PLATAFORMA PARA LA GESTION Y DISTRIBUCION DE CONTENIDOS PERSONALIZADOS MULTIMEDIA DE ACCESO UNIVERSAL

Francisco Álvarez Vaquero , José L. Sanz González , Jon Andrés Echevarria y Emilio Blanco Moran
E.T.S.I. de Telecomunicación-UPM
Universidad Politécnica de Madrid
Ciudad Universitaria s/n, 28040 Madrid
Telf: 913367366 ext 4046, Fax: 915439652
E-mail: fav@gcs.ssr.upm.es

Resumen

En este artículo se plantea un sistema de preparación y difusión de información multimedia para un entorno UMA (Universal Multimedia Access). La preparación y entrega de contenidos ha de tener en cuenta las características del canal y de los dispositivos de los clientes finales. Se desarrolla un modelo conceptual para describir las condiciones de la red bajo la que se transmite el contenido multimedia, así como la modalidad del dispositivo final destinatario del contenido. Se presenta una arquitectura para la adaptación de datos multimedia a esas condiciones de red y a las características de las terminales finales, bajo las limitaciones impuestas por la naturaleza del flujo de información a transmitir y por las preferencias de los usuarios.. Se mostrará el desarrollo de una aplicación piloto inspirada en servicios de pruebas actualmente existentes basados en plataformas experimentales y abiertas orientadas al mercado turístico y al del aprendizaje interactivo a distancia.

1. Introducción

La figura 1 presenta el diagrama de bloques de alto nivel de la arquitectura del motor UMA que presentamos en este artículo. El sistema UMA se compone de diversos elementos, mostrados en la figura 2. El más importante es el elemento en el que se incluye el motor UMA denominado plataforma UMA.

Dado que dicha plataforma UMA está pensada para llevar a cabo personalización de contenidos como única función, es necesario añadir otros elementos que permitan construir aplicaciones basadas en UMA.

Las funciones asociadas a cada elemento son:

1. Servidor de contenidos, que actúa como fuente para los contenidos.

2. Herramientas de descripción MPEG-7, empleadas para analizar el contenido multimedia disponible en el servidor de contenidos y para generar las descripciones MPEG-7 que serán almacenadas localmente o en el propio servidor de descripciones.

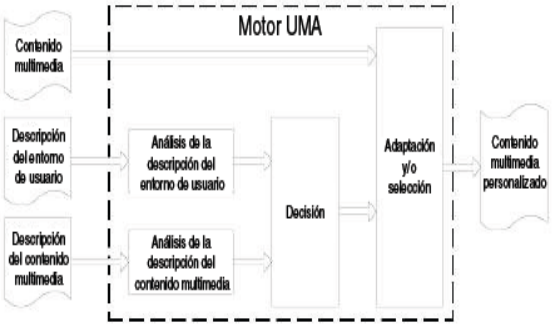

Figura 1. Diagrama de bloques de alto nivel del motor UMA.

3. Servidor de descripciones MPEG-7 que almacena las descripciones MPEG-7 recibidas desde las herramientas de descripción. En esta base de datos se almacena una descripción por cada URL asociada a la localización de un contenido multimedia. Este elemento proporciona la plataforma UMA con la descripción MPEG-7 para cada elemento deseado de contenido.

4. Servidor DIUED MPEG-21 que almacena las descripciones de los entornos de usuario (DIUED) recibidas desde los navegadores UMA. En esta base de datos existe una entrada por cada dirección IP. Proporciona las descripciones de los entornos a la plataforma UMA para cada usuario implicado.

5. Plataforma UMA que se corresponde con la aplicación que implementa la personalización de contenido necesaria para proporcionar la mejor experiencia posible a los sumarios finales, de acuerdo con sus peticiones. Actúa como un servidor de personalización.

Figura 2. Elementos principales de un sistema UMA.

6. Navegador UMA, que incluirá un navegador web convencional para acceder a los contenidos y para

permitir a los usuarios finales administrar su perfil de usuario y entorno a través de los menús apropiados. Esta información debe acabar en un servidor DIUED MPEG-21 o directamente en la plataforma UMA.

El servidor de contenidos, el servidor de descripciones MPEG-7 y el servidor DIUED MPEG-21 son servidores web, en nuestro caso servidores Apache, que implementan el protocolo http permitiendo a otras aplicaciones almacenar contenidos en el servidor a través del comando POST. El navegador UMA, la plataforma UMA y las herramientas de descripción MPEG-7 son herramientas implementadas en el contexto de este Proyecto. La plataforma UMA incluye un motor de personalización, mientras que las dos herramientas restantes, permiten la creación de descripciones para que sean consumidas por la plataforma UMA. El navegador UMA y las herramientas de descripción MPEG-7 se presentarán en el capítulo 7, mientras que la plataforma UMA se presentará detalladamente en los apartados inmediataente posteriores, en lo que a su organización interna se refiere y a lo largo del capítulo 6 en los referente al procesamiento de contenidos.

En principio, la personalización de contenidos puede llevarse a cabo en tres localizaciones diferentes de la red: en el cliente, en el servidor de contenidos y en cualquier punto entre servidor y cliente con la participación de intermediarios. El motor UMA estará activo en cada caso en alguno de estos puntos de la red. En los siguientes apartados presentamos las características, ventajas y desventajas de estos tres enfoques.

2. Plataforma UMA en el servidor de contenidos

Cuando un creador publica sus contenidos en el servidor de contenidos que proporciona ciertas funcionalidades, es natural que pretenda usarlas con el fin de mejorar la presentación de su contenido mediante lenguajes de descripción de páginas más ricos y dinámicos como PHP, ASP de Microsoft o
Java Server Pages de Sun Microsystems. Si el autor dispone de una plataforma UMA, conoce ciertos mecanismos disponibles de personalización de contenidos de antemano. De esta forma, esta configuración proporciona un mayor control a los autores mediante la unión de los procesos de creación y de personalización de contenidos. Esto les permite proporcionar descripciones sobre las adaptaciones del contenido a realizar en distintas circunstancias. Un autor podría instruir al motor UMA sobre las adaptaciones a realizar o no en función de resultados conocidos previamente. Esta característica simplifica en gran medida la labor de la creación de contenidos para entornos heterogéneos, dado que el autor necesita crear el contenido una sola vez, para que la plataforma UMA genere la adaptación pertinente del contenido en cada caso particular.

La figura 3 ilustra la configuración descrita. Dado que la plataforma UMA aloja el contenido de forma local, es muy probable que las descripciones del contenido estén disponibles también de forma local, dado que han sido creadas por la misma entidad: el autor del contenido con una herramienta de descripción.

Esta configuración posee la ventaja de facilitar la adaptación dinámica de contenidos, dado que el contenido está disponible localmente. Es la mejor solución cuando se requiere o se recomienda el uso de un entorno seguro para la administración y adaptación de los contenidos, como puedan ser las aplicaciones de comercio electrónico. En ellas los contenidos se suelen transmitir cifrados. Precisamente por este motivo, en principio sólo el servidor de contenidos es capaz de llevar a cabo la adaptación y el cifrado posterior. Este esquema fomenta el empleo de entornos en tiempo real o con contenidos afluentes, donde el proceso de adaptación se lleva a cabo de forma simultánea con la entrega de los contenidos al cliente.

