UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Implementación de una herramienta de evaluación de QoE automatizada y comparativa con evaluaciones subjetivas de usuarios reales.

Trabajo Fin de Grado

Grado en Tecnología Digital y Multimedia

AUTOR/A: Monteserín Placer, Lara

Tutor/a: Arce Vila, Pau

CURSO ACADÉMICO: 2022/2023

Resumen

El consumo de streaming de vídeo no deja de aumentar año tras año y el concepto de QoE cobra cada vez más importancia para los proveedores de red y de servicio, los fabricantes de los contenidos y los usuarios. Crear modelos que sean capaces de estimar con el mínimo error posible la calidad de la experiencia de visionado del usuario es difícil por el aumento de la complejidad en la tecnología, pero también muy útil.

En el presente trabajo se analizan modelos existentes en la actualidad para calcular la calidad de experiencia (QoE) en servicios de streaming adaptativo de vídeo sobre HTTP (HAS). Para ello, se han creado versiones distorsionadas de contenidos de vídeo de diferentes tipos simulando varios escenarios de condiciones de red, con cambios de calidad e interrupciones. A partir de dichos vídeos, se han considerado dos modelos distintos para la estimación de la calidad de experiencia.

En primer lugar, se ha estudiado la recomendación de la Unión Internacional de Telecomunicaciones (ITU). El modelo P.1203, específicamente diseñado para HAS, permite la evaluación subjetiva de forma automática de la QoE, sin la participación de usuarios. En segundo lugar, se ha diseñado una experiencia de visionado a través de una web para usuarios reales en la que se recogen sus valoraciones como método de modelado subjetivo de la calidad de experiencia.

Los resultados de cada modelo se consideran de forma independiente y también comparativa. Se proponen recomendaciones para la elaboración de futuros modelos apoyándose en el análisis de los resultados obtenidos.

Summary

The consumption of streaming video is increasing year after year and the concept of QoE is becoming more and more important for network and service providers, content manufacturers and users. Creating models that are able to estimate the quality of the user's viewing experience with as little error as possible is difficult due to the increasing complexity of the technology, but also very useful.

This paper evaluates existing models for calculating the quality of experience (QoE) in HTTP adaptive video over HTTP (HAS) streaming services. To this end, distorted versions of video content of different types have been created by simulating various scenarios of network conditions, quality changes and interruptions. From these videos, two different models for estimating the quality of experience were considered.

First, the International Telecommunication Union (ITU) recommendation has been studied. The P.1203 model, specifically designed for HAS, allows the automatic subjective evaluation of QoE, without the participation of users. Secondly, a web-based viewing experience has been designed for real users in which their ratings are collected as a method of subjective modelling of the quality of experience.

The results of each model are considered independently and comparatively. Recommendations for future modelling are proposed based on the analysis of the results obtained.

ÍNDICE DE CONTENIDOS

1. INTRODUCCIÓN 8
   1.1. Motivación del trabajo 10
   1.2. Fundamentación teórica 10
      1.2.1. Terminología y conceptos básicos 10
      1.2.2. Tecnologías de streaming sobre HTTP 11
         1.2.2.1. Tecnología de streaming adaptativo sobre HTTP (HAS) 12
         1.2.2.2. Tecnología de streaming adaptativo dinámico sobre HTTP (DASH) 13
      1.2.3. Calidad de servicio (QoS) y calidad de experiencia (QoE) 13
         1.2.3.1. Calidad de servicio (QoS) 13
         1.2.3.2. Calidad de experiencia (QoE) 14
      1.2.4. Métricas para la evaluación de la calidad de vídeo 14
         1.2.4.1. Métricas objetivas 14
         1.2.4.2. Métricas subjetivas 15
         1.2.4.3. Métricas híbridas 16
      1.2.5. Factores de influencia (IF) en la QoE 16
         1.2.5.1. Factores de influencia más importantes en aplicaciones HAS 17
      1.2.6. Gestión de la calidad de experiencia: Modelado de QoE 17
         1.2.6.1. Criterios de evaluación del rendimiento de un modelo de QoE 18
   1.3. Estado del arte 18
      1.3.1. Tipos de modelo de QoE 18
         1.3.1.1. Clasificación de modelos de QoE 18
         1.3.1.2. Modelos de QoE para HAS 21
   1.4. Metodología 22
      1.4.1. Estructura del trabajo 22
      1.4.2. Herramientas utilizadas 24

2. EXTRACCIÓN DE DATOS Y GENERACIÓN DE VÍDEOS 27
   2.1. Selección de vídeos referencia/secuencias de prueba 27
      2.1.1. Análisis SI/TI de vídeos 27

2.2. Selección de parámetros de codificación de los vídeos y generación de vídeos de distintas calidades con FFMPEG 28
2.3. Selección de los vídeos con mejor relación PSNR-bitrate 29
2.4. Segmentación de los vídeos para usarlos en HAS 32
2.5. Creación de servidor web 32
2.6. Desarrollo de la página y el reproductor web 33
2.7. Creación de los vídeos distorsionados 34
2.7.1. Puppeteer 34
2.7.2. Diseño de escenarios de red 35
2.7.3. Desarrollo del script usando Puppeteer 37
2.7.4. Parseo de los datos obtenidos 39
2.7.5. Reconstrucción de los vídeos distorsionados 40

3. MODELADO HÍBRIDO. RECOMENDACIÓN ITU P.1203 43
3.1. Recomendación P.1203 de la ITU-T 43
3.2. Uso del repositorio ITU-T Rec. P.1203 Standalone Implementation 44

4. MODELADO SUBJETIVO. EXPERIMENTO DE VISIONADO 46
4.1. Selección del grupo de visionado y diseño del experimento 46
4.2. Elaboración de los HTML y la hoja de estilos CSS asociada 46
4.2.1. Estructura de index.html 46
4.2.2. Estructura de las páginas con vídeos embebidos: de pagina2.html a pagina17.html 47
4.2.3. Estructura del cuestionario: formulario.html 48
4.2.4. Estructura de la última página: final.html 48
4.3. Elaboración del archivo JS para servir los contenidos y conectar con MySQL 49
4.4. Elaboración de la tabla para guardar los resultados en MySQL 51
4.5. Ejecución completa de un test desde el lado del desarrollador en local 52
4.6. Despliegue de la web usando un dominio de la UPV 56
4.7. Visualización de resultados almacenados en la BBDD 57

5. ANÁLISIS DE RESULTADOS 58
5.1. Presentación de los resultados 58
5.2. Cálculo de parámetros asociados a los resultados 61
5.3. Análisis de resultados del modelo híbrido 62
5.4. Análisis de resultados del modelo subjetivo 62

5.5. Análisis global 63
5.5.1. Análisis por escenario 64
5.5.2. Análisis por vídeo 65
5.5.3. Efecto de los factores de influencia 65

6. CONCLUSIONES 71
6.1. Aplicación de la fundamentación teórica al trabajo práctico 71
6.2. Valorción general 71
6.3. Limitaciones de la investigación y recomendaciones a futuro 72

7. BIBLIOGRAFÍA 74
8. ANEXO I 77
9. ANEXO II 102

ÍNDICE DE FIGURAS

Figura 1. Funcionamiento de la tecnología HAS 12
Figura 2. Relación entre los conceptos de QoS y QoE 14
Figura 3. Fases de la gestión de QoE 18
Figura 4. Tipos de modelo de QoE 19
Figura 5. Algunos de los modelos de evaluación de calidad más usados para imagen y vídeo 21
Figura 6. Descomposición en pasos del procedimiento a seguir en el presente trabajo 23
Figura 7. Gráfica SI/TI con los valores promedio de cada vídeo 28
Figura 8. Relación PSNR-bitrate vídeos codificados a partir de animacion.mp4 30
Figura 9. Relación PSNR-bitrate vídeos codificados a partir de noticias.mp4 30
Figura 10. Relación PSNR-bitrate vídeos codificados a partir de deportes.mp4 31
Figura 11. Relación PSNR-bitrate vídeos codificados a partir de serie.mp4 31
Figura 12. Contenido de la carpeta de archivos servidos con Apache 33
Figura 13. Aspecto de index.html 33
Figura 14. Aspecto de player.html reproduciendo deportes.mp4 34
Figura 15. Perfil de cambios de ancho de banda del escenario 1 35
Figura 16. Perfil de cambios de ancho de banda del escenario 2 36
Figura 17. Perfil de cambios de ancho de banda del escenario 3 36
Figura 18. Contenido de escenario1.json 37
Figura 19. Extracto de la consola durante una ejecución del script 38
Figura 20. Extracto de la consola con las estadísticas de la ejecución 39

Figura 21. Puppeteer controlando Chrome durante la ejecución de trazas.js 39
Figura 22. Proceso de reconstrucción de los vídeos distorsionados 42
Figura 23. Bloques del modelo ITU-T P.1203 43
Figura 24. Ejecución del modo 0 para uno de los vídeos 45
Figura 25. Ejecución del modo 3 para uno de los vídeos 45
Figura 26. Aspecto de index.html del test 47
Figura 27. Aspecto de pagina2.html 48
Figura 28. Aspecto de formulario.html. Parte 1 49
Figura 29. Aspecto de formulario.html. Parte 2 49
Figura 30. Aspecto de final.html 50
Figuras 31 y 32. Conexión de la base de datos con el servidor MySQL 51
Figura 33. Estructura de la tabla “puntuaciones” que guarda los valores del test 52
Figura 34. Lanzamiento de app.js 52
Figura 35. Ejecución de la web con el test y app.js en index.html 53
Figura 36. Ejecución de la web con el test y app.js en pagina3.html 53
Figura 37. Zoom de la Figura 35 54
Figura 38. Ejecución de la web con el test y app.js en final.html 55
Figura 39. Zoom de la Figura 37 55
Figura 40. Nueva fila generada en la tabla puntuaciones de mydatabase tras la ejecución completa de un test en local 56
Figura 41. Ejecución simultánea de los contenedores app_1 y db_1 con la respuesta a la inserción del parámetro guardarCSV 57
Figura 42. Correlación entre los resultados de ambos modelos 64

ÍNDICE DE TABLAS

Tabla 1. Comparación de soluciones de streaming adaptativo sobre HTTP 12
Tabla 2. Escala de evaluación de MOS 15
Tabla 3. Características de las secuencias de prueba 28
Tabla 4. Resultados ITU-T P.1203 para los 16 vídeos origen 58
Tabla 5. Resultados del test subjetivo ordenados 60
Tabla 6. Datos a utilizar del modelo híbrido. Resultados del modo 3 del modelo P1203.1. 61
Tabla 7. Datos a utilizar del modelo objetivo. Media, mediana y desviación estándar de las puntuaciones de los usuarios. 62
Tabla 8. Diferencia de resultados entre modelos 63
Tabla 9. Número de interrupciones generadas en cada vídeo y duración de las mismas 66

Tabla 10. Duración de los retardos iniciales de la carga de vídeos 67
Tabla 11. Media de las puntuaciones para los primeros y los últimos vídeos del test 68
Tabla 12. Puntuaciones medias de los vídeos y respuestas al formulario de cada espectador. 69
Tabla 13. Correspondencia entre las preguntas del test que contienen vídeos y los vídeos distorsionados 102
Tabla 14. Resultados obtenidos del test subjetivo para los 16 vídeos origen en la BBDD 103

PARTE I: INTRODUCCIÓN

1.1 Motivación del trabajo

En el año 2016, ya se preveía que el vídeo ocupase un 82% del tráfico de internet para 2021 [10]. El informe global publicado por Sandvine a principios de este 2023 indica que el 54% del tráfico total de internet había correspondido únicamente a vídeos en streaming en 2022, un 24% más que en 2021 [11].

En los últimos años, los avances que se han ido dando en las tecnologías de streaming han provocado un aumento del vídeo bajo demanda (VoD) y el vídeo en directo (Live). Aunque es cierto que se han desarrollado nuevas tecnologías inalámbricas (como el 5G), las tecnologías de red aún no están capacitadas para cumplir con la demanda de ancho de banda de los usuarios, que quieren disfrutar del contenido en cualquier momento y lugar. Por ello, para reducir los costes de transporte y el ancho de banda, es imprescindible aplicar una compresión con pérdidas a los datos multimedia en internet [2].

El streaming de vídeo era un mercado pequeño en sus inicios, y era suficiente con ofrecer el servicio. Sin embargo, ahora que ya hay muchas opciones entre las que los usuarios pueden decidir, es clave para las empresas asegurarse de que satisfacen las necesidades y expectativas de los clientes si quieren mantener su éxito.

Motivado por todo lo anterior, el concepto de calidad de experiencia (Quality of Experience o QoE) ha cobrado especial importancia en los últimos años. Lograr una buena QoE es complejo, porque depende de múltiples factores: dispositivos distintos (tamaño de pantalla, recursos computacionales, capacidad de almacenamiento, etc.), contenidos multimedia y condiciones de red cambiantes, cambios en el rendimiento de las CDNs (Content Distribution Networks, concepto explicado en el apartado 1.2.1), etc.

En las tecnologías de streaming tradicional es común encontrar errores de transmisión (jitter, retardo, pérdida de paquetes, etc.) que provocan molestias en la QoE del usuario. Como es prácticamente imposible prestar un servicio sin elementos que resulten molestos, el desarrollo de modelos de QoE puede ayudar a cuantificar la cantidad y el tipo de distorsiones, y la magnitud de su efecto en la QoE del usuario final. A su vez, puede servir para diseñar estrategias adecuadas que ayuden a eliminar o a reducir lo máximo posible dichas molestias. Los modelos deben ser fiables y precisos, y suelen aprovechar factores medidos a nivel de red y de aplicación (muchos de ellos, factores de calidad de servicio/ QoS) [3].

Sin embargo, los modelos objetivos de QoE no están diseñados para estimar la calidad a largo plazo, porque se diseñaron para estimar las deficiencias que eran consecuencia de la compresión o la pérdida de paquetes (normalmente, aquellas relacionadas con el proceso de transmisión) [2]. Estos modelos no tienen en cuenta deficiencias como el rebuffering o los cambios de calidad que sí ocurren cuando hablamos de aplicaciones HAS (HTTP Adaptive Streaming, concepto explicado ampliamente en el apartado 1.2.2.1). Se necesitan, por tanto, modelos de estimación de QoE para aplicaciones HAS que tengan en cuenta otros factores de influencia (IF) como los mencionados, además de los fallos derivados de la codificación con pérdidas.

Un buen modelo de QoE para HAS no solo sería interesante para la literatura, sino que en la práctica también reviste gran interés para los agentes económicos e industriales del sector multimedia. Entender las expectativas y la experiencia de los usuarios a través de la identificación de KPIs es fundamental para desarrollar futuras tecnologías y servicios y mejorar los existentes. El rendimiento del modelo decide la fiabilidad y la precisión de los siguientes pasos del proceso de gestión de QoE, como se verá en el apartado 1.2.4. La aplicabilidad varía en función de quién sea la parte interesada.

Los principales grupos que resultan beneficiados de la implantación de modelos de QoE para HAS serían los miembros de la cadena de procesado del contenido multimedia [2]: operadores de red, proveedores de servicio, fabricantes de los dispositivos y, por supuesto, usuarios finales.

- Con la creciente demanda de servicios OTT, los operadores de red tienen mucha presión para proporcionar una conectividad sin fisuras y de alta calidad. Los modelos de QoE pueden servirles para identificar los factores de influencia (IF) clave (que suelen ser los más relacionados con la QoS) y su respectivo impacto en la QoE, y por tanto tomar las acciones necesarias en cuanto a la asignación de recursos: codificación, almacenamiento en caché, balanceo de carga, estrangulamiento del ancho de banda, aprovisionamiento, etc. Así, optimizan el proceso de entrega mientras ofrecen una QoE razonable.

- Los proveedores de servicio o de contenidos OTT no pueden confiar únicamente en generar beneficios con la propia prestación del servicio. El entorno actual es muy competitivo, y con precios muy similares han de valorar los factores que pueden llevar al usuario a cambiarse a la competencia. Un factor de gran interés para los servicios de publicidad o de suscripción es, por ejemplo, la duración de visionado. Además, como los servicios HAS necesitan mucho espacio adicional para almacenar varias copias de cada archivo en el servidor, la optimización del bitrate puede ser otro factor clave para ahorrar espacio de almacenamiento y reducir la demanda de ancho de banda. Otros factores importantes podrían ser también el tamaño de segmento, la frecuencia y la magnitud de adaptación del reproductor o la popularidad del vídeo. Un modelo de QoE adecuado puede dar también una visión del impacto de estos factores para ahorrar costes de almacenamiento, optimizar la caché de vídeo, rediseñar el algoritmo de adaptación el cliente, etc.; además de garantizar la máxima QoE posible al usuario final.

- Los fabricantes ofrecen dispositivos distintos. La calidad percibida en cada dispositivo depende de varios factores, entre ellos el tamaño de la pantalla, que a su vez condiciona la capacidad de procesamiento. Un buen modelo de QoE puede valorar el impacto de cada característica del dispositivo (resolución, CPU, RAM, etc.) o de los códecs en la QoE, con el propósito de optimizarla.

- El éxito de los modelos y, en general, de la gestión de la QoE es de lo que depende la aceptación del servicio y la satisfacción por parte del usuario final, y también que aumente su disposición a adoptar servicios nuevos.

1.2 Fundamentación teórica

1.2.1 Terminología y conceptos básicos

A continuación se presentan las definiciones de algunos conceptos en el contexto en el que se tratan en el presente trabajo:

- **Ancho de banda**: Cantidad máxima de datos que pueden ser transmitidos a través de una conexión a Internet en un tiempo determinado.
- **Bitrate**: Cantidad de bits que se utilizan para representar un segundo de información de audio, video o cualquier otro tipo de archivo digital.
- **Buffer**: Espacio de memoria en el que se almacenan datos de manera temporal.
- **CDN (Content Distribution Network)**: Conjunto de servidores colocados en diferentes puntos de una red que contienen copias locales de ciertos contenidos (vídeos, imágenes, música, documentos, webs, etc.) que están almacenados en otros servidores generalmente alejados geográficamente, de forma que sea posible servir dichos contenidos de manera más eficiente.
- **Códec**: Programa, algoritmo o dispositivo que posibilita la codificación y decodificación de datos.
- **CRF (Constant Rate Factor)**: Parámetro para la compresión de vídeo que determina la calidad del vídeo codificado. Cuanto mayor sea su valor, mayor será la compresión y menores la calidad y el tamaño del vídeo resultante. El rango de valores está comprendido entre 0 y 51.
- **Frame**: Cada uno de los fotogramas que componen un contenido de vídeo.
- **GoP (Group of Pictures):** Estructura en la que se organizan los frames de tipo I, P y B de un vídeo codificado.
- **HTTP (Hypertext Transfer Protocol)**: Protocolo a través del que se realizan peticiones de datos de páginas web a través de internet.
- **Jitter**: Variación en el retardo durante una transmisión
- **Live Streaming**: Contenido multimedia en streaming que se transmite a través de internet en tiempo real o prácticamente en tiempo real. Se emite a criterio del servidor, que es quien controla la transmisión. Puede estar emitiéndose de forma simultánea a la grabación del contenido o en diferido.
- **Paquete**: Unidad mínima y fundamental de transporte de información en internet
- **QP (Quantization Parameter)**: Parámetro de cuantización en codificación de vídeo. Cuanto mayor sea su valor, mayor es la pérdida de calidad y menor el tamaño del vídeo codificado.
- **Rebuffering**: Parada no deseada en la reproducción de un vídeo que ocurre cuando no quedan datos en el buffer.

- **Retardo**: Retraso en la transmisión desde que la información se envía desde el origen hasta que llega a destino a través de la red.
- **Servicios OTT (Over The Top):** Audio, vídeo y otros contenidos disponibles a través de la conexión a internet sobre los que los proveedores tradicionales de internet no tienen influencia ni en el control ni en la distribución del contenido, como la televisión por cable o satélite.
- **TCP (Transmission Control Protocol):** Protocolo estándar para el envío de datos a través de la red que asegura una transmisión fiable a través de un enfoque orientado a la conexión entre emisor y receptor, en el que el receptor informa del recibo o de la necesidad de retransmisión de los paquetes erróneos o perdidos.
- **Throughput**: Cantidad de información real que se entrega en una cantidad de tiempo concreta teniendo en cuenta todos los factores adversos.
- **VoD (Video on Demand)**: Servicio OTT que permite al usuario visualizar un contenido concreto que está almacenado en un servidor en el momento que lo desee. El servidor espera a una petición del cliente, que es quien controla la transmisión.

1.2.2 Tecnologías de streaming sobre HTTP

El streaming es la tecnología que permite la transmisión de audio y vídeo a través de internet u otra red sin tener que descargar previamente los datos al dispositivo desde el que se visualiza el archivo, ya sea en directo o en diferido. La reproducción se inicia cuando se tienen los datos necesarios, y, a medida que la información (los paquetes) se va recibiendo, es posible ir reproduciendo sin necesidad de esperar a que todos los paquetes estén disponibles.

El cliente se conecta con el servidor a través de un dispositivo móvil, proporcionando la interfaz necesaria para la interacción. Una vez conectado, empieza a recibir paquetes, que va almacenando en un buffer. Cuando ha almacenado información suficiente, comienza la reproducción y continúa con la descarga de datos de manera simultánea. La tecnología tiene la capacidad de usar la información almacenada en caso de que la velocidad de descarga se reduzca, con el objetivo de evitar interrupciones. En caso de que el buffer se quede sin datos, ocurre el evento de *rebuffering* y la reproducción se corta hasta que vuelva a almacenarse la cantidad de información suficiente para continuar.

El vídeo en streaming puede reproducirse en todo tipo de aplicaciones, desde reproductores multimedia hasta páginas web que incluyan reproductores embebidos. Los clientes no están sujetos a ningún sistema operativo para poder reproducir los datos, porque la gran mayoría tienen aplicaciones para reproducir contenido en streaming [4].

Las tecnologías de streaming han evolucionado desde la descarga de ficheros, pasando por la descarga progresiva hasta el streaming adaptativo. En el modelo de descarga progresiva, el cliente va almacenando el vídeo en caché cuando las condiciones de red son favorables para compensar las ocasiones en las que no lo sean, debido al aumento del tráfico, la cobertura, las variaciones del canal según se muevan los usuarios, etc. El cliente puede empezar la reproducción antes de que todo el archivo se haya descargado, pero si el throughput disponible

es más bajo que el bitrate, ocurrirá un evento de *rebuffering.* Esta descripción coincide con la del funcionamiento del streaming vista previamente.

Sin embargo, a partir de 2016 varios informes [12] demostraron que un 99% del streaming bajo demanda basado en HTTP ocupaba alrededor del 99% del tráfico móvil. Con el incremento de la demanda de streaming, el *rebuffering* era cada vez más común y se buscaba una solución para una reproducción más fluida.

1.2.2.1 Tecnología de streaming adaptativo sobre HTTP (HAS)

La tecnología de streaming adaptativo sobre HTTP (HAS) se adapta a las condiciones disponibles de la red. El vídeo se codifica en muchas calidades (con bitrates o resoluciones distintas) que reciben el nombre de representaciones. Dichas representaciones se segmentan en intervalos cortos, típicamente de entre 2 y 10 segundos, que reciben el nombre de segmentos. El cliente va pidiendo periódicamente segmentos de vídeo al servidor, que va decodificando y reproduciendo. Dependiendo del ancho de banda y del throughput, el cliente puede cambiar entre representaciones en cada petición. El funcionamiento de la tecnología HAS se refleja en la Figura 1. El cliente, en función de su estado de red, adapta la calidad del vídeo para proporcionar una experiencia de streaming fluida al usuario final. [2]

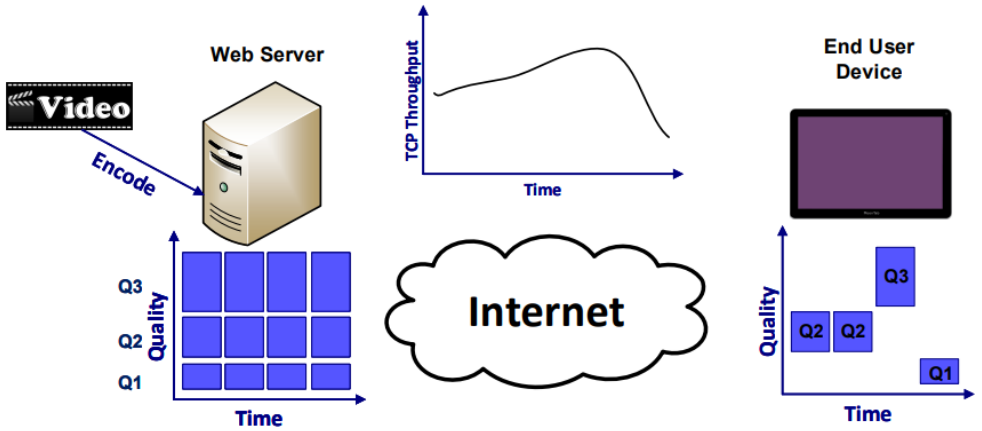


Figura 1. Funcionamiento de la tecnología HAS [2]

Como ventajas principales, HAS ofrece una transmisión fiable, capacidad de reutilización de la infraestructura de caché y permite atravesar cortafuegos [18]. Por ello, se ha convertido en la solución estándar para el streaming multimedia en internet. Algunas de sus principales implementaciones son Adobe HTTP Dynamic Streaming (HDS), Apple HTTP Live Streaming (HLS), Microsoft Smooth Streaming (MSS) y Dynamic Adaptive Streaming over HTTP (DASH). Todas comparten una lógica común, aunque hay variaciones en su manifiesto, el tamaño recomendado de segmento, etc. Un resumen se muestra en la Tabla 1.

Empresa	Códec	Longitud de segmento	Manifiesto	Formato
Microsoft Corporation	H.264, VC-1	2 segundos	Manifest (XML)	fMP4
Apple	H.264	10 segundos	Playlist (M3U8)	M2TS
Adobe Systems	H.264, VP6	2-5 segundos	Manifest (F4M)	fMP4
Estándar MPEG	Varios	Sin especificar	MPD (XML)	MP4 o M2TS

Tabla 1. Comparación de soluciones de streaming adaptativo sobre HTTP.

1.2.2.2 Tecnología de streaming adaptativo dinámico sobre HTTP (DASH)

En la tecnología de streaming adaptativo dinámico sobre HTTP (DASH), el video se codifica con diferentes niveles de representación y luego se divide en chunks de duraciones iguales que se almacenan en un servidor, como en el resto de tecnologías HAS. Muchos distribuidores OTT hacen el proceso inverso, segmentan y luego codifican para acelerar el proceso de codificación.

Cuando el cliente hace la primera petición, el servidor envía el archivo de manifiesto correspondiente (.mpd para DASH, .m3u8 para HLS), con los detalles del archivo de vídeo: duración, tamaño del segmento, niveles de representación disponibles, códec, etc. A continuación, el cliente mide el ancho de banda actual y el estado del buffer. En función de ambos, va solicitando las siguientes partes del segmento de vídeo según la lógica de adaptación de velocidad, es decir, con el bitrate adecuado [2]. De esta manera, se reducen el número y la duración de las interrupciones porque el buffer está vacío (lo que se conoce como *stalling*) y el ancho de banda se utiliza de forma óptima [3]. La lógica de adaptación de velocidad del cliente se puede basar en el throughput, en el buffer o en un enfoque híbrido.

DASH ofrece como ventaja que los proveedores puedan prometer muchos niveles de calidad adaptando sus bitrates, lo que les permite ofrecer niveles personalizados de servicio, adaptados a la QoE y al precio. Como desventaja, deja una inestabilidad de vídeo, consecuencia de los múltiples cambios de calidad a los que somete al usuario [13]. También que los recursos de red quedan infrautilizados y que la QoE es desigual a lo largo de la reproducción.

La implementación de modelos en el presente trabajo está enfocada en aplicaciones DASH. Dichos modelos intentarán optimizarse para minimizar las desventajas del estándar.

