REVISTA COLOMBIANA DE COMPUTACIÓN
Volumen 18, número 1
Págs. 27 - 45

Escenario para la transmisión de streaming adaptativo DASH-WebM

Scenario for adaptive streaming transmission DASH-WebM

Duvernei Ortiz T.
Universidad del Cauca
Correo: duberneyortiz@unicauca.edu.co

Gabriel E. Chanchí G.
Universitaria Colegio Mayor del Cauca
Correo: gchanchi@unimayor.edu.co

José L. Arciniegas H.
Universidad del Cauca
Correo: jlarci@unicauca.edu.co

Diego F. Durán D.
Universidad del Cauca

Wilmar Y. Campo M.
Universidad del Quindío
Correo: wycampo@uniquindio.edu.co

Fecha de recibido: 16/11/2016 y Fecha de aprobación: 15/02/2017

Resumen

Tradicionalmente el videostreaming ha sido soportado por los protocolos RTP y RTSP, de modo que el servidor gestiona una sesión diferente para cada cliente y coordina la entrega de paquetes. Actualmente el estándar de streaming adaptativo DASH ofrece otro enfoque a través de HTTP, de tal forma que el cliente extrae los datos del servidor, sin mantener el estado de la sesión. Así, se tiene como ventajas el pleno uso de la infraestructura de Internet y la adaptación del contenido multimedia al ancho de banda de la red. A pesar de lo anterior, el proceso de generación y distribución de contenidos multimedia DASH, requiere la ejecución secuencial de tareas de codificación, segmentación, creación del descriptor MPD y reproducción del contenido DASH. Para el caso de los contenidos multimedia WebM, las anteriores tareas son realizadas separadamente por un conjunto de herramientas libres, por lo que el proceso de generación del contenido DASH no es automático. En este artículo se propone un escenario de transmisión para streaming adaptativo DASH, cuyo principal componente es la herramienta DASHWebMConverter, la cual se encarga de automatizar el proceso de generación de contenidos DASH WebM. Adicionalmente, se presenta la evaluación del escenario de streaming adaptativo, mediante pruebas de seguimiento de ancho de banda y pruebas de consumo de memoria sobre los principales componentes del escenario.

Palabras Clave:

Videostreaming
Contenido multimedia
DASH
WebM

Abstract

Traditionally, video streaming technique has been supported by RTP and RTSP protocols, so that the server must manage a different session for each client and coordinate the delivery of packages. Currently the DASH standard offers a different approach using HTTP, in such a way that the client extracts data from server, without needing to maintain the session state. Thus, the advantages of DASH are: full use of existing Internet infrastructure and adaptation of multimedia content to the network bandwidth. Despite the above, the generation process for DASH multimedia content requires the sequential execution of multiple tasks such as: coding, segmentation, creation of MPD descriptor and content playback. In the case of WebM files, previous tasks are performed by a set of open source tools separately, which is why the generation process for DASH multimedia content is not automatic. In this paper we propose a transmission scenario for adaptive streaming DASH, whose main component is the tool called: "DashWebMConverter", which automates the generation process for DASH WebM content. Also, in this paper we present the evaluation of the scenario for the transmission of daptive streaming, using bandwidth tacking and memory consumption tests.

Keywords:

Video streaming
Multimedia content
DASH
WebM.

1. Introducción

En los últimos años, Internet se ha convertido en un importante canal para la transmisión de contenidos multimedia que utilizan HTTP como protocolo principal. De acuerdo a [1], en Norteamérica el *streaming* de contenidos de entretenimiento representa más del 68% del tráfico en redes de acceso fijo, en donde solo Netflix constituye el 31.6%. De igual manera en Europa el *streaming* de entretenimiento en tiempo real para redes de acceso fijo supera el 47.4%, como resultado del incremento en la disponibilidad de servicios de video OverThe Top (OTT). Por su parte, el *streaming* de contenidos de entretenimiento en América latina representa un 50% en cuanto al tráfico de acceso fijo y el 29% del tráfico de acceso móvil. Considerando lo anterior, y que el 35% del tráfico total de la red de internet en América Latina utiliza HTTP en la capa de aplicación, para su sistema de *streaming* multimedia, se puede establecer HTTP como el principal protocolo en las redes modernas [2].

El *streaming* es una técnica para la distribución de contenido multimedia en internet [3], en la cual no es necesario que el cliente descargue completamente esta información para consumirla, ya que se va almacenando en un *buffer* y se va ejecutando al mismo tiempo que se trasmite por la red. Tradicionalmente el *streaming* en internet se llevaba a cabo por medio de protocolos como *Real-time TransportProtocol* (RTP) y *Real Time Streaming Protocol* (RTSP). El protocolo RTP trabaja sobre UDP, por lo cual no garantiza que todos los paquetes lleguen a su destino, es por esto que el servidor debe gestionar una sesión diferente para cada cliente, además de coordinar la entrega de paquetes por medio de otros protocolos como son: RTSP y RTCP [4], por lo tanto se incrementa la información que se debe transmitir por la red.

Una solución a los inconvenientes del *streaming* mencionados anteriormente ha sido la técnica de descarga progresiva. A diferencia de RTP, esta hace uso del protocolo HTTP, razón por la cual, la sesión no tiene estado y el cliente extrae los datos desde el servidor. Lo anterior tiene como ventaja la explotación plena de la infraestructura existente de Internet, sin embargo este enfoque tiene como desventaja el hecho de que el protocolo HTTP añade una sobrecarga significativa en la transmisión [5]. Por lo anterior, la descarga básica progresiva sobre HTTP no se adecúa a las necesidades de los entornos de movilidad, teniendo en cuenta el problema frecuente de las fluctuaciones en el ancho de banda, generando así una nueva necesidad tecnológica, la cual sugiere que el flujo de video se adapte a las capacidades de ancho de banda y a las características del dispositivo de acceso, con el fin de ofrecer al usuario una secuencia de video continua, con la mejor calidad posible en la recepción.

A partir del problema de fluctuación del ancho de ancho de banda, una de las posibles alternativas es la técnica conocida como *streaming* adaptativo, la cual ha sido especificada en 3GPP como *Adaptive* HTTP *Streaming* (AHS) [6], y consiste en cortar el archivo multimedia en segmentos de igual duración que pueden ser codificados en diferentes resoluciones y tasas de bits, los cuales se suministran a un servidor web para ser descargados a través de peticiones HTTP estándar. Con el fin de establecer la relación entre tasas de bits, segmentos y el orden de los mismos, AHS hace

uso del archivo *Media Presentation Description* (MPD), el cual contiene una descripción formal sobre una colección de datos que representan las características técnicas del contenido multimedia [7]. Cada cliente primero solicita el MPD y a partir de esa información se indica a los segmentos individuales que se ajusten mejor a sus necesidades. Así entonces, el control está del lado del cliente, de tal manera que cada segmento puede cambiar a una velocidad de bits en función del ancho de banda que disponga, o a una resolución determinada dependiendo del dispositivo de acceso.

La industria ha desplegado varias soluciones propietarias de *streaming* adaptativo, entre las más representativas están: Microsoft *SmoothStreaming* [8], Apple HTTP *Live Streaming* [9] y Adobe *Dynamic Streaming* HTTP [10]. Estas plataformas de *streaming* suelen utilizar el estándar de video digital MPEG-4 H.264, junto con MPEG Advanced Audio Coding (AAC). H.264 permite lograr una mayor calidad de imagen de video para una frecuencia de bits determinada, mientras que AAC se encarga de la distribución eficiente de sonido a través de conexiones de ancho de banda moderada. Sin embargo, el inconveniente radica en que cada uno emplea sus propias técnicas de segmentación, secuencia de tiempo y formatos de MPD [11].

Con el fin de lograr la entrega eficiente de contenidos MPEG utilizando HTTP en sus diferentes formas: adaptativa, progresiva, *streaming*/descarga y además de garantizar la interoperabilidad entre las soluciones propietarias, MPEG (*Moving Picture ExpertGroup)* desarrolla *DynamicAdaptiveStreamingover* HTTP (DASH), el cual se convierte en un estándar internacional en noviembre de 2011, siendo publicado como ISO/IEC 23009-1:2012 en abril de 2012 [7]. De acuerdo con [12], DASH se puede definir como un sistema por el cual se proporcionan formatos, que habilitan la entrega eficiente y de alta calidad de servicios de *streaming*, que cuenta con las siguientes ventajas [13]: reutiliza la tecnología existente de contenedores, *códec*, DRM (Gestión digital de derechos); puede ser desplegado sobre la infraestructura actual de las redes de distribución HTTP; el usuario percibe un mejor servicio: menor tiempo de arranque, sin *buffer*; permite seleccionar la calidad en función de la red y capacidad del dispositivo; los cambios entre calidades son automáticos y transparentes al usuario; puede coexistir con tecnologías propietarias existentes.

