UNIVERSITAT POLITÈCNICA DE VALÈNCIA
DEPARTAMENTO DE COMUNICACIONES

MEJORA DEL STREAMING DE VÍDEO EN DASH CON CODIFICACIÓN DE BITRATE VARIABLE MEDIANTE EL ALGORITMO LOOK AHEAD Y MECANISMOS DE COORDINACIÓN PARA LA REPRODUCCIÓN, Y PROPUESTA DE NUEVAS MÉTRICAS PARA LA EVALUACIÓN DE LA QOE

TESIS DOCTORAL

Román Belda Ortega

Directores:
Dr. Juan Carlos Guerri Cebollada
Dr. Ismael de Fez Lava

Valencia, España
Marzo 2021

Abstract

This thesis presents several proposals aimed at improving video transmission through the DASH (Dynamic Adaptive Streaming over HTTP) standard.

This research work studies the DASH transmission protocol and its characteristics. At the same time, this work proposes the use of encoding with constant quality and variable bitrate as the most suitable video content encoding mode for on-demand content transmission through the DASH standard.

Based on the proposal to use the constant quality encoding mode, the role played by adaptation algorithms in the user experience when consuming multimedia content becomes more important. In this sense, this thesis presents an adaptation algorithm called Look Ahead which, without modifying the standard, allows the use of the information on the sizes of the video segments included in the multimedia containers to avoid making adaptation decisions that lead to undesirable stalls during the playback of multimedia content.

In order to evaluate the improvements of the presented adaptation algorithm, three models of objective QoE evaluation are proposed. These models allow to predict in a simple way the QoE that users would have in an objective way, using well-known parameters such as the average bitrate, the PSNR (Peak Signal-to-Noise Ratio) and the VMAF (Video Multimethod Assessment Fusion). All of them applied to each segment.

Finally, the DASH behavior in Wi-Fi environments with high user density is analyzed. In this context, there could be a high number of stalls in the playback because of a bad estimation of the available transfer rate due to the ON/OFF pattern of DASH download and to the variability of the access to the Wi-Fi environment. To relieve this situation, a coordination service based on SAND (MPEG's Server and Network Assisted DASH) is proposed, which provides an estimation of the transfer rate based on the information of the state of the clients' players.

Resumen

Esta tesis presenta diversas propuestas encaminadas a mejorar la transmisión de vídeo a través del estándar DASH (*Dynamic Adaptive Streaming over HTTP*).

Este trabajo de investigación estudia el protocolo de transmisión DASH y sus características. A la vez, plantea la codificación con calidad constante y bitrate variable como modo de codificación del contenido de vídeo más indicado para la transmisión de contenido bajo demanda mediante el estándar DASH.

Derivado de la propuesta de utilización del modo de codificación de calidad constante, cobra mayor importancia el papel que juegan los algoritmos de adaptación en la experiencia de los usuarios al consumir el contenido multimedia. En este sentido, esta tesis presenta un algoritmo de adaptación denominado Look Ahead el cual, sin modificar el estándar, permite utilizar la información de los tamaños de los segmentos de vídeo incluida en los contenedores multimedia para evitar tomar decisiones de adaptación que desemboquen en paradas no deseadas en la reproducción de contenido multimedia.

Con el objetivo de evaluar las posibles mejoras del algoritmo de adaptación presentado, se proponen tres modelos de evaluación objetiva de la QoE. Los modelos propuestos permiten predecir de forma sencilla la QoE que tendrían los usuarios de forma objetiva, utilizando parámetros conocidos como el bitrate medio, el PSNR (*Peak Signal-to-Noise Ratio*) y el valor de VMAF (*Video Multimethod Assessment Fusion*). Todos ellos aplicados a cada segmento.

Finalmente, se estudia el comportamiento de DASH en entornos Wi-Fi con alta densidad de usuarios. En este contexto, se producen un número elevado de paradas en la reproducción por una mala estimación de la tasa de transferencia disponible debida al patrón ON/OFF de descarga de DASH y a la variabilidad del acceso al medio de Wi-Fi. Para paliar esta situación, se propone un servicio de coordinación basado en la tecnología SAND (*MPEG's Server and Network Assisted DASH*) que proporciona una estimación de la tasa de transferencia basada en la información del estado de los players de los clientes.

Resum

Aquesta tesi presenta diverses propostes encaminades a millorar la transmissió de vídeo a través de l'estàndard DASH (*Dynamic Adaptive Streaming over HTTP*).

Aquest treball de recerca estudia el protocol de transmissió DASH i les seves característiques. Alhora, planteja la codificació amb qualitat constant i bitrate variable com a manera de codificació del contingut de vídeo més indicada per a la transmissió de contingut sota demanda mitjançant l'estàndard DASH.

Derivat de la proposta d'utilització de la manera de codificació de qualitat constant, cobra major importància el paper que juguen els algorismes d'adaptació en l'experiència dels usuaris en consumir el contingut. En aquest sentit, aquesta tesi presenta un algoritme d'adaptació denominat Look Ahead el qual, sense modificar l'estàndard, permet utilitzar la informació de les grandàries dels segments de vídeo inclosa en els contenidors multimèdia per a evitar prendre decisions d'adaptació que desemboquin en una parada indesitjada en la reproducció de contingut multimèdia.

Amb l'objectiu d'avaluar les possibles millores de l'algoritme d'adaptació presentat, es proposen tres models d'avaluació objectiva de la QoE. Els models proposats permeten predir de manera senzilla la QoE que tindrien els usuaris de manera objectiva, utilitzant paràmetres coneguts com el bitrate mitjà, el PSNR (*Peak Signal-to-Noise Ratio*) i el valor de VMAF (*Video Multimethod Assessment Fusion*). Tots ells aplicats a cada segment.

Finalment, s'estudia el comportament de DASH en entorns Wi-Fi amb alta densitat d'usuaris. En aquest context es produeixen un nombre elevat de parades en la reproducció per una mala estimació de la taxa de transferència disponible deguda al patró ON/OFF de descàrrega de DASH i a la variabilitat de l'accés al mitjà de Wi-Fi. Per a pal·liar aquesta situació, es proposa un servei de coordinació basat en la tecnologia SAND (*MPEG's Server and Network Assisted DASH*) que proporciona una estimació de la taxa de transferència basada en la informació de l'estat dels players dels clients.

Agradecimientos

Quien me conozca sabrá lo poco dado que soy a los agradecimientos por escrito principalmente por mi incapacidad para reflejar correctamente el sentimiento de gratitud en unas pocas palabras que ni hacen ni dejan de hacer.

Sin embargo, esta vez, muy a mi pesar, no soy capaz de encontrar ninguna excusa que me permita saltarme este punto.

Gracias Ismael, gracias Juan Carlos, gracias Pau. “Sin vosotros no estaría aquí”, en este caso, no es una frase hecha.

Índice

Índice de tablas

Índice de figuras

Acrónimos

ABR	Adaptive BitRate
ACR	Absolute Category Rating
C4	Criterion v4.0
CCR	Comparison Category Rating
CDN	Content Delivery Network
CRF	Constant Rate Framework
DANE	DASH-Aware Network Element
DASH	Dynamic Adaptive Streaming over HTTP
DASH-IF	DASH Industry Forum
DCR	Degradation Category Rating
DLM	Detail Loss Metric
DMOS	Differential MOS
DSCQS	Double-Stimulus Continuous Quality Scale
EBML	Extensible Binary Meta Language
FR	Full Reference
GoP	Group of Pictures
HAS	HTTP Adaptive Streaming
HD	High Definition
HLS	HTTP Live Streaming
HTTP	Hypertext Transfer Protocol
IEC	International Electrotechnical Commission
IF	Influence Factor
IP	Internet Protocol
ISO	International Organization for Standardization

ISOBMFF	ISO/IEC Base Media File Format
ITU	International Telecommunication Union
M2TS	MPEG-2 Transport Stream
MOS	Mean Opinion Score
MP4	MPEG-4 Part 14
MPD	Media Presentation Description
MPEG	Moving Picture Experts Group
MUSHRA	Multi-Stimuli with Hidden Reference and Anchor Points
NAT	Network Address Translation
NR	No Reference
OR	Outlier Ratio
PLCC	Pearson Linear Correlation Coefficient
PSNR	Peak Signal-to-Noise Ratio
QoE	Quality of Experience
QP	Quantization Parameter
RR	Reduced Reference
RRIQA	Reduced-Reference Image Quality Assessment
RTCP	RTP Control Protocol
RTP	Real-time Transport Protocol
SAMVIQ	Subjective Assessment of Multimedia Video Quality
SAND	Server and Network Assisted DASH
SARA	Segment-Aware Rate Adaptation
SDN	Software Defined Network
SROCC	Spearman's Rank Correlation Coefficient
SS	Single Stimulus
SSCQE	Single Stimulus Continuous Quality
SSIM	Structural Similarity Index Measure
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TI	Temporal Impairment Feature
TXOP	Transmit Opportunity
UDP	User Datagram Protocol
UHD	Ultra-High Definition

URL	Uniform Resource Locator
VIF	Visual Information Fidelity
VMAF	Video Multimethod Assessment Fusion
VNI	Visual Networking Index
VQA	Video Quality Assessment
VQEG	Video Quality Experts Group
VQF	Video Quality Fairness
VQM	Video Quality Model
XML	Extensible Markup Language

Capítulo 1

Introducción al streaming de vídeo en DASH

1.1. Transmisión de vídeo

La transmisión de vídeo a través de Internet es la principal fuente de tráfico [1] y todo parece indicar que va a seguir aumentando su importancia respecto al resto de tipos de tráficos, tal como refleja el informe de Cisco [1]. En dicho informe se indica que la suma de todas las formas de transmisión de vídeo a través de redes IP, incluyendo el vídeo bajo demanda, el intercambio de archivos de vídeo, los videojuegos en streaming y las videoconferencias, seguirá suponiendo entre el 80 y el 90% de todo el tráfico IP. También predice que el tráfico de vídeo a través de Internet supondrá un 82% en 2022.

En el informe de Cisco, el vídeo sobre Internet y el tráfico generado por juegos son los únicos tipos de contenidos que aumentan su porcentaje respecto al resto de los tipos de tráfico. El aumento tan importante del tráfico generado por la transmisión de vídeo, según Cisco, está motivado por el incremento de la cantidad y las calidades de los vídeos transmitidos donde, en 2022, el vídeo HD (*High Definition*) y UHD (*Ultra High Definition*), conjuntamente, llegaría a representar el 71% del todo el tráfico generado por este tipo de transmisión tal como se puede apreciar en la Figura 1.1.

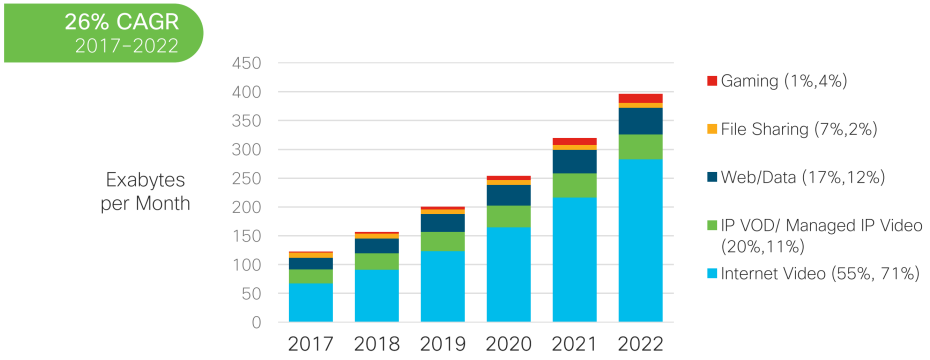


Figura 1.1. Evolución del tráfico IP en Internet. Fuente: *Cisco Visual Networking Index (VNI) Complete Forecast Update [1]*.

En relación con la importancia relativa de las calidades, la Figura 1.2 muestra como el vídeo UHD crecerá hasta representar el 22% de todo el tráfico de vídeo en 2022, mientras que el tráfico de vídeo HD apenas crecerá en importancia relativa. Por su parte, la transmisión de vídeo de calidades inferiores a HD, a pesar de mantener su volumen total, pasará a representar únicamente el 21% en 2022.

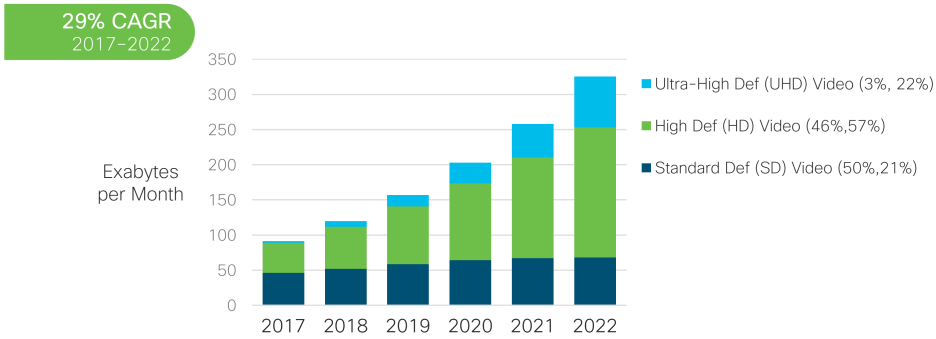


Figura 1.2. Evolución del tráfico IP en Internet de transmisión de vídeo por calidades. Fuente: *Cisco Visual Networking Index (VNI) Complete Forecast Update [1]*.

Estas cifras suponen un reto importante a muchos niveles, desde las redes de los proveedores de Internet a los dispositivos móviles de los usuarios, pasando por los reproductores de vídeo y los codecs de audio y vídeo. Estos retos han sido el motor del desarrollo tecnológico de los últimos años en el mundo de las telecomunicaciones y todo parece indicar que lo seguirá siendo en el futuro. Es por esto por lo que cualquier mejora en el streaming de vídeo puede tener un impacto importante en algunos de los niveles mencionados.

1.2. De servidor a cliente

En la última década ha habido un cambio fundamental en la forma en la que el contenido multimedia se distribuye a través de las redes IP. El elemento clave sobre el que ha pivotado la transmisión de vídeo ha sido el cambio de un modelo basado en servidores de vídeo que envían, de forma activa, el contenido al usuario (modelo servidor) a un modelo en el que es el propio usuario, a través de su reproductor de vídeo, quien solicita el contenido (modelo cliente).

En la Figura 1.3 se muestra el cambio entre el modelo servidor (1) y el modelo cliente (2). Este simple cambio de arquitectura tiene grandes beneficios a la hora de desplegar servicios de streaming de vídeo sobre Internet, tal como se expondrá a continuación.

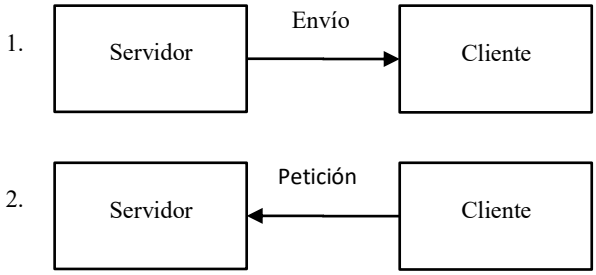


Figura 1.3. Modelos de streaming de vídeo.

En el modelo servidor, para transmitir un contenido bajo demanda, los servidores de streaming son quienes acceden al contenido multimedia, desempaquetan los flujos (*streams*) de audio y vídeo, vuelven a empaquetar los streams en un protocolo multimedia, e.g. RTP (*Real-time Transport Protocol*), y lo envían a los clientes sobre un protocolo de transporte, normalmente UDP (*User Datagram Protocol*). Este proceso se puede ver en la Figura 1.4.

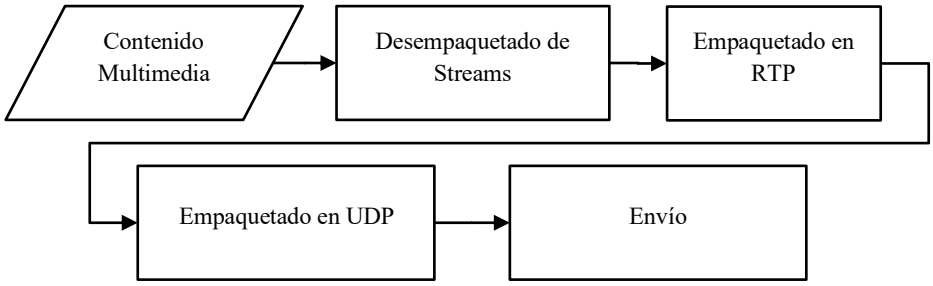


Figura 1.4. Proceso de envío de un stream multimedia en RTP.

El modelo servidor tiene varios inconvenientes, entre los que destacan:

- **Estado de la sesión**. Al ser el servidor quien envía de forma activa el contenido a los clientes es necesario que mantenga información del estado de todas las sesiones en curso.
- **Reempaquetado**. El coste del procesado hace que sea más costosa la escalabilidad de los servidores de este tipo.
- **Sin control de flujo**. Debido a la utilización de UDP como protocolo de transporte, el servidor de vídeo no tiene información directa del estado del canal. Para subsanar este punto es necesaria la utilización de algoritmos de adaptación de RTCP (*Real Time Control Protocol*).
- **Competencia por el ancho de banda**. Debido a la falta de un control de flujo, la correcta selección de la tasa de transferencia queda relegada a la utilización de un algoritmo de adaptación implementado a través de RTCP.
- **Acceso a través de NATs (*Network Address Translation*)**. Al iniciar la transmisión el servidor, si los clientes se encuentran detrás de un NAT pueden tener problemas para recibir los streams en los casos en que no sea posible direccionar correctamente los extremos de la comunicación por encontrarse en redes con direccionamiento IP privado [2].
- **Escalabilidad**. Debido a la utilización de protocolos específicos de la transmisión de contenido multimedia, a la necesidad de entender los formatos de los archivos multimedia y de gestionar el estado de la sesión de los clientes, la escalabilidad del servicio de streaming queda limitada al despliegue de software específico para el streaming en los centros de datos.

Como consecuencia de los inconvenientes descritos y ante la popularización del acceso a contenidos multimedia a través de Internet, surgieron, en un margen relativamente corto de tiempo, varias iniciativas de la industria dirigidas a subsanar los mismos.

En 2009 HLS (*HTTP Live Streaming*) y en 2011 DASH presentan dos alternativas al streaming de contenido bajo demanda basado en el modelo cliente presentado en la Figura 1.5. Estas alternativas tienen en común el cambio de estrategia: el cliente de vídeo es quien realiza las solicitudes de los trozos de vídeo a través de HTTP y permiten el cambio de calidad en la codificación de un trozo a otro.

En la Figura 1.5 se muestra una simplificación de un cliente de vídeo que solicita varios trozos de vídeo al servidor de vídeo.

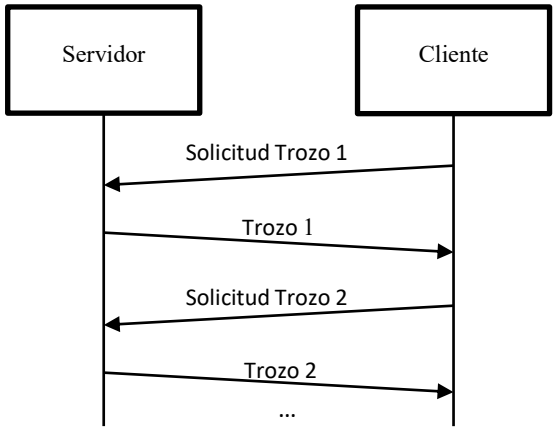


Figura 1.5. Flujo de peticiones/respuestas en HAS streaming.

El cliente de vídeo, conforme va recibiendo los trozos del vídeo, los reproduce de forma secuencial de forma que, si no hay problemas durante el proceso, se produce una reproducción ininterrumpida del vídeo original.

El conjunto de este tipo de técnicas de streaming se denomina HAS (*HTTP Adaptive Streaming*). En contraposición a los inconvenientes presentados en el streaming según el modelo servidor, podemos enumerar los beneficios de HAS:

- **Sin estado de sesión**. Los servidores de streaming quedan reducidos a servidores HTTP que responden a peticiones tipo GET sobre trozos de contenido multimedia. De este modo, dichos servidores no necesitan conocer la relación de las peticiones con los clientes y así ven reducida su carga.
- **Sin reempaquetado**. El servidor no tiene que abrir y leer el contenido de los archivos multimedia para volver a empaquetarlos en el protocolo de transporte, por lo que la carga computacional de servir el contenido es menor.
- **Control de flujo**. Al utilizar HTTP como protocolo de transporte, las peticiones de trozos de contenido multimedia por parte de los clientes están sometidas al control de flujo de TCP.
- **Competencia amigable por el ancho de banda**. Otro punto positivo de utilizar aq TCP es su sistema de equidad, comparada con UDP, en la utilización de la tasa de transferencia disponible.
- **Funcionamiento a través de *firewalls* y *proxies* HTTP**. Debido a la hegemonía de HTTP, este protocolo es ampliamente soportado por firewalls y proxies, de forma que la conectividad es mucho más confiable que con el resto de los protocolos.

- **Escalabilidad**. Al estar basado en HTTP, HAS facilita la integración con las redes de distribución de contenidos existentes y, de esta forma, facilita de manera sustancial la escalabilidad del servicio.

Es por estos motivos por los que la industria de la distribución de contenidos, e.g. YouTube, Netflix, HBO, Amazon Prime Video etc., se ha volcado con HAS como principal medio de distribución de contenido bajo demanda.

1.3. HTTP Adaptive Streaming

En este capítulo se presenta HAS y su especificación por parte de MPEG-DASH, puesto que son la base esta tesis. Debido a la variedad de las tecnologías utilizadas en esta tesis, los capítulos venideros extienden al presente con sus propias secciones de estado del arte que introducen, en cada caso, las tecnologías de relevancia pertinentes.

1.3.1. HAS

HAS se basa en la utilización de HTTP para la transmisión de segmentos de vídeo, que es como se denomina formalmente a los trozos de vídeo. Los clientes de vídeo obtienen dichos segmentos de forma secuencial y, si no se producen problemas en la obtención de los segmentos, los reproducen de forma ininterrumpida. Adicionalmente, cada segmento de vídeo puede estar codificado en distintas calidades de forma que el cliente puede elegir, para cada intervalo de tiempo, la calidad del segmento que desea obtener.

Para que se cumplan estas premisas es necesario que los segmentos de vídeo cumplan dos requisitos:

1. Los distintos segmentos de vídeo de las diferentes calidades deben compartir, exactamente, los mismos intervalos de tiempo a los que hacen referencia del contenido original.
2. Los segmentos de vídeo deben comenzar con un *frame* tipo I (*Intra frame* o *key frame*).

En la Figura 1.6 se muestra una representación de un contenido codificado en 3 calidades diferentes. La figura trata de ilustrar los requisitos sobre los segmentos de vídeo expuestos. Cada rectángulo representa un segmento de vídeo donde la Q identifica la calidad y la C el número de segmento. Se puede observar que todos los segmentos con el mismo número en la C tienen una duración idéntica mientras que varían en altura según a la calidad a la que pertenezcan.

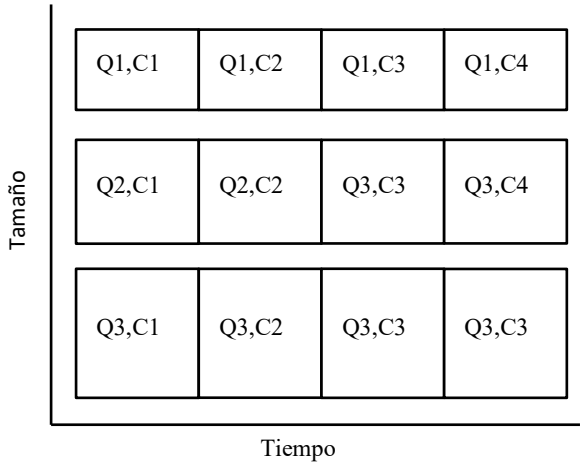


Figura 1.6. Segmentación de vídeo en HAS.

Para que los reproductores de vídeo sean capaces de reproducir un contenido como el representado en la Figura 1.6 necesitan conocer la información relativa a las distintas calidades, los formatos y codecs de los vídeos, así como la información de acceso, entre otras. Esta información está contenida en un archivo de metadatos que, de esta forma, se convierte en el punto de inicio de una reproducción HAS.

Son justamente los metadatos que describen el contenido a reproducir y el formato de los segmentos, las dos definiciones que se introducen en el estándar de MPEG-DASH [3], que se explican con más detalle en el siguiente punto.

1.3.2. DASH

MPEG-DASH es un sistema HAS definido en el estándar ISO/IEC 23009-1:2014 [3] y cuya primera versión se publicó en noviembre de 2011.

El estándar de DASH define el acceso a contenido multimedia a través del protocolo HTTP sin que el servidor de HTTP tenga que disponer de ninguna capacidad extra o especial para la distribución de contenido multimedia. Es por esto por lo que este estándar no trata de definir un procedimiento ni del servidor ni del cliente, sino el formato de los datos que se intercambian. DASH define dos tipos de formatos. Uno en formato de texto XML y el segundo, en formato binario:

- El formato XML define la *Media Presentation Description* (MPD). La MPD, como se verá con más detalle en el punto 1.3.2.2, es una estructura que define la reproducción de un contenido multimedia, también denominado archivo de manifiesto, al que se accede a través de identificadores tipo HTTP-URLs.
- El formato binario especifica el contenido de las respuestas HTTP ante una petición HTTP-GET o un HTTP-GET parcial con rango de bytes.

Así pues, el estándar DASH define el archivo de manifiesto destinado a describir el contenido a reproducir: la MPD y la forma en la que se transmiten los segmentos multimedia.

1.3.2.1. Modelo Cliente

El estándar DASH está basado en el modelo cliente tal como se ha definido en la introducción.

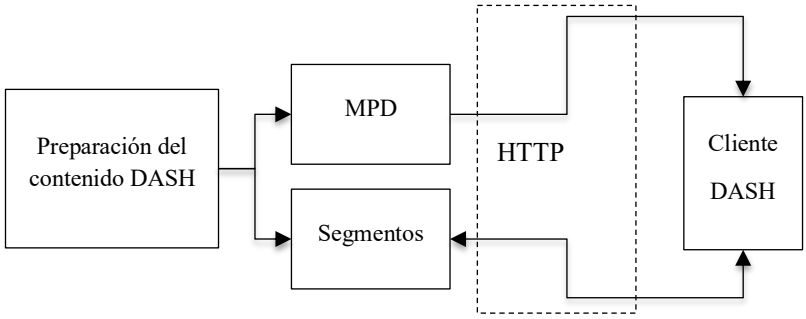


Figura 1.7. Ejemplo de Sistema DASH.

En la Figura 1.7 se muestra un sistema de ejemplo presentado en el propio estándar. En él se puede observar que la transmisión de los segmentos se realiza a través de HTTP. También se muestra como la MPD, que se genera a partir del contenido multimedia, puede transmitirse al cliente de DASH a través de HTTP, de la misma forma que se hace con el contenido multimedia.

En el estándar de DASH se incluye un modelo simplificado de un cliente de DASH tal como se muestra en la Figura 1.8. En la parte 1 del estándar sólo se trata la utilización como contenedor multimedia de los formatos *ISO Base Media File Format* (MP4) [4] y *MPEG-2 Transport Stream* (M2TS) [5], aunque sí que incluye una guía de cómo integrar otros formatos de contenedores con el propio modelo DASH. El modelo de cliente DASH muestra como "*DASH access engine*" es quien obtiene tanto la MPD como los segmentos de vídeo. A continuación, con la información contenida en la MPD relativa al formato de los contenedores se extraen los flujos de vídeo (representado en la figura como MPEG *format media*). Una vez extraídos los flujos multimedia será el componente *Media engine* el encargado de la decodificación para, finalmente, representarlo por la interfaz correspondiente al tipo de medio de que se trate.

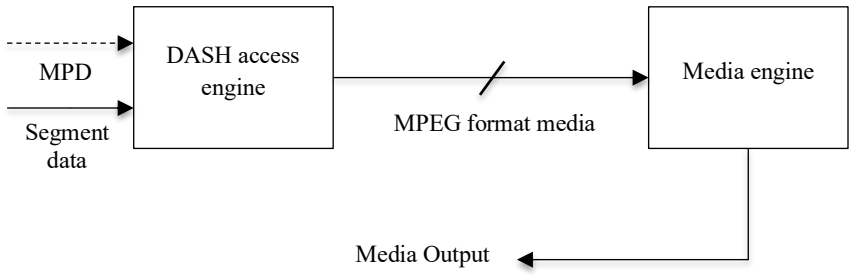


Figura 1.8. Modelo de Cliente DASH.

1.3.2.2. Media Presentation Description

La MPD es un archivo XML que define el contenido de una sesión de streaming DASH. El contenido de una sesión de DASH está compuesto por uno o más periodos (*Period*). Cada periodo representa un contenido continuo en el tiempo que comparte un mismo conjunto de *Media Content Components*. Cada uno de estos componentes puede ser de un tipo diferente: audio, vídeo o subtítulos.

La forma de describir cada uno de los tipos de contenido que forman un periodo es a través de los *AdaptationSets*. Cada *AdaptationSet* define un tipo de contenido con una o más representaciones donde cada una de ellas puede tener diferentes características de codificación. A su vez, cada representación dispondrá de uno o más segmentos.

Los segmentos también pueden estar divididos en subsegmentos. En este caso, la división está condicionada al formato del contenido. Por ejemplo, en el formato *ISO Base Media Format* un segmento debe contener un número entero de subsegmentos. Si un segmento se encuentra dividido en subsegmentos, el segmento deberá indicar un rango de bytes donde poder obtener los índices y tamaños de cada subsegmento.

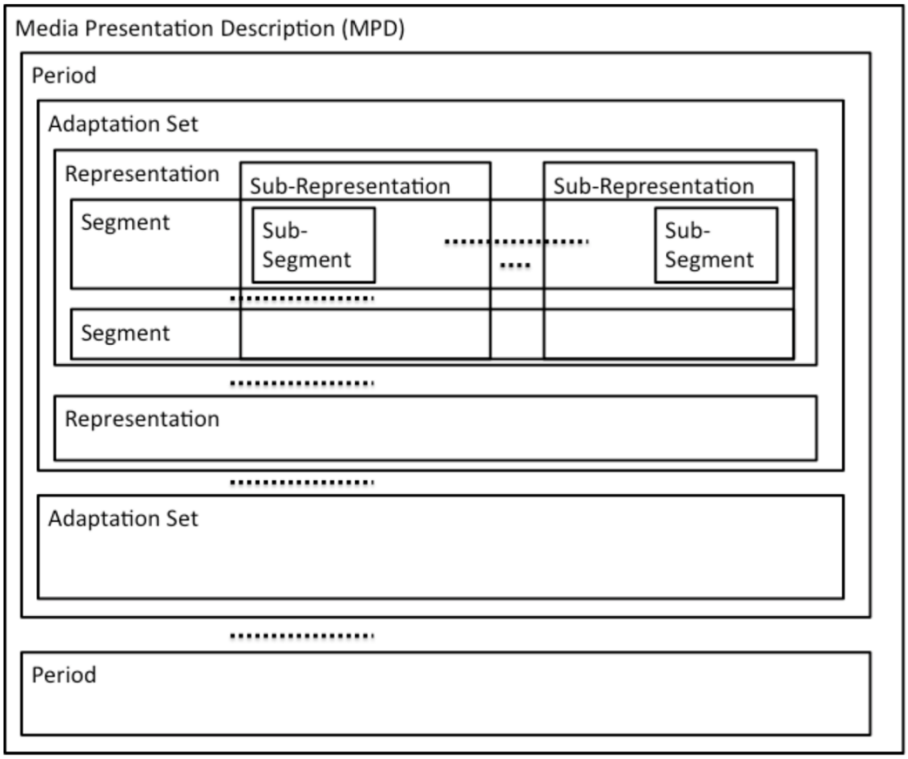


Figura 1.9. Formato de la MPD. Fuente: *Dynamic Adaptive Streaming Over HTTP (DASH)—Part 1 [3]*.

En la Figura 1.9 se muestra el esquema del formato de la MPD proporcionado en el estándar de DASH y la Figura 1.10 contiene un ejemplo de MPD de un vídeo con diferentes calidades de vídeo, una pista de audio en una única calidad y subtítulos en varios idiomas.

```
1.  <MPD xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 DASH-
    MPD.xsd" profiles="urn:mpeg:dash:profile:isoff-on-
    demand:2011" minBufferTime="PT2S" type="static" mediaPresentationDuration="PT734.16S">
2.    <Period id="0">
3.      <AdaptationSet id="0" contentType="audio" lang="en" subsegmentAlignment="true">
4.        <Representation id="0" bandwidth="68257" codecs="mp4a.40.2" mimeType="audio/mp4" a
    udioSamplingRate="44100">
5.          <AudioChannelConfiguration schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_conf
    iguration:2011" value="2"/>
6.          <BaseURL>audio-en.mp4</BaseURL>
7.          <SegmentBase indexRange="1078-1997" timescale="44100">
8.            <Initialization range="0-1077"/>
9.          </SegmentBase>
```

```
10.         </Representation>
11.       </AdaptationSet>
12.       <AdaptationSet id="1" contentType="video" maxWidth="1024" maxHeight="426" frameRate=
    "12800/512" subsegmentAlignment="true" par="12:5">
13.         <Representation id="1" bandwidth="254977" codecs="avc1.42c01f" mimeType="video/mp4
    " sar="265:267" width="256" height="106">
14.           <BaseURL>video-2.mp4</BaseURL>
15.           <SegmentBase indexRange="1242-2005" timescale="12800">
16.             <Initialization range="0-1241"/>
17.           </SegmentBase>
18.         </Representation>
19.         <Representation id="2" bandwidth="454957" codecs="avc1.42c01f" mimeType="video/mp4
    " sar="265:267" width="512" height="212">
20.           <BaseURL>video-1.mp4</BaseURL>
21.           <SegmentBase indexRange="1246-2009" timescale="12800">
22.             <Initialization range="0-1245"/>
23.           </SegmentBase>
24.         </Representation>
25.         <Representation id="3" bandwidth="803654" codecs="avc1.42c01f" mimeType="video/mp4
    " sar="355:356" width="1024" height="426">
26.           <BaseURL>video-0.mp4</BaseURL>
27.           <SegmentBase indexRange="1244-2007" timescale="12800">
28.             <Initialization range="0-1243"/>
29.           </SegmentBase>
30.         </Representation>
31.       </AdaptationSet>
32.       <AdaptationSet id="3" mimeType="text/vtt" lang="en">
33.         <Representation id="caption_en" bandwidth="254">
34.           <BaseURL>subtitle-en.vtt</BaseURL>
35.         </Representation>
36.       </AdaptationSet>
37.       <AdaptationSet id="4" mimeType="text/vtt" lang="es">
38.         <Representation id="caption_es" bandwidth="254">
39.           <BaseURL>subtitle-es.vtt</BaseURL>
40.         </Representation>
41.       </AdaptationSet>
42.       <AdaptationSet id="5" mimeType="text/vtt" lang="no">
43.         <Representation id="caption_no" bandwidth="254">
44.           <BaseURL>subtitle-no.vtt</BaseURL>
45.         </Representation>
46.       </AdaptationSet>
47.       <AdaptationSet id="6" mimeType="text/vtt" lang="ru">
48.         <Representation id="caption_ru" bandwidth="254">
49.           <BaseURL>subtitle-ru.vtt</BaseURL>
50.         </Representation>
51.       </AdaptationSet>
52.     </Period>
53.   </MPD>
```

Figura 1.10. Ejemplo de MPD.

En el ejemplo anterior se muestra una MPD de un contenido multimedia con un único periodo (`<Period id="0">`). Los periodos, dentro de una MPD pueden ser utilizados con diferentes objetivos, desde la creación de una lista de reproducción hasta la inserción de anuncios. El periodo mostrado contiene un *AdaptationSet* para las diferentes calidades del vídeo, otro para la única calidad disponible para el audio y cuatro más para cada uno de los idiomas disponibles para los subtítulos.

Dentro de un *AdaptationSet* se encuentran lo que hemos llamado las diferentes calidades de ese tipo de contenido. Sin embargo, más que calidades deberían llamarse representaciones (del inglés *Representation*) ya que, a pesar de que lo más frecuente es justamente insertar diferentes representaciones para proporcionar diferentes calidades, también es posible y común la utilización de las representaciones para proporcionar el contenido codificado con calidades parecidas, pero utilizando codecs diferentes.

En la definición de una representación como por ejemplo "`<Representation id="3" bandwidth="803654" codecs="avc1.42c01f" mimeType="video/mp4" sar="355:356" width="1024" height="426">`" cabe destacar el parámetro `bandwidth` que indica la tasa media de la representación (en bits por segundo). Este parámetro es ampliamente utilizado para clasificar el orden de las calidades de las representaciones. Además, se emplea por los mecanismos de adaptación, como se analizará en el capítulo siguiente.

Adicionalmente, parámetros como la resolución del contenido de vídeo, así como el codec con el que está codificado el vídeo, son de utilidad al player para discernir qué representaciones deben ser consideradas para su reproducción.

1.3.2.3. Segmentación

Para que un cliente de vídeo DASH pueda acceder a un contenido multimedia, éste debe de estar segmentado para que el cliente pueda realizar las peticiones del segmento que corresponda de forma independiente del resto de segmentos.

El proceso de segmentación depende enteramente del contenedor de vídeo. Se describen a continuación dos de los contenedores más utilizados hoy en día: ISOBMFF y WebM.

1.3.2.3.1. ISO Base Media File Format

ISO Base Media File Format, también llamado mp4 debido a la extensión típica de este tipo de archivos, es uno de los contenedores más utilizados para el almacenamiento de contenido multimedia.

Los archivos de este tipo estructuran su contenido según se muestra en la Figura 1.11.

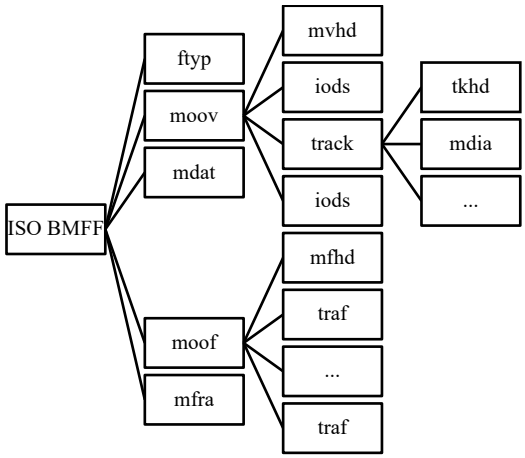


Figura 1.11. Estructura de un archivo ISOBMFF.

A cada elemento que forma parte de la estructura del archivo se le denomina *box* y se identifica con una etiqueta de cuatro letras. A continuación, se describen las principales etiquetas que se muestran en la Figura 1.11:

- ***ftyp*** (*File Type*): Debe ser única por archivo y contiene información de la versión y la compatibilidad del archivo.
- ***mdat*** (*Media Data*): Contiene información no estructurada.
- ***moov*** (*Movie*): Debe ser única por archivo y contiene información del contenido multimedia como las escalas de tiempos, duraciones e información de las características de la imagen de vídeo.
- ***moof*** (*Movie Fragment*): Inicialmente el formato requería que todo el contenido estuviera incluido en la etiqueta *moov*. Con la introducción de *moof*, un archivo mp4 pasa a poder contener un número indeterminado de segmentos de contenido multimedia.
- ***mfra*** (*Movie Random Fragment Access*): Esta estructura contiene la información necesaria para el acceso directo a los distintos fragmentos de contenido multimedia que contenga el archivo.