1.2.3 Calidad de servicio (QoS) y calidad de experiencia (QoE)

1.2.3.1 Calidad de servicio (QoS)

La calidad de servicio (QoS) es un conjunto de exigencias de requisitos de calidad sobre el comportamiento colectivo de todos los mecanismos y procedimientos en la red, que evalúa la capacidad de la red para proporcionar un servicio a un nivel garantizado [5].

Tradicionalmente, la QoS se ha usado para medir la eficacia de los servicios, sin tener en cuenta ningún factor relacionado con el usuario final (sus expectativas, los factores ambientales, etc.), o sea, sin caracterizar la experiencia perceptual del usuario. Lo único que se considera al medir la QoS son los parámetros de red: pérdida de paquetes, ancho de banda, jitter, retardo, interrupciones y throughput.

A pesar de que la mayoría de estudios previos se enfocaban en la mejora de la QoS, la realidad es que potenciar la QoS no supone una mejora directa de la experiencia final del usuario. Además, centrarse en aumentar la QoS a veces aumenta los gastos operativos de forma significativa, lo que supone un decremento en las ganancias de los operadores de servicio.

1.2.3.2 Calidad de experiencia (QoE)

Con el objetivo de satisfacer las expectativas del usuario final surge el concepto de calidad de experiencia (QoE). La QoE es, según la ITU [14], el grado de aceptabilidad general de una aplicación o servicio, tal y como la percibe el usuario final. Es resultado del cumplimiento de las expectativas con respecto a la utilidad o el disfrute del servicio según su personalidad y su estado actual. A diferencia de la QoS, que solo se basa en mediciones técnicas y se limita a los servicios de telecomunicaciones, la QoE es multidisciplinar, y abarca ámbitos como la psicología, el medio ambiente, etc. La Figura 2 muestra la relación entre ambos conceptos.

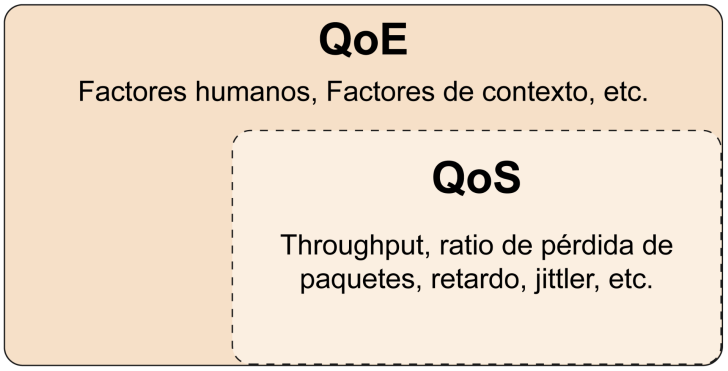


Figura 2. Relación entre los conceptos de QoS y QoE. [2]

Algunos parámetros de QoS pueden usarse como medidores de la QoE. Algunos de los más utilizados son el ancho de banda, el jitter, la velocidad de transmisión, la pérdida de paquetes (que puede generar discontinuidades o provocar la interrupción de la reproducción) o el retardo (tanto al comienzo de la reproducción como durante la misma) [2]. Por ejemplo, las aplicaciones de streaming en directo o en tiempo real requieren que el retardo sea menor a 150 milisegundos.

El proceso de evaluación de la QoE para un conjunto de usuarios de un servicio consiste en la medición de la misma utilizando un procedimiento específico y teniendo en cuenta todos los factores que influyen. Su principal objetivo es el diseño de un sistema que sea capaz de identificar los distintos factores y su respectiva influencia en la QoE del usuario final. Dichos sistemas reciben el nombre de modelos, como veremos en el apartado 1.2.4.

1.2.4 Métricas para la evaluación de la calidad de vídeo

Las métricas tradicionales para evaluar la calidad de vídeo se conocen como Video Quality Assessment o VQA. Pueden clasificarse en objetivas, subjetivas o híbridas.

1.2.4.1 Métricas objetivas

Las métricas objetivas son modelos matemáticos que buscan dar una puntuación de calidad que se acerquen a la que percibe el usuario. Algunos ejemplos son la Peak Signal to Noise Ratio (PSNR) o la Structural Similarity (SSIM). Ambas se crearon originalmente para analizar la calidad de imagen (IQA) y se adaptaron al análisis de vídeo siendo calculada como la media de las puntuaciones que recibe cada frame de la secuencia de vídeo.

Las ventajas de las métricas objetivas son su velocidad y facilidad de implementación, además de su bajo coste, en comparación con las subjetivas [3]. Entre las desventajas, destacan las principales:

- Las métricas objetivas podrían proporcionar puntuaciones similares para dos vídeos de calidad distinta, por lo que no siempre se puede establecer una correlación directa con la calidad percibida por el usuario.
- Necesita unas secuencias fuente para obtener resultados, algo no muy práctico en la mayoría de los escenarios reales.
- No están diseñadas para estimar la calidad a largo plazo porque originalmente solo calculaban las deficiencias provocadas por la pérdida de paquetes o la compresión en el proceso de transmisión (factores de QoS). No tienen en cuenta eventos de *rebuffering* ni cambios de calidad, habituales en las aplicaciones HAS. Por tanto, se requieren otros enfoques para el diseño de modelos de QoE para aplicaciones HAS que tengan en cuenta todas las deficiencias propias de las mismas, además de las debidas a la codificación con pérdidas.

1.2.4.2 Métricas subjetivas

Las métricas subjetivas de evaluación de la calidad de vídeo tienen en cuenta el feedback del usuario en forma de valoraciones. Normalmente, estos valores se registran en la escala Mean Opinion Score (MOS), que va del 1 al 5, siendo 1 la calidad mínima percibida por el usuario y 5 la máxima. En la Tabla 2 se muestra una descripción de cada punto de la escala [1].

Puntuación	Calidad	Distorsión
1	Mala	Molesta e inaceptable
2	Pobre	Molesta, pero aceptable
3	Aceptable	Perceptible y ligeramente molesta
4	Buena	Perceptible, pero no molesta
5	Excelente	Imperceptible

Tabla 2. Escala de evaluación de MOS.

Para que los tests subjetivos que se llevan a cabo sean replicables y válidos, la ITU define algunas recomendaciones (BT.500 y P.910) que incluyen una descripción detallada de cómo deben ser la configuración y la metodología a seguir [15] [22]. Esto incluye también el procesado de datos o la detección de outliers, entre otros. Las recomendaciones de la ITU se usarán como base para diseñar el experimento de visionado de la parte IV.

La ventaja principal de las métricas subjetivas es que proporcionan información sobre la verdadera calidad que experimentan los usuarios. Las desventajas, sin embargo, son lo costosas que son en cuanto a tiempo y recursos, que por dimensiones no son aptas para aplicarlas en el mundo real, y que solo puede evaluarse un número reducido de factores con el número limitado de participantes y el tiempo de duración del test.

1.2.4.3 Métricas híbridas

Uno de los enfoques más comunes, que pretende aprovechar las ventajas tanto de las métricas objetivas como de las subjetivas, es calcular los coeficientes de correlación y los valores del Mean Square Error (MSE) entre las puntuaciones MOS de la métrica objetiva VQA y los valores MOS reales del análisis subjetivo para una serie de secuencias de prueba [2].

Otro enfoque consiste en llevar a cabo un análisis *data-driven*, considerando el contexto del usuario junto con la evaluación de los parámetros técnicos de rendimiento, capturando datos del servicio y de los usuarios durante el uso de servicio a gran escala [3]. Es un enfoque prometedor, porque existen muchos *datasets* masivos con datos cada vez más precisos para caracterizar la percepción de los usuarios. Elimina el principal inconveniente de los tests objetivos (el conocimiento insuficiente del sistema de visión humano para dar estimaciones de calidad acertadas) y de los test subjetivos (costes muy elevados).

1.2.5 Factores de influencia (IF) en la QoE

La recomendación P.10/G.10 de la ITU [14] define como factor de influencia (IF) cualquier característica de un usuario, sistema, servicio, aplicación o contexto cuyo estado o configuración real pueda influir en la QoE del usuario final. Por tanto, se incluyen el tipo y las características de la aplicación o servicio, el contexto en el que se use, las expectativas que tenga el usuario al respecto de su uso y su cumplimiento, el trasfondo cultural, cuestiones socioeconómicas, perfiles psicológicos y estado emocional del usuario, entre muchos otros [2].

Los factores de influencia pueden agruparse en 4 categorías:

- **Del sistema:** Son aspectos básicos de calidad, es decir, aquellos que pueden medirse usando enfoques basados en QoS. Cubren los aspectos relacionados con multimedia, como cambios de calidad; relacionados con la red, como ancho de banda, retardo, jitter, pérdida de paquetes u otros que resulten en deficiencias como interrupciones temporales; y relacionados con el dispositivo de reproducción, como resolución de pantalla, códecs compatibles o formatos.

- **Humanos:** Son aspectos relacionados con el usuario. Cubre las características individuales como expectativas del servicio, efectos en la memoria, historial de uso del servicio (vídeos más reproducidos, etc.), situación demográfica y socioeconómica, constitución física y estado emocional, y nivel de atención, entre otros.

- **Del contexto:** Cubre los aspectos relacionados con la localización y el entorno del usuario (entorno de visionado, condiciones acústicas, etc.), momento del día, tipo de uso (búsqueda casual de vídeos, episodio nuevo de su serie favorita, etc.) y tiempo de consumo del servicio (en hora punta, cuánto de ese tiempo es de descarga, etc.), entre otros.

- **Del contenido:** Cubre los aspectos relacionados con las características del contenido que ofrece el servicio evaluado. Para vídeo son duración, tipo de vídeo y complejidad espacial y temporal.

1.2.5.1 Factores de influencia más importantes en aplicaciones HAS

Los principales IF que tienen un efecto directo sobre las aplicaciones HAS [2] son:

- **Cambios de calidad a lo largo del tiempo:** Según su frecuencia, magnitud, dirección y tiempo en el que la calidad es máxima. Se ha de analizar si suponen una ventaja o pueden llegar a empeorar la QoE de visionado del usuario.
- **Eventos de *rebuffering*:** Según su frecuencia, duración y localización temporal.
- **Calidad de codificación:** Según el tipo de contenido (mucho o poco movimiento). Se estima con el bitrate, el parámetro de cuantificación QP o métricas objetivas.
- **Retardo inicial durante la carga:** Cabe analizarlo en HAS, porque muchas veces es preferible que sea ligeramente mayor a cualquier interrupción durante la reproducción. Retrasos de hasta 10 segundos en el inicio de la reproducción tienen un aspecto despreciable en la QoE, en comparación con el *rebuffering.*
- **Características relacionadas con la memoria:** Según sea la calidad del vídeo al principio y al final de la reproducción, pueden ocurrir los fenómenos de *primacy* y de *recency*, respectivamente. Cuando la calidad al principio o al final del vídeo es mejor, en muchas ocasiones las puntuaciones de los usuarios son más altas. Es más común el fenómeno de *recency*, porque los usuarios tienen el recuerdo más reciente (especialmente si los vídeos son largos).
- **Compromiso del usuario:** Acciones del usuario durante la reproducción del vídeo que pueden alterar la QoE, como poner pausa, avanzar o retroceder, o cambiar la relación de aspecto de la pantalla
- **Contenido del vídeo:** Especialmente, cabe tener en cuenta las diferencias en la QoE percibida en función de la complejidad temporal del mismo.

1.2.6 Gestión de la calidad de experiencia: Modelado de QoE

La gestión de la QoE se divide en 3 fases: modelar, monitorizar y medir, y optimizar y controlar. Esta gestión se ha hecho especialmente difícil desde la popularización del streaming adaptativo, porque usuarios con preferencias distintas tienen acceso a múltiples redes desde terminales variadas, en localizaciones muy heterogéneas y desde muchas aplicaciones distintas [18]. En la Figura 3 pueden visualizarse las 3 fases.

Según la recomendación P.1201 de la ITU, un modelo de QoE es un algoritmo que permite estimar estimar la calidad percibida de una secuencia multimedia. Cada modelo tiene en cuenta unos IFs distintos.

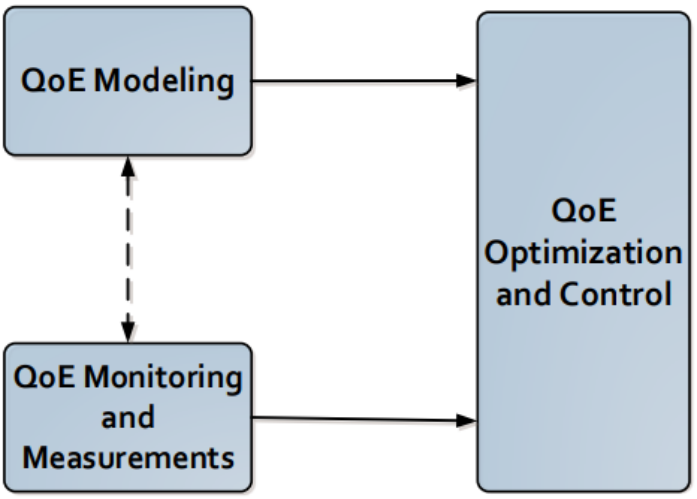


Figura 3. Fases de la gestión de QoE [2]

1.2.6.1 Criterios de evaluación del rendimiento de un modelo de QoE

Hay 3 criterios principales que sirven para evaluar la predicción hecha por un modelo de QoE [2]:

- **Precisión:** Habilidad del modelo para predecir las puntuaciones subjetivas con error bajo. Afecta a la efectividad y a la aplicabilidad del proceso de gestión de QoE. Puede medirse usando el Pearson Linear Correlation Coefficient (PLCC) entre la predicción y la puntuación real.
- **Monotonía:** Grado de acuerdo de la predicción con las magnitudes relativas de las puntuaciones subjetivas. Puede medirse usando el Spearman's Rank Correlation Coefficient (SROCC) entre la predicción y la puntuación real.
- **Consistencia:** Habilidad del modelo para mantener la precisión de la predicción sobre un amplio rango de secuencias de prueba con variedad de deficiencias de vídeo. Puede medirse usando medidas el Outlier Ratio (OR). Un valor bajo de OR indica una alta consistencia en la predicción (OR=0 significa que el modelo es estable para predecir la QoE). Un buen modelo debería proporcionar una perspectiva de cómo los IFs evaluados afectan a la QoE del usuario, que sirviese de ayuda para diseñar sistemas más eficientes y optimizados.

1.3 Estado del arte

1.3.1 Tipos de modelo de QoE

1.3.1.1 Clasificación de modelos de QoE

Los modelos de QoE pueden clasificarse siguiendo 3 criterios distintos [3]:

1. El tipo de información que usan para predecir la calidad
2. La cantidad de información que emplean de la señal
3. El nivel de detalle del modelado en relación con el sistema visual humano.

En la Figura 4 se muestra un resumen de los tipos de modelo de QoE.

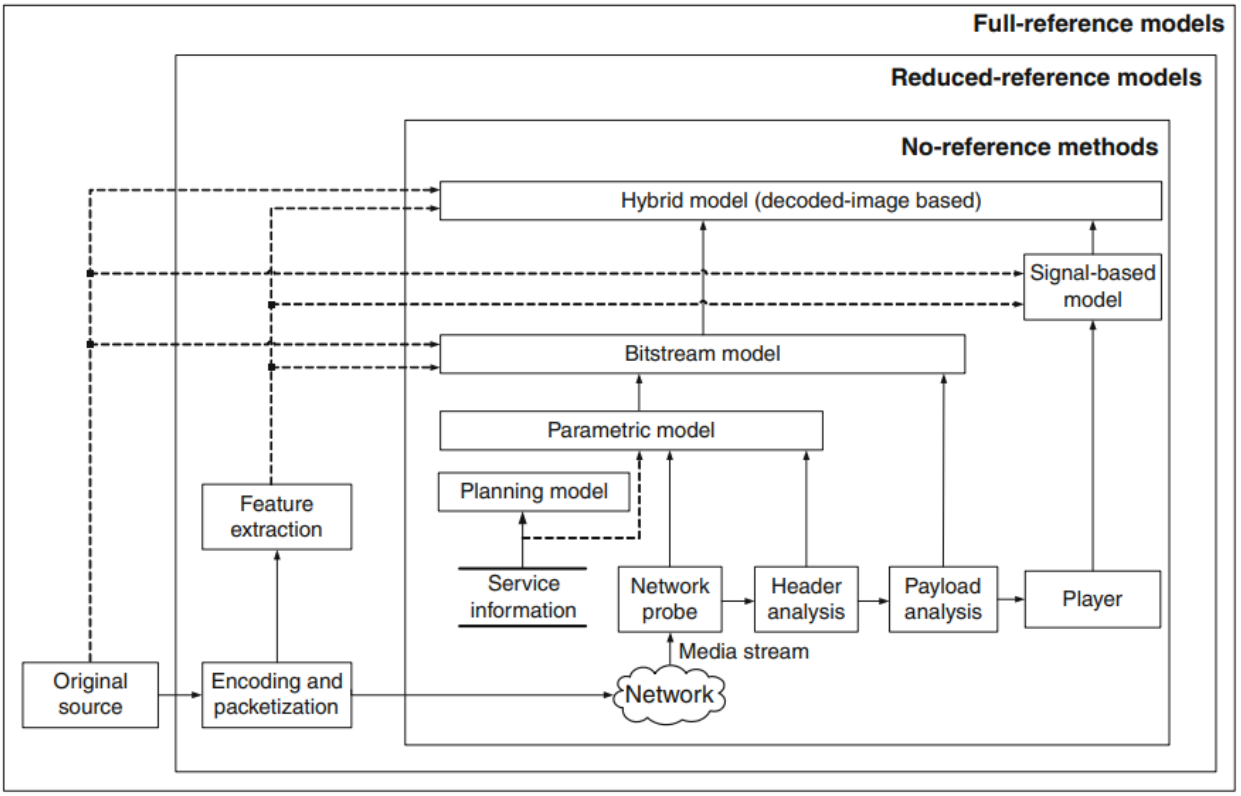


Figura 4. Tipos de modelo de QoE [3]

Según el tipo de información que emplean de la señal [2], pueden ser:

- **Basados en la señal (signal-based):** También reciben el nombre de píxel-based o modelos de la capa multimedia. Utilizan la señal decodificada para estimar la calidad de vídeo. Como no usan ninguna información específica del códec, se utilizan mucho en la comparación de códecs y la optimización de sistemas desconocidos. Dependiendo de la relación entre la entrada y la salida del sistema, o sea, de la cantidad de información de referencia que se necesite (nuestro segundo criterio), las métricas de calidad de vídeo se pueden clasificar, a su vez, en:
    - **Full-reference (FR):** Las métricas FR requieren la disponibilidad de información completa del vídeo original, es decir, de la señal original. Se calculan a partir de una comparación frame a frame entre la imagen/vídeo de referencia y el distorsionado. El vídeo de origen debe estar disponible en calidad original (sin alterar y sin comprimir) para que pueda haber una comparación directa (por ejemplo, píxel a píxel) entre la imagen/vídeo de referencia y el distorsionado. Como se dispone de toda la información, estas métricas suelen ser más precisas que sus homólogas (métricas RR o NR), pero no son adecuadas para la mayoría de las aplicaciones reales. Sin embargo, en general, algunas de las métricas de calidad más usadas en el campo del análisis de la calidad de imagen y vídeo son las FR. Algunos ejemplos son MSE, PSNR y SSIM.

    - **Reduced-reference (RR):** Las métricas RR solo requieren una cantidad limitada de información. En concreto, necesitan algunas características extraídas de la señal original. Por ello, normalmente son menos precisas que las métricas FR.

    - **No-Reference (NR):** No tienen acceso a la señal original y, consecuentemente, no usan ninguna información de referencia para predecir la calidad en base a la señal recibida. Algunas de las más comunes son DIIVINE, BRISQUE, BLIINDS y NIQE. Suelen ser menos precisas que sus homólogas, FR y RR.

- **Paramétricos:** Utilizan parámetros medidos que están relacionados con los paquetes o con la red con el objetivo de estimar la calidad. Se pueden clasificar en modelos de la capa de empaquetado y modelos de planificación.

    - **Modelos de la capa de empaquetado (packet-layer models):** Utilizan únicamente la información que puede extraerse de cabeceras de paquetes, como la tasa de bits, la tasa de pérdida de paquetes (PLR), la velocidad de fotogramas (frame rate), el tipo de fotogramas, etc., y no requieren información sobre la señal multimedia. Por tanto, estos modelos no son intrusivos, son fáciles de implementar y computacionalmente muy baratos. Como no se dispone de información sobre la carga útil de datos transmitidos (*payload*), estos modelos no son adecuados para soluciones individuales de supervisión de la QoE, como la determinación del efecto de la dependencia del contenido en la QoE del usuario final, por ejemplo.

    - **Modelos de planificación (planning models):** No requieren información de entrada. Estos modelos estiman la calidad basándose en la información de planificación de la calidad disponible durante la fase de planificación de las redes y los terminales. Información como el bitrate esperado, el PLR, el tipo de códec, etc. se usa como input. Algunos de estos modelos son de los más ampliamente usados en el campo de los servicios de videotelefonía, y también para aplicaciones de streaming de imagen y vídeo.

- **Flujo de bits (bitstream):** Tienen en cuenta el flujo de bits codificado y la información de la capa de empaquetado. Se extraen algunas características que se utilizan como entrada del modelo: bitrate, frame rate, el parámetro de cuantificación (QP), PLR, vector de movimiento, tamaño de macrobloque (MBS), coeficientes DCT, etc. Estos modelos son relativamente baratos desde el punto de vista computacional y pueden utilizarse para monitorizar la QoE en tiempo real. Además, han encontrado una aplicación recientemente en el campo de los servicios de transmisión multimedia, siendo la ITU-T Rec. P.1203 una de las recomendaciones que ha sido aprobada más recientemente para servicios de streaming audiovisual adaptativos sobre transporte fiable, y que además será utilizada en el punto 3.1.

  Aunque estos muestran una correlación superior con las puntuaciones subjetivas de calidad, solo son adecuados para un códec específico. Por ello, los modelos de flujo de bits que pueden minimizar su dependencia del rendimiento de parámetros específicos del códec, como el tamaño de MB, tamaño del vector de movimiento, etc. resultan más útiles y son más aceptados.

- **Híbridos:** Habitualmente son los modelos más efectivos, porque combinan 2 o más de los anteriores, pudiendo así utilizar mucha más información como input en comparación. Usan como entrada una señal, un flujo de bits y/o la información de cabecera de los paquetes.

La Figura 5 muestra algunos de los modelos de evaluación de calidad más utilizados.

Metric	Year	Model Type	Modality
Peak Signal-to-Noise Ratio (PSNR) [49]		FR	Images - Frames
Structural Similarity Index Metric (SSIM) [44]	2004	FR	Images - Frames
Video Multimethod Assessment Fusion (VMAF) [45]	2016	FR	Images - Frames
Visual Information Fidelity (VIF) [50]	2006	FR	Images - Frames
HDR-VDP-2: A calibrated visual metric for visibility and quality predictions in all luminance conditions [51]	2011	FR	HDR images
Video Quality Metric (VQM) [52]	2004	FR	Video
Reduced Reference Entropic Differencing (RRED) [53]	2013	RR	Video
Spatial Efficient Entropic Differencing for Quality Assessment (SpEED-QA) [54]	2017	RR	Video
Blind Image Quality Index (BIQI) [55]	2010	NR	Images - Frames
Blind/Referenceless Image Spatial QUality Evaluator (BRISQUE) [56]	2012	NR	Images - Frames
Naturalness Image Quality Evaluator (NIQE) [57]	2013	NR	Images - Frames
HDR Image GRADient based Evaluator (HIGRADE) [58]	2017	NR	HDR images

Figura 5. Algunos de los modelos de evaluación de calidad más usados para imagen y vídeo [18]

1.3.1.2 Modelos de QoE para HAS

La mayoría de modelos que se han visto en el apartado anterior están estandarizados para analizar la calidad de vídeo, pero solo son válidos para predecir la calidad de secuencias muy cortas (de aproximadamente 10 segundos) y no tienen en cuenta las variaciones en la calidad y las interrupciones típicas de la tecnología HAS.

Los modelos diseñados específicamente para HAS tienen que considerar el impacto de la adaptación temporal de la calidad como un factor de influencia (IF) muy importante, porque una de los principales problemas del funcionamiento del HAS es precisamente la fluctuación de la calidad de la secuencia de vídeo a lo largo de la reproducción.

En los modelos HAS, la calidad varía en muchas ocasiones durante la reproducción, y la adaptación a la calidad suele alcanzarse después de más de 20 segundos. Por tanto, los modelos tienen que estar adaptados para estimar la calidad durante secuencias de vídeo más largas (de unos pocos segundos a bastantes minutos). Deben estar preparados para las fluctuaciones de calidad durante toda la reproducción e, incluso, para posibles cambios de dispositivo. Como los modelos de estimación de QoE están preferiblemente desarrollados a partir de los resultados de tests subjetivos, hace falta una revisión de los métodos subjetivos estandarizados existentes para que sean válidos también para HAS.

Algunos de los resultados más significativos de los test subjetivos, y que sugieren tener en cuenta unos factores de influencia concretos para el desarrollo de modelos HAS, son los siguientes [2]:

- Se ha visto que los sujetos reaccionan en cantidades de tiempo distintas a grandes degradaciones o mejoras repentinas de la calidad.
- La localización de la degradación o de la mejora de calidad influye en la valoración total del usuario, lo que revela un efecto de *recency*.
- La percepción de las distorsiones espaciales en el vídeo puede verse enormemente modificada por sus cambios temporales. Es decir, si el vídeo tiene una gran complejidad temporal (mucho movimiento), los cambios de calidad se perciben con mayor facilidad.
- Se puede percibir un efecto de histéresis respecto a las variaciones de calidad a lo largo del tiempo. Esto significa que, cuando ha habido un cambio en la calidad durante la reproducción, durante un tiempo el usuario puede seguir percibiendo la anterior.
- El estudio de la adaptación de vídeo para modelar el impacto del bitrate y el frame rate en la calidad percibida sugiere que para secuencias de vídeo con alta complejidad temporal, la adaptación del frame rate dará como resultado una mejor calidad que la adaptación del QP. Y que, por el contrario, para vídeos con poco movimiento es mejor adaptar el QP.

Los modelos de QoE para HAS emplean modelos existentes para evaluar la calidad de intervalos cortos (en torno a 10 segundos) y luego combinan dichas puntuaciones en una única estimación de calidad usando técnicas de agrupación/unificación temporal.

Hay muchos algoritmos de agrupación temporal, desde los más simples (que consideran el máximo, el mínimo o la calidad media) a aquellos que integran las propiedades temporales de la percepción humana, efectos en la memoria y propiedades transitorias. Que algunos algoritmos sean simples no significa que sus resultados sean peores. De hecho, las evaluaciones de las técnicas de agrupación temporal usando PSNR y SSIM como predictores de calidad concluyen que la media de las puntuaciones individuales puede conseguir resultados similares a los métodos de agrupación temporal más sofisticados [3].

1.4 Metodología

1.4.1 Estructura del trabajo

En términos generales, se ha creado un sistema capaz de generar vídeos distorsionados emulando escenarios de red concretos a partir de vídeos originales servidos en la web con la tecnología DASH. Empleando dicho sistema, se ha evaluado la QoE de los vídeos resultantes usando el modelo híbrido de la Unión Internacional de Telecomunicaciones (ITU) y un modelo subjetivo propio, consistente en la elaboración de un test de evaluación de los vídeos distorsionados por parte de personas reales. De esta forma se ha logrado implementar una herramienta completa de evaluación de QoE automatizada para vídeos en streaming que incluye dos modelos de evaluación distintos.

