PES de audio o vídeo, o

- información de control, conocida como *program specific information* (PSI).

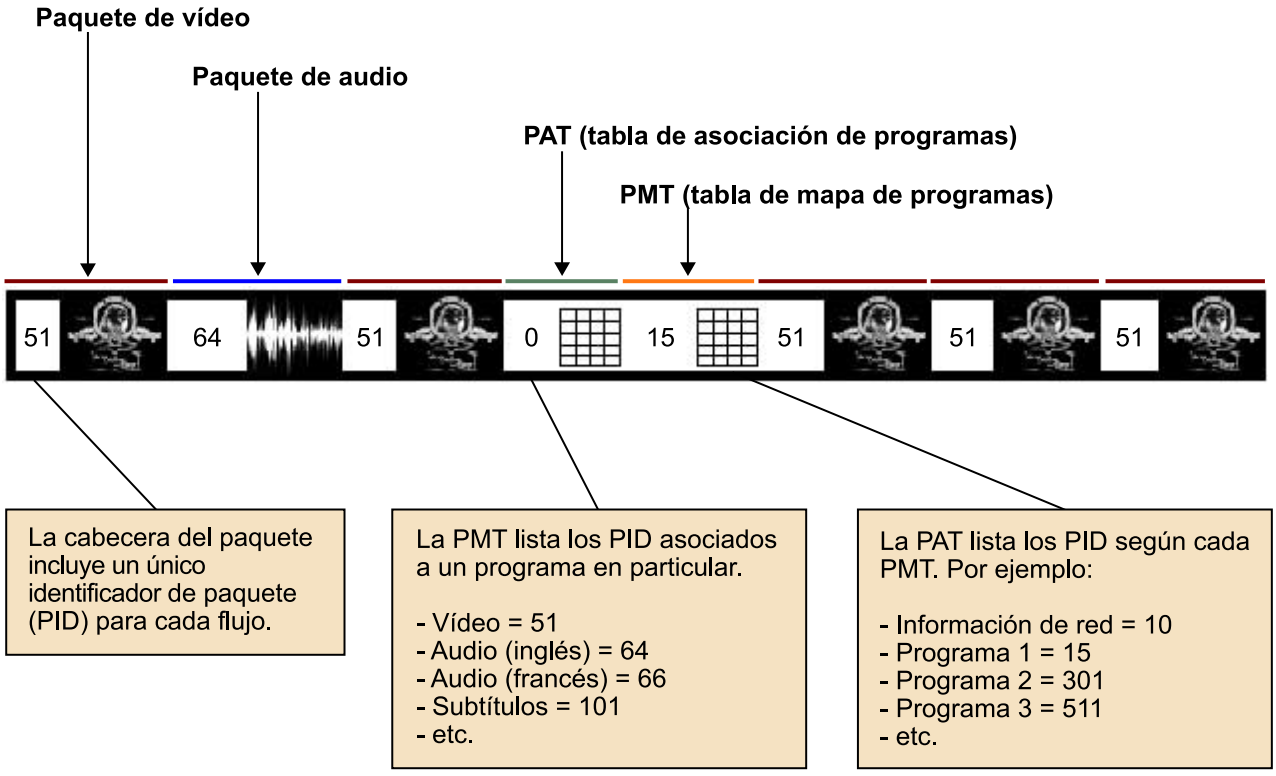
La PSI está compuesta por cuatro tablas diferentes:

- la *program association table* (PAT),
- la *program map table* (PMT),
- la *conditional access table* (CAT) y
- la *network information table* (NIT).

Dentro de estas tablas se encuentra toda la información necesaria para extraer de forma correcta cada flujo de vídeo, audio y datos y seleccionar así los diferentes programas en el receptor. Las tablas PAT y PMT siempre están presentes dentro del flujo de datos, mientras que las otras dos pueden depender del flujo de datos que se transmite.

En la tabla PAT (identificada con PID = 0x0000), se asocia el número de programa con el PID en el que vendrá la PMT para dicho programa. Por ejemplo, en la ilustración 13 se muestra que en los paquetes con PID número 15 van a ser transmitidos los paquetes correspondientes a la PMT del programa 1. A su vez, dentro de la tabla PMT y entre otras cosas, se encuentra qué PID –y por lo tanto, qué ES– están asociados a qué programas. En este ejemplo se define que el PID número 64 es un ES de audio correspondiente al programa número 1.