A pesar de las ventajas que provee el estándar DASH, en cuanto a la transmisión de contenidos multimedia, de tal manera que el flujo se adapte a los problemas de fluctuación de ancho de banda; el proceso de generación de contenido multimedia DASH requiere la ejecución de múltiples operaciones secuenciales como son: codificación, segmentación y creación del archivo descriptor MPD. En el caso de los contenidos multimedia en formato Webm, las funciones de codificación son realizadas por la herramienta *FFmpeg*, las operaciones de segmentación y agregación de los encabezados son ejecutadas por la herramienta *sample_muxer* de la librería *libwebm* y la tarea de creación del archivo de manifiesto MPD es desarrollada por la herramienta *webm_dash_manifest* de la librería *webm-tools*. Lo anterior muestra que aunque existen un conjunto de herramientas libres que permiten adecuar el contenido multimedia al formato establecido por el estándar DASH, este proceso no ha sido automatizado, razón por la cual el despliegue del contenido multimedia requiere un esfuerzo mayor del lado del servidor.

En el presente artículo se propone un escenario de *streaming* adaptativo para la transmisión de contenidos multimedia WebM, el cual busca automatizar el proceso de generación, distribución y consumo de contenido multimedia WebM, teniendo en cuenta el estándar DASH. Uno de los principales componentes del escenario de *streaming* adaptativo es la herramienta software llamada: *DASHWebMConverter*, la cual se desarrolló en el presente trabajo con el fin de optimizar el proceso de codificación de contenido multimedia WebM según el estándar DASH. La herramienta propuesta fue desarrollada en el lenguaje *Python*, y permite la integración y ejecución secuencial de las tareas de codificación, segmentación y creación del archivo descriptor MPD. Para ejecutar las anteriores tareas, esta herramienta usa en segundo plano las librerías: *libwebm, webm-tools*, así como la herramienta *FFmpeg*. Este artículo está estructurado de la siguiente forma: en la sección 2 se presenta la metodología usada para la conformación del escenario de transmisión; en la sección 3 se describe el conjunto de librerías que se tuvieron en cuenta para la conformación del escenario de transmisión DASH y el desarrollo de la herramienta automática de codificación; en la sección 4 se presentan los componentes extremo a extremo del escenario de transmisión DASH propuesto; en la sección 5 se presentan los resultados

de la aplicación de las pruebas de seguimiento de ancho de banda y consumo de memoria sobre los componentes del escenario de *streaming* DASH; y finalmente en la sección 6 se presentan las conclusiones y trabajos futuros obtenidos a partir del presente.

2. Metodología

Para la conformación del escenario de transmisión DASH, se definieron cinco fases a saber: exploración de herramientas libres, diseño del escenario de transmisión, construcción del escenario de transmisión y evaluación (ver Figura 1). En la fase de exploración de herramientas libres, se realizó un proceso de búsqueda de herramientas libres para realizar los procesos de codificación, distribución y consumo de contenidos multimedia adaptativos. En la fase de diseño del escenario, se escogieron las herramientas más adecuadas para la conformación del escenario de *streaming* adaptativo DASH. En la fase de construcción del escenario, se realizó la automatización del proceso de codificación DASH y se desarrolló un prototipo básico de codificación y consumo de contenido multimedia adaptativo. Finalmente en la fase de evaluación del escenario, se realizaron pruebas de consistencia sobre el mismo y pruebas de consumo de memoria sobre los diferentes componentes del escenario. La fase de exploración de herramientas libres es abordada en la sección 3, la fase de diseño del escenario es presentada sección 4, la fase de construcción del escenario es abordada en las secciones 4 y 5 y la fase de evaluación es considerada en la sección 6.

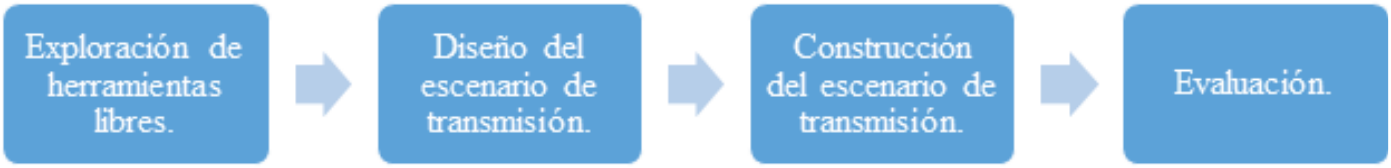

Figura 1. Fases metodología.

3. Herramientas del escenario DASH

A continuación se presenta una breve descripción de las herramientas libres utilizadas para la conformación del escenario para el despliegue extremo a extremo de *streaming* adaptativo DASH WebM. Así mismo, se describen las librerías que fueron consideradas para la configuración e implementación de la herramienta de codificación de *streaming* adaptativo DASH. Dentro de estas herramientas están: *FFmpeg, libwebm, webm-tools, webm-dash-javascript.*

***FFmpeg*:** Es una herramienta de línea de comandos perteneciente al proyecto libre *FFmpeg*, cuyas funciones principales son: convertir archivos multimedia entre formatos, cambiar el tamaño del video, generar archivos multimedia a diferentes frecuencias de muestreo y velocidad de fotogramas, entre otras. Contiene múltiples bibliotecas útiles para codificar, decodificar, *mux* y *demux*, prácticamente cualquier tipo de archivo multimedia. Dentro del presente trabajo, esta herramienta fue usada para cambiar el formato original de diferentes contenidos multimedia al formato de WebM. Así mismo *FFmpeg* es utilizada en segundo plano por la herramienta propuesta en este trabajo, con el fin de codificar el contenido multimedia Webm original a diferentes tasas de bits.

***Libwebm*:** Es una biblioteca de código abierto asociada al proyecto WebM de Google Inc.[1], la cual contiene el *script sample_muxer*, escrito en lenguaje C++. Este *script* es utilizado para agregar índices de búsqueda para cada fotograma clave y para alinear puntos de sincronización en archivos multimedia WebM. Esta biblioteca provee además software

1 Web Media Project [http://www.webmproject.org/]

para el control de *streaming* DASH y el acceso a los segmentos multimedia descargables. El *script simple muxer* es utilizado en segundo plano por la herramienta propuesta en el presente trabajo, con el fin de agregar los índices de búsqueda a los contenidos multimedia previamente codificados en el formato WebM.

***webm*-tools:** Es otra biblioteca de código abierto asociada al proyecto WebM de Google Inc., la cual contiene *scripts* escritos en el lenguaje C++, útiles para cifrar flujos de video y generar el archivo de manifiesto XML DASH, usado para describir el contenido multimedia del *streaming* adaptativo. El *script webm_dash_manifest* de la biblioteca *webm-tools* es utilizado en segundo plano por la herramienta propuesta en este trabajo, para la generación del archivo de manifiesto XML DASH de los contenidos multimedia previamente codificados e indexados.

webm-*dash*-javascript: Es una librería basada en JavaScript, la cual permite la integración del estándar de *streaming* adaptativo DASH en la Web, para lo cual hace uso de las extensiones para manejo de medios en HTML5 propias del navegador Google Chrome (Media Source API) [14]. Gracias a sus componentes *bandwidth_manager.js, dash_player.js, dash_parser.js, webm_parser.js*, entre otros, es posible realizar el consumo de segmentos de medios basados en WebM y segmentos basados en el formato de archivo ISO Base Media. En el presente trabajo se hizo uso de las anteriores librerías javascript y del componente HTML5 de reproducción, para el consumo del contenido multimedia en formato DASH desde una página web en el navegador Google Chrome.

4. Escenario de *streaming* adaptativo DASH WebM

En esta sección se presenta el diagrama extremo a extremo del escenario de *streaming* adaptativo, el cual fue configurado con ayuda de la herramienta *DASHWebmConverter*. El escenario está formado por tres módulos principales (ver Figura 2): módulo de codificación, módulo de difusión y módulo de recepción.