Este apartado tiene como objetivo resumir los pasos que se han seguido en la elaboración del presente trabajo. A continuación, se expone una descomposición del procedimiento, que está representada además en el diagrama de la Figura 6.

Figura 6. Descomposición en pasos del procedimiento a seguir en el presente trabajo

1. **Selección de las secuencias de prueba:** Se establecen como criterios que cubran varios tipos de contenidos, que sean de duración suficiente para aplicar los distintos escenarios de red y que presenten niveles de información espacio-temporal distintos. Para medir esto último se incluye el análisis SI/TI, explicado con detalle en el apartado 2.1.1.

2. **Preparación de los contenidos:** Se eligen los parámetros de codificación y se codifican 96 vídeos con FFMPEG, combinando valores distintos de CRF y resolución. El script también calcula el PSNR y el bitrate de cada vídeo generado. Según las curvas PSNR-bitrate, se eligen los 12 vídeos con mejor relación entre ambas y se segmentan con unos parámetros específicos de FFMPEG. Se emplea la librería ffmpeg-python.

3. **Streaming del contenido en DASH:** Se crea un contenedor usando Docker, desde el que se inicia un servidor web usando una imagen predefinida. Se diseña una página web usando HTML y CSS con un reproductor de Shaka embebido en la que se puedan reproducir los vídeos segmentados haciendo uso del servidor web creado.

4. **Emulación automatizada de distintos escenarios de red:** Utilizando la librería Puppeteer y Javascript, se aplican cambios automatizados en el ancho de banda disponible a los vídeos origen, según 3 escenarios de red previamente diseñados. De esta forma, se fuerzan cambios de calidad e interrupciones durante el streaming de los mismos. Se extrae una traza de los eventos del reproductor de vídeo.

5. **Reconstrucción de las secuencias distorsionadas:** Se emplean los datos extraídos del paso anterior y los segmentos de todas las calidades para reconstruir vídeos que recreen los cambios de calidad y las interrupciones ocurridas al aplicar cada escenario de red. Estos vídeos se usarán como entrada de los modelos de QoE.

6. **Evaluación híbrida de la QoE:** Se usa una implementación en python de la recomendación ITU-T P.1203 para obtener valores de calidad del 1 al 5 de los vídeos reconstruidos y también de los originales. Se obtienen valores para cada uno de sus modos: 0, 1, 2 y 3.

7. **Evaluación subjetiva de la QoE:** Se elige un grupo de visionado para los vídeos distorsionados. Para que puedan visualizarlos, se elabora una web usando HTML y CSS de nuevo. La web incluye: una página de instrucciones, páginas con cada vídeo a visualizar y un desplegable para puntuar la calidad del 1 al 5, un formulario con preguntas sobre las condiciones de visionado y una página final. Se usa Javascript para iniciar un servidor, servir los HTML y configurar la inserción de los resultados en una base de datos MySQL. Finalmente se despliega la web usando Docker para que los participantes puedan completar el test desde distintas ubicaciones y dispositivos.

8. **Análisis de los resultados obtenidos y conclusiones:** Se presentan los resultados obtenidos para cada modelo y se calculan parámetros asociados a los mismos con el objetivo de analizarlos de forma individual y comparativa. Se extraen conclusiones sobre el efecto de los factores de influencia en el modelo subjetivo y sobre la validez de los datos extraídos de ambos modelos. Se presentan las limitaciones de la investigación y las recomendaciones a futuro.

1.4.2 Herramientas utilizadas

A continuación, se presentan todas las herramientas que se han utilizado en la elaboración del trabajo.

Lenguajes de programación y/o de marcado y estilos:

- Python: Lenguaje de programación. Con él se ha elaborado el código de los scripts que sirven para: codificar vídeos, segmentar vídeos, parsear los datos de la traza de Puppeteer, reconstruir los vídeos distorsionados y añadir interrupciones a los vídeos distorsionados.

- Javascript: Lenguaje de programación, Con él se ha elaborado el código de los scripts que sirven para: ejecutar Puppeteer en Chrome e iniciar un servidor, servir los ficheros HTML del test subjetivo en la web, y extraer parámetros para insertarlos en la BBDD que administra (la configura, la crea e inserta los datos extraídos de la web).

- HTML: Lenguaje de marcado. Con él se han elaborado: el código de la página web en la que se sirven los vídeos para luego reproducir los contenidos generados e ir cambiando el ancho de banda de forma automática y el código de la web en la que se implementa el test subjetivo.

- CSS: Lenguaje de estilos. Con él se han diseñado los estilos de todas las páginas web.

- JSON: Formato de texto. Se ha utilizado para guardar las trazas con los eventos del reproductor de vídeo y para definir los escenarios de red en diccionarios con pares de claves *tiempo* y *ancho de banda*.

Herramientas y tecnologías web o de desarrollo

- *siti-tools*: Librería de Python. Se ha empleado para realizar el análisis SI/TI de las secuencias de prueba a través del cálculo de su información espacial y temporal. Dicho análisis está explicado con más detalle en el apartado 2.1.1.

- FFMPEG: Herramienta para línea de comandos. Se ha usado para codificar y segmentar vídeos, concatenar segmentos de vídeo con el método *concat*, y generar vídeos estáticos para crear interrupciones a partir de un frame.

- *ffmpeg-python:* Librería de Python. Ha permitido usar FFMPEG en todos los scripts escritos en Python, en lugar de hacerlo directamente en la línea de comandos.

- Docker: Plataforma de contenerización. A través de ella se ha creado el contenedor en el que configurar un servidor web Apache para servir los contenidos de vídeo en la parte II. En la parte IV, se ha definido el test subjetivo como una aplicación multi-contenedor (servidor y base de datos)

- Terminal de comandos (CMD): Intérprete de comandos en Windows. Se ha utilizado para lanzar o ejecutar los scripts escritos en Python y Javascript.

- Chrome: Navegador web de Google. Se ha elegido como navegador en el que usar todas las webs creadas por su compatibilidad con Google Puppeteer para hacer las pruebas con los diferentes escenarios de red automatizados y también con la librería shaka player.

- Puppeteer: Librería de Node.js con la que controlar Chrome desde una interfaz de programación. Ha permitido automatizar los cambios de ancho de banda disponible en el navegador, posibilitando cambios de calidad e interrupciones.

- Shaka player: Librería de Javascript para streaming de vídeo adaptativo. Se ha utilizado su reproductor multimedia para reproducir los contenidos DASH en las páginas web.

- GitHub: Plataforma de desarrollo cooperativo de código. Se ha clonado el repositorio [19a] que contiene la implementación de la recomendación ITU-T P.1203 en Python.

- Node.js: Plataforma de código abierto para ejecutar código Javascript en el servidor. Se ha elaborado una aplicación Javascript usando una imagen Node.js, que además permite manejar la base de datos haciendo uso del framework Express.js

- Express.js: Framework para Node.js que facilita el manejo de solicitudes y respuestas HTTP. Permite obtener el valor de los parámetros de las consultas HTTP entrantes (los valores del formulario) y almacenarlos en otra variable para luego insertarlos en la base de datos.

- *body-parser:* Middleware usado en combinación con Node.js y Express.js. Se ha utilizado para analizar las solicitudes HTTP POST y transformarlas en objetos Javascript con los que trabajar en el código.

- *express-session:* Middleware de Express.js que permite mantener la información cuando se hacen varias solicitudes HTTP en la misma sesión. Se ha utilizado para asignar un identificador único a cada sesión, permitiendo que no se pierdan los datos introducidos en el test durante la interacción del usuario con la web.

- VMWare: Plataforma de virtualización. Se ha empleado para ejecutar un SO Linux en Windows y poder pasar el modelo ITU a los vídeos distorsionados.

Herramientas de manipulación de datos:

- Excel: Software de hoja de cálculo. Se ha empleado para crear algunas figuras (diseño de los escenarios de red, información espacio-temporal de los vídeos, etc.) y para el cálculo de parámetros asociados a los resultados (medias, desviaciones estándar, etc.).
- MySQL: Sistema de gestión de bases de datos relacionales. Se ha empleado para la creación de la base de datos que almacena los valores introducidos por los espectadores durante la realización del test. Como ayuda, se ha utilizado también su interfaz gráfica MySQL WorkBench.
- *pandas:* Librería de Python. Se ha usado para convertir los resultados de la traza parseados en un CSV y para extraer de dicho CSV una lista con todos los segmentos que se han descargado durante la reproducción.

Todo el código desarrollado en el presente trabajo puede verse en el Anexo I y todas las tablas que no aparecen a lo largo del documento, pero sí se mencionan, están en el Anexo II.

PARTE II: EXTRACCIÓN DE DATOS Y GENERACIÓN DE VÍDEOS

2.1 Selección de vídeos referencia/secuencias de prueba

Se han elegido 4 vídeos de referencia. Para la selección, se han seguido 3 criterios principales:

- Teniendo en cuenta la importancia de la información espacial y temporal como factores de influencia en los modelos de QoE para HAS, los vídeos han de presentar características distintas en cuanto a movimiento y detalles (información espacial y temporal).
- Los vídeos han de cubrir varios tipos de contenidos: animación, película/serie, noticias y eventos deportivos.
- La longitud de los vídeos debe ser suficiente para que haya varios cambios de calidad, pero no excesivamente largos para que el espectador no pierda la atención durante el experimento de visionado y los resultados de las pruebas subjetivas sigan siendo concluyentes. Se ha decidido usar vídeos con una longitud de 1 minuto.

Los vídeos se han extraído de la página web de Radio Televisión Española (RTVE) [21] y son los siguientes:

- *animacion.mp4:* Corto de animación *Hero* creado íntegramente en Blender.
- *noticias.mp4:* Entrevista a Fernando Alonso en el Gran Premio de Montmeló 2023.
- *deportes.mp4:* Combate de taekwondo femenino entre Uzbekistán y Brasil.
- *serie.mp4:* Capítulo de la serie televisiva *La otra mirada.*

Se han recortado los vídeos a la duración establecida y se les ha eliminado el audio.

ffmpeg -i videoEntrada.mp4 –ss 00:00:00 –t 00:01:00 -an -c:v copy videoSalida.mp4

2.1.1 Análisis SI/TI de vídeos

Para analizar la complejidad espacial y temporal de las secuencias de prueba, se puede hacer un cálculo del factor de información espacial (SI, Spatial Information) y del de información temporal (TI, Temporal Information). El factor SI mide la actividad espacial de cada escena mediante el cálculo de la desviación estándar, sobre los píxeles de cada fotograma. El factor TI se basa en la diferencia de movimiento, que es la existente entre los valores de los píxeles en la misma ubicación en el espacio entre fotogramas adyacentes. [9]

Se han extraído las características de información espacial y temporal de los 4 vídeos disponibles en formato CSV, haciendo uso de la librería *siti-tools* de Python.

siti -of csv video.mp4 > video.csv

En la Tabla 3 se muestran los valores promedio de la información espacial (SI) y temporal (TI) por separado. También se ha elaborado una gráfica SI/TI (Figura 7) en la que se muestran los valores de las escenas seleccionadas finalmente, que serán las secuencias de prueba para la implementación de los modelos objetivo y subjetivo de QoE.

Nombre vídeo	Información SI	Información TI
animacion.mp4	45.58	14.50
noticias.mp4	83.31	8.56
deportes.mp4	91.96	15.20
serie.mp4	35.60	4.34

Tabla 3. Características de las secuencias de prueba.

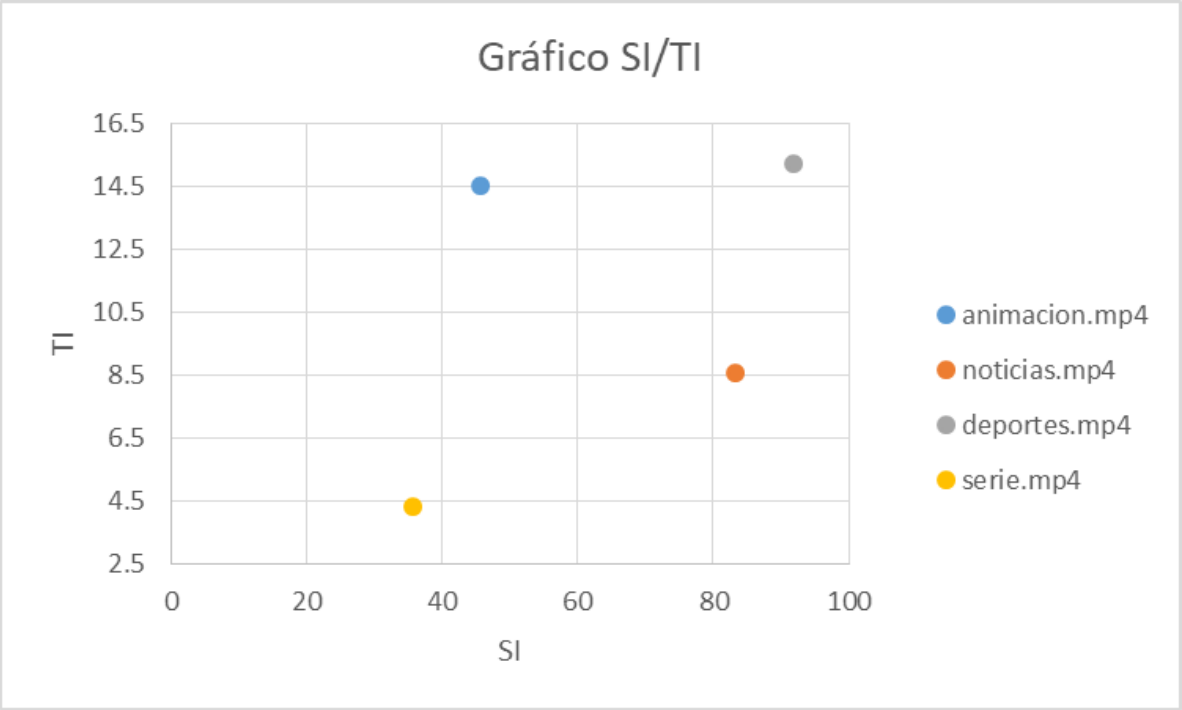


Figura 7. Gráfica SI/TI con los valores promedio de cada vídeo.

2.2 Selección de parámetros de codificación y generación de vídeos de distintas calidades con FFMPEG

Se ha elaborado un script que codifica un total de 96 vídeos, con las combinaciones de 8 valores de calidad (cambiando el parámetro CRF) y de 3 resoluciones distintas de cada uno de los 4 vídeos de prueba. Para ello, se ha empleado principalmente la librería ffmpeg-python [6]. El script es *calidades.py*, visible en el punto 1 del Anexo I. Los valores de CRF empleados son 20, 22, 25, 27, 30, 35, 40 y 45; y las resoluciones 416x234, 960x540 y 1280x720.

Los siguientes son los parámetros que se han elegido para codificar:

- **-vcodec “libx264”:** Cada vídeo se codifica en H.264, y por tanto el GoP es cerrado por defecto, asegurando que no hay dependencias de ninguna trama del GoP siguiente.
- **-g 24, -r 24**: El tamaño de GoP es de 24 frames (*g = 24*), es decir, de un segundo, porque se fija el framerate a 24 fps (*r = 24*). Que el GoP sea de un segundo permite que sea múltiplo de cualquier tamaño de segmento que se elija para segmentar posteriormente, lo que posibilita que los segmentos contengan un número entero de GoP completos, facilitando los cambios de escena y el procesamiento de los vídeos.
- **-b_strategy 0:** Se impide a FFMPEG que vaya cambiando el número de tramas B introducidas en cada GoP según considere más adecuado para mantener siempre la misma estructura.
- **-bf 2:** Por defecto, ffmpeg no incluye tramas B en el GoP cuando codifica en H.264, por lo que se colocan 2 tramas B entre cada frame P, según recomienda la propia herramienta.
- **-sc_threshold:** Evita que se generen fotogramas Intra por la detección de escenas o se generen tramas B adicionales.
- **-aspect “16/9”:** Fuerza el aspect ratio de los vídeos de salida a 16/9 para evitar problemas durante el proceso de reconstrucción de los vídeos

Por último, el script calcula para cada vídeo los valores de bitrate y PSNR medio, con el objetivo de seleccionar aquellos con mejor relación entre ambos parámetros para el desarrollo de los modelos de QoE. Los datos de bitrate y PSNR se almacenan en un fichero CSV, desde el que se llevará a cabo el siguiente análisis.

2.3 Selección de los vídeos con mejor relación PSNR-bitrate

Para cada uno de los 96 vídeos codificados, se representan los valores de PSNR y de bitrate. Se han elaborado 4 representaciones, una para cada vídeo origen, que contienen series de datos de 8 puntos en forma de curva correspondientes a las 3 resoluciones con las que se ha llevado a cabo el proceso de codificación.

Teniendo en cuenta la envolvente convexa, se eligen las 3 representaciones de cada vídeo origen (una por resolución, para que el streaming sea adaptativo) que tienen el mejor PSNR para el bitrate que presentan. Es decir, aquellas que ofrecen una calidad mejor para una tasa de codificación de bits más baja. Siguiendo este criterio se seleccionan 12 representaciones de vídeo en total, que son las que se segmentarán y se usarán para introducir las distorsiones en los apartados posteriores. Este procedimiento es el que siguen las plataformas que ofrecen contenidos en streaming, como Netflix, para optimizar sus procesos de codificación de vídeo.
En las figuras 8, 9, 10 Y 11 se presenta el análisis y se señalan los vídeos que han sido seleccionados. Las representaciones que se segmentarán son:

- Para *animacion.mp4:* 234-crf40.mp4, 540-crf35.mp4, 720-crf30.mp4
- Para *noticias.mp4:* 234-crf35.mp4, 540-crf35.mp4, 720-crf35.mp4

- Para *deportes.mp4:* 234-crf35.mp4, 540-crf40.mp4, 720-crf40.mp4
- Para *serie.mp4:* 234-crf35.mp4, 540-crf35.mp4, 720-crf35.mp4

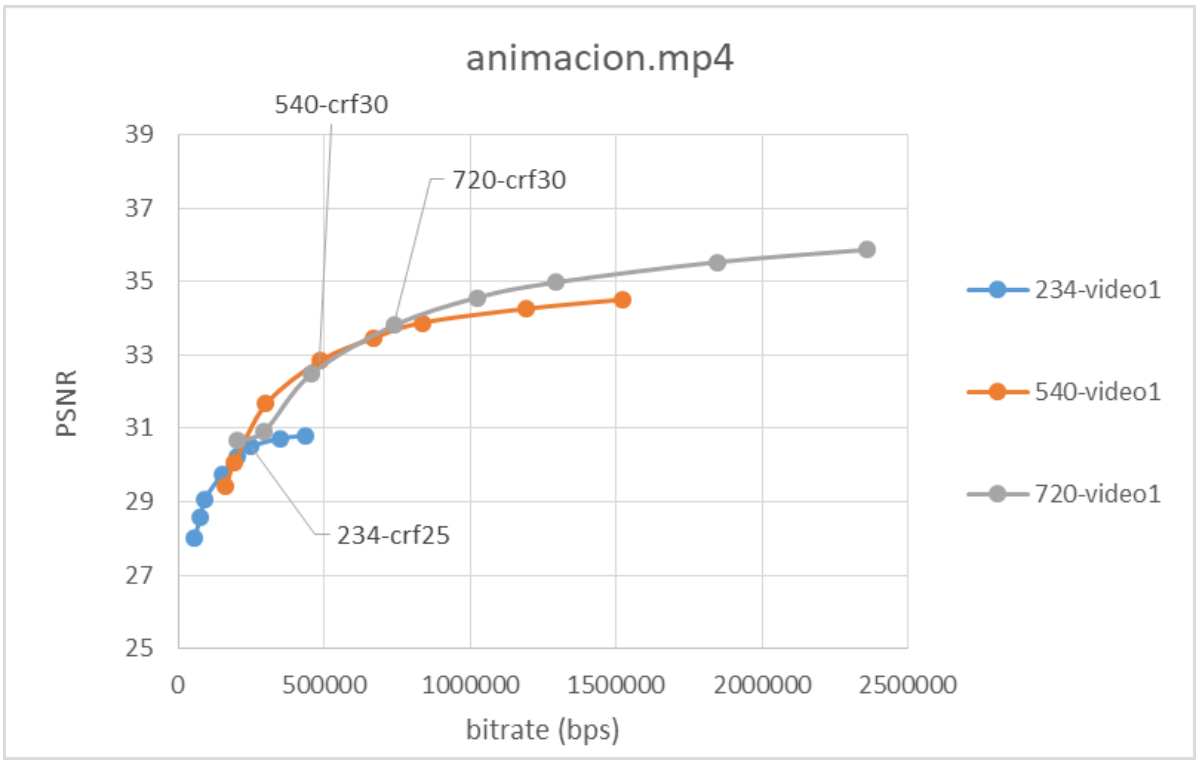


Figura 8. Relación PSNR-bitrate vídeos codificados a partir de animacion.mp4

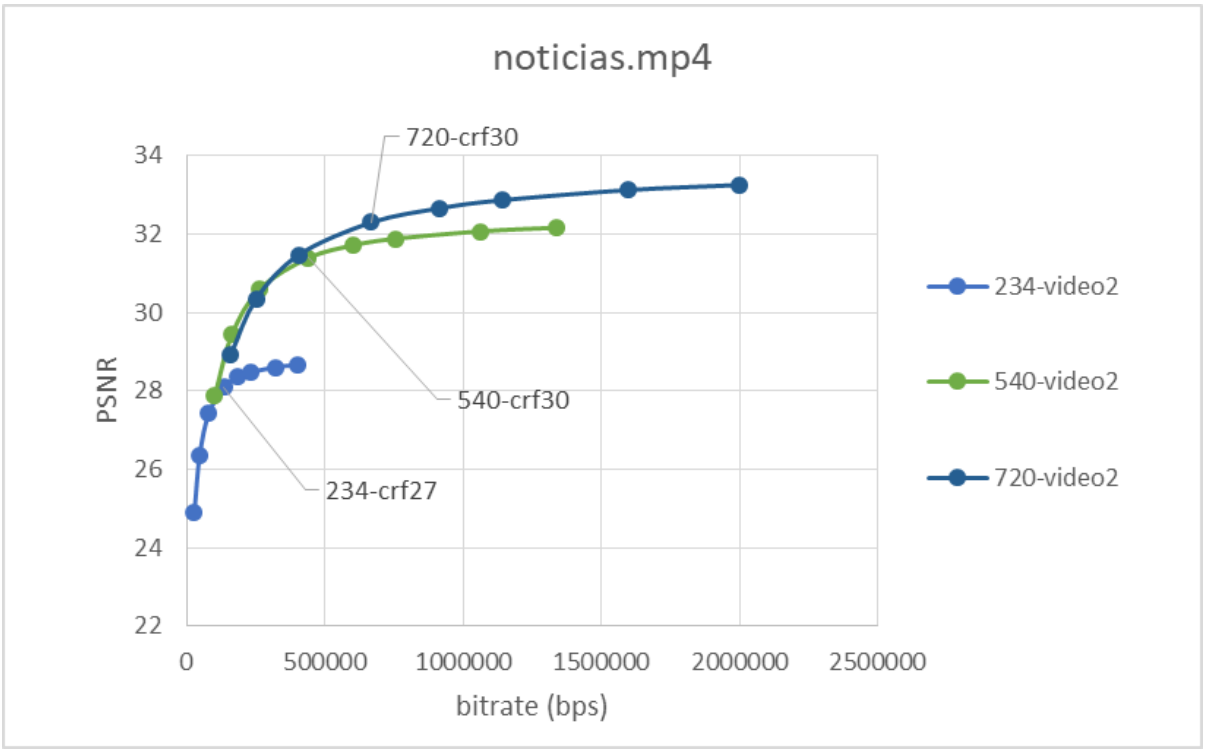


Figura 9. Relación PSNR-bitrate vídeos codificados a partir de noticias.mp4

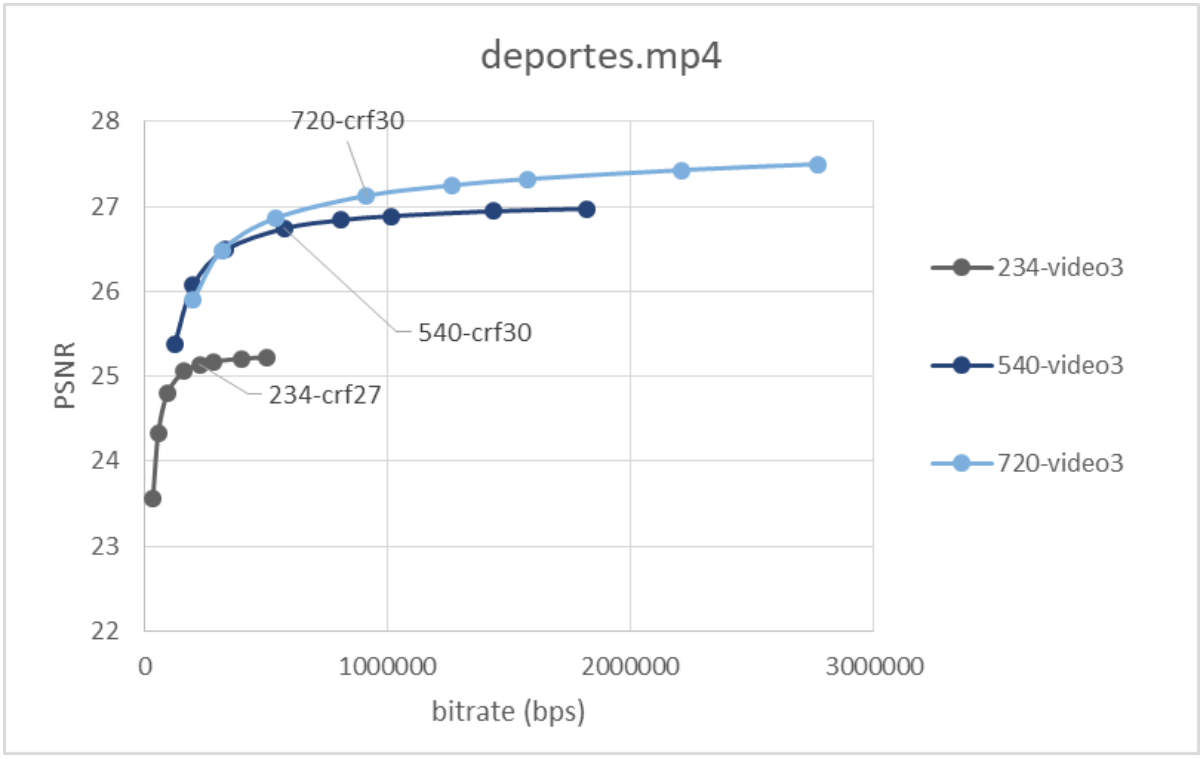


Figura 10. Relación PSNR-bitrate vídeos codificados a partir de deportes.mp4

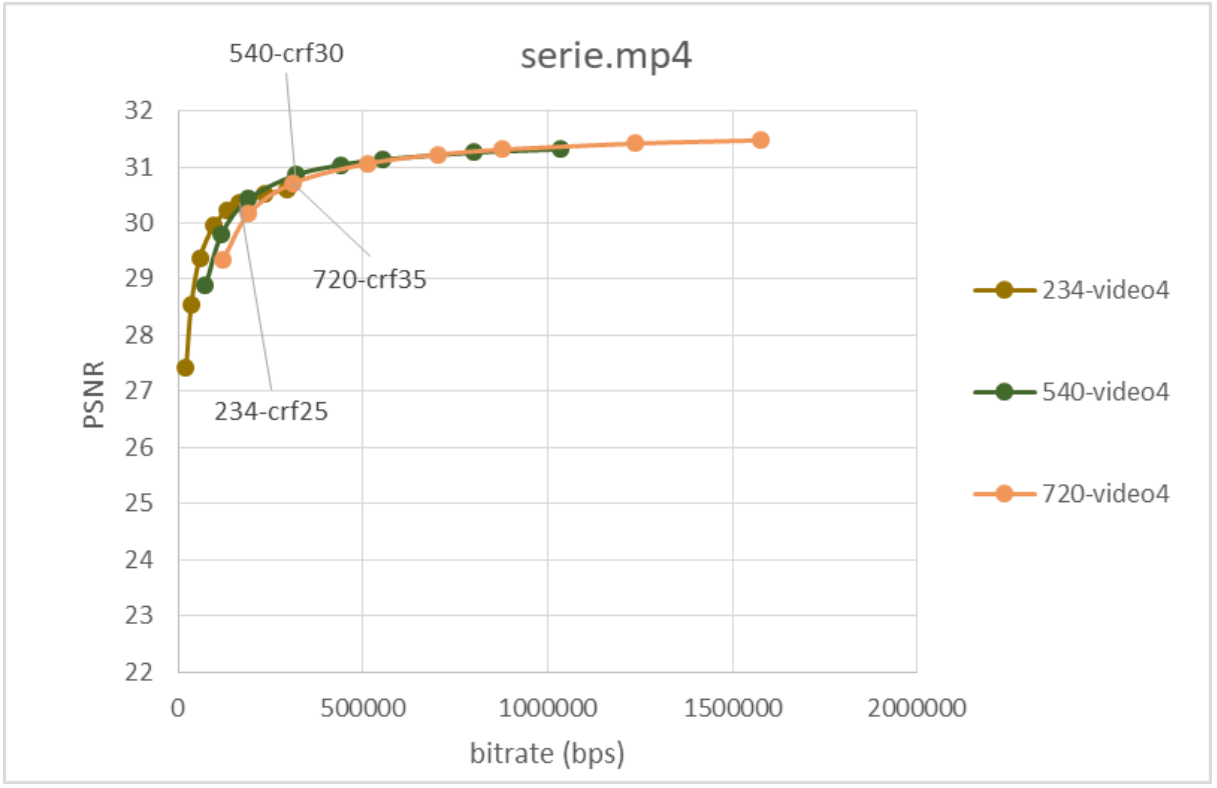


Figura 11. Relación PSNR-bitrate vídeos codificados a partir de serie.mp4

2.4 Segmentación de los vídeos para usarlos en HAS

Para hacer uso de streaming adaptativo (HAS) y que el algoritmo de adaptación de calidad pueda ir seleccionando el segmento más oportuno en función de las condiciones de red y del dispositivo, es necesario segmentar los vídeos codificados antes de servirlos.