Ilustración 13. Ejemplo de tablas PAT y PMT



5. Las tecnologías de publicación

Llegado el momento de elegir la manera mediante la cual publicar el material audiovisual por red, hay que plantearse qué tecnología utilizar y el tipo de formato del material. A lo largo de los años han ido mejorando en paralelo tanto las tecnologías de publicación en la red como los formatos contenedores de vídeo que agilicen tal publicación. Ambas líneas de desarrollo están íntimamente relacionadas.

5.1. Tecnologías de publicación

Como tecnologías de publicación audiovisual en Internet podemos definir cuatro, que presentamos en orden cronológico (coincidente con un aumento de la complejidad):

1) Vídeo incrustado

2) Descarga progresiva

3) *Streaming* de vídeo

4) P2P

5.1.1. Vídeo incrustado

Este modelo de plataforma de publicación básica fue la primera aproximación a la integración de audio y vídeo en la Web. Requería un servidor web, por lo que hacía uso del protocolo HTTP clásico, sencillo y con mecanismo de confirmación de los datos enviados por parte del cliente, pensado para la transmisión de pequeños ficheros a un conjunto simultáneo de clientes. El fichero de vídeo "se incrustaba" o integraba en una página web en HTML como cualquier otro fichero mediante las instrucciones *HREF*, *EMBED*, *BGSOUND*, etc.

El cliente solicitaba el fichero de vídeo, que se descargaba temporalmente en la caché del programa navegador que utilizaba, exactamente igual a la de cualquier otro tipo de fichero que pueda publicarse en una página web (imágenes, PDF, DOC, etc.).

La mayoría de los formatos de archivo solo permiten abrir el contenido cuando se ha descargado completamente, como por ejemplo cualquier documento de Office, los antiguos Adobe PDF (versión 5 o anterior), etc. Sin embargo, otros

sí posibilitan iniciar su visualización tras comenzar la descarga y acceder a un mínimo de su contenido, como por ejemplo las imágenes JPEG grabadas en formato progresivo, los documentos PDF modernos, etc.

En aquellos tiempos, los formatos contenedores de vídeo no estaban preparados para ser reproducidos de forma progresiva, por lo que se generaba una espera directamente proporcional a la duración del material. Algo peor que este retardo era que el equipo cliente debía poseer suficientes recursos para cargar el fichero en memoria y poder abrirlo: sin duda, no era válido para dispositivos móviles. A pesar de todas estas limitaciones, este sistema de vídeo incrustado se utiliza actualmente para la publicación de vídeos de duración breve.

Ejemplo: Macromedia Flash SWF (1998)

El formato SWF es el más antiguo y el primero que posibilitó incrustar audio y vídeo en una animación Flash. Al publicar el SWF en una página de un servidor web tradicional, el *plug-in* o complemento Flash del navegador utilizado para visualizarla necesitaba descargar el fichero por completo para reproducirlo, por lo que este sistema solo era interesante para vídeos de duración de unos pocos segundos.

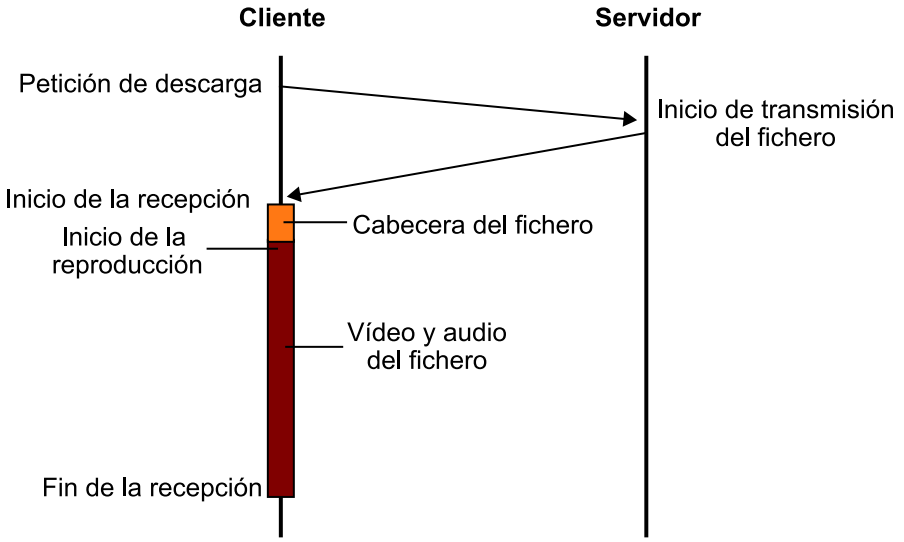
Otras limitaciones claras eran que, para cambiar una sola secuencia de ese vídeo, se debía rehacer el fichero SWF por completo y volverlo a publicar. Más todavía: la carga en memoria de vídeos de larga duración generaba desincronizaciones con el audio tras la reproducción de dos o tres minutos de la película.

