Introducción al MPEG-4

José Ignacio Estévez Damas

1 de febrero de 2007

Resumen

Este documento constituye una introducción al estándar MPEG-4. No pretende ser una descripción exhaustiva, para la cual me remito a las referencias seleccionadas en el cuestionario. La arquitectura de referencia DMIF (parte 6 de MPEG-4) se describe en otro documento.

Introducción.

MPEG-4 fue finalizado entre 1999 y 2000, suponiendo un esfuerzo de mejora sobre los anteriores estándares MPEG-1 y MPEG-2. Como siempre la necesidad de cambio vino impuesta por el deseo de acometer aplicaciones nuevas y más complejas. En este sentido, hay que citar:

- Multimedia en internet.
- Videojuegos.
- Comunicación interpersonal (videoconferencia, videotelefonía).
- Medios de almacenamiento interactivos.
- Servicios de bases de datos multimedia.
- Sistemas de emergencia remotos.
- Vigilancia remota.
- Multimedia en conexiones sin hilos (wireless).
- Aplicaciones de difusión de contenidos multimedia.

Para lograr aproximarse a todos estos campos fue necesario un cambio no sólo cuantitativo respecto a MPEG-2, sino más bien cualitativo, de orden conceptual. Mientras MPEG-1/2 se centran en contenido de audio y video (AV) digital basándose en el *frame*, MPEG-4 se centra en la descripción de escenas mediante el uso de objetos que mantienen entre si, ciertas relaciones de carácter espacial y temporal.

De esta manera MPEG-4 permite entrar de lleno en el campo de la interactividad, individualizando las acciones para cada objeto a niveles como la codificación, decodificación, composición (lugar en el objeto multimedia). Además, MPEG-4 permite integrar objetos de muy diferente naturaleza, logrando por ejemplo incorporar al mismo objeto multimedia video natural y elementos sintéticos. Por lo tanto, frente a la interactividad pobre de MPEG-2 (parar video, continuar video, ir hacia adelante, ir hacia atrás), uno de los avances de MPEG-4 es la interactividad a nivel de objeto. De esta manera, los autores de contenidos MPEG-4 pueden permitir a los usuarios modificar escenas borrando objetos, añadiéndolos, colocándolos en otro lugar, o alterar el comportamiento de los mismos.

Todo este esfuerzo sería vano si no se tuvieran en cuenta las características de los sistemas de comunicación que han aparecido en estos años. Por ello, los desarrolladores de MPEG-4 fueron conscientes dela hetereogeneidad de las redes de comunicación. Es importante recordar en este punto la gran variedad de anchos de banda existentes, así como en general las diferentes calidades de servicio que nos podemos encontrar. En relación a este punto, MPEG-4 se diseñó para permitir la construcción de contenidos *escalables*, pudiendo ser distribuidos en una variedad de plataformas.

Parámetros principales.

MPEG-4 soporta los modos de escaneado de MPEG-2, es decir, progresivo y entrelazado. Además permite una gran flexibilidad en cuanto a resoluciones: desde 8*8 hasta 2048*2048. En cuanto al espacio de color admite monocromo, Y Cr Cb, e incluso Y Cr Cb con un canal de transparencia (canal alfa). Es similar a MPEG-2 en cuanto a las resoluciones para las crominancias: 4:0:0, 4:2:0 y 4:2:2.

MPEG-4 Video está optimizado para una gran variedad de redes y sistemas de comunicación. Así puede soportar diferentes modos: menos de 64 Kbs (conexiones muy lentas), de 64 a 384 Kbs (velocidad intermedia) y de 384 a 4 Mbs (velocidad alta). MPEG-4 puede llegar a trabajar con 1Gbit/s.

Estas prestaciones pueden ser revisadas, para lo que existe el denominado *Requirements Group* encargado de su actualización en función de las necesidades y avances tecnológicos.

Los objetos visuales en MPEG-4

Los objetos visuales en una escena son descritos matemáticamente y su posición puede expresarse en un espacio bidimensional o tridimensional. Los objetos de audio pueden también ser situados a lo largo del video. Una vez colocados los objetos en la escena, el usuario puede cambiar su punto de vista. El cálculo de reposicionamiento en la pantalla se realiza en el decodificador.

El lenguaje de MPEG-4 para describir y modificar dinámicamente una escena se denomina *Binary Format for Scenes (BIFS)*. BIFS tiene similitudes con el lenguage VRML [**?**] (*Virtual Reality Modeling Language*), que es uno de los lenguajes más extendidos para configurar escenas sintéticas 3D. Una diferencia importante es que mientras VRML es un lenguage de texto, BIFS se constituye por códigos binarios por lo que las descripciones son unas 10 o 15 veces más compactas.