Continuando con el uso de ffmpeg-python, se ha elegido emplear un tamaño de segmento de 4 segundos para el proceso de segmentación de los 12 vídeos seleccionados. El script es *segmentado.py*, visible en el punto 2 del Anexo I. Se han seguido las recomendaciones de la documentación de HLS de Apple y las de la empresa Bitmovin, que aconseja establecer un tamaño de entre 2 y 4 segundos para lograr un buen compromiso entre eficiencia de codificación y flexibilidad para la adaptación del stream a los cambios en el ancho de banda. [16]

Los siguientes son los parámetros que se han elegido para segmentar:

- **-seg_duration 4:** Tamaño de segmento en segundos.
- **-vcodec copy:** No se modifica el códec de los vídeos de entrada.
- **adaptation_sets "id=0,streams=v":** Asigna todas las representaciones de vídeo a un único Adaptation Set para que el algoritmo pueda ir cambiando de calidad según las condiciones de red.
- **-f dash:** Segmenta los contenidos según el estándar DASH.

Los ficheros .m4s generados facilitan el proceso de concatenación de segmentos durante la reconstrucción del vídeo en el apartado 2.7.5.

2.5 Creación del servidor web

Una vez se dispone de los contenidos segmentados, es necesario un servidor. Se utiliza Docker para crear un contenedor en el que se configura un servidor web Apache. De esta forma no es necesario instalar el servidor web. Los paquetes se almacenan en dicho servidor HTTP, que entrega el contenido empaquetado al navegador. Para iniciar el contenedor, se ejecuta el siguiente comando:

docker run -d --name player -p 8082:80 -v %cd%:/usr/local/apache2/htdocs/ httpd:2.4

Se ha creado una instancia de un contenedor Apache 2.4 de nombre *player* usando la imagen *httpd:2.4* de Docker Hub [17]. Las solicitudes que se realizan en el puerto 8082 se redirigen al puerto 80 en el contenedor. La estructura de la carpeta cuyos contenidos se servirán se muestra en la Figura 12. El contenido de los ficheros de la carpeta se muestra en el apartado 2.6.

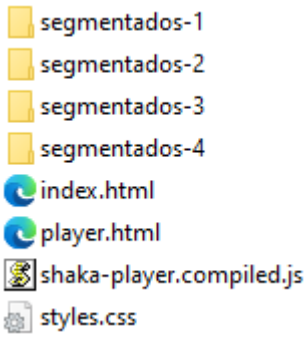


Figura 12. Contenido de la carpeta de archivos servidos con Apache

2.6 Desarrollo de la página y el reproductor web

Se necesita disponer de una página web con un reproductor embebido para poder reproducir los contenidos generados. Para ello, se necesitan 4 ficheros.

- *index.html:* Es la página principal. En ella se muestran los 4 vídeos disponibles para la reproducción con su portada. Incluye una cabecera en la que se entrelaza con el archivo de estilos CSS y un contenedor con las portadas de los 4 contenidos. Su aspecto se muestra en la Figura 13 y su código está en el punto 3 del Anexo I.

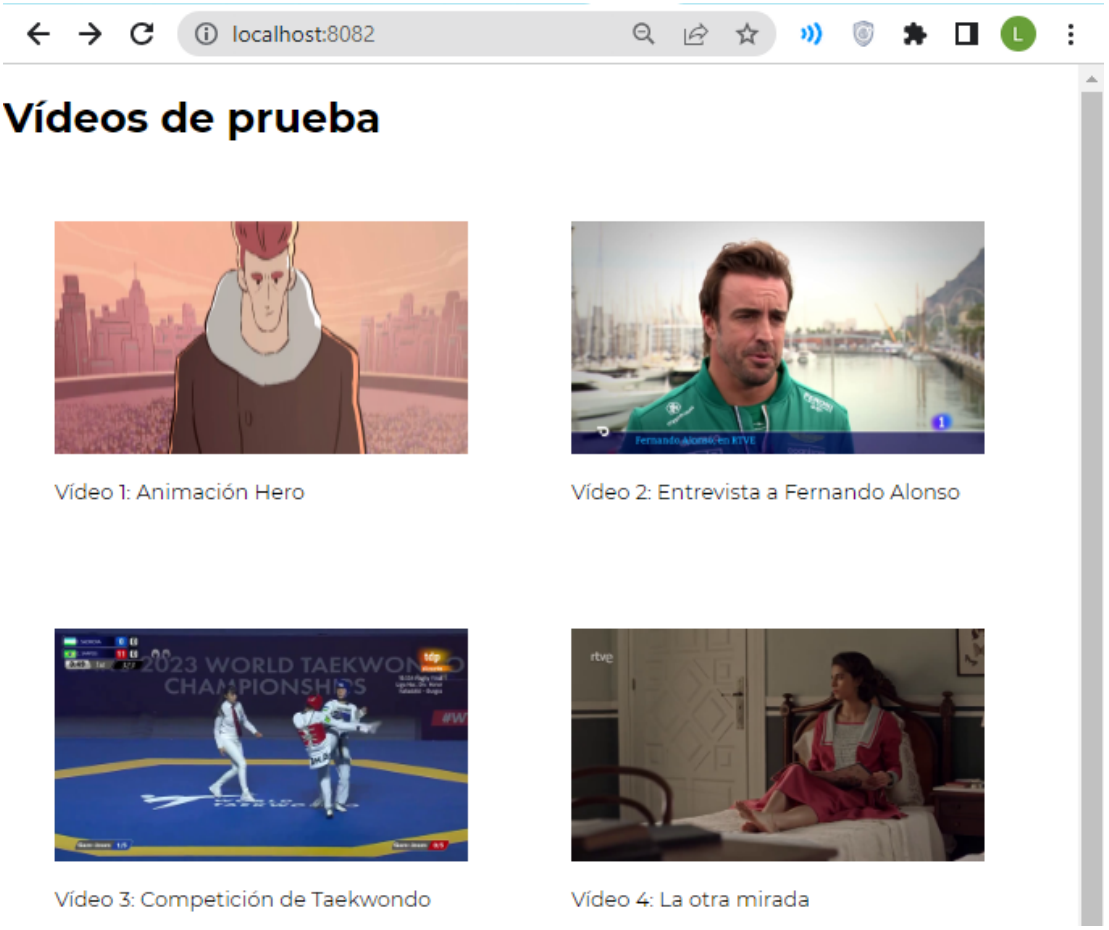


Figura 13. Aspecto de index.html

- *player.html:* Es la página de reproducción de los contenidos. En la cabecera hace referencia al fichero javascript con el reproductor de Shaka Player compilado. Incluye un tag de vídeo y un tag de script que crea y maneja el player de vídeo basado en la librería de Shaka Player. Su aspecto se muestra en la Figura 14 y su código está en el punto 4 del Anexo I.

Figura 14. Aspecto de player.html reproduciendo deportes.mp4

- *styles.css:* Da un tamaño a la clase thumbnail de 320px de ancho por 180px de alto. Asigna a la clase *movieContainer* un margen y un padding de 20px cada uno. Si se pasa el ratón por encima, el cursor pasará a puntero (*pointer*). Su código está en el punto 5 del Anexo I.

- *shaka-player.compiled.js:* Librería con el reproductor de Shaka Player compilado.

2.7 Creación de los vídeos distorsionados

2.7.1 Puppeteer

Puppeteer es una librería de Node que ofrece una API para controlar Chrome de forma automatizada sobre el protocolo DevTools. Entre los usos más comunes se encuentran generar capturas de pantalla o PDFs, probar algunas extensiones de Chrome, automatizar acciones como envío de formularios o entradas por teclado y capturar trazas de páginas web para diagnosticar problemas de rendimiento [7].

En este caso, se ha empleado Puppeteer para dos labores principales:

- *Automatizar cambios en las condiciones de red:* Se cambia el ancho de banda disponible, y por tanto el throughput de descarga de los vídeos, para que se produzcan cambios de calidad y se sirvan los segmentos de distintas resoluciones y valores de CRF elegidos en el apartado 2.3 de forma dinámica.

- *Extraer una traza con los eventos del reproductor de vídeo:* Se extraen los datos de todo lo ocurrido en un fichero JSON para su posterior tratamiento.

2.7.2 Diseño de escenarios de red

A cada uno de los 4 vídeos origen con sus 3 representaciones de calidad, se le aplican 3 escenarios de red distintos para extraer datos y evaluar cómo cambian los eventos de cambios de calidad e interrupciones en cada uno de ellos.

Cada escenario se define en un fichero JSON, en el que se especifican 2 parámetros:

- *bw:* Indica el ancho de banda que se desea fijar en kbps
- *time*: Hace referencia al tiempo en milisegundos durante el que se mantiene el ancho de banda indicado en el parámetro *bw.*

Se han diseñado escenarios de red distintos, buscando los cambios en el comportamiento del reproductor. Los valores de *bw* se han seleccionado para que el reproductor cambie entre todas las calidades disponibles.

- *Escenario 1:* El ancho de banda se va aumentando desde un valor de 100 kbps hasta un valor de 800 kbps. Se hace un cambio de *bw* cada 5000 milisegundos. El perfil con los cambios del valor de bw se muestra en la Figura 15.

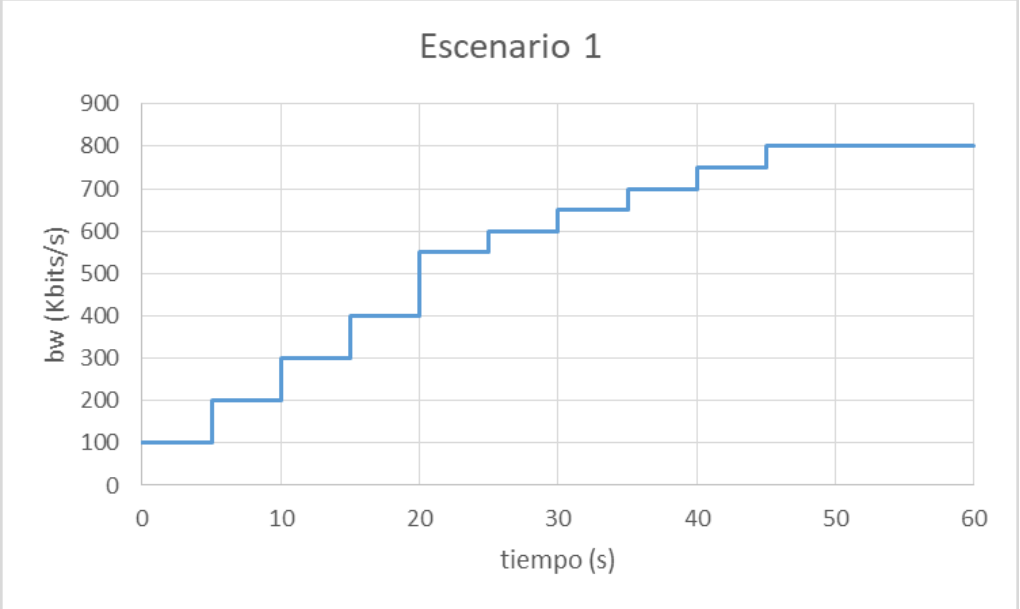


Figura 15. Perfil de cambios de ancho de banda del escenario 1

- *Escenario 2:* El ancho de banda se reduce de manera drástica hasta los 100 kbps y vuelve a aumentar hasta el valor máximo, de 800 kbps. Se repite el procedimiento de

forma cíclica hasta que el vídeo termina. Se hace un cambio de *bw* cada 5000 milisegundos. Se ha comprobado a través de la experimentación que estas condiciones son suficientes para generar interrupciones. El perfil con los cambios del valor de bw se muestra en la Figura 16.

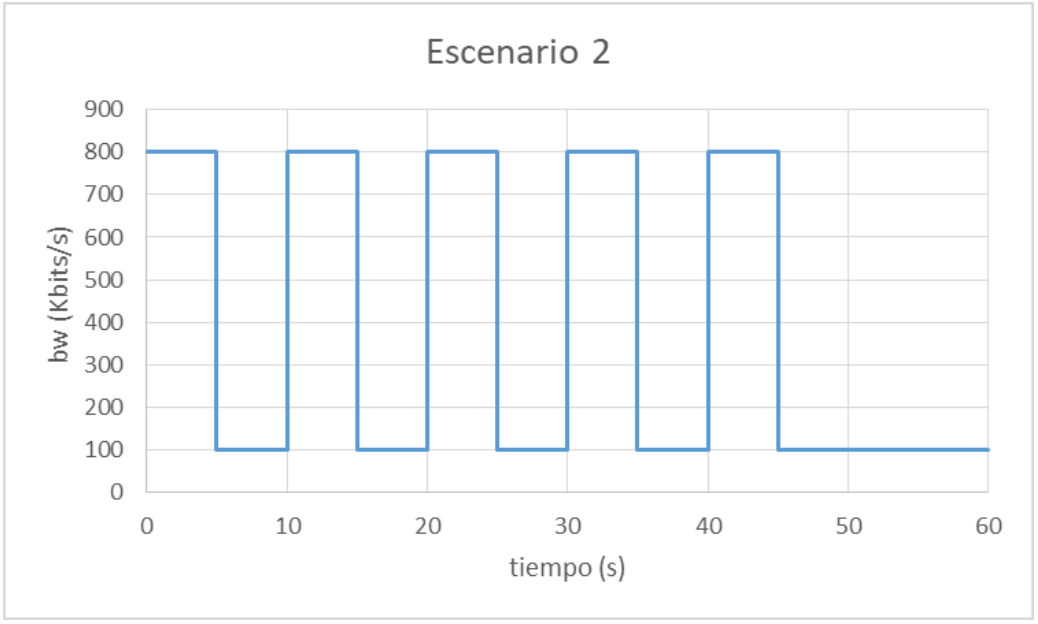


Figura 16. Perfil de cambios de ancho de banda del escenario 2

- *Escenario 3:* El ancho de banda va reduciéndose de manera paulatina desde los 800 kbps hasta los 100 kbps, y luego aumenta de forma drástica hasta el valor máximo de 800 kbps de nuevo. El primer cambio de *bw* se hace pasados 10 segundos y los demás van haciéndose cada 5 segundos. El perfil con los cambios del valor de bw se muestra en la Figura 17.

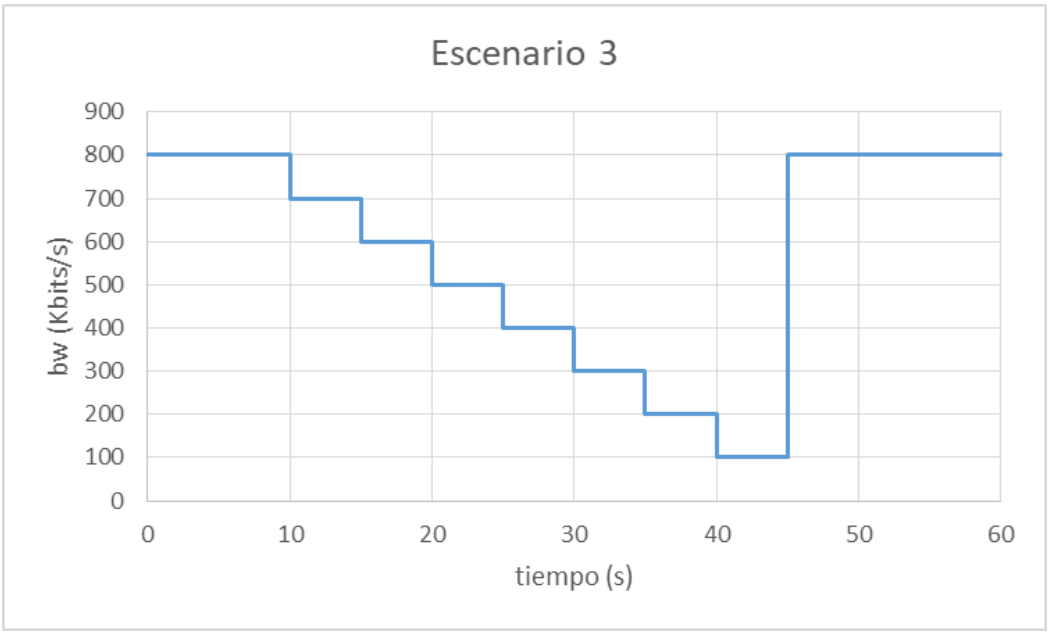


Figura 17. Perfil de cambios de ancho de banda del escenario 3

En la Figura 18 se muestra un ejemplo de la estructura del *escenario1.json.* La estructura de *escenario2.json* y *escenario3.json* siguen la misma lógica, y están en el punto 6 del Anexo I.

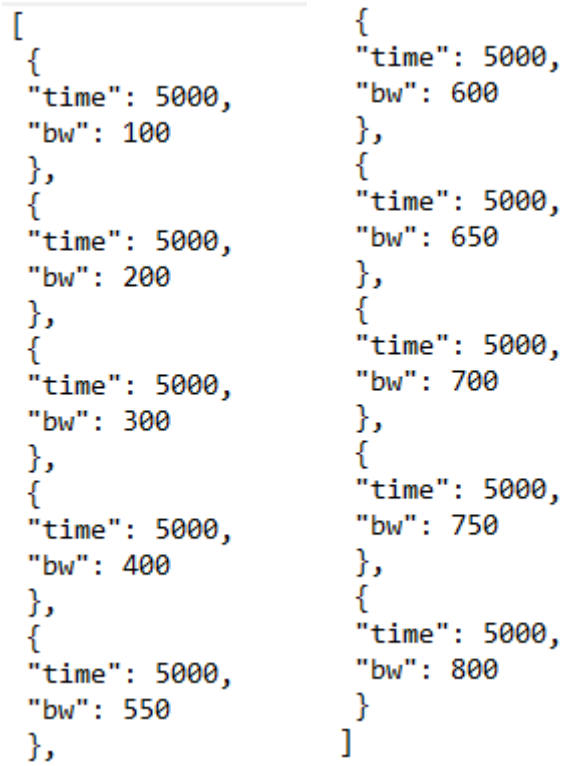


Figura 18. Contenido de escenario1.json

2.7.3 Desarrollo del script usando Puppeteer

Una vez definidos los escenarios de red, se ha elaborado un script para automatizar los cambios en las condiciones de red y extraer las trazas de los eventos de interrupción y los cambios de calidad del algoritmo de reproducción de Chrome. Para la ejecución del script se emplea el mismo servidor configurado en el apartado 2.5, empleando el contenedor *player* de Docker, que sigue activo.

En el script *trazas,js,* que está en el punto 7 del Anexo I:

1. Se define una función asíncrona que crea una variable *browser* con *devtools* en modo True para poder acceder a las herramientas de desarrollador. Se desactiva también el modo *headless* que viene por defecto para ver la ejecución y se define el *executablePath* para que use el Chrome del sistema en vez de Chromium.

2. Se crea un fichero HAR (HTTP Archive) que registra todas las interacciones entre Chrome y la página web. Captura los datos extraídos sobre las solicitudes y respuestas HTTP realizadas durante la navegación. Los resultados se almacenan en un archivo *results.json.*

3. Se crea una página *page,* se carga el manifiesto MPD y se espera a que el reproductor cargue completamente. Se usa el método *evaluate* de *page* para guardar el *timestamp* de los eventos de reproducción, pausa y finalización del vídeo.

4. Se crea una función *getBuffer* para mostrar la ocupación del buffer una vez por segundo.

5. Se crea una sesión CDP que establece una conexión entre Puppeteer y Chrome, lo que permite enviar comandos personalizados al navegador y recibir respuestas. Durante el tiempo que dura el vídeo, se aplican las condiciones definidas en los escenarios de red definidos en el apartado 2.7.2 usando el método send con el comando Network.emulateNetworkConditions. Con el método on de page se imprimen los eventos de respuesta response para saber cuándo se ha descargado cada segmento.

En la Figura 19, se muestra un extracto de los resultados mostrados por consola en una ejecución. Se pueden observar los cambios en la ocupación del buffer (*BUFFER*), el *timestamp* de los cambios de calidad (*TIMESTAMPNETWORKCONDITIONS*) y el momento de descarga de un segmento (*SEGMENTODESCARGADO*).

```
SEGMENTODESCARGADO  1687716767740  GET 200 http://localhost:8082/segmentados-2/chunk-stream3-00004.m4s
TIMESTAMPNETWORKCONDITIONS  1687716765989 http://localhost:8082/segmentados-2/chunk-stream3-00004.m4s
TIMESTAMPNETWORKCONDITIONS  1687716765989 http://localhost:8082/segmentados-2/chunk-stream3-00004.m4s
BUFFER 1687716768001   10000
BUFFER 1687716769003   9000
BUFFER 1687716770007   8000
BUFFER 1687716771006   7000
SEGMENTODESCARGADO  1687716771048  GET 200 http://localhost:8082/segmentados-2/init-stream2.m4s
TIMESTAMPNETWORKCONDITIONS  1687716770997 http://localhost:8082/segmentados-2/init-stream2.m4s
TIMESTAMPNETWORKCONDITIONS  1687716770997 http://localhost:8082/segmentados-2/init-stream2.m4s
TIMESTAMPNETWORKCONDITIONS  1687716770997 http://localhost:8082/segmentados-2/init-stream2.m4s
SEGMENTODESCARGADO  1687716771094  GET 200 http://localhost:8082/segmentados-2/chunk-stream2-00004.m4s
TIMESTAMPNETWORKCONDITIONS  1687716770997 http://localhost:8082/segmentados-2/chunk-stream2-00004.m4s
TIMESTAMPNETWORKCONDITIONS  1687716770997 http://localhost:8082/segmentados-2/chunk-stream2-00004.m4s
TIMESTAMPNETWORKCONDITIONS  1687716770997 http://localhost:8082/segmentados-2/chunk-stream2-00004.m4s
BUFFER 1687716772007   6000
BUFFER 1687716773009   5000
BUFFER 1687716774014   4000
BUFFER 1687716775020   3000
BUFFER 1687716776039   2000
SEGMENTODESCARGADO  1687716776138  GET 200 http://localhost:8082/segmentados-2/init-stream0.m4s
TIMESTAMPNETWORKCONDITIONS  1687716776005 http://localhost:8082/segmentados-2/init-stream0.m4s
TIMESTAMPNETWORKCONDITIONS  1687716776005 http://localhost:8082/segmentados-2/init-stream0.m4s
TIMESTAMPNETWORKCONDITIONS  1687716776005 http://localhost:8082/segmentados-2/init-stream0.m4s
TIMESTAMPNETWORKCONDITIONS  1687716776005 http://localhost:8082/segmentados-2/init-stream0.m4s
SEGMENTODESCARGADO  1687716776258  GET 200 http://localhost:8082/segmentados-2/chunk-stream0-00004.m4s
```

Figura 19. Extracto de la consola durante una ejecución del script

6. Por último, se usa el método *getStats()* de Shaka Player para almacenar el tiempo durante el que el reproductor está cargando información al buffer, reproduciendo el vídeo o en pausa en *STATEHISTORY*, en segundos. También se imprime el tiempo total de *buffering* y el número total de frames decodificados en *STATS*. En la Figura 20 se puede ver otro extracto de la consola con los datos mencionados anteriormente.

```
STATEHISTORY  1687716761411 buffering 4.253999948501587
STATEHISTORY  1687716765665 playing 11.631999969482422
STATEHISTORY  1687716777297 buffering 8.746999979019165
STATEHISTORY  1687716786044 playing 48.35199999809265
STATEHISTORY  1687716834396 paused 15.68500018119812
STATS 13.000999927520752 1440
FIN  1687716850322
```

Figura 20. Extracto de la consola con las estadísticas de la ejecución

La Figura 21 muestra una ejecución de Puppeteer controlando Chrome. En la parte superior del navegador se muestra el mensaje *"Un software automatizado de pruebas está controlando Chrome".*

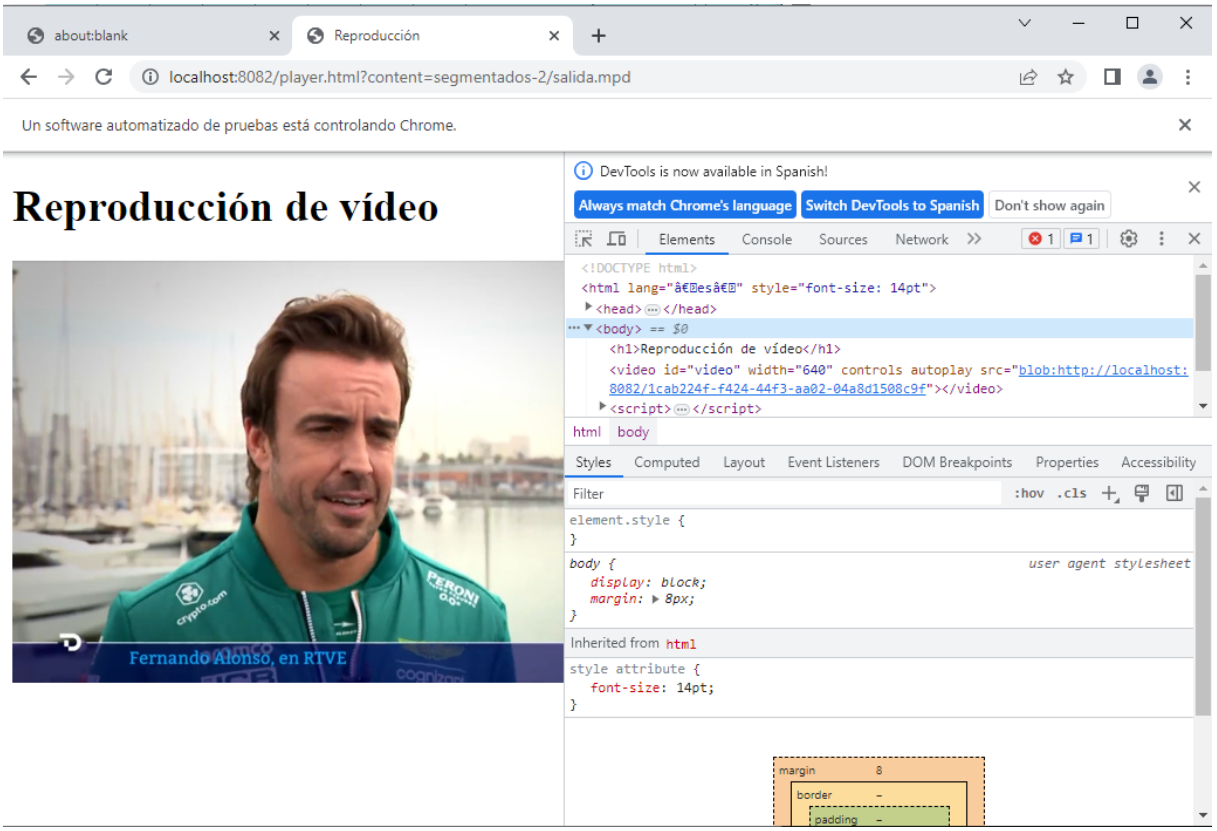


Figura 21. Puppeteer controlando Chrome durante la ejecución de trazas.js

2.7.4 Parseo de los datos obtenidos

Los datos obtenidos del fichero HAR a partir del script anterior se pasan por un nuevo script, *parseJSON.py,* cuyo contenido se encuentra en el punto 8 del Anexo I Se extraen de *results.json* para analizarlos y descomponerlos en un formato más útil y manipulable. Los parámetros extraídos son:

- Tipo de segmento
- *Timestamp* con el tiempo de inicio de la solicitud del segmento
- Tamaño de segmento en bytes
- Tiempo que tarda en recibirse el segmento
- Tiempo que la solicitud ha estado bloqueada antes de enviarse
- Tiempo que tarda en enviarse el segmento después del bloqueo

- Tiempo que la solicitud ha estado en espera antes de recibir respuesta
- Tiempo que se tarda en establecer la conexión con el servidor tras enviar la solicitud.