5.1.2. Descarga progresiva

Estas limitaciones y la creciente demanda por contenido multimedia en la web llevaron al desarrollo de un mecanismo de distribución de **descarga progresiva** por el que la reproducción es posible al recibir una parte "suficiente" de datos.

La descarga progresiva, además de reducir la espera del cliente, elude los problemas de sincronía audio-vídeo y, sobre todo, simplifica los requerimientos técnicos del reproductor en términos de memoria, lo que abre la puerta a los dispositivos móviles.

Ilustración 14. Esquema de la descarga progresiva de un fichero de vídeo



No todos los formatos son compatibles con este sistema de descarga. El fichero de vídeo debe tener un formato contenedor que sea capaz de soportarlo. Su estructura debe poseer, al menos, una cabecera que prepare el programa reproductor y le indique los datos básicos del tipo de codificación y la duración para preveer las necesidades de memoria que requerirá, y además es necesario que los datos de vídeo y audio estén debidamente combinados, pues si el audio está después de todos los datos de vídeo (o viceversa) no será posible reproducir nada en condiciones hasta no recibir todo el fichero. Tal como se puede ver en la tabla resumen de tipos de contenedores, en el caso de archivos de vídeo, algunos tipos de formatos contenedores están preparados para *streaming*, es decir, para poder ser visualizados sin esperar a su descarga completa.

Ved también

Podéis ver la tabla resumen de tipos de contenedores en el módulo 2 de esta asignatura.

Sin embargo, la técnica de descarga progresiva sigue basándose en el uso de un servidor web genérico, diseñado principalmente para almacenar webs. Cuando un cliente hace una petición, todo el ancho de banda disponible se utiliza para descargar los datos al cliente de la forma más rápida posible.

En condiciones normales, un servidor web envía numerosos ficheros pequeños a un número limitado de usuarios simultáneos, lo que divide el ancho de banda entre los envíos simultáneos; pero al ser de pequeño tamaño, es suficiente. Sin embargo, en el momento en que ha de enviar ficheros de audio y vídeo a un número creciente de clientes simultáneos, el ancho de banda siempre será un recurso escaso.

No hay una distribución inteligente del ancho de banda entre los diferentes clientes: aquellos que descargan un vídeo codificado a 256 Kbps lo recibirán a la misma velocidad que quienes se descargan otro de 2 Mbps, pues el servidor web no analiza previamente las características del fichero.

El tradicional protocolo web HTTP opera además utilizando protocolo de transporte TCP, que asegura la recepción de los datos (tras los envíos debe recibir las confirmaciones de recepción del cliente; si no las recibe, reenvía los datos). Esta seguridad en la transmisión no es eficiente para audio y vídeo, no tanto por el volumen de datos sino porque la pérdida de alguno no afecta sensiblemente a ojos y oídos del cliente, y porque si reenvía datos que se han perdido previamente, no puede asegurarse de que lleguen a tiempo para ser reproducidos.

Ejemplo: descarga progresiva y el formato contenedor FV de Macromedia Flash (2003)

Macromedia incorpora la descarga progresiva al introducir el nuevo formato contenedor FLV (Flash Video). Aparte de las mejoras incorporadas en la codificación de audio y vídeo, el fichero SWF podía asociarse a un fichero externo FLV de vídeo, con lo que el proyecto se independizaría del contenido audiovisual.

Para preveer pausas involuntarias de la reproducción del audio y el vídeo por falta de datos en conexiones lentas, posteriormente Flash redefinió la **suficiencia de datos** como el elemento imprescindible para asegurar la reproducción completa del fichero FLV, haciendo una previsión conservadora según la velocidad de descarga estimada, el peso total del fichero y la duración del vídeo.

Ved también

Podéis ver las mejoras incorporadas en la codificación de audio y vídeo en la tabla resumen de tipos de contenedores en el módulo 2 de esta asignatura.

Por estas razones se definió un tercer mecanismo de distribución, el ***streaming***, añadiendo un servidor adicional al servidor web (o servicio adicional en el mismo servidor web) cuyas características superan las limitaciones indicadas.