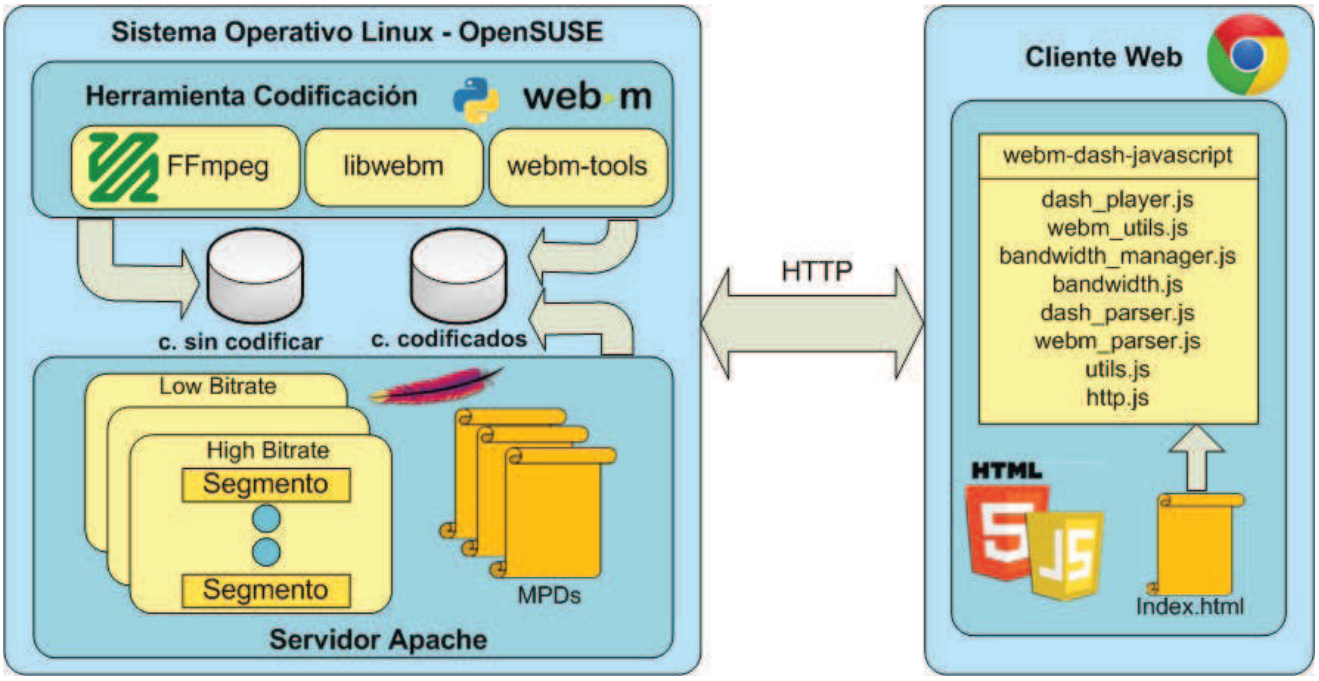

Figura 2. Escenario de *Streaming* Adaptativo.

Módulo de codificación: Este módulo es el responsable de realizar los procesos de codificación, segmentación y generación del archivo descriptor del contenido multimedia adaptativo MPD. Dentro de este módulo se encuentran los contenidos multimedia WebM a codificar y la herramienta automática de codificación *DashWebMConverter*, en

conjunto con las librerías y herramientas usadas en segundo plano (*FFmpeg, libwebm y webm-tools*). Los anteriores componentes fueron desplegados en el presente trabajo a través del sistema operativo Linux OpenSUSE, el cual facilita la configuración e integración de las herramientas que corren en segundo plano (*FFmpeg, libwebm, webm-tools*), así como el monitoreo de la herramienta de codificación propuesta, ver Figura 2.

Módulo de difusión: Este módulo es el encargado de almacenar y transmitir los contenidos multimedia WebM DASH solicitados por los clientes de *streaming* web mediante peticiones HTTP. Cada uno de los contenidos WebM está codificado en diferentes calidades o tasas de bits y tiene asociado un archivo descriptor MPD, el cual incluye características de los flujos de video y audio tales como: URL de los flujos, tipo de contenedor, *codecs* de audio y video, tasa de *bits*, resolución de pantalla del video, tasa de muestreo del audio, entre otros. Los contenidos multimedia y sus archivos de descripción MPD se encuentran alojados dentro de un servidor web, lo cual es requisito fundamental para la recepción de peticiones desde los clientes web y para la transmisión de *streaming* adaptativo basado en el protocolo HTTP. A nivel de implementación, en este trabajo se hizo uso del servidor web Apache, corriendo sobre el sistema operativo Linux OpenSUSE.

Módulo de recepción: Este módulo está formado por el cliente Web de *streaming*, el cual solicita mediante peticiones HTTP, el contenido multimedia desde el servidor de *streaming* adaptativo DASH y lo reproduce en una página Web, ver Figura 3. Durante el proceso de solicitud y reproducción, el cliente web se encarga de estimar periódicamente el ancho de banda disponible, de tal manera que según esta información y la descrita en el archivo de manifiesto MPD asociado a cada contenido, se puedan reproducir los segmentos de contenidos multimedia adecuados. Para estimar el ancho de banda y decodificar el archivo de manifiesto MPD del contenido multimedia adaptativo, se hace uso de la librería *javascriptwebm-dash-javascript*. Con respecto a la reproducción del contenido multimedia adaptativo, esta librería incluye un cliente web demo (*dash-player.html*), que permite la reproducción del contenido multimedia adaptativo, mediante la integración del componente de video de HTML5 y las funcionalidades de estimación ancho de banda y decodificación del manifiesto MPD. Este cliente muestra además la variación del ancho de banda durante el proceso de reproducción, mediante franjas de color rojo, las cuales representan las diferentes tasas de *bits* en las que ha sido codificado el contenido multimedia adaptativo (ver Figura 3). Finalmente es importante mencionar que para lograr el cambio eficiente entre segmentos multimedia DASH, es necesario utilizar el navegador Chrome en su versión 18 o superior.

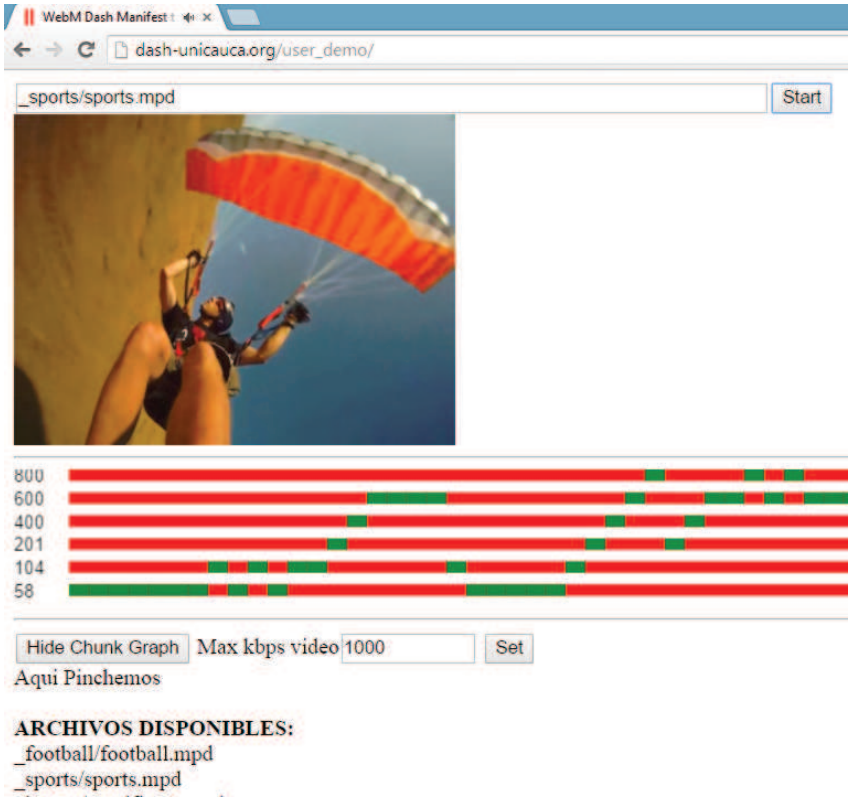

Figura 3. Cliente web de *streaming* adaptativo.

5. Herramienta automática de codificación

Como parte del escenario de transmisión de *streaming* adaptativo DASH y de manera más específica en lo que se refiere a las tareas de generación del contenido multimedia DASH WebM, el escenario está formado por la herramienta *DASHWebMConverter*. Esta herramienta facilita la integración y ejecución secuencial de tres tareas principales: codificación, segmentación y generación del archivo descriptor MPD. Estas tareas permiten la adecuación de archivos multimedia WebM al formato del estándar de *streaming* adaptativo DASH. Para realizar los procesos anteriores, la herramienta *DashWebMConverter* se encarga de ejecutar o invocar en segundo plano las librerías y/o herramientas: *FFmpeg, libwebm y webm-tools*, las cuales corren sobre el sistema operativo Linux. Esta sección está dividida en tres partes principales; en la primera sección se describe el proceso de codificación de contenidos multimedia adaptativos WebM seguido por la herramienta de codificación propuesta; en la segunda sección se presenta el diagrama funcional modular de la herramienta *DashWebMConverter*; y en la tercera sección se describe la estructura del archivo de manifiesto MPD generado por la herramienta de codificación.

5.1 Proceso de codificación de *streaming* adaptativo

En la Figura 4 se presentan las diferentes fases del proceso de codificación de contenidos multimedia adaptativos WebM DASH, realizado por la herramienta propuesta. Los archivos WebM para *streaming* adaptativo, deben contar con flujos de audio y video multiplexados, con un índice de búsqueda asociado a cada fotograma clave del flujo de video, de tal forma que se permita el cambio entre diversos flujos de *streaming* sin discontinuidad, a medida que

varía el ancho de banda. A partir de lo anterior, una de las funciones de la herramienta de codificación es adaptar los contenidos WebM para ser transmitidos, ya que por defecto cada uno de estos archivos corresponde a un *stream*, sin fragmentación y sin puntos de sincronización.