Otra diferencia entre BIFS y VRML es que el primero permite adquirir progresivamente la escena (*streamable scene*) mientras que el segundo requiere la descarga completa. Por último BIFS permite situar objetos 2D y 3D, mientras que VRML solo admite objetos 3D.

Ojetos visuales.

MPEG-4 introduce el contenido multimedia en la escena mediante el uso de objetos visuales. Veamos algunos ejemplos:

- El objeto **simple**. Este objeto es un video de forma rectangular de altura y ancho arbitrarios, preparado para tasas bajas de bits. Utiliza codificación simple basada en video object planes (VOPs es la denominación de frame en MPEG-4) I y P.
- El objeto **simple escalable** es una extensión del objeto simple con escalabilidad tanto temporal como espacial.
- El objeto **core** representa una mejora cualitativa respecto al objeto simple. Se introducen VOPs bidireccionales (B-VOP) y pueden tener una

forma arbitraria. La forma arbitraria se consigue incluyendo un canal de transparencia no progresiva (como GIF).

- El objeto **Main** es el objeto de video que da una mayor calidad. La transparencia puede ser progresiva. Soporta además modo entrelazado y sprites.
- El objeto **N-bits** es igual al objeto core, pero permite modificar la profundidad de bits en la capa de luminancia y en las de crominancia (de 4 a 12 bits).
- El objeto **still scalable texture** es una imagen estática con forma arbitraria. La codificación es muy eficiente ya que usa la transformada *wavelet* para el escalado y la descarga incremental.

Los objetos mencionados hasta ahora son los adecuados para representar imágenes naturales. Los objetos de tipo sintético son:

- El objeto **animated 2d mesh** (red bidimensional animada) combina una red sintética con video natural. Este video natural usa las mismas características que el objeto Core. Se utiliza la red bidimensional para mapear el video en puntos de la red que pueden ser animados, cono lo que se consigue deformar dinámicamente el video. Este objeto puede tener forma arbitraria.
- El objeto **basic animated texture** permite usar una red animada sobre la que se proyecta una imagen estática.
- Tenemos el objeto **simple face**, que tiene una herramienta para animación facial. Además MPEG-4 prevé el uso de una interfaz text-to-speech.

Esquema general del transporte de los datos

Streams elementales.

El esquema de transporte es bastante versatil. Se basa en el uso de los denominados *elementary streams* (ES). Los ESs pueden usarse para transportar la información concerniente tanto a descriptores de objetos (ver más abajo) como al propio contenido multimedia. El BIFS que describe las escena también tiene su propio ES.

Un objeto puede tener asociado uno o varios ESs relativos al contenido multimedia. Por ejemplo, un objeto visual con escalabilidad SNR, tendrá un ES para la información base, y otro para la información "de mejora".

MPEG-4 tiene además un medio para informar al sistema acerca de como debe ser decodificado un objeto. Se trata de los denominados descriptores de objetos. Cada descriptor de objeto contiene descriptores de ES, que informan por ejemplo de qué decodificador debe ser utilizado para cada ES. El descriptor de objetos es enviado en un ES especial.

Estructura de capas.

La estructura de los codificadores y decodificadores de MPEG-4 puede entenderse como una estructura de capas.

La primera capa se denomina "capa de compresión". En esta capa se producen los ESs mediante algoritmos de codificación. Los algoritmos de codificación relativos al contenido multimedia se especifican en MPEG-4 Video y MPEG-4 Audio. La codificación de la escena (BIFS) y los descriptores de objetos se especifican en MPEG-4 Sistemas. El problema que se aborda en las siguientes capas que conforman MPEG-4 es el relativo a la organización de la información para posibilitar el transporte de los datos, atendiendo a los requerimientos específicos de los contenidos multimedia, así como a las características de las redes.

Uno de los requerimientos más importantes de los contenidos multimedia es la necesidad de sincronización. Por lo tanto, el codificador / decodificador de MPEG-4 tiene una capa de sincronización. El funcionamiento de esta capa se describe en MPEG-4 Sistemas. En este punto del codificador MPEG-4, los ESs son divididos en paquetes a los que se les añade información temporal relativa a la decodificación y a la presentación.

De este modo, la información temporal que recibe el decodificador incluye la velocidad del reloj del codificador y los *time stamps* (marcas temporales) de los paquetes. Los time stamps pueden ser de dos tipos indicando respectivamente cuándo debe ser decodificado el paquete y cuándo debe ser presentado. Esta dualidad es necesaria, ya que el orden de decodificación no tiene que coincidir necesariamente con el orden de presentación. Un ejemplo de esto es la forma en que muchos videos se codifican utilizando frames previos y posteriores.