Todos los resultados extraídos de *results.json* se almacenan en *results.txt.* Además, se usa la librería de Python, *pandas,* para convertir los resultados en un archivo CSV *results.csv* desde el que resulte más cómodo leer los datos.

2.7.5 Reconstrucción de los vídeos distorsionados

A partir de los datos obtenidos con *getStats()* y de los segmentos de los que se dispone en todas las calidades, es posible reconstruir vídeos en los que se recreen los cambios de calidad y las interrupciones que han ocurrido bajo unas condiciones de red concretas. Estos vídeos estarán distorsionados, por tanto, y servirán como entrada de los modelos de evaluación de QoE de las partes 3 y 4 del trabajo.

Al haber segmentado los vídeos en ficheros .m4s en el apartado 2.4, este proceso de reconstrucción resulta más sencillo, puesto que los vídeos distorsionados pueden construirse concatenando dichos segmentos.

En el script *reconstruccion.py*, visible en el punto 9 del Anexo I, se leen los resultados de *results.csv* y se extraen, usando la librería *pandas*:

- Una lista con todos los segmentos (chunks) que se han descargado durante la reproducción
- Una lista con todos los segmentos de tipo init (inits) que se han descargado durante la reproducción

Con los segmentos almacenados en ambas listas se obtienen los segmentos que se han reproducido correctamente (completos), que son los últimos en la lista para cada número XX distinto de segmento (*chunk-streamN-000XX.m4s)*.

Por ejemplo, para la lista:

chunks =

['chunk-stream3-00001.m4s', 'chunk-stream0-00001.m4s', chunk-stream0-00002.m4s', 'chunk-stream2-00003.m4s', 'chunk-stream2-00004.m4s', 'chunk-stream0-00005.m4s', 'chunk-stream2-00006.m4s', 'chunk-stream0-00007.m4s', 'chunk-stream2-00008.m4s', 'chunk-stream2-00009.m4s', 'chunk-stream0-00009.m4s', 'chunk-stream0-00010.m4s', 'chunk-stream2-00011.m4s', 'chunk-stream0-00012.m4s', 'chunk-stream0-00013.m4s', 'chunk-stream2-00013.m4s', 'chunk-stream2-00014.m4s', 'chunk-stream0-00015.m4s']

correspondiente a aplicar el escenario 2 al vídeo *serie.mp4*, los chunks que se han reproducido correctamente son:

completos =

['chunk-stream0-00001.m4s', 'chunk-stream0-00002.m4s', 'chunk-stream2-00003.m4s', 'chunk-stream2-00004.m4s', 'chunk-stream0-00005.m4s', 'chunk-stream2-00006.m4s', 'chunk-stream0-00007.m4s', 'chunk-stream2-00008.m4s', 'chunk-stream0-00009.m4s',

'chunk-stream0-00010.m4s', 'chunk-stream2-00011.m4s', 'chunk-stream0-00012.m4s', 'chunk-stream2-00013.m4s', 'chunk-stream2-00014.m4s', 'chunk-stream0-00015.m4s']

Para que los segmentos puedan reproducirse, antes de concatenarse entre sí es necesario que cada uno de ellos se concatene con su init correspondiente, porque contiene información necesaria para la reproducción del segmento. Así, para cada segmento incluido en la lista de segmentos completos, en función del nombre del segmento (*chunk-stream0, chunk-stream1, chunk-stream2 o chunk-stream3),* se usa el comando *type* de Windows para concatenarlo con el init que corresponda.

Por ejemplo, para el primer segmento de lista anterior, que es de la calidad más baja (*chunk-stream0*), el script ejecuta el siguiente comando:

type init-stream0.m4s chunk-stream0-00001.m4s > segmento0-1.mp4

El resultado es una lista de segmentos que ya disponen de la información suficiente para ser concatenados entre sí.

Por ejemplo, para el caso anterior la lista resultante tendría el siguiente aspecto:

chunks_con_inits =

['segmento0-0.mp4', 'segmento0-1.mp4', 'segmento2-2.mp4', 'segmento2-3.mp4', 'segmento0-4.mp4', 'segmento2-5.mp4', 'segmento0-6.mp4', 'segmento2-7.mp4', 'segmento0-8.mp4', 'segmento0-9.mp4', 'segmento2-10.mp4', 'segmento0-11.mp4', 'segmento2-12.mp4', 'segmento2-13.mp4', 'segmento0-14.mp4']

Sin embargo, como en el apartado 2.2 se han generado vídeos con distintas resoluciones, es necesario reescalar los segmentos a unas dimensiones únicas para que sea posible concatenarlos correctamente. Los vídeos se escalan a 720p y con el parámetro CRF = 0, para asegurar que mantienen su calidad original. El resultado es la misma lista de segmentos, pero ahora reescalados.

videos_recod =

['recod_segmento0-0.mp4', 'recod_segmento0-1.mp4', 'recod_segmento2-2.mp4', 'recod_segmento2-3.mp4', 'recod_segmento0-4.mp4', 'recod_segmento2-5.mp4', 'recod_segmento0-6.mp4', 'recod_segmento2-7.mp4', 'recod_segmento0-8.mp4', 'recod_segmento0-9.mp4', 'recod_segmento2-10.mp4', 'recod_segmento0-11.mp4', 'recod_segmento2-12.mp4', 'recod_segmento2-13.mp4', 'recod_segmento0-14.mp4']

Por último, usando el método *concat* de FFMPEG se concatenan todos los segmentos de la lista de *videos_recod,* generándose un vídeo de 1 minuto de duración (como los originales) con todos los cambios de calidad que han ocurrido a lo largo de la reproducción con un escenario de unas condiciones de red concretas.

El único paso restante para conseguir la reconstrucción completa de los vídeos distorsionados es el de añadir las interrupciones que hayan ocurrido durante cada reproducción usando Puppeteer. Para ello, se ha elaborado el script *interrupciones.py*, visible en el punto 10 del Anexo I.

En el script se introducen los valores recogidos con la función *getStats()* en el apartado 2.7.3; es decir, se extraen:

- Una lista con los valores temporales de los instantes, en segundos, en los que el vídeo se ha quedado parado porque el buffer se ha vaciado.
- Una lista con las duraciones de las interrupciones.

De nuevo empleando FFMPEG, se genera un vídeo para cada interrupción, extrayendo el frame del instante temporal en el que se ha producido la parada

ffmpeg.input(input_video, ss=frame_time[i]).output('frame{}.png'.format(i), vframes=1).run()

y repitiéndolo en bucle durante el tiempo que haya persistido la misma:

ffmpeg.input('frame{}.png'.format(i), loop=1, t=duracion[i]).output(output_video).run()

A continuación, se divide el vídeo generado de *reconstruccion.py* en todos los momentos en los que se hayan producido interrupciones, Así, si por ejemplo el vídeo se ha parado en los instantes 0, 35 y 47 segundos, el vídeo original se dividirá en 3 vídeos: uno de los 0 a los 35 segundos, otro de los 35 a los 47 segundos, y un último de los 47 a los 60 segundos totales.

Por último, se concatenan los trozos de vídeo con cambios de calidad y los vídeos generados que representan las interrupciones en el orden correcto haciendo uso de *ffmpeg.concat* de nuevo. Se obtiene un vídeo distorsionado, con cambios de calidad e interrupciones tal y como han ocurrido simulando unas condiciones de red específicas. Su duración se calcula como:

Duración vídeo distorsionado = Duración vídeo original + Duración interrupciones

En la Figura 22 se encuentra resumido todo el proceso de reconstrucción de los vídeos.

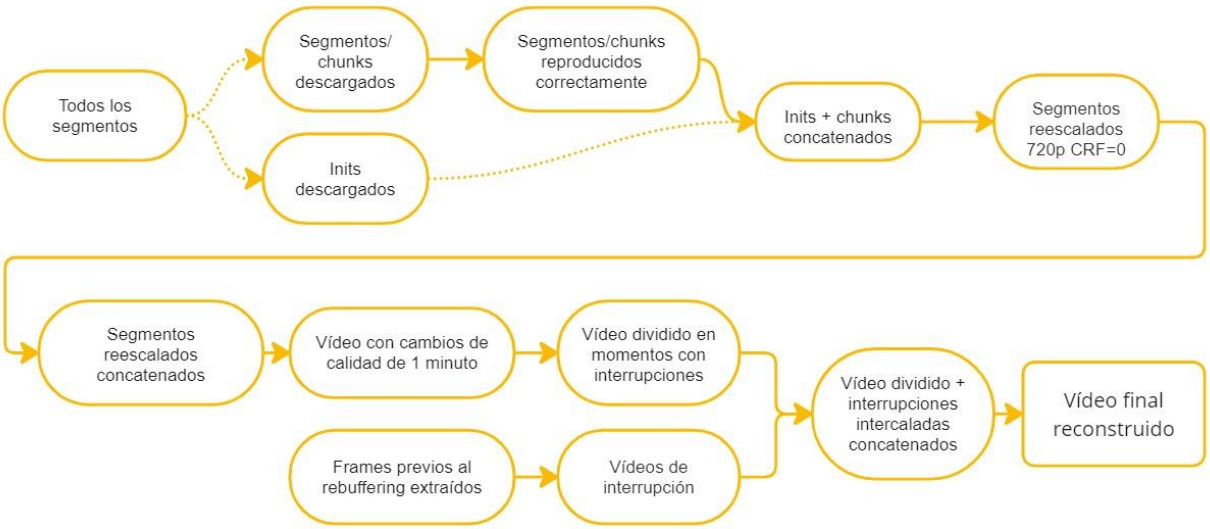


Figura 22. Proceso de reconstrucción de los vídeos distorsionados.

PARTE III: MODELADO HÍBRIDO. RECOMENDACIÓN ITU P.1203

3.1 Recomendación P.1203 de la ITU-T

La recomendación P.1203 de la Unión Internacional de Telecomunicaciones (ITU) hace una *"evaluación paramétrica de la calidad basada en el tren de bits de los servicios audiovisuales de emisión de secuencias de descarga progresiva y adaptativa a través de un transporte fiable".* [8] Como se menciona en el apartado 1.3.1.1, la ITU-T Rec. P.1203 es una de las recomendaciones más recientes para servicios HAS.

El modelo de predicción de la recomendación P.1203 usa una escala muy similar a la de tipo MOS, explicada en el apartado 1.2.4.2. Es decir, otorga a los vídeos que entran como input una puntuación del 1 al 5, aunque en este caso incluye decimales. Se basa en calificaciones subjetivas de la calidad de la experiencia obtenidas en experimentos con participantes reales, así como en un modelo paramétrico para la evaluación de la calidad audiovisual del streaming de vídeo adaptativo a partir de parámetros de QoS. Es por ello un modelo de tipo híbrido.

Como se observa en la Figura 23, el modelo consta de tres bloques principales: un módulo de estimación de la calidad de vídeo (P.1203.1, Pv), un módulo de estimación de la calidad de audio (P.1203.2, Pa) y un módulo de integración de la calidad (P.1203.3, Pq) compuesto por módulos de integración temporal audio/vídeo y de estimación del impacto de los cortes.

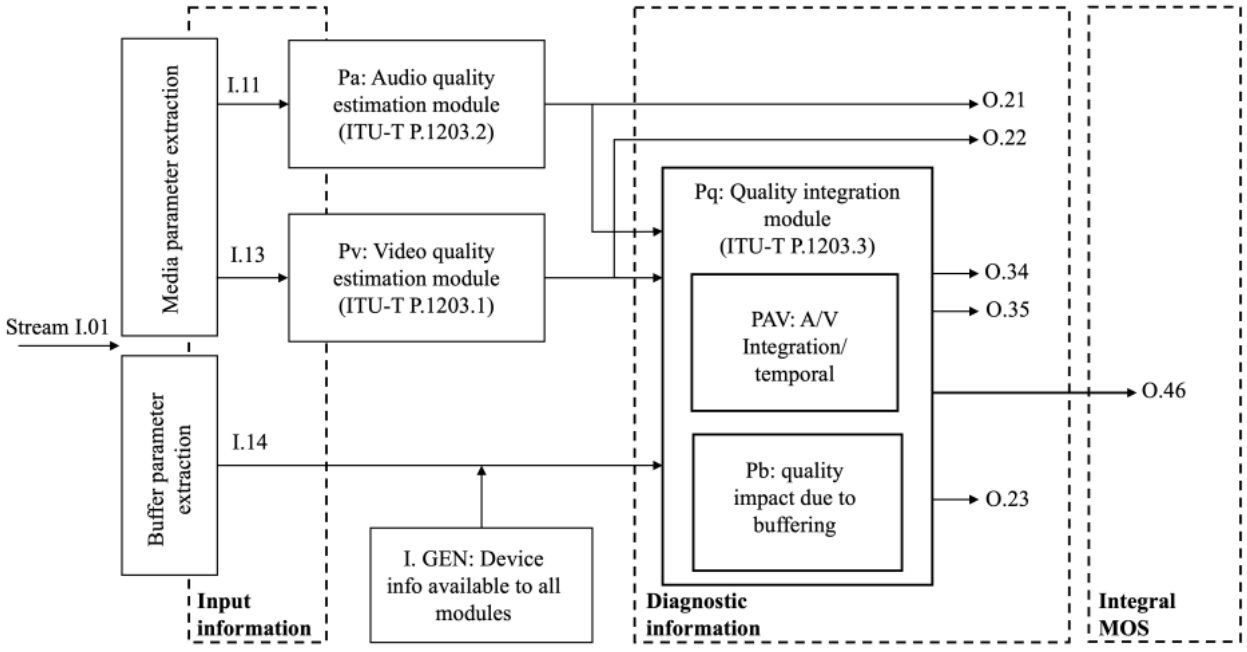


Figura 23. Bloques del modelo ITU-T P.1203 [14]

El modelo Pv puede funcionar en cuatro modos de operación, desde el 0 hasta el 3. Dichos modos se distinguen según la cantidad de información de entrada disponible I.13. [9]

- Modo 0: Utiliza la información de los metadatos: códec, resolución, bitrate de cada representación, retardo inicial, duración de los segmentos e información sobre las paradas producidas durante la reproducción.

- Modo 1: Utiliza la información que usa el modo 0 añadiendo la información de los flujos multimedia que se extrae de la cabecera de los paquetes, es decir, los tipos y los tamaños de los frames.
- Modo 2: Utiliza la información que usa el modo 1 añadiendo un 2% de los valores QP de todos los frames.
- Modo 3: Utiliza la información que usa el modo 1 añadiendo todos los valores QP de los frames del input de vídeo.

Cada uno de los modos es más complejo en cuanto a información de entrada y a algoritmos, pero también más exacto que el anterior [14].

Existe un repositorio en GitHub [19a] que contiene la implementación de esta recomendación en Python. La ITU-T Rec. P.1203 Standalone Implementation se puede emplear para vídeos codificados en H.264 con resolución Full HD (1920x1080) o inferior y de duración comprendida entre los 30 segundos y los 5 minutos. Dado que los vídeos distorsionados que se han generado en la parte II del trabajo cumplen todos los requisitos, se ha decidido utilizar dicha implementación como modelo híbrido de la QoE.

3.2 Uso del repositorio ITU-T Rec. P.1203 Standalone Implementation

El software permite la entrada de los vídeos como input. Una vez clonado el repositorio y desde la línea de comandos, se puede ejecutar el comando que calcula la QoE, y en él pueden especificarse varias opciones. Las elegidas son las siguientes:

python3 -m itu_p1203 video.mp4 --mode {0,1,2,3} --only-pv

- --mode indica el modo elegido para la evaluación de la QoE cuando se cargan vídeos como input.
- --only-pv indica que se desea utilizar el modelo P.1203.1, también llamado Pv.

Se ha ejecutado el comando anterior para cada cada uno de los 12 vídeos distorsionados generados en la parte II, además de para los 4 originales. Para todos ellos se han usado los 4 modos que proporciona el script.

Todos los modelos proporcionan una lista en formato JSON como salida con una puntuación de calidad de vídeo por segundo. El modo 0 proporciona un único valor de calidad (el mismo valor cada segundo), lo que es lógico teniendo en cuenta que solo dispone de la información que se proporciona en los metadatos. Se puede ver la salida por la terminal en la Figura 24.

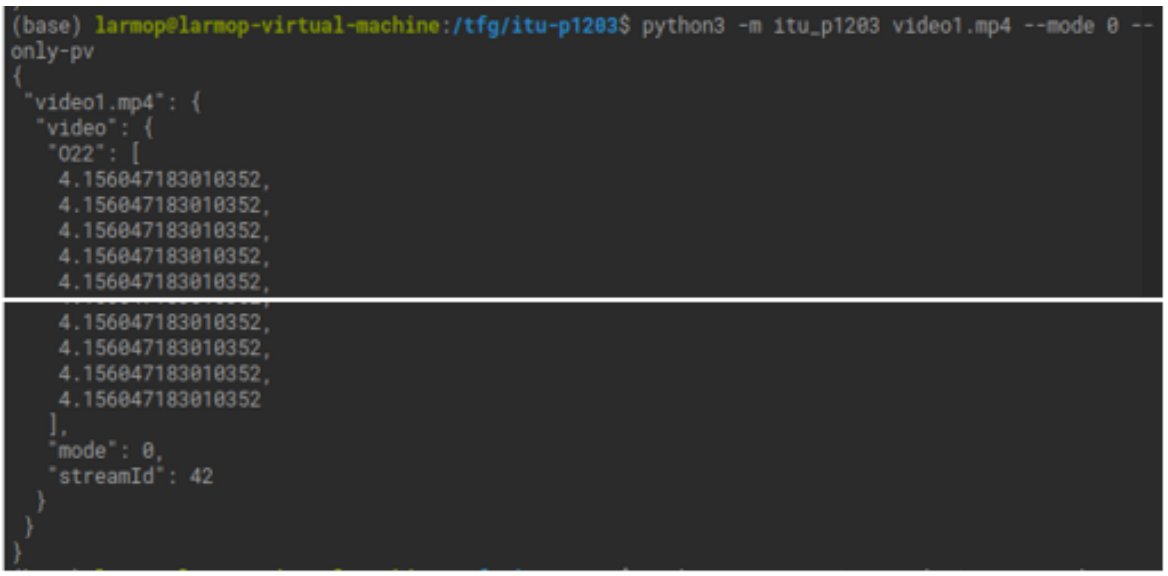


Figura 24. Ejecución del modo 0 para uno de los vídeos

El resto de los modos proporcionan valores distintos de calidad de vídeo por segundo. Los modos 2 y 3 necesitan de un script extractor de QPs que se incluye en el repositorio. En la Figura 25 se puede ver la ejecución (con la lista de valores recortada) del modo 3 para uno de los vídeos.

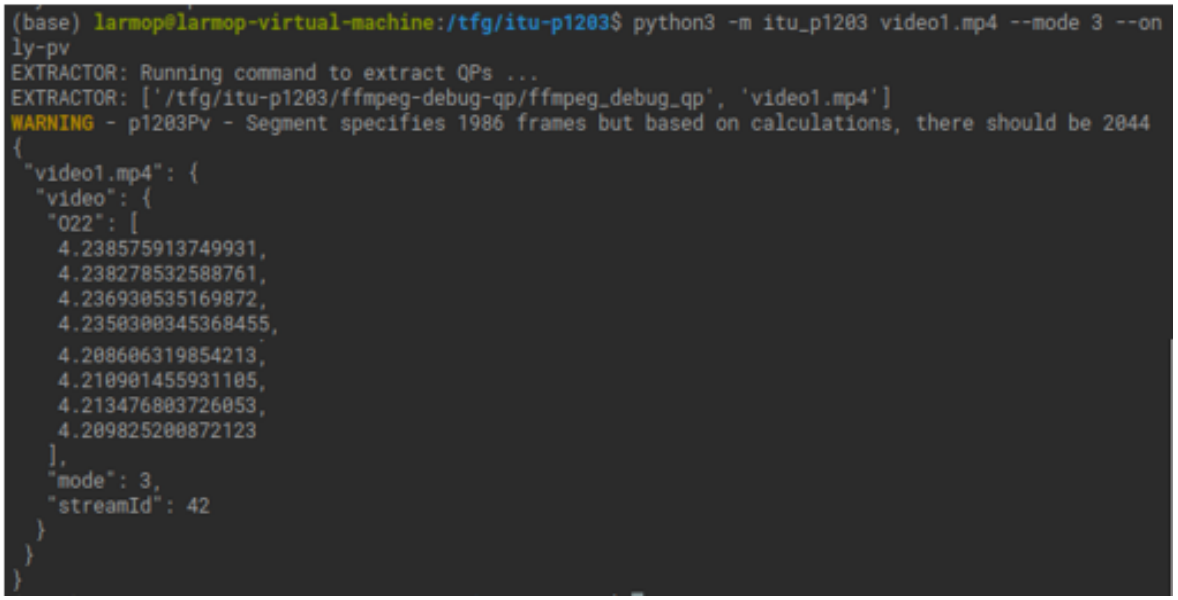


Figura 25. Ejecución del modo 3 para uno de los vídeos

Una vez obtenidos todos los valores, se han exportado a un fichero .txt y se ha utilizado una hoja Excel para calcular el valor medio de cada lista generada. Con los resultados de los valores medios se ha elaborado una tabla final en la que se recogen las puntuaciones del modelo híbrido para cada vídeo y cada modo. La Tabla 4 con dichos resultados se muestra en el apartado 5.1.

PARTE IV: MODELADO SUBJETIVO. EXPERIMENTO DE VISIONADO

4.1 Selección del grupo de visionado y diseño del experimento

Para la realización del experimento de visionado se ha elegido a 20 personas pertenecientes a distintos segmentos demográficos: hombres y mujeres de edades contenidas entre los 16 y los 60 años situados en ubicaciones geográficas distintas. Además, usan dispositivos de visualización distintos y están expuestos a diferentes condiciones lumínicas; datos que servirán, entre otros, para contextualizar los resultados.

El diseño del experimento consiste en presentar a los individuos los 12 vídeos generados en el apartado 2.7.5; es decir, los vídeos distorsionados que han servido también como entrada del modelo expuesto en la parte III del trabajo. Se les presentan también los 4 vídeos originales. Para cada vídeo, se pide al grupo de visionado que elija una puntuación del 1 al 5 según la calidad que hayan percibido, siendo 1 la puntuación más baja posible y 5 la más alta.

En total, el test incluye un total de 16 vídeos y un formulario de preguntas al final, que recoge información sobre las condiciones lumínicas de la habitación en la que se encuentran los espectadores, el tipo de dispositivo de visualización, su nivel de atención durante la realización del test, su nivel de energía y si han pausado, avanzado o retrocedido alguno de los vídeos.

La prioridad a la hora de presentar los contenidos en el experimento de visionado es asegurar que la reproducción de los vídeos no está sujeta a los cambios de las condiciones de red. Si eso ocurriera, los resultados obtenidos no serían representativos en absoluto, puesto que el objetivo del experimento es precisamente evaluar los vídeos originales bajo unas condiciones de red preestablecidas y controladas. Esta condición, unida a las posibilidades que ofrece que el test esté alojado en una página web, como que pueda realizarse en cualquier momento y lugar, han dado lugar a que se implemente el experimento en una web con reproductores de vídeo embebidos. Los vídeos se sirven como archivos mp4 completos, de forma que las condiciones de red no alteran su reproducción.

4.2 Elaboración de los HTML y la hoja de estilos CSS asociada

Se han elaborado 19 ficheros HTML: uno para la introducción y la explicación del funcionamiento del test (*index.html),* 16 para alojar cada vídeo y su correspondiente desplegable de puntuaciones (*pagina2.html* a *pagina17.html*), uno con las preguntas finales (*formulario.html*) y uno para agradecer la participación en el experimento (*final.html*). También se ha creado una hoja de estilos en cascada CSS para diseñar los estilos de las páginas (*styles.css*).

4.2.1 Estructura de *index.html*

En *index.html*, cuyo código está en el punto 12 del Anexo I, se incluye una explicación del objetivo del test y de qué tipo de contenidos se van a visualizar a lo largo del mismo. Además, se explica cómo avanzar entre las páginas y el sistema de puntuación de los vídeos. En la Figura 26 se muestra el aspecto de *index.html.* En el archivo CSS asociado se establecen el color del fondo y el tamaño y el tipo de fuente.