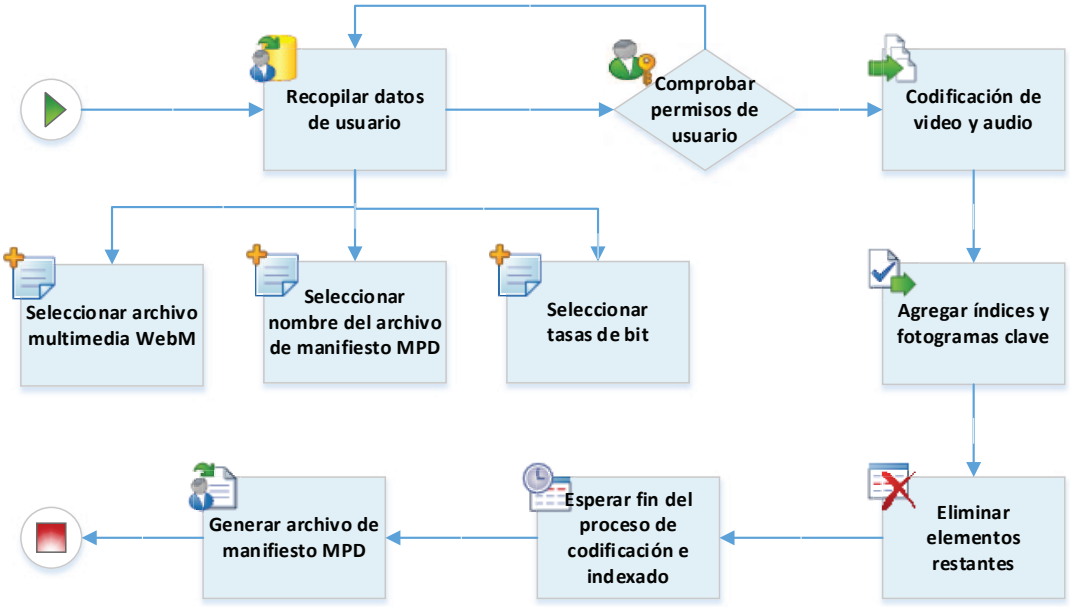

Figura 4. Proceso de codificación de *streaming* adaptativo.

La primera fase del proceso de codificación, consiste en recibir desde la interfaz gráfica de la herramienta, tres datos básicos para este proceso, como son: la ruta del archivo multimedia WebM, las tasas de video a codificar, de acuerdo a la calidad del contenido multimedia y el nombre del documento de manifiesto MPD (ver Figura 5).

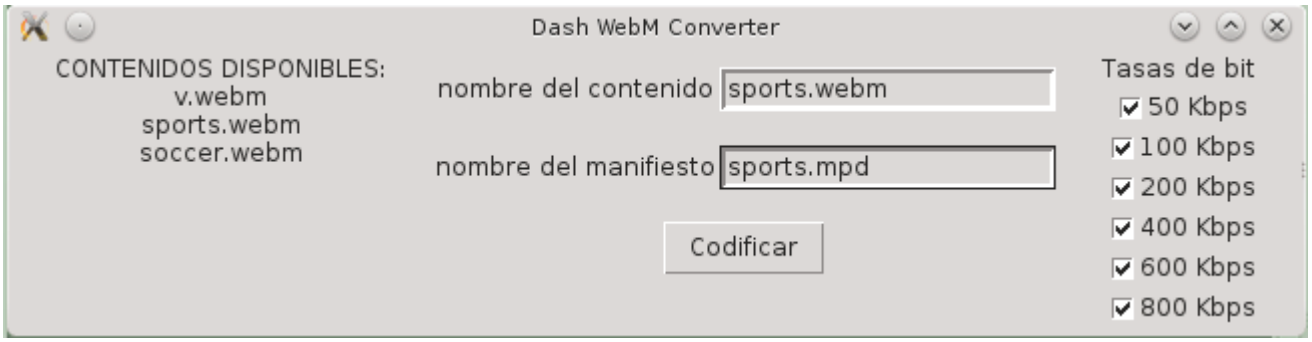

Figura 5. Interfaz Gráfica de la herramienta *DashWebMConverter.*

Una vez brindados y comprobados los permisos del sistema operativo, se da inicio a la fase de codificación, en la cual la herramienta *DashWebMConverter* se encarga de generar en paralelo tantos flujos de video WebM, como tasas de *bits* se hayan escogido en la primera fase. De igual forma, se genera un solo flujo de audio WebM por todos los flujos de videos codificados. Para realizar lo anterior, la herramienta *DashWebMConverterhace* invocaciones en segundo plano y en paralelo a la herramienta *FFmpeg*, la cual se ejecuta como un proceso del sistema operativo Linux. Es importante mencionar que dentro de los parámetros usados para la invocación de *FFmpeg*, se debe tener en cuenta el uso de la librería *libvpx* (*códec* VP8) para la codificación de video y de la librería *libvorbis* para la codificación de audio. Así mismo, se debe indicar a *FFmpeg* el intervalo de fotogramas máximo y mínimo entre cada fotograma clave. Dichos valores se utilizan para alinear todos los puntos de sincronización en las secuencias de video a canjear (ver Figura 6).

Figura 6. Fase de codificación de los contenidos WebM.

Después de que son generados los flujos de video WebM con diferentes tasas de *bit* y considerando que la herramienta *FFmpeg* no permite la alineación de los parámetros "*Cluste*" y "*Cues*" de los contenidos multimedia, la siguiente fase del proceso consiste en alinear los puntos de sincronización de las secuencias de vídeo y agregar los índices de búsqueda para cada fotograma clave dentro de los mismos flujos de video. Para realizar lo anterior, la herramienta *DashWebMConverter* ejecuta en segundo plano el *script simple_muxer* de la librería *libwebm*. Los contenidos resultantes tras la ejecución del anterior proceso (nombre terminado en "_sm") son presentados en la Figura 7, el resto de contenidos innecesarios de las fases anteriores son eliminados por la herramienta *DashWebMConverter*.

Finalmente, la última fase consiste en la generación del archivo de manifiesto MPD, el cual contiene información de los flujos de video y audio disponibles para canjear al momento de ser solicitados por el cliente web de *streaming* DASH. Para generar el archivo MPD, la herramienta *DashWebMConverter* ejecuta en segundo plano el *script webm_dash_manifiest* de la librería *webm-tools,* enviándole como parámetro el listado de archivos asociados a los flujos de video y audio generados a partir de las tasas de *bits* asignadas en la fase 1. El documento de manifiesto MPD generado, comprende características de los flujos de video y audio tales como: URL de los flujos, tipo de contenedor, *codecs* de audio y video, tasa de *bits*, resolución de pantalla del video, tasa de muestreo del audio, entre otros. En la Figura 7 se muestran los archivos de *streaming* adaptativo generados, en conjunto con su archivo descriptor MPD.

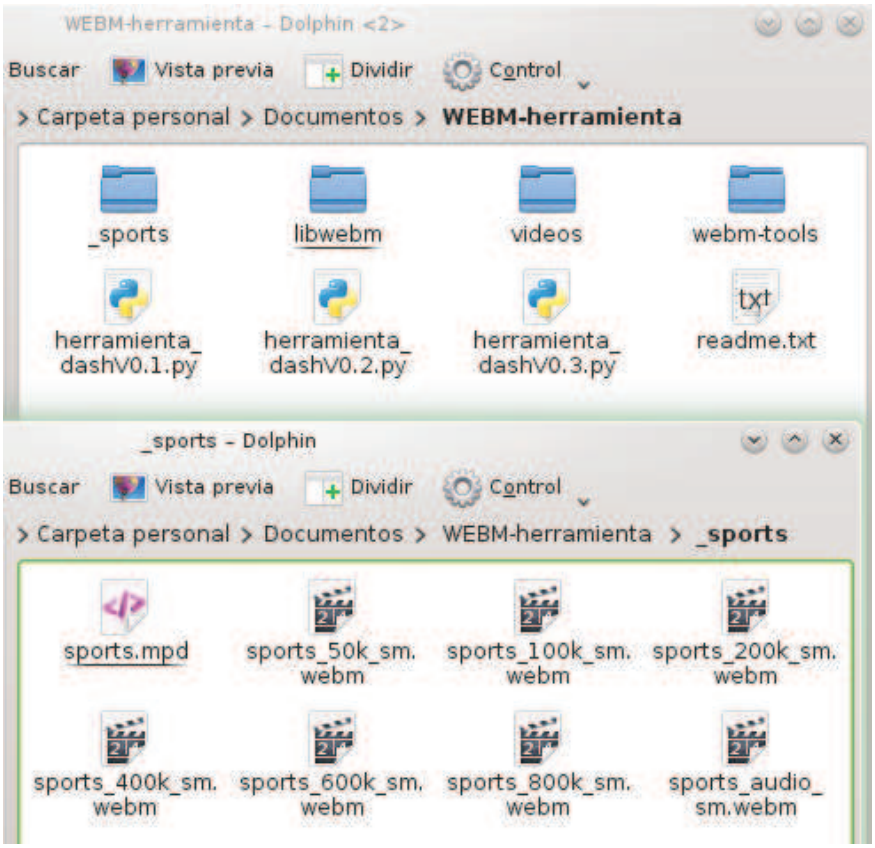

Figura 7. Recursos de la herramienta de codificación.