Además de estas cuestiones técnicas, es necesario considerar la administración de derechos de propiedad intelectual, así como las implicaciones legales y comerciales de este esquema.

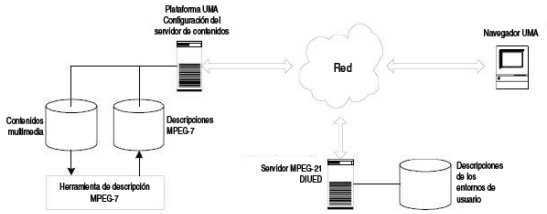

Figura 3. Plataforma UMA en el servidor de contenidos.

3. Plataforma UMA en un servidor proxy

En la adaptación de contenidos basada en un intermediario, el usuario cursa las peticiones contra su proxy quien propaga esta petición hasta el servidor de contenidos de parte de usuario final. El servidor de contenidos responde transmitiendo el contenido sin adaptar (suponemos que es incapaz de ello en este esquema) y la plataforma UMA situada en el intermediario lo recibe, lleva a cabo la personalización del contenido si ello es necesario y envía el contenido resultante hasta el cliente final.

Es lógico y habitual considerar que el ancho de banda entre el servidor de contenidos y el intermediario es mucho mayor que el que se establece entre el intermediario y el cliente final. Por ejemplo, el proxy puede ser el encargado de transformar contenido web existente basado en HTML y en su descripción MPEG-7. La figura 5.4 muestra esta configuración. Después de recibir el contenido, la plataforma UMA carga la descripción MPEG-7 y el DIUED MPEG-21, y decide qué tipo de personalización lleva a cabo basándose en esta información.

Una arquitectura basada en un intermediario como la expuesta simplifica la localización de la plataforma UMA en términos geográficos cercanos a los clientes, dado que, por ejemplo, sería posible emplear la infraestructura de un proveedor de acceso para dar servicio a múltiples usuarios. Este esquema implica que los contenidos de los servidores de contenidos permanecen estáticos dado que un intermediario da servicio tomando los contenidos de múltiples servidores eventualmente, dando lugar a un aprovechamiento técnico y económico de la infraestructura mayor que con el esquema anterior.
Un autor puede tener escaso o ningún control sobre las operaciones de personalización de sus contenidos llevadas a cabo, por intermediarios que presten servicios a los usuarios finales.

4. Plataforma UMA en el cliente

En la práctica existen situaciones en las que las terminales finales poseen suficientes capacidades para participar en el proceso de adaptación de contenidos, seleccionando el tipo de variación más adecuada de entre las disponibles o bien en algunos casos llevando a cabo tareas de adaptación de baja complejidad. A pesar de que estas situaciones no son hoy en día comunes en el ámbito de las terminales móviles, este esquema posee una ventaja inherente sobre las dos configuraciones precedentes: la privacidad de los datos del usuario, que se garantiza dado que la información sobre las características de las terminales, preferencias y descripción del entorno no tienen porque salir de la terminal final. La figura 5.ilustra esta configuración. El servidor de contenidos proporciona el contenido y el servidor MPEG-7 proporciona las descripciones a las terminales que incorporan el motor UMA, Basándose en la descripción del contenido, la terminal decide la variación más adecuada conociendo las condiciones del entorno de usuario.

La descripción de los contenidos juega un papel muy importante en este esquema. La descripción es transmitida hasta la terminal, haciendo uso de ancho de banda y memoria, y la terminal tiene que analizarla para tomar la decisión óptima sobre la personalización a llevar a cabo. Precisamente porque la decisión se lleva a cabo en las terminales finales, donde los contenidos no están disponibles, la descripción del contenido debe ser lo suficientemente explícita para que el motor UMA decida la mejor personalización de entre las posibles a aplicar.

Está claro que la descripción generada para un terminal final con unas capacidades limitadas tiene que ser particular y orientada al entorno en el que va a ser consumida. Por un lado completa y por otro lado simple siendo un compromiso difícil de alcanzar. Esta configuración consume un mayor ancho de banda porque multiplica las transmisiones de información

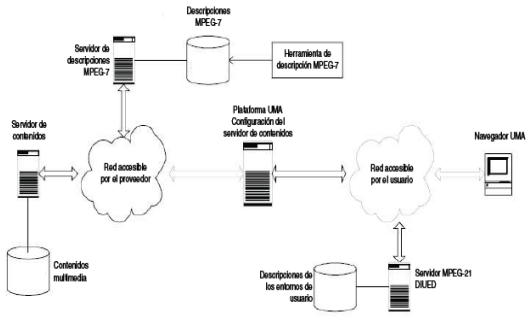

Figura 4. Plataforma UMA situada en un servidor *proxy*

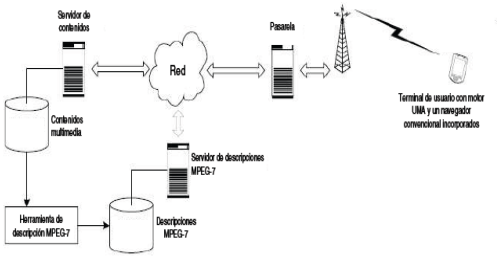

Figura 5. Motor UMA en el terminal de usuario

necesarias para que la descripción de los contenidos alcance al terminal final.

5. Diseño de la plataforma UMA

La plataforma UMA implementada puede ser configurada como un servidor de contenidos o como un servidor basado en un intermediario. Varios módulos componen la plataforma UMA, el motor UMA, los módulos requeridos por el motor UMA para procesar el contenido y sus descripciones, y los módulos que actúan de interfaz tanto con el administrador como con la red.

La figura 6 muestra los módulos principales de una plataforma UMA.

1. Interfaz gráfica de usuario, GUI, Graphical User Interface que permite mostrar y administrar el módulo de interfaz con la plataforma. Depende en gran medida de la plataforma y sólo el administrador de la plataforma debería tener acceso a ella.

2. Administrador de la interfaz de red, NIM, Network Interface manager responsable de las comunicaciones entre la plataforma UMA y el resto de los módulos componentes del sistema. Es también responsable de obtener descripciones. Proporciona todas las funciones para interactuar con la red, y permite a la plataforma configurarse como servidor de contenidos o bien como intermediario.

3. Motor UMA responsable de la personalización de contenidos. Contiene los módulos necesarios para llevar a cabo estas tareas, ejecutándose de forma secuencial y paralela según los casos.

4. Librería DIUED MPEG-7 de software, que ofrece la posibilidad de leer y escribir descripciones en formato

MPEG-7. Dado que tanto DIUED y MPEG-7 emplean las mismas tecnologías básicas para las representaciones de las descripciones, es decir XML, esta librería se emplea indistintamente como interfaz para los programadores de MPEG-7 y de DIUED MPEG-21.

5. Librería XML que es el módulo responsable de analizar las expresiones XML, con el fin de leer las descripciones MPEG-7 y DIUED.

6. Librería para el tratamiento de imágenes que proporciona un conjunto de herramientas para procesar imágenes. La librería Image Magick1 junto con los algoritmos de adaptación de temperatura empleados por la plataforma para llevar a cabo las tareas de procesamiento de imágenes.

7. Librería para el tratamiento de vídeo que proporciona un conjunto de herramientas para procesar vídeo. Mencionar la librería usada en la implementación ISO MPEG-2/MPEG-1.