5.1.3. *Streaming* de vídeo

El servidor de *streaming* analiza la cabecera del contenedor del material antes de su transmisión, con lo que conoce la velocidad de codificación[35] necesaria para su correcta reproducción, de modo que reserva un ancho de banda idéntico para su *streaming*. Esta gestión inteligente personalizada optimiza los recursos mejor que cualquiera de los otros mecanismos de transmisión vistos.

(35)En inglés, *bit rate*.

Aún más, a diferencia de la descarga progresiva, es posible comenzar la reproducción del *streaming* desde cualquier punto del fichero. La estructura del contenedor de los ficheros preparados para *streaming* se divide en pequeñas unidades, cada una con información relativa a su porción de material audiovisual y con una marca de tiempo que indica su posición en el conjunto del fichero.

Ya que el servidor de *streaming* analiza las características del material que va a transmitir, se condicionan los formatos contenedores que pueden utilizarse en cada tipo de servidor de *streaming*: un servidor Windows Media Server no puede transmitir ficheros en Real Video o Real Audio, pero sí Windows Media Video o Windows Media Audio.

Siempre hay un retraso entre la llegada de los datos al cliente y el inicio de la reproducción al llenar una memoria intermedia que previene de las variaciones en el tiempo de llegada de los paquetes por cambios de congestión de la red.

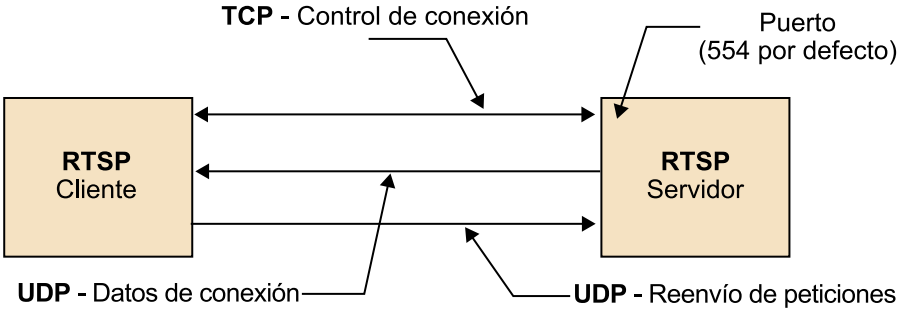
El *streaming* previene además la descarga de los ficheros de audio y vídeo por parte del cliente, y reduce así las posibilidades de copiado ("pirateo"), al enviar los paquetes de datos directamente a la aplicación cliente y descartándolos una vez reproducidos.

Así, utiliza el protocolo UDP, que no retransmite los datos perdidos en aras de un mejor rendimiento en transmisiones de datos en tiempo real, tolerantes a la pérdida de datos. Por su naturaleza, UDP tiene mayor prioridad que TCP en los centros de distribución de tráfico, incluso en redes congestionadas, lo que asegura su servicio. Algunos fabricantes utilizan incluso UDP Resend, un esquema de retransmisión de datos perdidos que solo reenvía esos datos al cliente si hay tiempo para su reproducción (por ejemplo, Microsoft en sus antiguos Windows Media Server).

Entre el cliente y el servidor de *streaming* se crea una conexión paralela a la UDP pero en protocolo TCP para transmitir las órdenes de control del reproductor cliente (*Play*, *Pause*, etc.) al servidor o viceversa para el control remoto de cámaras IP, para enviar un código de tiempo cuadro a cuadro, etc. Este sistema de uso de UDP para datos y TCP para control se denomina RTSP[36].

[36]RTSP es la sigla de *real time streaming protocol.*

Ilustración 15. Conexión RTSP



En aquellas situaciones en las que el protocolo RTSP no está permitido, algunos fabricantes implementan soluciones compatibles con HTTP y su puerto 80, por ejemplo en las redes privadas. Estas soluciones se basan en HTTP Streaming, una versión modificada de HTTP. Y en redes de muy alta velocidad y de alta calidad (de bajas pérdidas de datos) como las redes ópticas, se está volviendo hacia atrás utilizando TCP como protocolo de todo el sistema para asegurar la máxima calidad de reproducción del material transmitido.