5.2 Diagrama modular de la herramienta

En esta sección se presenta el diagrama modular de la herramienta *DASHWebMConverter*, la cual fue desarrollada en el lenguaje Python. Tal como se mencionó en la sección anterior, la función de la herramienta *DASHWebMConverter* es ejecutar de manera secuencial las tareas de codificación, segmentación y generación del archivo de manifiesto MPD para contenidos multimedia adaptativos DASH WebM. En el diagrama de la Figura 8 se distinguen los siguientes los siguientes bloques funcionales principales: interfaz gráfico de usuario, lanzador de hilos, ejecutor de comandos, lector de archivos multimedia.

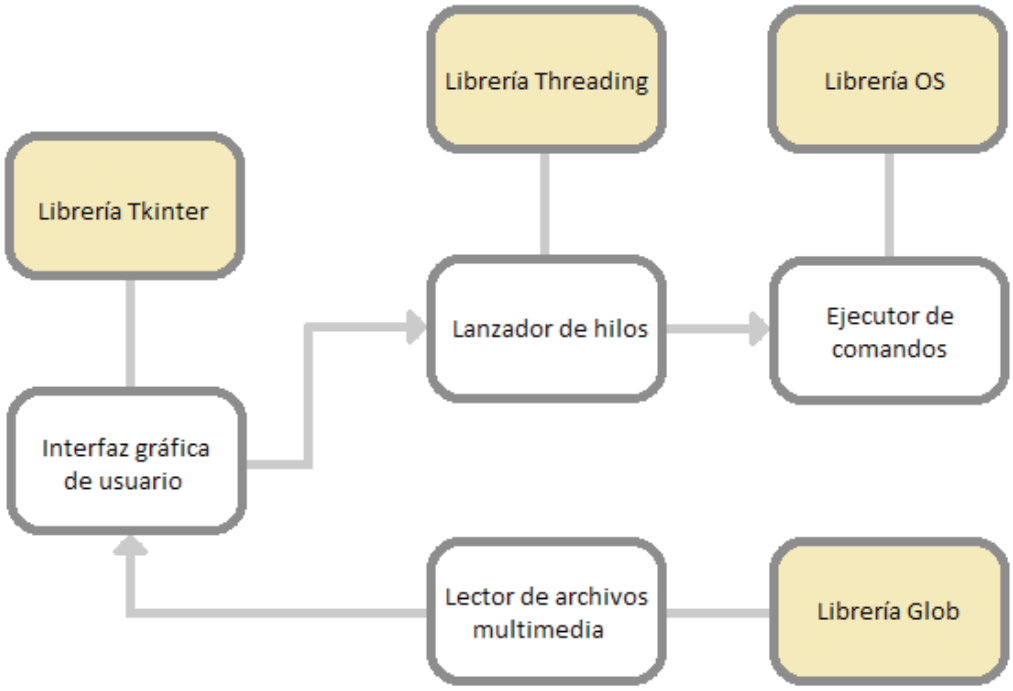

Figura 8. Diagrama de bloques herramienta de codificación.

El módulo de interfaz gráfica de usuario está conformado por la librería *Tkinter* de Python, la cual permite desplegar diferentes componentes gráficos como: campos de texto, etiquetas y botones, a través de los cuales se solicita el archivo multimedia WebM base a codificar, el nombre del documento de manifiesto MPD y las tasas de *bits* a codificar. De manera simultánea en que se lanza la herramienta *DashWebMConverter*, se puede visualizar en la interfaz gráfica el contenido multimedia disponible para codificar, esta tarea es realizada por el modulo lector de archivos multimedia, a través de la librería *Glob*, ver Figura 8.

Por su parte el módulo Lanzador de Hilos, se encarga de crear tantas instancias de hilos o procesos simultáneos, como tasas de *bits* se hayan elegido desde la interfaz gráfica, más un hilo adicional para el contenido de audio. Para lo anterior, la herramienta *DashWebMConverter* hace uso de la librería *Threading* de Python. Cada hilo de video hace una invocación en segundo plano de la herramienta *FFmpeg*, enviando como parámetro una tasa de codificación diferente, mientras que en el caso del hilo de audio, se escoge una tasa de *bits* estándar. Al terminar la tarea de codificación por parte de cada hilo lanzado, la herramienta *DashWebMConverter* ejecuta el *script sample_muxer* de la librería *libwebm*, con el fin de agregar los índices de búsqueda para cada fotograma clave de cada flujo de video.

Finalmente, el módulo ejecutor de comandos, es el encargado de ejecutar comandos del sistema operativo Linux a través de la librería OS de Python. Este módulo interactúa de manera directa con la consola del sistema operativo y permite lanzar en segundo plano y durante diferentes momentos del proceso de codificación, las herramientas y/o librerías: *FFmpeg*, y los *scripts sample_muxer y webm_dash_manifest.*

5.3 Archivo descriptor MPD

En esta sección se presenta el esquema del documento de manifiesto MPD generado por la herramienta de codificación *DashWebMConverter* en su fase final, ver Figuras 9 y 10. A modo de ilustración se considerará el documento MPD usado para el despliegue del escenario de *streaming* presentado en la Figura 3. Este documento es generado por la

herramienta *DashWebMConverter* a través de la ejecución en segundo plano del *script webm_dash_manifest* perteneciente a la librería *webm*-tools. El documento MPD está basado en el lenguaje de marcado extensible XML y contiene las siguientes configuraciones dentro de la etiqueta principal <*MPD*>:

- El parámetro "Perfil de presentación" se encuentra por defecto fijado en el perfil: "webm-on-demand", indicando así el modo de transmisión del contenido multimedia (ver Figura 9).
- La etiqueta "Periodo" incluye el punto de inicio y la duración total del contenido multimedia (ver Figura 9).

```
 9   profiles="urn:webm:dash:profile:webm-on-demand:2012">
10   <Period id="0" start="PT0S" duration="PT264.16S" >
11     <AdaptationSet id="0" mimeType="video/webm" codecs="vp8" width="320" height="240"
12     subsegmentAlignment="true" subsegmentStartsWithSAP="1" bitstreamSwitching="true">
13       <Representation id="0" bandwidth="74285">
14         <BaseURL>sports_50k_sm.webm</BaseURL>
15         <SegmentBase indexRange="1925436-1926323">
16           <Initialization range="0-248" />
17         </SegmentBase>
18       </Representation>
19       <Representation id="1" bandwidth="119023">
20         <BaseURL>sports_100k_sm.webm</BaseURL>
21         <SegmentBase indexRange="3435010-3435898">
22           <Initialization range="0-248" />
23         </SegmentBase>
24       </Representation>
```

Figura 9. Documento de descripción MPD.

- La etiqueta <*AdaptationSet*> con atributo id=0, describe la información de los flujos de video codificados a diferentes tasas de *bits*, especificando el formato de empaquetamiento, el códec de video y la resolución del contenido multimedia, ver Figura 6. Adicionalmente esta etiqueta incluye los atributos: *segmentAlignmentFlag* = *true* y *bitstreamSwitchingFlag* = *true*, los cuales al estar fijados en "*true*", indican que es posible el intercambio de *streams* de video durante la reproducción del contenido multimedia, de acuerdo al ancho de banda disponible.
- La etiqueta <*AdaptationSet*> con atributo id=1, describe el único flujo de audio asociado los diferentes flujos de video codificados, indicando el formato de empaquetamiento, el *códec* de audio y la tasa de muestreo utilizados, ver Figura 6.
- La información más específica de cada uno de los flujos de video y audio codificados, está contenida dentro de la etiqueta <*Representation*> con un id para cada *stream* individual (ver Figura 10). Esta etiqueta hace parte del nodo de nivel superior <*AdaptationSet*>, y se encarga de describir el ancho de banda asociado a cada flujo de audio o video y la URL de cada *stream* dentro del servidor de difusión.

```
50     <AdaptationSet id="1" mimeType="audio/webm" codecs="vorbis" audioSamplingRate="44100"
51     subsegmentStartsWithSAP="1">
52       <Representation id="6" bandwidth="119772">
53         <BaseURL>sports_audio_sm.webm</BaseURL>
54         <SegmentBase indexRange="3096600-3097489">
55           <Initialization range="0-4520" />
56         </SegmentBase>
57       </Representation>
58     </AdaptationSet>
```

Figura 10. Documento de descripción MPD.

6. Evaluación del escenario de transmisión

En cuanto a las pruebas realizadas sobre el escenario de transmisión, se evaluó el comportamiento del escenario con respecto a diferentes valores de ancho de banda. De igual manera se realizaron pruebas de consumo de memoria RAM y porcentaje de CPU de la herramienta de codificación *DASHWebMConverter*, así como la herramienta *FFmpeg* (utilizada en segundo plano).