Para fragmentar este tipo de contenedor se pueden tomar dos alternativas: 1) dividir el contenedor en varios sub-contenedores donde cada archivo contiene un segmento más un contenedor inicial con la información relativa al propio contenido; y 2) modificar el contenedor para que incluya la información relativa a la segmentación.

```
[ftyp] size=8+24
  major_brand = isom
  minor_version = 200
  compatible_brand = isom
  compatible_brand = iso2
  compatible_brand = avc1
  compatible_brand = mp41
[free] size=8+0
[mdat] size=8+148838990
[moov] size=8+207977
```

Figura 1.12. Estructura de un contenedor ISOBMFF sin segmentar.

```
[ftyp] size=8+28
  major_brand = isom
  minor_version = 200
  compatible_brand = isom
  compatible_brand = iso2
  compatible_brand = avc1
  compatible_brand = mp41
  compatible_brand = iso5
[moov] size=8+703
[moof] size=8+388
[mdat] size=8+2074
[moof] size=8+804
[mdat] size=8+6427
[moof] size=8+796
[mdat] size=8+6885
 …
[mfra] size=8+1811
```

Figura 1.13. Estructura de un contenedor ISOBMFF segmentado.

La Figura 1.12 y la Figura 1.13 muestran las estructuras de un contenedor ISOBMFF sin segmentar y segmentado, respectivamente. Se puede apreciar como cuando el contenedor no está segmentado hay un único par de boxes *moov* y *mdat*, donde se encuentra todo el contenido multimedia, mientras que el contenedor segmentado dispone de un *moov* y una pareja de *moof*/*mdat* por cada segmento contenido. Adicionalmente, el contenedor ISOBMFF segmentado dispone de un último box de tipo *mfra*. Esta estructura contiene los índices a cada uno de los segmentos del contenedor de forma que accediendo primero a esta estructura es posible conocer la posición de los segmentos dentro del contenedor.

1.3.2.3.2. WebM

WebM es un contenedor multimedia surgido de WebM Project [6]. Este proyecto fue inicialmente fomentado por Google mediante la compra de On2 y la liberación de su propiedad intelectual sobre el codec de vídeo VP8 [7].

El formato del contenedor WebM es un subconjunto del de Matroska [8] que sólo soporta códecs de código libre como son Vorbis [9] y Opus [10] para el audio, y VP8 y VP9 [11][12] para el vídeo.

Matroska, y por lo tanto WebM, utiliza EBML (*Extensible Binary Meta Language*) para formatear los datos y, al ser de código abierto, cuenta con toda su especificación publicada en su web que, además, es de muy buena calidad.

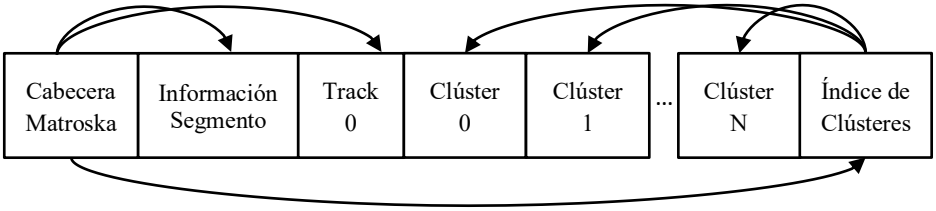


Figura 1.14. Estructura de un contenedor WebM segmentado.

La Figura 1.14 muestra una simplificación de la estructura de un archivo WebM. En ella se puede observar de izquierda a derecha:

- **Cabecera Matroska**: Contiene información acerca de las versiones de EBML y Matroska. así como índices al resto de elementos salvo a los clústeres.
- **Información Segmento**: Pese a su nombre, no tiene relación con los segmentos de DASH. Contiene identificadores del contenido y de contenido previo y subsiguiente si los hubiera.
- **Track**: Información básica de los tracks de contenidos como el tipo de contenido, y detalles de la codificación dependiente del tipo de contenido.
- **Clúster**: Los clústeres son los elementos donde se alojan los contenidos codificados.
- **Índice de Clusters**: El índice de clústeres contiene información para acceder de forma aleatoria a cualquiera de los clústeres del contenedor. De esta forma, se habilita la búsqueda rápida.

Cada clúster de Matroska puede contener un número entero de GoPs (*Group of Pictures*). La agrupación de los GoPs en clústeres es arbitraria y en la documentación sólo se menciona que la implementación de referencia intenta hacer clústeres que contengan 5 segundos o 5 megabytes.

El único requisito para que un archivo de Matroska pueda usarse para streaming DASH es que contenga, efectivamente, el índice de clústeres o *Cueing Data* que es como se denomina en la documentación oficial.

Sin embargo, normalmente, el proceso de creación de contenido DASH basado en WebM incluye un proceso extra para reorganizar la estructura de una manera más conveniente. Dos puntos son sobre los que se suele actuar:

- La posición del segmento de índice de clústeres en el contenedor.
- La organización de los clústeres.

En relación con lo primero, al crear un contenido, el codificador no sabe de antemano el tamaño que van a tener los clústeres, por lo que la creación del elemento de *Cueing Data*

se retrasa hasta haberse procesado todo el contenido. Así pues, el proceso de segmentación suele incluir mover esta información al inicio del archivo. A continuación, se muestran dos figuras que detallan la estructura del mismo contenido antes y después de ser segmentado.

```
+ EBML head
|+ EBML version: 1
|+ EBML read version: 1
|+ Maximum EBML ID length: 4
|+ Maximum EBML size length: 8
|+ Document type: webm
|+ Document type version: 2
|+ Document type read version: 2
+ Segment: size 8564707
|+ Seek head (subentries will be skipped)
|+ EBML void: size 154
|+ Segment information
| + Timestamp scale: 1000000
| + Multiplexing application: Lavf58.20.100
| + Writing application: Lavf58.20.100
| + Duration: 00:05:00.027000000
|+ Tracks
| + Track
|  + Track number: 1 (track ID for mkvmerge & mkvextract: 0)
|  + Track UID: 1
|  + Lacing flag: 0
|  + Language: eng
|  + Codec ID: V_VP9
|  + Track type: video
|  + Default duration: 00:00:00.041666666 (24.000 frames/fields per second for a video
track)
|  + Video track
|   + Pixel width: 1280
|   + Pixel height: 534
|   + Video colour information
|    + Colour range: 1
|    + Horizontal chroma siting: 1
|    + Vertical chroma siting: 2
|+ Cluster
|+ Cues (subentries will be skipped)
```

Figura 1.15. Estructura de un archivo WebM sin segmentar.

```
+ EBML head
|+ EBML version: 1
|+ EBML read version: 1
|+ Maximum EBML ID length: 4
|+ Maximum EBML size length: 8
|+ Document type: webm
|+ Document type version: 4
|+ Document type read version: 2
+ Segment: size 8562156
|+ Seek head (subentries will be skipped)
|+ EBML void: size 25
|+ Segment information
| + Timestamp scale: 1000000
| + Duration: 00:04:59.999000000
| + Multiplexing application: libwebm-0.2.1.0
| + Writing application: https://github.com/google/shaka-packager version 796974d-
release
|+ Tracks
| + Track
```

```
|  + Track number: 1 (track ID for mkvmerge & mkvextract: 0)
|  + Track UID: 1
|  + Track type: video
|  + Codec ID: V_VP9
|  + Codec's private data: size 12
|  + Video track
|   + Pixel width: 1280
|   + Pixel height: 534
|   + Display width: 1280
|   + Display height: 534
|+ Cues (subentries will be skipped)
|+ Cluster
```

Figura 1.16. Estructura de un archivo WebM segmentado.

En la Figura 1.15 y en la Figura 1.16 se observa como el elemento *Cues* pasa de la última posición en la Figura 1.15 a la penúltima posición en la Figura 1.16. A pesar de parecer que sigue lejos de las primeras posiciones del archivo, este cambio de posición es muy importante pues el grueso del tamaño de estos archivos está en el elemento *Cluster*.

El segundo punto sobre el que se actúa al segmentar un archivo WebM es sobre la organización de los clústeres. El proceso de reestructuración afecta a como se organizan los GoPs de los flujos de vídeo en Clústeres de Matroska pero no modifica los GoPs, pues esto requeriría de una recodificación del flujo. De esta forma, el proceso de segmentación tiene que atenerse a las características del flujo multimedia y, en la medida de lo posible, intentar alcanzar el objetivo marcado en la segmentación.

A modo de ejemplo, la Figura 1.17 muestra las entradas iniciales del segmento *Cueing Data* de un archivo WebM segmentado a 10s mientras que la Figura 1.18 contiene la misma información, pero, esta vez, el tamaño de segmentación elegido es de 20 s.

```
timestamp=00:00:00.028000000 duration=- clúster_position=807 relative_position=-
timestamp=00:00:10.695000000 duration=- clúster_position=215596 relative_position=-
timestamp=00:00:21.361000000 duration=- clúster_position=514819 relative_position=-
timestamp=00:00:32.028000000 duration=- clúster_position=857729 relative_position=-
timestamp=00:00:42.695000000 duration=- clúster_position=1164891 relative_position=-
timestamp=00:00:53.361000000 duration=- clúster_position=1492100 relative_position=-
timestamp=00:01:04.028000000 duration=- clúster_position=1763258 relative_position=-
timestamp=00:01:14.695000000 duration=- clúster_position=2081946 relative_position=-
timestamp=00:01:20.028000000 duration=- clúster_position=2237968 relative_position=-
timestamp=00:01:30.695000000 duration=- clúster_position=2532022 relative_position=-…
```

Figura 1.17. Estructura de un segmento de Cueing Data con segmentación a 10s.

```
timestamp=00:00:00.028000000 duration=- clúster_position=555 relative_position=-
timestamp=00:00:21.361000000 duration=- clúster_position=514541 relative_position=-
timestamp=00:00:42.695000000 duration=- clúster_position=1164587 relative_position=-
timestamp=00:01:04.028000000 duration=- clúster_position=1762928 relative_position=-
timestamp=00:01:20.028000000 duration=- clúster_position=2237610 relative_position=-
timestamp=00:01:41.361000000 duration=- clúster_position=2832064 relative_position=-
timestamp=00:02:02.695000000 duration=- clúster_position=3420379 relative_position=-
timestamp=00:02:24.028000000 duration=- clúster_position=4033404 relative_position=-
timestamp=00:02:40.028000000 duration=- clúster_position=4533627 relative_position=-
timestamp=00:03:01.361000000 duration=- clúster_position=5138436 relative_position=-…
```

Figura 1.18. Estructura de un segmento de Cueing Data con segmentación a 20s.

En la Figura 1.17 y en la Figura 1.18 se puede observar que, si bien se aproximan bastante, los instantes de tiempo en los que comienza cada clúster no están separados exactamente por el tamaño de segmento elegido en cada caso, pero cuando coinciden las estampas de tiempo de dos clústeres (uno de cada segmentación) lo hacen exactamente, denotando así que hacen referencia al mismo GoP del contenido.

Esto no quiere decir que los tamaños de los segmentos no puedan ser exactamente los mismos para todos ellos si no que, para que esto ocurra, es necesario tomar medidas a la hora de codificar los contenidos para forzar el tamaño de los GoPs. En el ejemplo mostrado no se forzó ningún tamaño de GoP, por lo que el proceso de codificación decidió el tamaño de cada GoP en función de las características del contenido a codificar.

1.4. Codificación de Vídeo

Un vídeo sin comprimir en formato 1080p24 (una resolución de 1920x1080 con 24 frames por segundo) vendría a ocupar unos 142 MB por cada segundo, por lo que se necesitaría de una tasa de transferencia de casi 1200 Mbps para poder realizar una transmisión en tiempo real. Debido a esto, salvo en el caso de los productores de contenidos que suelen guardar el contenido original sin pérdidas, todo el contenido que se almacena y se transmite por las redes está comprimido con algún codec de vídeo. En estos momentos, los codecs de vídeo más populares son MPEG2, H.264, H.265, VP9 y AV1. Todos ellos son codecs con pérdidas puesto que no es posible reconstruir exactamente el contenido original a partir de su versión codificada.

Todos estos codecs tienen la capacidad de crear contenido en diferentes calidades según las necesidades y el caso de uso. Y han sido precisamente los casos de uso los que han marcado los dos modos en los que estos codecs pueden funcionar: bitrate constante y bitrate variable. Por un lado, el modo de funcionamiento de bitrate constante tiene como objetivo ajustarse, en función de los parámetros admitidos, a una tasa de bits determinada con el objetivo de que no haya grandes variaciones en dicha tasa. Por el otro, el modo de bitrate variable utiliza como parámetro de entrada un objetivo de calidad de la codificación que intentará mantener ante los cambios de compresibilidad de las sucesiones de imágenes a costa de introducir cambios en la tasa de bits resultante.

Puesto que ambos modos de funcionamiento de los diferentes codecs han sido ideados e implementados por sus desarrolladores desde un inicio y con un fin claro, no podemos decir que uno de los modos sea más natural o de un orden mayor que el otro. Sin embargo, si prestamos atención al criterio de semejanza del contenido codificado al contenido original, sí que podemos decir que el modo de funcionamiento de bitrate variable es más constante en su relación con el contenido original que el modo de bitrate constante.

Si tomamos como ejemplo el vídeo “Elephants Dream” [13] y lo codificamos con el codec VP9 con ambos modos, podemos obtener dos gráficas como las que se muestran en la Figura 1.19 y la Figura 1.20. En ambas se compara la codificación del vídeo original en formato Y4M con un vídeo codificado con bitrate variable, usando el parámetro CRF

(*Constant Rate Factor*) con un valor de 45, y con otro codificado con bitrate constante. En ambos casos el bitrate medio de los vídeos es de 1,13 Mbps.

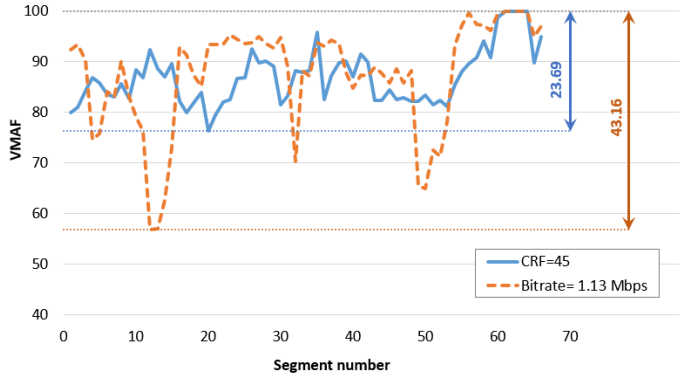


Figura 1.19. Comparación del VMAF de cada segmento del vídeo "Elephants Dream" codificado a calidad constante (CRF 45) y bitrate constante (1,13 Mbps).

La Figura 1.19 muestra el valor VMAF (*Video Multimethod Assessment Fusion*) de cada segmento de vídeo. VMAF [14]-[16], como se explicará con más detalle en el Capítulo 3 es una medida objetiva que trata de predecir la calidad de un vídeo según perciben los usuarios las degradaciones producidas por el proceso de codificación. El valor de VMAF, en el eje Y, está en la escala 0-100 donde un valor mayor representa una mayor calidad percibida por los usuarios. En la gráfica podemos observar como la variación de la calidad de los segmentos es bastante menor en el vídeo codificado con bitrate variable que en el codificado con bitrate constante. En particular, según está representado, entre el valor máximo y el mínimo de cada tipo de codificación existe, en este caso, una diferencia de casi el doble de variación de la calidad en la codificación con bitrate constante respecto a la codificación con bitrate variable.

El significado de esta figura se complementa con la Figura 1.20 en la que podemos apreciar el tamaño de los segmentos resultantes para los mismos vídeos. La gráfica muestra en el eje X el número de segmento de los vídeos y en el eje Y el tamaño en MB.

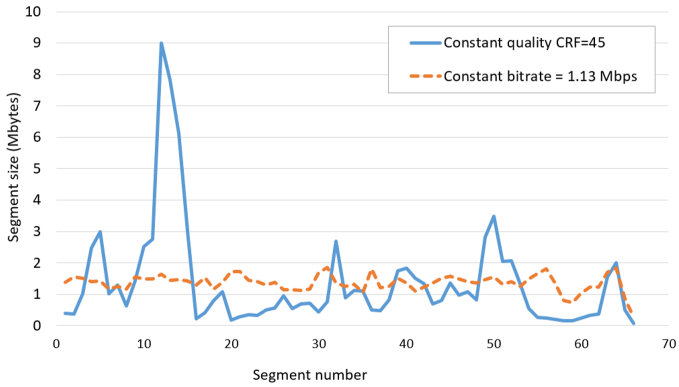


Figura 1.20. Comparación del tamaño de cada segmento del vídeo "Elephants Dream" codificado a calidad constante (CRF 45) y bitrate constante (1,13 Mbps).

Si nos centramos en los segmentos del 10 al 15 podemos observar el pico más significativo que se produce en cada gráfica. Por un lado, en la Figura 1.19 se muestra una bajada drástica de la calidad del vídeo codificado con bitrate constante mientras en la Figura 1.20 se muestra un aumento muy importante del tamaño de los segmentos en el vídeo codificado con bitrate variable. El mismo tipo de correlación, aunque de menor intensidad, se puede observar en el resto de los picos de estas gráficas. Por supuesto, estas variaciones vienen determinadas por el contenido en sí, sus características espaciales y temporales y cómo afronta su compresión el codec en cuestión.

Esta contraposición entre la calidad y el tamaño ante los cambios en la compresibilidad del vídeo es ampliamente conocida [17] y motivo primordial de la propia existencia de los dos modos de funcionamiento de los codificadores de vídeo.

La codificación con bitrate constante, desde el punto de vista del autor, ha eclipsado a la codificación con calidad constante en tantos escenarios que ha llegado a perderse de vista las características y consecuencias de su utilización.

Existen muchos casos de uso donde la codificación con bitrate constante está claramente justificada. La televisión digital, por ejemplo, dispone de canales unidireccionales de un ancho de banda determinado, por lo que podría producir problemas si se intentara transmitir un stream de vídeo en el que, durante un periodo de tiempo, su bitrate instantáneo superase dicha tasa. También, en el mismo escenario, sería más difícil establecer un parámetro de calidad constante puesto que dicho parámetro puede generar streams con bitrates muy diferentes en función de las características del contenido. En la Figura 1.21 se muestran los tamaños de los segmentos para dos vídeos codificados con el mismo parámetro de calidad constante para dos vídeos diferentes ("Elephants Dream" y "Tears of Steel"). En el eje X está representado el número de segmento, y su correspondiente tamaño en MB en el eje Y. Adicionalmente, se muestra con líneas discontinuas el bitrate medio de cada vídeo. En la gráfica se puede observar que, pese a estar codificados con el mismo parámetro de calidad constante, tanto el perfil de tamaños

de los segmentos como el bitrate medio de los vídeos es muy diferente. Esto es debido a las características intrínsecas de los vídeos en cuestión.

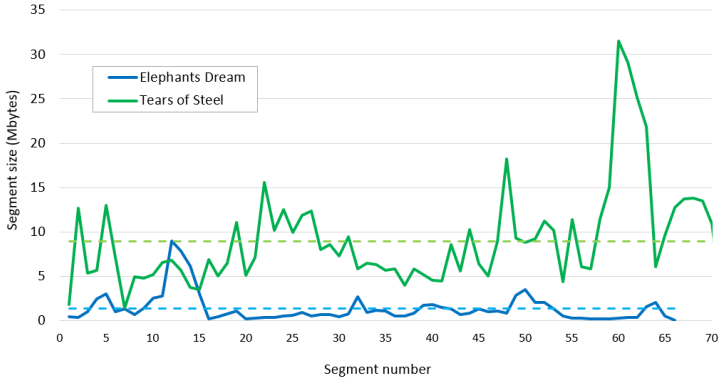


Figura 1.21. Comparación del tamaño de los segmentos de los vídeos "Elephants Dream" y "Tears of Steel" codificados con el mismo parámetro de calidad constante.

Otro escenario en el que la codificación con tasa constante tiene claros beneficios es en el sector de las videoconferencias. Aquí, la importancia de que exista el mínimo retardo posible entre el emisor y el receptor invita a utilizar protocolos de transmisión no orientados a la conexión. En estos casos, es recomendable evitar la generación de picos de transmisión para no poner a prueba las redes de comunicación y así prevenir, en la medida de lo posible, la pérdida de información en la transmisión.

En estos ejemplos, la utilización de streams de vídeo codificados con tasa constante tienen un objetivo claro puesto que la alternativa podría poner en riesgo el propio servicio. Sin embargo, **el beneficio de conocer el bitrate resultado de la codificación a priori se ha extendido a casos de uso donde no es estrictamente necesario, jugando un papel de simplificación de la problemática**.

Uno de los casos de uso más evidentes donde la codificación con tasa constante simplifica la problemática es en el almacenamiento de vídeo en un medio físico. En este caso, si nuestro objetivo es guardar una película con la mejor calidad posible en un medio de almacenamiento de un tamaño fijo podemos tomar dos aproximaciones:

1. Dividir el espacio de almacenamiento del que se dispone entre la duración de la película.
2. Realizar una búsqueda del parámetro de calidad constante que, para la película en cuestión, genere una codificación con el tamaño deseado.

Es evidente que la segunda aproximación es mucho más costosa puesto que requiere un proceso de prueba y error muy caro en términos computacionales, pero la primera aproximación tiene costes relacionados con la calidad de la codificación generada.

En la práctica, es difícil ignorar los costes de la segunda aproximación ya que requiere un trabajo extra, pero la primera aproximación, sin ser intrínsecamente menos apta, puede llevar a desestimar sus costes asociados en términos de calidad y su variación.

En el caso del vídeo bajo demanda, objeto de este trabajo de investigación, un contenido, tras codificarse, se pone a disposición de los usuarios. De esta forma, el proceso de codificación cobra especial importancia puesto que cualquier decisión repercutirá a todos los que consuman el contenido en cuestión.

Teniendo en cuenta el escenario, podemos analizar las ventajas y los inconvenientes de ambos modos de codificación.

Por un lado, la codificación con bitrate constante:

- Genera un contenido cuyo tamaño final puede ser predicho de antemano.
- La variación del tamaño de los segmentos es menor.
- La variación de la calidad de los segmentos es mayor.

Y por el otro, la codificación con bitrate variable:

- El tamaño final de un contenido a una calidad determinada dependerá de forma importante de las características del contenido.
- La variación del tamaño de los segmentos es mayor.
- La variación de la calidad de los segmentos es menor.

Los dos modos de codificación difieren en tres puntos: tamaño final de una codificación, variación del tamaño de los segmentos y variación de la calidad.

Puesto que estamos considerando el caso de una transmisión de vídeo a través de DASH, el tamaño final de cada una de las calidades es menos importante ya que los usuarios tienen a su disposición varias de ellas. Incluso, si el tamaño, por alguna razón, debiera estar determinado, la naturaleza del streaming bajo demanda donde se codifica una vez para poner a disposición de un, potencialmente, gran número de usuarios, hace que pueda ser factible generar un elevado número de calidades para, a posteriori, ver cuál se adapta mejor al tamaño requerido.

Como hemos visto anteriormente, la variación del tamaño de los segmentos y la variación de la calidad están inversamente relacionadas. Con esto, la disyuntiva a la hora de codificar queda entre minimizar la variación de la calidad o minimizar la variación de los tamaños de los segmentos.

Finalmente, mientras que la mayor fluctuación de la calidad es una variable que únicamente se constata en el usuario que consume el contenido, la de los tamaños de los segmentos supone elevar la variación de una variable que, de no tener en consideración, puede provocar problemas en la reproducción.

Es por esto por lo que en la literatura [18]-[20] se encuentran numerosos ejemplos de codificación con bitrate constante mientras que la codificación con bitrate variable está presente en menos estudios.

1.5. Objetivos y Contribuciones

El principal objetivo de esta tesis es:

> ***Evaluar las implicaciones de la utilización de contenido codificado con calidad constante en el streaming de vídeo utilizando el estándar DASH y proponer mejoras con el objetivo de aumentar la calidad de experiencia percibida por los usuarios del servicio.***

Para alcanzar este objetivo, se han definido cuatro objetivos intermedios o hitos, los cuales representan las principales contribuciones de esta tesis:

- **Objetivo 1: Análisis y evaluación de DASH, su relación con los contenedores de contenido multimedia y los modos de codificación (bitrate constante / bitrate variable).**

  En esta tesis se estudia el protocolo de streaming DASH así como su interdependencia con los contenedores multimedia. Además, analiza los efectos de los distintos tipos de codificación (bitrate constante / bitrate variable) en el streaming sobre la estructura de los contenedores.

- **Objetivo 2: Análisis, evaluación e implementación de algoritmos de adaptación DASH y evaluación de su eficacia en la reproducción de vídeo codificado con calidad constante.**

  Las paradas en la reproducción debidas al vaciamiento del buffer es el principal, aunque no único, factor para medir el rendimiento de los algoritmos de adaptación. En este sentido, se propone un algoritmo (Look Ahead) que tiene en cuenta el tamaño cambiante de los segmentos de vídeo en su proceso de adaptación con el objetivo de evitar paradas en la reproducción. Para su evaluación, se implementa el algoritmo propuesto, así como varios de los principales algoritmos de la literatura para la plataforma Android.

- **Objetivo 3: Análisis y evaluación de diferentes modelos de calidad de experiencia.**

  Con el fin de evaluar el rendimiento de los diferentes algoritmos de adaptación estudiados, se evalúan varios modelos de calidad de experiencia existentes en la literatura. Adicionalmente, se proponen tres nuevos modelos de fácil utilización que extienden una propuesta de la literatura con el fin de utilizar datos más próximos a las calidades efectivas de los segmentos. Los datos utilizados son mejores representantes de la calidad puesto que hacen referencia a cada uno de los segmentos. En particular, se ha utilizado el bitrate medio, el PSNR y el valor

de VMAF. Todos ellos referidos a cada uno de los segmentos que componen la reproducción.

- **Objetivo 4: Análisis y evaluación de DASH en un entorno Wi-Fi con alta densidad de usuarios.**

  Los clientes de vídeo DASH de redes Wi-Fi con una alta densidad de usuarios se enfrentan a retos adicionales para evitar la interrupción de la reproducción. Esto es debido al patrón típico de descarga de DASH (ON/OFF) que dificulta la estimación de la tasa de transferencia disponible y las variaciones propias de los enlaces Wi-Fi. Con el fin de mejorar la experiencia de los usuarios en esta situación, se propone un mecanismo para la coordinación de los clientes de vídeo que permite reducir, de forma significativa, el número de paradas.

Capítulo 2

Algoritmo de adaptación Look Ahead

2.1. Introducción

2.1.1. Algoritmos de adaptación

Los algoritmos de adaptación ABR (*Adaptive BitRate*) son la lógica que se aplica en los reproductores de vídeo a la hora de decidir la representación del siguiente segmento del contenido multimedia en la línea de tiempo del que no se dispone ya alguna representación en el *buffer* de reproducción.

Dicho así cabría esperar que, en el caso de reproducir un contenido con una única representación, todos los algoritmos se comportasen de la misma forma. Evidentemente, la representación seleccionada sí será la misma. Sin embargo, hay varios elementos adicionales que interactúan con la obtención de los segmentos. Estos elementos son:

- **Gestión del buffer**: Tiene como principales tareas decidir cuándo puede comenzar la reproducción de un contenido, cuándo debe detenerse y volver a comenzar en caso de parada, y cuándo se debe iniciar la descarga de un segmento.
- **Estimación de la tasa de transferencia**: Tiene como único objetivo proporcionar una estimación de la tasa de transferencia de la que se va a disponer en el futuro inmediato. La fuente principal de información en la que se basan las

implementaciones de este elemento es la tasa de transferencia medida en la descarga de los segmentos anteriores.

- **Información del contexto**: Adicionalmente, los algoritmos de adaptación pueden hacer uso de la información de contexto del dispositivo en el que se va a reproducir el contenido para adaptar el algoritmo. Por ejemplo, la resolución de la pantalla y el tipo de dispositivo puede tenerse en cuenta a la hora de discriminar alguna representación, así como la tecnología de conexión (Wi-Fi, 4G, 5G, etc.) puede ser relevante a la hora de estimar la tasa de transferencia disponible o de seleccionar una representación inicial.
- **Selección de la representación**: Selecciona la representación del siguiente segmento que se debe obtener basándose en la información proporcionada por el resto de los módulos del ABR.

En el caso planteado donde únicamente existe una representación del contenido, la gestión del buffer puede jugar un papel diferenciador entre algoritmos. Por ejemplo, dos implementaciones diferentes de la gestión del buffer pueden tomar decisiones diferentes sobre en qué momento deben volver a descargar segmentos partiendo de un estado donde ambas tenían el buffer al máximo. Una de ellas podría decidir descargar el siguiente segmento nada más disponga de espacio libre en el buffer y la otra cuando el tamaño del buffer baje del 50%. Esta pequeña diferencia podría tener consecuencias tanto en la calidad de experiencia, al reaccionar mejor a los cambios en la tasa de transferencia disponible alguno de los modos, como en el consumo de batería, por la diferencia entre los intervalos de utilización de las comunicaciones.

A pesar de la clara diferenciación de tareas, es frecuente encontrar en la literatura la descripción de los algoritmos ABR como entidades únicas que gestionan todo lo referente a la descarga de los segmentos, pero, a su vez, es poco común encontrar trabajos donde se separen claramente las implicaciones sobre cada uno de los elementos descritos que intervienen en la adaptación. Los reproductores de vídeo comerciales, como veremos más adelante, suelen hacer la división de la funcionalidad de los ABRs en módulos, tal como se ha mencionado anteriormente, ya que esta división de tareas facilita la escritura y mantenimiento del código de los reproductores de vídeo, además de proporcionar versatilidad a las implementaciones de los ABRs.

2.1.2. Variación del tamaño de los segmentos

En la mayoría de los trabajos donde se proponen nuevos algoritmos de adaptación no se tiene en cuenta la variación del tamaño de los segmentos [18]-[20]. En estos casos, se definen los vídeos utilizados como de tasa constante según la utilizada en la codificación de los contenidos.

Si bien es muy poco probable que dos segmentos tengan exactamente el mismo tamaño, éste puede ser razonablemente parecido al utilizar parámetros de codificación de tasa constante. Sin embargo, utilizar este tipo de codificación, como se introdujo en el capítulo anterior, no parece tener como objetivo otro que la eliminación de la variabilidad del

tamaño de los segmentos a costa de la introducción de una mayor variabilidad en la calidad de cada representación.

En este capítulo se propone la utilización de los codecs de vídeo en modo de calidad constante como modalidad natural de codificación de flujos de vídeo para su distribución bajo demanda utilizando DASH.

En este caso, esta tesis defiende que predecir el tamaño del siguiente segmento en función del bitrate medio del vídeo supone una falta de precisión que, en determinadas circunstancias, puede llevar a producir paradas en la reproducción debidas al vaciamiento del buffer.

De la misma forma, se propone un nuevo algoritmo que hace uso de la información de los tamaños de los segmentos presente en los contenedores multimedia con el fin de evitar paradas en la reproducción debidas a esta variabilidad.

2.2. Estado del arte

Debido a que la especificación de DASH, como se ha visto en el Capítulo 1 se limita a la definición del archivo de metadatos que define al contenido, la MPD, y el formato de la transferencia del propio contenido, los contenedores multimedia, el proceso de selección de la representación queda exento de cualquier limitación. Esta situación ha favorecido la proliferación de trabajos que, como este, utilizan esta libertad para intentar mejorar la experiencia de los usuarios a la hora de reproducir un vídeo.

En general, se puede considerar que un algoritmo de adaptación tiene como objetivos:

1. **Evitar que la reproducción se detenga** a causa del vaciamiento del buffer de reproducción.
2. En convivencia con el punto anterior, **maximizar la calidad del contenido reproducido**.
3. En convivencia con los puntos anteriores, **minimizar el número de cambios de calidad**.
4. **Minimizar el retardo inicial** en lo posible en convivencia con todos los puntos anteriores.

Obtener los mejores valores para estos objetivos no es siempre viable debido a las limitaciones y variabilidad de las redes de comunicación. De esta forma, los algoritmos ABR se deben enfrentar al reto de ofrecer la mejor experiencia a los usuarios en función de las condiciones y características del streaming.

A la hora de clasificar los ABR, en [21] se propone una clasificación según la información que se utiliza para la toma de decisiones en:

- **Basados en la estimación de la tasa de transferencia**: Utilizan principalmente la estimación de la tasa de transferencia para tomar la decisión de adaptación.

- **Basados en el buffer**: La información del estado del buffer es la principal a la hora de tomar las decisiones de adaptaciones.
- **Mixtos**: Ambas informaciones son utilizadas en la adaptación.

Si bien es posible desarrollar un ABR que únicamente tenga en cuenta uno de los dos datos, en la práctica, casi todos los algoritmos son mixtos. Por ejemplo, el algoritmo BOLA [19] se diseñó de forma que únicamente tenía en consideración el estado del buffer para la adaptación y su evaluación demostraba que funciona correctamente cuando la reproducción de un contenido se encuentra en un estado estacionario. No obstante, al iniciar la reproducción, hacer saltos en la línea de tiempo o al recuperarse la tasa de transmisión después de una bajada abrupta, el algoritmo BOLA sufre de un proceso de recuperación lento debido a la bajada en el estado del buffer que implican estos supuestos. Es por esto por lo que a continuación de la propuesta de BOLA los autores hicieron la propuesta de mejora del algoritmo pasando a denominarse BOLA-E [20]. En esta ocasión, el algoritmo evolucionado sí hace uso de la tasa de transferencia disponible para mejorar el comportamiento del algoritmo en las situaciones mencionadas.

A continuación, se detalla la librería de reproducción utilizada en este trabajo de investigación y los algoritmos de adaptación utilizados, así como otros destacados de la literatura. Siendo esta una tesis con un alto grado de experimentalidad y pruebas, es fundamental presentar el funcionamiento interno del software utilizado con un mínimo de detalle con el fin de entender las implicaciones de las implementaciones de los algoritmos utilizados.

2.2.1. ExoPlayer

Partiendo de las dos principales plataformas móviles iOS y Android, nos encontramos con que iOS impone ciertas limitaciones al streaming en las aplicaciones que se distribuyen a través de la App Store [32]. Además, la plataforma está profundamente integrada con HLS, por lo que los pocos desarrollos en esta línea estás destinados a traducir otros estándares como DASH a HLS [33]. El caso de Android es diametralmente opuesto. La segmentación de la plataforma ha hecho posicionarse a su principal promotor, Google, en un sistema de desarrollo de librerías externas a la plataforma que, en la medida de lo posible, ocultan la complejidad de la fragmentación y permite la modificación y aportación por parte de los usuarios (los desarrolladores de aplicaciones).

Es justamente este último escenario donde se enmarca ExoPlayer [34]. ExoPlayer es una librería para la reproducción de contenido multimedia para la plataforma Android que soporta los principales sistemas HAS: HLS, *Smooth Streaming* y DASH. El objetivo de la librería es ofrecer a los desarrolladores una estructura modular de todos los componentes necesarios para la reproducción. El énfasis en la modularidad hace que sea posible sustituir cada componente por una implementación aportada por el usuario. Al tratarse de una librería de código libre en la que el propio Google ha puesto un gran esfuerzo, ExoPlayer ha alcanzado gran popularidad entre los proyectos desarrollados para la plataforma Android. En 2018, según datos de Google [35] , habían más de 200.000 aplicaciones en la Google Play Store que incluían la librería. Entre estas aplicaciones se

encuentran las aplicaciones de las principales plataformas de streaming como YouTube, Netflix, HBO, Amazon Prime Video, etc.

De entre toda la adaptabilidad que proporciona ExoPlayer, este trabajo de investigación se centra en la parte encargada de la selección de la representación de la cual el player obtendrá el siguiente segmento. Dicha parte, que desde un punto de vista externo denominamos algoritmo de adaptación, está compuesta, en ExoPlayer, por tres módulos bien diferenciados: estimación de la tasa de transferencia disponible, gestión del buffer y selección de representación, que se introducen a continuación.

2.2.1.1. Estimación de la Tasa de Transferencia

La estimación de la tasa de transferencia es el módulo que forma parte del algoritmo de adaptación con la jerarquía más sencilla.

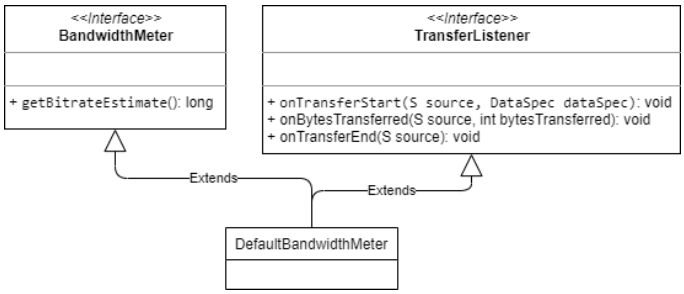


Figura 2.1. Jerarquía de *DefaultBandwidthMeter* (git commit 7d3f54a37).

En la Figura 2.1 se muestra como DefaultBandwidthMeter, que es la implementación por defecto de la estimación de la tasa de transferencia disponible de ExoPlayer, implementa dos interfaces: *BandwidthMeter* y *TransferListener*. Por un lado, *TransferListener* le permite a *DefaultBandwidthMeter* recibir información de los datos que se están recibiendo durante la transferencia de los segmentos a través de la implementación de *onBytesTransferred(),* tal y como se puede observar en la Figura 2.2.

```
1.  /**
2.   * Called incrementally during a transfer.
3.   *
4.   * @param source The source performing the transfer.
5.   * @param bytesTransferred The number of bytes transferred since the previous
     call to this
6.   *     method (or if the first call, since the transfer was started).
7.   */
8.  void onBytesTransferred(S source, int bytesTransferred);
```

Figura 2.2. Firma del método *onBytesTransferred()* definido en la interfaz *TransferListener* (git commit 8331defc7).

Por el otro, la interfaz *BandwidthMeter* requiere de la implementación del método *getBitrateEstimate()* cuya firma se muestra en la Figura 2.3.

```
1.  /**
2.   * Returns the estimated bandwidth in bits/sec, or {@link #NO_ESTIMATE} if an
     estimate is not
3.   * available.
4.   */
5.  long getBitrateEstimate();
```

Figura 2.3. Firma del método *getBitrateEstimate()* definido en la interfaz *BandwidthMeter* (git commit 8331defc7).

Es, pues, misión del estimador de la tasa de transferencia transformar la información recibida a través de los métodos definidos en *TransferListener*, según se estime oportuno en cada caso para así, cuando sea requerido, poner a disposición el resultado a través de la interfaz *BandwidthMeter*.

En lo que se refiere a la implementación de *DefaultBandwidthMeter* (git commit 8331defc7), a partir de la interpretación del código fuente se deduce que la tasa de transferencia estimada se obtiene como el percentil 0,5 de una ventana deslizante ordenada por pesos, donde el peso de un valor es la raíz cuadrada del propio valor y un peso máximo acumulado en la ventana, por defecto, de 2000. Los valores introducidos en la ventana son las mediciones proporcionadas a través de la interfaz *BandwidthMeter* convertidos a bits por segundo.

2.2.1.2. Gestión del Buffer

La gestión del buffer queda fuera de la especificación de DASH, por lo que cada implementación es libre de ejecutarla como estime oportuno. En ExoPlayer esta gestión está gobernada por la interfaz *LoadControl* y tiene como implementación por defecto a *DefaultLoadControl*, tal y como se muestra en la Figura 2.4.

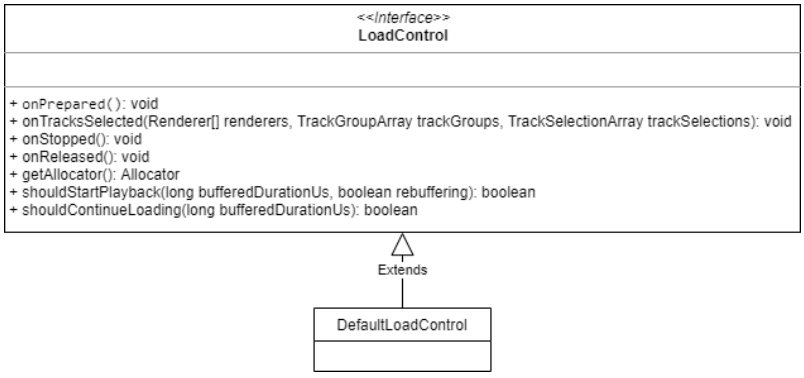


Figura 2.4. Jerarquía de *DefaultLoadControl* (git commit 7d3f54a37).