Ejemplo: Adobe Flash Media Server (2005)

Adobe da un paso más y plantea una solución más robusta y consistente para distribuir audio y vídeo que utilizar descarga progresiva en un servidor web tradicional, ofreciendo un servidor dedicado (Adobe Flash Media Server) que gestiona de manera inteligente la capacidad de conexión de cada cliente y optimiza el uso de sus recursos: *streaming*, por el que el vídeo comienza antes que con la descarga progresiva, consume menos recursos del servidor y del cliente y facilita el seguimiento de incidencias y la facturación del servicio.

No apuesta por RTSP, sino por un protocolo de *streaming* propietario RTMP o, desde la versión 10.1 de Flash Player, por HTTP Streaming, en este último caso, ligado al uso del contenedor FLV con códec MPEG-4.

5.1.4. P2P

Esta nueva línea en pleno desarrollo rompe con el modelo cliente-servidor utilizado hasta ahora y apuesta por un modelo descentralizado basado en las redes ***peer-to-peer*** (**P2P**), en las que cada cliente puede ser consumidor y, a la vez, publicar y distribuir. Se apoya en alguno de los protocolos de *streaming* explicados.

Un ejemplo ilustrativo: Adobe Flash Player 10 (2010)

Desde esta versión se capacita a los clientes Flash Player a compartir audio, vídeo y datos a través de una conexión directa a otros clientes sin pasar por servidor. Utiliza un protocolo propietario de Adobe, RTMFP, basado en UDF.

5.2. Los contenedores de vídeo para *streaming*

No todos los formatos contenedores son compatibles con *streaming*, tal como se vio en módulos previos. Por ejemplo, el antiguo formato AVI 1.0 (que posteriormente se modificó para aceptar *streaming*) se estructura en bloques de datos numerados denominados *chuncks*, y posee una única cabecera al inicio del archivo donde indica las características del contenido: duración total del vídeo; velocidad máxima de transferencia para su descarga progresiva; número de flujos de audio, vídeo y datos (subtítulos) que contiene; espacio recomendado que hay que reservar en la memoria intermedia del receptor para almacenar *chuncks*; ancho y alto del vídeo en píxeles; frecuencia de cuadro; profundidad de color; códec de vídeo utilizado; códec de audio utilizado, etc.

En principio, parece tener toda la información necesaria para ser reproducido, pero solo desde el inicio. No es posible reproducir a partir de cierto punto si no se ha descargado previamente el fichero hasta, al menos, ese punto. Así pues, es válido para descarga progresiva pero no para *streaming*. Un contenedor de vídeo compatible con *streaming* ha de posibilitar la reproducción desde cualquier punto del fichero. En particular:

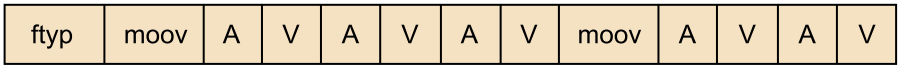
- debe poseer una cabecera que indique las características principales del contenido de audio y vídeo para notificar al servidor de *streaming* qué recursos de red deberá reservar para su transmisión en condiciones óptimas.

- debe estructurarse en bloques independientes de audio y vídeo denominados comúnmente átomos, cada uno con una cabecera que lo caracteri-

ce y posea marcas de sincronía para identificarlo dentro del conjunto del fichero y posibilitar el acceso directo a ese punto.

- cada átomo debe tener la información audiovisual suficiente como para poder iniciar la reproducción en ese punto, es decir, debe abarcar un segmento sincronizado de cada flujo (audio, vídeo y datos). Por lo general, un átomo son varios segundos de audio o vídeo.

Ilustración 16. Estructura de cualquier fichero de vídeo preparado para *streaming*.



Una cabecera con información general y una serie de átomos con contenido fragmentado de audio y vídeo. Si el fichero es muy grande, es habitual añadir algún átomo intermedio.

La ubicación de los flujos en cada átomo debe ser multiplexada, es decir, en su interior se deben combinar alternativamente y de manera inteligente los datos de todos los flujos. Si se almacenase primero el flujo de vídeo, después el de audio y finalmente el de datos, hasta la lectura completa del bloque de vídeo no se podría comenzar a reproducir el vídeo junto con el audio.