En cuanto a las pruebas de seguimiento de ancho de banda se hizo uso de la herramienta libre *Dummynet*, la cual consta de un emulador de red en vivo. Entre las funciones principales de esta herramienta están: emulación de ancho de banda, pérdida de paquetes y retardo, entre otras funcionalidades. Esta herramienta de emulación es multiplataforma y puede operar sobre sistemas operativos como: OSX, Linux, Windows y FreeBSD. Estas pruebas fueron realizadas sobre el sistema operativo Windows 7, para lo cual fue necesario utilizar la arquitectura de 32 bits puesto que se requiere el uso de un controlador sin firma digital para el adaptador de red. La prueba de consumo de ancho de banda se realizó sobre un servicio de VoD construido a partir del escenario de *streaming* adaptativo DASH (ver Figura 11).

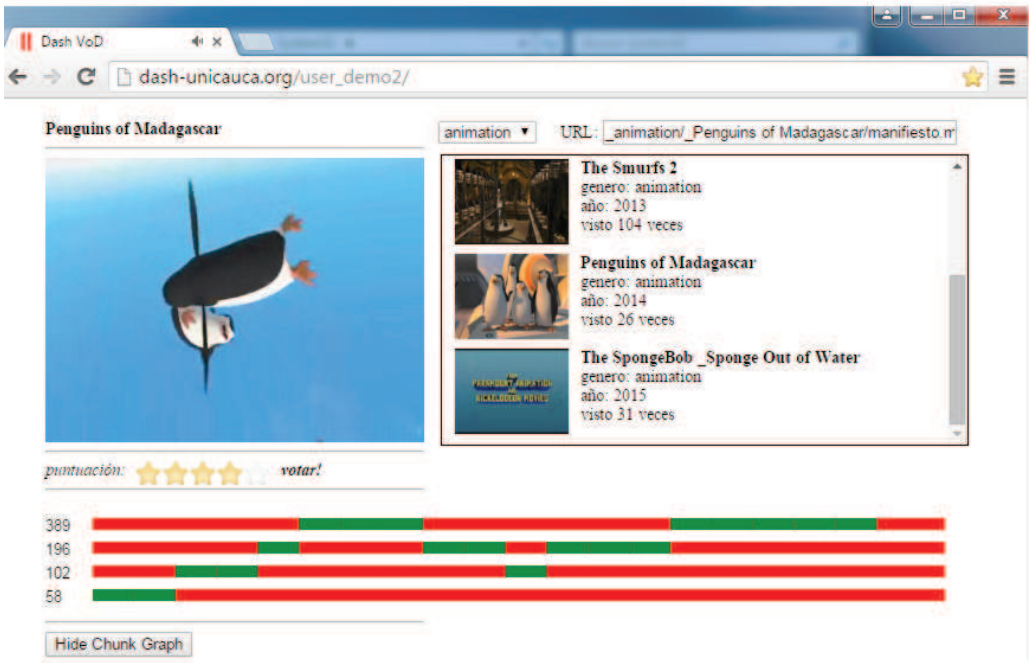

Figura 11. Servicio de VoD DASH.

Los valores de ancho de banda utilizados para la realización de la prueba fueron: 20, 100, 200, 400, 600, 800 y 1500 *kbit*/s, teniendo en cuenta las diferentes tasas de codificación de los contenidos multimedia utilizados. Es importante mencionar que el valor real del ancho de banda de la red, ofrecida por el operador es de 1 *Mbit*/s. Los datos arrojados por la realización de la prueba son presentados en la Figura 12. En la parte izquierda de la gráfica se muestra el monitor de recursos de Windows, donde se indica el ancho de banda delimitado por *dummynet*, y en la parte derecha aparece el sistema de medición de ancho de banda, desarrollado sobre el cliente web del entorno de *streaming* adaptativo DASH.

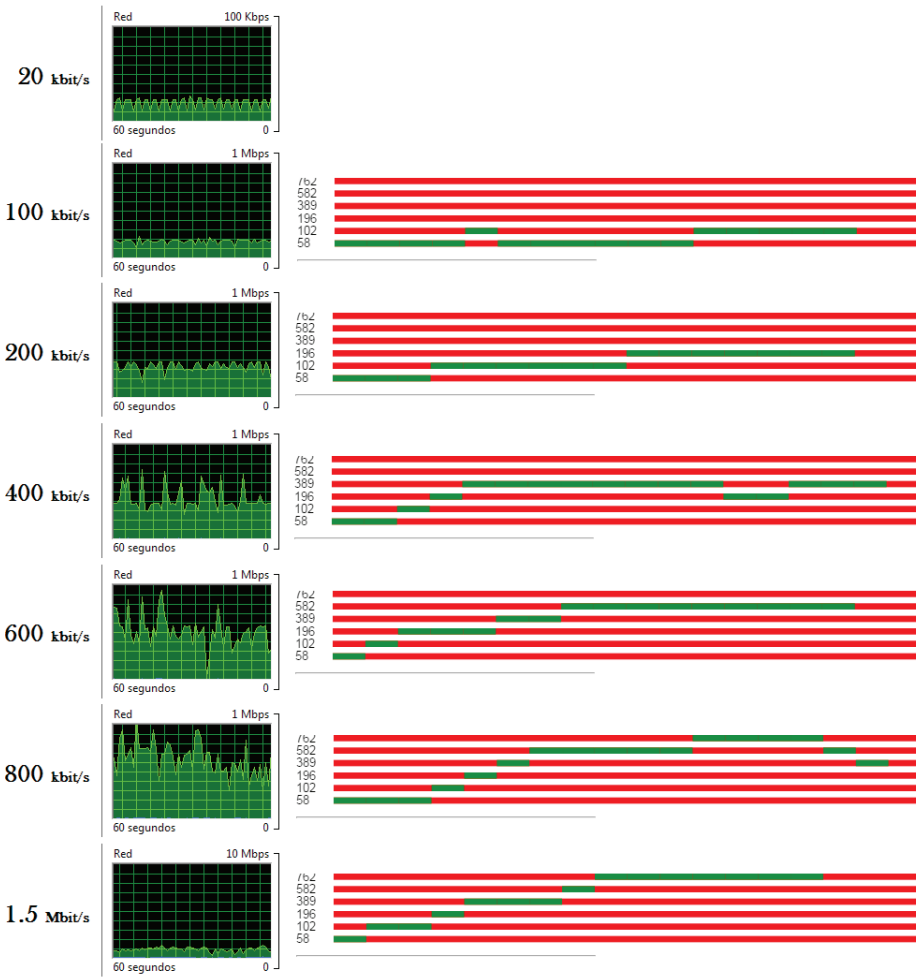

Figura 12. Emulación de ancho de banda. Fuente: propia.

Como puede observarse en la Figura 12, la herramienta *dummynet* cumple satisfactoriamente su función de emular el ancho de banda de la red, según los diferentes valores considerados por la prueba. Así mismo, según los resultados encontrados tras la realización de la prueba, se puede concluir que el servicio de VoD sigue los cambios emulados en el ancho de banda, de tal manera que el escenario de *streaming* adaptativo cumple de manera satisfactoria los requisitos especificados por el estándar DASH.

En lo que respecta a las pruebas de consumo de memoria RAM y porcentaje de CPU, se desarrolló una herramienta de monitoreo en el lenguaje Python, la cual permite obtener cada 5 segundos, los valores de memoria y porcentaje de CPU de los procesos del sistema operativo asociados a la herramienta de codificación y la herramienta *FFmpeg*. La obtención de estos valores los realiza la herramienta de monitoreo, mediante el comando "*psaux*" de Linux, el cual brinda un reporte de la cantidad de memoria RAM y el porcentaje de CPU utilizados por cada uno de los procesos activos del sistema operativo. Así mismo, se usa el comando *grep* y el lenguaje de programación *awk* (pertenecientes al sistema operativo Linux), con el fin de filtrar la información de los procesos.

En las Figuras 13 y 14 se muestran los resultados de las pruebas de porcentaje de uso de CPU y consumo de memoria RAM, realizadas sobre la herramienta *FFmpeg* (usada en segundo plano por la herramienta propuesta en este artículo), durante la codificación de un contenido multimedia WebM a 6 tasas de *bits* diferentes (50,100, 200,400,600 y 800 kbps). Dado que la herramienta de codificación lanza tantos hilos de *FFmpeg* como tasas de *bits* a codificar, más un hilo adicional para el contenido de audio, en las Figuras 13 y 14 se presenta el comportamiento de los 7 hilos (incluyendo

el de audio) usados para el despliegue del ejemplo de codificación de la figura 3. De acuerdo a la Figura 13, el hilo de audio (Hilo 7) es el primero en terminar su ejecución con un tiempo cercano a los 10 segundos y es el que menos porcentaje de CPU ocupa, con valores entre 19% y 22%. Así mismo, cabe resaltar que los hilos destinados a la codificación de video emplean un porcentaje de CPU similar durante todo el proceso, el cual oscila entre 22% y 30%.