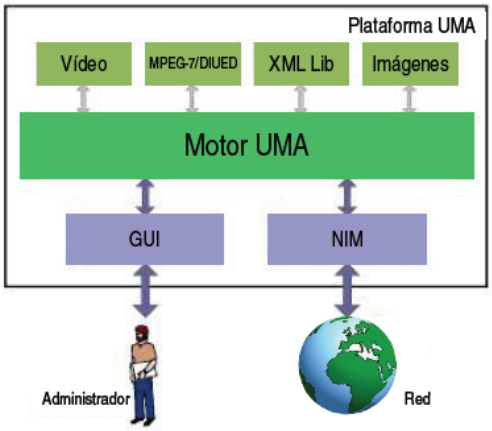

Figura 6. Módulos que constituyen la plataforma UMA.

El modelo de estructuras de datos para el motor UMA presentado en la figura 7 muestra la organización de todos los datos relacionados con cada petición individual de un usuario. Este diagrama ilustra la forma en la que los objetos se relacionan entre si; todas las estructuras de datos se relacionan mediante un objeto Request. Este elemento es el tipo básico de datos empleado como la base sobre la que organizar las relaciones entre la información relativa a cada petición individual. Este tipo de datos es instanciado cada vez que el NIM recibe un comando HTTP solicitando contenido a través del puerto correspondiente. El resto de objetos se referencian a través de este. Así, cuando un mensaje pasa de un módulo a otro, la referencia a Request es pasada igualmente, asegurando que solo un módulo en cada momento mantiene la referencia al objeto Request. Este procedimiento puede ser considerado como un testigo o token que pasa entre los componentes del sistema habilitando el tratamiento de la información.
El objeto User agrega la información sobre el usuario que cursa la petición de contenido multimedia. El objeto User almacena la dirección IP y puerto del terminal de usuario. Una referencia a la descripción del entorno de usuario se almacena en el campo DIUEDurl; cuando esta descripción ha sido procesada, una referencia al resultado del análisis se incluye también en el DIUED. Tras completar la petición, el objeto User y MMContent se almacenan en las entradas de tablas caché para ser empleadas en un futuro en el mismo contexto o en contextos similares.

El objeto MMContent reúne la información relevante sobre el contenido multimedia en si. la tabla 5.3 muestra sus atributos. Cada una de las instancias de este objeto hace referencia a cierto tipo de contenido multimedia referenciado mediante contentURL. Los datos que forman el contenido multimedia son accedidos mediante una referencia interna ContentData. El tipo de medio que caracteriza al contenido multimedia se identifica mediante contentType. La descripción del contenido se localiza en la URL designada por descriptionURL. Cuando la descripción del contenido ha sido procesada, las referencias internas creadas a los descriptores y a los esquemas de descripción se almacenan en los campos correspondientes.

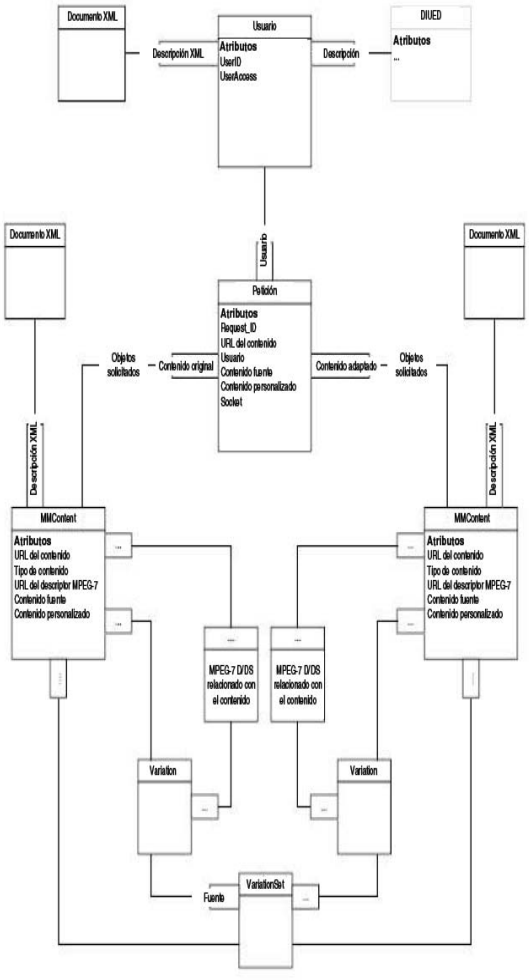

Figura 7. Modelo de objetos.

InfoStatus. Cada módulo emplea este tipo de datos para informar al PM/M de que un mensaje está siendo enviado desde un módulo hasta otro. El PM/M reenvía el mensaje infoStatus a la interfaz gráfica de usuario. Esta clase posee una copia de la información relacionada con el mensaje que esta siendo enviado en un momento dado entre dos modulos

VariationSet. Esta clase se corresponde con la correspondiente clase VariationSet de MPEG-7 .

La clase Variation se corresponde con el esquema de descripción Variation de MPEG-7.

DIUEDConsumer. Esta clase de datos se corresponde con el descriptor DIUED. Se trata de
uno de los descriptores implementados en las librerías MPEG-7/DIUED. Es un esquema que hace uso de algunas de las herramientas de MPEG.7, como las relativas a las preferencias de los usuarios, y comparte un formato común XML, por lo que las librerías sobre MPEG-7 y DIUED se han implementado conjuntamente.

6. Escenarios de eventos

6.1. Escenario 1: Petición del usuario

Cuando el NIM recibe una petición de contenidos a través del protocolo HTTP, envía un mensaje MSG_REQUEST al módulo URP. El DIUED base y las diferencias recibidas con la petición se emplean para procesar la petición. El módulo URP analiza la petición para comprobar si está presente en la base de datos de la caché de DIUED del usuario peticionario, del contenido en sí o bien en la caché de las descripciones MPEG-7. Si no aparecen las descripciones, URP envía en mensaje MSG_GET_MMCD o MSG_GET_DIUED al módulo NIM para gestionar cada caso según corresponda y obtener las descripciones ausentes. Este escenario se ha representado en la figura .8.

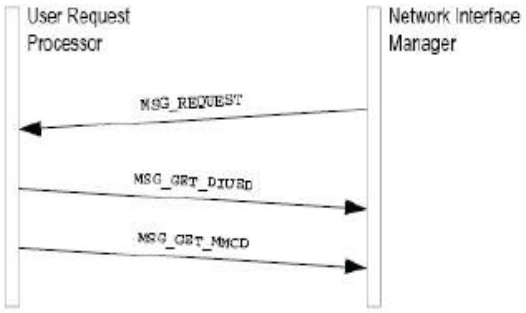

Figura 8. Recepción de una petición HTTP.