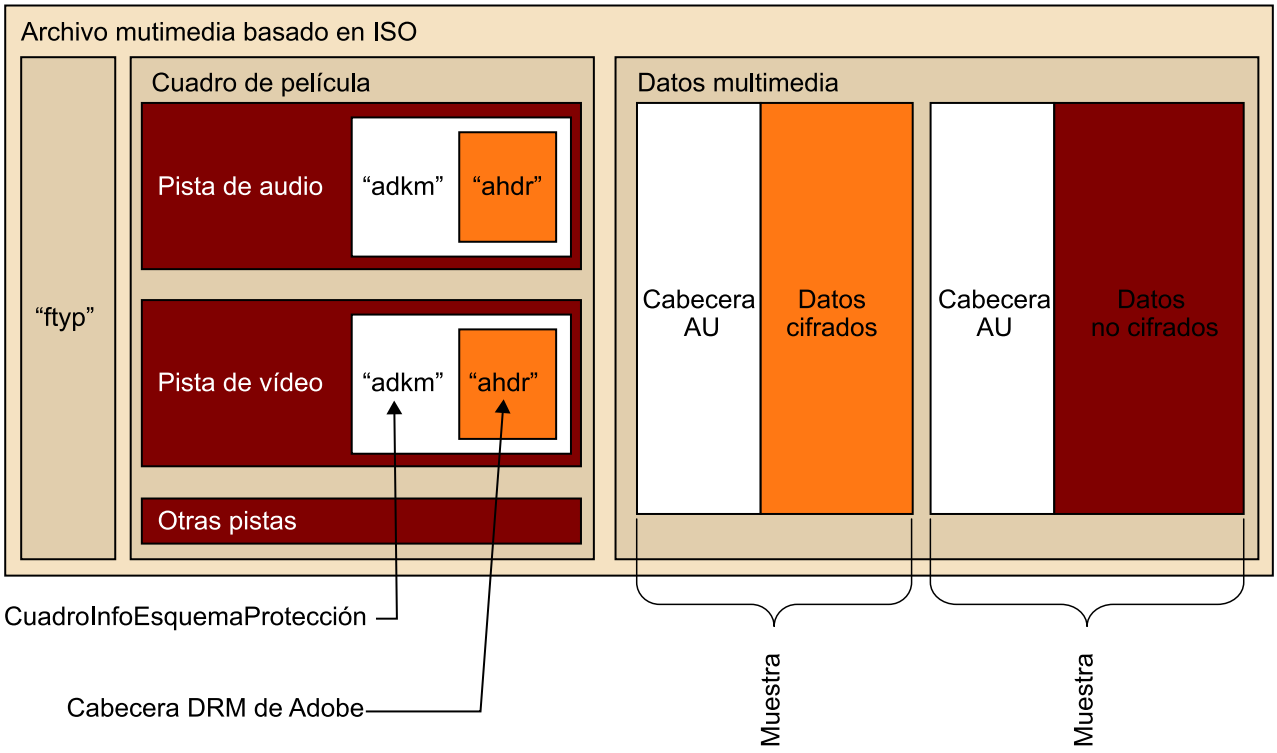
Un ejemplo: Flash Video y el formato base ISO

El contenedor Flash Video es el estándar de *streaming* más extendido actualmente. Es seleccionado como el estándar de referencia para empresas que ofrecen este servicio, como YouTube, BBC Online, Hulu, Yahoo Video, etc. Este contenedor se almacena como fichero de vídeo en los formatos FLV y el mejorado F4V (2010), y puede ser integrado dentro de un fichero SWF.

El F4V, al igual que otros contenedores como MP4, 3GPP, etc., se inspira en una idea más universal, la especificación MPEG-4 Parte 12, denominada por lo general **ISO base**, un modelo global consensuado y bien especificado respaldado universalmente por la organización ISO, la cual lo diseñó a partir del contenedor QuickTime para facilitar la edición, presentación, gestión e intercambio del material por cualquier medio, independientemente del protocolo utilizado.

Centrándonos en el caso F4V, su estructura es básicamente ISO, añadiendo sus propias interpretaciones:

Ilustración 17. Estructura de Flash F4V



Puede verse que es modular, descomponible en objetos muy sencillos cada uno con una misión específica. Cada objeto se denomina caja[37] y no hay nada que no forme parte de alguna, y cada caja puede contener otras subcajas. Sin duda, la idea general de átomo que se comentaba anteriormente coincide por completo con el uso de la caja como elemento básico de este contenedor.