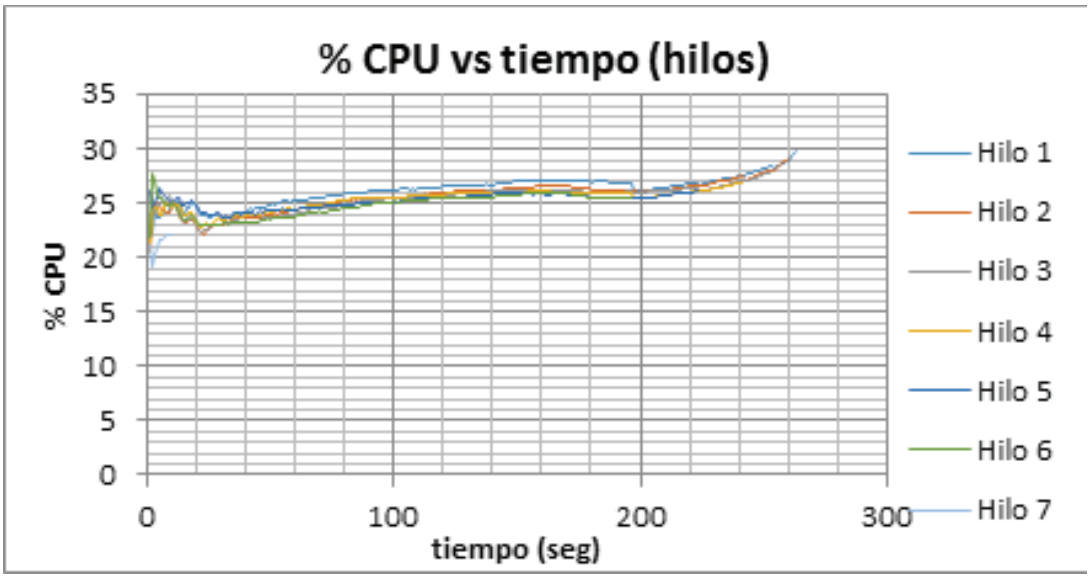

Figura 13. % CPU vs tiempo (FFmpeg).

De acuerdo a la Figura 14, el hilo de audio (Hilo 7) es el primero que termina su ejecución y es el que menos cantidad de memoria RAM consume, con un valor de 0.7 MB. Entre tanto los 6 hilos de video tienen un consumo similar de memoria que oscila entre 1.2 MB y 1.4 MB durante el proceso de codificación. Teniendo en cuenta lo anterior, en promedio el consumo de cada hilo de video en la herramienta *FFmpeg* es de 1,2 MB.

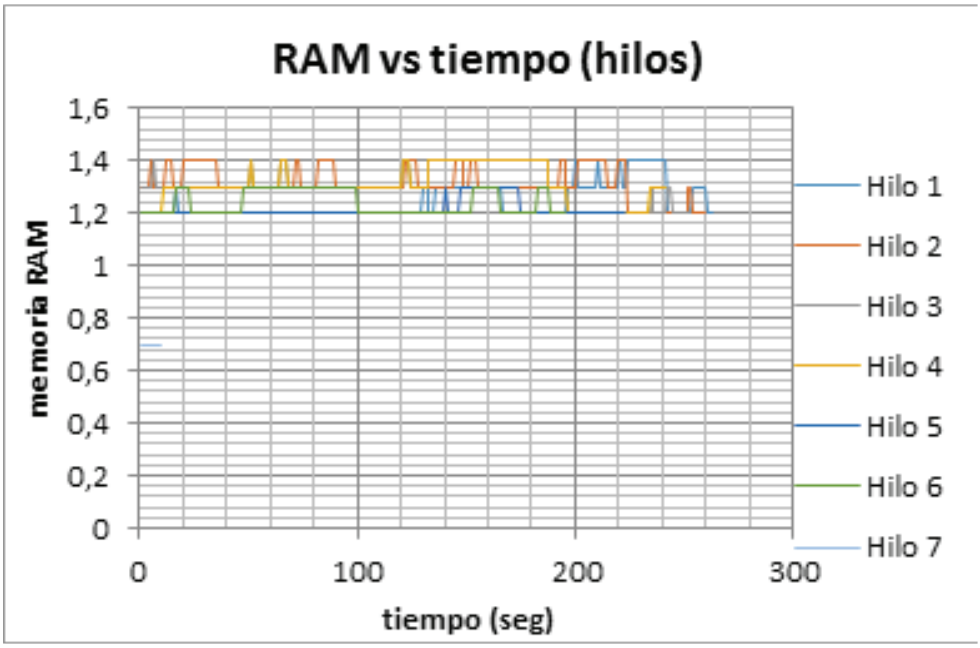

Figura 14. RAM vs tiempo (*FFmpeg*).

En las Figuras 15 y 16 se muestran los resultados de las pruebas de porcentaje de uso de CPU y consumo de memoria RAM, realizadas sobre la herramienta propuesta en este trabajo, durante la codificación de un contenido multimedia WebM a 6 tasas de *bits* diferentes. De acuerdo a la Figura 15, el porcentaje de uso de CPU de la herramienta de codificación presenta un comportamiento creciente entre los 0 y 100 segundos, alcanzando un valor de 20 %. Por su parte entre los 100 y 200 segundos, el porcentaje de uso de CPU se mantiene alrededor del 21 %. Finalmente, después de los 200 segundos, el porcentaje de CPU tiene un comportamiento creciente alcanzando un valor máximo de 26%.

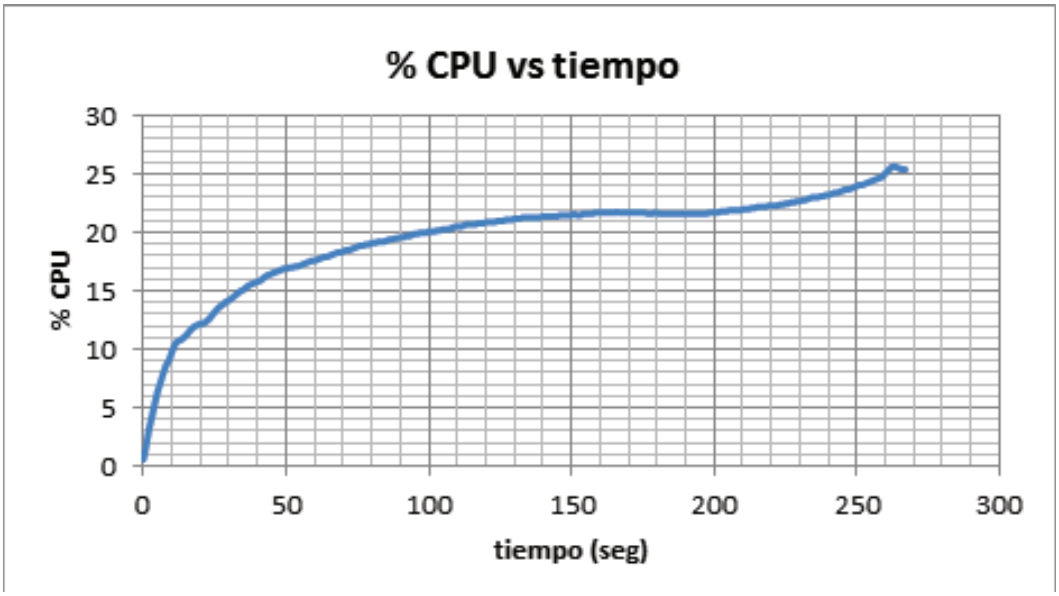

Figura 15. % CPU vs tiempo.

De acuerdo a la Figura 16, el consumo de memoria de la herramienta de codificación se mantiene casi constante hasta los 240 segundos, con un valor de 0.8 MB. Después de los 240 segundos, el valor de consumo de memoria disminuye a 0.7 MB hasta el final del proceso de codificación.

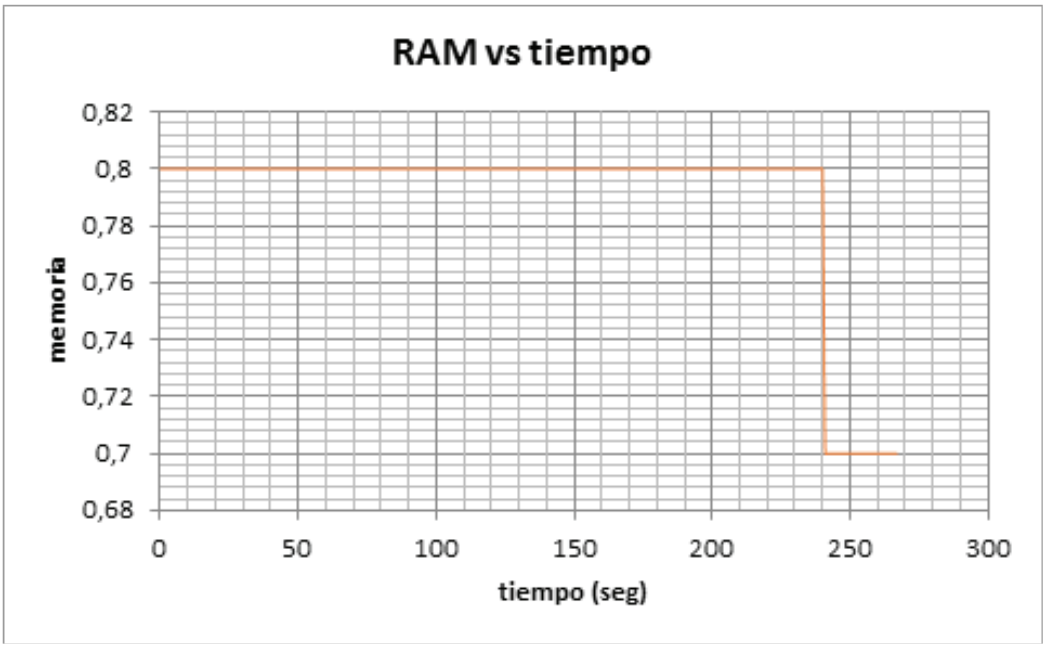

Figura 16. RAM vs tiempo.