El resultado de la capa de sincronización pasa a la capa de distribución. Esta última tiene un multiplexador en dos capas. La primera capa de multiplexado es gestionada conforme a lo especificado por la arquitectura de referencia DMIF (Delivery Multimedia Integration Framework) (parte 6 del estándar MPEG-4). Esta primera capa de multiplexado, permite la agrupación de diferentes ESs con la herramienta FlexMux (definida en MPEG-4 Systms), obteniéndose así los denominados *FlexMux streams*. El multiplexado en esta capa puede ser usado por ejemplo para agrupar ESs con similares prestaciones de calidad, reducir el número de conexiones a la red o disminuir el retraso.

La siguiente capa de multiplexado realiza lo que se denomina multiplexado de transporte. Constituye la especificación de un interfaz para adaptar el sistema de paquetes y señales de control en MPEG-4 a la capa de transporte que se elija para su distribución. Esto implica la multiplexación de los FlexMux streams dando lugar a los denominados TransMux streams. Es importante destacar que MPEG-4 no especifica el protocolo de transporte de los datos, a diferencia de MPEG-2. Por lo tanto, la manera concreta en la que se construyen los streams TransMux depende de la implementación particular del codificador/decodificador MPEG-4 en relación al protocolo de transporte seleccionado.

El creador de contenidos MPEG-4 dispone de una interfaz entre la aplicación y la capa de transporte, denominada DAI y que se especifica en el denominado Delivery Multimedia Integration Framework. DAI es un API necesario dada la libertad de elección en la implementación del codificador/decodificador, proporcionando una puerta de acceso homogénea para el desarrollador de aplicaciones multimedia en MPEG-4.

Funcionamiento general del decodificador

Atendiendo a la codificación MPEG-4 el decodificador funciona del siguiente modo. Los bitstreams de la red pasan a la la capa de transporte como streams

TransMux. Estos streams deben ser demultiplexados en FlexMux Streams. A su vez, los FlexMux streams son demultiplexados en ESs. Estos ESs tienen información de sincronización añadida, por lo que pueden ser procesados por los decodificadores correctos obteniendo así la información de los objetos de audio / video que componen la escena.

El decodificador comienza interpretando una ES que contiene el objeto inicial. En este objeto inicial está el BIFS y los descriptores de objetos que componen la escena. Esta información debe ser combinada con la denominada Stream Map Table que indica la localización de los ESs.

Codificación de objetos de video en MPEG-4

Introducción

Un objeto de video (VOs) es un segmento de video con forma arbitraria que tiene un significado semántico. Una representación 2D de un VO en un instante determinado se denomina *video object plane* (VOP). El VOP queda definido por su textura y su forma. Entendemos por textura la información contenida en sus canales de luminancia y crominancias.

Estructura básica del codificador

En general cuando un VOP es rectangular la codificación del video es similar a la de MPEG-1/2. Debemos recordar que MPEG-4 debe ser capaz de decodificar los *bit streams* de MPEG-1/2 y H.263. EN MPEG-4 se debe codificar la forma (para VOs de forma arbitraria), compensación de movimiento predictiva (vectores de movimiento) y codificación con transformada discreta coseno de los errores resultantes de la compensación del movimiento. Al igual que los estándares previos de MPEG, la codificación de los VOP se realiza a nivel de macrobloques.

La división en macrobloques se realiza de tal forma que sea necesario usar el mínimo número de macrobloques para cubrir la forma arbitraria. Al igual que en MPEG-1 o en MPEG-2, MPEG-4 soporta VOPs con codificación intra (I), VOPs con codificación predictiva (P) y VOPS con codificación bidireccional (B).

Así pues, la estructura general del codificador MPEG-4 es la misma, que la del codificador MPEG-1/2 salvo que se añade un bloque relacionado con la forma del VOP (ya que en MPEG-4 no será siempre rectangular). En el codificador se aplica la DCT a la diferencia entre el VOP actual y el resultado de la compensación de movimiento aplicada sobre un VOP de referencia decodificado (salvo si es un I-VOP, cuya codificación es intra). El resultado de la DCT es cuantizado y codificado con códigos de longitud variable. Como siempre. el resultado del cuantizador es decuantizado, para luego aplicársele la transformada inversa coseno. Esto corresponde a una versión decodificada del error de predicción. Para obtener un VOP, se corrige mediante el VOP de referencia que le corresponda y su vector de movimiento. De esta manera se obtiene el VOP decodificado utilizado como referencia para codificar el VOP actual. Los vectores de movimientos son codificados y enviados al bitstream. Análogamente, el codificador de forma toma el VOP de entrada y obtiene el bitstream que representa

la forma arbitraria del VO.