De entre los métodos que define la interfaz *LoadControl* tienen especial importancia los dos últimos que aparecen en la Figura 2.4 y de los que se muestra su firma y documentación en el extracto de la Figura 2.5.

```
1.  /**
2.   * Called by the player to determine whether sufficient media is buffered for
     playback to be
3.   * started or resumed.
4.   *
5.   * @param bufferedDurationUs The duration of media that's currently buffered.

6.   * @param rebuffering Whether the player is rebuffering. A rebuffer is define
    d to be caused by
7.   *     buffer depletion rather than a user action. Hence this parameter is fa
    lse during initial
8.   *     buffering and when buffering as a result of a seek operation.
9.   * @return Whether playback should be allowed to start or resume.
10.  */
11. boolean shouldStartPlayback(long bufferedDurationUs, boolean rebuffering);
12.
13. /**
14.  * Called by the player to determine whether it should continue to load the s
    ource.
15.  *
16.  * @param bufferedDurationUs The duration of media that's currently buffered.

17.  * @return Whether the loading should continue.
18.  */
19. boolean shouldContinueLoading(long bufferedDurationUs);
```

Figura 2.5. Extracto del archivo *LoadControl.java* del proyecto ExoPlayer (git commit 8331defc7).

Por un lado, *shouldStartPlayback()* es el encargado de indicar si el player debe iniciar la reproducción tanto al inicio de la reproducción como en el caso de que hubiera habido alguna parada debido a que el buffer se hubiese vaciado (indicado por el parámetro booleano *rebuffering*). Por el otro, *shouldContinueLoading()* debe indicar si se debe iniciar/continuar la descarga de segmentos.

La implementación por defecto de esta interfaz es *DefaultLoadControl* y su funcionamiento se describe a continuación:

DefaultLoadControl puede estar en dos estados: *buffering* y *idle* y cambia de estados según las siguientes reglas:

- En estado *buffering*
    - Pasa a *idle* en el momento en el que se alcancen 30 s de buffer.
- En estado *idle*

- Pasa a *buffering* en el momento que el buffer descienda de 15 s.

En cuanto al inicio de la reproducción, *DefaultLoadControl* marca dos condiciones separadas:

- Se iniciará la reproducción inicial al alcanzar 2,5 s de buffer.
- Se iniciará la reproducción después de una parada por vaciado de buffer al alcanzar 5 s.

No obstante, hay que tener en cuenta que el hecho de pasar de estado *buffering* a *idle* no detiene una descarga en curso. La descarga en cuestión continuará y al finalizar, si sigue en estado *idle*, no se descargarán más segmentos.

2.2.1.3. Selección de la representación

La selección de la representación es el tercer módulo que compone las implementaciones de los algoritmos de adaptación en ExoPlayer. El funcionamiento principal está regido por la interfaz *TrackSelection* cuya jerarquía se muestra en la Figura 2.6.

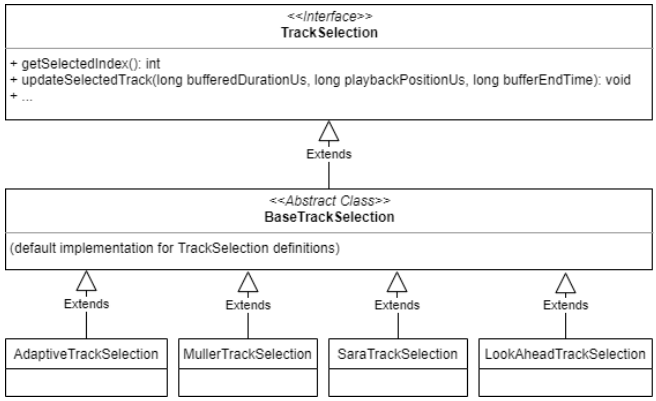


Figura 2.6. Jerarquía de clases de *TrackSelection* (git commit 7d3f54a37).

En la Figura 2.6 se puede observar que todas las implementaciones heredan de la clase abstracta *BaseTrackSelection* que proporciona una implementación común a algunos de los métodos definidos en la interfaz *TrackSelection*.

La interfaz *TrackSelection* es una de las más complejas debido al gran número de métodos que declara y por tener, quizá, más de una función claramente definida ya que *TrackSelection*, a través de sus implementaciones, también tiene la capacidad de añadir representaciones a una lista negra, de modo que dichas representaciones dejen de ser elegibles durante un cierto tiempo. Abriendo un paréntesis sobre la capacidad de añadir representaciones a una lista negra, se podría considerar que viola la regla de función única y, por lo tanto, puede ser un punto de mejora en futuras versiones de la librería. En cualquier caso, un ejemplo de la utilidad que proporciona sería la utilización de la información de contexto para la selección de representaciones apropiadas. Por ejemplo, al crear la lista negra, las implementaciones de los algoritmos de adaptación podrían

agregar las representaciones de vídeo con resoluciones superiores a las de la pantalla del dispositivo o representaciones de audios con un número de canales mayor al disponible. Teniendo en cuenta, por supuesto, que la pantalla y la salida de audio pueden cambiar durante la reproducción de vídeo como, por ejemplo, al conectar una pantalla externa o al conectar una nueva salida de audio.

Siguiendo con la implementación de la lógica de selección de representaciones, los métodos más significativos para este trabajo de investigación son los que se muestran en la Figura 2.7.

```
1.  /**
2.   * Returns the index of the selected track.
3.   */
4.  int getSelectedIndex();
5.
6.  /**
7.   * Updates the selected track.
8.   *  @param bufferedDurationUs The duration of media currently buffered in mic
    roseconds.
9.   * @param playbackPositionUs
10.  * @param bufferEndTime
11.  */
12. void updateSelectedTrack(long bufferedDurationUs, long playbackPositionUs, lo
    ng bufferEndTime);
```

Figura 2.7. Extracto del archivo *TrackSelection.java* del proyecto ExoPlayer (git commit 8331defc7).

ExoPlayer, en el momento en que vaya a descargar el siguiente segmento de un determinado flujo multimedia, invocará a la implementación de *getSelectedIndex()* del algoritmo que corresponda. Para que la información que pueda devolver este método esté actualizada, ExoPlayer, con anterioridad, habrá invocado *updateSelectedTrack()* que es donde realmente se tomarán las decisiones puesto que es quien recibe como parámetros la posición de la reproducción y estado el buffer, entre otros.

Tal como está definido en ExoPlayer, la información de la que cuentan las implementaciones de *TrackSelection* acerca de las diferentes representaciones es la que se le ofrece a través de la clase *Format*. Cada una de las representaciones que compongan el flujo multimedia está caracterizada por una instancia de esta clase que contiene, entre otras, información acerca de las resoluciones, codecs, bitrates, datos de inicialización de los codecs, etc.

2.2.2. Algoritmos de adaptación de vídeo

A continuación, se pasa a describir los algoritmos de adaptación que han sido implementados en la librería ExoPlayer para ser evaluados en esta tesis: Müller, SARA y ExoPlayer. Asimismo, se describen brevemente otros algoritmos existentes en la literatura.

2.2.2.1. Algoritmo Müller

En [18] se describe el algoritmo de adaptación Müller que es utilizado en software de código libre como VLC. El algoritmo utiliza la estimación de la tasa de transferencia y el estado de buffer, por tanto es un algoritmo mixto, para calcular una nueva estimación de la tasa de transferencia disponible. Cuanto menor sea el estado del buffer, menor será la nueva estimación de la tasa de transferencia generada. La generación de la nueva estimación se rige por la siguiente fórmula:

$$max\_bw(s_i) = \begin{cases} bw(s_{i-1}) * 0.3 & if \quad 0 \leq bl_i < 0.15 \\ bw(s_{i-1}) * 0.5 & if \quad 0.15 \leq bl_i < 0.35 \\ bw(s_{i-1}) & if \quad 0.35 \leq bl_i < 0.50 \\ bw(s_{i-1}) * (1 + 0.5 * bl_i) & if \quad 0.50 \leq bl_i \leq 1 \end{cases} \quad (2.1)$$

Donde $i \in [1, n]$ es el índice del segmento, n es el número de segmentos que componen el vídeo, bl_i es el nivel del buffer al proceder a descargar el segmento i, y $bw(s_i)$ es la función que obtiene la estimación de la tasa de transferencia en el momento en que se va a proceder a la descarga de i.

Una vez generada la estimación de la tasa de transferencia disponible, el algoritmo Müller seleccionará la representación cuyo bitrate medio sea el mayor que no sobrepase dicha estimación. De esta forma, el estado del buffer va condicionando las representaciones que serán seleccionadas por el algoritmo.

2.2.2.2. Algoritmo SARA

El algoritmo SARA (*Segment-Aware Rate Adaptation*), propuesto en [22], es un algoritmo de adaptación mixto que sí que tiene en consideración la variación de los tamaños de los segmentos dentro de una misma representación. El algoritmo, utiliza esta información adicionalmente a la estimación de la tasa de transferencia disponible y al tamaño del buffer para la selección de la representación del siguiente segmento.

El algoritmo se basa en la utilización de diferentes estrategias para la selección de la representación del siguiente segmento en función del buffer. Las posibles opciones son: inicio rápido, incremento adicional, cambio agresivo y descarga pospuesta.

Las estrategias se seleccionan en función del estado del buffer de reproducción. Para ello el algoritmo define tres niveles de compleción (I, $B\alpha$, $B\beta$) como se puede ver en la Figura 2.8.

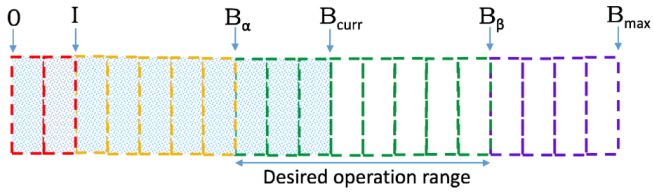


Figura 2.8. Niveles de buffer del algoritmo SARA. Fuente: *SARA: Segment-Aware Rate Adaptation Algorithm for Dynamic Adaptive Streaming over HTTP* [22].

El comportamiento de los diferentes estados del algoritmo es:

- **Inicio rápido**: Cuando el estado del buffer es menor de *I* se selecciona la representación con menor bitrate medio.
- **Incremento adicional**: Una vez que el buffer de ocupación supera el nivel de *I,* y si así lo permite la estimación del tiempo de descarga del siguiente segmento, se selecciona la siguiente representación con mejor calidad.
- **Cambio agresivo**: Cuando el estado del buffer se sitúa entre $B\alpha$ y B_β el algoritmo selecciona la representación que mejor se adapta al estado estimado de la red.
- **Descarga pospuesta**: Cuando la ocupación del buffer supera el nivel de B_β se pospone la petición de la siguiente representación seleccionada hasta que el nivel del buffer vuelva a caer por debajo de B_β.

Adicionalmente, el algoritmo define la forma en la que se debe estimar la tasa de transferencia disponible. Para ello, establece una media armónica ponderada donde la estimación de la tasa de transferencia disponible para el segmento *n* queda definida según la siguiente fórmula:

$$H_n = \frac{\sum_{i=1}^{n} \omega_i}{\sum_{i=1}^{n} \frac{\omega_i}{d_i}}, \tag{2.2}$$

donde ω_i es el peso proporcional al tamaño del segmento *i*, y d_i es la tasa de descarga del segmento *i*. El tiempo requerido para la descarga del siguiente segmento es predicho por ω_{n+1}/H_n, el cual, es comparado con diferentes umbrales del buffer para decidir la representación del siguiente segmento. Como se muestra en la sección 2.5, este método de estimación de la tasa de transferencia disponible reacciona lentamente ante los cambios en la propia tasa, lo que puede provocar paradas en la reproducción.

Esta propuesta utiliza la MPD para alojar la información de todos los tamaños de los segmentos de todas las representaciones tal y como se muestra en la Figura 2.9.

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD xmlns="urn:mpeg:dash:schema:mpd:2011" minBufferTime="PT1.500000S" type="static" mediaPresentationDuration="PT0H9M56.46S" profiles="urn:mpeg:dash:profile:isoff-live:2011">
  <Period duration="PT0H9M56.46S">
    <AdaptationSet mimeType="video/mp4" segmentAlignment="true" group="1" maxWidth="480" maxHeight="360" maxFrameRate="24" par="4:3">
      <Representation id="320x240 45.0kbps" mimeType="video/mp4" codecs="avc1.42c00d" width="320" height="240" frameRate="24" sar="1:1" startWithSAP="1" bandwidth="45226" >
        <SegmentTemplate timescale="96" media="media/BigBuckBunny/4sec/bunny_$Bandwidth$bps/BigBuckBunny_4s$Number$%d.m4s" startNumber="1" duration="384" initialization="media/BigBuckBunny/4sec/bunny_$Bandwidth$bps/BigBuckBunny_4s_init.mp4" />
        <SegmentSize id="BigBuckBunny_4s1.m4s" size="168.0" scale="ºits"/>
        <SegmentSize id="BigBuckBunny_4s2.m4s" size="184.0" scale="Kbits"/>
        <SegmentSize id="BigBuckBunny_4s3.m4s" size="200.0" scale="Kbits"/>
        …
        <SegmentSize id="BigBuckBunny_4s150.m4s" size="32.0" scale="Kbits"/>
      </Representation>
      <Representation id="320x240 89.0kbps" mimeType="video/mp4" codecs="avc1.42c00d" width="320" height="240" frameRate="24" sar="1:1" startWithSAP="1" bandwidth="88783" >
        <SegmentTemplate timescale="96" media="media/BigBuckBunny/4sec/bunny_$Bandwidth$bps/BigBuckBunny_4s$Number$%d.m4s" startNumber="1" duration="384" initialization="media/BigBuckBunny/4sec/bunny_$Bandwidth$bps/BigBuckBunny_4s_init.mp4" />
        <SegmentSize id="BigBuckBunny_4s1.m4s" size="336.0" scale="Kbits"/>
        <SegmentSize id="BigBuckBunny_4s2.m4s" size="360.0" scale="Kbits"/>
        <SegmentSize id="BigBuckBunny_4s3.m4s" size="400.0" scale="Kbits"/>
        …
        <SegmentSize id="BigBuckBunny_4s150.m4s" size="60.8" scale="Kbits"/>
      </Representation>
    …
    </AdaptationSet>
  </Period>
</MPD>
```

Figura 2.9. Extracto de MPD adaptada para la utilización del algoritmo SARA.

Adicionalmente, las modificaciones propuestas sobre la MPD suponen una desviación del estándar DASH.

2.2.2.3. Algoritmo ExoPlayer

La librería de reproducción de contenido multimedia ExoPlayer dispone de una implementación por defecto de un ABR al que denominamos, también, ExoPlayer. Este algoritmo es el utilizado por la librería en caso de que el usuario no proporcione otra implementación.

El algoritmo ExoPlayer, en el momento de tomar la decisión de qué representación hay que obtener el siguiente segmento, selecciona la representación que mejor se ajusta a la tasa de transferencia disponible. La selección se hace comprobando todas las representaciones y seleccionando la representación con el mayor bitrate medio que sea

menor o igual a la estimación de la tasa de transferencia ($\widehat{bw}$), que está ponderada por un factor λ, como se muestra en (2.3).

$$q[i+1] = \max\{Q: bw_{q_j}[i+1] \leq \lambda \cdot \widehat{bw}\}, \tag{2.3}$$

donde $Q=\{q_0, q_1, \ldots, q_{k-1}\}$ es el vector con las k calidades disponibles. Una vez seleccionada la representación que mejor se ajusta, si ésta difiere de la última representación seleccionada, el algoritmo revertirá la nueva representación seleccionada por la anterior en dos casos que se describen a continuación y rigen según la fórmula (2.4):

- Si la nueva representación seleccionada ($bw[i+1]$) tiene un bitrate medio mayor que la anterior ($bw[i]$) y el tamaño del buffer (b) es menor que (β_{min}) (por defecto 10 s).
- Si la nueva representación seleccionada tiene un bitrate medio menor que la anterior y el tamaño de buffer es mayor que (β_{max}) (por defecto 25 s).

$$if\ (bw[i+1] > bw[i]\ and\ b < \beta_{min})\ \text{or}\ (bw[i+1] < bw[i]\ and\ b > \beta_{max})\ then\ q[i+1] = q[i]. \tag{2.4}$$

La estimación de la tasa de transferencia disponible y la gestión del buffer están gestionadas por las implementaciones por defecto de la propia librería.

2.2.2.4. Otros algoritmos de adaptación

En la literatura, y debido a que la lógica de los algoritmos de adaptación queda fuera de la especificación de DASH, existe un gran número de trabajos que versan sobre la mejor forma de realizar la selección de la representación a descargar.

Entre los distintos algoritmos que aparecen en la literatura cabe destacar BOLA [19], como se ha comentado anteriormente. BOLA es un algoritmo basado en el estado del buffer que utiliza esta información para mapear el bitrate del siguiente segmento. Para ello utiliza dos funciones de rendimiento: 1) la calidad de la reproducción basada en el bitrate medio de los segmentos reproducidos; y 2) el porcentaje de tiempo que la reproducción está en estado de reproducción (no está parada debido a un vaciamiento del buffer), denominado por los propios autores como "suavidad de reproducción". El algoritmo utiliza la optimización de Lyapunov [22] para maximizar las dos funciones anteriores en base a un parámetro que define el peso de la suavidad de la reproducción frente a la calidad de los contenidos. Para ajustar el comportamiento del algoritmo al modelo elegido, BOLA considera una serie de slots de tiempo predefinidos en los que se pueden tomar las decisiones y que los segmentos de vídeo únicamente pueden empezar a ser reproducidos en el momento en que estén totalmente descargados. Este algoritmo tiene la relevancia de ser el algoritmo por defecto del player de referencia de DASH-IF (*DASH Industry Forum*): Dash.js [23].

A pesar del interés que hubiera tenido la utilización de este algoritmo para las diferentes evaluaciones llevadas a cabo en esta tesis existen diferentes causas que han impedido su

utilización. Por ejemplo, el lenguaje de programación y las características del player de vídeo sobre el que se basan harían difícil su desarrollo para ExoPlayer, que es la librería de reproducción DASH empleada en este trabajo de investigación. Como ejemplo, BOLA, durante su ejecución, tiene la posibilidad de abandonar la descarga en curso de un segmento si el algoritmo decide que hay riesgo de sufrir una parada en pro de una representación que sí que pueda descargarse en el tiempo disponible. Esta capacidad, que sí que está soportada por Dash.js, no se encuentra disponible en ExoPlayer por lo que su implementación no podría ser completa.

En [20] se propone una evolución del algoritmo BOLA. El objetivo de BOLA-E (*BOLA Enhanced*), que es como se denomina esta propuesta, es paliar algunos puntos donde BOLA sufría un rendimiento pobre. En particular, BOLA-E pasa a tener en cuenta la estimación de tasa de transferencia disponible para adaptarse más rápidamente a las bajadas drásticas de tasa de transferencia disponible, mejora el proceso de inicio y salto en la línea temporal de forma que estos procesos se gestionan de forma diferente para acelerar la estabilización y propone la funcionalidad, denominada *fast-switching*, de descartar segmentos ya descargados si estima que pueden ser sustituidos por otros de mayor calidad.

En el mismo trabajo que el BOLA-E, en [20] también se introduce DYNAMIC, que es un algoritmo híbrido que usa una aproximación basada en la tasa de transferencia disponible cuando el nivel del buffer es bajo y pasa a BOLA-E cuando el nivel del buffer es alto. La motivación de este algoritmo es que el algoritmo BOLA-E se ve bastante perjudicado por el proceso de buffering, puesto que un estado con el buffer en cotas bajas le obliga a elegir calidades bajas, aunque la tasa de transferencia disponible sea suficiente para mantener mejores calidades.

También se pueden encontrar en la literatura algoritmos de adaptación como PANDA [25] o ELASTIC [26], cuyo objetivo es mitigar los efectos de la competencia de múltiples clientes DASH en una misma red o compartiendo un cuello de botella común.

Otro trabajo interesante es [27], donde los autores proponen un algoritmo con dos estados: *Low Start* y *Steady State*, así como una función de la calidad de la experiencia basada en el bitrate de cada segmento (esta propuesta modifica la MPD para incluir el tamaño de los segmentos). También propone un interesante mecanismo para la estimación de la tasa de transferencia disponible basado en el análisis continuo de la llegada de los paquetes de la conexión de TCP/IP.

A modo de referencia, cabría mencionar algoritmos como TCP-Like AIMD, propuesto por Liu et a. [28] y FESTINE [29] como ejemplos de algoritmos basados en la tasa de transferencia, así como el mencionado BOLA [19] y *Buffer Based Rate Selection* [30] como algoritmos basados en el estado del buffer. Puede encontrarse una descripción más detallada que las que aquí se proporcionan, así como un mayor detalle en las mismas en el trabajo de Bentaleb et al. [31].

2.3. Algoritmo Look Ahead

El algoritmo de adaptación que se propone en esta tesis, Look Ahead, nace de la idea básica de aprovechar la información presente en los propios contenedores multimedia para extraer el tamaño de los segmentos que conforman el contenido. Para ello parte con dos limitaciones: 1) únicamente puede aplicarse para contenido bajo demanda puesto que, evidentemente, en el contenido en vivo no se dispone de la información del tamaño de los futuros segmentos en el contenedor; y 2) el contenido debe presentarse en forma de un único archivo segmentado.

El segundo requisito se debe a la necesidad de la presencia de algún tipo de estructura del contenedor que disponga de la información de la posición relativa de los segmentos respecto al contenedor. En el apartado 1.3.2.3 se muestran estas estructuras al final de los archivos puesto que es en esa posición donde se suelen escribir dichas estructuras al crearse el contenido, ya que la información que contienen se va generando durante el proceso de codificación, por lo que hasta que no se codifica el último segmento no se puede escribir. Sin embargo, y a pesar de que esta posición es perfectamente válida, el proceso de preparación suele incluir la traslación de las estructuras de índices a las primeras posiciones. La razón es, simplemente, una optimización del proceso de inicialización que se detalla a continuación.

```
1.  <Representation id="1" bandwidth="254977" codecs="avc1.42c01f" mimeType="vide
    o/mp4" sar="265:267" width="256" height="106">
2.    <BaseURL>video-2.mp4</BaseURL>
3.    <SegmentBase indexRange="1242-2005" timescale="12800">
4.      <Initialization range="0-1241"/>
5.    </SegmentBase>
6.  </Representation>
7.  <Representation id="2" bandwidth="454957" codecs="avc1.42c01f" mimeType="vide
    o/mp4" sar="265:267" width="512" height="212">
8.    <BaseURL>video-1.mp4</BaseURL>
9.    <SegmentBase indexRange="1246-2009" timescale="12800">
10.     <Initialization range="0-1245"/>
11.   </SegmentBase>
12. </Representation>
```

Figura 2.10. Extracto de una MPD donde se observan los rangos de inicialización e índices de segmentos.

En la Figura 2.10 se muestra un extracto de MPD donde se muestran dos representaciones del mismo contenido. En ambas están resaltados tanto el rango de inicialización como el rango de índices. Estos datos corresponden a los bytes, dentro de sus respectivos archivos, donde se encuentran las estructuras necesarias para el acceso a los segmentos de vídeo.

El rango de inicialización contiene información de las características del vídeo incluyendo los datos de inicialización del códec. Por su parte, la estructura a la que apunta

el rango de índices contiene la información de los rangos de bytes de los archivos de vídeo que se deben solicitar para acceder de forma independiente a cada segmento.

Siguiendo con la optimización del proceso de inicialización, podemos observar como ambos rangos, para cada representación, son consecutivos. De esta forma, y si el cliente de vídeo así lo soporta, se puede inicializar una representación con una única petición HTTP en vez de con dos.

Teniendo claro que la información relativa al tamaño de los segmentos está intrínsecamente relacionada con los rangos de índices de los contenidos, es evidente que los reproductores de vídeo disponen de dicha información. Sin embargo, no se ha encontrado ningún reproductor de código abierto que ponga esta información a disposición de las distintas partes del código de forma que sea posible su utilización, por ejemplo, en los algoritmos de adaptación.

2.3.1. Propuesta

Como se ha comentado con anterioridad, el algoritmo de adaptación Look Ahead nace de la idea de aprovechar la información existente en los contenedores multimedia para conocer de antemano el tamaño de los segmentos de todas las representaciones y así poder utilizar esta información para la selección de la representación del siguiente segmento.

De esta forma, el algoritmo Look Ahead tiene como objetivo evitar las paradas durante la reproducción de contenidos, especialmente, cuando esas paradas están relacionadas con la variabilidad del tamaño de los segmentos.

Al calcular cuál debe ser la representación seleccionada para el siguiente segmento $i+1$, Look Ahead intenta proporcionar la máxima calidad, sin incurrir en paradas en la reproducción. Es decir, de entre las k representaciones posibles $Q=\{q_0, q_1, \ldots, q_{k-1}\}$, donde q_j es la representación del segmento j (teniendo en cuenta que $q_j < q_{j+1}$), el algoritmo selecciona la mayor q_j que permita una reproducción sin paradas.

Look Ahead propone un proceso iterativo que consiste en calcular el bitrate medio de los siguientes z segmentos para todas las representaciones disponibles desde z=1 a θ, donde θ es el número máximo de futuros segmentos a tener en cuenta a la hora de calcular la media de una representación. En cada iteración, el algoritmo selecciona la representación cuyo bitrate medio de los siguientes z segmentos, τ_z, sea lo mayor posible sin superar la tasa de transferencia estimada, como se muestra en (2.5).

$$\tau_z(i+1,j) = \frac{\sum_{m=i+1}^{i+z} S_{m,q_j}}{\sum_{m=i+1}^{i+z} t_m},\ \tau(i+1,j) < \widehat{bw}, \tag{2.5}$$

donde i es el segmento actual, S_{m,q_j} es el tamaño del segmento m para la representación q_j, t_m es la duración del segmento m, y $\widehat{bw}$ es la estimación de la tasa de transferencia disponible. En los casos en los que la codificación y segmentación se hayan hecho utilizando un tiempo de segmento fijo, t_m será el mismo para todos los segmentos.

La representación seleccionada para el siguiente segmento, *ξ(i+1)*, será la menor de las obtenidas en las *θ* iteraciones tal como se muestra en (2.6):

$$\xi(i+1) = min\,\{\tau_z\},\ z = 1 \dots \Theta \tag{2.6}$$

Considerar diferentes iteraciones teniendo en cuenta un número diferente de segmentos futuros es una medida conservadora que pretende hacer al algoritmo más cauto. Así, se anticipa la llegada de futuros segmentos con un tamaño superior a la media, controlando la representación que será seleccionada inmediatamente para así someter a menor presión al buffer. De esta forma, el parámetro *θ* puede tener un impacto significativo en la QoE de los usuarios.

A modo de ejemplo, en el caso particular de *θ=3*, para calcular la representación del segmento *u*, se deben calcular los bitrates medios de cada representación. Esto es, calcular *τ(u,j)*, donde *j* ϵ [*0*, *k-1*], y generar los vectores *T(u)* como se muestra en (2.7), (2.8) y (2.9).

$$\mathrm{T}(u)_{z=1} = \left[\frac{S_{u,0}}{t_u} \quad \frac{S_{u,1}}{t_u} \quad \dots \quad \frac{S_{u,k-1}}{t_u}\right]. \tag{2.7}$$

$$\mathrm{T}(u)_{z=2} = \left[\frac{S_{u,0}+S_{u+1,0}}{t_u+t_{u+1}} \quad \frac{S_{u,1}+S_{u+1,1}}{t_u+t_{u+1}} \quad \dots \quad \frac{S_{u,k-1}+S_{u+1,k-1}}{t_u+t_{u+1}}\right]. \tag{2.8}$$

$$\mathrm{T}(u)_{z=3} = \left[\frac{S_{u,0}+S_{u+1,0}+S_{u+2,0}}{t_u+t_{u+1}+t_{u+2}} \quad \dots \quad \frac{S_{u,k-1}+S_{u+1,k-1}+S_{u+2,k-1}}{t_u+t_{u+1}+t_{u+2}}\right]. \tag{2.9}$$

De cada vector *T(u)* se selecciona la mejor representación que cumpla la condición $\tau(u,j) < \widehat{bw}$. Una vez calculados todos los vectores $T(u)_{z=1..\theta}$ se genera un vector $T_q(u)=\{q_{(z=1)}, q_{(z=2)}, \dots, q_{(z=\theta)}\}$ con las representaciones seleccionadas. La representación seleccionada para el segmento *u* será la menor del vector $T_q(u)$, i.e. $\xi(u) = min\,\{T_q(u)\}$.

2.3.2. Implementación

Como se ha visto con anterioridad, la librería ExoPlayer pone a disposición del algoritmo de adaptación toda la información de los flujos multimedia en instancias de la clase *Format*. Ahí podemos encontrar prácticamente toda la información que se dispone de un flujo de audio/vídeo/texto salvo el tamaño de los segmentos que lo conforman. Así pues, para implementar el algoritmo propuesto es necesario modificar las definiciones que la propia librería hace de los diferentes módulos y cómo se relacionan entre sí.

Con el objetivo de minimizar el impacto de las modificaciones necesarias para conseguir la funcionalidad deseada, se ha optado por una implementación que aprovecha el comportamiento de inicialización existente y que únicamente requiere añadir una nueva definición en la interfaz *TrackSelection*. En particular el método añadido se puede observar en la siguiente figura.

```
1.  void dashSegmentIndexLoaded(int trackIndex, ChunkIndex chunkIndex);
```

Figura 2.11. Modificación de *TrackSelection* para la recepción de índices de segmentos (git commit 8331defc7).

A través de la función *dashSegmentIndexLoaded* la librería de reproducción informa al algoritmo de adaptación de que se han obtenido los índices de los segmentos (*chunkIndex*), y por tanto sus tamaños, de la representación identificada por *trackIndex*.

Este será el punto de entrada de la información de índices de segmento para la implementación de Look Ahead, así como para cualquier otra implementación que lo requiera. Para que la modificación de la interfaz *TrackSelection* no implique la modificación del resto de algoritmos ya desarrollados para ExoPlayer, se ha optado por realizar la implementación de la estructura de datos y la ingesta de los propios datos en la clase abstracta *BaseTrackSelection* que, tal como se muestra en la Figura 2.6, es la base, valga la redundancia, de todas las implementaciones de *TrackSelection*.

Con esta implementación se consigue poner a disposición de todos los algoritmos de adaptación la información relativa al tamaño de los segmentos de las representaciones que hayan sido inicializadas. Y es justamente en la inicialización donde los algoritmos que deseen hacer uso de esta información, incluido Look Ahead, deberán realizar el proceso que se pasa a describir.

Como prácticamente toda la funcionalidad en ExoPlayer, la obtención de los segmentos de los flujos multimedia también está modularizada, de forma que puede ser fácilmente sustituible. En la Figura 2.12 se puede observar la jerarquía de *ChunkSource*, la diferenciación entre DASH y *Smooth Streaming* a través de sendas subinterfaces y sus implementaciones por defecto.

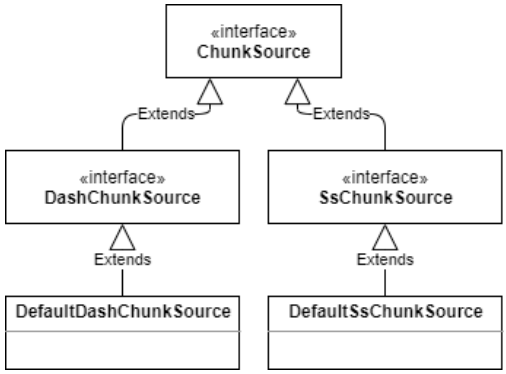


Figura 2.12. Jerarquía de clases de *ChunkSource* (ExoPlayer git commit 7d3f54a37).

DefaultDashChunkSource es, de esta forma, la clase encargada de la obtención de los segmentos de los flujos multimedia. Para ello, y debido al carácter asíncrono del proceso de descarga de segmentos al resto de hilos de ejecución del player, la clase *ChunkSourceStream*, cuya función se podría simplificar como la de conexión entre el

buffer de reproducción y la obtención de los segmentos, invocará el método *getNextChunk()* de *DefaultDashChunkSource* del que se muestra un extracto a continuación.

```
1.  public final void getNextChunk(MediaChunk previous, long playbackPositionUs, ChunkHolder
     out) {
2.  ...
3.     long bufferedDurationUs = previous != null ? (previous.endTimeUs - playbackPositionUs
    ) : 0;
4.     trackSelection.updateSelectedTrack(bufferedDurationUs, playbackPositionUs, previous !
    = null ? previous.endTimeUs  : 0);
5.     RepresentationHolder representationHolder =
6.         representationHolders[trackSelection.getSelectedIndex()]; //Here it is called aga
    in getSelectedIndex()
7.     if (representationHolder.extractorWrapper != null) {
8.       Representation selectedRepresentation = representationHolder.representation;
9.       RangedUri pendingInitializationUri = null;
10.      RangedUri pendingIndexUri = null;
11.      if (representationHolder.extractorWrapper.getSampleFormats() == null) {
12.        pendingInitializationUri = selectedRepresentation.getInitializationUri();
13.      }
14.      if (representationHolder.segmentIndex == null) {
15.        pendingIndexUri = selectedRepresentation.getIndexUri();
16.      }
17.      if (pendingInitializationUri != null || pendingIndexUri != null) {
18.        // We have initialization and/or index requests to make.
19.        out.chunk = newInitializationChunk(representationHolder, dataSource,
20.            trackSelection.getSelectedFormat(), trackSelection.getSelectionReason(),
21.            trackSelection.getSelectionData(),pendingInitializationUri,pendingIndexUri);
22.        return;
23.      }
24.    }
25. ...
26.  }
```

Figura 2.13. Extracto del método *getNextChunk()* de *DefaultDashChunkSource* (ExoPlayer git commit 7d3f54a37).

La invocación de este método por parte *ChunkSampleStream* tiene como objetivo obtener el siguiente segmento del stream multimedia en función del segmento anterior y la posición de reproducción. El resultado de la invocación deberá ser la incorporación al parámetro de entrada (*ChunkHolder out*) del segmento descargado.

En el extracto de la Figura 2.13 se muestra como en las líneas 11-16 se comprueba si la representación ya dispone de la información de inicialización y de los índices de segmentos. En el caso de que falte alguno de los dos elementos se procederá a realizar la petición de inicialización (líneas 17-22). En esta situación, al no obtener el segmento

deseado, si no la inicialización de la representación, *ChunkSampleStream* volverá a invocar *getNextChunk()* de *DefaultChunkSource* con los mismos parámetros de entrada.

Es este comportamiento del que se hace uso para minimizar el impacto de las modificaciones propuestas. Para aprovecharlo, los algoritmos que deseen conocer los tamaños de los segmentos de todas las representaciones deberán, antes de realizar una selección de representación efectiva, seleccionar todas las representaciones disponibles de forma secuencial para que, de esta forma, *DefaultChunkSource* las inicialice. A modo de ejemplo, la Figura 2.14 muestra un código que se asegura de que antes de proceder a realizar la selección de la representación se selecciona cada una de las representaciones disponibles.

```
1.  // init all tracks
2.  int uninitializedTrackIndex = selectUninitializedTrack();
3.  if (!(uninitializedTrackIndex < 0)) {
4.    selectedIndex = uninitializedTrackIndex;
5.    return;
6.  }
```

Figura 2.14. Extracto del método *updateSelectedTrack()* de *LookAheadTrackSelection* (git commit 7d3f54a37).

De esta forma, y teniendo en cuenta el comportamiento descrito de *DefaultDashChunkSource*, se consigue inicializar todas las representaciones disponibles.

Para terminar con las modificaciones realizadas a ExoPlayer, sólo queda mostrar el punto en el que se ofrece la información de los índices de segmentos a las diferentes implementaciones de *TrackSelection* (a través de *BaseTrackSelection*). Para ello, en la Figura 2.15 se muestra destacada la línea 16 que es la que hace uso del método que hemos añadido a la interfaz *TrackSelection*.