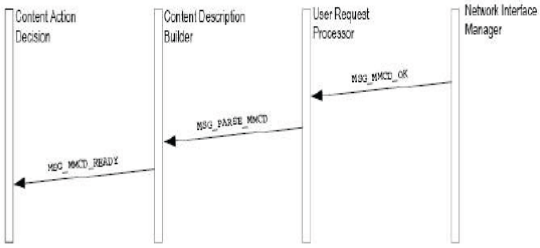

Figura 9. Recepción de una descripción de contenidos

6.2. Escenario 2: Proceso de la descripción del contenido multimedia

Una vez que se dispone de la descripción MPEG-7 relevante para el contenido tratado en un momento determinado, obtenido mediante una URL o desde la caché local, la URP recibe un mensaje, que reenvía hasta MMCDBuilder la descripción y las cabeceras HTTP mediante un mensaje MSG_PARSE_MMCD. MMCDBuilder analiza el contenido de la descripción en memoria y comprueba su validez en términos sintácticos y semánticos para garantizar su buen uso. Cuando la labor de análisis finaliza, MMCBuilder envía un mensaje MSG_MMCD_READY al módulo CAD para que éste, a su vez, pueda analizarlo y decidir las acciones de personalización a llevar a cabo. Este escenario se muestra en la figura 9.
Si la descripción MPEG-7 no está disponible en forma de URL o en la caché, se cursa un mensaje
MSG_NO_MCD en vez de un mensaje MSG_MMCD_OK; el resto del proceso es similar pero únicamente las cabeceras HTTP se procesan para obtener información del contenido.

6.3. Escenario 3: Proceso de la descripción del entorno del usuario

Cuando se obtiene el descriptor DIUED, se emite un mensaje MSG_DIUED_OK que llega al módulo
URP que reenvía la descripción y las cabeceras HTTP al constructor DIUEDBuilder con el mensaje asociado MSG_PARSE_DIUED. DIUEDBuilder analiza las descripciones en memoria y comprueba su validez. Cuando el análisis del descriptor termina, DIUEDBuilder envía un mensaje al módulo CAD para que este módulo pueda analizar el contenido y determinar las acciones de personalización de contenidos a tomar. Este escenario se muestra en la figura 10. En el caso de que el DIUED no se encuentre disponible en el servidor DIUED MPEG-21 o bien en la caché local, un DIUED por defecto generado con valores predefinidos se usa en su lugar y el resto del proceso continúa sin variación.

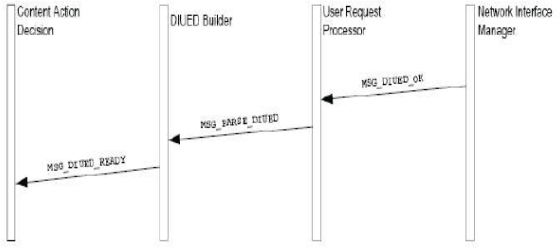

Figura 10. Recepción de la descripción del entorno de usuario.

6.4. Escenario 4: Proceso del contenido multimedia

Cuando el módulo CAD recibe los mensajes MSG_MMCD_READY y MSG_DIUED_READY, se inicia el proceso de decisión. Después de tomar las decisiones correspondientes, el CAD envía la lista de acciones de personalización a realizar al módulo CC junto con el mensaje MSG_CONTENT_ACTIONS. El módulo CC envía un mensaje MSG_GET_CONTENT al módulo NIM
con el fin de que obtenga la variación del contenido indicada. Una vez obtenido este contenido, NIM lo anuncia mediante un mensaje MSG_CONTENT_OK que envía al módulo CC, de forma que la personalización sobre el contenido puede dar comienzo en el caso de que sea posible y requerido.
Antes de que la personalización del contenido dé comienzo, CC envía un mensaje MSG_CONTENT_READY al módulo NIM, de forma que ambos módulos pueden iniciar sus respectivas tareas en paralelo. Mientras que el contenido personalizado con las variaciones indicadas se está generando en el módulo CC, el módulo NIM puede enviar aquellas porciones de contenido que vayan quedando disponibles para su transmisión. Cuando se completa la operación de personalización del contenido, se indica mediante un flag activado en el contexto de la petición, que informa al NIM de que se ha completado el servicio a la petición previamente cursada.
Cuando el NIM termina de enviar el contenido al usuario, envía un mensaje MSG_REQUEST_COMPLETE al módulo PM/M de manera que en este momento se pueden actualizar
las tablas de caché con la información que eventualmente pueda ser añadida a estos registros.

7. Diseño de los módulos de la plataforma UMA

El diseño de los módulos que componen la plataforma UMA requiere identificar cada funcionalidad llevada a cabo por la plataforma y la definición de las clases y objetos implicados. Después de establecer esta arquitectura básica, se establece el tránsito de mensajes entre los módulos.

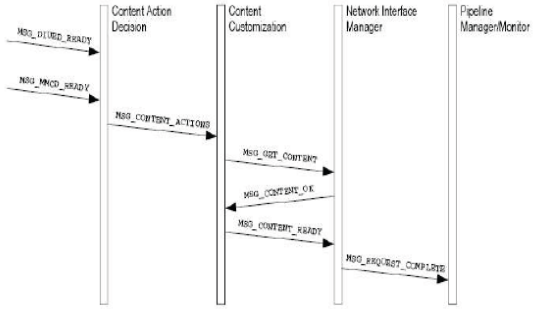

Figura 11. Final del proceso de análisis del DIUED y de la información MPEG-7.

7.1. GUI

Cierto conjunto de funcionalidades, asociadas con el control y la administración de la plataforma UMA, están disponibles para el usuario o administrador de la plataforma a través de la interfaz gráfica de usuario. El resto del motor UMA es independiente del GUI dado que se ejecutan sobre hilos independientes que se comunican entre sí mediante mensajes especialmente implementados para ello. Cuando se inicia la operación de la plataforma UMA, el GUI es el único módulo en ejecución. El usuario debe iniciar la ejecución del motor UMA a través de la interfaz. PM/M, a su vez, lanza a ejecución al resto de los módulos.

7.2. NIM

El módulo NIM es el encargado de interceptar las peticiones de los usuarios para que el motor UMA sea capaz de procesarlas. Este módulo fue originalmente diseñado para proporcionar integración entre la red y el motor UMA, de forma que sea posible realizar pruebas en un escenario real. Con este módulo incorporado a la configuración, los modelos de servidor de contenidos o de servidor intermediario de contenidos resultan fáciles de implementar, dado que el NIM obtiene los contenidos de un disco local o bien a través de una URL. En caso de haber prescindido de él los contenidos se obtendrían exclusivamente vía disco local, y la interfaz de usuario tendría una participación mayor en el proceso, lo que no resulta deseable, desde el punto de vista del encapsulamiento de las funciones asignadas a los distintos módulos que componen la plataforma UMA. Esencialmente, el módulo NIM se constituye con una implementación parcial del protocolo HTTP para permitir al motor UMA realizar las siguientes funciones:
1. Recibir las peticiones de los usuarios a través de comandos GET en un puerto predeterminado y en una dirección IP asociada.
2. Recibir la información referente a los entornos de usuario mediante comandos GET en un puerto predeterminado y en una dirección IP asociada. Esto permite a los navegadores UMA enviar el DIUED directamente hasta la plataforma UMA.
3. Obtener contenidos desde cualquier URL accesible cuando otro módulo los solicita.
4. Obtener descripciones desde cualquier URL accesible cuando otro módulo las solicita.

5. Enviar el contenido adaptado de vuelta hasta los usuarios finales una vez que el módulo CC ha indicado la disponibilidad del contenido mediante el mensaje predeterminado.

Con el fin de integrar el motor UMA en cualquier otro sistema que requiera personalización, la modificación necesaria para adaptar la plataforma a las nuevas condiciones, afecta únicamente a este módulo.