Codificación de los vectores de movimiento

En el bitstream los datos sobre el movimiento son enviados como vectores de movimientos. En MPEG-4 los vectores de movimiento son codificados de forma predictiva. Además añade algunas técnicas avanzadas de compensación del movimiento como el uso de vectores de movimiento que pueden apuntar fuera del área del VOP de referencia, compensación del movimiento con solape y utilización de cuatro vectores de movimiento por macrobloque.

Como cada VOP puede tener una forma arbitraria es posible que un píxel del VOP a codificar no tenga una contrapartida en el VOP de referencia ya que estaría fuera de los límites del mismo. En ese caso se utiliza un técnica de extrapolación para obtener una versión en el VOP de referencia.

Codificación de la textura

La codificación de la textura se realiza del mismo modo que en MPEG-1/2. Es decir cada macrobloque tiene asociados 4 bloques de 8x8 en la luminancia y dos bloques de 8x8 en las crominancias. Si el macrobloque se extiende fuera de los límites del VOP entonces debe ser rellenado antes de aplicar la codificación basada en la transformada coseno. MPEG-4 también admite una técnica más compleja para enfrentar este problema. Se trata de una versión de la DCT que se puede adaptar a la forma. En este caso se codifican única y exclusivamente los píxeles que pertenecen al VOP, consiguiéndose majores ratios de compresión a costa de una mayor complejidad en la implementación.

Codificación de la forma

MPEG-4 es uno de los pocos estándares que soporta la codificación de la forma. El codificador utilizado se basa en una máscara de bits y se denomina *bitmap-based shape coder*. En este caso, la forma y la transparencia son definidas respectivamente por un canal alfa binario y por un canal alfa de escala de grises.

MPEG-4 tiene herramientas para la codificación con y sin pérdidas de estos canales. Además soporta la codificación intra forma y la codificación entre formas.

Para la codificación intra sin pérdidas, los canales binarios alfa se dividen en bloques de 16 por 16, formando una disposición rectangular. Los bloques que quedan dentro de la forma se asignan opacos mientras que los que quedan en el exterior se asignan transparentes. Los píxeles que quedan en el resto de los bloques (bloques de contorno) son escaneados en modo *raster* (de izquierda a derecha y de arriba abajo) y codificados mediante el algoritmo denominado codificación aritmética basada en el contexto.

El contexto se determina a partir de los 10 píxeles vecinos. Este contexto nos lleva a una tabla de probabilidades desde donde obtenemos el espacio de códigos a usar en codificación aritmética. Para cada bloque de contorno esta técnica se aplica a dos versiones del bloque: el original y el resultante de la operación de transposición. La versión que menos bits produzca es la usada en la codificación.

Para la codificación predictiva, se utilizan VOPs de referencia que pueden ser anteriores y posteriores al VOP actual. La técnica de compensación de mo-

vimiento es aplicada y la diferencia es procesada también mediante codificación aritmética. El contexto se obtiene a partir de 9 píxeles vecinos tanto en el VOP anterior como en el VOP posterior. Los vectores de movimiento también son codificados de forma predictiva.

La codificación con perdidas se obtiene mediante la no transmisión de la diferencia entre formas si se trata de una codificación predictiva o bien por submuestreo del canal binario por un factor 2 o 4 antes de realizar la codificación aritmética.

Por otra parte, el canal de transparencia puede contener niveles desde el 0 (transparente) al 255 (opaco). Este canal es codificado del mismo modo que la luminancia.

Codificación de *sprites*

En MPEG-4 se considera que aquellos VO que son estáticos durante una escena o sus variaciones pueden ser aproximados por transformaciones de dilatación - contracción se codifican de modo diferente y reciben el nombre de sprites. Obsérvese que los sprites son adecuados para transmitir el fondo de las secuencias de video.

Los sprites se codifican del mismo modo que los I-VOP y son almacenados en un buffer del decodificador ya que son usados repetitivamente para reconstruir la secuencia de video. Este tipo de representación puede mejorar notablemente la codificación ya que es necesario almacenar sólo una versión de los mismos, mientras que las transformaciones necesarias sólo requieren de unos pocos coeficientes.

Codificación de los objetos trama en MPEG-4

Una trama o red *mesh* es un particionado de una imagen en piezas poligonales. Las tramas se vienen usando en imagen por computador desde hace mucho tiempo para representar objetos en 2D y 3D. MPEG-4 soporta tramas 2D donde se pueden mapear objetos visuales tanto naturales como artificiales además de imágenes estáticas. Los vértices de las tramas se denominan nodos y pueden ser desplazados para crear patrones de movimiento.