```
1.  @Override
2.  public void onChunkLoadCompleted(Chunk chunk) {
3.    if (chunk instanceof InitializationChunk) {
4.
5.      InitializationChunk initializationChunk = (InitializationChunk) chunk;
6.      int trackIndex =trackSelection.indexOf(initializationChunk.trackFormat);
7.      RepresentationHolder representationHolder = representationHolders[trackIn
    dex];
8.      // The null check avoids overwriting an index obtained from the manifest
    with one obtained
9.      // from the stream. If the manifest defines an index then the stream shou
    ldn't, but in cases
10.     // where it does we should ignore it.
11.     if (representationHolder.segmentIndex == null) {
12.       SeekMap seekMap = representationHolder.extractorWrapper.getSeekMap();
13.       if (seekMap != null) {
14.         ChunkIndex chunkIndex = (ChunkIndex) seekMap;
```

```
15.          representationHolder.segmentIndex = new DashWrappingSegmentIndex(chun
    kIndex);
16.          trackSelection.dashSegmentIndexLoaded(trackIndex, chunkIndex);
17.        }
18.      }
19.    }
20.  }
```

Figura 2.15. Método *onChunkLoadCompleted* () de *DefaultDashChunkSource* (git commit 7d3f54a37).

Con esta modificación es suficiente para que, a través de *BaseTrackSelection*, los distintos algoritmos de adaptación dispongan de la información relativa al tamaño de los segmentos de las distintas representaciones.

2.4. Metodología

2.4.1.1. Algoritmos evaluados

Ante la falta de disponibilidad de algoritmos de adaptación para ExoPlayer v2, se han desarrollado e integrado con la librería los anteriormente comentados algoritmos Müller y SARA, además del propuesto Look Ahead. La selección de estos algoritmos se ha basado, por un lado, en la adaptabilidad de ExoPlayer para proporcionar la información y las capacidades necesarias por los algoritmos y, por otro, en que las especificaciones de las diferentes propuestas fueran suficientes para desarrollar una implementación razonablemente parecida a sus respectivas definiciones.

A modo de ejemplo, algoritmos como el BOLA-E, aparte de su gran complejidad, requieren que el player disponga de la capacidad de, una vez iniciada la descarga de un segmento, detener la descarga en curso para iniciar la descarga de otra representación del mismo segmento. ExoPlayer, en el momento de realizar esta tesis, no dispone de dicha capacidad, por lo que la implementación de este algoritmo para ExoPlayer mermaría sus capacidades. Por otro lado, algoritmos como SABRE [36] utilizan técnicas que no están disponibles en las plataformas móviles, como son el cambio de parámetros de la pila de TCP/IP. De la misma forma, SQUAD [27] presenta un algoritmo de adaptación que, entre otras cosas, propone un novedoso método para la estimación de la tasa de transferencia disponible que utiliza información de la llegada de los paquetes de TCP para hacer la estimación. En la plataforma Android no es posible conocer los detalles acerca de la llegada de los paquetes en una conexión TCP/IP, por lo que la implementación de esta propuesta no es viable.

A las implementaciones desarrolladas en esta tesis hay que añadir la implementación que va incluida en la propia librería: *AdaptiveTrackSelection* y que hemos denominado ExoPlayer por simplicidad.

De esta forma, la evaluación constará de 4 algoritmos de adaptación: Müller, ExoPlayer, Look Ahead y SARA. De los cuales, los tres primeros utilizarán las implementaciones

por defecto, tanto de la estimación de la tasa de transferencia como de la gestión del buffer. En el caso de SARA, utilizará una estimación de la tasa de transferencia propia tal y como está definida en [22] y con los umbrales del buffer: I=5, B_{α}=12,5, B_{β}=25 y B_{max}=30. Estos umbrales han sido derivados de los propuestos por los autores pero escalados a un tamaño del buffer de 30 s por equidad con el resto de algoritmos ya que los valores que proponen los autores están adaptados a un tamaño del buffer de 40 s.

En lo que respecta a Look Ahead, se ha establecido $\theta=1$ para su comparación con el resto de los algoritmos puesto que este es el peor caso en relación con las paradas, como se puede ver en el apartado 2.5 Evaluación. Adicionalmente, se evaluará por separado el efecto de θ en la reproducción.

2.4.1.2. Vídeos de prueba

Los vídeos utilizados para las pruebas han sido creados por la "Blender Foundation" [37]. Estos vídeos están publicados bajo licencia Creative Commons por lo que son de libre utilización. En particular, se han utilizados dos vídeos de alrededor de 10 minutos cada uno: "Elephants Dream" y "Tears of Steel". Adicionalmente, ante la falta de material, a disposición y de larga duración, se ha creado un vídeo de alrededor de 45 minutos, al que denominamos "Mix", resultado de la concatenación de los dos vídeos mencionados anteriormente, así como de dos más del mismo origen: "Sintel" y "Big Buck Bunny". El contenido de estos vídeos contempla tanto la animación creada por ordenador como personajes reales con efectos especiales creados por ordenador. En la Tabla 2.1 se muestran los datos básicos de los vídeos que forman la prueba.

Tabla 2.1. Características de los vídeos de prueba.

Vídeo	Duración (s)	Número de Segmentos	Tamaño de Imagen	Codec
Elephants Dream	654	66	1920x1080	VP9
Tears of Steel	734	74	1920x1080	VP9
Mix (Sintel - Big Buck Bunny - Elephants Dream - Tears of Steel)	2757	276	1920x1080	VP9

Los vídeos has sido codificados utilizando el codec VP9 y creando 12 representaciones, todas con la misma resolución de Full HD (1080p24) y un tamaño de segmento fijo de 10 segundos. Las diferentes representaciones han sido codificadas utilizando el parámetro de calidad constante CRF desde 5 hasta 60 en pasos de 5. En la Figura 2.16 se muestran los perfiles de bitrate de 4 de las 12 representaciones, por claridad, del vídeo "Elephants Dream".

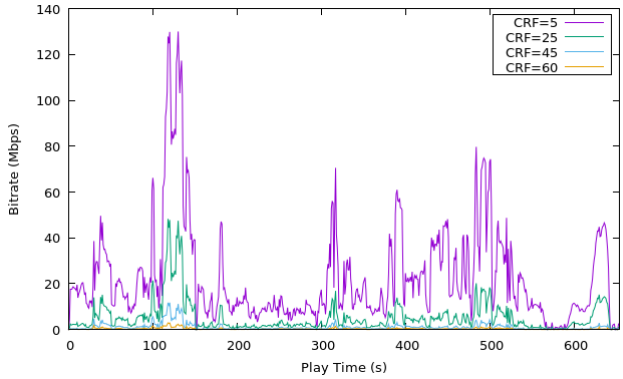


Figura 2.16. Evolución del bitrate en el tiempo para diferentes valores de CRF del vídeo "Elephants Dream".

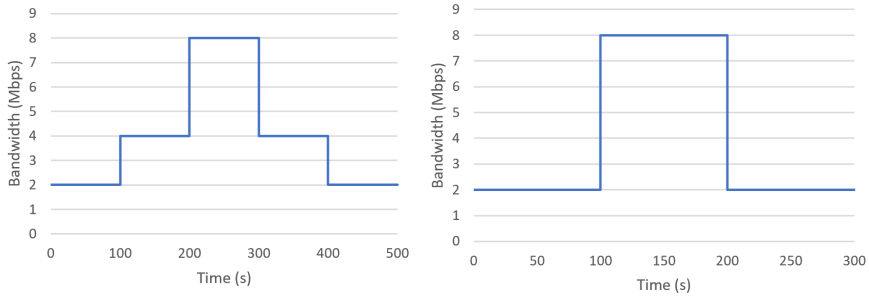
Merece la pena resaltar la gran variabilidad del bitrate cuando la codificación se lleva a cabo con parámetros de calidad constante. Si bien es cierto que esta gráfica contiene de las variaciones más abruptas de los vídeos utilizados, parece evidente que tener en cuenta esta información puede resultar de gran utilidad a la hora de reproducir el contenido. A modo de ejemplo, se puede observar que el pico en el segundo 120 de la calidad con CRF=5 (130 Mbps) quintuplica la media de la propia calidad (22,75 Mbps). En el caso de CRF=45, el pico mencionado (11,75 Mbps) es diez veces mayor que la media (1,13 Mbps).

2.4.1.3. Escenarios de prueba

Las evaluaciones se han realizado utilizando 9 escenarios de conectividad (canales), con diferentes tasas de transmisión y variabilidad de la tasa de transferencia disponible. Del total, 4 son escenarios con una tasa de transferencia constante (1, 2, 5 y 10 Mbps) y 5 lo son con una tasa de transferencia variable. En particular, los dos primeros definen un cambio en la tasa de transferencia cada 100 s de forma cíclica. El primero entre 2, 4, 8 y 4 Mbps y el segundo entre 2 y 8 Mbps. Este último canal es utilizado en el siguiente capítulo para la evaluación las prestaciones de Look Ahead, pero se introduce aquí con el fin de agrupar las definiciones de los escenarios utilizados. La Figura 2.17 muestra el perfil de tasa de transferencia de los dos escenarios sintéticos con tasa de transferencia variable.

Los tres últimos escenarios con tasa de transferencia variable, por su parte, están definidos por medidas de campo sobre redes 4G llevadas a cabo por la Ghent University [38], de las cuales dos fueron tomadas a bordo de autobuses (4G-bus y 4G-bus2) y la última en coche (4G-car). La Figura 2.18 muestra los escenarios 4G-car y 4G-bus que han sido utilizados para la evaluación de los algoritmos de adaptación mientras que la Figura 2.19 muestra la traza del escenario 4G-bus2 utilizado para la evaluación del parámetro θ del algoritmo Look Ahead. En las trazas 4G se puede observar que, aunque

la medias son significativas, hay instantes de tiempo donde se producen caídas importantes de la tasa de transferencia.



a) 2, 4, 8 y 4 Mbps b) 2 y 8 Mbps

Figura 2.17. Tasa de transferencia de los escenarios sintéticos con tasa de transferencia variable.

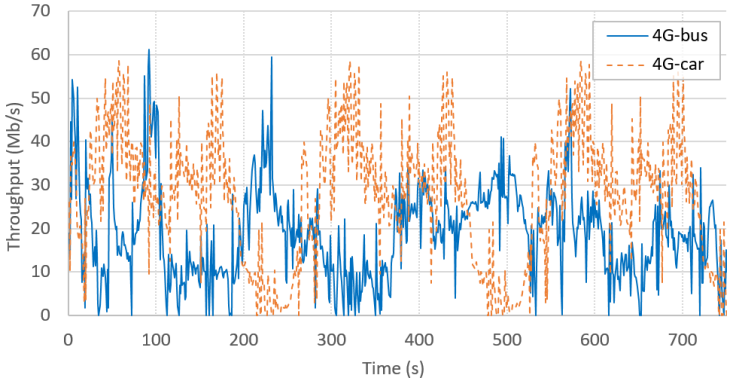


Figura 2.18. Tasa de transferencia de los escenarios 4G-bus y 4G-car.

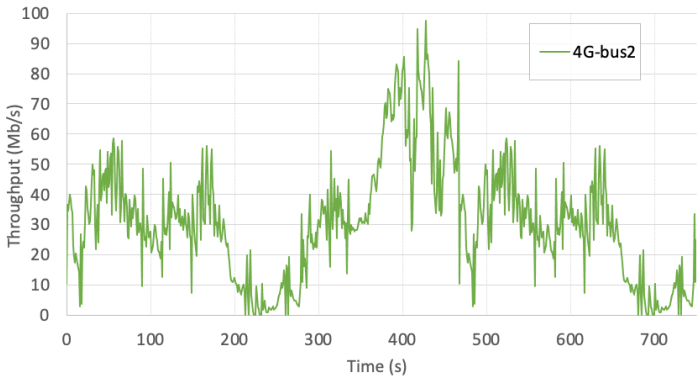


Figura 2.19. Tasa de transferencia del escenario 4G-bus2.

Para aplicar los diferentes canales durante las pruebas se ha desarrollado un módulo de control de la velocidad de la descarga para ExoPlayer.

El módulo desarrollado vuelve a hacer uso de la arquitectura modular de ExoPlayer. En este caso, la obtención de los contenidos también es una estructura modular, de modo que cualquier implementación de *DataSource* es suficiente para la obtención de los contenidos. En particular, ExoPlayer propone una subinterfaz de *DataSource* llamada *HttpDataSource* que, como su propio nombre indica, trata de agrupar todos los orígenes de datos que tengan en común la utilización de HTTP como protocolo de transporte. La implementación por defecto de esta interfaz es *DefaultDataSource*. Para crear el módulo de control de velocidad se ha creado la clase *RateLimitedDataSource* a partir de *DefaultDataSource* y se ha añadido la funcionalidad de limitación a través de la utilidad *RateLimiter* de la librería *Guava* [39].

De esta forma, *RateLimitedDataSource* se inicializa con la información relativa al escenario de anchos de banda. Esta información la utiliza el módulo para ir variando la tasa a la que se permite a la conexión HTTP informar de la llegada de los bytes que compongan la descarga del contenido.

2.4.1.4. Entorno de ejecución

En relación con la ejecución de las pruebas, éstas se han llevado a cabo en un ordenador *HP Pavilion dv6* (i7/6GB). Gracias al módulo de control de la velocidad, la reproducción del contenido y el propio contenido se han podido ubicar en la misma máquina. Para la ejecución del player de vídeo se ha utilizado el simulador de Android del *Android SDK* y como servidor de contenido se ha utilizado *Apache 2.4*.

Para la obtención de los resultados mostrados en la evaluación se han realizado 5 iteraciones de cada algoritmo, canal y vídeo estudiado de modo que los resultados obtenidos cuentan con intervalos de confianza estrechos. En particular, las pruebas realizadas suman algo más de 111 horas de reproducción de contenido.

En lo que respecta a la replicabilidad de los experimentos, se ha puesto a disposición de quien esté interesado un sitio web [40] que contiene tanto los contenidos como la aplicación para la plataforma Android que se ha utilizado en las pruebas. Adicionalmente, el código relacionado con la propuesta se puede encontrar en el repositorio público creado a tal efecto [41].

2.4.1.5. Parámetros evaluados

Tal como se destaca en la literatura [42], existe un buen número de indicadores de rendimiento para el streaming sobre DASH. En esta evaluación se han utilizado 5 de los más importantes:

- **Calidad media**: La calidad media de los segmentos seleccionados para la reproducción en función de la representación. Corresponde con el promedio de las representaciones reproducidas en la visualización de un contenido.

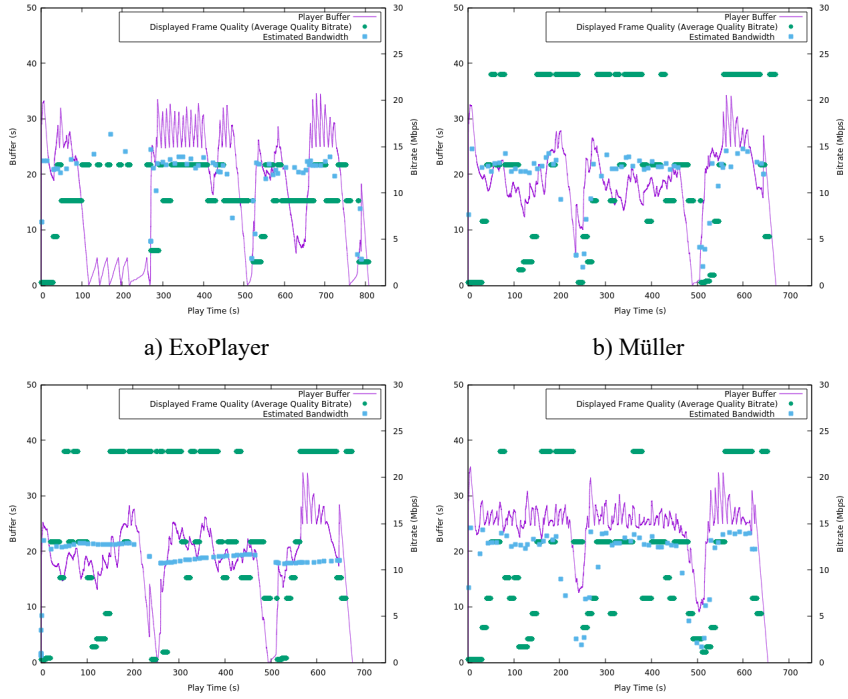
- **Número de paradas**: Número de paradas debidas al vaciamiento del buffer de reproducción.
- **Tiempo total de parada**: Tiempo de parada total acumulado durante los eventos de parada.
- **Número de cambios de representación**: Número total de cambios de representación sin tener en cuenta la distancia entre las representaciones seleccionadas.
- **Retardo inicial**: Tiempo transcurrido entre el evento de inicio de la reproducción y el momento en que se muestra la primera imagen.

2.5. Evaluación

En este apartado se presenta la evaluación realizada del algoritmo de adaptación de vídeo propuesto, Look Ahead, con otros algoritmos existentes en la literatura. En particular, se evaluarán los 4 algoritmos de adaptación mencionados en siete de los escenarios propuestos. Los parámetros que se evalúan son: número de paradas, duración total de las paradas, la representación media de los segmentos reproducidos y el número de cambios de representación.

2.5.1. Trazas de ejecución

A modo de extracto de esta evaluación, la Figura 2.20 muestra una iteración en particular del vídeo “Elephants Dream” utilizando los diferentes algoritmos en cuestión evaluados en el canal 4G-car.



a) ExoPlayer b) Müller

c) SARA d) Look Ahead

Figura 2.20. Evaluación de diferentes algoritmos en el canal 4G-car para el vídeo “Elephants Dream”.

Las gráficas de la Figura 2.20 muestran 3 parámetros: el tamaño del buffer en segundos, la estimación de la tasa de transferencia y la representación seleccionada representada por el bitrate medio de cada representación en Mbps.

Analizando las figuras, lo primero que se puede observar es que en la reproducción conducida por el algoritmo ExoPlayer se producen bastantes paradas, mientras que utilizando los algoritmos Müller y SARA se producen una y dos, respectivamente. Por su parte, al utilizar el algoritmo Look Ahead no se produce ninguna parada.

Los momentos en los que el buffer se vacía en las distintas reproducciones, en general, se corresponde con los instantes de tiempo en los que, o bien se produce una bajada en la tasa de transferencia disponible, o bien un incremento de los tamaños de los segmentos respecto a las medias de sus correspondientes representaciones. Como ejemplo, la Figura 2.16 muestra un incremento generalizado del tamaño de los segmentos del vídeo “Elephants Dream” entre los 110 y 140 segundos que se corresponde con las paradas que se producen con el algoritmo ExoPlayer Figura 2.20a. De la misma forma, en la Figura 2.16 se puede observar un nuevo pico en el tamaño de los segmentos alrededor del segundo 500 que se une a una bajada en la tasa de transmisión disponible en ese mismo instante en la traza del canal 4G-car (Figura 2.18). Esta conjunción tiene como efecto una

drástica bajada en el tamaño de los buffers de las reproducciones que en el caso de Müller y SARA provoca un vaciado completo y, por tanto, una interrupción en la reproducción como muestran las figuras Figura 2.20b y Figura 2.20c.

La sucesión de vaciamientos de los buffers y sus consiguientes periodos de rebuffering, el buffer debe alcanzar los 5 segundos para que se reanude la reproducción, son los responsables del retraso acumulado en la reproducción. En las reproducciones mostradas en la Figura 2.20 los retrasos acumulados ascienden a 163, 21 y 24 segundos en los algoritmos ExoPlayer, Müller y SARA respectivamente. Look Ahead, en lo referente al estado del buffer, se mantiene bastante estable durante toda la reproducción solamente sufriendo dos bajadas importantes que, en este caso, suponen llegar a un mínimo de 9 segundos de buffer por lo que no se producen paradas en la reproducción.

2.5.2. Comparativa de los algoritmos

La Tabla 2.2 y la Tabla 2.3 muestran los resultados de la evaluación de los algoritmos estudiados en 7 escenarios diferentes. Las tablas muestran la media del número de paradas, la duración media de éstas, la representación media (con un rango de 0 a 11 puesto que el contenido está formado por 12 representaciones) y el número medio de cambios de representación. Los mejores valores en cada uno de los escenarios están resaltados en negrita.

Tabla 2.2. Comparación del número y la duración de las paradas.

		Número de paradas				Duración de las paradas (s)			
	Canal	Look Ahead	SARA	Müller	ExoP.	Look Ahead	SARA	Müller	ExoP.
Elephants Dream	1 Mbps	**0,00**	1,00	1,00	5,00	**0,00**	5,98	5,81	93,76
	2 Mbps	**0,00**	1,00	1,00	5,00	**0,00**	5,95	5,48	93,24
	5 Mbps	**0,00**	**0,00**	**0,00**	6,00	**0,00**	**0,00**	**0,00**	101,18
	10 Mbps	**0,00**	**0,00**	**0,00**	5,20	**0,00**	**0,00**	**0,00**	77,81
	2-4-8-4 Mbps	**0,00**	**0,00**	**0,00**	3,00	**0,00**	**0,00**	**0,00**	47,86
	4G-bus	**0,00**	0,40	0,60	4,00	**0,00**	2,58	9,49	82,56
	4G-car	**0,00**	2,00	0,80	7,80	**0,00**	21,46	13,00	156,73
Tears of Steel	1 Mbps	**0,00**	**0,00**	**0,00**	3,00	**0,00**	**0,00**	**0,00**	34,32
	2 Mbps	**0,00**	**0,00**	**0,00**	3,00	**0,00**	**0,00**	**0,00**	34,20
	5 Mbps	**0,00**	**0,00**	**0,00**	4,00	**0,00**	**0,00**	**0,00**	57,91
	10 Mbps	**0,00**	**0,00**	**0,00**	5,40	**0,00**	**0,00**	**0,00**	51,27
	2-4-8-4 Mbps	**0,00**	**0,00**	**0,00**	4,00	**0,00**	**0,00**	**0,00**	28,91
	4G-bus	**0,00**	**0,00**	**0,00**	4,00	**0,00**	**0,00**	**0,00**	45,35
	4G-car	**0,00**	2,60	0,80	7,80	**0,00**	30,04	7,43	94,66

Tabla 2.3. Comparación de la representación media y del número de cambios de representación.

		Representación media [0-11]				Número de cambios de representación			
	Canal	Look Ahead	SARA	Müller	ExoP.	Look Ahead	SARA	Müller	ExoP.
Elephants Dream	1 Mbps	3,27	**3,66**	3,64	1,94	44,80	49,20	49,00	**2,00**
	2 Mbps	3,30	**3,66**	3,64	1,94	45,20	49,20	49,20	**2,00**
	5 Mbps	6,56	**7,41**	7,25	5,71	47,40	48,20	45,20	**2,80**
	10 Mbps	8,17	**8,79**	8,66	6,66	37,20	40,40	42,40	**3,00**
	2-4-8-4 Mbps	6,07	6,17	**6,48**	4,90	46,40	51,60	44,60	**10,80**
	4G-bus	**8,74**	8,06	7,93	8,28	35,00	45,60	43,80	**28,40**
	4G-car	8,53	**9,10**	8,74	8,67	38,00	38,60	37,80	**27,60**
Tears of Steel	1 Mbps	2,78	**3,28**	**3,28**	1,95	52,00	52,40	54,00	**2,00**
	2 Mbps	2,77	**3,29**	3,28	1,95	52,60	53,80	54,60	**2,00**
	5 Mbps	6,07	6,61	**6,62**	5,84	43,60	54,40	50,80	**2,00**
	10 Mbps	7,09	**7,95**	7,94	6,81	48,60	48,20	46,20	**2,20**
	2-4-8-4 Mbps	5,52	5,78	**6,06**	5,05	45,60	53,40	54,20	**11,40**
	4G-bus	6,73	**8,45**	8,20	8,02	43,60	44,20	51,80	**37,40**
	4G-car	7,99	**8,23**	8,17	8,26	50,20	43,80	43,20	**18,80**

Los resultados mostrados en estas tablas están en línea con los mostrados en las figuras previas donde ExoPlayer, Müller y SARA sufren paradas en algunos escenarios, mientras que Look Ahead completa la reproducción en todos los escenarios sin paradas. Cabe destacar que en los escenarios con tasa de transmisión constante de 1 y 2 Mbps, a excepción de Look Ahead, todos los algoritmos tienen paradas. En los casos de Müller y ExoPlayer la justificación de las paradas en escenarios de tasa constante está en la variabilidad de los tamaños de los segmentos. En ambos casos, los algoritmos seleccionan una representación basándose en su bitrate medio sin conocer el tamaño real del segmento a descargar hasta que la descarga ya ha sido iniciada. Por su parte, SARA, a pesar de disponer de la información de los tamaños de los segmentos, también sufre una parada de media en estos escenarios. En su caso, la razón de estas paradas debe buscarse en la propia gestión de los diferentes estados que propone el algoritmo, puesto que SARA sí que dispone, de antemano, del tamaño de los segmentos que conforman las diferentes representaciones.

En los escenarios con tasas de transferencia basadas en patrones 4G, ningún algoritmo presenta paradas en el caso del escenario 4G-bus para el vídeo Tears of Steel, mientras que Look Ahead es el único que no lo hace en los escenarios 4G para el vídeo Elephants Dream.

La mejora debida a la ausencia de paradas por parte de Look Ahead tampoco puede achacarse a una exagerada bajada en la calidad de las representaciones seleccionadas puesto que, de media, Look Ahead escoge una representación un 4,7% inferior a la mejor representación media en el caso de Elephant Dream y de un 11% menor en el caso de Tears of Steel.

El algoritmo ExoPlayer es el peor parado en cuanto al número y duración de las paradas lo que denota su pobre funcionamiento cuando el contenido a reproducir está codificado con calidad constante. Este mal funcionamiento está justificado, en parte, por los pocos

cambios de representación que efectúa el algoritmo que le hacen obtener los mejores resultados en relación con este parámetro pero que limitan su capacidad de adaptación.

2.5.3. Evaluación del parámetro θ de Look Ahead

Para completar la evaluación del algoritmo propuesto, se ha analizado el efecto que el parámetro θ tiene sobre la reproducción. Este parámetro, como se ha visto con anterioridad, define el número de segmentos futuros que el algoritmo tendrá en cuenta para la selección de la representación del siguiente segmento. En este caso se han evaluado el número de paradas, la representación media y el número de cambios de representación. Se han considerado dos escenarios de tasas de transferencia: uno en donde no se producen paradas en la reproducción y otro en el que sí que las hay.

En lo que respecta al primer escenario, en la Figura 2.21 y Figura 2.22 se muestra el comportamiento del algoritmo Look Ahead para diferentes valores de θ: 1, 2, 3, 5, 10 y 20 al reproducir los vídeos "Elephants Dream" (ED) y "Tears of Steel" (ToS) en las trazas de tasa de transferencia constante de 1 y 5 Mbps y con la traza 4G-bus. Para todos los casos no se producen paradas en la reproducción y el tiempo de inicio de la reproducción no es significativo en ninguno de ellos variando entre 1,4 y 2 s en el caso de 1 Mbps, entre 0,7 y 1 s en el caso de 5 Mbps, y entre 0,4 y 0,6 s en el caso del escenario 4G-bus.

La Figura 2.21 muestra la representación media de 5 iteraciones para cada combinación de vídeo, escenario y valor de θ. En la figura se ve como el algoritmo va tomando, de media, representaciones con menor bitrate medio a medida que el valor de θ va aumentando. En lo que respecta al número de cambios de representación, la Figura 2.22 refleja que, en los escenarios con tasa de transferencia constante, en general, al aumentar el valor de θ se reduce el número de cambios de representación, mientras que en el caso del escenario 4G-bus la tendencia no es clara.

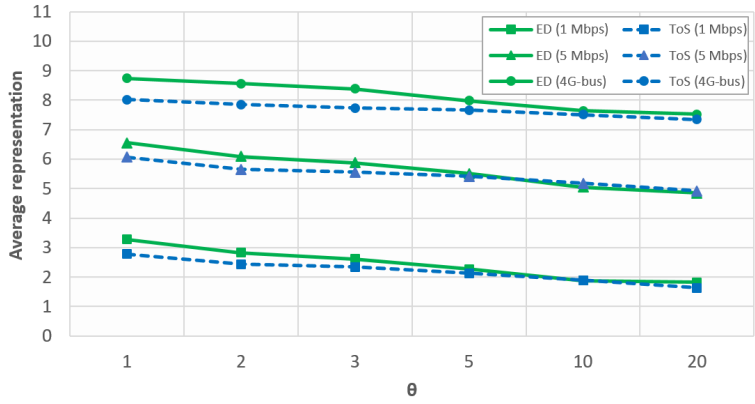


Figura 2.21. Representación media de Look Ahead para diferentes valores de θ.

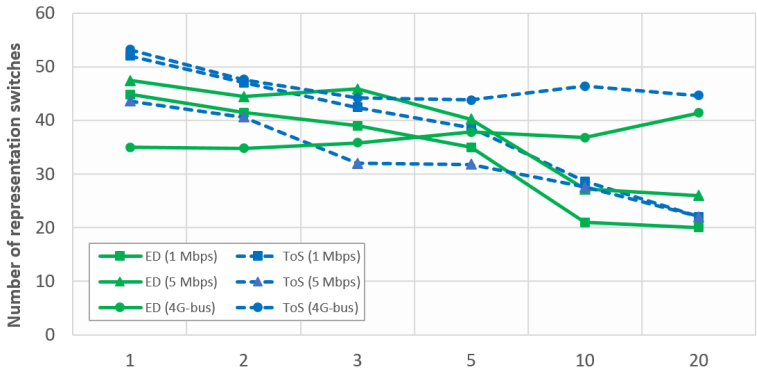


Figura 2.22. Número de cambios de representación de Look Ahead para diferentes valores de θ.

Pasando al segundo escenario donde sí que se producen paradas en la reproducción, se evalua el efecto de variar el valor de θ en el número y duración de las paradas por vaciado de buffer. Para ello se ha utilizado el vídeo "Mix" sobre una traza de tasa de transferencia altamente variable también obtenida de la Ghent University [38]. La traza en cuestión es de una red 4G medida mientras se realiza un desplazamiento en autobús. A esta traza la hemos denominado 4G-bus2.

La Figura 2.23 muestra el número de paradas y la representación media (eje derecho) para diferentes valores de θ. En la figura se puede observar que el número de paradas desciende a medida que va aumentando el valor de θ pasando de 3 paradas para θ=1 a la ausencia de paradas a partir de θ=4. En lo referente a la representación media, la figura muestra como la diferencia entre θ=1 y θ=4 es de alrededor de 0,7 (representación media) mientras que la diferencia total entre el mínimo y el máximo valor de θ es menor de 1 (representación media). Por su parte, el eje izquierdo muestra el valor de QoE_{VMAF} para los diferentes valores de θ. El valor proporcionado por el modelo QoE_{VMAF}, como se verá en detalle en el Capítulo 3, integra tanto la calidad del vídeo reproducido como los datos de paradas en la reproducción y cambios de representaciones. En la Figura 2.23 se puede ver como este valor va aumentando de forma significativa a medida que descienden el número de paradas. Una vez que el valor de θ proporciona reproducciones sin paradas, el valor arrojado por el modelo QoE_{VMAF} pasa a descender de forma muy gradual.

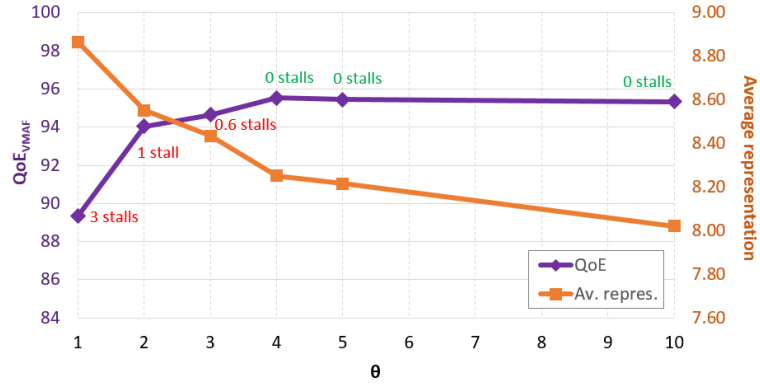


Figura 2.23. Evaluación de θ al reproducir el vídeo "Mix" sobre la traza 4G-bus2.

Siguiendo con la evaluación de θ, la Figura 2.24 muestra el estado del buffer del reproductor de vídeo al reproducir el vídeo "Mix" utilizando la traza 4G-bus2 para diferentes valores de θ. Para θ=1 se producen tres paradas que se corresponden a los instantes de tiempo donde el buffer de vídeo llega a vaciarse (durante la descarga de los segmentos 22, 116 y 162). Cuando el valor de θ pasa a ser de 2 se puede observar cómo, de media, el estado del buffer mejora, ya que pasa a acercarse menos veces a cotas bajas, aunque sigue produciéndose una parada. Finalmente, el usar el valor de θ=4 tiene como efecto una menor fluctuación del buffer que, al no vaciarse, no provoca paradas en la reproducción.

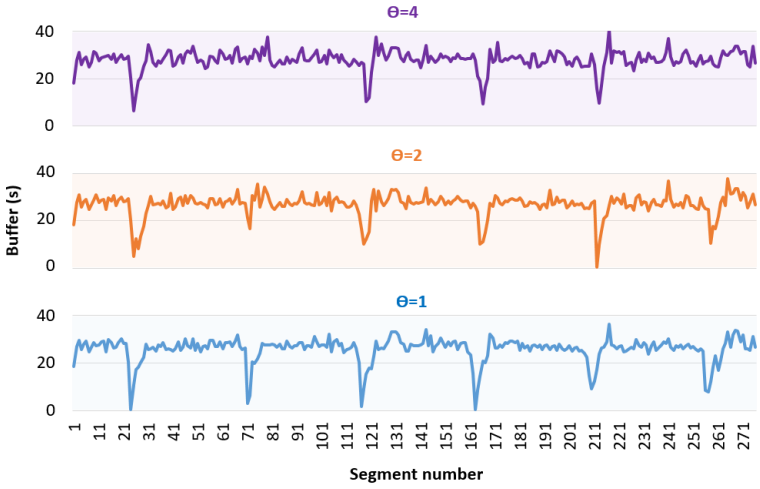


Figura 2.24. Evaluación del estado del buffer para diferentes valores de θ al reproducir el vídeo "Mix" sobre la traza 4G-bus2.

2.6. Conclusiones y trabajo futuro

Este capítulo ha expuesto como la variabilidad del bitrate de los vídeos afecta a la eficacia de los algoritmos de adaptación. Incluso en escenarios con tasas de transferencia fijas, los algoritmos que no tienen en cuenta esta variabilidad pueden sufrir paradas en la reproducción por vaciamiento del buffer.

La variación en los tamaños de los segmentos es indefectible, aunque mucho más notoria en los vídeos codificados con calidad constante. Si bien este tipo de codificación incrementa la variabilidad de los segmentos, una correcta gestión de esta variabilidad puede aportar una mejora en la calidad de experiencia de los usuarios.

Se ha propuesto un algoritmo de adaptación, Look Ahead, que es capaz de obtener y utilizar el tamaño de los segmentos a la hora de realizar la selección de las representaciones que se deben obtener. Para ello, Look Ahead hace uso de las estructuras propias de los contenedores multimedia, por lo que su implementación es totalmente compatible con el estándar DASH y no necesita ninguna modificación.

El algoritmo propuesto se ha implementado para la librería de reproducción de vídeo ExoPlayer. Para su evaluación se han implementado, adicionalmente, los algoritmos de adaptación Müller y SARA que, junto al algoritmo que lleva por defecto ExoPlayer, hace un total de 4 algoritmos utilizados en la evaluación. De entre ellos, SARA también tiene en cuenta el tamaño de los segmentos futuros, aunque su implementación requiere de una modificación de la MPD tal y como está definida en DASH.

Los resultados muestran como Look Ahead mejora al resto de algoritmos tanto en el número como en la duración de las paradas en la reproducción de contenido codificado con calidad constante. Como contrapartidas, el algoritmo propuesto selecciona, de media, una representación ligeramente inferior al mismo tiempo que eleva el número de cambios de representación.

De los algoritmos evaluados en este capítulo, Look Ahead es el único que no hace uso del estado del buffer para la toma de decisiones y, a pesar de ello, arroja buenos resultados. Estos resultados animan a trabajar en una evolución del algoritmo (Look Ahead with Buffer Budget) que haga uso de la información del estado del buffer para, por ejemplo, evitar bajar la representación mostrada de un segmento si, predeciblemente, al segmento siguiente se va a volver a subir de calidad y el estado del buffer así lo permite. Dicha evolución de Look Ahead considerando el estado del buffer forma parte del trabajo futuro.

Otro punto de interés para el autor de esta tesis es la aparente limitación que los diferentes trabajos que tratan algoritmos de adaptación, incluido este, se imponen con relación al tamaño del buffer de reproducción. Si bien todos los algoritmos deben ser evaluados en escenarios que supongan un reto para la reproducción, la limitación del tamaño del buffer sin causas justificadas puede, así mismo, ocultar soluciones simples a determinadas situaciones. Por ejemplo, las bajadas de la tasa de transferencia momentáneas o crear

nuevos objetivos para los algoritmos, como la descarga de segmentos ya descargados con mejor calidad en determinadas situaciones.

Finalmente, con el objetivo de que las pruebas sean replicables se han hecho públicos tanto los contenidos utilizados como una aplicación para la plataforma Android que puede utilizarse para replicar las pruebas (https://lookahead.iteam.upv.es/) [40]. Adicionalmente se ha publicado una herramienta que facilita la creación de contenido DASH (https://github.com/comm-iteam/dashgen) [43] y las modificaciones e implementaciones de los algoritmos de adaptación desarrollados para ExoPlayer (https://github.com/comm-iteam/ExoPlayer) [41].

Capítulo 3

Nuevas métricas objetivas de QoE para DASH

3.1. Introducción

En la ITU-T P.10/G.100 [44] se define la calidad de experiencia (*Quality of Experience*, QoE) como "El nivel de disfrute o de fastidio del usuario de una aplicación o servicio". Continúa, añadiendo una nota en la que advierte que se espera que la definición evolucione debido a que es un campo activo de la investigación en la actualidad.

Los factores que influyen en este dato son, según la propia ITU (*International Telecommunication Union*), "el tipo y las características de la aplicación o el servicio, el contexto de uso, las expectativas del usuario con respecto a la aplicación o servicio y su cumplimiento, el trasfondo cultural del usuario y otros factores cuyo número bien puede ampliarse con las investigaciones en la materia".

Son tantos y tan variados los factores que intervienen, que la única forma de conocer la experiencia de un usuario es preguntándole directamente. Claro está, este método no es escalable ni mucho menos puede obtenerse de manera simultánea a la prestación del servicio, por lo que es de interés desarrollar otros mecanismos indirectos de calcular, al menos con una estimación, cuál sería la respuesta de los usuarios en caso de que se les preguntase directamente.

En este capítulo se presentan algunas propuestas de estimadores de la calidad de la experiencia y se proponen varios métodos novedosos que mejoran las estimaciones

existentes. Las variables utilizadas en los modelos propuestos en este capítulo están agrupadas en la Tabla 3.1.

Tabla 3.1. Resumen de variables utilizadas en los modelos QoE propuestos.

B_k	Ocupación del buffer de reproducción durante la descarga del segmento k	T_s	Retardo del inicio de la reproducción
C_k	Tasa de transferencia en la descarga del segmento k	$VMAF(\cdot)$	Función de cálculo del PSNR
d	Duración total del vídeo	β	Ponderación del peso de las variaciones en la calidad de los segmentos
K	Número de segmentos de un vídeo	γ	Ponderación del peso del tiempo en estado de parada
L	Duración de los segmentos	δ	Ponderación del peso del retraso en el inicio de la reproducción
$PSNR(\cdot)$	Función de cálculo del PSNR	ζ	Ponderación del peso de los cambios de representación
$q(\cdot)$	Función creciente de mapeo del bitrate a la calidad	η	Ponderación del peso del tiempo en estado de parada
$\mathfrak{R}$	Conjunto de bitrates medios de las representaciones de un vídeo	λ	Ponderación del peso de la calidad del vídeo
R_k	Bitrate medio considerado para el segmento k	μ	Ponderación del peso del tiempo en estado de parada
$\mathfrak{R}_S$	Conjunto de bitrates medios de cada segmento de cada representación	ξ_k	Representación seleccionada para el segmento k

3.2. Estado del arte

Existen dos familias diferentes de test para la evaluación de la calidad de experiencia experimentada por los usuarios al consumir contenido multimedia: 1) test subjetivos, que se basan en la obtención de la información de los propios usuarios; y 2) test objetivos, que utilizan algoritmos para realizar la estimación de la calidad de experiencia con el objetivo de aproximarse lo más posible a la realidad experimentada por los propios usuarios. A continuación se introducen ambas familias.

3.2.1. Medidas subjetivas