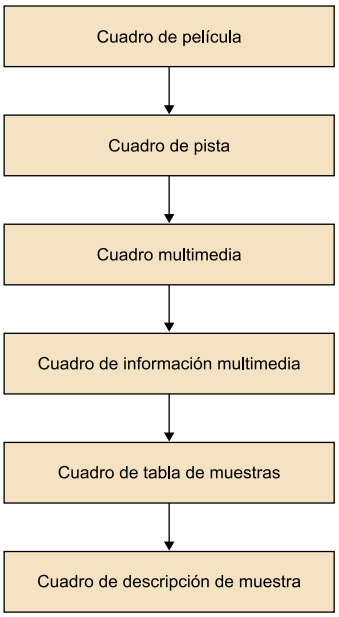
(37)En inglés, *box*.

a) Caja *ftyp*

En primer lugar, hay una caja **tipo de fichero** (*ftyp*) de valor de tres o cuatro caracteres que informa así al programa reproductor del tipo de material que contiene. Una lista puede encontrarse en MP4REG > Registrered types. En ella puede verse claramente que no aparece indicado ningún *ftyp* que indique Flash Video. Adobe se acogió a ISO base pero de manera unilateral, sin registrarse. Para tener una lista de todos los *ftyp* (registrados o no), acceded a la página web "Complete List of all known MP4 / QuickTime «ftyp» designations", donde aparece *f4v* como el de Flash Video.

b) Caja MovieBox *moov*

La caja o átomo *moov* es la cabecera única del fichero *f4v*; puede contener otras cajas que definen en conjunto la estructura de los datos (un TrackBox o *trak* para el vídeo, otro TrackBox o *trak* para audio, MediaBox o *mdia*, etc.; ved gráfico adjunto), y está presente en todos los contenedores que se acogen a ISO base. Incluye un índice de *frames*, *track hint*, útil tanto en *streaming* como en descarga progresiva.



c) Caja MediaData *mdat*

A continuación, y de forma multiplexada o paralela, tenemos las pistas de vídeo y de audio dentro de la gran caja *mdat*. Cada una de las pistas se subdivide en pequeños subbloques, cada uno marcado con un código de tiempo, y pueden estar protegidos bajo contraseña u otro mecanismo de protección o no, todo un guiño a su uso en aplicaciones de valor añadido.

Así, cualquier punto del vídeo y audio no es accesible de manera instantánea: al desear reproducir desde un punto específico, que se sitúa siempre dentro de un subbloque, es imprescindible leer la cabecera de este para confirmar los derechos de acceso al material y desplazarse dentro del subbloque hasta acceder al punto exacto deseado. Eso implica siempre cierto tiempo de análisis hasta poder reproducir el material audiovisual. Sin duda, cuanto más pequeños son los subbloques, menor es el tiempo de acceso, pero mayor es la sobrecarga en bits que supone el creciente número de cabeceras, por lo que resulta necesario un compromiso entre la eficiencia en el acceso y el tamaño del fichero.

Ejercicio: validación de ficheros *flv*

Antes de colgar un fichero FLV o F4V en la red, es recomendable aplicarle la herramienta *flvcheck* (versión Windows o Linux), pues valida su estructura de cajas. Está limitada a ficheros menores de 4 Gbytes y pensada para FLV, pero funciona con la mayoría de los F4V.

Se ejecuta en la línea de comando y su sintaxis es:

```
flvcheck –f "fichero_por_analizar.f4v" -v
```

El resultado es la validación del fichero y, además, un volcado de los valores de cada caja *ftyp* y *moov*. Por ejemplo:

```
C:\Users\Alex>flvcheck -f "Flash10 P2P.f4v" -v

FLVCheck version 1.0 - Utility to validate flv and mp4 media files.
Copyright (C) 2008 Adobe Systems Incorporated. All rights reserved.
www.adobe.com

File: Flash10 P2P.f4v
ftyp/ 28
moov/ 648485
moov/mvhd/ 108
moov/trak/ 373763
moov/trak/tkhd/ 92
moov/trak/mdia/ 373663
moov/trak/mdia/mdhd/ 32
moov/trak/mdia/hdlr/ 44
moov/trak/mdia/minf/ 373579
moov/trak/mdia/minf/vmhd/ 20
moov/trak/mdia/minf/dinf/ 36
moov/trak/mdia/minf/stbl/ 373515
moov/trak/mdia/minf/stbl/stsd/ 171
```