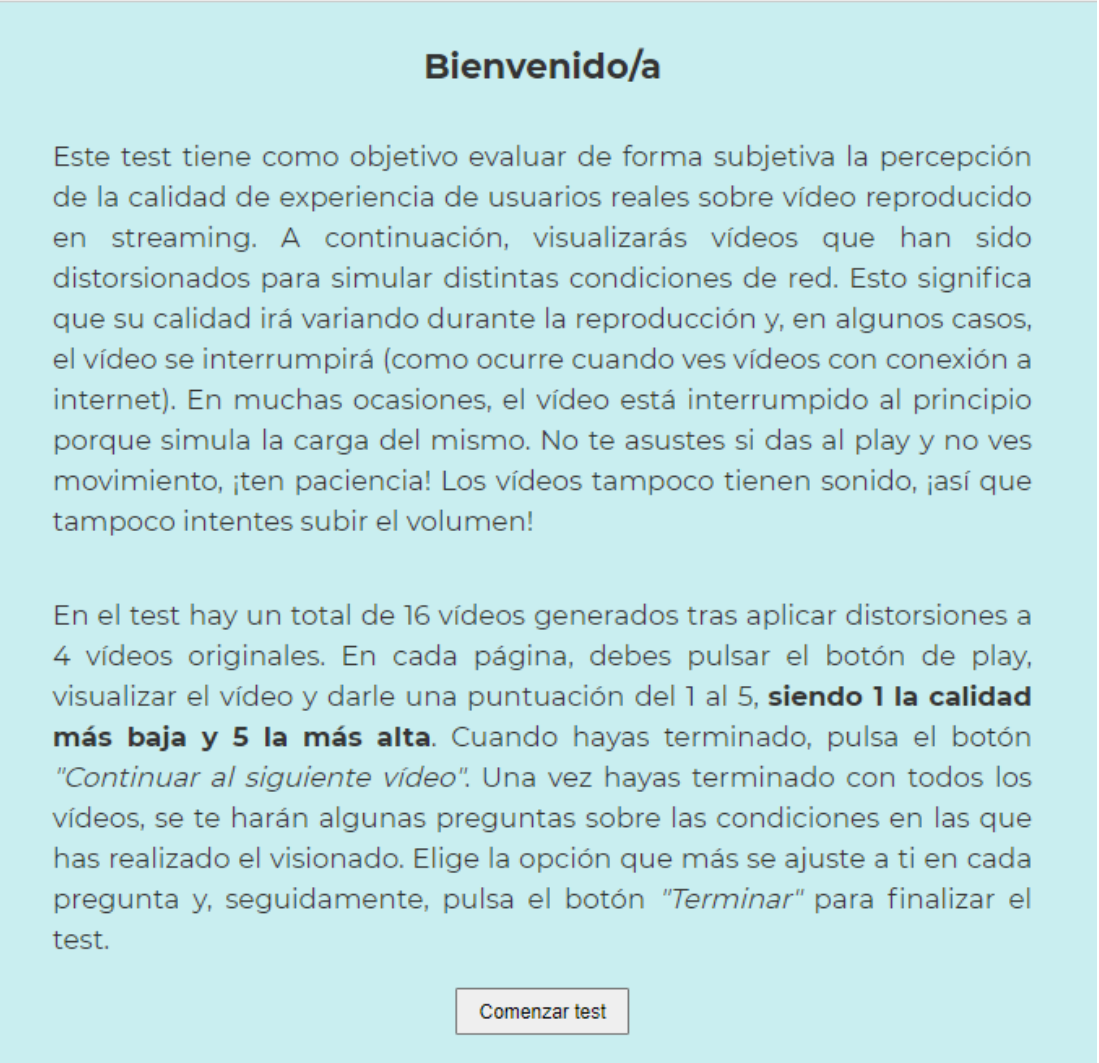


Figura 26. Aspecto de index.html del test

4.2.2 Estructura de las páginas con vídeos embebidos: de *pagina2.html* a *pagina17.html*

Las páginas HTML incluyen:

- Un vídeo embebido en un tag de *video.*
- Un formulario *form* con opciones del 1 al 5, un atributo *action="/submit"* y un atributo *method="POST"* para enviar posteriormente los valores a la base de datos.
- Un botón de tipo *submit* con el texto *Continuar al siguiente vídeo.*

En la Figura 27 se muestra el aspecto de *pagina2.html*, cuyo código está en el punto 13 del Anexo I. En el archivo CSS asociado se eligen los estilos necesarios para que el reproductor de vídeo, el desplegable y el botón sean del tamaño adecuado y estén centrados.

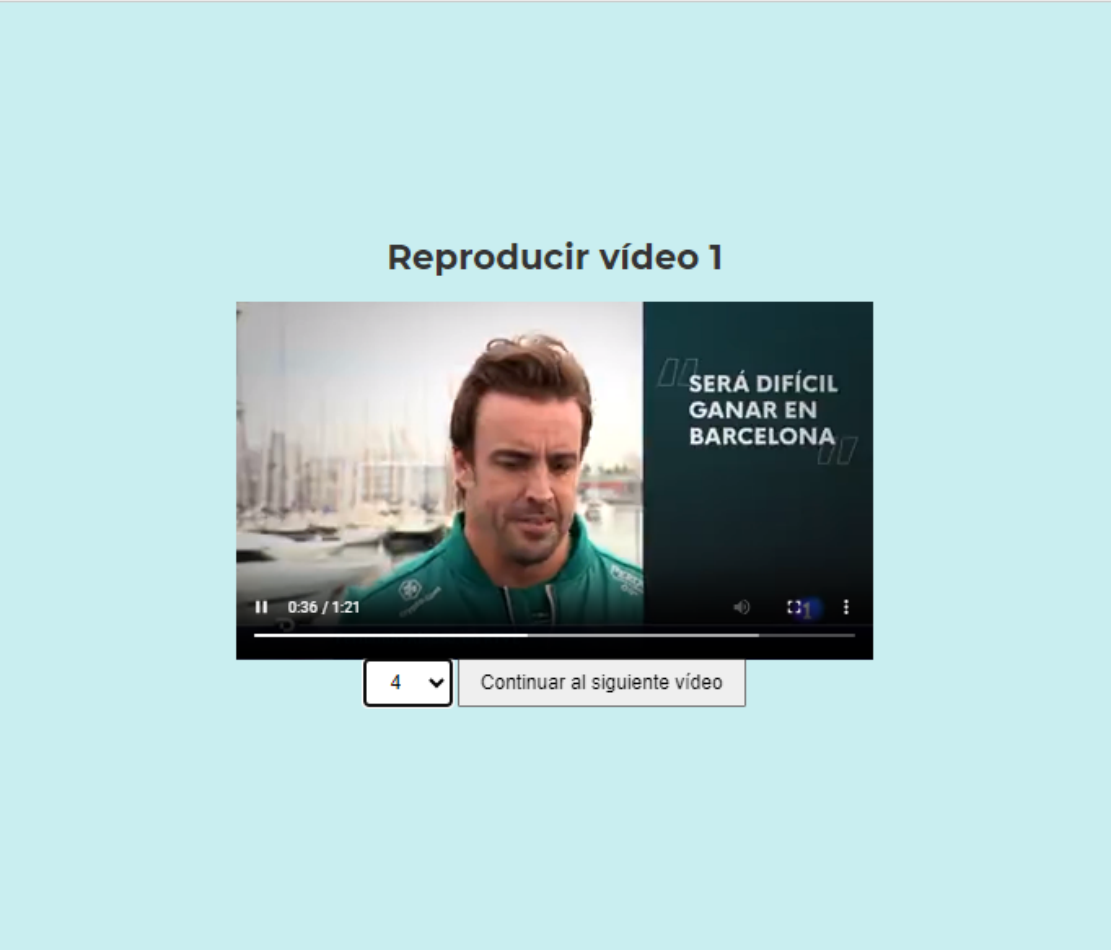


Figura 27. Aspecto de pagina2.html

4.2.3 Estructura del cuestionario: *formulario.html*

En *formulario.html* se incluyen:

- Un formulario *form* con los atributos *action="/submit"* y *method="POST"*, que incluye 5 preguntas. Cada pregunta tiene varios inputs de tipo *radio* para mostrar las opciones de respuesta.
- Un botón con el texto *Terminar* que lleva a la última página HTML.

En las Figuras 28 y 28 se muestra el aspecto de *formulario.html.* El código correspondiente puede verse en el punto 14 del Anexo I. En el archivo CSS asociado se eligen los estilos necesarios para que el texto de la página tenga un tamaño y una colocación distintos al del resto de páginas.

4.2.4 Estructura de la última página: *final.html*

La estructura de la última página del test es sencilla. Incluye un texto de agradecimiento por haber realizado el test. En la Figura 30 se muestra el aspecto de *final.html.* Su código está en el punto 15 del Anexo I. Se aplican también los estilos de *styles.css.*

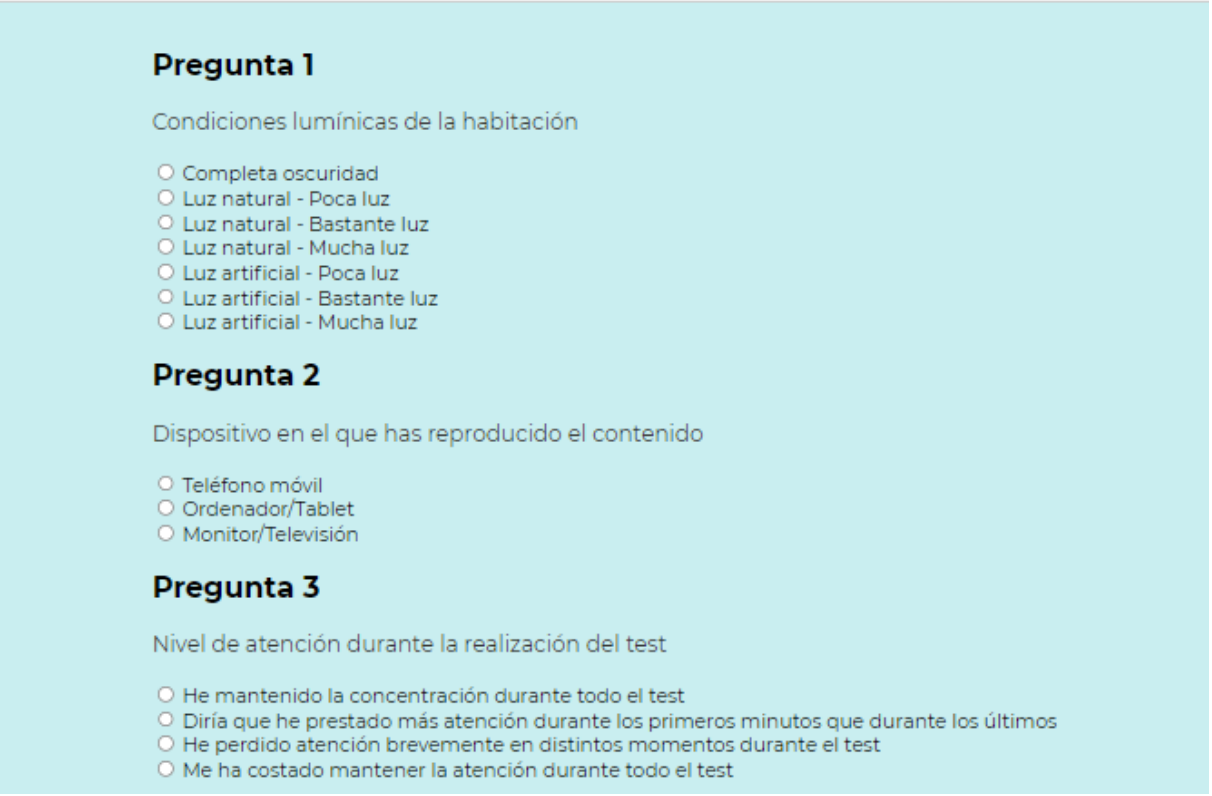


Figura 28. Aspecto de formulario.html. Parte 1

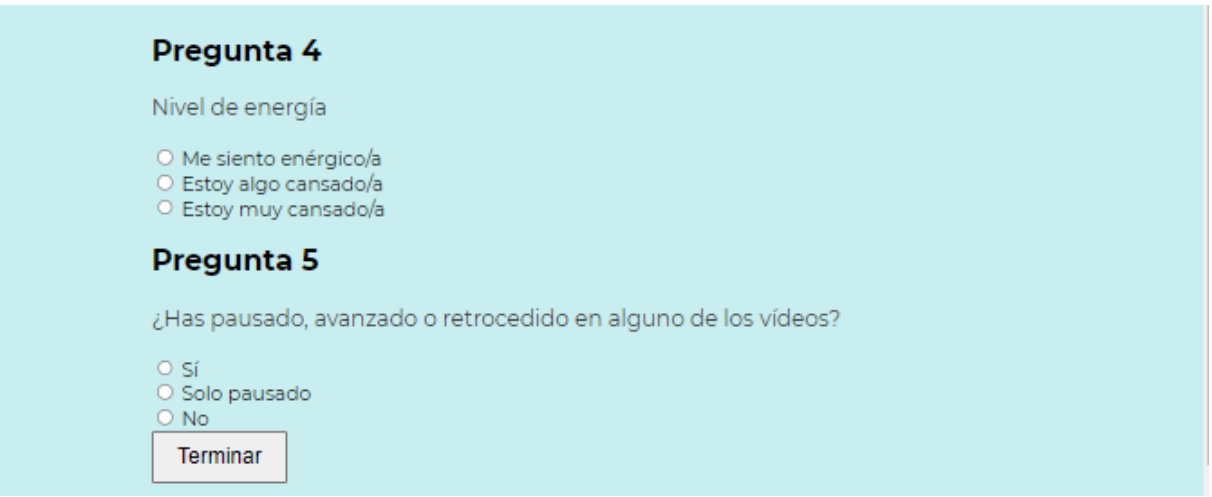


Figura 29. Aspecto de formulario.html. Parte 2

4.3 Elaboración del archivo JS para servir los contenidos y conectar con MySQL

Una vez creados todos los ficheros HTML necesarios, hace falta un archivo Javascript *app.js,* cuyo contenido está en el punto 11 del Anexo I, que lleve a cabo 3 tareas principales:

1. Configuración, creación e inserción de datos en la base de datos de MySQL *mydatabase*
2. Servicio de los ficheros HTML en la web y extracción de parámetros para su posterior inserción en la BBDD
3. Inicio del servidor en el puerto 3000

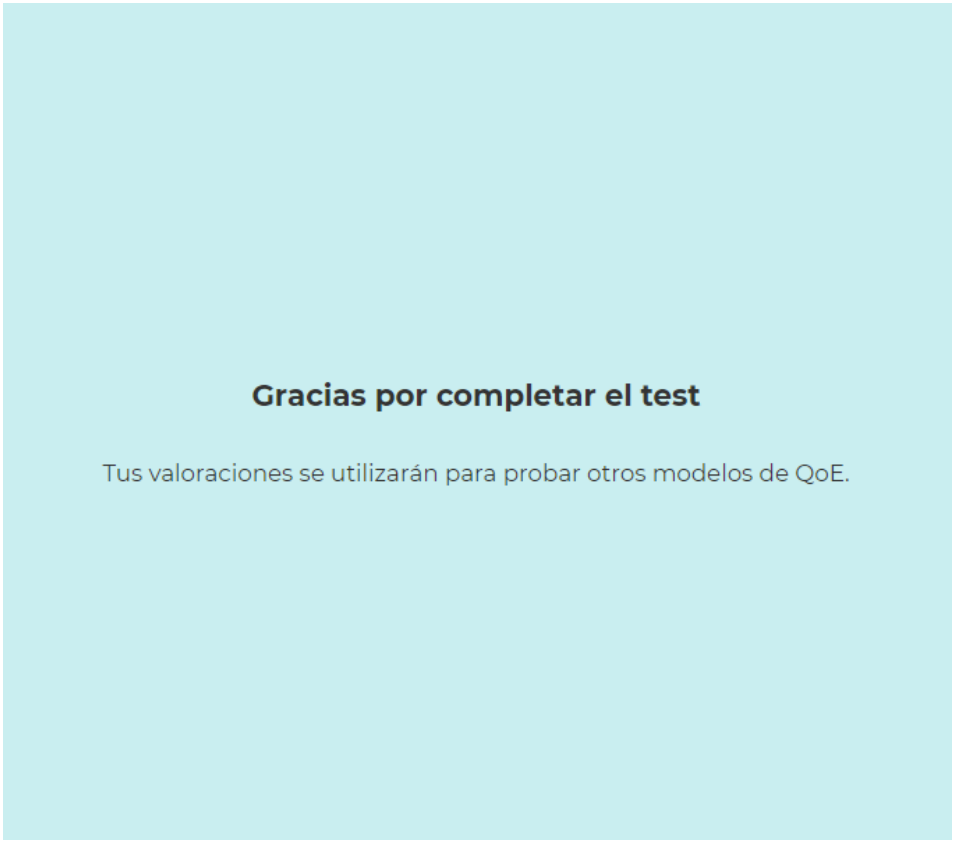


Figura 30. Aspecto de final.html

En primer lugar, se configura la conexión a la base de datos pasando como parámetros el host, el usuario, la contraseña, el nombre de la base de datos y el puerto. Después, se crea la conexión:

const connection = mysql.createConnection(dbConfig);

Se hace uso del middleware *body-parser* para Node.js, que extrae los datos del cuerpo de la solicitud y los asigna a la propiedad req.body para acceder a ellos con más facilidad.

También se emplea el framework de Node.js *express* para manejar las solicitudes HTTP GET y POST. Además, se usa el middleware *express-session* para ir almacenando los datos de la sesión hasta su inserción en la BBDD.

Se procesa la solicitud GET y se almacena el valor del parámetro *page,* que corresponde con el nombre del HTML que se quiere servir en cada momento. La línea *app.use(express.static(__dirname));* permite que no se sirvan sólo los ficheros HTML, sino también los vídeos y la hoja de estilos CSS asociada.

El servidor está configurado para manejar las solicitudes POST en la ruta */submit* especificada como atributo en todos los formularios de los ficheros HTML. De esta forma, se obtienen todos los datos que se han enviado desde el test. Por ejemplo, la primera puntuación se almacena como

const pregunta1 = req.body.pregunta1;

Los valores se almacenan en una variable única *storedValues*, que se va actualizando con los valores de las variables *pregunta1* a *pregunta21*. Si la variable es *undefined*, se asigna el valor previamente almacenado en storedValues para esa propiedad, asegurando que los valores se mantienen a lo largo de la sesión.

Una vez que todos los campos se han completado, es decir, que el espectador ha completado el test, se realiza la inserción de los datos en la base de datos a través de una consulta SQL.

const sql = 'INSERT INTO puntuaciones SET ?';

const values = storedValues;

Se reinicia la variable *storedValues* para que los datos del siguiente test se inserten en una nueva fila de la tabla de la BBDD.

req.session.storedValues = {};

Para que, además de enviar los datos de los formularios, los botones de cada archivo HTML sirvan para acceder a la página siguiente del test en función del valor de la página actual, se usa la propiedad *redirect* para pasar a la siguiente.

if (currentPage === 'pagina2') {

res.redirect('/?page=pagina3');

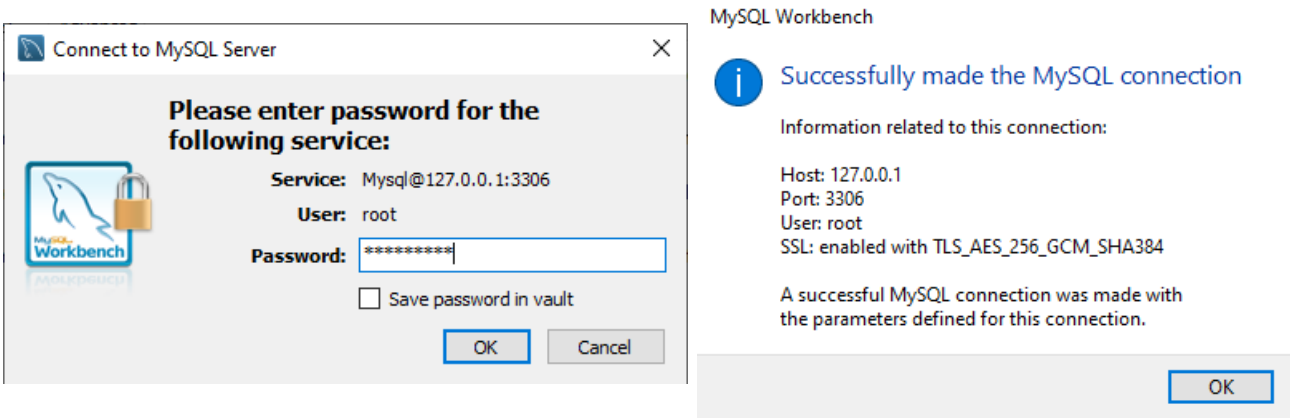
Por último, se inicia el servidor en el puerto 3000.

app.listen(3000);

4.4 Elaboración de la tabla para guardar los resultados en MySQL

Para almacenar los resultados enviados desde *app.js* se ha creado una base de datos en MySQL. Haciendo uso de la interfaz gráfica MySQL WorkBench [20], una vez se ha realizado una conexión a un servidor MySQL, se crea una nueva base de datos de nombre *mydatabase*. En realidad, es necesario hacer este paso antes de empezar con todo el proceso explicado en la sección 4.3.

Se establece la conexión con el servidor y se comprueba que haya sido satisfactoria, como se muestra en las Figuras 31 y 32.



Figuras 31 y 32. Conexión de la base de datos con el servidor MySQL

Se crea la tabla para alojar los valores extraídos del test. En la Figura 33 se puede ver su estructura. En la tabla se incluye una columna con un valor de identificación *ID*, los valores de las preguntas 1 a 21 (las 16 valoraciones de vídeos y las respuestas a las 5 preguntas del cuestionario final) y el timestamp con el día y la hora en la que se ha realizado el test *fecha_registro.*

```
CREATE TABLE puntuaciones (
    id INT AUTO_INCREMENT PRIMARY KEY,
    pregunta1 VARCHAR(255),
    pregunta2 VARCHAR(255),
    pregunta3 VARCHAR(255),
    pregunta4 VARCHAR(255),
    pregunta5 VARCHAR(255),
    pregunta6 VARCHAR(255),
    pregunta7 VARCHAR(255),
    pregunta8 VARCHAR(255),
    pregunta9 VARCHAR(255),
    pregunta10 VARCHAR(255),
    pregunta11 VARCHAR(255),
    pregunta12 VARCHAR(255),
    pregunta13 VARCHAR(255),
    pregunta14 VARCHAR(255),
    pregunta15 VARCHAR(255),
    pregunta16 VARCHAR(255),
    pregunta17 VARCHAR(255),
    pregunta18 VARCHAR(255),
    pregunta19 VARCHAR(255),
    pregunta20 VARCHAR(255),
    pregunta21 VARCHAR(255),
    fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Figura 33. Estructura de la tabla "puntuaciones" que guarda los valores del test

4.5 Ejecución completa de un test desde el lado del desarrollador en local

En primer lugar, se lanza *app.js*. Así, se establece la conexión con la BBDD y se inicia el servidor como se muestra en la Figura 34.