La ITU ha publicado recomendaciones que proporcionan la metodología para la realización de evaluaciones subjetivas formalmente. Por ejemplo, la recomendación ITU-R BT.500 [45] describe varios métodos para estandarizar los test subjetivos conteniendo procedimientos y requisitos para la selección y correcta configuración de las pantallas, la selección de los sujetos de pruebas o la determinación de las pruebas y secuencias de vídeo óptimas. En la misma línea, una recomendación más reciente es la ITU-T P.913 [46] que, a su vez, es la evolución de la ITU-P.910 [47] y de la ITU-T P.911 [48]. Esta

recomendación describe métodos de valoración subjetiva no interactivos para la evaluación de la calidad audiovisual de la distribución de vídeo a través de redes como Internet.

Cada recomendación proporciona diferentes metodologías para calcular la QoE de los usuarios. En este sentido, una de las técnicas más utilizadas para la medición de la QoE es *Mean Opinion Score* (MOS), en la que diferentes usuarios valoran su experiencia respecto a la reproducción de un vídeo utilizando una escala entre 1 (baja satisfacción) y 5 (alta satisfacción). El MOS se calcula como la media del conjunto de las valoraciones proporcionadas por los sujetos de prueba.

Existen distintas formas de calcular el MOS. La metodología más simple se conoce con el nombre de *Single-Stimulus* (SS). Cuando se utiliza este método, los sujetos de prueba proporcionan su valoración basándose en una única visualización del contenido. A esta categoría pertenecen diferentes estrategias como *Absolute Category Rating* (ACR), *Single Stimulus Continuous Quality* (SSCQE), *Subjective Assessment of Multimedia Video Quality* (SAMVIQ) o *Multi-Stimuli with Hidden Reference and Anchor Points* (MUSHRA).

Aparte del MOS, existe otra medida de la QoE subjetiva de relevancia denominada *Differential MOS* (DMOS) en la cual un vídeo es comparado con otro de referencia. Entre las estrategias para evaluar el DMOS se encuentran *Double-Stimulus Continuous Quality Scale* (DSCQS), *Degradation Category Rating* (DCR) o *Comparison Category Rating* (CCR).

3.2.2. Medidas objetivas

3.2.2.1. Factores de influencia de la QoE

La realización de test subjetivos implica un elevado consumo de recursos derivados de la utilización de usuarios para la obtención de las valoraciones. Estos costes motivan la existencia de test objetivos, que son realizados por algoritmos que estiman la experiencia que los usuarios han tenido al consumir un contenido.

Para desarrollar métodos objetivos de evaluación de la QoE es necesario identificar los factores que influyen (IF, *Influence Factor*) a la QoE, su magnitud y sus interacciones si las hubiera.

En la propuesta de Skorin-Kapov et al. [49] se organizan los IFs en cuatro categorías:

a) **IFs del sistema**. Están formados por los datos más técnicos del sistema e incluyen todo lo relativo a la red de transmisión, tasa de transferencia disponible, interrupciones temporales, resoluciones, etc.

b) **IFs humanos**. Los factores humanos son relativos al consumidor del contenido e incluyen la expectativa, el contexto sociocultural, el estado de ánimo, relaciones previas con el servicio, etc.

c) **IFs de contexto**. Pertenecen al contexto los IFs relativos al tiempo y la localización del acto de consumición del servicio. Como ejemplo de esta categoría estarían el ambiente, la hora del día, el tipo de consumo (casual, contenido altamente esperado), etc.

d) **IFs de contenido**. Relativos a las características del contenido como la duración, la categoría del contenido o las complejidades espaciales o temporales.

De esta forma, el objetivo de los modelos de QoE es tomar como estrada los IFs que se consideren oportunos para obtener como resultado una estimación de la QoE que han experimentado los usuarios del servicio.

3.2.2.2. Evaluación del rendimiento

La forma de evaluar el rendimiento de un modelo de QoE descansa en tres factores, tal y como se dice en Video Quality Experts Group (VQEG) FRTV Phase I [50] y en VQEG FRTV Phase II [51]:

- **Precisión de la predicción**. Se refiere a la capacidad del modelo de predecir cuál hubiera sido la valoración subjetiva con poco error.
- **Monotonicidad de la predicción**. Se refiere al grado de concordancia de la predicción del modelo con las magnitudes relativas de las puntuaciones de la valoración subjetiva.
- **Consistencia de la predicción**. Hace referencia a la capacidad de un modelo de mantener predicciones precisas ante un amplio abanico de secuencias de vídeo y tipos de deficiencias visuales.

La precisión de la predicción se puede evaluar utilizando el *Pearson Linear Correlation Coefficient* (PLCC) entre las predicciones objetivas y subjetivas. De la misma forma, la monotonicidad puede ser evaluada con el *Spearman's Rank Correlation Coefficient* (SROCC) utilizando los mismos datos de entrada que para el PLCC. Por último, la consistencia de la predicción se puede evaluar con el *Outlier Ratio* (OR) que representa la ratio de puntuaciones "falsas" en relación con el total de las puntuaciones que están fuera del rango marcado por [MOS - 2σ, MOS + 2σ].

3.2.2.3. Clasificación de los modelos de QoE objetivos

Como bien se detalla en [52], existen múltiples métodos para la clasificación de los métodos objetivos de evaluación de la QoE. Los parámetros de entrada, el tipo de aplicación, el tipo de medidas, etc. pueden utilizarse como método de clasificación. En este trabajo de investigación vamos a introducir el método presentado por Takahashi et al. [53].

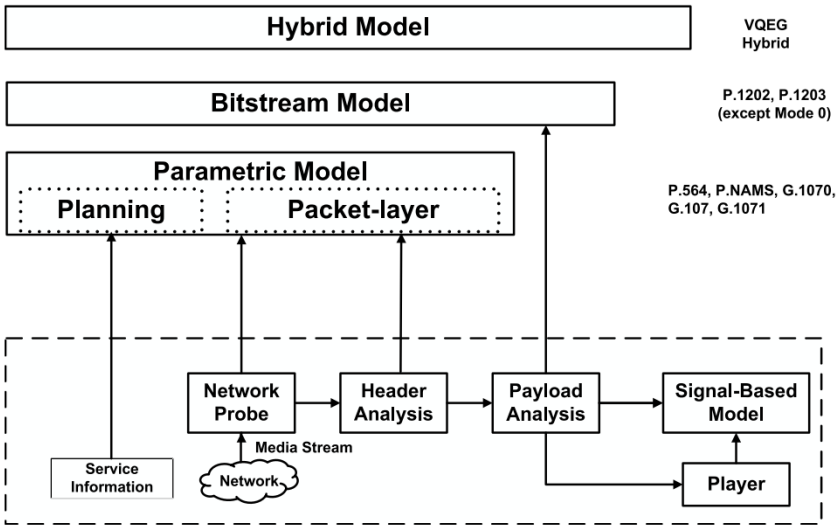


Figura 3.1. Clasificación de modelos QoE para aplicaciones de streaming de vídeo. Fuente: *QoE Modeling for HTTP Adaptive Video Streaming—A Survey and Open Challenges [52]*.

En la Figura 3.1 se pueden observar las diferentes clases de modelos que se describen a continuación.

3.2.2.3.1. Modelos basados en la señal

Este tipo de modelos utilizan la señal para evaluar la calidad del contenido a través de técnicas de *Video Quality Assessment* (VQA). Las métricas de VQA, en función de la cantidad de datos del contenido original que requieran, se pueden clasificar en *full reference* (FR), *reduced reference* (RR) y *no reference* (NR):

- **FR**. Disponen de acceso total al contenido previo al proceso de codificación y transmisión. Debido a que utilizan el contenido original, estas técnicas suelen ser más precisas que las RR o NR, pero añaden un nivel extra de complejidad, por lo que es más difícil su aplicación en entornos reales. Como ejemplo de este tipo de técnicas tenemos PSNR (*Peak Signal-to-Noise Ratio*), SSIM (*Structural Similarity Index Measure*) [55], recomendaciones de la ITU [56][57] y VMAF [13]-[15].
- **RR**. Este tipo de métricas disponen de un acceso limitado al contenido original. Forman parte de este tipo de técnicas las métricas RRIQA (*Reduced-Reference Image Quality Assessment*) [58], C4 (*Criterion v4.0*) [59] e ITU-T J.246 [60].
- **NR**. No disponen de acceso al contenido original y se basan en las características de la señal recibida para predecir la calidad. Algunas de las técnicas de este tipo son DIIVINE [61], BLIINDS [62], BRISQUE [63] y NIQE [64].

3.2.2.3.2. Modelos paramétricos

Los métodos paramétricos usan los datos medidos o esperados de las redes de comunicaciones para estimar la calidad. A su vez, pueden dividirse en modelos *packet-layer* y *planning*, los cuales hacen uso de la información de la red o la planificada para una determinada red, respectivamente. Son ejemplos de este tipo de modelos ITU-T Rec. P.564 for speech [65], ITU-T Rec. P. NAMS [66] y ITU-T Rec. G.1070 [67].

3.2.2.3.3. Modelos basados en el bitstream

Los modelos basados en el bitstream hacen uso del bitstream codificado y la información que de ellos se pueda extraer sin llegar a decodificar el contenido multimedia. En este tipo de modelos es frecuente utilizar información como el bitrate, frame rate, *Quantization Parameter* (QP), vectores de movimiento, etc. En esta categoría entran modelos como ITU-T P.1203 [68]. Al hacer uso de información relativa al codificador, estas técnicas suelen presentar dependencias del codec para el que fueron diseñadas.

3.2.2.3.4. Modelos híbridos

Por último, los modelos híbridos hacen uso de dos o más de los modelos anteriores para realizar sus predicciones. En esta categoría, como se verá en el siguiente punto, entran los modelos propuestos por la ITU-T P.1203 [68] en sus modos 2 y 3.

3.2.2.4. ITU-T P.1203

La ITU-T P.1203 [68] describe un modelo para la evaluación objetiva de la calidad percibida por los usuarios de un servicio de streaming de contenido audiovisual. El estándar ITU-T P.1203 ha sido desarrollado específicamente para los sistemas de streaming tipo HAS. La propuesta tiene cuatro modos de funcionamiento que hacen uso de menos a más cantidad de información del flujo multimedia en función del ordinal de sus nombres. Los modos definidos son:

- **Modo 0**: Utiliza información de los metadatos (archivos de manifiesto de DASH) como el codec y el bitrate de las diferentes representaciones, así como el retardo inicial y la información sobre las paradas producidas durante la reproducción
- **Modo 1**: Hace uso de la información utilizada en el Modo 0 con la adición de información de los flujos multimedia obtenida de la inspección de la cabecera de los paquetes.
- **Modo 2**: Hace uso de la información utilizada en el Modo 1 más un 2% de los bytes del total de la transmisión que es utilizada para un análisis gramatical parcial.
- **Modo 3**: Hace uso de la información utilizada en el Modo 2 y de la información de los flujos multimedia basada en el análisis gramatical del total del bitstream.

Según la clasificación de Takahashi et al. [53], los modos de funcionamiento 0 y 1 son modelos paramétricos mientras que los modos 2 y 3 podrían clasificarse tanto como

paramétricos o como basados en el bitstream; por lo que su clasificación es de modelos híbridos.

Como se ha mencionado con anterioridad, la ITU-T P.1203 depende de los codecs utilizados en la codificación del contenido y el único soportado para el contenido visual es H.264. Es por esto por lo que, como se verá en la evaluación, para poder hacer uso del estándar a través de la herramienta "ITU-T Rec. P.1203 Standalone Implementation" [69] se ha utilizado la extensión desarrollada por el grupo de GitHub "Telecommunication-Telemedia-Assesment" [70] que da soporte para el codec VP9; aunque únicamente en Modo 0.

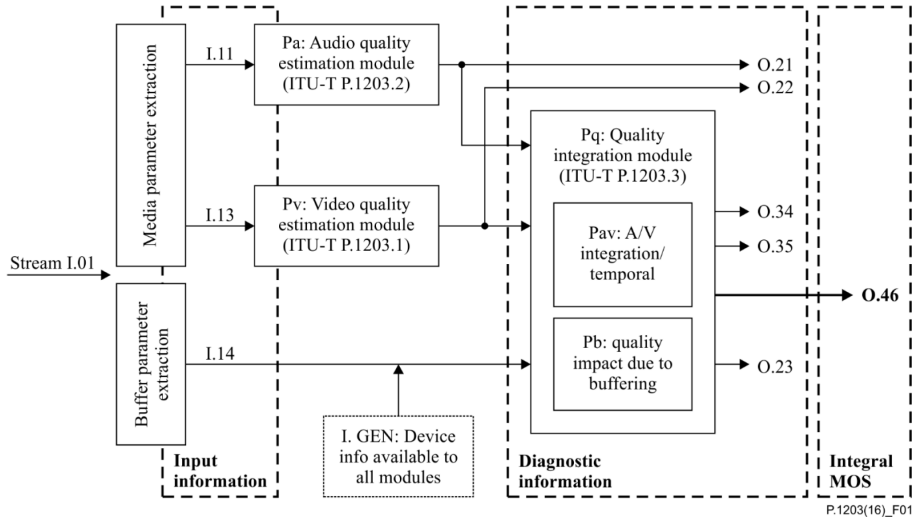


Figura 3.2. Diagrama de los diferentes módulos y componentes definidos en la ITU-P.1203. Fuente: *ITU-T P.1203 [68]*.

En la Figura 3.2 se pueden observar los diferentes módulos por los que está formado el modelo. Por un lado, los módulos Pa, Pv e I son los encargados de generar información relativa a la calidad de los flujos de audio, de vídeo e información relativa al dispositivo con independencia de las vicisitudes del proceso de streaming, respectivamente. Y por el otro, el módulo Pq es el que integra las informaciones generadas por los anteriores módulos con los datos relativos al proceso de streaming HAS como son los eventos de parada por vaciamiento del buffer y los cambios producidos en las calidades de los flujos. Cada uno de los módulos mencionados están definidos en sendos documentos de la ITU tal como aparece en la propia Figura 3.2.

3.2.2.5. ITU-T P.1204

Como se ha visto en el punto anterior, la ITU-T P.1203 adolece de una falta de versatilidad debido al soporte de un único codec de vídeo. Esta situación se corrige en la nueva recomendación que la ITU ha desarrollado y publicado: la ITU-T P.1204 [71]. Esta

recomendación propone un nuevo módulo de estimación de la calidad de los flujos de vídeo (Pv) de forma que pueda ser integrado en el modelo propuesto por la recomendación ITU-T P.1203 tal y como se muestra en la Figura 3.3.

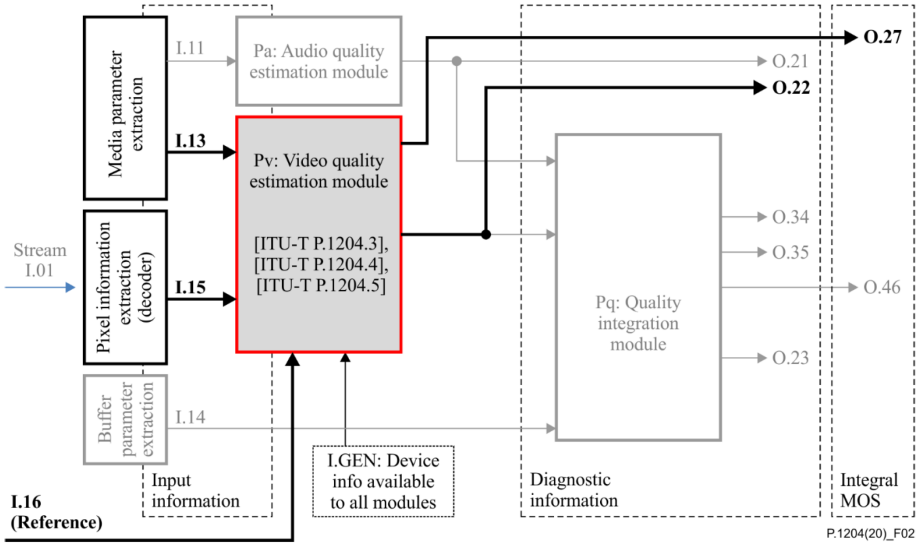


Figura 3.3. Diagrama de integración del módulo Pv de la ITU-T P.1204 en la arquitectura propuesta por la ITU-T P.1203. Fuente: *ITU-T P.1204 [71]*.

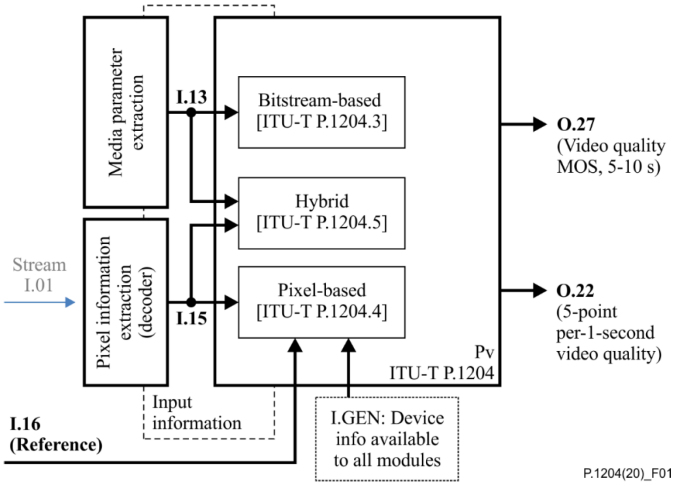


Figura 3.4. Detalle del modelo de estimación de la calidad de vídeo propuesto en la ITU-T P.204. Fuente: *ITU-T P.1204 [71]*

En la Figura 3.4 se puede ver el detalle del módulo de VQA especificado en la recomendación. En particular, se puede observar cómo existen tres especificaciones distintas para la estimación, que tratan de cubrir escenarios en los que se tienen diferentes niveles de acceso a los flujos de vídeo.

Atendiendo a los documentos de los que consta esta recomendación, vemos en la Figura 3.4 que no aparecen mencionados los documentos ITU-T P.1204.1 ni ITU-T P.1204.2 debido a que, en el momento de escribir esta tesis, aún están en proceso de desarrollo. Estos documentos definirán métodos de VQA que se basarán en:

- **ITU-T P.1204.1**: La información de la capa de transporte de la sesión de streaming.
- **ITU-T P.1204.2**: La información de las cabeceras de los frames de vídeo. Equivalente al Modo 0 de la ITU-T P.1203.

Por su parte, los modelos de VQA que sí que están publicados en estos momentos son:

- **ITU-T P.1204.3**: Hace uso de la información del bitstream del flujo de vídeo sin necesidad de decodificarlo. Equivalente al Modo 1 de la ITU-T P.1203.
- **ITU-T P.1204.4**: Hace uso de la información de los píxeles del flujo de vídeo, así como de la información de píxel del vídeo original. Este modelo es por lo tanto RR/FR.
- **ITU-T P.1204.5**: Hace uso tanto de la información del bitstream como de la información del vídeo decodificado y su original.

Esta recomendación, la ITU-P.1204, tiene una fecha de publicación (enero 2020) posterior a la de la realización de las evaluaciones incluidas en esta tesis, por lo que no ha sido posible utilizarla para su comparación con los modelos propuestos.

3.2.2.6. Modelo QoE normalizado

Como se ha visto en puntos anteriores, existe una gran variedad de modelos para la estimación de la QoE que cubren un amplio espectro de escenarios. En este sentido, las propuestas presentadas en esta tesis se basan en el modelo de Yin et al. [72] puesto que es un modelo sencillo, de aplicación directa y altamente citado en la literatura.

En el trabajo presentado por Yin et al. en [72], se plantea valorar los diferentes tipos de algoritmos de adaptación (basados en el buffer o en la predicción de la tasa de transferencia) en su conjunto utilizando un entorno de simulación. Para ello se basa en lo que denominan estrategias típicas de los tipos de algoritmos. Por ejemplo, en el caso de los algoritmos basados en la tasa de transferencia considera que se elige siempre la representación que mejor se ajusta a dicha tasa.

Por este motivo, entre otros, la presente tesis dista mucho de coincidir con los resultados presentados en [72]. Sin embargo, el modelo propuesto por Yin et al. para evaluar la QoE, como se explica a continuación, es simple y de fácil aplicación, por lo que ha sido utilizado como base en los modelos de calidad de experiencia propuestos en esta tesis.

Yin et al. proponen una fórmula en la que la QoE se calcula como la suma del QoE de cada segmento. De esta forma, Yin et al. definen la calidad de la experiencia en la reproducción de los segmentos desde 1 hasta *K* como la suma ponderada de tres componentes: la calidad del vídeo, la variación de la calidad del vídeo y el tiempo total en estado de parada por vaciamiento del buffer como se muestra en (3.1).

$$QoE_1^K = \sum_{k=1}^{K} q(R_k) - \lambda \sum_{k=1}^{K-1} |q(R_{k+1}) - q(R_k)| - \mu \sum_{k=1}^{K} \left(\frac{LR_k}{C_k} - B_k\right),\ R_k \in \Re\,, \quad (3.1)$$

donde *K* es el número de segmentos del vídeo, $R_k \in \Re$ (donde $\Re$ es el conjunto de los bitrates medios de las representaciones disponibles para el contenido) es el bitrate medio de la representación seleccionada para el segmento *k*, *q(·)* es una función creciente que mapea el bitrate seleccionado R_k a la calidad percibida por el usuario $q(R_k)$, *L* es la duración en segundos de cada segmento, C_k es velocidad de descarga para el segmento *k*, B_k es la ocupación del buffer de reproducción en el instante de tiempo en el que el segmento *k* está siendo descargado y, por último, λ y μ son valores positivos que ponderan el peso de la calidad del vídeo y el tiempo en el estado de parada debido a un vaciamiento en el buffer de reproducción, respectivamente.

En relación con los parámetros de balanceo del peso de la predicción, una λ pequeña implica que el usuario no es particularmente susceptible a la variación de la calidad del vídeo, mientras que una μ grande indica que el usuario reacciona muy negativamente ante las paradas en la reproducción. Debido a que las interrupciones en la reproducción alteran la experiencia mucho más que los cambios de calidad [73]-[78], el valor de μ suele ser mucho mayor que el de λ.

Yin et al. definen un modelo de QoE normalizado para comparar el rendimiento de los algoritmos de adaptación al máximo teórico calculado suponiendo que la tasa de transferencia disponible es conocida tal y como se muestra en (3.2).

$$nQoE_1^K = \frac{QoE_1^K}{QoE_{opt}}. \quad (3.2)$$

3.3. Modelos de QoE propuestos

En esta sección se presentan los diferentes modelos objetivos de estimación de la QoE que se proponen en este trabajo de investigación.

3.3.1. Modelo QoE modificado

Inicialmente, se propone una modificación del modelo de QoE normalizado de Yin et al. La propuesta se muestra en (3.3):

$$QoE_1^K = \sum_{k=1}^{K} q(R_k) - \lambda \sum_{k=1}^{K-1} |q(R_{k+1}) - q(R_k)| - \mu \sum_{k=1}^{K} \left(\frac{LR_k}{C_k} - B_k\right),\ R_k \in \Re_S. \quad (3.3)$$

Aunque parece la misma fórmula que (3.1), contiene una diferencia significativa. En (3.1) $R_k \in \Re$, por lo que el valor del bitrate es el mismo para todos los segmentos de una misma representación, mientras que en la fórmula propuesta en (3.3) $R_k \in \Re_S$, donde $\Re_S$ contiene

un conjunto de valores diferente para cada segmento. De esta forma, R_k no pertenece al conjunto de bitrates medios disponibles en la MPD del estándar de DASH, ya que el bitrate de cada representación, generalmente, cambia para cada segmento. Por ejemplo, en un vídeo con una única representación cuyo bitrate medio sea de 500 kbps, el valor de $\Re$ será de 500 kbps para todos sus segmentos, mientras que $\Re_S$ puede tener valores diferentes que difieran más o menos del bitrate medio (e.g., 481, 497, 504 kbps…). Otro ejemplo lo podemos encontrar en la Figura 1.20 donde se puede comparar el tamaño de los segmentos de un mismo vídeo codificado tanto en modo de calidad constante como en modo de bitrate constante.

Tener en consideración el bitrate específico de cada segmento en lugar de usar la media de todos los que forman una representación hace que esta propuesta basada en el modelo de QoE de Yin et al. sea más precisa. Especialmente si, como se lleva a cabo en este trabajo de investigación, se aplica a contenidos codificados con calidad constante.

3.3.2. Modelo QoE basado en PSNR

El PSNR es una métrica de VQA que tiene una probada relación con el QoE: cuando el PSNR aumenta, la QoE al visualizar un contenido también aumenta [79]. Debido a que el bitrate y el PSNR tienen una relación logarítmica creciente, las variaciones en el bitrate no tienen incrementos proporcionales y estos dependen del valor del bitrate. Por ejemplo, una pequeña variación en el bitrate suele implicar una variación alta del PSNR para bitrates bajos [80]. En la Figura 3.5 se muestra la curva de PSNR en función del bitrate medio del vídeo Big Buck Bunny codificado con VP9 donde se ve como afecta una variación en el bitrate medio de 500 Kbps dependiendo de la parte de la tabla. En particular, para este caso concreto, un aumento de 500 Kbps a 1000 Kbps en el bitrate medio supone un aumento de 3,3 dB mientras que un aumento de la misma cuantía de 4000 a 4500 Kbps aumenta el PSNR en 0,3 dB.

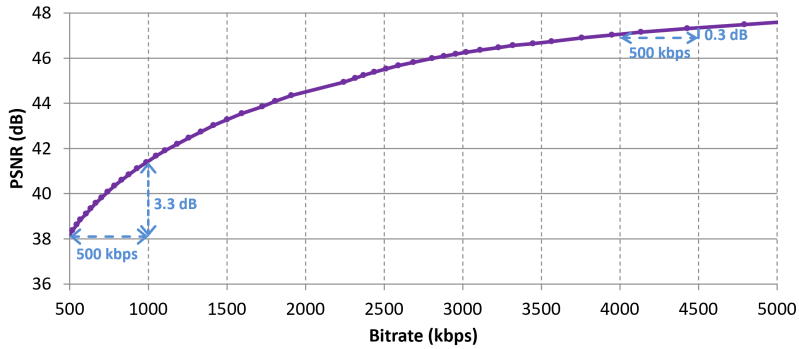


Figura 3.5. Curva del PSNR en función del bitrate para el vídeo Big Buck Bunny codificado con VP9.

Tanto el PSNR como el bitrate son medidas objetivas importantes, pero el PSNR ofrece una relación más directa en lo que respecta a la QoE. Además, la falta de linealidad entre el bitrate y el PSNR puede hacer pensar que el modelo propuesto por Yin et al. puede ser

mejorado. Bien es cierto que el modelo de Yin et al. está basado en una función creciente *q(·)* que afecta al bitrate (por tanto, podría ser tanto una función lineal como logarítmica), pero dicha función no está especificada en su propuesta [72].

En este sentido, esta tesis propone un nuevo modelo de QoE basado en un parámetro tan importante como lo es el PSNR, tal como se muestra en (3.4).

$$QoE'_{PSNR} = \frac{1}{K}\sum_{k=1}^{K} PSNR(\xi_k) - \zeta \frac{1}{K-1}\sum_{k=1}^{K-1}|PSNR(\xi_{k+1}) - PSNR(\xi_k)| - \eta \cdot 10\log_{10}\left(1 + \frac{1}{d}\sum_{k=1}^{K}\left[\frac{LR_k}{C_k} - B_k\right]\right) - \delta \cdot 10\log_{10}(1 + T_s),\ R_k \in \Re_S, \quad (3.4)$$

donde K es el número de segmentos del vídeo, ξ_k es la representación seleccionada para el segmento k, $PSNR(\xi_k)$ es el PSNR de la representación seleccionada del segmento k, d es la duración total del vídeo en segundos, L es la duración en segundos de cada segmento, $R_k \in \Re_S$ es el bitrate de la representación seleccionada del segmento k, C_k es la tasa de transferencia efectiva en la obtención del segmento k, B_k es la ocupación del buffer de reproducción cuando se inicia la transferencia del segmento k, T_s es el retardo en segundos del inicio de la reproducción y, por último, ζ, η y δ son parámetros positivos para la ponderación del peso de los cambios de representación, el tiempo en estado de parada debido al vaciamiento del buffer y el retraso en el inicio de la reproducción, respectivamente. Como en el caso del modelo QoE modificado, R_k no pertenece al conjunto de bitrates medios disponibles en la MPD.

Con el fin de establecer un límite al valor mínimo para el caso en el que se produzcan un elevado número de paradas, así como para evitar que el modelo genere valores negativos, definimos QoE_{PSNR} como:

$$QoE_{PSNR} = \max(QoE'_{PSNR}, 0). \quad (3.5)$$

La idea en la que se basa este modelo es la misma que la de los dos modelos presentados. El valor máximo del modelo de QoE sería el valor del propio VQA, el PSNR del vídeo reproducido. Este valor será disminuido por: 1) los cambios en el PSNR de los segmentos; 2) las paradas durante la reproducción debidas al vaciamiento del buffer; y 3) el retardo en el inicio de la reproducción.

Conceptualmente (3.4) puede representarse como se muestra en (3.6), donde la ratio de parada (*stalling ratio*) corresponde al tiempo de parada debido al vaciamiento del buffer dividido entre el tiempo total de la reproducción.

$$QoE'_{PSNR} = Average\ PSNR - \zeta * Average\ PSNR\ switches - \eta \cdot 10\,log_{10}(1 + stalling\ ratio) - \delta \cdot 10\,log_{10}(1 + startup\ delay). \quad (3.6)$$

De esta forma, cuando el tiempo de parada debido al vaciamiento del buffer es cero, el tercer elemento de (3.6) también lo será, por lo que no habrá penalización alguna debida a este componente.

Es importante señalar que, a diferencia de la fórmula propuesta por Yin et al., el modelo propuesto en este punto tiene unidades. En particular, las unidades son dB ya que los cuatro elementos de la suma están expresados en dB.

El valor máximo del modelo QoE_{PSNR} es la media del PSNR de los segmentos que lo conforman. Así, en el caso ideal de una reproducción de contenidos sin paradas, sin cambios de representación ni retardo en el inicio de la reproducción, el valor de QoE_{PSNR} será el del PSNR de la representación reproducida menos la variación del PSNR de sus segmentos promediada con el número de éstos. Hay que tener en cuenta que los diferentes segmentos de una misma representación pueden tener valores de PSNR diferentes. Por otro lado, valores de QoE_{PSNR} cercanos a cero, producidos principalmente por un alto porcentaje de tiempo de reproducción detenido por vaciamiento del buffer, indica una pobre, incluso inaceptable, experiencia del usuario.

Por lo tanto, el valor del modelo QoE_{PSNR} propuesto está acotado con un máximo correspondiente al valor del PSNR de la representación de mayor calidad (por lo que depende del vídeo reproducido) y con un mínimo de cero. Por consiguiente, el modelo QoE_{PSNR} propuesto puede proveer información de la calidad de experiencia por sí mismo sin necesidad de ser comparado con otros valores, mientras que el propuesto por Yin et al. necesita ser comparado con un caso ideal. Es importante remarcar que, del mismo modo que el PSNR es dependiente del contenido, también lo es el modelo QoE_{PSNR}.

La principal dificultad de aplicar la fórmula propuesta es calcular el valor de PSNR de todos los segmentos de todas las representaciones. Este proceso puede suponer un coste en tiempo y en recursos computacionales importante que crece con el número de representaciones, así como con la duración del contenido. Con el objetivo de simplificar este procedimiento, como resultado de este trabajo de investigación se ha desarrollado y hecho público el código fuente del programa "dashgen" en la plataforma de repositorios GitHub [43]. Este programa simplifica el proceso de codificación de las representaciones y obtención del valor de PSNR para cada uno se los segmentos de las representaciones.

Para comprobar cómo el parámetro η afecta al QoE_{PSNR}, la Figura 3.6 muestra el QoE_{PSNR} para diferentes valores de porcentaje de tiempo de parada respecto a la duración total del contenido y diferentes valores de η. En la figura, el parámetro de PSNR medio se ha fijado a 44 dB, la variación de PSNR media se ha establecido a 4 dB, ζ=1, y δ=0. Con esta configuración, en el caso de que no haya paradas, el QoE_{PSNR} obtenido sería 40 dB, tal y como se muestra en la propia figura. La figura muestra como el parámetro η tiene un impacto significativo en el QoE_{PSNR}. Por ejemplo, cuando η=5, si el porcentaje de tiempo en parada es del 3% se obtiene un QoE_{PSNR} muy pobre (10 dB). En contraste, cuando η=2 el mismo porcentaje de tiempo en parada supone un QoE_{PSNR} aceptable de 28 dB.

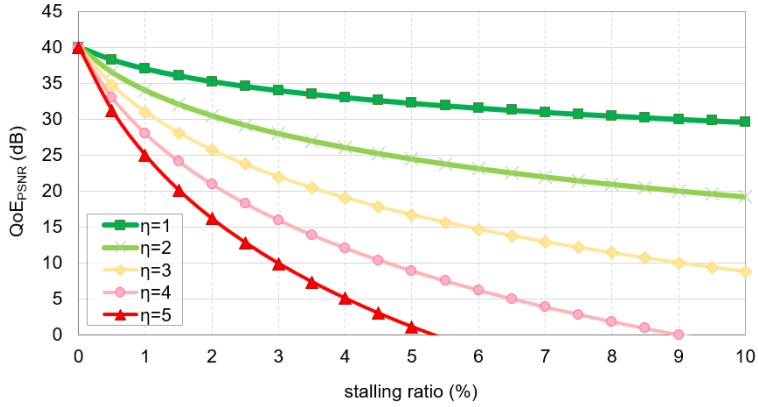


Figura 3.6. QoE$_{PSNR}$(dB) para diferentes valores de la ratio de parada debida al vaciamiento del buffer y η.

3.3.3. Modelo QoE basado en VMAF

VMAF [14]-[16] es un método de VQA desarrollado por Netflix y usado por un gran número de herramientas de vídeo como FFmpeg y Elecard StreamEye. VMAF utiliza como entrada los valores de las métricas *Visual Information Fidelity* (VIF) [81], *Detail Loss Metric* (DLM) y *Temporal Impairment Feature* (TI) fusionadas a través de una regresión *Support Vector Machine* (SVM) [82] con un modelo de *machine-learning* incorporado en la propia métrica de VMAF. El modelo utilizado por VMAF ha sido entrenado mediante medidas subjetivas realizadas para el desarrollo de la propia métrica.

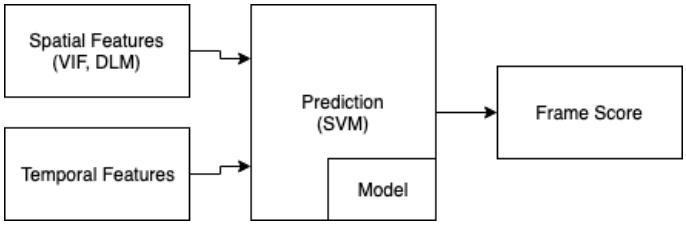


Figura 3.7. Esquema del sistema VMAF.

En lo referente al uso de VIF, VMAF, en vez de utilizar el valor de VIF, hace uso de las cuatro escalas internas que esta métrica genera, mientras que el valor de VIF sería la combinación estas cuatro escalas en un único valor. Así, el modelo SVM hace uso de las seis características como entrada (las 4 de VIF, DLM y TI) para generar un valor por cada frame de vídeo. El valor final de VMAF es la media aritmética de los valores individuales de los frames que componen el vídeo.

Al igual que ocurre con el PSNR, los valores de VMAF obtenidos para las diferentes calidades de un vídeo codificado no tienen una relación lineal con el bitrate de las propias representaciones. En la Figura 3.8 muestra los distintos valores de VMAF obtenidos para las diferentes bitrates medios del vídeo Elephants Dream codificaco con VP9. En la figura

se puede ver como un incremento de 500 a 1000 Kbps supone un aumento del VMAF de 12 mientras que aumentar de 4000 a 4500 Kbps incrementa el valor de VMAF resultante en 0,4.

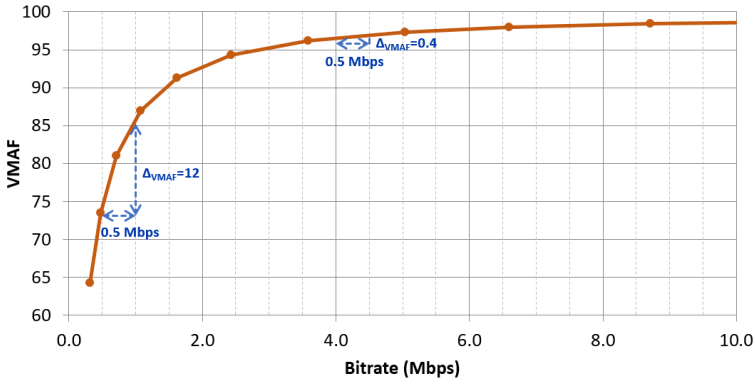


Figura 3.8. Curva del VMAF en función del bitrate para el vídeo Elephants Dream codificado con VP).

Tomando como referencia el sistema VMAF, en esta tesis se presenta un modelo de QoE basado en VMAF, tal como se muestra en (3.7):

$$QoE'_{VMAF} = \frac{1}{K}\sum_{k=1}^{K} VMAF(\xi_k) - \beta \frac{1}{K-1}\sum_{k=1}^{K-1}|VMAF(\xi_{k+1}) - VMAF(\xi_k)| - \gamma \cdot \frac{1}{d}\sum_{k=1}^{K}\left[\frac{LR_k}{C_k} - B_k\right] - \delta \cdot T_s,\ R_k \in \Re_S, \quad (3.7)$$

donde *VMAF(ξk)* es el valor de VMAF del segmento *k* de la representación seleccionada, β y γ son parámetros con valores positivos que ponderan el peso de las variaciones de la calidad de los segmentos y el tiempo de parada debido al vaciamiento del buffer, respectivamente. El resto de los parámetros son los mismos que los encontrados en la Ecuación 3.4.

Es importante resaltar que el modelo QoE$_{VMAF}$, al igual que el modelo QoE$_{PSNR}$, puede proporcionar información por sí mismo acerca de la calidad de la experiencia en la reproducción de un vídeo. QoE$_{VMAF}$ tiene la misma escala que la propia métrica de VMAF, esto es, el valor máximo es 100 (una excelente QoE) y el valor mínimo es 0 (una QoE pésima); por tanto, establecemos un límite al valor mínimo de QoE$_{VMAF}$ de la siguiente forma:

$$QoE_{VMAF} = \max(QoE'_{VMAF}, 0). \quad (3.8)$$

Conceptualmente, (3.7) puede expresarse como se muestra en la Ecuación 3.9:

$$QoE'_{VMAF} = Average\ VMAF - \beta * Average\ VMAF\ switches - \gamma \cdot stalling\ ratio - \delta \cdot start\_up\ delay. \quad (3.9)$$

También en este caso, la principal problemática para hacer uso del modelo es el cálculo

del valor de VMAF para todos los segmentos de todas las representaciones. De la misma forma que para el modelo QoE_{PSNR}, la herramienta desarrollada en el contexto de esta tesis y publicada en GitHub [43] incluye la funcionalidad de calcular todos los valores de VMAF necesarios para el cálculo del modelo propuesto.

En la misma línea que el ejemplo mostrado en la Figura 3.6 y haciendo uso de la Ecuación 3.7, la Figura 3.9 muestra el valor de QoE_{VMAF} para diferentes valores de ratio de parada y de γ. En la figura, el parámetro VMAF del vídeo se ha fijado a 95, la variación media entre el VMAF de los segmentos consecutivos se ha fijado a 5, β= 1 y δ=0 (el tiempo de retardo en el inicio de la reproducción no se tiene en consideración). Con esta configuración, en el caso de no producirse paradas el valor de QoE_{VMAF} resultante es de 90. La Figura 3.9 muestra como el parámetro γ tiene una gran repercusión sobre el valor del modelo QoE_{VMAF}. Como ejemplo, cuando γ=1800, si la ratio de parada de reproducción es del 4 % de la duración del vídeo, obtendríamos un valor muy pobre: QoE_{VMAF}= 18. Como contraste, cuando γ=600, la misma ratio de tiempo de parada supone un valor aceptable QoE_{VMAF}=66.

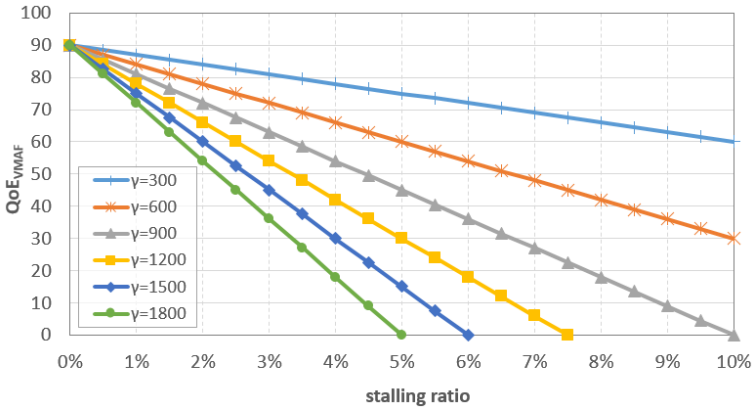


Figura 3.9. QoE_{VMAF} para diferentes valores de la ratio de parada debida al vaciamiento del buffer y γ.

3.4. Metodología

Para llevar a cabo la evaluación de los diferentes modelos de QoE se han seleccionado los algoritmos de adaptación Müller [18] y SARA [22] así como el algoritmo propuesto Look Ahead, presentado en el capítulo anterior. A diferencia del Capítulo 2, no se ha tenido en consideración el algoritmo de adaptación incluido en la propia librería (ExoPlayer) por su pobre rendimiento en comparación con el resto.

Como también se ha comentado en el capítulo anterior, los algoritmos seleccionados cumplen las condiciones, a diferencia de la mayoría de las propuestas de algoritmos de adaptación para DASH, de proveer suficiente información en su especificación para

poder ser implementado y de proponer un comportamiento compatible con su implementación para la librería de reproducción ExoPlayer.