7.3. PM/M

Este módulo tiene como principal función asignada la de administrar y monitorizar al motor UMA.

Cuando otro módulo envía o recibe un mensaje, PM/M recibe un mensaje informando sobre este evento. Haciendo uso de este método, PM/M posee la capacidad de registrar el estado de cada petición y módulo en el sistema. Todos los mensajes registrados le son entregados al GUI, para ser tratados o presentados como el administrador de la plataforma establezca. Las tareas de este módulo se pueden descomponer en las siguientes tres:

1. Crear o lanzar a ejecución módulos de la plataforma.
2. Detener la ejecución de módulos de la plataforma.
3. Administrar el contenido de las tablas caché.

Cuando el NIM completa el procesamiento de una petición y lo indica mediante el preceptivo envío del contenido tratado hasta el usuario final, PM/M recibe información sobre el contexto de la petición. Una referencia a los datos generados u obtenidos por la plataforma para satisfacer una petición pueden, por tanto, ser insertados en las tablas caché para hacer un uso posterior. Sólo el módulo PM/M puede escribir en las tablas caché. Los demás módulos de la plataforma poseen permisos de sólo lectura. En cada momento se mantienen dos tablas caché.

1. Caché de usuarios conteniendo objetos del tipo User correspondientes a cierta dirección IP, tal y como se muestra en la figura 12.
2. Caché de contenidos conteniendo referencias a los contenidos procesados, ordenados según la URL y descripción asociadas a cada uno de ellos, como se muestra en la figura13.

La tabla de usuarios se emplea para almacenar las descripciones contenidas en el DIUED base obtenida desde los servidores DIUED MPEG-21. Cada petición, como ya se ha descrito, envía un DIUED diferencial, con respecto al DIUED base almacenada en las tablas. La tabla de contenidos almacena objetos del tipo MMContent que hacen referencia a las fuentes del contenido, a la descripción del contenido, así como a las eventuales variaciones del contenido que puedan existir, como puede apreciarse en la figura 13. Los objetos del tipo MMContent contenidos en esta tabla, hacen referencia a las fuentes de los contenidos, exclusivamente. Las referencias a los objetos del tipo MMContent relacionadas con variaciones de los contenidos no se incluyen directamente en esta tabla, sino que se emplean las referencias Variation, que apuntan a los resultados de las variaciones previamente aplicadas a los contenidos. La figura 13 ilustra estos casos, donde una misma fuente de contenido tiene asociadas diversas variaciones que pueden ser localizadas mediante los objetos y el procedimiento descrito.
El módulo PM/M puede implementar también una función de monitorización o de watchdog, dado que a través de los mensajes infoStatus puede conocer el estado de cada módulo de los que componen la plataforma.

7.4. URP

El módulo URP da proceso a las peticiones de los usuarios y es e responsable de usar las tablas de caché para la mejora del rendimiento. Este módulo posee también la tarea asignada de reunir toda la información a cerca de los usuarios que cursan las peticiones y acerca de las URL de las peticiones.
Las funciones, más esquemáticamente, son:
1. Comprobar si el contenido al que las peticiones hacen referencia, así como las descripciones de los entornos de usuario, se encuentran almacenadas en las tablas caché.
2. Solicitar al módulo NIM la obtención de las descripciones de los contenidos multimedia relevantes a partir del servidor de descripciones MPEG-7.
3. Solicitar al módulo NIM la obtención de las descripciones de los entornos de usuario relevantes a partir del servidor de descripciones DIUED MPEG-21.

4. Si resulta necesario, examinar las cabeceras de las peticiones GET de los usuarios para extraer información acerca de estos.

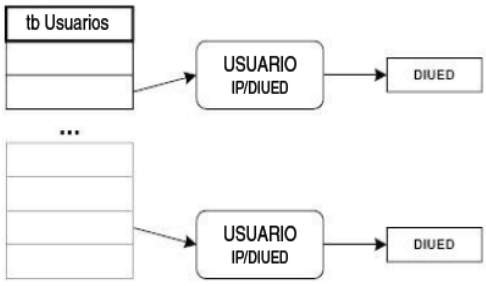

Figura 12. Tabla de usuarios ordenada por dirección IP.

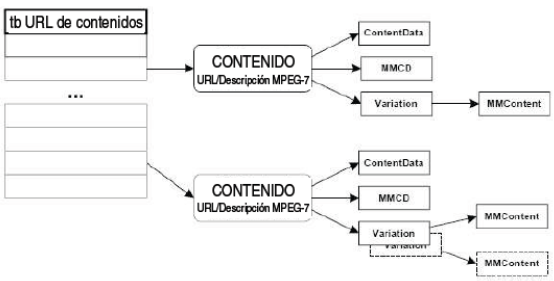

Figura 13. Tabla de contenidos ordenada por URL.

5. Si el contenido no se encuentra disponible, solicitar al NIM la obtención de las cabeceras HTTP del contenido, y analizar estas cabeceras para extraer información acerca de los contenidos.
6. Enviar la descripción del entorno del usuario obtenida al constructor DIUEDBuilder.
7. Enviar la descripción del contenido multimedia obtenido al constructor MMCDBuilder.

7.5. MMCDBuilder

Este módulo analiza y valida las descripciones MPEG-7. Para ello hace uso de las librerías MPEG-7/DIUED. Las principales funciones de este módulo son:
1. Generar la descripción del contenido para analizar su validez sintáctica.
2. Analizar las descripciones del contenido para comprobar su validez semántica.
3. Instanciar las estructuras internas con los descriptores del contenido relevantes para el análisis llevado a cabo por el motor UMA.
El resultado de este proceso consiste en un árbol de objetos almacenado en memoria conteniendo la información presente en la descripción MPEG-7. Una referencia interna a las estructuras se le pasa al módulo de decisión sobre los contenidos.

7.6. DIUEDBuilder

Este módulo genera y analiza la validez del DIUED. Para ello hace uso de la librería MPEG-7/DIUED. Las funciones de este módulo son:
1. Generar las descripciones de los entornos de usuario, para comprobar la validez sintáctica.
2. Analizar las descripciones de los entornos de usuario para comprobar su validez semántica.
141 3. Instanciar las estructuras de datos internas con los descriptores de los entornos de usuario, relevantes para el análisis a realizar por el motor UMA.
Cuando no se dispone de un DIUED para un terminal en concreto, el constructor DIUEDBuilder examina las cabeceras HTTP incluidas en la petición de contenidos con el fin de generar internamente un DIUED válido. Es en estas situaciones las informaciones relativas a las terminales de usuario pueden ser muy difíciles de obtener, dado que no todas las terminales envían suficiente información.
El resultado de este proceso es el de un árbol de objetos almacenado en memoria y conteniendo la descripción DIUED. Una referencia interna a las estructuras de datos apropiadas se le pasa al módulo de decisión sobre contenidos.

7.7. CAD

Este módulo está relacionado con la decisión acerca del tipo de operaciones de adaptación que deben ser hechas sobre los contenidos multimedia. Este módulo sabe exactamente qué métodos de adaptación son los disponibles, en las etapas segmentadas posteriores, y decidir las operaciones sobre los contenidos basadas en estos métodos disponibles, después de comparar las descripciones de los contenidos con las descripciones de los entornos de usuario.