Descarga de la herramienta *flvcheck*

La herramienta *flvcheck* se puede descargar desde la página web Productivity tools and sample downloads de Adobe.

```
moov/trak/mdia/minf/stbl/stsd/avc1/ 155
moov/trak/mdia/minf/stbl/stsd/avc1/avcC/ 69
moov/trak/mdia/minf/stbl/stts/ 117560
moov/trak/mdia/minf/stbl/stsc/ 12372
moov/trak/mdia/minf/stbl/stsz/ 58796
moov/trak/mdia/minf/stbl/co64/ 8256
moov/trak/mdia/minf/stbl/ctts/ 117560
moov/trak/mdia/minf/stbl/stss/ 58792
moov/trak/ 274183
moov/trak/tkhd/ 92
moov/trak/mdia/ 274083
moov/trak/mdia/mdhd/ 32
moov/trak/mdia/hdlr/ 68
moov/trak/mdia/minf/ 273975
moov/trak/mdia/minf/smhd/ 16
moov/trak/mdia/minf/dinf/ 36
moov/trak/mdia/minf/stbl/ 273915
moov/trak/mdia/minf/stbl/stsd/ 103
moov/trak/mdia/minf/stbl/stsd/mp4a/ 87
moov/trak/mdia/minf/stbl/stsd/mp4a/esds/ 51
moov/trak/mdia/minf/stbl/stts/ 32
moov/trak/mdia/minf/stbl/stsc/ 12372
moov/trak/mdia/minf/stbl/stsz/ 253144
moov/trak/mdia/minf/stbl/co64/ 8256
moov/trak/ 423
moov/trak/tkhd/ 92
moov/trak/mdia/ 323
moov/trak/mdia/mdhd/ 32
moov/trak/mdia/hdlr/ 55
moov/trak/mdia/minf/ 228
moov/trak/mdia/minf/nmhd/ 12
moov/trak/mdia/minf/dinf/ 36
moov/trak/mdia/minf/stbl/ 172
moov/trak/mdia/minf/stbl/stsd/ 32
moov/trak/mdia/minf/stbl/stsd/amf0/ 16
moov/trak/mdia/minf/stbl/stts/ 32
moov/trak/mdia/minf/stbl/stsc/ 36
moov/trak/mdia/minf/stbl/stsz/ 32
moov/trak/mdia/minf/stbl/co64/ 32
uuid/ 56013
mdat/ 48922658
11-07-17 20:19:04 Flash10 P2P.f4v passed
```

Si el análisis devuelve un error, esta herramienta solo puede corregirlo si es un fichero FLV y el error está en los *metadata* (si está en la estructura, poco puede hacer). En este caso se aplica:

```
flvcheck -m "fichero_corrupto.flv" -v
```

Lista de mensajes de error y avisos generados por *flvcheck* en la página web: "Checking video files".

Ejercicio: validación de fichero *f4v*

Hay una herramienta específica de aplicación a ficheros F4V que detecta errores y corrige problemas de estructura: Adobe F4V Post Processor.

Descarga de la herramienta *f4vpp*

La herramienta *f4vpp* se puede descargar desde la página web Productivity tools and sample downloads de Adobe.

Entre sus opciones, nos centramos en la no documentada *–c*, que corrige estructura:

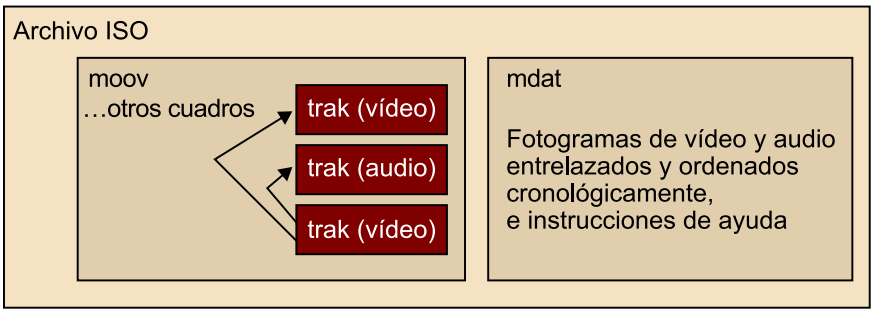
```
f4vpp -c "fichero.f4v"
```

En la documentación asociada de la herramienta se indican los diferentes mensajes de error que puede generar en la detección de problemas.

Otro ejemplo: 3GPP2 (2007)

De modo informativo, el formato 3GPP2 (cuyos ficheros tienen la extensión .3g2) también se basa en el contenedor base ISO, y su estructura es representable de la siguiente manera:

Ilustración 18. Estructura de 3GPP2



Básicamente similar a la de Flash Video, al tener un origen común.