Los algoritmos mencionados han sido desarrollados para su integración con la librería ExoPlayer v2, la librería desarrollada por Google para la reproducción de contenido DASH. La utilización de un player real, en lugar de utilizar simulaciones, tiene ventajas significativas a la hora de obtener información precisa de la reproducción de los contenidos. Por ejemplo, al usar una implementación real en vez de una simulación de reproducción, el estado del buffer se actualiza nada más un frame es analizado gramaticalmente desde la conexión HTTP y no únicamente al finalizar completamente la transmisión de un segmento. Las simulaciones que no analizan el contenido de los segmentos durante el proceso de descarga no son capaces de detectar fehacientemente las paradas en la reproducción debidas al vaciamiento del buffer ya que podría encontrarlas donde, en realidad, no las hubiera.

A pesar de que en la literatura existen un buen número propuestas de algoritmos ABR (se puede encontrar un completo estudio en [31]), no se ha podido adaptar algunos de los algoritmos más populares como el algoritmo BOLA [19] implementado en el player web "dash.js" como ya se mencionara en la sección 2.2.2.4.

Los algoritmos de adaptación considerados en esta tesis han sido testeados en diferentes escenarios: 4 canales con una tasa de transferencia constante (1, 2, 5 y 10 Mbps) y 4 canales con una tasa de transferencia variable. En particular, el primer canal variable cambia entre 2, 4, 8 y 4 Mbps en bucle cada 100 s; el segundo cambia entre 2 y 8 Mbps cada 100 s; mientras que los dos últimos son dos canales obtenidos de medidas de campo llevadas a cabo por la Ghent University, específicamente un coche y un autobús en movimiento [38], al igual que se utilizaron en la sección 2.4.

3.4.1. Evaluación objetiva

En la evaluación objetiva se hace uso de los principales factores de influencia que afectan a la QoE (número y duración de las paradas, ratio de paradas y número de cambios de representación/calidad) para calcular la QoE de los modelos propuestos.

En lo que respecta a los modelos basados en el bitrate, hemos fijado λ=1 y μ=6000, lo que significa que un segundo en estado de parada tiene la misma penalización que un cambio de representación con un decremento del bitrate de 6000 kbps. Estos valores se sugieren en [83] mientras que [72] propone μ=3000, un valor más permisivo con las paradas de reproducción. Debido a que en los trabajos de Yin et al. no hay detalles sobre la función $q(\cdot)$, por simplicidad se ha considerado $q(x)=x$, ya que cumple con el único requisito de ser una función incremental. Para el modelo de QoE basado en PSNR, se ha establecido ζ=1 y η=3, mientras que para el modelo basado en VMAF se ha establecido β=1 y γ=900. Los valores utilizados para β y γ están sugeridos en [84] donde se justifica que una ratio de parada del 1 % se considera ligeramente molesto para los usuarios mientras que una ratio del 10 % no se considera aceptable.

Analizando la Figura 3.6, para η=3, cuando la ratio de parada es del 10% el valor del modelo QoE_{PSNR} es menor de 10 dB, el cual es considerado inaceptable para los usuarios. Siguiendo el mismo criterio, en la Figura 3.9 se muestra como el valor de γ que más se aproxima a la condición anterior es γ=900, ya que de esta forma el valor del modelo QoE_{VMAF} para una ratio de parada del 10% es QoE_{VMAF}=0. Sin embargo, en este capítulo se incluye la evaluación del rendimiento de los modelos QoE_{PSNR} y QoE_{VMAF} para diferentes valores de η y γ respectivamente.

Adicionalmente, y con el objetivo de hacer una comparación precisa con respecto al modelo de QoE presentado por Yin et al., en la evaluación no se ha considerado el retardo inicial de la reproducción puesto que Yin et al. no lo tiene en cuenta. Por lo tanto, δ=0.

El modelo de QoE propuesto por la ITU-T P.1203 [68] también ha sido utilizado para estimar la QoE de las evaluaciones. Esta recomendación describe un conjunto de módulos con métodos paramétricos para la valoración objetiva de la calidad en una sesión de streaming de contenido multimedia. Esta recomendación proporciona predicciones sobre el impacto de la codificación y la transmisión sobre redes IP. Este estándar ha sido desarrollado específicamente para la evaluación de sistemas de streaming tipo HAS. La ITU-T P.1203 tiene en cuenta tanto el retraso en la inicialización como las paradas producidas por vaciamiento del buffer mientras que la calidad del vídeo es evaluada con algoritmos NR. La recomendación incluye cuatro modos diferentes de utilización con sendos niveles de complejidad. Puesto que los vídeos utilizados en este trabajo de investigación están codificados con el codec VP9 y la propuesta de la ITU está diseñada específicamente para ser utilizada con el codec H.264, se ha hecho uso de la implementación disponible en [69] con las extensiones para VP9 disponibles en [70]. Debido a las limitaciones de la extensión para VP9, la evaluación ha sido llevada a cabo utilizando la ITU-T P.1203 en modo 0 puesto que no existe, actualmente, una implementación para VP9 que soporte los niveles 1, 2 o 3.

En la evaluación, los vídeos utilizados han sido codificados utilizando VP9, un codec de vídeo desarrollado por Google, publicado con licencia de código libre y libre de patentes. Debido a estos motivos, el codec de vídeo VP9 es, hoy en día, uno de los más utilizados para la transmisión de vídeo por Internet.

Se han utilizado tres vídeos para realizar la evaluación producidos por la Blender Foundation [37]: “Elephants Dream”, “Tears of Steel” y un vídeo de mayor duración al que denominamos “Mix” (al igual que en la sección 2.4) que es el resultado de concatenar los dos vídeos anteriores junto con los vídeos “Sintel” y “Big Buck Bunny”.

Tanto la codificación de los vídeos como el entorno de ejecución son idénticos a los descritos en la sección 2.4.

Finalmente, para la obtención de los datos mostrados en la sección de evaluación de este capítulo se han realizado 5 iteraciones para cada combinación de algoritmo, canal de transmisión y vídeo evaluado. De esta forma, se han obtenido intervalos de confianza estrechos. En particular, se han efectuado un total de 98 horas de reproducción de

contenido para la generación de las evaluaciones objetivas que han generado una cantidad de datos superior a los 40 GB.

3.4.2. Evaluación subjetiva

La evaluación subjetiva ha sido desarrollada teniendo en cuenta las recomendaciones del estándar ITU-T P.913 [46]. Para esta evaluación el único canal de transmisión utilizado ha sido el 4G-car ya que, como se verá, es el canal más exigente.

La evaluación subjetiva ha sido llevada a cabo en un laboratorio de la Universitat Politècnica de València a lo largo de 3 semanas en sesiones de 1 hora de duración. Un máximo de 4 personas participó en las sesiones para optimizar el tiempo de evaluación. Los usuarios no interactuaron entre sí puesto que, en todo momento, había dos coordinadores encargados de asegurar el correcto desarrollo de las pruebas. Un total de 24 personas (15 hombres y 9 mujeres) participaron en el estudio, con un rango de edades entre 20 y 42 años.

En las sesiones de evaluación el contenido se mostró a los usuarios en un ordenador Apple iMac (modelo A1225) con una resolución de 1920x1200 y una relación de aspecto de 16:10. La configuración del color y brillo de la pantalla se configuró según las recomendaciones del fabricante. La luminosidad de la sala era de 25 lux y la distancia de la pantalla a los usuarios se estableció a tres veces la altura de la propia pantalla (3H).

Para la evaluación subjetiva se utilizaron, nuevamente, los vídeos “Elephants Dream” y “Tears of Steel”. Es importante resaltar que la duración de los vídeos (entre 10 y 12 minutos) permite evaluar la QoE percibida por los usuarios de forma apropiada. De hecho, es precisamente la duración de los vídeos evaluados uno de los puntos fuertes de la propuesta puesto que la mayoría de los estudios similares se desarrollan con vídeos de corta duración.

De la misma forma que en la evaluación objetiva, se han usado tamaños de segmento de 10 segundos codificados con VP9 a una resolución de 1920x1080 y con un frame rate de 24 fps. Por su parte, los niveles de calidad utilizados también son los mismos que para la evaluación objetiva. Esto es, 12 calidades utilizando el parámetro CRF para la codificación desde 5 hasta 60. Como se detalla en la evaluación, en este caso únicamente se han considerado dos algoritmos de adaptación (Look Ahead y Müller) produciendo, así, cuatro casos de estudio: A (el vídeo “Elephants Dream” y el algoritmo Look Ahead), B (el vídeo “Elephants Dream” y el algoritmo Müller), C (el vídeo “Tears of Steel” y el algoritmo Look Ahead) y D (el vídeo “Tears of Steel” y el algoritmo Müller). Los detalles de los casos de prueba se pueden observar en la Tabla 3.2.

Tabla 3.2. Características de los vídeos utilizados en la evaluación subjetiva.

Caso	Vídeo	Algoritmo	Duración (s)	Nº de Segmentos	Resolución	Frame rate (fps)	Codec
A	Elephants Dream	Look Ahead	654	66	1920x1080	24	VP9
B	Elephants Dream	Müller	654	66	1920x1080	24	VP9
C	Tears of Steel	Look Ahead	734	74	1920x1080	24	VP9
D	Tears of Steel	Müller	734	74	1920x1080	24	VP9

Como se ha mencionado, existen diferentes escalas para la medición del MOS. Entre ellas, la más usada es la ACR (*Absolute Category Rating*) de 5 puntos: 5 (Excelente), 4 (Buena), 3 (Correcta), 2 (Pobre) y 1 (Mala). Esta métrica es la utilizada en la primera parte de las sesiones de evaluación. En la segunda, cuando se comparan los dos algoritmos, se utiliza la escala CCR (*Comparison Category Rating*) en la cual el contenido es comparado con un contenido previo en conformidad con la siguiente escala: 3 (mucho mejor), 2 (mejor), 1 (ligeramente mejor), 0 (igual), -1 (ligeramente peor), -2 (peor) y -3 (mucho peor).

Cada sesión se dividió en dos partes de 30 minutos aproximadamente con un receso de 5 minutos. En cada una de las partes, los participantes evaluaron uno de los dos vídeos evaluados (“Elephants Dream” o “Tears of Steel”). En cada parte se llevó a cabo dos evaluaciones diferentes: 1) la evaluación de cada algoritmo por separado (test ACR); y 2) la evaluación de la calidad visual de un algoritmo respecto al otro (test CCR). Es interesante destacar que el orden de los vídeos y de los algoritmos fue aleatorizado para cada sujeto manteniendo los casos con el mismo vídeo de forma consecutiva.

La Figura 3.10 muestra un ejemplo de secuencia seguida para la evaluación subjetiva. Al principio de la evaluación se realizó una prueba de agudeza visual para comprobar la validez de los resultados proporcionados por cada usuario. Finalmente, al completar la prueba, a cada usuario se le pidió que ordenase del mayor al menor factor de influencia en su evaluación los siguientes aspectos: paradas en la reproducción, retardo en el inicio de la reproducción, calidad de los vídeos mostrados y cambios en la calidad durante la reproducción.

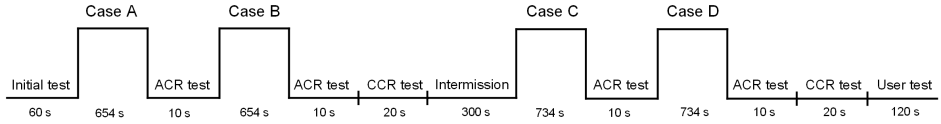


Figura 3.10. Secuencia de ejemplo de la evaluación subjetiva.

El estudio subjetivo ayuda a evaluar el rendimiento de los mecanismos de QoE propuestos. Para ello, es importante establecer una relación entre el MOS y la QoE de las propuestas. En este sentido, se pueden encontrar trabajos en la literatura que estudian la relación entre el MOS y métricas objetivas como PSNR, SSIM, VQM (*Video Quality Model*) o VMAF. La Figura 3.11 muestra la relación entre el MOS y el PSNR extraído de un estudio llevado a cabo por la University of Waterloo [85] usando diferentes dispositivos y tamaño de pantalla. En la mencionada figura, el eje Y puede ser trasladado a la escala ACR considerando que el valor más alto (100) corresponde a ACR=5 y el valor más bajo (0) corresponde a ACR==1. En la figura se puede observar una nube de puntos correspondiente a todos los tamaños de pantalla utilizados en el estudio, pero, en general, el MOS incrementa cuando así lo hace el PSNR. Por otro lado, entre los estudios realizados por Netflix con relación a VMAF, [86] presenta una correspondencia entre la escala ACR y la VMAF, tal como se muestra en la Figura 3.12. En la gráfica se puede observar que, en general, se obtiene una QoE razonable cuando el valor obtenido para VMAF es mayor de 60, una buena QoE para valores de VMAF mayores de 80 y una QoE excelente para valores de VMAF cercanos a 100.

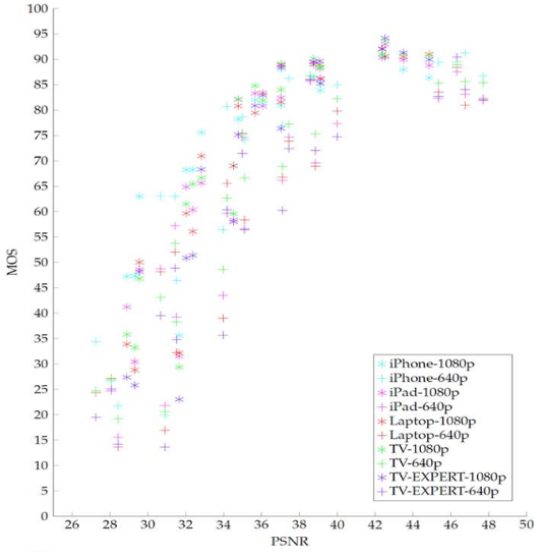


Figura 3.11. Relación entre el MOS y el PSNR. Fuente: *The SSIMplus Index for Video Quality-of-Experience Assessment [85].*

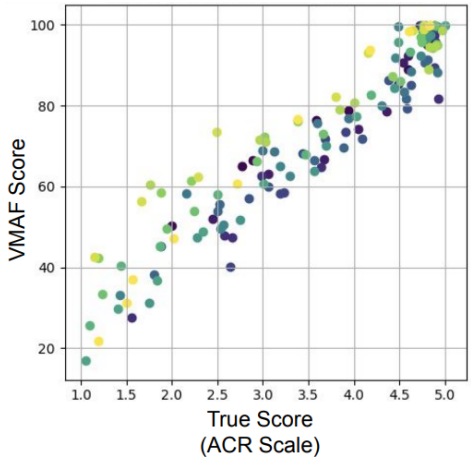


Figura 3.12. Relación entre el MOS y el VMAF. Fuente: *Measuring Video Quality with VMAF: Why you should care [86].*

3.5. Evaluación

3.5.1. Evaluación objetiva

En esta sección se evalúan los principales parámetros que afectan a la QoE en la visualización de vídeos: el número y la duración de las paradas, la representación media de la reproducción y el número de cambios de representación. Los datos obtenidos para los algoritmos y canales de transmisión estudiados se pueden encontrar en la Tabla 3.3, con relación al vídeo "Elephants Dream", y en la Tabla 3.5, en relación con el vídeo "Tears of Steel". Los datos presentados en estas tablas son utilizados para calcular los diferentes modelos de QoE analizados en este trabajo de investigación. Los resultados se muestran en la Tabla 3.4, para el vídeo "Elephants Dream", y en la Tabla 3.6, para el vídeo "Tears of Steel". Todas las tablas presentan la media obtenida sobre 5 iteraciones de cada combinación de algoritmo, canal de transmisión y vídeo. Adicionalmente, en las tablas se muestra la diferencia, en porcentaje o valor absoluto, respecto al algoritmo que presenta el mejor resultado en cada escenario.

Con relación al primer vídeo mostrado en la Tabla 3.3, se puede observar que el algoritmo Look Ahead obtiene mejores resultados que SARA y Müller en términos de paradas en la reproducción ya que, como se muestra en la tabla, Look Ahead no presenta paradas mientras que sí que lo hacen SARA y Müller en los canales de transmisión más exigentes (1 Mbps, 2 Mbps y los dos canales 4G). La tabla refleja que, a pesar de sufrir paradas en la reproducción, el algoritmo Look Ahead no acarrea un descenso significativo en cuanto a la representación media obtenida. De hecho, la representación media obtenida por el algoritmo Look Ahead en todos los escenarios es ligeramente menor que el mejor dato

obtenido en cada caso. En referencia al número de cambios de representación, Look Ahead ofrece, en general, mejores valores que el resto de los algoritmos evaluados.

Tabla 3.3. Evaluación del vídeo "Elephants Dream" con diferentes algoritmos y canales de transmisión.

Canal	Algoritmo de adaptación	Número de paradas	Duración de las paradas (s)	Ratio de parada	Representación media [0-11]		Cambios de representación	
1 Mbps	Müller	1,00	5,81	0,89%	3,64	-0,42%	49,00	+9,38%
	SARA	1,00	5,98	0,91%	**3,66**	max	49,20	+9,82%
	Look Ahead	**0,00**	**0,00**	**0,00%**	3,27	-10,49%	**44,80**	min
2 Mbps	Müller	1,00	5,48	0,84%	3,64	-0,42%	49,20	+8,85%
	SARA	1,00	5,95	0,91%	**3,66**	max	49,20	+8,85%
	Look Ahead	**0,00**	**0,00**	**0,00%**	3,30	-9,62%	**45,20**	min
5 Mbps	Müller	**0,00**	**0,00**	**0,00%**	7,25	-2,27%	**45,20**	min
	SARA	**0,00**	**0,00**	**0,00%**	**7,41**	max	48,20	+6,64%
	Look Ahead	**0,00**	**0,00**	**0,00%**	6,56	-11,59%	47,40	+4,87%
10 Mbps	Müller	**0,00**	**0,00**	**0,00%**	8,66	-1,57%	42,40	+13,98%
	SARA	**0,00**	**0,00**	**0,00%**	**8,79**	max	40,40	+8,60%
	Look Ahead	**0,00**	**0,00**	**0,00%**	8,17	-7,06%	**37,20**	min
2-4-8-4 Mbps	Müller	**0,00**	**0,00**	**0,00%**	**6,48**	max	**44,60**	min
	SARA	**0,00**	**0,00**	**0,00%**	6,17	-4,77%	51,60	+15,70%
	Look Ahead	**0,00**	**0,00**	**0,00%**	6,07	-6,37%	46,40	+4,04%
4G-bus	Müller	0,60	9,49	1,45%	7,93	-9,24%	43,80	+25,14%
	SARA	0,40	2,58	0,39%	8,06	-7,81%	45,60	+30,29%
	Look Ahead	**0,00**	**0,00**	**0,00%**	**8,74**	max	**35,00**	min
4G-car	Müller	0,80	13,00	1,99%	8,74	-3,90%	**37,80**	min
	SARA	2,00	21,46	3,28%	**9,10**	max	38,60	+2,12%
	Look Ahead	**0,00**	**0,00**	**0,00%**	8,53	-6,19%	38,00	+0,53%

Tabla 3.4. Evaluación de los modelos de QoE (λ=1, μ=6000, ζ=1, η=3, β=1, γ=900) para el vídeo "Elephants Dream".

Canal	Algoritmo de adaptación	QoE by Yin et al. (M)		QoE Modificado (M)		QoE PSNR (dB)		QoE VMAF		QoE P.1203	
1 Mbps	Müller	40,03	-41,26%	11,01	-73,89%	31,02	-7,83	70,43	-7,65%	4,24	-0,34
	SARA	37,69	-44,69%	9,94	-76,42%	30,81	-8,04	70,95	-6,97%	4,24	-0,34
	Look Ahead	**68,15**	max	**42,17**	max	**38,84**	max	**76,26**	max	**4,58**	max
2 Mbps	Müller	40,10	-42,50%	12,62	-70,44%	31,36	-7,52	70,88	-7,41%	4,24	-0,34
	SARA	36,27	-47,99%	9,80	-77,06%	30,83	-8,06	70,32	-8,14%	4,24	-0,34
	Look Ahead	**69,74**	max	**42,71**	max	**38,88**	max	**76,55**	max	**4,58**	max
5 Mbps	Müller	340,19	-3,46%	186,74	-6,51%	44,57	-0,42	91,99	-1,06%	4,61	-0,01
	SARA	**352,39**	max	**199,75**	max	**44,99**	max	**92,98**	max	**4,61**	max
	Look Ahead	241,16	-31,56%	162,47	-18,66%	43,67	-1,32	90,68	-2,48%	4,60	-0,01
10 Mbps	Müller	560,55	-3,42%	337,25	-1,22%	47,18	-0,09	94,27	-0,89%	4,63	-0,01
	SARA	**580,38**	max	**341,43**	max	**47,27**	max	**95,11**	max	**4,64**	max
	Look Ahead	464,00	-20,05%	289,87	-15,10%	46,46	-0,80	93,92	-1,26%	4,63	-0,01
2-4-8-4 Mbps	Müller	**288,99**	max	**122,93**	-6,48%	**43,56**	max	**86,99**	-0,07%	**4,60**	max
	SARA	217,97	-24,58%	117,41	-10,69%	42,07	-1,49	85,71	-1,53%	4,60	max
	Look Ahead	249,27	-13,74%	131,46	max	42,86	-0,70	87,04	max	4,60	max
4G-bus	Müller	377,47	-38,83%	174,44	-52,02%	33,98	-13,67	78,04	-16,30%	4,40	-0,22
	SARA	402,49	-34,77%	225,37	-38,02%	40,92	-6,73	89,13	-4,40%	4,49	-0,14
	Look Ahead	**617,04**	max	**363,60**	max	**47,65**	max	**93,24**	max	**4,62**	max
4G-car	Müller	548,47	-0,02%	311,10	-14,31%	32,92	-14,36	72,95	-22,37%	4,23	-0,40
	SARA	499,57	-8,94%	274,16	-24,49%	28,34	-,3	61,95	-34,07%	3,97	-0,66
	Look Ahead	**548,60**	max	**363,07**	max	**47,27**	max	**93,97**	max	**4,63**	max

Tabla 3.5. Evaluación del vídeo "Tears of Steel" con diferentes algoritmos y canales de transmisión.

Canal	Algoritmo de adaptación	Número de paradas	Duración de las paradas (s)	Ratio de parada	Representación media [0-11]		Cambios de representación	
1 Mbps	Müller	**0,00**	**0,00**	**0,00%**	3,28	-0,08%	54,00	+3,85%
	SARA	**0,00**	**0,00**	**0,00%**	**3,28**	max	52,40	+0,77%
	Look Ahead	**0,00**	**0,00**	**0,00%**	2,78	-15,36%	**52,00**	min
2 Mbps	Müller	**0,00**	**0,00**	**0,00%**	3,28	-0,17%	54,60	+3,80%
	SARA	**0,00**	**0,00**	**0,00%**	**3,29**	max	53,80	+2,28%
	Look Ahead	**0,00**	**0,00**	**0,00%**	2,77	-15,60%	**52,60**	min
5 Mbps	Müller	**0,00**	**0,00**	**0,00%**	**6,62**	max	50,80	+16,51%
	SARA	**0,00**	**0,00**	**0,00%**	6,61	-0,08%	54,40	+24,77%
	Look Ahead	**0,00**	**0,00**	**0,00%**	6,07	-8,32%	**43,60**	min
10 Mbps	Müller	**0,00**	**0,00**	**0,00%**	7,94	-0,07%	**46,20**	min
	SARA	**0,00**	**0,00**	**0,00%**	**7,95**	max	48,20	+4,33%
	Look Ahead	**0,00**	**0,00**	**0,00%**	7,09	-10,80%	48,60	+5,19%
2-4-8-4 Mbps	Müller	**0,00**	**0,00**	**0,00%**	**6,06**	max	54,20	+18,86%
	SARA	**0,00**	**0,00**	**0,00%**	5,78	-4,51%	53,40	+17,11%
	Look Ahead	**0,00**	**0,00**	**0,00%**	5,52	-,8%	**45,60**	min
4G-bus	Müller	**0,00**	**0,00**	**0,00%**	8,20	-2,93%	51,80	+17,19%
	SARA	**0,00**	**0,00**	**0,00%**	**8,45**	max	**44,20**	min
	Look Ahead	**0,00**	**0,00**	**0,00%**	8,02	-5,06%	53,20	+20,36%
4G-car	Müller	0,80	7,43	1,01%	8,17	-0,79%	**43,20**	min
	SARA	2,60	30,04	4,09%	**8,23**	max	43,80	+1,39%
	Look Ahead	**0,00**	**0,00**	**0,00%**	7,99	-2,92%	50,20	+16,20%

Tabla 3.6. Evaluación de los modelos de QoE (λ=1, μ=6000, ζ=1, η=3, β=1, γ=900) para el vídeo "Tears of Steel".

Canal	Algoritmo de adaptación	QoE by Yin et al. (M)		QoE Modificado (M)		QoE PSNR (dB)		QoE VMAF		QoE P.1203	
1 Mbps	Müller	52,15	-3,32%	53,97	-1,75%	**36,79**	max	82,23	-1,15%	4,58	-0,01
	SARA	**53,94**	max	**54,93**	max	36,78	-0,01	**83,19**	max	4,58	-0,01
	Look Ahead	42,88	-20,52%	47,84	-12,92%	36,13	-0,66	79,47	-4,47%	**4,59**	max
2 Mbps	Müller	51,19	-1,47%	54,01	-0,05%	36,68	-0,07	**82,63**	max	4,58	-0,01
	SARA	**51,96**	max	**54,03**	max	**36,75**	max	82,23	-0,49%	4,58	-0,01
	Look Ahead	42,37	-18,46%	47,76	-11,61%	36,10	-0,64	79,47	-3,82%	**4,59**	max
5 Mbps	Müller	**220,29**	max	226,07	-0,33%	**41,16**	max	94,68	-0,09%	4,64	max
	SARA	216,40	-1,77%	**226,83**	max	41,11	-0,04	**94,76**	max	4,64	max
	Look Ahead	197,01	-10,57%	203,41	-10,32%	40,74	-0,42	94,06	-0,73%	4,62	-0,02
10 Mbps	Müller	**442,57**	max	398,68	-1,45%	42,85	-0,06	95,97	-0,53%	4,64	-0,06
	SARA	427,21	-3,47%	**404,53**	max	**42,91**	max	**96,48**	max	**4,70**	max
	Look Ahead	298,48	-32,56%	320,08	-20,88%	41,82	-1,09	95,33	-1,19%	4,64	-0,06
2-4-8-4 Mbps	Müller	**171,80**	max	**199,42**	max	**40,33**	max	**92,69**	max	**4,61**	max
	SARA	153,78	-10,49%	183,75	-7,86%	39,80	-0,53	90,59	-2,26%	4,60	-0,01
	Look Ahead	154,88	-9,85%	172,30	-13,60%	39,91	-0,42	91,57	-1,21%	4,60	-0,01
4G-bus	Müller	441,90	-18,96%	410,85	-19,04%	42,77	-0,80	95,96	-0,94%	4,64	-0,01
	SARA	**545,28**	max	**507,49**	max	**43,57**	max	**96,87**	max	**4,65**	max
	Look Ahead	375,31	-31,17%	425,60	-16,14%	42,76	-0,81	95,94	-0,96%	**4,65**	max
4G-car	Müller	**550,97**	max	508,80	-0,82%	33,89	-8,99	83,03	-13,21%	**4,26**	max
	SARA	399,78	-27,44%	384,93	-24,97%	21,52	-21,36	55,10	-42,41%	3,77	-0,50
	Look Ahead	453,31	-17,72%	**513,02**	max	**42,88**	max	**95,67**	max	4,11	-0,15

Pasando a los valores de QoE presentados en la Tabla 3.4, se ve como los cinco modelos ofrecen resultados coherentes respecto a los canales con 1 y 2 Mbps: el algoritmo que provee el valor de QoE más alto es Look Ahead, esto es, el único algoritmo que no sufre paradas en la reproducción. Además, es el algoritmo con el menor número de cambios de representación y, a pesar de obtener la representación media más baja, la diferencia respecto al mejor caso es de sólo el 10 %. De los dos canales 4G se extraen conclusiones similares. Sin embargo, en el caso particular del canal 4G-car en relación con el modelo de QoE propuesto por Yin et al. aparece un resultado incoherente, ya que el valor de QoE obtenido por los algoritmos Look Ahead y Müller es prácticamente el mismo a pesar de que: el segundo algoritmo tiene una ratio de parada del 2 % con una duración media de 13 s, la representación media de Müller es apenas un 5 % mejor y los dos algoritmos tienen el mismo número de cambios de representación. Finalmente, en los canales donde ningún algoritmo sufre paradas los mejores valores de QoE corresponden a los algoritmos que mejor representación media obtienen en cada caso.

En lo que respecta al vídeo "Tears of Steel", la Tabla 3.5 muestra como el único escenario donde se producen paradas es el 4G-car. En ese canal, el algoritmo SARA, de media, está en estado de parada un 4 % del tiempo, lo que representa una duración media de 30 s. Por su parte, Müller presenta una ratio de parada del 1 % con una media de 7,4 s en parada, mientras que Look Ahead no presenta paradas. Analizando los valores obtenidos por los

diferentes modelos de QoE para este mismo canal, la Tabla 3.6 muestra resultados incoherentes por parte de los modelos de QoE de Yin et al. y la ITU-T P.1203, puesto que estos modelos otorgan la mejor calificación al algoritmo Müller a pesar de sufrir paradas en la reproducción. Estos valores parecen contradictorios puesto que el algoritmo de adaptación Look Ahead no sufre paradas en ese escenario y la representación media de Müller es únicamente un 2,25 % mejor que la correspondiente de Look Ahead. En el resto de los escenarios, todos los modelos de QoE ofrecen valores razonables puesto que, en ausencia de paradas, los algoritmos con mejor representación media suelen obtener las mejores calificaciones mientras no exista una diferencia importante en el número de cambios de representación.

A modo de resumen, podemos decir que, en base a los resultados, los modelos de QoE propuestos mejoran al de Yin et al. ya que este último produce resultados incoherentes en algunas situaciones. Aunque es cierto que el modelo de QoE de Yin et al. permite asignar más peso a las paradas (a través del parámetro μ) y por tanto los resultados pueden variar tal y como se sugiere en [72] y en [83]. Además, los modelos QoE propuestos en la tesis basados en PSNR y VMAF ofrecen resultados consistentes en todos los escenarios. Finalmente, en relación con el modelo ITU-T P.1203, observamos que ofrece resultados muy optimistas en la mayoría de los escenarios estudiados (el menor valor de QoE obtenido es de 3,97 para el vídeo "Elephants Dream" y de 3,77 para "Tears of Steel"). Este caso particular puede deberse a la limitación de utilizar el modo más simple del modelo puesto que es el único compatible con el contenido codificado con VP9.

Los resultados obtenidos con relación al vídeo "Mix" se muestran en la Tabla 3.7 y Tabla 3.8. Para este vídeo, todos los algoritmos estudiados, incluido Look Ahead, producen paradas en la reproducción por vaciamiento del buffer en los canales más exigentes (en particular en los canales 4G-car y el canal escalonado con 2-8 Mbps). En cualquier caso, es otra vez el algoritmo Look Ahead el que ofrece los mejores resultados con relación al número y duración de las paradas sin apenar reducir la representación media y con un número similar de cambios de representación.

Para el análisis de los modelos de QoE, pasamos a examinar el caso particular del canal de transmisión 4G-car. En este canal, de media, Müller produce 5,6 paradas con una duración total media de 63,19 s, SARA produce 9,2 paradas con una duración total media de 100,40 s y Look Ahead produce 4 paradas con una duración total media de 24,37 s. La representación media es ligeramente superior en el caso del algoritmo SARA (8,87) que en el de Look Ahead (8,81) y Müller (8,66); Y el número de cambios de representación es bastante similar en todos los algoritmos de adaptación (161,2, 163,4 y 168,2 respectivamente). Teniendo en consideración estos datos, parece claro que el algoritmo Look Ahead tiene un mejor comportamiento debido a la diferencia en el número y la duración de las paradas. Sin embargo, en el modelo de QoE propuesto por Yin et al. el algoritmo Müller obtiene una mejor valoración (2227 M) que SARA (2120 M) y Look Ahead (2110 M). En un escenario real costaría comprender como una reproducción con una calidad media y un número de cambios de representación muy parecidos, pero con un número y tiempo de parada menor obtiene peores resultados. En

contraste, el modelo de QoE Yin et al. modificado muestra un resultado completamente diferente. En este caso, el algoritmo Look Ahead obtiene mejor valoración (1610 M) que SARA (1113 M) y Müller (1215 M). En la misma línea están los modelos de QoE propuestos basados en el PSNR y el VMAF. Así, el valor de QoE_{PSNR} para el algoritmo Look Ahead (35,74 dB) es mucho mejor el QoE_{PSNR} para Müller (25,50 dB) y SARA (19,72 dB), mientras que el valor de QoE_{VMAF} para Look Ahead (88,37) también es mejor que el QoE_{VMAF} para Müller (72,72) y SARA (60,74). Finalmente, la ITU-T P.1203 proporciona resultados coherentes, pero no tan notorios como los obtenidos por los modelos propuestos.

Tabla 3.7. Evaluación del vídeo "Mix" con diferentes algoritmos y canales de transmisión.

Canal	Algoritmo de adaptación	Número de paradas	Duración de las paradas (s)	Ratio de parada	Representación media [0-11]		Cambios de representación	
1 Mbps	Müller	1,00	6,86	0,25%	**3,62**	max	194,80	+1,4%
	SARA	1,00	9,05	0,33%	3,55	-1,9%	208,40	+8,4%
	Look Ahead	**0,00**	**0,00**	**0,00%**	3,17	-12,4%	**192,20**	min
10 Mbps	Müller	**0,00**	**0,00**	**0,00%**	**8,74**	max	**166,20**	min
	SARA	**0,00**	**0,00**	**0,00%**	8,13	-7,0%	192,00	+15,5%
	Look Ahead	**0,00**	**0,00**	**0,00%**	7,87	-10,0%	174,80	+5,2%
2-8 Mbps	Müller	1,80	22,48	0,82%	**9,16**	max	**173,40**	min
	SARA	4,60	46,91	1,70%	8,23	-10,2%	185,60	+7,0%
	Look Ahead	**1,00**	**3,71**	**0,13%**	7,87	-14,1%	174,60	+0,7%
4G-bus	Müller	**0,00**	**0,00**	**0,00%**	8,74	-3,2%	197,80	+14,2%
	SARA	2,40	42,44	1,54%	**9,03**	max	**173,20**	min
	Look Ahead	**0,00**	**0,00**	**0,00%**	8,69	-3,8%	174,20	+0,6%
4G-car	Müller	5,60	63,19	2,29%	8,66	-2,4%	168,20	+4,3%
	SARA	9,20	100,44	3,64%	**8,87**	max	**161,20**	min
	Look Ahead	**4,00**	**24,37**	**0,88%**	8,81	-0,7%	163,40	+1,4%

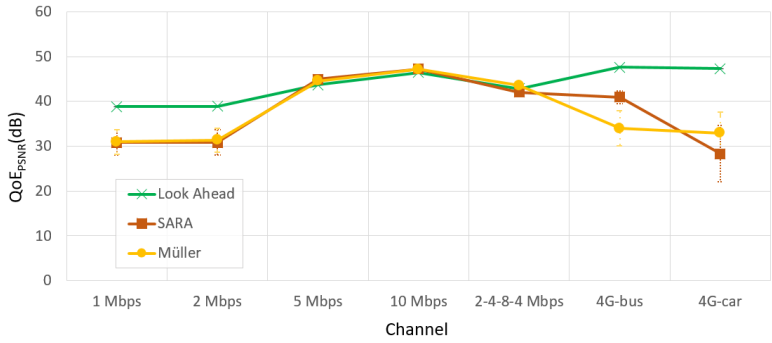
Tabla 3.8. Evaluación de los modelos de QoE (λ=1, μ=6000, ζ=1, η=3, β=*1* γ=900) para el vídeo "Mix".

Canal	Algoritmo de adaptación	QoE by Yin et al. (M)		QoE Modificado (M)		QoE PSNR (dB)		QoE VMAF		QoE P.1203	
1 Mbps	Müller	**192,07**	max	166,72	-13,8%	35,21	-3,20	**80,67**	max	4,13	-0,46
	SARA	152,46	-20,6%	144,06	-25,5%	34,01	-4,40	79,74	-1,2%	4,13	-0,46
	Look Ahead	176,27	-8,2%	**193,43**	max	**38,41**	max	80,58	-0,1%	**4,59**	max
10 Mbps	Müller	**2133,48**	max	**1473,66**	max	**46,70**	max	**96,81**	max	4,65	-0,01
	SARA	1559,89	-26,9%	1152,67	-21,8%	45,74	-0,96	96,11	-0,7%	**4,66**	max
	Look Ahead	1443,09	-32,4%	1212,40	-17,7%	45,35	-1,35	95,91	-0,9%	4,65	-0,01
2-8 Mbps	Müller	**2481,90**	max	**1650,38**	max	36,88	-6,28	88,65	-5,9%	4,07	-0,09
	SARA	1403,60	-43,4%	916,80	-44,4%	28,27	-14,89	79,17	-16,0%	3,52	-0,64
	Look Ahead	1432,60	-42,3%	1201,11	-27,2%	**43,16**	max	**94,20**	max	**4,16**	max
4G-bus	Müller	1956,69	-9,2%	1229,98	-21,1%	46,26	-0,34	95,67	-1,1%	**4,65**	max
	SARA	**2154,50**	max	1354,25	-13,1%	30,69	-15,91	81,74	-15,5%	3,93	-0,72
	Look Ahead	2010,71	-6,7%	**1558,94**	max	**46,60**	max	**96,74**	max	**4,65**	max
4G-car	Müller	**2227,83**	max	1214,99	-24,6%	25,50	-10,24	72,72	-17,7%	3,37	-0,08
	SARA	2119,87	-4,8%	1113,45	-30,9%	19,72	-16,02	60,74	-31,3%	3,03	-0,42
	Look Ahead	2100,26	-5,7%	**1610,78**	max	**35,74**	max	**88,37**	max	**3,45**	max

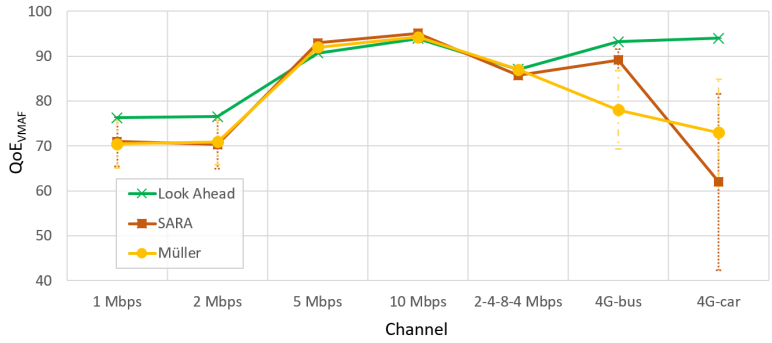
Al examinar otros escenarios como, por ejemplo, el canal escalonado 2-8 Mbps, se obtienen conclusiones similares. En dicho canal, a pesar de la duración de las paradas obtenidas utilizando los algoritmos SARA y Müller, es este último quien obtiene los mejores resultados, por mucho, si atendemos al modelo de QoE propuesto por Yin et al. [72]. Por el contrario, tanto los modelos propuestos basados en el PSNR como en el VMAF, así como la ITU-T P.1203, otorgan sus mejores valores de QoE al algoritmo Look Ahead.

A partir de aquí, y con el objetivo de simplificar el análisis, pasaremos a considerar únicamente los modelos propuestos basados en el PSNR y en VMAF, así como el estándar ITU-T P.1203.

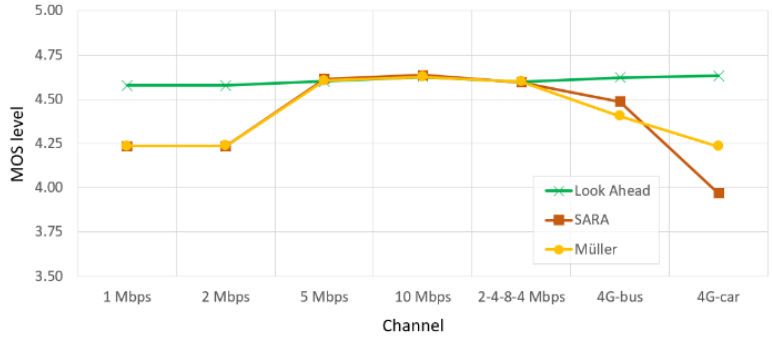
En este sentido, la Figura 3.13 muestra la evaluación del vídeo "Elephants Dream" para los diferentes algoritmos estudiados y los modelos de QoE basados en PSNR, basados en VMAF y el modelo de la ITU-T P.1203. Así, se puede observar de forma visual como los tres modelos ofrecen resultados similares en todos los escenarios, proporcionando los valores de QoE más bajos en los escenarios más exigentes. En la Figura 3.13-a (modelo QoE_{PSNR}), los valores han sido obtenidos fijando ζ=1*,* η=3*,* δ=0. Adicionalmente, cada algoritmo, para cada canal, muestra un umbral inferior obtenido con η=4 (una mayor penalización de las paradas) y un umbral superior obtenido con η=2 (una menor penalización de las paradas). De la misma forma, en la Figura 3.13-b (modelo QoE_{VMAF}) se ha fijado β=1, γ=900, δ=0 con los umbrales inferiores y superiores obtenidos estableciendo γ=1500 y γ=300, respectivamente. Observando las tres gráficas, se puede comprobar como los tres modelos ofrecen resultados similares, aunque la ITU-T P.1203 ofrece resultados más optimistas.



(a) QoE_{PSNR} (ζ=1, η=3, δ=0).



(b) QoE_{VMAF} (β=1, γ=900, δ=0).

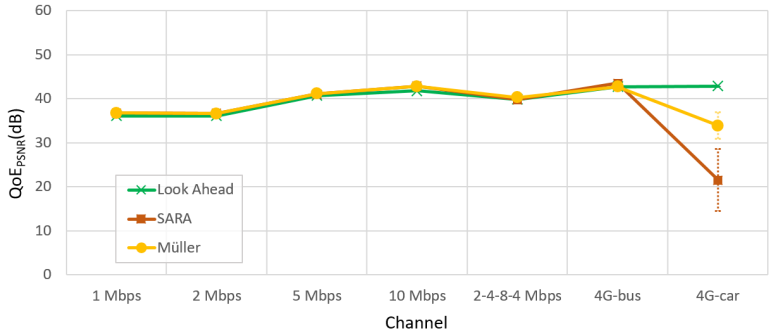


(c) $QoE_{ITU\text{-}T\ P.1023}$.