7. Conclusiones y trabajos futuros

La herramienta propuesta, permite optimizar el proceso de codificación, segmentación y generación del archivo de manifiesto MPD para contenidos multimedia adaptativos WebM DASH. Para lo anterior, esta herramienta integra en segundo plano las funcionalidades provistas de manera separada por las librerías *libewbm*, *webm-tools* y la herramienta de codificación *FFmpeg*. Así mismo, la herramienta *DashWebMConverter* optimiza el tiempo de codificación al lanzar de manera simultánea tantos hilos como tasas de *bits* estén asociadas a un contenido multimedia.

A través de la herramienta propuesta, es posible el despliegue de un escenario de transmisión de *streaming* adaptativo WebM basado en el estándar DASH. Este escenario tiene como ventaja la reducción de la infraestructura para el montaje de servicios de transmisión de contenido multimedia, al no requerir el uso de un servidor de *streaming*, puesto que los contenidos y la lógica de navegación a través de ellos se encuentra centralizada en un servidor Web, evitando así problemas de sincronización.

De acuerdo a las mediciones de consumo de memoria RAM realizadas sobre la herramienta *FFmpeg*, para un contenido multimedia con 6 tasas de bits a codificar, la cantidad de memoria RAM asignada a cada hilo de video es en promedio de 0.2 MB. Lo anterior permite inferir que el hecho de usar procesamiento multihilo no representa un incremento considerable en la cantidad de memoria RAM, con respecto a las ventajas obtenidas de optimización de tiempo.

La librería *javascriptwebm-dash-javascript* permite el despliegue adecuado del contenido multimedia DASH WebM desde un cliente web. Para lo anterior, esta librería estima el ancho de banda disponible en el lado del cliente, y de acuerdo a este valor obtiene el segmento de contenido multimedia apropiado para la reproducción, según la información disponible en el manifiesto MPD. De esta forma, *Webm-Dash-Javascript* facilita la integración y reproducción de contenidos multimedia WebM en formato DASH a través del componente de video provisto por HTML5.

El uso de lenguaje Python para la construcción de la herramienta *DashWebMConverter*, facilita la invocación en segundo plano de las librerías y herramientas auxiliares: *libwebm*, *webm-tools* y *FFmpeg*, las cuales permiten realizar el proceso de codificación, segmentación y generación del archivo descriptor MPD. Así mismo, el soporte de características multihilo por parte de Python, permite a la herramienta *DashWebMConverter* optimizar el tiempo de codificación, al lanzar de manera simultánea, tantos hilos de codificación de video como tasas de *bits* estén asociadas a un contenido multimedia.

El escenario de *streaming* adaptativo en el que se enmarca la herramienta automática de codificación, recoge e integra las herramientas y librerías más adecuadas del mundo del software libre, para el despliegue de un servicio básico de transmisión y recepción de contenidos multimedia adaptativos WebM.

El escenario de transmisión presentado en este artículo, abre un abanico de posibilidades para la implementación de un servicio de video bajo basado en *streaming* adaptativo DASH, de tal manera que se incluyan, aparte del catálogo de contenidos, servicios interactivos asociados de recomendación y valoración del contenido multimedia.

Finalmente, como trabajo futuro se pretende ampliar el funcionamiento de la herramienta de codificación automática *DashWebMConverter*, de tal manera que se pueda extender la funcionalidad de codificación a contenidos multimedia adaptativos en formato MPEG4.

Agradecimientos

Este trabajo ha contado con el apoyo del programa de Doctorados Nacionales de Colciencias, Convocatoria 528 de 2011.

Referencias

[1] Sandvine Intelligent Broadband Networks, Global Internet Phenomena Report, 2H, 5-23, 2013.

[2] Muller, C.; Lederer, S.; Timmerer, C.; Hellwagner, H., Dynamic adaptative streaming over HTTP/2.0, IEEE Internacional Conference on Multimedia and Expo (ICME), 1-6, 2013.

[3] Ozer, JA., Producing Streaming Video for Multiple Screen Delivery, Ed. Doceo Publishing, 436, 2013.

[4] Chu, D.; Jiang, Ch.; Hao, Z.; Jiang, W., The Design and Implementation of Video Surveillance System Based on H.264, SIP, RTP/RTCP and RTSP, Proceedings of the 2013 Sixth International Symposium on Computational Intelligence and Design (ISCID '13),2, 39-43, 2013.

[5] Muller C., Timmerer C., A VLC media player plugin enabling dynamic adaptive streaming over HTTP, Proceedings of the 19th ACM international conference on Multimedia (MM '11), 723-726, 2011.

[6] 3GPP TS 26.234., Transparent end-to-end Packet-switched Streaming Service (PSS), http://www.3gpp.org/DynaReport/26234.htm., 2010.

[7] ISO/IEC 23009-1:2012., Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats,14,3, 2014.

[8] Microsoft Corporation, IIS Smooth Streaming Technical Overview, http://www.microsoft.com/en-us/download/details.aspx?id=17678, 2009.

[9] May, P.; Pantos, R., HTTP Live Streaming draft-pantos-http-live-streaming-07, http://tools.ietf.org/html/draft-pantos-http-live-streaming-07, 2011.

[10] Adobe, HTTP Dynamic Streaming, http://www.adobe.com/products/hds-dynamic-streaming.html, 2012.

[11] CSI Analysing Converting technologies, DASH it all, Ed. Goran Nastic, 39, 2013.

[12] Núñez, B.C., DASH: Un estandar MPEG para streaming sobre HTTP, Facultatd'Informatica de Barcelona - UniversitatPolitecnica de Catalunya, 2013.

[13] Gambín, T., Desarrollo de un servicio de televisión interactiva HbbTV según el estándar ETSI TS 102 796 v1.1.1, Universidad Politécnica de Cartagena – E.T.S., 111-112, 2012.

[14] Rainer, B.; Lederer, S.; Muller, C.; Timmerer, C., A Seamless Web Integration of Adaptive HTTP Streaming, 20th European Signal Processing Conference (EUSIPCO 2012),1519-1523, 2012.

Sobre los autores

Duvernei Ortiz T. Estudiante de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca, Popayán-Cauca, Colombia.

Gabriel E. Chanchí G. Ingeniero en Electrónica y Telecomunicaciones, Magister en Ingeniería Telemática, Universidad del Cauca, Popayán-Cauca, Colombia, Estudiante de Doctorado en Ingeniería Telemática de la Universidad del Cauca.

*José L. Arciniegas H***.** Ingeniero en Electrónica y Telecomunicaciones, Especialista en Redes y Servicios Telemáticos, Doctor en Ingeniería de Sistemas Telemáticos, Universidad del Cauca, Popayán-Cauca, Colombia, Profesor de planta Facultad de Ingeniería de la Universidad del Quindío. Profesor de planta del programa de Ingeniería Electrónica de la Universidad del Cauca.

Diego F. Duran D. Ingeniero en Electrónica y Telecomunicaciones, Magister en Ingeniería Telemática, Universidad del Cauca, Popayán-Cauca, Colombia, Estudiante de Doctorado en Ingeniería Telemática de la Universidad del Cauca.

Wilmar Y. Campo M. Ingeniero en Electrónica y Telecomunicaciones, Magister en Ingeniería Área Telemática, Doctor en Ingeniería Telemática, Universidad del Quindío, Armenia-Quindío, Colombia, Profesor de planta del programa de Ingeniería Electrónica de la Universidad del Quindío.

Este artículo se cita:

IEEE

D. Ortiz T., G. E. Chanchí G., J. L. Arciniegas H., and D. F. Durán D., “Escenario para la transmisión de streaming adaptativo DASH-WebM,” Rev. Colomb. Comput., vol. 18, no. 1, pp. 27–45, 2017.

APA

Ortiz T., D., Chanchí G., G. E., Arciniegas H., J. L., & Durán D., D. F. (2017). Escenario para la transmisión de streaming adaptativo DASH-WebM. Revista Colombiana de Computación, 18 (1), 27–45.