7.8. CC

Este módulo de personalización de contenidos es el responsable de llevar a cabo las operaciones de adaptación sobre los contenidos, previamente indicadas por el CAD. Ejecuta tareas relacionadas con la señalización e integra varias librerías que proporcionan funciones para la manipulación de diferentes tipos de contenidos.
Este módulo es también el responsable de solicitar al NIM la obtención de las variaciones del contenido indicadas con el fin de obtener las variaciones necesarias en el momento en el que los contenidos están disponibles.

8. Modelo de tratamiento de contenidos

Hasta este momento, se han presentado la organización interna de la plataforma UMA y las funciones desempeñadas por cada uno de los módulos que la componen. Desde un principio, se considera un requisito de la plataforma la capacidad de manipular contenidos multimedia de tipos básicos, como texto, audio, imágenes y vídeo. Este capítulo describe el proceso de adaptación de contenidos llevado a cabo por los módulos de la plataforma. Asimismo, se presentan dos módulos específicos para la adaptación de imagen y vídeo.

Con respecto a la adaptación de contenidos, el modo de funcionamiento más relevante del motor UMA, es el modo normal y será el empleado por defecto. Cuando se inicia el proceso de decisión sobre la personalización, el tipo de medio tratado es analizado y, dependiendo de los resultados de este análisis, las correspondientes funciones de decisión son invocadas. En este contexto, se han implementado funciones para el tratamiento de imagen y vídeo.

Todas las decisiones tomadas por cada una de estas funciones se almacenan en 2 variables accedidas por el módulo de personalización de contenidos. La primera variable, llamada Action Stack, es empleada para almacenar las acciones de la actuación de contenido. La segunda variable se llama Action_Parameter_Stack y se emplea para almacenar los parámetros relacionados con cada una de las acciones y adaptación de contenidos.

Un módulo de personalización puede considerarse un conjunto de operaciones para adaptar un cierto tipo de contenidos el módulo de adaptación de contenidos posee una tabla donde varios módulos de personalización se alistan apareciendo como disponibles para llevar a cabo la personalización del contenido. Cada función de decisión debe primero elegir o señalar cuáles de entre los módulos de personalización van a ser

empleados para manipular el contenido deseado después de seleccionar las acciones relevantes para la adaptación y sus correspondientes parámetros.

Las dos primeras entradas de Action_Stack contiene unos comandos para la lógica del módulo de personalización de contenidos: la primera entrada selecciona en el módulo debe personalización específico deber procesal del contenido. La segunda entrada indica al módulo de procesamiento y personalización de contenidos como debe llevarse a cabo la adaptación en si.

Con este fin la segunda entrada de Action_Stack puede albergar uno de los tres siguientes valores:

1. ADAPT: las decisiones de personalización de contenidos se aplican sobre el contenido original.
2. VARIATION: en este caso las acciones de personalización de contenidos se realizan sobre una variación del contenido original previamente disponible.
3. REMOVED: El contenido va a ser eliminado. Estos se corresponden, por ejemplo, al caso en el que ciertos objetos en el contexto de un contenido compuesto no vayan a ser usados.

La segunda parte de la salida del módulo de decisión sobre las acciones sobre contenidos corresponden acciones de adaptación específicas de sus parámetros correspondientes.

Mientras que el algoritmo de decisión comienza a generar las acciones para llevar a cabo la adaptación, éstas serán almacenadas temporalmente; después de que la última acción para la adaptación y haya sido almacenada, el algoritmo insertas en la pila un símbolo para indicar que el proceso de personalización se encuentra completamente definido.

8.1. Algoritmos de personalización de contenidos

El módulo de decisión sobre contenidos es el responsable de la elección de los métodos precisos de personalización que se van a emplear sobre los contenidos a adaptar. Este bloque es consciente de los métodos disponibles en el módulo de personalización de contenidos y debe decidir qué acciones se llevan a cabo sobre los contenidos asándose en la información disponible en los descriptores de contenido y entorno, MMCD y DIUED. Estas acciones son ejecutadas posteriormente por el módulo de personalización de contenidos, que recibe la lista de acciones a ejecutar.

Cuando ambas descripciones se reciben, el módulo de decisión sobre contenidos inicia el proceso de decisión. En primer lugar, comprueba el estado del motor UMA y actúa de acuerdo con este estado de dos formas posibles:

1. Modo “Bypass”, en el que el motor UMA se configura paea que no se realicen operaciones de personalización sobre los contenidos, por lo que éstos se transmiten sin modificaciones.
2. Modo normal, en el que el motor UMA se configura para que los contenidos se adapten empleando las acciones determinadas tras el análisis de los descriptores de contenidos y del entorno de usuario, así como teniendo en consideración el juego de operaciones de adaptación disponibles.

Se ha desarrollado una aplicación práctica llamada Describe7 que demuestra la viabilidad de los enfoques adoptados y su eficacia. Se trata de una herramienta de descripción de contenidos basada en la norma MPEG-7. En los apartados que siguen se presentan sus resultados y potencialidades.

9. Aplicación practica

La herramienta de descripción MPEG-7 se emplea para llevar a cabo labores de marcaje o descripción manual de contenidos multimedia mediante los descriptores y esquemas presentados a lo largo de este memoria. Es uno de los módulos de la plataforma propuestos. Estas descripciones almacenadas en ficheros de texto ASCII se pueden almacenar en un disco local o en servidores MPEG-7 remotos.

Esta herramienta permite crear instancias de varios descriptores MPEG-7. La herramienta de descripción MPEG-7 debería implementar todos los descriptores y esquemas de descriptores definidos en la norma MPEG-7 considerados útiles para los propósitos de la personalización de contenidos multimedia. Sin embargo, en el contexto de este Proyecto, se ha limitado su alcance a ciertos descriptores relevantes. La ampliación de sus capacidades es una labor mecánica, dado que la lista de descriptores y su relación jerárquica con los esquemas de descripción correspondientes se especifican en ficheros de texto.

9.1. Características y estructura del programa

9.1.1. Programación orientada a objetos

Para desarrollar los módulos que componen el sistema UMA se deben tomar decisiones cuidadosamente, en lo referente a los lenguajes y plataformas de desarrollo, porque determinan en gran medida las características del código generado. En el caso de la herramienta de descripción se tomó la decisión de emplear un enfoque orientado a objetos por varios motivos:

1. La programación orientada a objetos genera programas naturales, más comprensibles y próximos a la realidad perceptible y al lenguaje natural humano.
2. Es uniforme ya que la representación de los objetos lleva implícito tanto el análisis como el diseño y la codificación de los mismos.
3. Comprensible porque los datos que componen los objetos y los métodos que los manipulan se agrupan en clases, que se corresponden con las estructuras de información que el programa trata.