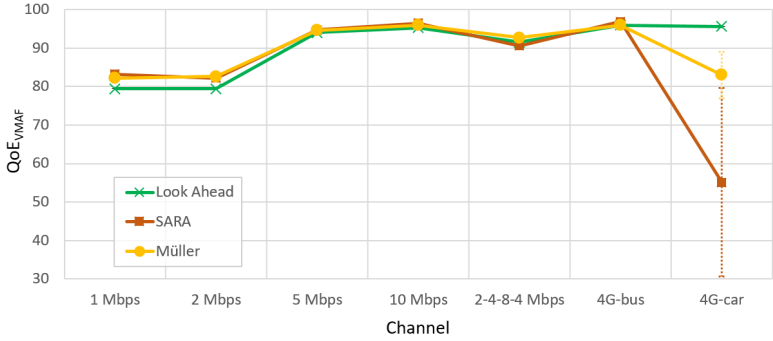
Figura 3.13. QoE del vídeo “Elephants Dream”.

Al analizar los resultados obtenidos para el vídeo “Tears of Steel”, Figura 3.14, se obtienen resultados similares. La única diferencia remarcable es el valor del algoritmo Look Ahead en el canal de transmisión 4G-car para el modelo ITU-T P.1203. Como se ha mencionado al analizar las tablas, este valor resulta incoherente teniendo en cuenta la

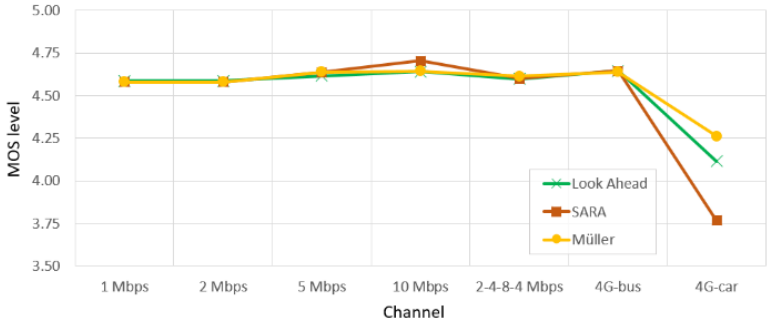
ausencia de paradas por parte del algoritmo Look Ahead y que la representación media obtenida es únicamente un 2,92 % menor respecto al mejor caso.



(a) QoE_{PSNR} (ζ=1, η=3, δ=0).



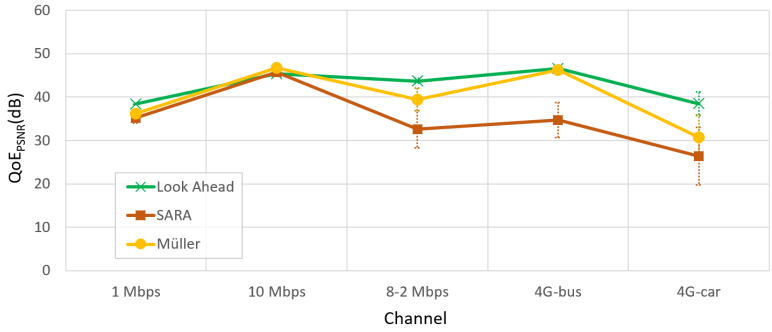
(b) QoE_{VMAF} (β=1, γ=900, δ=0).



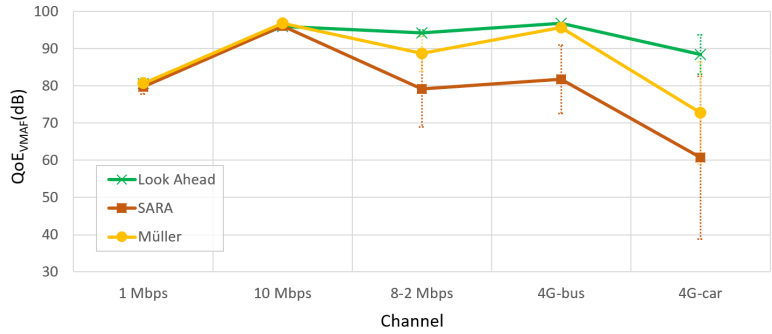
(c) $QoE_{ITU\text{-}T\ P.1023}$.

Figura 3.14. QoE del vídeo “Tears of Steel”.

Finalmente, la Figura 3.15 muestra como los tres modelos de QoE analizados se comportan de forma parecida para el vídeo "Mix", con la salvedad de la alta penalización que supone el número de paradas del canal 4G-car para el modelo de la ITU.



(a) QoE_{PSNR} (ζ=1, η=3, δ=0).



(b) QoE_{VMAF} (β=1, γ=900, δ=0).

(c) $QoE_{ITU\text{-}T\ P.1023}$.

Figura 3.15. QoE del vídeo "Mix".

3.5.2. Evaluación subjetiva

Una vez analizados los modelos QoE en diferentes escenarios, se ha realizado la evaluación subjetiva de un canal en particular. Concretamente, consideraremos el canal con las condiciones más exigentes de entre los estudiados en la evaluación objetiva; esto es, el canal 4G-car. Los vídeos evaluados son los producidos por la utilización de los dos mejores algoritmos para el canal en cuestión: Look Ahead y Müller.

Con el objetivo de simplificar la evaluación subjetiva, en esta sección se consideran 4 casos diferentes tal como se muestra en la Figura 3.16: caso A (el vídeo "Elephants Dream" usando el algoritmo Look Ahead), B ("Elephants Dream" con Müller), C ("Tears of Steel" con Look Ahead) y D ("Tears of Steel" con Müller).

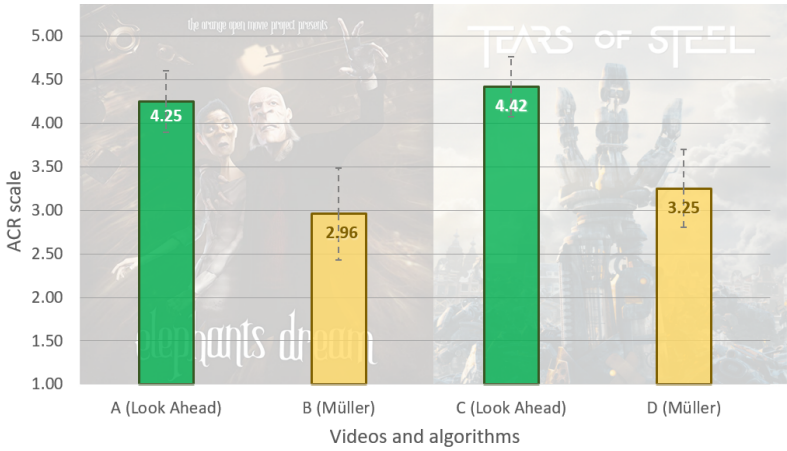


Figura 3.16. Evaluación del MOS para el canal 4G-car.

La Figura 3.16 muestra la evaluación de MOS obtenida usando la escala ACR para los vídeos "Elephants Dream" a la izquierda y "Tears of Steel" a la derecha. Cabe destacar que las figuras incluyen el intervalo de confianza de 99 % para cada par de vídeo y algoritmo. Como se muestra en la figura, consideramos que los intervalos de confianza son bastante estrechos, siendo el mayor de ellos de ±0,53. En lo referente al vídeo "Elephants Dream", vemos unos valores de MOS para los algoritmos Look Ahead y Müller de 4,25 y 2,96, respectivamente. Si comparamos estos datos con los obtenidos en la Tabla 3.4 y la Figura 3.13, podemos observar que los modelos propuestos basados en el PSNR y el VMAF ofrecen resultados coherentes considerando el mapeo entre MOS-ACR y PSNR/VMAF mostrado en la Figura 3.11 y la Figura 3.12 : 47,27 dB y 32,92 dB para el modelo QoE_{PSNR}, y 93,97 y 72,95 para QoE_{VMAF} para los algoritmos Look Ahead y Müller, respectivamente. Por otro lado, a pesar de que la estimación del modelo $QoE_{ITU\text{-}T\ P.1203}$ para el algoritmo Look Ahead (4,63) está muy próxima a la obtenida con los test subjetivos (4,25), la misma para el algoritmo Müller (4,23) difiere de la obtenida en 1,27.

Las mismas consideraciones surgen al analizar el vídeo "Tears of Steel". En este caso, los dos valores de MOS (caso C, 4,42 y caso D, 3,25) superan ligeramente a los del vídeo

anterior. En lo que respecta al QoE_{PSNR}, en este caso, $QoE_{PSNR}(C)$ es menor que la del vídeo anterior (42,88 dB) y el $QoE_{PSNR}(D)$ es ligeramente mayor (33,89 dB). Por el contrario, el modelo de QoE basado en VMAF ofrece los resultados más coherentes teniendo en consideración la Figura 3.12: el $QoE_{VMAF}(C)$ sube ligeramente (95,67) mientras que el $QoE_{VMAF}(D)$ incrementa hasta 83,03 pero sigue lejos de $QoE_{VMAF}(C)$. Finalmente, los valores obtenidos del estándar de la ITU no están en línea con los de la Figura 3.16 puesto que $QoE_{ITU\text{-}T\ P.1203}(C)$=4,11 es peor que $QoE_{ITU\text{-}T\_P.1203}(D)$=4,26.

Por lo tanto, según los resultados, aunque los tres modelos de QoE analizados proporcionan resultados coherentes, el modelo que más se aproxima a los datos obtenidos mediante el estudio subjetivo es el modelo basado en VMAF propuesto en esta tesis. Sin embargo, hay que tener en cuenta que las evaluaciones llevadas a cabo utilizando el modelo de la ITU lo han sido utilizando el modo de funcionamiento más básico, de forma que los resultados que se pudieran obtener con el resto de los modos de funcionamiento podrían mejorar su rendimiento.

La Tabla 3.9 resume los valores obtenidos en los cuatro casos para el canal 4G car. Es importante mencionar que los modelos propuestos (QoE_{PSNR} y QoE_{VMAF}) pueden ser refinados ajustando los parámetros ζ, η, δ, β, y γ. En este sentido, la tabla muestra los valores de QoE para diferentes de η y γ.

Tabla 3.9. Comparación de la evaluación subjetiva con los modelos de QoE objetivos (β=1, ζ=1, δ=0) para el canal 4G-car.

Case	MOS	ITU-T P.1203	QoE_{PSNR} (dB)			QoE_{VMAF} [0-100]		
			η=2	η=3	η=4	γ=300	γ=900	γ=1500
A	4,25	4,63	47,27	47,27	47,27	93,97	93,97	93,97
B	2,96	4,23	37,67	32,92	28,17	84,88	72,95	61,02
C	4,42	4,11	42,88	42,88	42,88	95,67	95,67	95,67
D	3,25	4,26	36,93	33,89	30,86	89,10	83,03	76,96

Adicionalmente, la Figura 3.17 muestra los resultados del estudio realizado para los dos vídeos y algoritmos. En particular, la figura muestra una comparación subjetiva entre los vídeos mostrados utilizando los algoritmos Müller y Look Ahead. En el eje Y se muestra la escala CCR (entre -3 y 3) comparando la reproducción con el algoritmo Müller respecto a la propia de Look Ahead. Los resultados muestran que ninguna de las 24 personas que realizaron el test consideró que la reproducción usando el Algoritmo Müller fuera mejor que la reproducción usando Look Ahead.

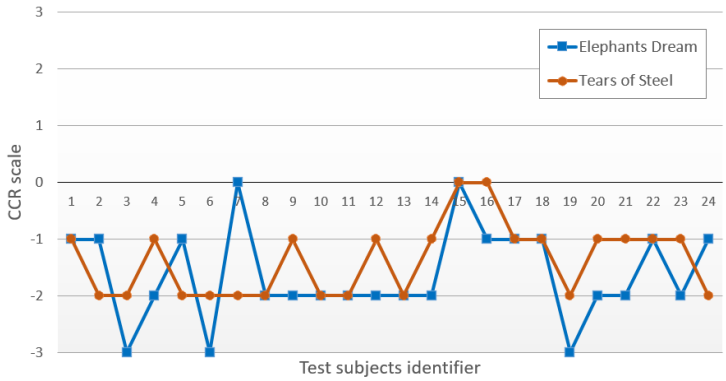


Figura 3.17. Comparación entre los algoritmos Müller y Look Ahead para el canal 4G-car.

Finalmente, se presentan los resultados obtenidos a la pregunta: "Como usuario de un servicio de vídeo bajo demanda, ¿cuál de las siguientes métricas tiene mayor prioridad? Ordenar de mayor (1) a menor (4) prioridad: a) Que la calidad de reproducción sea la mejor posible; b) Que no existan interrupciones o que sean muy limitadas; c) Que no se aprecien cambios de calidad significativos; d) Que el retardo inicial sea reducido".

En la Figura 3.18 se muestran las respuestas a las prioridades elegidas por los usuarios. En ella se ve claramente la tendencia de las prioridades de los usuarios: no tener paradas en la reproducción, seguida de la calidad de vídeo mostrada, después al número de cambios de calidad y, finalmente, a que el retraso inicial sea reducido. En particular, el 70,83 % de los sujetos consideró que las paradas en la reproducción es el factor más importante a la hora de evaluar la calidad de la reproducción. Este porcentaje sube hasta el 87,50 % si se tiene en cuenta a los usuarios que consideran que las paradas son uno de los dos factores más importantes; ningún usuario consideró que la presencia de paradas fuera el factor menos relevante. Además, 2 de cada 3 usuarios consideró que la calidad del vídeo mostrado es una de las dos métricas más importantes (16,67 % consideró a la calidad del vídeo como el factor más importante). En el otro lado se encuentran el número de cambios de representación y el retardo inicial. El primero fue elegido tercera métrica más relevante por el 54,17 % de los usuarios y como métrica menos relevante por el 20,83 % mientras que el 62,50 % de los usuarios consideró que el retraso inicial es la métrica menos relevante entre las cuatro opciones del estudio, aunque es la más relevante para 1 de cada 8 usuarios.

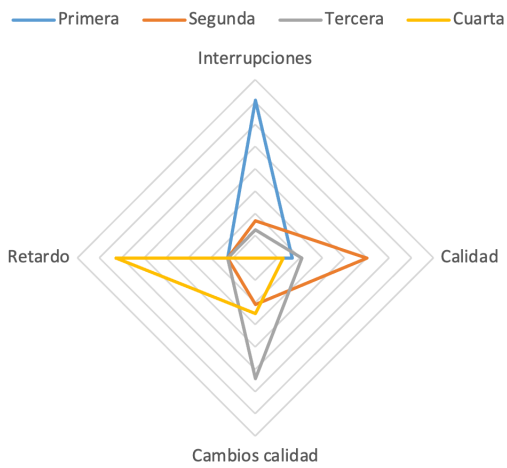


Figura 3.18. Respuestas de las prioridades de los usuarios respecto a las métricas de calidad.

3.5.3. Rendimiento de los métodos objetivos

Como se ha comentado en el punto 3.2.2.2, el Video Quality Experts Group propone una serie de medidas como medio de evaluar el rendimiento de los algoritmos de predicción de la QoE respecto a los valores obtenidos mediante métodos subjetivos.

Tabla 3.10. Coeficiente de correlación de Pearson (PLCC) de los modelos objetivos respecto a la evaluación subjetiva.

	ITU	Yin	Yin modif	PSNR			VMAF		
				η=2	η=3	η=4	γ=300	γ=900	γ=1500
PLCC	0,146	-0,402	0,211	0,537	0,579	0,594	0,608	0,604	0,602

Tabla 3.11. Coeficiente de correlación de Spearman (SROCC) de los modelos objetivos respecto a la evaluación subjetiva.

	ITU	Yin	Yin modif	PSNR			VMAF		
				η=2	η=3	η=4	γ=300	γ=900	γ=1500
SROCC	-0,038	-0,277	0,376	0,523	0,572	0,572	0,615	0,615	0,615

En la Tabla 3.10 y en la Tabla 3.11 se muestran los resultados de la evaluación de la precisión y de la monotonicidad de los modelos estudiados a través del coeficiente de correlación de Pearson y del coeficiente de correlación de Spearman, respectivamente.

El coeficiente de correlación de Pearson indica la dependencia lineal entre dos variables con independencia de la escala. El coeficiente puede tomar valores entre -1 y 1. Los valores cercanos a los extremos indican una relación lineal inversa o positiva mientras que los valores cercanos a 0 denotan una falta de relación lineal, aunque pudiera haberla de otro tipo. Por su parte, el coeficiente de correlación de Spearman, que puede tomar los mismos valores que el PLCC, indica la adaptabilidad de la relación de dos variables a

una función monotónica. De esta forma, cuando más extremo sea su valor mayor será el nivel de asociación, tanto positivo como negativo.

Analizando los valores obtenidos de la comparación entre las predicciones de los modelos de QoE con los resultados de la evaluación subjetiva, la ITU-T P.1203 obtiene, para ambos coeficientes, valores cercanos a 0 mientras que el modelo de Yin et al. obtiene los valores más negativos -0,402 y -0,277. El modelo propuesto de Yin modificado ya ofrece valores positivos dejando el PLCC en 0,211 y el SROCC 0,376. Por último, los modelos basados en el PSNR y en VMAF son los que mejor resultados obtienen con unos rangos de [0,537, 0,594] y [0,523, 0,572] de PLCC y SROCC (en función del valor de η), respectivamente, para el modelo basado en PSRN; y [0,602, 0,608] y [0,615, 0,615] de PLCC y SROCC, (en función del valor de γ) respectivamente, para el modelo basado en VMAF.

Por último, al evaluar la consistencia de la predicción en función de Outlier Ratio (OR) obtenemos el valor de 0 para todos los modelos que predicen valores compatibles con el MOS (ITU-T P.1203 y VMAF). El valor obtenido indica que los modelos ofrecen sus predicciones dentro del rango [MOS - 2σ, MOS + 2σ].

Según estos criterios de rendimiento, los modelos basados en VMAF y PSNR son, en este orden, los que mejor resultados ofrecen con una corta separación entre ellos y a una distancia relativamente significativa del resto de modelos.

3.6. Conclusiones y trabajo futuro

En este capítulo se han presentado tres nuevos modelos para calcular la calidad de experiencia (QoE) en la reproducción vídeos de forma objetiva: el modelo QoE propuesto por Yin et al. modificado, el modelo QoE basado en el PSNR y el modelo basado en VMAF. Se han realizado evaluaciones objetivas y subjetivas que demuestran que las tres propuestas ofrecen resultados más próximos a la realidad, en términos de calidad de experiencia, que el modelo propuesto por Yin et al. [72] y la recomendación ITU-T P.1203 [68]. La evaluación ha sido llevada a cabo utilizando dos algoritmos ABR bien conocidos: Müller y SARA, así como el algoritmo Look Ahead.

Como parte del trabajo futuro, cabe destacar que los modelos de QoE propuestos basados en el PSNR y el VMAF pueden ser mejorados incorporando otros factores de influencia que afectan a la experiencia de los usuarios como el número de paradas y no únicamente el tiempo total de parada. En general, es más molesto para los usuarios tener muchas paradas de corta duración que tener pocas paradas pero más largas [87]. Por ejemplo, en la reproducción de un vídeo, los usuarios prefieren sufrir una parada de 10 segundos que 10 paradas de 1 segundo. Adicionalmente, es importante analizar cómo afecta el número de cambios de calidad a la QoE percibida. Este tema ha sido más profundamente analizado en [88] y [89]. También, es interesante comparar los modelos propuestos con el software ATLAS [90] y con modos más complejos de funcionamiento de la ITU-T P.1203. En este sentido la ITU ha publicado el estándar ITU-T P.1204 [71], introducido

en la sección 3.2.2.5, de forma que pasa a incluir los codecs de vídeo (AVC, HEVC y VP9) así como mayores resoluciones (hasta UHD), entre otras características.

Otra vía de trabajo que abre este estudio es la posibilidad de introducir medidas objetivas de la calidad del vídeo como VMAF en el proceso de codificación. Dejando de lado el sobrecoste computacional que puede requerir, utilizar los valores de VMAF de los segmentos codificados para crear representaciones donde dicho valor sea lo más estable posible, podría ayudar a mejorar la calidad de experiencia de los usuarios al no introducir diferencias sustanciales en la calidad percibida entre segmentos. Al mismo tiempo, concedería el beneficio a los algoritmos de adaptación de saber que mantener una representación dada minimiza los cambios de calidad. Este último punto, aunque aparentemente intuitivo, no se cumple siempre ni siquiera en el caso de utilizar codificación de calidad constante.

Es importante enfatizar que tanto los algoritmos desarrollados (https://github.com/comm-iteam/ExoPlayer) [41] como la aplicación y contenido de pruebas (https://lookahead.iteam.upv.es/) [40] están disponibles de forma que las evaluaciones aquí presentadas pueden ser fácilmente reproducibles. Para la evaluación de los modelos propuestos, se ha desarrollado y publicado una herramienta que facilita la codificación y cálculo de los parámetros requeridos por los modelos como los valores de PSNR y VMAF de cada segmento (https://github.com/comm-iteam/dashgen) [43]. De esta forma, estas herramientas permiten la reproducción de los resultados presentados.

Capítulo 4

Coordinación de la reproducción de clientes DASH en entornos Wi-Fi

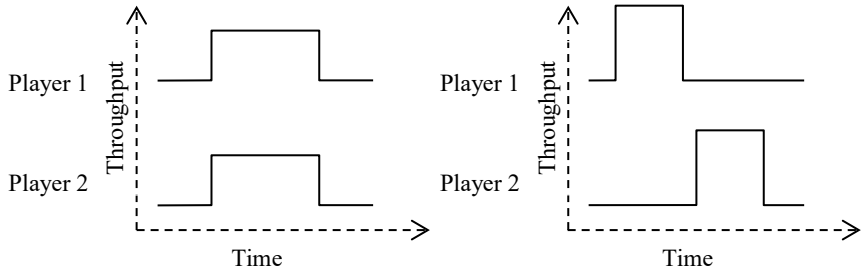
4.1. Introducción

Como se ha comentado anteriormente, DASH [3] se ha convertido en uno de los principales mecanismos para la transmisión de vídeo sobre redes IP. La adopción de DASH supone un cambio de modelo que traslada toda la lógica de la reproducción al player de vídeo. Adicionalmente, la utilización de HTTP para la transmisión de los segmentos ha eliminado la necesidad de un servicio específico de streaming que, a su vez, facilita la utilización de CDNs (*Content Delivery Network*) para escalar los servicios. Sin embargo, DASH también introduce nuevos retos, sobre todo en el lado del receptor cuando el último salto es Wi-Fi y además está compartido por un gran número de dispositivos.

En este tipo de escenarios como, por ejemplo, salas de eventos, estadios y medios de transporte (autobuses, aviones, barcos…) es frecuente ofrecer servicios de streaming de vídeo relacionados con la propia actividad o el entretenimiento. En estos entornos, los reproductores de vídeo compiten entre ellos para descargar los segmentos de vídeo, lo cual tiene efectos adversos como la inestabilidad y la desigualdad en la distribución de la tasa de transferencia disponible [91].

La Figura 4.1 muestra un ejemplo donde dos players de vídeo coinciden de diferentes formas. En el primer ejemplo, mostrado en la Figura 4.1a, los dos clientes coinciden

durante todo el tiempo de descarga, por lo que su percepción de la tasa de transferencia disponible es significativamente menor a la percibida por los players del ejemplo de la Figura 4.1b, donde no coinciden. Debido a la alternancia de los estados de descarga y de espera (ON/OFF), la visión del estado de la red de un dispositivo puede verse alterada por la casuística del momento.



a) Caso 1 b) Caso 2

Figura 4.1. Diferentes escenarios de solapamiento de la descarga de los segmentos.

A este problema se suma que la utilización de redes Wi-Fi introduce otro factor de variabilidad y de desigualdad en el acceso al medio de transmisión. En entornos con alta densidad de dispositivos, por múltiples factores, se negocian y se establecen velocidades de enlace muy dispares que, además, varían con el tiempo. En esta situación los dispositivos con mayores velocidades de enlace ocupan el medio mucho menos tiempo que los dispositivos con bajas velocidades de enlace. Esto implica que para que un dispositivo con alta velocidad de enlace libere el medio de una forma significativa para otro dispositivo con una baja velocidad de enlace, la reducción de las transferencias debe ser importante.

Estos problemas, como se ve en la sección 4.2, se pueden intentar resolver de múltiples formas; y si la solución implica la interacción de distintos dispositivos de la red es inevitable la utilización de un protocolo de comunicación para el envío de la información pertinente. Y es precisamente este punto (el protocolo de comunicación y los mensajes a transmitir) el que viene a estandarizar *Server and Network Assisted DASH* (SAND), definida en la ISO/IEC FDIS 23009-5:2016 – Part 5 [92].

En este capítulo se presenta una implementación de DASH-SAND que tiene como objetivo la coordinación del recurso de la tasa de transferencia entre los reproductores de vídeo que se encuentran en una misma red Wi-Fi con una alta concentración de usuarios.

4.2. Estado del arte

MPEG SAND tiene como objetivo permitir una mejor cooperación del cliente, servidor y red en la transmisión de contenido DASH. Si bien esto no es una prerrogativa de DASH-SAND, lo que sí que introduce es la estandarización de mensajes y protocolos para que

proveedores de servicio, operadores de red y usuarios finales puedan interactuar sin restricciones de compatibilidad.

Existen en la literatura diversos trabajos relacionados con DASH-SAND aunque, debido a que es un estándar relativamente novedoso, no existen estudios que traten el escenario que se contempla en esta tesis. Cabe destacar, por ejemplo, el trabajo de Khorov et al. [93], en el que se propone añadir un servicio DANE (*DASH-Aware Network Element*) en el propio punto de acceso. Este servicio DANE sería el encargado de modificar la forma en la que el punto de acceso pone el medio de transmisión a disposición de los distintos clientes. La forma en la que se propone distribuir la tasa de transferencia disponible entre los clientes de vídeo es la utilización del *Transmit Opportunity* (TXOP) del punto de acceso. Si bien este tipo de modificaciones son interesantes para ser estudiadas, en la práctica existen multitud de dispositivos Wi-Fi que pueden no formar parte de la red en cuestión y es difícil modificar la forma en la que los dispositivos comerciales acceden al medio.

Kleinrouweler et al. [94] evalúan la utilización de una gestión de colas de tráfico para reducir el número de parones en la reproducción de vídeo en redes cableadas con un elevado número de clientes de vídeo (600). A pesar de que tanto el entorno de pruebas (red Ethernet) como la aproximación al problema es diferente, en dicho estudio se pueden encontrar puntos comunes con el trabajo realizado en la propuesta aquí presentada ya que, en ambos casos, existe una necesidad de que los clientes de vídeo informen a un servicio central de cierta información y de que exista una tasa de transferencia disponible en todo momento para la trasmisión de las comunicaciones de control entre el servicio de gestión y los clientes de vídeo. Este último punto, en el caso del escenario considerado es, si cabe, más crítico debido a la utilización de redes Wi-Fi.

De los mismos autores que el trabajo anterior, en [95] se propone como mecanismo de obtención de la información relativa a las características del tráfico de los clientes de vídeo la utilización de un dispositivo SDN (*Software Defined Network*). Otra solución basada en SDN es la presentada por Lee et al. [96], donde se propone la utilización de un punto de acceso con capacidades de SDN que intercepta las peticiones de los clientes de vídeo. Gracias a esta interceptación, el sistema es capaz de estimar el tamaño del buffer de los clientes, así como sus anchos de banda. Esta propuesta tiene varios puntos que dificultan su implementación en un despliegue real. La más importante es que el sistema no puede funcionar utilizando la versión segura de HTTP, es decir, HTTPS. Otros puntos difícilmente implementables serían: la modificación de la MPD de DASH para incluir información relativa al PSNR de los segmentos, y la interceptación y modificación de la representación solicitada por los clientes.

Finalmente, en la propuesta de Cofano et al. [97] se evalúan tres aproximaciones diferentes a la adaptación de streaming asistida por la red: 1) por reserva de ancho de banda; 2) estimación del ancho de banda disponible; y 3) utilización conjunta de los tipos anteriores. La utilización del modo 2 es, según este trabajo, el mecanismo de colaboración cliente-red que mejores resultados proporciona teniendo en cuenta el parámetro VQF (*Video Quality Fairness*).

4.3. Propuesta

Con el objetivo de mejorar la experiencia de los usuarios en el acceso al contenido multimedia en entornos Wi-Fi, se propone una solución basada en DASH-SAND que permite la coordinación de los reproductores de vídeo de forma que se produzca un menor número de paradas en la reproducción. La propuesta consta de dos partes, tal como refleja la Figura 4.2:

- **Servidor de contenido**. Aparte de servir contenido, realiza funciones de DANE, tal como se comenta a continuación.
- **Clientes de vídeo**. Son reproductores DASH a los que se les ha añadido capacidades de utilizar SAND para comunicar su estado y recibir indicaciones sobre el estado de la red.

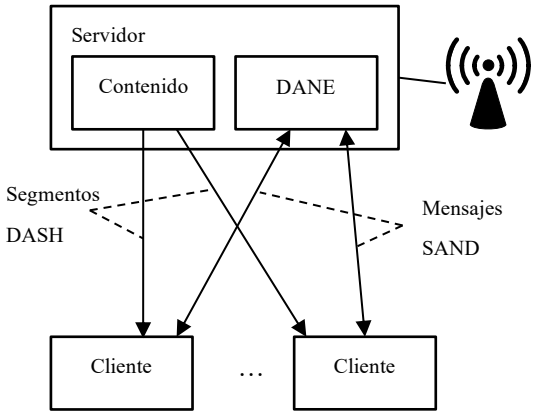


Figura 4.2. Solución SAND para la mejora del servicio DASH.

En la Figura 4.2 se puede observar que, adicionalmente a la descarga de los segmentos de DASH, los clientes de vídeo mantienen una comunicación bidireccional (a través de conexiones *WebSocket*) con el elemento DANE desarrollado. En particular, los clientes de vídeo realizan dos tipos de notificaciones: 1) periódicas, utilizadas a modo de *heartbeat* y para indicar el tamaño del buffer si está en estado de reproducción; 2) al producirse un cambio en la estimación de la tasa de transferencia disponible o en la representación que se está visualizando.

En la Figura 4.3 se muestra un ejemplo de la comunicación proactiva del elemento DANE en la comunicación de las tasas de transferencia estimadas por él mismo. En la figura se pueden observar dos clientes de DASH (C1 y C2). Cuando los clientes terminan de descargar un segmento de vídeo (línea verde descendiente) comunican la estimación de la tasa de transferencia al elemento DANE (flecha verde). Al recibir una nueva estimación de la tasa de transferencia disponible realizada por algún cliente, el elemento DANE genera y envía una nueva estimación a todos los reproductores participantes con la nueva información integrada (flecha azul).

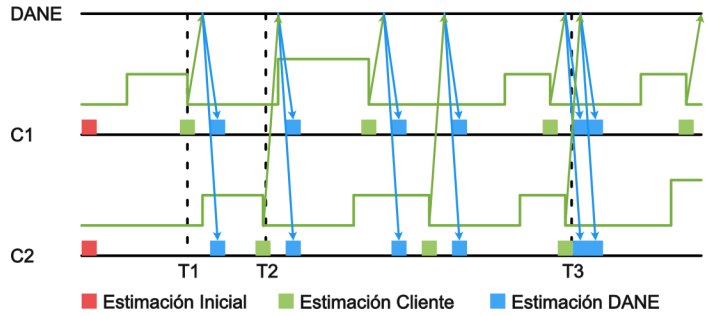


Figura 4.3. Ejemplo de comunicación proactiva de la estimación de la tasa de transferencia.

Siguiendo con el ejemplo de la figura, se pude observar cómo los clientes C1 y C2 terminan las descargas de sus primeros segmentos en los instantes de tiempo T1 y T2, respectivamente. En la figura se muestra que, a pesar de que el instante de tiempo T2 es anterior al inicio de la descarga del segundo segmento por parte del cliente C1, la estimación enviada por el elemento DANE llega después del inicio de la descarga por lo que no puede ser tenida en cuenta. Por su parte, en el instante de tiempo T3 se puede ver como el cliente C2 finaliza la descarga de un segmento justo después de que el cliente C1 haga lo propio. En este caso, el elemento DANE generará dos estimaciones consecutivas que hará llegar al conjunto de los clientes.

El objetivo del DANE es mantener actualizada una estimación de la tasa de transferencia disponible, la cual se envía de forma proactiva a todos los clientes de vídeo cuando ésta se modifica. Por su parte, los clientes de vídeo, tras la descarga de un segmento, comunican su estimación de la tasa de transferencia disponible al DANE, desencadenando la actualización de dicha estimación del propio DANE.

De esta forma, cada cliente de vídeo *i*, a la hora de tomar la decisión de qué representación descargar en el instante *t*, actualiza su estimación efectiva ($\widehat{bw_i'}$) al valor menor entre su propia estimación ($\widehat{bw_i}$) y la indicada por el DANE ($\widehat{bw_{DANE}}$), siempre y cuando exista esta última:

$$\widehat{bw_i'}(t) = \begin{cases} \min\left(\widehat{bw_i}(t), \widehat{bw_{DANE}}(t)\right) & si \quad \exists\, \widehat{bw_{DANE}}(t) \\ \widehat{bw_i}(t) & si \quad \nexists\, \widehat{bw_{DANE}}(t) \end{cases}. \tag{4.1}$$

Para llegar a su estimación, el DANE calcula la media aritmética de los anchos de banda estimados por cada uno de los *N* clientes de vídeo con una reproducción en curso (Ecuación 4.2) y les aplica una minoración en función de cada uno de los estados en los que se encuentren sus buffers de reproducción según la Ecuación 4.3.

$$bw'_{DANE}(t) = \frac{1}{N}\sum_{i=1}^{N} \widehat{bw_i}(t), \tag{4.2}$$

$$bw_{DANE}(t) = \max\left[0, bw'_{DANE}(t) * \left(1 - 3 * \left(\frac{n_r}{N} + \frac{n_y}{2N}\right)\right)\right], \tag{4.3}$$

donde:

- n_y: número de clientes cuyo nivel de buffer en el instante t está entre 10 y 20 s (zona amarilla).
- n_r: número de clientes cuyo nivel de buffer en el instante t es menor de 10 s (zona roja).

Así, los clientes que estén en zona roja provocarán una reducción de la tasa de transferencia el doble que aquellos que están en zona amarilla. Según Ecuación (4.3) si un tercio de los clientes está en zona roja, o dos tercios se encuentra en zona amarilla, el valor de bw_{DANE} será 0, por lo que $\widehat{bw_t'}$ será 0 −según (4.1) −, y el algoritmo de adaptación de DASH actuará en consecuencia (eligiendo la representación de menor bitrate).

4.4. Metodología

Para evaluar la implicación de la introducción del mecanismo SAND propuesto en el escenario descrito, se ha efectuado una comparativa entre el funcionamiento del servicio de streaming con y sin el mecanismo SAND activado. Para ello, se emplean vídeos codificados a diferentes calidades utilizando el estándar DASH, concretamente tres vídeos: “Elephants Dream”, “Tears of Steel” y “Sintel”.

Los vídeos se reproducen de forma pseudo-aleatoria, de manera que cada dispositivo reproduce el mismo número de veces cada uno de los vídeos. En las pruebas realizadas, cada dispositivo reproductor realiza un total de 9 reproducciones por lo que, teniendo en cuenta que la duración aproximada de cada vídeo es de 10 minutos, la duración de las pruebas ha sido 90 minutos aproximadamente. En total, se han realizado tres pruebas de cada caso, haciendo un promedio para obtener los datos presentados en la sección de evaluación.

Los vídeos han sido codificados a 720p con H.264, utilizando CRFs entre 5 (mejor calidad) y 55 (menor calidad), en intervalos de 5, esto es, un total de 11 calidades de vídeo.

Para comprobar el funcionamiento de la propuesta en diferentes entornos, se han utilizado dos algoritmos de adaptación: el algoritmo adaptativo que incluye la librería ExoPlayer y el algoritmo Look Ahead.

Los parámetros evaluados son, al igual que en capítulos anteriores, los siguientes: 1) número de interrupciones/paradas durante la reproducción de un vídeo; 2) duración de las interrupciones; 3) representación (calidad) media de la reproducción; y 4) número de cambios de calidad, definido como número de veces que se muestra un segmento de una calidad/representación diferente a la del segmento anterior.

El entorno de pruebas está formado por: servidor, punto de acceso Wi-Fi (802.11n 5 GHz) y un panel de 36 tabletas (18 tabletas de 7 pulgadas y 18 tabletas de 9 pulgadas, como se observa en la Figura 4.4).

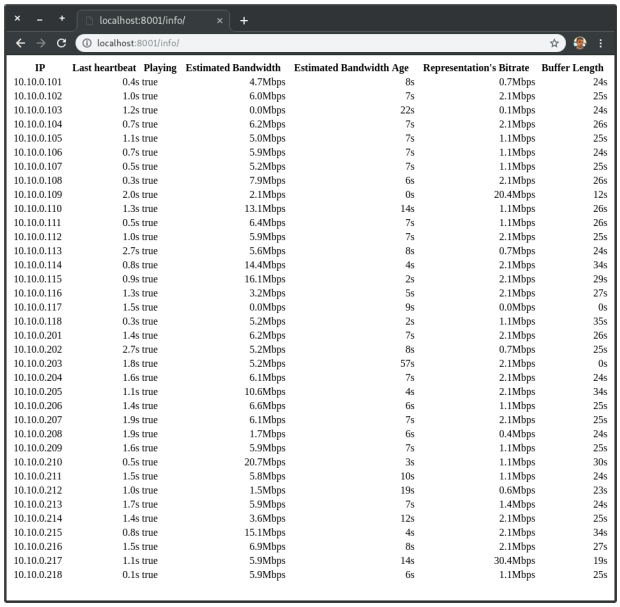
Figura 4.4. Panel de tabletas de pruebas.

El player de vídeo ha sido desarrollado a partir del proyecto ExoPlayer v2. Al iniciar, el player de vídeo busca la presencia del elemento DANE. Si lo encuentra, establece una comunicación con él a través de un WebSocket. Esta conexión será utilizada durante toda la sesión de pruebas para la comunicación de las tabletas con el DANE.

Por otro lado, para facilitar la realización de los test, se ha desarrollado una aplicación que permite la ejecución de las pruebas en las tabletas de forma desatendida. La aplicación muestra, a través de una interfaz web, información del estado de cada una de las tabletas, por ejemplo, el estado del buffer. En la Figura 4.5 se ve una captura de pantalla con la web informativa de la aplicación de pruebas. La información que se muestra es, de izquierda a derecha:

- IP de cada una de las tabletas.
- Tiempo (en segundos) que ha pasado desde que se recibió la última comunicación de una tableta en particular (*heartbeat*).
- Indicación de si la tableta está en estado de reproducción de un contenido.
- Tasa de transferencia estimada calculada por la propia tableta.
- Tiempo (en segundos) que ha pasado desde que se recibió la última comunicación de la tasa de transferencia estimada de una tableta en particular.
- Tasa de bitrate media de la representación en reproducción.
- Tamaño del buffer (en segundos) de cada cliente.

Una vez todas las tabletas han iniciado la aplicación de pruebas, se puede enviar un test al conjunto de ellas. Al recibir dicho mensaje de test, las tabletas inician la reproducción, de manera gradual, de los contenidos que se encuentran en el mensaje.



localhost:8001/info/

IP	Last heartbeat	Playing	Estimated Bandwidth	Estimated Bandwidth Age	Representation's Bitrate	Buffer Length
10.10.0.101	0.4s	true	4.7Mbps	8s	0.7Mbps	24s
10.10.0.102	1.0s	true	6.0Mbps	7s	2.1Mbps	25s
10.10.0.103	1.2s	true	0.0Mbps	22s	0.1Mbps	24s
10.10.0.104	0.7s	true	6.2Mbps	7s	2.1Mbps	26s
10.10.0.105	1.1s	true	5.0Mbps	7s	1.1Mbps	25s
10.10.0.106	0.7s	true	5.9Mbps	7s	1.1Mbps	24s
10.10.0.107	0.5s	true	5.2Mbps	7s	1.1Mbps	25s
10.10.0.108	0.3s	true	7.9Mbps	6s	2.1Mbps	26s
10.10.0.109	2.0s	true	2.1Mbps	0s	20.4Mbps	12s
10.10.0.110	1.3s	true	13.1Mbps	14s	1.1Mbps	26s
10.10.0.111	0.5s	true	6.4Mbps	7s	1.1Mbps	26s
10.10.0.112	1.0s	true	5.9Mbps	7s	2.1Mbps	25s
10.10.0.113	2.7s	true	5.6Mbps	8s	0.7Mbps	24s
10.10.0.114	0.8s	true	14.4Mbps	4s	2.1Mbps	34s
10.10.0.115	0.9s	true	16.1Mbps	2s	2.1Mbps	29s
10.10.0.116	1.3s	true	3.2Mbps	5s	2.1Mbps	27s
10.10.0.117	1.5s	true	0.0Mbps	9s	0.0Mbps	0s
10.10.0.118	0.3s	true	5.2Mbps	2s	1.1Mbps	35s
10.10.0.201	1.4s	true	6.2Mbps	7s	2.1Mbps	26s
10.10.0.202	2.7s	true	5.2Mbps	8s	0.7Mbps	25s
10.10.0.203	1.8s	true	5.2Mbps	57s	2.1Mbps	0s
10.10.0.204	1.6s	true	6.1Mbps	7s	2.1Mbps	24s
10.10.0.205	1.1s	true	10.6Mbps	4s	2.1Mbps	34s
10.10.0.206	1.4s	true	6.6Mbps	6s	1.1Mbps	25s
10.10.0.207	1.9s	true	6.1Mbps	7s	2.1Mbps	25s
10.10.0.208	1.9s	true	1.7Mbps	6s	0.4Mbps	24s
10.10.0.209	1.6s	true	5.9Mbps	7s	1.1Mbps	25s
10.10.0.210	0.5s	true	20.7Mbps	3s	1.1Mbps	30s
10.10.0.211	1.5s	true	5.8Mbps	10s	1.1Mbps	24s
10.10.0.212	1.0s	true	1.5Mbps	19s	0.6Mbps	23s
10.10.0.213	1.7s	true	5.9Mbps	7s	1.4Mbps	24s
10.10.0.214	1.4s	true	3.6Mbps	12s	2.1Mbps	25s
10.10.0.215	0.8s	true	15.1Mbps	4s	2.1Mbps	34s
10.10.0.216	1.5s	true	6.9Mbps	8s	2.1Mbps	27s
10.10.0.217	1.1s	true	5.9Mbps	14s	30.4Mbps	19s
10.10.0.218	0.1s	true	5.9Mbps	6s	1.1Mbps	25s

Figura 4.5. Web de información de estado del panel de tabletas.

4.5. Evaluación

En este apartado se analiza el funcionamiento del módulo SAND desarrollado, evaluando diferentes parámetros como el número y duración de interrupciones, la calidad media de reproducción y el número de cambios de representación.

En primer lugar, la Figura 4.6 muestra el número medio de paradas por contenido reproducido. Se puede observar que, mientras que con ExoPlayer sin usar SAND el número medio de interrupciones por cada tableta es superior a 2,5, cuando se usan mecanismos SAND el número de interrupciones es 0,6, lo que representa una reducción del 75 %, mientras que en el caso de Look Ahead la reducción es, aunque no tan elevada, bastante significativa (un 35 %).

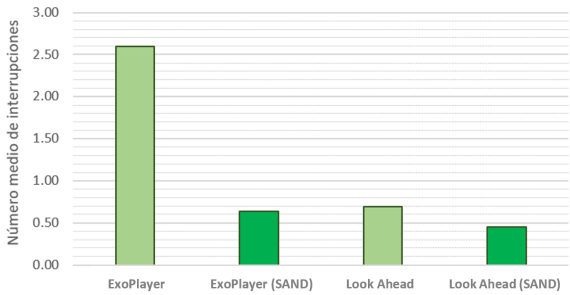


Figura 4.6. Evaluación del número de paradas.

En relación con la duración media de las paradas, la Figura 4.7 muestra la misma tendencia que con el número de paradas. De esta forma, el uso de SAND hace reducir la duración media de las paradas por reproducción de 35,4 a 9,8 segundos (-72 %) cuando se usa el algoritmo ExoPlayer, y de 12,3 a 7,7 segundos (-37 %) cuando se emplea Look Ahead.

Para reducir el número medio de interrupciones, el mecanismo SAND implementado se basa en que el servidor insta a las tabletas a que disminuyan su calidad (representación) media de reproducción, para así reducir el tráfico en la red y dar más opciones de acceso al medio a las tabletas que tienen problemas en la recepción de contenido. Por tanto, la calidad media de reproducción de los vídeos se reduce, tal como se refleja en la Figura 4.8. La figura muestra en el eje Y la representación media, que en este caso va de 0 a 10, ya que para las pruebas se han utilizado un total de 11 representaciones. Como se puede observar, aunque la calidad media se reduce, la reducción no es muy elevada (un 20 % en ExoPlayer y un 21 % en Look Ahead), sobre todo teniendo en cuenta que la reducción en términos de duración de las interrupciones ha sido mucho mayor (un 72 % en ExoPlayer y un 37 % en Look Ahead).

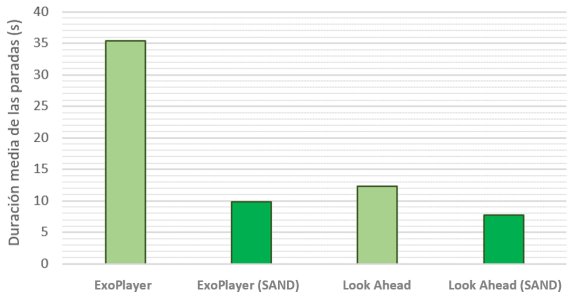


Figura 4.7. Evaluación de la duración de las paradas.

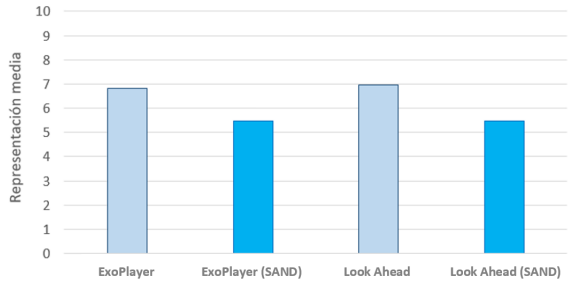


Figura 4.8. Evaluación de la representación media de la reproducción.

Finalmente, la Figura 4.9 muestra el número medio de cambios de representación por contenido reproducido. En este caso, la utilización de la propuesta basada en SAND produce un mayor número de cambios de calidad, lo cual puede empeorar la calidad de experiencia de usuario si dichos cambios son muy significativos (por ejemplo, cambiar

de una representación alta a una muy baja). Como vemos, el número de cambios de representación afecta más a ExoPlayer (25 cambios de calidad más utilizando SAND) que a Look Ahead (18 cambios).

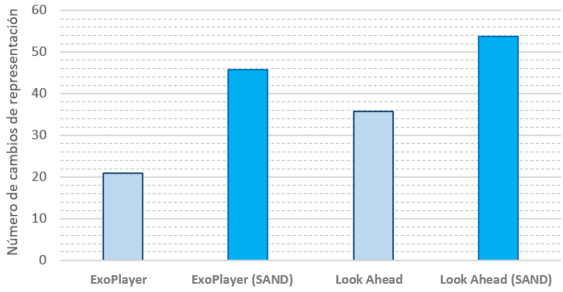


Figura 4.9. Evaluación del número medio de cambios de representación.

4.6. Conclusiones y trabajo futuro

La utilización de DASH supone un nuevo reto en escenarios con redes Wi-Fi y una alta densidad de usuarios de streaming de vídeo. Es por esto por lo que se está trabajando en la estandarización de mecanismos que permitan coordinar la reproducción del contenido, como la tecnología SAND.

En este capítulo se ha presentado un mecanismo de coordinación de clientes de vídeo, basado en SAND, que utiliza la información de estado del conjunto de los dispositivos para ofrecer una alternativa de estimación de la tasa de transferencia disponible a la calculada por los propios reproductores de vídeo. Con esta estimación, mientras unos clientes reducen su tasa de reproducción, el resto tiene más opciones de acceder al medio, mejorando así la calidad de experiencia (QoE) media de los usuarios.

La solución propuesta, a diferencia de las encontradas en la bibliografía, no requiere de modificación alguna en los algoritmos de acceso al medio ni de un control, ni por parte del servidor ni de los clientes, de la tasa de transferencia disponible percibida por los clientes de vídeo.

La evaluación llevada a cabo muestra como, en presencia del elemento DANE propuesto, se reduce de forma significativa el número de paradas en cada reproducción, así como la duración de estas. Al mismo tiempo, la representación media de las reproducciones desciende en torno a un 20%. En cuanto al número de cambios de representación, éstos también sufren un incremento debido a las indicaciones del DANE.

A pesar de la menor representación media y de un mayor número de cambios de calidad, a falta de un análisis más exhaustivo de la QoE, se puede considerar que la solución propuesta supone una mejora en la calidad percibida debido a la gran importancia que las interrupciones en la visualización de un contenido tienen sobre la QoE.

Si bien es cierto que son importantes los beneficios que proporciona la solución propuesta en el escenario indicado, no lo es menos que la propuesta está basada en las características del escenario en cuestión. Es decir, se ha diseñado en función de las velocidades de conexión, el número de puntos de acceso y el número de clientes. Con el fin de hacer la propuesta más flexible a las variaciones de estos elementos, como trabajo futuro queda profundizar en los mecanismos de estimación de las tasas de transferencias disponibles tanto de los dispositivos individuales como del conjunto o subconjuntos de estos.

Otra línea de trabajo futuro que se deriva de este trabajo es la sincronización del acceso al medio (a nivel de aplicación) por parte del elemento DANE, con el fin de evitar la pérdida de tasa de transferencia total debida a la competencia de un elevado número de dispositivos por el acceso al medio.

Referencias

[1] T. J. Barnett, S. Jain, U. Andra, and T. Khurana, “Cisco Visual Networking Index (VNI) Complete Forecast Update, 2017-2022,” available online at: https://www.cisco.com/c/dam/m/en_us/network-intelligence/service-provider/digital-transformation/knowledge-network-webinars/pdfs/1211_BUSINESS_SERVICES_CKN_PDF.pdf, published: Dec. 2018, accessed: Sep. 2020.

[2] V. Paulsamy and Z. Technologies, “Network Convergence and the NAT/Firewall Problems,” in *Proc. of the 36th Annual Hawaii Int. Conf. On System Sciences (HICSS)*, Big Island, HI, USA, Jan. 2003, pp. 1-10.

[3] International Organization for Standardization, “Information Technology - Dynamic Adaptive Streaming Over HTTP (DASH) - Part 1: Media Presentation Description and Segment Formats,” ISO/IEC 23009-1:2019, 2019.

[4] International Organization for Standardization, “Information Technology - Coding of Audiovisual Objects - Part 12: ISO Base Media File Format,” ISO/IEC 14496-12:2005, 2005.

[5] International Organization for Standardization, “Information Technology - Generic Coding of Moving Pictures and Associated Audio Information: Systems,” ISO/IEC 13818-1:2000, 2000.

[6] The WebM Project webpage, available online at: https://www.webmproject.org, accessed: Sep. 2020.

[7] J. Bankoski, P. Wilkins, and Y. Xu, “Technical overview of VP8, an open source video codec for the web,” in *Proc. Of the IEEE Int. Conf. Multimed. and Expo (ICME)*, Barcelona, Spain, Jul. 2011, pp. 1-6.

[8] Matroska Media Container webpage, available online at: https://www.matroska.org, accessed: Sep. 2020.

[9] Xiph webpage, “Vorbis audio compression,” available online at: https://xiph.org/vorbis/, accessed: Sep. 2020.

[10] Opus Interactive Audio Codec webpage, available online at: https://opus-codec.org, accessed: Sep. 2020.

[11] J. Bankoski, R. S. Bultje, A. Grange, Q. Gu, J. Han, J. Koleszar, D. Mukherjee, P. Wilkins, and Y. Xu, "Towards a next generation open-source video codec," in *Proc. of SPIE, Visual Information Processing and Communication IV*, Burlingame, CA, USA, vol. 8666, id. 866606 13, 2013.

[12] D. Grois, D. Marpe, A. Mulayoff, B. Itzhaky, and O. Hadar, "Performance comparison of H.265/MPEG-HEVC, VP9H.264/MPEG-AVC encoders," in *Proc. of the 30th Picture Coding Symposium (PCS)*, San Jose, CA, USA, 2013, pp. 394-397.

[13] Orange Open Movie Team webpage, "Elephants Dream," available online at: https://orange.blender.org/, accessed: Sep. 2020.

[14] Netflix GitHub repository, "VMAF Development Kit (VDK 1.5.3)," available online at: https://github.com/Netflix/vmaf, accessed: Sep. 2020.

[15] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara, "Toward a Practical Perceptual Video Quality Metric," available online at: https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652, published: Jun. 2016, accessed: Sep. 2020.

[16] R. Rassool, "VMAF reproducibility: Validating a perceptual practical video quality metric," in *Proc. of the IEE Int. Symp. on Broadband Multimedia Systems and Broadcasting (BMSB)*, Cagliari, Italy, 2017, pp. 1-2.

[17] T. V. Lakshman, A. Ortega, and A. R. Reibman, "VBR video: tradeoffs and potentials," in *Proc. of the IEEE*, vol. 86, no. 5, pp. 952-972, 1998.

[18] C. Müller, S. Lederer, and C. Timmerer, "An evaluation of dynamic adaptive streaming over HTTP in vehicular environments," in *Proc. of the 4th Workshop on Mobile Video (MoVid)*, Chapel Hill, NC, USA, Feb. 2012, pp. 37-42.

[19] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *Proc. of IEEE Int. Conf. On Computer Communications (INFOCOM)*, San Francisco, CA, USA, Apr. 2016, pp. 1-9.

[20] K. Spiteri, R. Sitaraman, and D. Sparacio, "From theory to practice: Improving bitrate adaptation in the DASH reference player," in *Proc. of the 9th ACM Multimed. Syst. Conf. (MMSys)*, Netherlands, Amsterdam, Jun. 2018, pp. 123-137.

[21] T. C. Thang, H. T. Le, A. T. Pham, and Y. M. Ro, "An evaluation of bitrate adaptation methods for HTTP live streaming," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 4, pp. 693-705, 2014.

[22] P. Juluri, V. Tamarapalli, and D. Medhi, "SARA: Segment-Aware Rate Adaptation Algorithm for Dynamic Adaptive Streaming over HTTP," in *Proc. of the IEEE Int. Conf. on Communication Workshop (ICCW)*, London, United Kingdom, Jun. 2015, pp. 1765-1770.

[23] Dash.js GitHub repository, "Dash-Industry-Forum/dash.js," available online at: https://github.com/Dash-Industry-Forum/dash.js/wiki, accessed: Sep. 2020.

[24] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," Morgan & Claypool, 2010.

[25] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and Adapt: Rate Adaptation for HTTP Video Streaming at Scale," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719-733, Apr. 2014.

[26] L. De Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo, "ELASTIC: A client-side controller for dynamic adaptive streaming over HTTP (DASH)," in *Proc. of the 20th Int. Pack. Video Work,* San Jose, CA, USA, Dec. 2013, pp. 1-8.

[27] C. Wang, A. Rizk, and M. Zink, "SQUAD: A Spectrum-based Quality Adaptation for Dynamic Adaptive Streaming over HTTP," in *Proc. of the 7th Int. Conf. on Multimedia Systems (MMSys)*, Klagenfurt, Austria, May 2016, pp. 1-12.

[28] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate adaptation for adaptive HTTP streaming,"in *Proc. of the Int. Conf. on Multimedia Systems (MMSys)*, Santa Clara, CA, USA, Feb. 2011, pp. 169-174.

[29] J. Jiang, V. Sekar, and H. Zhang, "Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE," in *Proc. of the 8th Int. Conf. on Emerging Networking Experiments and Technologies* (CoNEXT), New York, NY, USA, pp. 97-108.

[30] T. Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," *Comput. Commun. Rev.*, vol. 44, no. 4, pp. 187-198, 2015.

[31] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media over HTTP," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 1, pp. 562-585, 2018.

[32] Apple Developer webpage, "App Store Review Guidelines," available online at: https://developer.apple.com/app-store/review/guidelines, accessed: Sep. 2020.

[33] Google GitHub repository, "Universal DASH Transmuxer," available online at: https://github.com/google/universal-dash-transmuxer, accessed: Sep. 2020.

[34] ExoPlayer webpage, available online at https://exoplayer.dev, accessed: Sep. 2020.

[35] Android Developers Channel at YouTube, "Building feature-rich media apps with ExoPlayer," available online at https://www.youtube.com/watch?v=svdq1BWl4r8, accessed: Sep. 2020.

[36] A. Mansy, B. Ver Steeg, and M. Ammar, "SABRE: A client based technique for mitigating the buffer bloat effect of adaptive video flows," in *Proc. of the 4th ACM Multimed. Syst. Conf. MMSys*, Oslo, Norway, Feb. 2013, pp. 214-225.

[37] Blender Foundation webpage, available online at: https://www.blender.org/foundation, accessed: Sep. 2020.

[38] Ghent University, "4G/LTE Bandwidth Logs," available online at: http://users.ugent.be/~jvdrhoof/dataset-4g, accessed: Sep. 2020.

[39] Guava webpage, available online at: https://guava.dev, accessed: Sep. 2020.

[40] Instituto de Telecomunicaciones y Aplicaciones Multimedia webpage, “Look Ahead Demo,” available online at: https://lookahead.iteam.upv.es, accessed: Sep. 2020.

[41] COMM iTEAM Github repository, “ExoPlayer con Look Ahead Integrado,” available online at https://github.com/comm-iteam/ExoPlayer, accessed: Sep. 2020.

[42] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hobfeld, and P. Tran-Gia, “A Survey on Quality of Experience of HTTP Adaptive Streaming,” *IEEE Commun. Surv. Tutorials*, vol. 17, no. 1, pp. 469–492, Jan. 2015, doi: 10.1109/COMST.2014.2360940.

[43] COMM iTEAM GitHub repository, “dashgen,” available online at https://github.com/comm-iteam/dashgen, accessed: Sep. 2020.

[44] International Telecommunication Union (ITU-T), “Vocabulary for performance, quality of service and quality of experience,” Recommendation ITU-T P.10/G.100, 2017.

[45] International Telecommunication Union, “Methodology for the subjective assessment of the quality of television pictures. BT Series, Broadcasting service,” Recommendation ITU-R BT.500-14, 2019.

[46] International Telecommunication Union, “Methods for the subjective assessment of video quality, audio quality and audiovisual quality of Internet video and distribution quality television in any environment,” Recommendation ITU-T P.913, 2016.

[47] International Telecommunication Union, “Subjective Video Quality Assessment methods for multimedia applications,” Recommendation ITU-T P.910, 2008.

[48] International Telecommunication Union, “Subjective audiovisual quality assessment methods for multimedia applications,” Recommendation ITU-T P.911, 1998.

[49] L. Skorin-Kapov and M. Varela, “A multi-dimensional view of QoE: the ARCU model,” in *Proc. of the 35th Int. Conv. MIPRO*, Opatija, Croatia, May 2012, pp. 662-666.

[50] Video Quality Experts Group (VQEG), “VQEG FRTV Phase I Final Report,” available online at: https://www.its.bldrdoc.gov/vqeg/projects/frtv-phase-i/frtv-phase-i.aspx, published: 2000, accessed: Oct. 2020.

[51] Video Quality Experts Group (VQEG), “VQEG FRTV Phase II Final Report,” available online at: https://www.its.bldrdoc.gov/vqeg/projects/frtv-phase-ii/frtv-phase- ii.aspx, published: 2003, accessed: Oct. 2020.

[52] N. Barman and M. G. Martini, “QoE Modeling for HTTP Adaptive Video Streaming-A Survey and Open Challenges,” *IEEE Access*, vol. 7, pp. 30831-30859, 2019.

[53] A. Takahashi, D. Hands, and V. Barriac, “Standardization activities in the ITU for a QoE assessment of IPTV,” *IEEE Commun. Mag.*, vol. 46, no. 2, pp. 78-84, 2008.

[54] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600-612, Apr. 2004.

[55] International Telecommunication Union, "Objective Perceptual Video Quality Measurement Techniques for Digital Cable Television in the Presence of a Full Reference," recommendation ITU-T J.144, Mar. 2004.

[56] International Telecommunication Union, "Objective Perceptual Multimedia Video Quality Measurement in the Presence of a Full Reference," recommendation ITU-T J.247, Aug. 2008.

[57] International Telecommunication Union, "Objective Perceptual Multimedia Video Quality Measurement of HDTV for Digital Cable Television in the Presence of a Full Reference," recommendation ITU-T J.341, Mar. 2016.

[58] Q. Li and Z. Wang, "Reduced-reference image quality assessment using divisive normalization-based image representation," *IEEE J. Sel. Topics Signal Process.*, vol. 3, no. 2, pp. 202-211, 2009.

[59] M. Carnec, P. Le Callet, and D. Barba, "Objective quality assessment of color images based on a generic perceptual reduced reference," *Signal Processing: Image Commun.*, vol. 23, no. 4, pp. 239-256, 2008.

[60] International Telecommunication Union, "Perceptual Visual Quality Measurement Techniques for Multimedia Services Over Digital Cable Television Networks in the Presence of a Reduced Bandwidth Reference," recommendation ITU-T J.246, Aug. 2008.

[61] A. Mittal, A. K. Moorthy, and A. C. Bovik, "No-reference image quality assessment in the spatial domain," *IEEE Trans. Image Process.*, vol. 21, no. 12, pp. 4695-4708, 2012.

[62] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a 'Completely Blind' Image Quality Analyzer," *IEEE Signal Process. Lett.*, vol. 20, no. 3, pp. 209–212, 2013.

[63] M. A. Saad, A. C. Bovik, and C. Charrier, "Blind image quality assessment: A natural scene statistics approach in the DCT domain," *IEEE Trans. Image Process.*, vol. 21, no. 8, pp. 3339-3352, 2012.

[64] A. K. Moorthy and A. C. Bovik, "A two-step framework for constructing blind image quality indices," *IEEE Signal Process. Lett.*, vol. 17, no. 5, pp. 513-516, 2010.

[65] International Telecommunication Union, "Conformance Testing for Voice Over IP Transmission Quality Assessment Models," recommendation ITU-T P.564, Nov. 2007.

[66] International Telecommunication Union, "Parametric Non-Intrusive Assessment of Audiovisual Media Streaming Quality," recommendation ITU-T P.1201, Oct. 2012.

[67] International Telecommunication Union, "Opinion Model for Video-Telephony Applications," recommendation ITU-T G.1070, Jul. 2012.

[68] International Telecommunication Union, “Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport,” recommendation ITU-T P.1203, Oct. 2017.

[69] ITU-T P.1203 GitHub repository, “ITU-T Rec. P.1203 Standalone Implementation,” available online at: https://github.com/itu-p1203/itu-p1203, accessed: Sep. 2020.

[70] Telecommunication Telemedia Assessment GitHub repository, “Codec Extension for ITU-T P.1203,” available online at: at https://github.com/Telecommunication-Telemedia-Assessment/itu-p1203-codecextension, accessed: Sep. 2020.

[71] International Telecommunication Union, “Video quality assessment of streaming services over reliable transport for resolutions up to 4K,” Recommendation ITU-T P.1204, Jan. 2020.

[72] X. Yin, V. Sekar, and B. Sinopoli, “Toward a principled framework to design dynamic adaptive streaming algorithms over HTTP,” in *Proc. of the 13th ACM Workshop on Hot Topics in Networks (HotNets)*, Los Angeles, CA, USA, Oct. 2014, pp. 1-7.

[73] R. K. P. Mok, E. W. W. Chan, and R. K. C. Chang, “Measuring the quality of experience of HTTP video streaming,” in *Proc. of the 12th IFIP/IEEE Int. Symp. on Integr. Netw. Manage. (IM) Workshops*, Dublin, Ireland, May 2011, pp. 485-492.

[74] T. Hoßfeld, M. Seufert, C. Sieber, and T. Zinner, “Assessing effect sizes of influence factors towards a QoE model for HTTP adaptive streaming,” in *Proc. of the 6th Int. Workshop Qual. Multimedia Exper. (QoMEX)*, Singapore, Sep. 2014, pp. 111-116.

[75] K. D. Singh, Y. Hadjadj-Aoul, and G. Rubino, “Quality of experience estimation for adaptive HTTP/TCP video streaming using H.264/AVC,” in *Proc. of the IEEE Consumer Commun. Netw. Conf. (CCNC)*, Las Vegas, NV, USA, Jan. 2012, pp. 127-131.

[76] H. T. T. Tran, N. P. Ngoc, A. T. Pham, and T. C. Thang, “A multi-factor QoE model for adaptive streaming over mobile networks,” in *Proc. of the IEEE Globecom Workshops (GC Wkshps)*, Washington, DC, USA, Dec. 2016, pp. 1-6.

[77] Y. Liu, S. Dey, F. Ulupinar, M. Luby, and Y. Mao, “Deriving and validating user experience model for DASH video streaming,” *IEEE Trans. Broadcast.*, vol. 61, no. 4, pp. 651-665, Dec. 2015.

[78] K. Zeng, H. Yeganeh, and Z. Wang, “Quality-of-experience of streaming video: Interactions between presentation quality and playback stalling,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Phoenix, AZ, Sep. 2016, pp. 2405-2409.

[79] S. Winkler and P. Mohandas, “The evolution of Video Quality Measurement: from PSNR to hybrid metrics,” *IEEE Trans. Broadcast.*, vol. 54, no. 3, pp. 660-668, 2008.

[80] M. P. Sharabayko, O. G. Ponomarev, and R. I. Chernyak, “Intra compression efficiency in VP9 and HEVC,” *Applied Mathematical Sciences*, vol. 7, no. 137, pp. 6803-6824, 2013.

[81] H. R. Sheikh and A. C. Bovik, “Image information and visual quality,” in *IEEE Trans. on Image Process.*, vol. 15, no. 2, pp. 430-444, 2006.

[82] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, pp. 273-297, 1995.

[83] Y. Shuai and T. Herfet, “A buffer dynamic stabilizer for low-latency adaptive video streaming,” in *Proc. of the Int. Conf. on Consumer Electronics (ICCE)*, Berlin, Germany, Sep. 2016, pp. 1-5.

[84] Huawei, “Mobile Video Service Performance Study,” Huawei White Paper, available online at: http://www.ctiforum.com/uploadfile/2015/0701/20150701091255294.pdf, published: Jun. 2015, accessed: Sep. 2019.

[85] University of Waterloo webpage, “The SSIMplus Index for Video Quality-of-Experience Assessment,” available online at: https://ece.uwaterloo.ca/~z70wang/research/ssimplus, published: Nov. 2014, accessed: Sep. 2019.

[86] C. Bampis, “Measuring Video Quality with VMAF: Why you should care,” AOMedia Research Symposium, San Francisco, CA, USA, Oct. 2019.

[87] D. Ghadiyaram, J. Pan, and A. C. Bovik, “A subjective and objective study of stalling events in mobile streaming videos,” *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 29, no. 1, pp. 183-197, 2019.

[88] S. Tavakoli, S. Egger, M. Seufert, R. Schatz, K. Brunnström, and N. García, “Perceptual quality of HTTP adaptive streaming strategies: cross-experimental analysis of multi-laboratory and crowdsourced subjective studies,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 8, pp. 2141-2153, 2016.

[89] C. Moldovan, K. Hagn, C. Sieber, W. Kellerer, and T. Hoßfeld, “Keep calm and don’t switch: about the relationship between switches and quality in HAS,” in *Proc. of the Int. Teletraffic Congress (ITC)*, Genoa, Italy, Sep. 2017, pp. 1-6.

[90] C. G. Bampis and A. C. Bovik, “Feature-based prediction of streaming video QoE: Distortions, stalling and memory,” *Signal Processing: Image Communication*, vol. 68, pp. 218-228, 2018.

[91] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, “What happens when HTTP adaptive streaming players compete for bandwidth?,” in *Proc. of the Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Toronto, Canada, Jun. 2012, pp. 9-14.

[92] International Organization for Standardization, “Information technology - Dynamic adaptive streaming over HTTP (DASH) - Part 5: Server and network assisted DASH (SAND),” ISO/IEC 203009-5:2017, Feb. 2017.

[93] E. Khorov, A. Krasilov, M. Liubogoshchev, and S. Tang, “SEBRA: SAND-enabled bitrate and resource allocation algorithm for network-assisted video streaming,” in

Proc. of the Int. Conf. on Wireless and Mobile Computing, Networking and Communications (WiMob), Rome, Italy, Nov. 2017, pp. 1-8.

[94] J. W. Kleinrouweler, B. Meixner, and P. Cesar, "Improving Video Quality in Crowded Networks Using a DANE," in *Proc. of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, Taipei, Taiwan, 2017, pp. 73-78.

[95] J. W. Kleinrouweler, S. Cabrero, and P. Cesar, "Delivering stable high-quality video: an SDN architecture with DASH assisting network elements," in *Proc. of the 7th Int. Conf. on Multimedia Systems (MMSys)*, Klagenfurt, Austria, May 2016, pp. 1-10.

[96] H. Lee, Y. Go, and H. Song, "SDN-assisted HTTP adaptive streaming over Wi-Fi network," in *Proc. of the Int. Conf. on Software Defined Systems (SDS)*, Barcelona, Spain, Apr. 2018, pp. 205-210.

[97] G. Cofano, L. De Cicco, T. Zinner, A. Nguyen-Ngoc, P. Tran-Gia, and S. Mascolo, "Design and experimental evaluation of network-assisted strategies for HTTP adaptive streaming," in *Proc. of the 7th Int. Conf. on Multimedia Systems (MMSys)*, Klagenfurt, Austria, May 2016, pp. 1-12.

Anexo A. Listado de publicaciones

A continuación, se encuentra la lista de publicaciones que aportan la mayor parte del contenido de esta tesis. Se ha utilizado la siguiente notación: *R* hace referencia a artículos publicados en revistas, *C* hace lo propio con artículos presentados en congresos y *L* se refiere a capítulos de libro.

Capítulo 2

[R.1] R. Belda, I. de Fez, P. Arce, and J. C. Guerri, "Look ahead to improve QoE in DASH streaming," *Multimedia Tools and Applications*, vol. 79, pp. 25143-25170, 2020.

[C.1] R. Belda, I. de Fez, P. Arce, and J. C. Guerri, "Algoritmo de adaptación DASH sensible al tamaño del segmento," in *Proc. of the Simposium Nacional de la Unión Científica Internacional de Radio (URSI)*, Granada (Spain), Sep. 2018, article S7.1.3.

[C.2] R. Belda, I. de Fez, P. Arce, and J. C. Guerri, "Look Ahead: a DASH adaptation algorithm," in *Proc. of the IEEE Int. Symp. On Broadband Multimedia Systems and Broadcasting (BMSB)*, Valencia (Spain), Jun. 2018, article no. 158.

Capítulo 3

[R.2] I. de Fez, R. Belda, and J. C. Guerri, "New objective QoE models for evaluating ABR algorithms in DASH," *Computer Communications*, vol. 158, pp. 126-140, 2020.

Capítulo 4

[C.3] R. Belda, I. de Fez, P. Arce, and J. C. Guerri, "Transmisión inalámbrica multimedia coordinada con DASH-SAND," in *Proc. of the Simposium Nacional de la Unión Científica Internacional de Radio (URSI)*, Málaga (Spain), Sep. 2020.

Adicionalmente, el autor de esta tesis ha participado en las siguientes publicaciones:

[R.3] R. Belda, I. de Fez, F. Fraile, P. Arce, and J. C. Guerri, "Hybrid FLUTE/DASH video delivery over mobile wireless networks," *Transactions on Emerging Telecommunications Technologies*, vol. 25, no. 11, pp. 1070-1082, 2014.

[R.4] F. Fraile, N. Blasco, I. de Fez, R. Belda, P. Arce, and J. C. Guerri, "Study and evaluation of four video codecs -H.264, H.265, VP8 and VP9- for public transport entertaining systems," *Waves 2016 (iTEAM UPV Journal)*, vol. 8, pp. 5-15, 2016.

[R.5] P. Acelas, P. Arce, W. Castellanos, R. Belda, I. de Fez, F. Fraile, V. Murcia, T. R. Vargas, M. Monfort, and J. C. Guerri, "An interactive wireless-based telemedicine system for back care applications," *Waves 2010 (iTEAM UPV Journal)*, vol. 2, pp. 56-65, 2010.

[R.6] I. de Fez, F. Fraile, R. Belda, and J. C. Guerri, "Performance evaluation of AL-FEC LDPC codes for push content applications in wireless unidirectional environments," *Multimedia Tools and Applications*, vol. 60, no. 3, pp. 669-688, 2012.

[R.7] R. Belda, P. Arce, I. de Fez, F. Fraile, and J. C. Guerri, "Android real-time audio communications over local wireless," *Waves 2012 (iTEAM UPV Journal)*, vol. 4, 2012.

[R.8] I. de Fez, F. Fraile, R. Belda, and J. C. Guerri, "Analysis and evaluation of adaptive LDPC AL-FEC codes for content download services," *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 641-650, 2012.

[R.9] A. Miquel, R. Belda, I. de Fez, P. Arce, F. Fraile, J. C. Guerri, F. Martínez, and S. Gallardo, "A power consumption monitoring, displaying and evaluation system for home devices," *Waves 2013 (iTEAM UPV Journal)*, vol. 5, pp. 5-13, 2013.

[R.10] P. Arce, I. de Fez, R. Belda, and J. C. Guerri, "Proxy-based near real-time TV content transmission in mobility over 4G with MPEG-DASH transcoding on the cloud," *Multimedia Tools and Applications*, vol. 78, no. 18, pp. 26399-26425, 2019.

[C.4] R. Belda, I. de Fez, F. Fraile, V. Murcia, P. Arce, and J. C. Guerri, "Multimedia System for Emergency Services over TETRA-DVBT Networks," in *Proc. of the 34th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*, Parma (Italy), Sep. 2008, pp. 142-149.

[C.5] R. Belda, V. Murcia, F. Fraile, P. Arce and J. C. Guerri, "Video On Demand Services for Emergency Scenarios Over Heterogeneous TETRA-DVBT Networks," in *Proc. Wireless Personal Multimedia Communications Symposium (WPMC)*, Lapland (Finland), Sep. 2008.

[C.6] R. Belda, J. M. Oztarman, I. de Fez, and J. C. Guerri, "Señalización SMS para el Acceso a Redes Sociales," presented at the Telecom I+D 2010, Valladolid (Spain), Sep. 2010.

[C.7] I. de Fez, F. Fraile, R. Belda, and J. C. Guerri, "Implementación y evaluación de la codificación LDPC para la transmisión de ficheros en entornos

unidireccionales," in *Proc. of Jornadas de Ingeniería Telemática (JITEL)*, Valladolid (Spain), Sep. 2010, pp. 229-236.

[C.8] F. Fraile, I. de Fez, R. Belda, and J. C. Guerri, "Evaluation of a background push download service for personal multimedia devices," in *Proc. International Conference in Consumer Electronics (ICCE)*, Las Vegas (USA), Jan. 2011, pp. 231-232.

[C.9] T. R. Vargas, P. Arce, I. de Fez, V. Murcia, F. Fraile, R. Belda, P. Acelas, and J. C. Guerri, "Solutions to improve the video streaming service over heterogeneous networks," in *Proc. of the 1st Workshop on Future Internet: Efficiency in High-Speed Networks (W-FIERRO)*, Cartagena (Spain), Jul. 2011, pp. 85-92.

[C.10] I. de Fez, F. Fraile, R. Belda, and J. C. Guerri, "Evaluation of adaptive LDPC AL-FEC codes for content download services," in *Proc. of the IEEE International Conference on Multimedia and Expo (ICME)*, Barcelona (Spain), Jul. 2011, pp. 1-6.

[C.11] N. Blasco, M. de Diego, R. Belda, I. de Fez, P. Arce, F. J. Martínez-Zaldívar, A. González, and J. C. Guerri, "Distributed sensor network for noise monitoring in industrial environment with Raspberry Pi," in *Proc. of the Int. Conf. on Intelligent Systems and Applications (INTELLI-IAP)*, Nice (France), Jul. 2017, pp. 51-55.

[L.1] F. Fraile, P. Arce, R. Belda, I. de Fez, J. C. Guerri, and A. Pajares, "Social Aware TV Content Delivery over Intelligent Networks," *Social Media Retrieval*, Ed. Springer, pp. 373-391, 2013.

Anexo B. Listado de proyectos

Listado de proyectos de investigación e innovación en los que el autor de esta tesis ha participado:

[P.1] Modela-TV: Personalización de servicios y gestor de modelos de negocio en TV móvil. FIT-330300-2007-15.

[P.2] MIQUEL: Sistema multimedia sobre entornos inalámbricos aplicado a los trastornos del sistema músculo-esquelético. TEC-2007-68119-C02-01/TCM.

[P.3] RAUDO: Red interactiva multiplataforma de distribución de contenidos audiovisuales. TSI-020302-2008-115.

[P.4] Redes híbridas para la provisión de servicios turísticos. TSI-020302-2008-94 / TSI-020302-2010-165.

[P.5] AV-MOV: Contenidos multimedia para el transporte multimodal. TSI-020301-2009-11.

[P.6] GrafiTV: Sistema de información interactiva y personalizada sobre contenidos audiovisuales. TSI-090100-2011-173.

[P.7] HAUS: Hogar digital y contenidos audiovisuales adaptados a los usuarios. IPT-2011-1049-4300000-AR.

[P.8] IMMERSIVE TV: Una aproximación a los medios inmersivos. TSI-020302-2010-6.

[P.9] COMINN: Centro comercial interactivo con interacción natural. IPT-2012-0883-430000.

[P.10] Aplicación para comunicaciones por voz a través de bluetooth y Wi-Fi en dispositivos con sistema operativo Android. Proyectos de prueba de concepto, INNOVA 2012 (Universitat Politècnica de València).

[P.11] SONDIVIBI: Dispositivo de monitorización y evaluación de la calidad de señal en sistemas de televisión digital. Proyectos de prueba de concepto, INNOVA 2014 (Universitat Politècnica de València).

[P.12] Distribución escalable de contenidos multimedia adaptados a las preferencias y contexto de los usuarios. Proyectos de investigación multidisciplinares 2012 (Universitat Politècnica de València). PAID-05-12.

[P.13] Plataforma avanzada de conectividad en movilidad. CDTI IDI-20150126.

[P.14] Smart Sound Processing for the digital living. TEC2015-67387-C4-1-R.

[P.15] Desarrollo de una plataforma de entretenimiento para entornos náuticos. CDTI TIC-20170102.