4. Es flexible porque al tener relacionados los procedimientos que manipulan los datos con los datos a

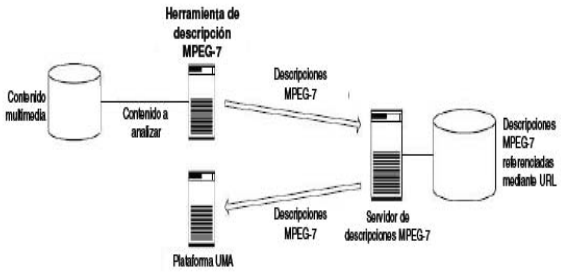

Figura 14. La herramienta de descripción MPEG-7 en el contexto del sistema propuesto

tratar, cualquier cambio realizado sobre ellos quedará reflejado en cualquier lugar donde estos datos aparezcan.
5. Es estable porque permite aislar y diferenciar aquellos datos que permanecen inalterados en el tiempo de aquellos que sí varían.
6. Es confiable porque la naturaleza modular de los objetos permite realizar modificaciones a una parte de un programa sin que esto afecte a otras partes. Los objetos aíslan el conocimiento y la responsabilidad al entorno donde se desenvuelven. Este aislamiento otorga la posibilidad de probar y validar cada componente de manera independiente. Una vez que se valida un componente, se puede reutilizar con garantías.
7. Es reutilizable. Al igual que los módulos, se pueden reutilizar los objetos en diversos programas. A diferencia de los módulos, la programación orientada a objetos proporciona la herencia y el polimorfismo para, respectivamente, permitir extender objetos existentes y escribir código genérico. Esto hace que el código sea más claro y mantenible a la par que recorta el tiempo de desarrollo.
8. Es mantenible cuando el código ha sido bien diseñado. Para corregir un error, tan sólo se debe corregir una instancia localizada del problema. Debido a que el cambio de la implementación es transparente, todos los demás objetos se benefician de la mejora. El lenguaje natural del código debe permitir que otros desarrolladores entiendan el cambio sin problemas.
9. Es extensible porque los programas no son estáticos; deben crecer y cambiar con el paso del tiempo para permanecer útiles. La programación orientada a objetos otorga al programador diversas herramientas para extender el código. Entre estas características están la herencia, el polimorfismo, la sobrecarga de métodos, la delegación y una variedad de patrones de diseño.
10 Es oportuno, porque la programación orientada a objetos contribuye a lograr rápidos ciclos de desarrollo. Reduce los tiempos perdidos del ciclo de desarrollo proporcionando software confiable, reutilizable y fácilmente extensible.

El software natural simplifica el diseño de sistemas complejos. Aunque no se debe de pasar por alto un diseño cuidadoso, el software natural puede agilizar los ciclos de diseño porque permite concentrarse en el problema que se intenta resolver.

Cuando un programa se divide en una cantidad determinada de objetos, el desarrollo de cada componente puede realizarse en paralelo. Diversos programadores pueden trabajar de manera independiente con las clases. Esta programación en paralelo da como resultado tiempos de desarrollo más rápidos.

Uno de los puntos clave a destacar es que la programación orientada a objetos no sustituye a ninguna metodología ni lenguaje de programación anterior.

Todos los programas que se realizan según el paradigma de la programación orientada a objetos, pueden realizarse igualmente mediante programación estructurada. Su uso en la actualidad se justifica porque el desarrollo de todas las nuevas herramientas basadas en una interfaz de usuario gráfica como los proporcionados por los sistemas operativos contemporáneos es mucho más sencillo.

Es claro, que las características asociadas a este paradigma de la programación, facilitan enormemente la consecución de los objetivos planteados en términos de modularidad, flexibilidad y extensibilidad.

Para el desarrollo de las aplicaciones mediante un lenguaje orientado a objetos, se han valorado los dos grandes lenguajes en este segmento: Java y C++. Es claro, en la actualidad al menos, que C++ podría cubrir las necesidades de un proyecto de desarrollo que englobase a todos los módulos del sistema propuesto. Su mayor rendimiento y eficiencia en términos generales lo hacen ideal para el desarrollo de la plataforma y de los servidores. Sin embargo, considerando el contexto en el que la herramienta de descripción, que es el módulo desarrollado, puede ejecutarse se ha escogido el lenguaje Java.

9.2. Estructura del programa

La aplicación Describe7 se estructura en cinco módulos independientes. Dos de ellos, han sido desarrollados por el autor y pueden consultarse en la parte de este Proyecto titulada “Planos”. Su cometido es el de proporcionar al usuario una interfaz gráfica cómoda y eficaz para llevar a cabo la tarea de anotar o describir contenidos, especificando el valor de los correspondientes descriptores MPEG-7. Esta descripción se hace desplegando un árbol de descriptores. Ya se explicó la relación jerárquica entre los esquemas y los descriptores, y su configuración arborescente. Esta relación y configuración se expresan de forma natural a través de la interfaz. Para ello es necesario emplear DomEcho.java, módulo encargado de presentar los descriptores en pantalla y de analizar las expresiones.

Existen otros tres módulos auxiliares que hace un uso intensivo de las funciones incorporadas a las bibliotecas para manipular contenidos multimedia y para manipular

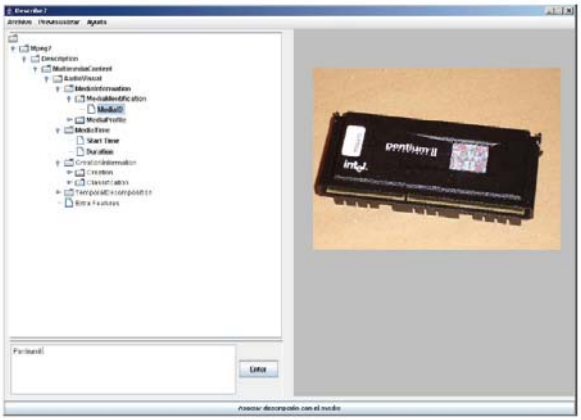

Figura 6.3. Descripción de una imagen con el árbol de descriptores desplegado parcialmente

expresiones XML. Los tres primeros proporcionan a Describe7 la habilidad para presentar en pantalla contenido multimedia y en particular imágenes, audio o vídeo codificados en algunos de los formatos más populares, con el fin de que un operador pueda visualizar y/o escuchar el material antes de su descripción manual. Estos módulos y librerías proporcionan la capacidad de controlar la reproducción a través de la interfaz gráfica en el caso de los ficheros con contenido audiovisual.

Este mismo módulo, es capaz de salvar las descripciones MPEG-7 en formato XML en ficheros de forma permanente.

10. Conclusiones

Parece claro que las aplicaciones multimedia en las comunicaciones móviles representan ya un sector en alza progresiva. El reto de los entornos UMA presenta varios problemas técnicos que han sido tratados a lo largo de este Proyecto. La definición de un entorno UMA establece que "el objetivo de la tecnología UMA es hacer disponible diferentes presentaciones de la misma información, más o menos complejas adaptadas a los tipos de medios tratados, capacidades de las terminales, redes y preferencias de los usuarios.

Basándonos en esta definición, se ha diseñado e implementado un sistema UMA siguiendo un enfoque descendente en lo que al nivel de abstracción se refiere.

1. Sistema UMA, en el que se analiza el problema íntegro al más alto nivel. Se diseña y prueba un marco completo para satisfacer las demandas UMA.
2. Plataforma UMA, como elemento de la red que lleva a cabo las personalizaciones de los contenidos; se desarrollan un motor de personalización y las interfaces correspondientes a la plataforma.
3. Motor UMA, que se corresponde con el más bajo nivel de adaptación y que implementa los algoritmos de adaptación y selección de contenidos.

El motor de personalización de contenidos se compone de cuatro partes, bien descritas: la herramienta de descripción de contenido multimedia, la herramienta de descripción del entorno de usuario, los algoritmos de decisión sobre la personalización y los algoritmos de adaptación de contenidos multimedia.

Con respecto a la descripción de los contenidos multimedia, se ha propuesto y considerado varias alternativas. Aquella que mejor satisface los requisitos UMA fue la norma MPEG-7. De la experiencia adquirida durante el manejo de esta norma, se concluye que:

1. MPEG-7 proporciona potentes herramientas para describir contenido audiovisual y en particular para ser aplicada en entornos UMA.
2. MPEG-7 no proporciona una solución para la descripción de determinados tipos de contenidos compuestos, por lo que no resulta suficientemente adecuado por sí sola en entornos en los que el contenido adaptado sea compuesto.
3. MPEG-7 es genérica, de forma que su ámbito de aplicación es amplio.
4. MPEG-7 integra objetos de bajo y alto nivel de abstracción.
5. MPEG-7 es ampliable a través del lenguaje DDL.

Mientras se planteaba una solución al problema de la descripción de los entornos de usuario, la conclusión más relevante obtenida, fue la de la carencia de un marco normalizado y generalista para la descripción de entornos o contextos de usuario. En el ámbito de sistemas UMA con terminales reales, esta carencia es especialmente grave. Habitualmente se dispone de muy poca información con respecto a los entornos de los usuarios que acceden a los contenidos. Se deberían realizar esfuerzos en la normalización de un marco que permita intercambiar este tipo de información de forma eficiente y completa, cubriendo el amplio espectro de terminales comerciales disponibles en la actualidad.

Los algoritmos de decisión sobre los contenidos desarrollados en esta memoria resultan simples, dado que la personalización considerada se llevaba a cabo sobre tipos de medios predefinidos: imágenes y vídeo.

Estos algoritmos consideran en cualquier caso las capacidades de visualización de las pantallas de las terminales y el estado y características de la red para la adaptación de vídeo. Las pruebas realizadas permiten afirmar que los métodos de adaptación escogidos ofrecen buenos resultados para las características de los entornos de usuario seleccionados. La decisión es una parte del procesoUMA que no consume muchos recursos, pero que tiene una gran influencia sobre el rendimiento global de la plataforma. Los contenidos compuestos requieren de algoritmos más complejos debido a la dependencia entre las distintas componentes del mismo recurso audiovisual.

Como se esperaba, los algoritmos de adaptación, tienen una influencia determinante sobre el rendimiento global del sistema UMA. Los algoritmos implementados requieren manipular contenidos descomprimidos en memoria. Dado que el proceso de adaptación consume muchos recursos de cómputo y no está optimizado, supone el mayor cuello de botella para la plataforma. Por tanto, es claro que mejorar la eficiencia de este componente, mediante algoritmos más eficientes es un método seguro para mejorar la eficiencia global de la plataforma.

Desde un punto de vista general, se concluye que la existencia de un servidor de descripciones MPEG-7, así como el de un servidor de descripciones de entornos de usuario, son esenciales para facilitar el acceso a las características necesarias para adaptar adecuadamente el contenido. Los componentes de red implementados permiten hacerse una idea de cómo obtener un sistema más completo y de cómo los componentes y los conceptos asociados a las plataformas UMA pueden ser explotados en un escenario real.

Los componentes clave en una plataforma UMA incluyen la descripción de los entornos de los usuarios la descripción de los contenidos multimedia, los algoritmos de decisión sobre la personalización y los algoritmos de adaptación. Todas estas áreas convergen en las plataformas UMA y resultan interdependientes.

Referencias

[1] A. Vetro, C. Christopoulos, T. Ebrahimi. "Universal multimedia access", IEEE Signal Processing Magazine, pág. 16. Marzo 2003.

[2] W. Ma, I. Bedner, G. Chang, A. Kuchinsky, J Zhang. "A framework for adaptative content delivery in heterogeneous network environments". Proceedings of SPIE/ACM Conference on Multimedia Computing and Networking. Febrero 2000.

[3] MPEG Requirements Group. "MPEG-7 applications", ISO/MPEG N3934, Pisa MPEG Meeting. Enero 2001.

[4] P. Van Beek, J. Smith, T. Ebrahimi. "Metadata-driven multimedia access". IEEE Signal Processing Magazine, pp. 40-52, Marzo 2003.

[5] E. Fossback, P. Manzanares, J. Yago, J. Perkis. "An MPEG-21 framework for streaming media". Procceedings of IEEE Workshop pn Multimedia Signal Processing 2001, pp. 147-152, Octubre 2001.

[6] H. Sun, A. Vetro, K. Asai. "Resource adaptation based on MPEG-21 usage environment descriptions". Proceedings of 2003 IEEE International Symposium on Circuits and Systems, pp. 536- 539. Mayo 2003.

[7] A. Perkis, J. Zhang, T. Holvorsen, J. Kjode, F. Rivas. "A streaming media engine using digital item adaptation". Proceedings of IEEE Workshop on Multimedia Signal Processing 2002, pp. 73-76.Diciembre 2002.

[8] J. Bormans, J. Gelissen, A. Perkis. "MPEG-21: The 21st century multimedia framework", IEEE Signal Processing Magazine, pp. 53-62. Marzo 2003.

[9] A. Vetri, H. Sun. "Media conversion to support mobile users". Proceedings of Canadian Conference on Electrical and Computer Engineering 2001, pp. 607-612. Mayo 2001.

[10] K. Lee, H. Chang, S. Chun, H. Choi, S. Sull. "Perception-based image transcoding for universal multimedia access". Proceedings of 2001 IEEE International Conference on Image Processing, pp. 475-478. Octubre 2001.

[11] C. Wang et al. "FGS-based video streaming test-bed for MPEG-21 universal multimedia access with digital item adaptation". Proceedings of the 2003 IEEE International Symposium on Circuits and Systems, pp. 364-367.Mayo 2003.

[12] T. Echigo, K. Masumitsu, M. Teraguchi, M. Etoh, S. Sekiguchi. "Personalized delivery of digest video managed on MPEG-7". Proceedings of International Conference on Information Technology: Coding and Computing, pp. 216-220. Mayo 2003

[13] Y. Chen et al. "Personalized multimedia services using a mobile service platform". Proceedings of Conference on Wireless Comminications and Networking, vol. 2, pp. 918-925. Junio 2002

[14] T. Lemlouma, N. Layaïda. "A framework for Media Resources Manipulation in an Adaptation and Negociation Architecture". INRIA Rhône-Alpes Research Unit. Enero 2001

[15] R. Mohan, J. Smith, C. S. Li. "Adapting Multimedia Internet Content for Universal Access". IEEE Transactions on Multimedia. Marzo 1999.

[16] W. Ma, I. Bedner, G. Chang, A. Kuchinsky, J Zhang. "A framework for adaptative content delivery in heterogeneous network environments". Proceedings of SPIE/ACM Conference on Multimedia Computing and Networking. Febrero 2000.

[17] J. R. Smith. "Digital Video Libraries and the Internet". IEEE Communications, Enero 1999.

[18] J. R. Smith, V. Castelli, C. S. Li. "Adaptative Storage, Rerieval of Large Compressed Images". Proceedings of IS&T/SPIE Symposium on Electronic Imaging.Enero 1999.