```
C:\Users\laram\Desktop\Todo\Universidad\TFG\parte4>node app.js
Servidor iniciado en el puerto 3000
Conexión a la base de datos establecida
```

Figura 34. Lanzamiento de app.js

Se carga *localhost:3000* en el navegador, y se muestra la primera página, *index.html.* Se puede ver la ejecución de la página web y *app.js* en paralelo en las Figuras 35, 36, 37, 38 y 39 en distintos momentos de la realización del test.

Figura 35. Ejecución de la web con el test y app.js en index.html

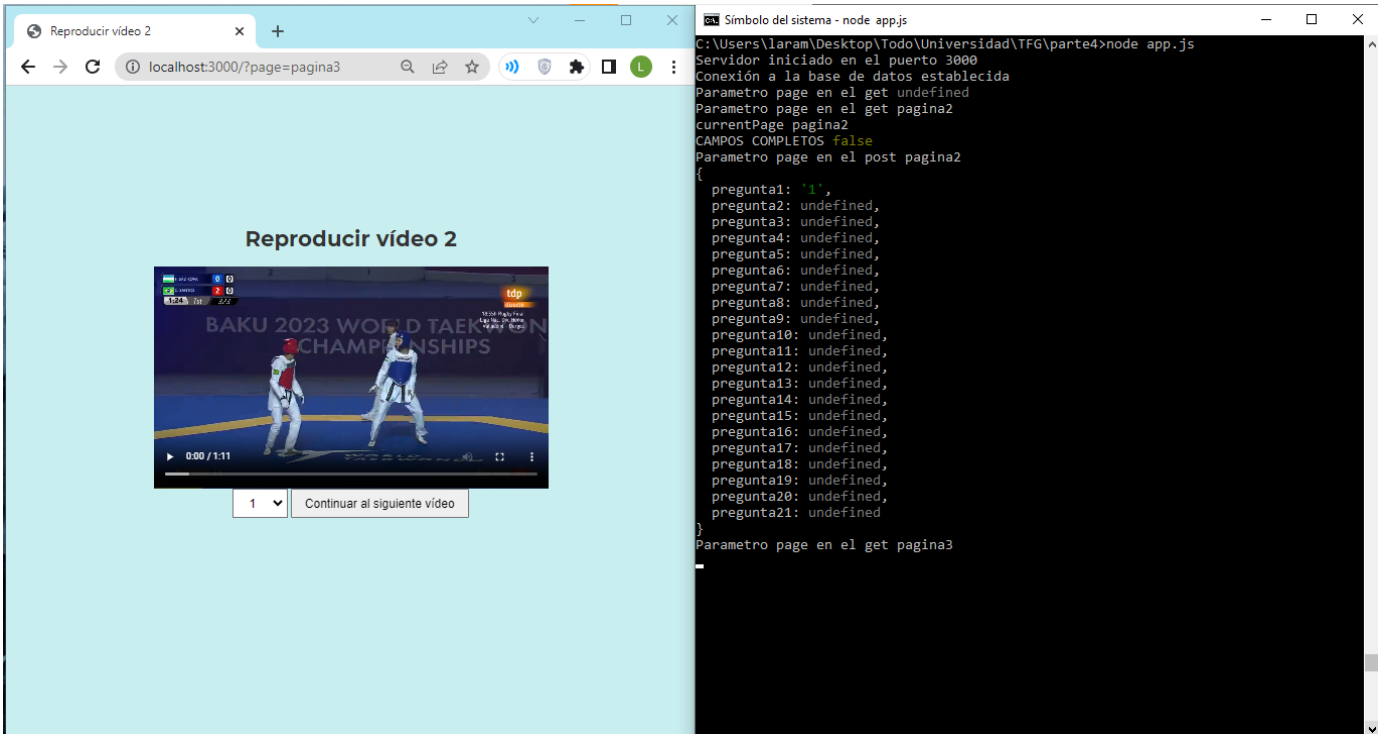


Figura 36. Ejecución de la web con el test y app.js en pagina3.html

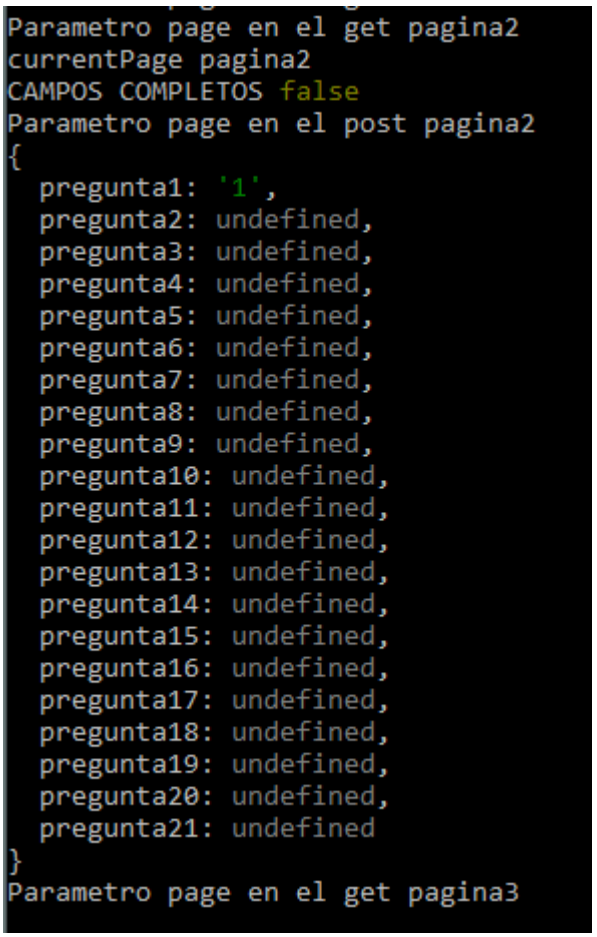


Figura 37. Zoom de la Figura 36

Cada vez que se pulsa el botón de *Continuar al siguiente vídeo,* se imprimen por consola los siguientes valores:

- *Parámetro page en el post:* Es el valor de la solicitud HTTP POST
- *CAMPOS COMPLETOS:* Indica si se han llenado los campos de todos los formularios en el test
- *Valores almacenados en storedValues*
- *Parámetro page en el get:* Es el valor de la solicitud HTTP GET

En la Figura 37 se pueden observar dichos valores cuando se carga *pagina3.html.* El parámetro page en el post es *pagina2*, puesto que al pasar a la página 3 se guarda en *storedValues* la puntuación del primer vídeo (alojado en *pagina2.html*). *CAMPOS COMPLETOS* es *false* porque solo se ha llenado un campo. El único valor almacenado en *storedValues* es *pregunta 1: '1'*, porque solo se ha evaluado un vídeo. El parámetro page en el get es *pagina3*, porque es la última solicitud HTTP GET que se ha hecho para estar mostrando *pagina3.html*.

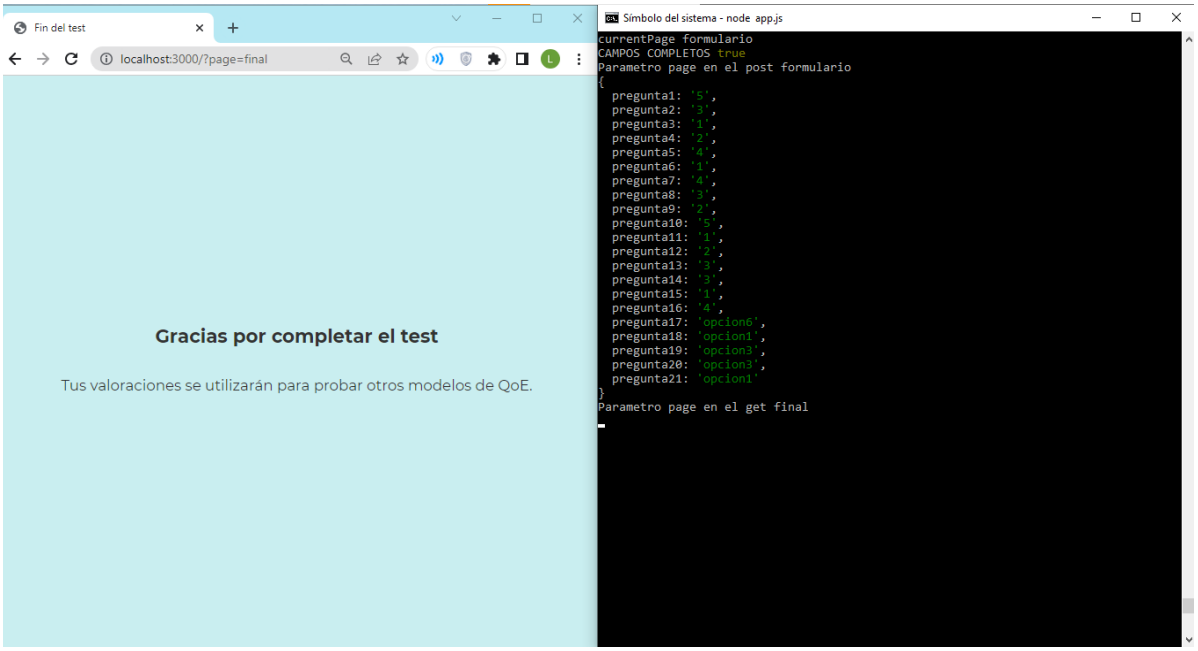


Figura 38. Ejecución de la web con el test y app.js en final.html

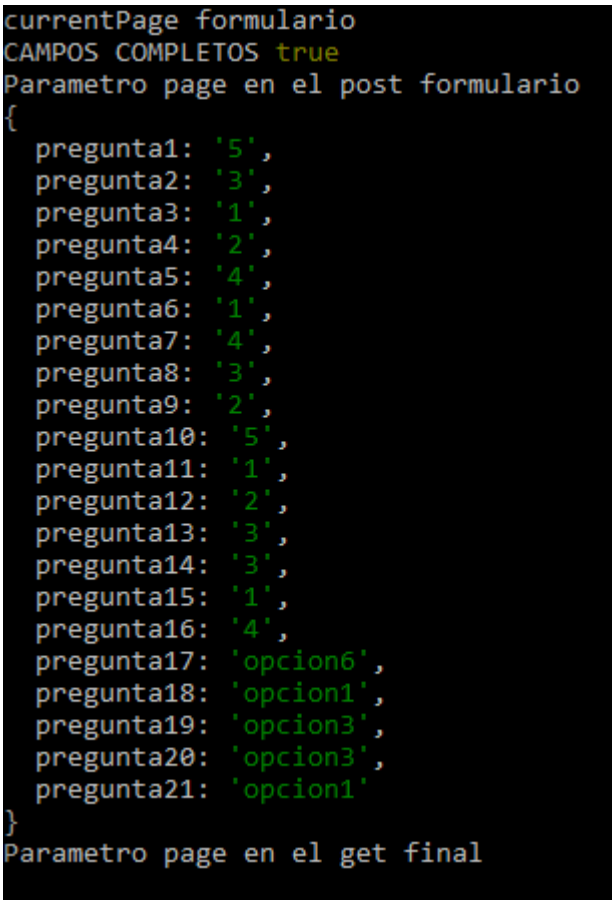


Figura 39. Zoom de la Figura 38

En la Figura 39 se pueden observar los valores impresos por consola cuando se carga *final.html.* El parámetro page en el post es *formulario*, puesto que al pasar a la página final se guardan en *storedValues* los valores del cuestionario (alojado en *formulario.html*). *CAMPOS COMPLETOS* es true porque ahora sí, se han completado todos los campos. En *storedValues* hay almacenado un valor para cada campo. El parámetro page en el get es *final*, porque es la última solicitud HTTP GET que se ha hecho para estar mostrando *final.html.*

Después de este último paso, al haber pulsado el botón *Terminar* en *formulario.html*, debería haberse almacenado una nueva fila en la tabla *puntuaciones.* Al hacer una consulta en SQL como la siguiente:

*SELECT * FROM mydatabase.puntuaciones;*

se puede comprobar en la Figura 40 que una nueva fila con todos los campos se ha generado en la base de datos *mydatabase.*

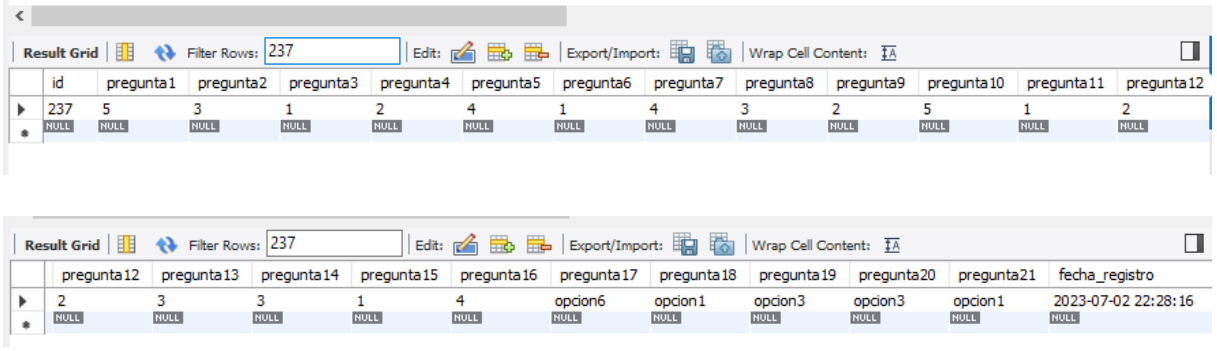


Figura 40. Nueva fila generada en la tabla puntuaciones de mydatabase tras la ejecución completa de un test en local

4.6 Despliegue de la web usando un dominio de la UPV

Con el objetivo de que la página web creada esté disponible en remoto y los participantes en el experimento de visionado puedan completar el test en sus distintos dispositivos y desde lugares diferentes, se ha desplegado la web en un ordenador del iTeam y se ha publicado usando el dominio http://carxofa.iteam.upv.es/

Para poder hacerlo, previamente se ha definido el experimento como una aplicación multi-contenedor de dos servicios: el servidor web y la base de datos. Se han creado un fichero *Dockerfile* y un fichero *docker-compose.yml* para definir la configuración de la aplicación y administrar los contenedores asociados a la web y la BBDD. También se crea un fichero *setup.sql* para crear la base de datos en remoto y una tabla *puntuaciones* con la misma estructura que la creada en local en el apartado 4.4.

Desde el *Dockerfile* se define la imagen de Node JS a utilizar, se instalan las dependencias, se indica el puerto en el que se ejecuta el servidor y se ejecuta la aplicación Node JS, *app.js*. Desde el *docker-compose.yml* se definen los servicios de la app y la base de datos, cada uno con la imagen que utilizan, los puertos, los volúmenes y las variables de entorno correspondientes. Se define el servicio *app* como dependiente del servicio de base de datos *db*, para conectarlos entre sí.

Una vez configurados los contenedores, se puede desplegar la web con la funcionalidad de guardado de los resultados en la BBDD.

4.7 Visualización de resultados almacenados en la BBDD

Una vez todos los participantes han completado el test, se descargan los resultados pasando el parámetro *guardarCSV* a la dirección web. Según se indica en el código de *app.js*, si se pasa dicho parámetro se hace una llamada a la función *guardarCSV*, que parsea los datos y se los descarga de forma automática a un fichero CSV del que se extraerán para posteriormente analizarlos en la parte V.

En la Figura 41, se muestra la respuesta en la terminal de comandos del paso del parámetro *guardarCSV* una vez ejecutado el comando *docker-compose up:* "*Guardando en CSV*". También se observa la ejecución simultánea de los contenedores *app_1* y *db_1.*

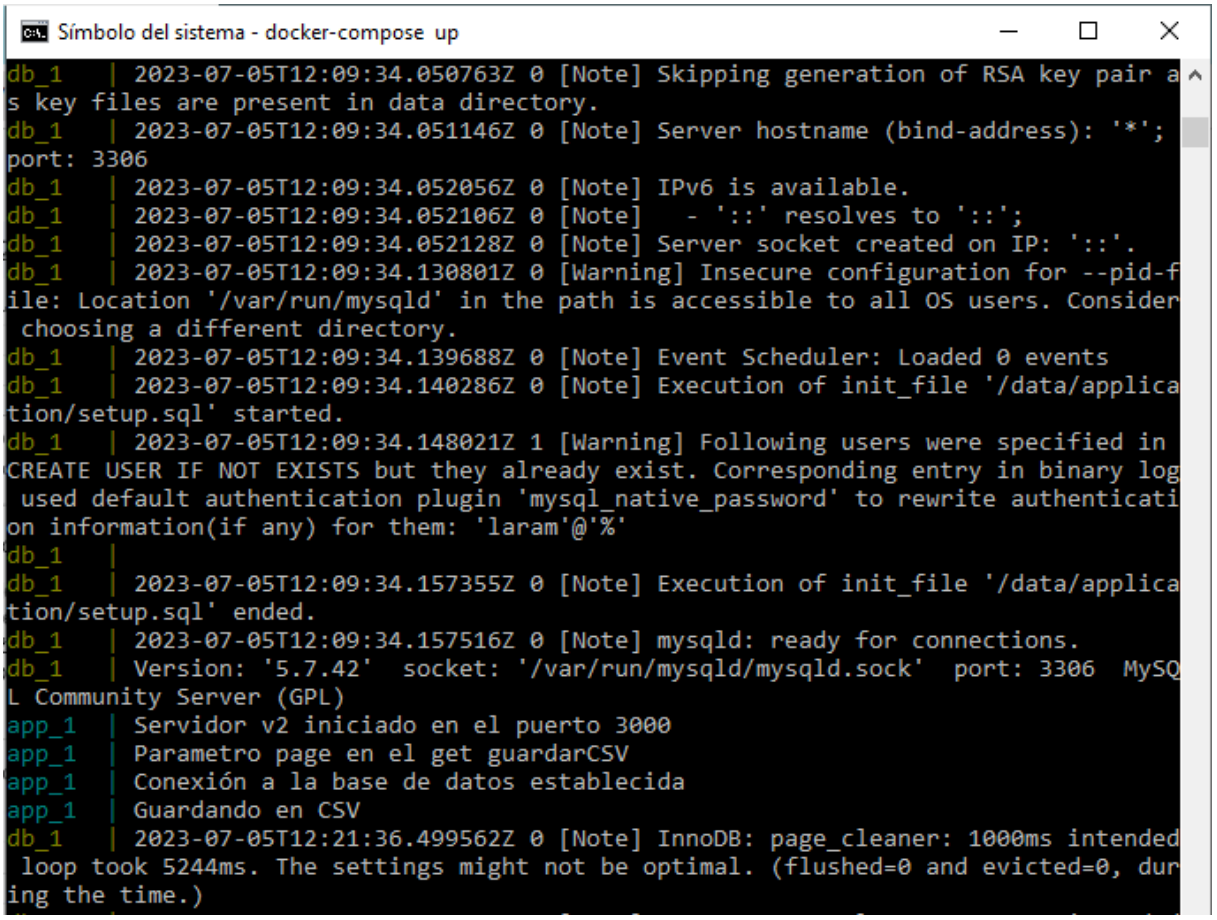


Figura 41. Ejecución simultánea de los contenedores app_1 y db_1 con la respuesta a la inserción del parámetro guardarCSV

PARTE V. ANÁLISIS DE RESULTADOS

5.1 Presentación de los resultados

Una vez probados ambos modelos, el híbrido ya implementado previamente en Python y el subjetivo implementado a través de la creación de una página web y una base de datos, se presentan los resultados obtenidos para cada uno de ellos.

Los vídeos a los que se hace referencia en las tablas son:

- Vídeo 1: animacion.mp4
- Vídeo 2: noticias.mp4
- Vídeo 3: deportes.mp4
- Vídeo 4: serie.mp4

Para el modelo híbrido, creado a partir de la recomendación ITU-T P.1203, los resultados obtenidos son los siguientes, representados en la Tabla 4.

	Modo 0	Modo 1	Modo 2	Modo 3
Vídeo 1	4.156	4.590	4.520	4.238
Vídeo 1 - Escenario 1	4.515	3.579	3.503	2.857
Vídeo 1 - Escenario 2	4.519	3.788	3.580	3.472
Vídeo 1 - Escenario 3	4.526	3.928	2.988	3.469
Vídeo 2	3.801	4.251	4.340	4.730
Vídeo 2 - Escenario 1	3.477	3.014	3.125	3.200
Vídeo 2 - Escenario 2	3.156	3.194	3.255	3.210
Vídeo 2 - Escenario 3	4.292	4.286	4.031	4.172
Vídeo 3	4.859	3.951	4.413	4.500
Vídeo 3 - Escenario 1	3.228	2.562	3.012	2.214
Vídeo 3 - Escenario 2	3.217	2.584	2.164	2.245
Vídeo 3 - Escenario 3	4.227	3.742	3.718	3.213
Vídeo 4	3.974	4.026	4.581	4.703
Vídeo 4 - Escenario 1	4.177	3.679	2.843	3.252
Vídeo 4 - Escenario 2	4.169	3.610	3.188	3.237
Vídeo 4 - Escenario 3	4.176	3.689	2.762	2.834

Tabla 4. Resultados ITU-T P.1203 para los 16 vídeos origen.

Para el modelo subjetivo, creado a través del experimento de visionado, los resultados obtenidos están representados en la Tabla 14 (situada en el Anexo II) en el orden en el que se recogen en el propio test. Este orden se muestra previamente en la Tabla 13 (también en el Anexo II) y será un parámetro más a tener en cuenta en la parte VI. V1 se corresponde con el Vídeo 1 y E1 con el Escenario 1, y así sucesivamente.

En la Tabla 5 se presentan los resultados ordenados por vídeos. T1 se corresponde con la primera pregunta sobre las condiciones de visionado y O1 con la primera opción disponible entre las respuestas a la pregunta, y así sucesivamente.

Nº Test	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
V1	1	5	5	4	5	4	5	5	5	5	5	4	5	5	5	1	5	5	4	4
V1-E1	1	2	3	3	3	3	3	2	3	2	3	3	2	3	2	3	3	4	4	2
V1-E2	1	3	3	4	2	4	4	2	2	1	4	4	3	3	2	2	2	2	3	3
V1-E3	1	3	4	3	4	4	3	3	4	1	5	3	4	3	3	5	5	4	3	3
V2	2	5	5	4	5	5	5	5	4	5	5	4	5	4	5	5	5	5	5	4
V2-E1	4	3	2	4	3	2	4	3	3	3	3	1	3	3	3	3	3	3	2	2
V2-E2	3	1	2	3	2	3	2	2	3	1	2	1	2	2	3	4	3	3	2	1
V2-E3	4	4	3	4	4	3	4	3	3	3	3	2	3	3	3	4	3	4	2	4
V3	5	5	5	2	5	5	5	5	5	5	5	3	5	5	5	5	5	5	5	3
V3-E1	3	2	1	2	2	2	2	1	2	1	2	1	1	3	2	2	2	2	1	3
V3-E2	3	3	1	2	2	3	2	2	2	2	3	1	3	3	3	2	2	3	1	2
V3-E3	4	3	3	3	3	3	4	3	3	2	4	2	4	3	1	3	3	2	1	3
V4	5	5	5	4	5	4	5	5	5	5	5	3	5	5	5	5	5	5	5	5
V4-E1	3	4	3	2	4	3	4	3	4	4	4	2	4	4	2	3	3	3	2	1
V4-E2	4	3	3	2	3	2	2	2	3	3	3	1	3	4	2	3	2	2	2	3
V4-E3	4	4	4	2	4	4	4	3	3	3	3	2	3	3	3	4	4	2	2	1
T1	O3	O6	O7	O6	O6	O3	O3	O3	O6	O3	O3	O6	O3	O3	O2	O5	O3	O3	O6	O6

T2	O1	O1	O1	O1	O1	O2	O1	O1	O2	O1	O2	O3	O2	O1	O1	O1	O2	O2	O1	O1
T3	O4	O1	O1	O3	O1	O2	O3	O1	O3	O3	O3	O1	O1	O3	O2	O3	O2	O3	O2	O4
T4	O2	O1	O2	O2	O2	O2	O2	O2	O2	O2	O1	O2	O1	O3	O2	O2	O2	O1	O2	O1
T5	O3	O3	O3	O1	O1	O1	O1	O1	O3	O2	O3	O1	O1	O3	O1	O3	O1	O3	O1	O1

Tabla 5. Resultados del test subjetivo ordenados

Se presentan de nuevo las cuestiones sobre las condiciones de visionado, ya mostradas en las Figuras 28 y 29 del apartado 4.2.4.

- T1: Condiciones lumínicas de la habitación
    - O1: Completa oscuridad
    - O2: Luz natural - Poca luz
    - O3: Luz natural - Bastante luz
    - O4: Luz natural - Mucha luz
    - O5: Luz artificial - Poca luz
    - O6: Luz artificial - Bastante luz
    - O7:Luz artificial - Mucha luz
- T2: Dispositivo en el que se ha reproducido el contenido
    - O1: Teléfono móvil
    - O2: Ordenador/Tablet
    - O3: Monitor/Televisión
- T3: Nivel de atención durante la realización del test
    - O1: He mantenido la concentración durante todo el test
    - O2: Diría que he prestado más atención durante los primeros minutos que durante los últimos
    - O3: He perdido atención brevemente en distintos momentos durante el test
    - O4: Me ha costado mantener la atención durante todo el test
- T4: Nivel de energía
    - O1: Me siento muy enérgico/a
    - O2: Estoy algo cansado/a
    - O3: Estoy muy cansado/a
- T5: ¿Ha pausado, avanzado o retrocedido en alguno de los vídeos?
    - O1: Sí
    - O2: Solo pausado
    - O3: No

5.2 Cálculo de parámetros asociados a los resultados

Como se explica en la propia especificación de la ITU [14] y ya se ha puntualizado en el apartado 3.1, cada uno de los modos del modelo P.1203.1 predice con mayor exactitud la calidad de vídeo de los archivos multimedia que se introducen como input. Por ello, se emplearán los datos obtenidos del modo 3 como resultado del modelo híbrido, que se muestran en la Tabla 6.

MODELO HÍBRIDO	Modo 3		Modo 3
Vídeo 1	4.238	Vídeo 3	4.500
Vídeo 1 - Escenario 1	2.857	Vídeo 3 - Escenario 1	2.214
Vídeo 1 - Escenario 2	3.472	Vídeo 3 - Escenario 2	2.245
Vídeo 1 - Escenario 3	3.469	Vídeo 3 - Escenario 3	3.213
Vídeo 2	4.730	Vídeo 4	4.703
Vídeo 2 - Escenario 1	3.200	Vídeo 4 - Escenario 1	3.252
Vídeo 2 - Escenario 2	3.210	Vídeo 4 - Escenario 2	3.237
Vídeo 2 - Escenario 3	4.172	Vídeo 4 - Escenario 3	2.834

Tabla 6. Datos a utilizar del modelo híbrido.
Resultados del modo 3 del modelo P1203.1.

Por otro lado, en la Tabla 7 se muestran los valores de la media, la varianza y la desviación estándar de las puntuaciones que han dado los participantes en el test subjetivo a cada uno de los 16 vídeos. Serán los datos a considerar como resultado del modelo subjetivo.

MODELO SUBJETIVO	Media	Mediana	Desviación estándar
Vídeo 1	4.35	5	1.23
Vídeo 1 - Escenario 1	2.7	3	0.73
Vídeo 1 - Escenario 2	2.7	3	0.98
Vídeo 1 - Escenario 3	3.4	3	1.10
Vídeo 2	4.6	5	0.75
Vídeo 2 - Escenario 1	2.85	3	0,75
Vídeo 2 - Escenario 2	2.25	2	0,85
Vídeo 2 - Escenario 3	3.3	3	0,66
Vídeo 3	4.65	5	0,88
Vídeo 3 - Escenario 1	1.85	2	0,67
Vídeo 3 - Escenario 2	2.25	2	0,72
Vídeo 3 - Escenario 3	2.85	3	0,88
Vídeo 4	4.8	5	0,52

Vídeo 4 - Escenario 1	3.1	3	0,91
Vídeo 4 - Escenario 2	2.6	3	0,75
Vídeo 4 - Escenario 3	3.1	3	0,91

Tabla 7. Datos a utilizar del modelo objetivo.
Media, mediana y desviación estándar de las puntuaciones de los usuarios

5.3 Análisis de resultados del modelo híbrido

Según los resultados proporcionados por la implementación de la recomendación P.1203 de la ITU-T, se puede observar cómo los datos de los modos de orden mayor son más distintos entre sí. Una de las principales desventajas de los modelos de evaluación de QoE objetivos es precisamente, como se puntualizó en el apartado 1.2.4.1, que proporcionan puntuaciones bastante similares para vídeos de calidad distinta. Este modelo híbrido comparte dicha desventaja, pero se va corrigiendo a medida que se prueban modos de orden mayor. Como se indica en el propio repositorio [19a], cuanto mayor sea el modo, más precisa será la predicción.

Observando por tanto los resultados proporcionados por el modo 3, los valores están comprendidos entre 2.214 y 4.730. Sólo un 25% de las puntuaciones del modelo híbrido están por debajo de 3, lo que indica que la mayoría de los vídeos tienen una calidad aceptable o mayor según la escala MOS para los 3 escenarios de ancho de banda diseñados.

Teniendo en cuenta las ventajas del modelo, que es rápido y fácil de implementar y que implica muy bajo coste, la valoración del mismo es favorable.

5.4 Análisis de resultados del modelo subjetivo

Para evaluar cómo de concluyentes son los resultados obtenidos, se han calculado algunos parámetros asociados a los mismos. Los valores de la media están comprendidos entre 2.25 y 4.8, unos valores máximo y mínimo muy similares a los obtenidos para el modelo híbrido. Sin embargo, en este caso, un 43% de las valoraciones están por debajo de la calidad aceptable según la escala MOS. Los valores de la mediana indican, sin embargo, que solo un 19% de los vídeos están por debajo de dicha calidad, así que es interesante por tanto el análisis de la desviación típica de las puntuaciones de los usuarios para cada uno de los vídeos presentados.

Los valores de desviación típica varían entre 0.66 y 1.23. Aunque el valor máximo no es excesivo como para que los resultados no sean representativos, indica que hay valores atípicos que alteran los resultados. Por ejemplo, y como se puede observar en la Tabla 5, el vídeo 1, que es uno de los 4 vídeos originales, ha recibido 18 puntuaciones mayores o iguales a 4 y 2 puntuaciones de valor 1, a pesar de no haber sufrido ninguna alteración.

Teniendo en cuenta las ventajas del modelo, que refleja fielmente la experiencia de los usuarios, y a pesar de las limitaciones comentadas previamente, la valoración del mismo es también favorable.

5.5 Análisis global

En términos generales, el modelo subjetivo ha resultado ser más exigente que el híbrido, puesto que las puntuaciones obtenidas para los mismos vídeos son más bajas. En la Tabla 8 se hace una comparativa entre los valores obtenidos en el modo 3 del modelo híbrido y las medias de los valores obtenidos en el modelo subjetivo, y se calcula la diferencia entre el primer valor y el segundo para cada vídeo. El mayor valor de diferencia obtenido entre ambos modelos es de 0.960, que no es excesivamente alto ni se ha obtenido para los vídeos en los que se espera un mayor consenso, es decir, en los originales.

	Modo 3	Test	Diferencia
Vídeo 1	4.238	4.35	-0.112
Vídeo 1 - Escenario 1	2.857	2.7	0.157
Vídeo 1 - Escenario 2	3.472	2.7	0.772
Vídeo 1 - Escenario 3	3.469	3.4	0.069
Vídeo 2	4.73	4.6	0.130
Vídeo 2 - Escenario 1	3.2	2.85	0.350
Vídeo 2 - Escenario 2	3.21	2.25	0.960
Vídeo 2 - Escenario 3	4.172	3.3	0.872
Vídeo 3	4.5	4.65	-0.150
Vídeo 3 - Escenario 1	2.214	1.85	0.364
Vídeo 3 - Escenario 2	2.245	2.25	-0.005
Vídeo 3 - Escenario 3	3.213	2.85	0.363
Vídeo 4	4.703	4.8	-0.097
Vídeo 4 - Escenario 1	3.252	3.1	0.152
Vídeo 4 - Escenario 2	3.237	2.6	0.637
Vídeo 4 - Escenario 3	2.834	3.1	-0.266

Tabla 8. Diferencia de resultados entre modelos.

Para los 4 vídeos originales se han obtenido puntuaciones superiores a 4 en ambos modelos, todas ellas superiores a 4.5 excepto las otorgadas al vídeo 1.

Para visualizar la correlación entre ambos modelos de forma más clara, se han representado los resultados de cada uno en la Figura 42 acompañados de una línea de tendencia, que tiene un valor de R^2 de 0.834.

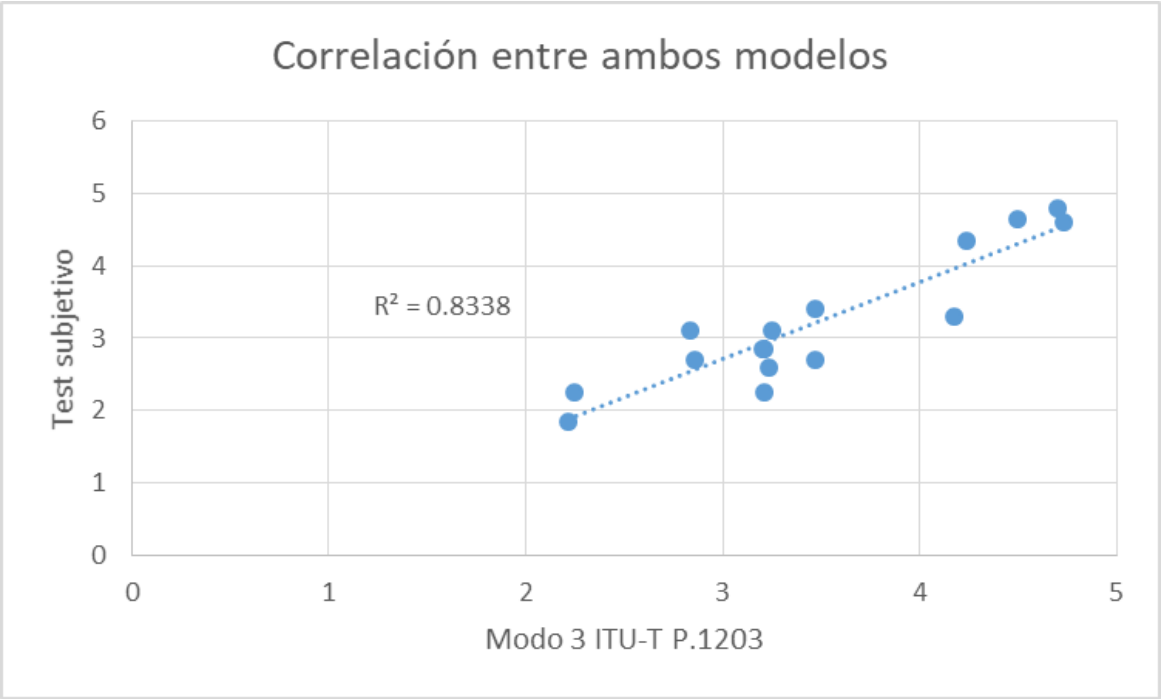


Figura 42. Correlación entre los resultados de ambos modelos.

5.5.1 Análisis por escenario

Según los escenarios de red definidos en el apartado 2.7.2, se puede hacer un análisis de cada modelo.

- *Escenario 1:* El ancho de banda aumenta de forma paulatina cada 5 segundos desde el valor mínimo hasta el valor máximo. Para el modelo híbrido el valor medio obtenido es de 2.88; y para el modelo subjetivo es de 2.63. Los resultados de ambos modelos son muy similares. El valor está entre calidad mala y aceptable en una escala MOS.
- *Escenario 2:* El ancho de banda va cambiando entre su valor máximo y mínimo de forma drástica cada 5 segundos. Para el modelo híbrido el valor medio obtenido es de 3.04; y para el modelo subjetivo es de 2.70. Los resultados de ambos modelos son similares, aunque para el modelo subjetivo el valor ya es aceptable en una escala MOS.
- *Escenario 3:* El ancho de banda se reduce de forma paulatina cada 5 segundos desde el valor máximo hasta el valor mínimo, y en los últimos instantes vuelve a incrementarse de forma drástica hasta el valor máximo. Para el modelo híbrido el valor medio obtenido es de 3.42; y para el modelo subjetivo es de 3.16. Los resultados de ambos modelos son similares, aunque para el modelo híbrido el valor está más cerca de ser aceptable y para el subjetivo ya está entre aceptable y buena en una escala MOS.

En general, los mejores valores de calidad se obtienen para el tercer escenario, seguido del segundo y, en último lugar, del primero.

Los valores y las conclusiones obtenidas en el presente apartado se comentarán en el apartado 5.5.3, puesto que también tienen relación directa con el efecto de los factores de influencia.

5.5.2 Análisis por vídeo

El análisis SI/TI del apartado 2.1.1 ha permitido conocer los datos sobre la complejidad espacial y temporal de cada vídeo.

- *Vídeo 1:* El vídeo *animacion.mp4* es el que presenta alta complejidad temporal y baja complejidad espacial. Para el modelo híbrido el valor medio obtenido es de 3.27; y para el modelo subjetivo es de 2.93. Los resultados de ambos modelos son similares, aunque el proporcionado por el modelo subjetivo es ligeramente más alto.

- *Vídeo 2:* El vídeo noticias.mp4 es el que presenta baja complejidad temporal y alta complejidad espacial. Para el modelo híbrido el valor medio obtenido es de 3.53; y para el modelo subjetivo es de 2.80. El resultado del modelo híbrido es bastante superior al del modelo subjetivo.

- *Vídeo 3:* El vídeo deportes.mp4 es el que presenta alta complejidad temporal y espacial. Para el modelo híbrido el valor medio obtenido es de 2.56; y para el modelo subjetivo es de 2.32. Los resultados de ambos modelos son similares indican que la calidad percibida es baja según la escala MOS.

- *Vídeo 4:* El vídeo serie.mp4 es el que presenta baja complejidad temporal y espacial. Para el modelo híbrido el valor medio obtenido es de 3.11; y para el modelo subjetivo es de 2.93. Los resultados de ambos modelos son similares e indican que la calidad es aceptable según la escala MOS.

En general, los mejores valores de calidad haciendo la media entre ambos modelos se obtienen para el vídeo 2, seguido del vídeo 1, del vídeo 4 y, en último lugar, del vídeo 3.

Los valores y las conclusiones obtenidas en este apartado también se comentarán en el apartado 5.5.3.

5.5.3 Efecto de los factores de influencia

De los factores de influencia presentados en el apartado 1.2.5, han entrado a evaluarse exclusivamente aquellos que tienen un impacto mayor sobre las tecnologías HAS (los presentados en el 1.2.5.1), que pertenecen mayoritariamente a las categorías de factores humanos y del contexto. Según [2], los fenómenos de rebuffering, cambio de calidad o los relacionados con la codificación son los más influyentes en la tecnología del streaming adaptativo. En particular, se han valorado los que se exponen a continuación:

1. *Frecuencia y magnitud de los cambios de calidad*: Teniendo en cuenta los resultados proporcionados por ambos modelos, se puede concluir que los cambios constantes en la calidad han repercutido negativamente en la QoE de los usuarios, como es lógico. No se entra a analizar la frecuencia de los cambios de calidad porque en los 3 escenarios los cambios se han hecho con la misma frecuencia, cada 5 segundos. El escenario con cambios de calidad de mayor magnitud es el segundo, que no es el que da ni los mejores ni los peores resultados en lo que se refiere a QoE. De acuerdo con lo expuesto en el apartado 1.3.1.2, los participantes de los test subjetivos tienden a reaccionar en tiempos distintos a los cambios repentinos de calidad [2], lo que podría explicar por qué no hay unos resultados decantados para ninguno de los extremos para el modelo subjetivo del escenario 2. En definitiva, la magnitud de los cambios de calidad no parece un factor de influencia clave en la calidad de experiencia para los vídeos analizados.

2. *Eventos de rebuffering*: En la Tabla 9 se presenta el número de interrupciones generadas al aplicar cada escenario a cada vídeo. Los eventos de retardo inicial durante la carga no están incluidos como interrupciones. El escenario 2 presenta en total 6 interrupciones, seguido el escenario 1 con 2 interrupciones, y del escenario 3 sin interrupciones. Como se ha mencionado en el apartado 5.5.1, el escenario 3 es el que ha recibido valores de calidad más altos para ambos modelos y es también el único escenario que no presenta interrupciones, lo que indica una relación directa entre la existencia de interrupciones y una QoE más baja. De los otros dos escenarios, aunque el escenario 2 ha recibido mejores puntuaciones y presenta más interrupciones que el escenario 1, cabe destacar 2 datos:

    a. Los vídeos del escenario 2 que presentan más de una interrupción son los dos que presentan también una alta complejidad espacial (información SI alta), lo que también puede alterar la percepción de calidad del espectador.

    b. La duración de las interrupciones de los vídeos a los que se ha aplicado el escenario 1 es mayor que la de las interrupciones de aquellos a los que se ha aplicado el escenario 2, aunque sean menos numerosas.

La presencia y la duración de las interrupciones se presenta como factor de influencia a tener en cuenta en los modelos de QoE.

	Nº interrupciones	Duración total (s)
Vídeo 1 - Escenario 1	1	5.6
Vídeo 2 - Escenario 1	0	0
Vídeo 3 - Escenario 1	1	5.0
Vídeo 4 - Escenario 1	0	0
Vídeo 1 - Escenario 2	1	3.1
Vídeo 2 - Escenario 2	2	4.2
Vídeo 3 - Escenario 2	2	4.4
Vídeo 4 - Escenario 2	0	0
Vídeo 1 - Escenario 3	0	0
Vídeo 2 - Escenario 3	0	0
Vídeo 3 - Escenario 3	0	0
Vídeo 4 - Escenario 3	0	0

Tabla 9. Número de interrupciones generadas en cada vídeo y duración de las mismas.

3. *Retardo inicial durante la carga*: Todos los vídeos presentan cierto retardo inicial durante la carga de los vídeos. En la Tabla 10 se presenta la duración de cada uno de ellos. La suma de las duraciones de los retardos presentados para todos los vídeos es muy similar para los 3 escenarios. Como se comentó en el apartado 1.2.5.1, retrasos de hasta 10 segundos en el inicio de la reproducción tienen un efecto despreciable en la QoE. A nivel individual, solo dos vídeos superan los 10 segundos de retardo inicial; y ninguno de ellos se corresponde con el

escenario 3, que es el que ha obtenido unos valores de QoE mayores para ambos modelos. Tiene sentido por tanto pensar que el retardo inicial es uno de los IF claves para la evaluación de la QoE en este caso.

	Duración retardo (s)
Vídeo 1 - Escenario 1	9.9
Vídeo 2 - Escenario 1	11.8
Vídeo 3 - Escenario 1	6.1
Vídeo 4 - Escenario 1	8.0
Vídeo 1 - Escenario 2	10
Vídeo 2 - Escenario 2	13.6
Vídeo 3 - Escenario 2	10.2
Vídeo 4 - Escenario 2	10.1
Vídeo 1 - Escenario 3	3.2
Vídeo 2 - Escenario 3	7.3
Vídeo 3 - Escenario 3	10.2
Vídeo 4 - Escenario 3	9.3

Tabla 10. Duración de los retardos iniciales de la carga de vídeos.

4. *Fenómenos de primacy y recency*: Estos parámetros son puramente subjetivos, así que solo se consideran los resultados del modelo subjetivo. Respecto a los fenómenos de *primacy* y *recency*, cabe evaluarlos de dos formas distintas:

    a. *En cada vídeo*: Incluso considerando exclusivamente los resultados del modelo subjetivo, los mejores resultados de QoE siguen siendo para el escenario 3, bastante alejados del 2 y del 1, en ese orden. Según se han definido los escenarios, los vídeos a los que se ha aplicado el escenario 1 tienen mayor ancho de banda disponible al principio de la reproducción, y aquellos a los que se ha aplicado el escenario 2 presentan mejor ancho de banda disponible al final de la reproducción. El escenario 3 presenta un ancho de banda máximo tanto al principio como al final de la reproducción, durante 10 y 15 segundos respectivamente.

    Siguiendo este razonamiento, parece darse un efecto de *primacy*, puesto que los dos escenarios con mayor ancho de banda al inicio de la reproducción son los mejor valorados en los test subjetivos. De la misma forma, se puede intuir un efecto de *recency* porque, aunque el escenario 2 muestra una gran bajada de ancho de banda en los últimos segundos, se puede comprobar visualmente que la información almacenada en el buffer en ese momento es suficiente como para que la degradación de la calidad no sea excesivamente perceptible antes de la finalización del vídeo. Que exista un efecto de *recency* es especialmente

probable teniendo en cuenta que los vídeos son largos, como ya se mencionaba en el apartado 1.2.5.

b. *En cada test:* Observando las puntuaciones de cada participante del test, toma importancia el orden de visualización de los vídeos en conjunto para analizar si existe un efecto general de *primacy* o de *recency.* Según las Tablas 13 y 14 (ambas en el Anexo II), y con ayuda de la Tabla 11, se puede observar que la media de las puntuaciones de los 3 primeros vídeos es notablemente inferior a la de los 3 últimos: 2.45 frente a 3.58. Sin embargo, hay que tener en cuenta la información disponible sobre los vídeos concretos a los que corresponden.

   En lo que se refiere a los vídeos iniciales, el Vídeo 2 - Escenario 2 es el vídeo con más retardo inicial de los 16, y también es uno de los que más interrupciones presenta, junto con el Vídeo 3 - Escenario 2. El Vídeo 3 - Escenario 1 presenta una de las interrupciones más largas de todos los vídeos.

   Respecto a los últimos, el Vídeo 1 es un original, así que no es justo comparar su puntuación con los demás; y ninguno de los otros dos vídeos presenta interrupciones ni los retardos iniciales más largos.

   No es posible concluir por tanto si existe un efecto de *recency* para la visualización de todo el test.

	Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 14	Pregunta 15	Pregunta 16
Correspondencia	V2 - E2	V3 - E1	V3 - E2	V4 - E1	V1	V4 - E3
Media puntuaciones	2.25	1.85	2.25	3.1	4.55	3.1

Tabla 11. Media de las puntuaciones para los primeros y últimos vídeos del test.

5. *Complejidad temporal (TI):* Como se menciona en el apartado 1.3.1.2, en los vídeos de alta complejidad temporal los cambios de calidad tienden a ser percibidos con mayor facilidad. Como esta es una cuestión subjetiva, solo se toman como referencia los resultados del modelo subjetivo.

   Según lo analizado en el apartado 5.5.2, el vídeo 3 es con mucha diferencia el peor valorado. Sin embargo, el vídeo 1 es el mejor valorado junto al vídeo 4.. Según lo analizado en el apartado 2.1.1, los dos vídeos con mayor complejidad temporal son el vídeo 3 y el vídeo 1. Por tanto, no parece haber una relación directa entre la complejidad temporal y la QoE percibida en este caso.

En la Tabla 5 se pueden ver las opciones seleccionadas por los usuarios para las preguntas del formulario final, etiquetadas de T1 a T5. En la Tabla 12 se presenta la misma información, pero acompañada de la puntuación media de cada espectador para todos los vídeos en conjunto con el objetivo de poder analizar la influencia de los factores de influencia considerados.

Media espectador	T1	T2	T3	T4	T5
3,000	O3	O1	O4	O2	O3
3,438	O6	O1	O1	O1	O3
3,250	O7	O1	O1	O2	O3
3,000	O6	O1	O3	O2	O1
3,500	O6	O1	O1	O2	O1
3,375	O3	O2	O2	O2	O1
3,625	O3	O1	O3	O2	O1
3,063	O3	O1	O1	O2	O1
3,375	O6	O2	O3	O2	O3
2,875	O3	O1	O3	O2	O2
3,688	O3	O2	O3	O1	O3
2,313	O6	O3	O1	O2	O1
3,438	O3	O2	O1	O1	O1
3,500	O3	O1	O3	O3	O3
3,063	O2	O1	O2	O2	O1
3,625	O5	O1	O3	O2	O3
3,438	O3	O2	O2	O2	O1
3,375	O3	O2	O3	O1	O3
2,750	O6	O1	O2	O2	O1
2,750	O6	O1	O4	O1	O1

Tabla 12. Puntuaciones medias de los vídeos y respuestas al formulario de cada espectador.

6. *Compromiso del usuario*: La variable T5 de la Tabla 12 hace referencia a la pregunta *¿Ha pausado, avanzado o retrocedido en alguno de los vídeos?.* Hay 8 noes con una media de 3.41, y 12 síes (menos uno que ha marcado la opción “solo pausado”) con una media de 3.1. La diferencia entre ambos resultados no parece indicar una fuerte relación entre la alteración de la reproducción de los vídeos y un descenso en la QoE percibida.

El resto de IFs valorados en esta lista no están incluidos en la lista del apartado 1.2.5.1, pero se preguntan también en el formulario del test para tenerlos en cuenta. Son los siguientes:

7. *Dispositivo:* La variable T2 de la Tabla 12 hace referencia a la pregunta referida al dispositivo en el que se ha reproducido el contenido. Hay 13 reproducciones desde teléfonos móviles con una media de 3.19, y 7 desde ordenadores o tablets (menos una de ellas desde un monitor o TV) con una media de 3.29. La diferencia entre ambos resultados es casi imperceptible, aunque como se sugirió en el apartado 1.1, medir este factor es importante porque puede suponer la clave para los fabricantes de dispositivos.

8. *Iluminación:* La variable T1 de la Tabla 12 hace referencia a la pregunta referida a las condiciones lumínicas de la habitación. Hay 11 personas que reprodujeron el contenido con luz natural con una puntuación media de 3.31, y 9 personas que lo hicieron con luz artificial (todas menos una con bastante o mucha luz) con una media de 2.75. En este caso, la diferencia entre los resultados parece sugerir una relación entre el uso de luz natural y una mejor percepción de la calidad del contenido.

9. *Energía:* La variable T4 de la Tabla 12 hace referencia a la pregunta referida al nivel de energía del espectador. Hay 5 respuestas de espectadores que se sentían muy enérgicos con una puntuación media de 3.34, y 15 respuestas de espectadores que se encontraban algo cansados (solo uno muy cansado) con una media de 3.18. La diferencia en los niveles de energía no parece haber afectado a su evaluación de la QoE de los vídeos.

10. *Concentración:* La variable T3 hace referencia al nivel de atención del espectador durante la realización del test. Hay 6 personas que afirmaron haber mantenido la atención durante todo el test con una puntuación media de 3.17, 10 personas que no mantuvieron la atención por completo en ningún momento o la perdieron en algunos momentos con una media de 3.28, y 4 prestaron más atención durante los primeros minutos que durante los últimos con una media de 3.16. Los 3 resultados son muy similares y no indican una relación directa entre el nivel de concentración de los espectadores y la QoE que experimentaron al visualizar los vídeos.

PARTE VI. CONCLUSIONES

6.1 Aplicación de la fundamentación teórica al trabajo práctico

Aunque se ha mostrado la relación entre el contenido teórico y el desarrollo práctico a lo largo del presente trabajo, en este apartado se expone de forma resumida cómo la parte I ha servido de base para construir el resto del contenido del documento.

Los tipos de métricas existentes para la evaluación de la calidad de un contenido de vídeo se habían expuesto en el apartado 1.2.4. La implementación de la recomendación ITU-T P.1203 es un modelo híbrido, aunando las ventajas de las métricas objetivas y subjetivas: coste bajo, facilidad de implementación, velocidad o información más acertada sobre la percepción real de los usuarios. El modelo subjetivo desarrollado en la parte IV del trabajo, sin embargo, usa una métrica puramente subjetiva. La información sobre la percepción de los usuarios es completa, pero su uso ha supuesto un coste más alto de tiempo y de recursos y su aplicabilidad es por ello mucho más pequeña que la del modelo híbrido, además de porque el número de participantes y la duración de los tests son limitados.

Según los criterios de clasificación presentados en el apartado 1.3.1, los modelos híbrido y subjetivo empleados para la evaluación de la QoE son No Reference (NR), porque no utilizan ninguna información del vídeo original. En ambos, la única entrada es el vídeo a evaluar, pero no se incluye el vídeo original a partir del que se ha generado el input ni ninguna referencia sobre él. Por otro lado, la recomendación ITU-T P.1203 es un modelo *parametric bitstream-based* [14], es decir, un modelo híbrido, combinación de un modelo paramétrico y de uno de tipo bitstream o flujo de bits. Por su parte, el modelo subjetivo desarrollado es de tipo signal-based, porque se basa únicamente en la evaluación de características derivadas directamente de la señal de vídeo. En concreto, se basa en la percepción de las distorsiones visuales por parte de los participantes.

De los factores de QoS que se pueden usar para estimar la QoE, especificados en el apartado 1.2.3.2, se ha decidido utilizar el ancho de banda como parámetro de referencia. Como se indicaba, se ha seguido un procedimiento específico para medirla (en este caso, dos, uno para cada modelo); y se han tenido en cuenta los factores que influyen en él (o bien el sistema ya los tenía en cuenta, como para el modelo híbrido, o bien se han obtenido, como en el modelo subjetivo). Por tanto, se han reunido los requisitos necesarios para considerar las técnicas empleadas como modelos de evaluación de QoE.

Como también se expuso previamente, para la elaboración del experimento de visionado se han tenido en cuenta las recomendaciones BT.500 y P.910 de la ITU. Las recomendaciones seguidas han sido: evaluación por escala numérica (del 1 al 5), pruebas de calidad degradada (mismo vídeo con mayor o menor degradación de la calidad para ver hasta dónde es aceptable) y pruebas en diferentes condiciones de visualización (iluminación, tipo de dispositivo, etc.) para ver cómo afectan a la calidad percibida.

6.2 Valoración general

El objetivo principal del presente trabajo ha sido desde un principio automatizar el proceso de reconstrucción de secuencias de vídeo que han sido sometidas a unas condiciones de red

concretas, para luego colocarlas embebidas en una web propia y crear un experimento de modelado subjetivo de la calidad de experiencia. Para completar la investigación, se ha decidido comparar los resultados de dicho modelo subjetivo con los de otro modelo ya existente, que ha sido la implementación en python de un modelo híbrido que sigue las recomendaciones de la ITU-T P.1203.

Durante la elaboración del trabajo y especialmente en la parte final de análisis, se han ido extrayendo conclusiones. Aunque se ha encontrado cierta correlación entre los resultados de ambos modelos, sí que es cierto que los resultados del experimento subjetivo no se ajustan por completo a los que proporciona la herramienta híbrida. Los valores atípicos perjudican considerablemente los resultados del modelo subjetivo, como se analiza en el apartado 5.4, provocando así diferencias con los resultados del modelo híbrido. Sus puntuaciones son ligeramente más bajas.

Los vídeos originales reciben puntuaciones superiores a 4 por parte de los dos modelos, pero ni siquiera el modelo híbrido ha dado un resultado superior a 4.7 para ninguno de ellos, lo que sugiere que la calidad original de los vídeos utilizados, que son de resolución 720x480, puede haber sido un problema para la obtención de resultados más satisfactorios. Si la resolución de los vídeos originales no era muy alta, es más probable que cueste distinguir con tanta claridad los cambios en la calidad de los mismos.

En cuanto al efecto de los factores de influencia, se ha concluido que aquellos que han tenido mayor impacto sobre los resultados obtenidos son: la presencia y la duración de los eventos de rebuffering, el retardo en el inicio de la reproducción, los fenómenos de *primacy* y *recency* en cada vídeo y la iluminación de la habitación. El resto de los factores de influencia analizados no parecen haber tenido un impacto notable en los resultados.

6.3 Limitaciones de la investigación y recomendaciones a futuro

Para la elaboración del test subjetivo se han considerado factores de influencia del sistema (cambios de calidad, de ancho de banda y de dispositivo), humanos (nivel de atención, energía), del contexto (condiciones lumínicas) y del contenido (vídeos de distinto tipo y complejidad espacial y temporal). Como se expone de forma detallada en el apartado 6.1, únicamente se han seguido únicamente las recomendaciones de la ITU que han sido posibles para la elaboración del test. Además, cabe tener en cuenta que los factores de influencia a medir, el número de participantes y la duración del test son limitados, y por tanto los resultados podrían ser más significativos si todos ellos se incrementaran.

Además, cuando en el modelado intervienen personas reales, influyen también las expectativas y preferencias personales del espectador [3]. De la misma forma, la existencia de múltiples factores de influencia complica aún más el diseño de los modelos.

Por otro lado, algunas recomendaciones para la creación de otros modelos de QoE en base a los resultados obtenidos son:

- Modelar en función de los IFs que han tenido mayor impacto en la elaboración del presente trabajo: rebuffering, retardo inicial, *primacy* y *recency* e iluminación.

- Estudiar los factores de influencia no solo a nivel individual, sino también la relación entre ellos para hacer predicciones más precisas. [24]
- Usar los 3 criterios de evaluación del rendimiento de los modelos de QoE expuestos en el apartado 1.2.6.1: precisión, monotonía y consistencia. Para ello habría que calcular el PLCC, el SROCC y el OR, respectivamente.
- Probar con otros tamaños de segmento para ver el impacto del cambio entre varios sobre la QoE. Esta información sería especialmente útil para los proveedores de servicio, como se puntualiza en el apartado 1.1.
- Usar un modelo alternativo a la implementación de la ITU-T P.1203 para comparar los resultados obtenidos con el modelo desarrollado, como la implementación en GitHub de la ITU-T P.1204.3. [25] Como ventaja, es más fácil y más rápida de aplicar. Aunque está diseñada para vídeos de hasta 10 segundos de duración, ya está publicado un nuevo apéndice *Long term integration module (Pq)* que aún no está implementado en GitHub [23]
- Diseñar un sistema híbrido que pueda almacenar datos de la red y del usuario de forma simultánea para modelar la relación entre las métricas de QoS y la QoE [24]
- Usar otros parámetros de QoS además del ancho de banda para generar vídeos con cambios de calidad e interrupciones, como cualquiera de los expuestos en el apartado 1.2.3.2: jitter, velocidad de transmisión, pérdida de paquetes o retardo.
- Emplear técnicas de Machine Learning para el desarrollo de modelos.

BIBLIOGRAFÍA

[1] Wang, Q., Dai, H., Wu, D., & Xiao, H. (2018). Data analysis on video streaming QoE over mobile networks. Eurasip Journal on Wireless Communications and Networking, 2018(1). https://doi.org/10.1186/s13638-018-1180-8

[2] Barman, N., & Martini, M. G. (2019). QoE Modeling for HTTP Adaptive Video Streaming–A Survey and Open Challenges. IEEE Access, 7, 30831-30859. https://doi.org/10.1109/access.2019.2901778

[3] Mller, S., & Raake, A. (2014). Quality of Experience: Advanced Concepts, Applications and Methods. En *Springer eBooks*. http://ci.nii.ac.jp/ncid/BB16277955

[4] Alvarez Pérez, V., & Aguila Medina, D. (2013). Herramienta para la medición de la calidad de experiencia en servicios streaming de audio y video sobre redes cableadas de datos entre extremos [Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas]. UCI.

[5] Wang, Y., Sun, M., Wang, K., & Zhang, L. (2016). Quality of experience estimation with layered mapping for hypertext transfer protocol video streaming over wireless networks. International Journal of Communication Systems, 29(14), 2084-2099. https://doi.org/10.1002/dac.2911

[6] FFMpeg-Python: Python bindings for FFMPEg — fFMpeg-Python documentation. (s. f.). https://kkroening.github.io/ffmpeg-python/

[7] Network Throttling in Puppeteer • Fahim Dalvi. (2018, 5 febrero). Fahim Dalvi. https://fdalvi.github.io/blog/2018-02-05-puppeteer-network-throttle/

[8] Tsbmail. (2017). P.1203 : Evaluación paramétrica de la calidad basada en el tren de bits de los servicios audiovisuales de emisión de secuencias de descarga progresiva y adaptativa a través de un transporte fiable. https://www.itu.int/rec/T-REC-P.1203/es

[9] Castillo, P. F. G. (2022). Evaluación de la QOE en un sistema de streaming adaptativo de vídeo 3D basado en DASH. https://doi.org/10.4995/thesis/10251/186354

[10} Cisco, “Cisco Visual Networking Index: Forecast and Methodology, 2016-2021.”https://www.cisco.com/c/en/us/solutions/collateral/serviceprovider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf, June 2017

[11] Cantor, L. (Director). (2016). THE GLOBAL INTERNET PHENOMENA REPORT JANUARY 2023. Sandvine.

[12] Wang, Y., Sun, M., Wang, K., & Zhang, L. (2016). Quality of experience estimation with layered mapping for hypertext transfer protocol video streaming over wireless networks. International Journal of Communication Systems, 29(14), 2084-2099. https://doi.org/10.1002/dac.2911

[13] Eswara, N., Manasa, K., Kommineni, A., Chakraborty, S., Sethuram, H. P., Kuchi, K., Kumar, A., & Channappayya, S. S. (2018). A continuous QOE evaluation framework for video streaming over HTTP. IEEE Transactions on Circuits and Systems for Video Technology, 28(11), 3236-3250. https://doi.org/10.1109/tcsvt.2017.2742601

[14] ITU Telecommunication Standardization Sector, "ITU-T Rec P 1203 Models and tools for quality assessment of streamed media," 2017

[15] Brweb. (2023). BT.500: Methodologies for the Subjective Assessment of the Quality of Television Images. https://www.itu.int/rec/R-REC-BT.500-15-202305-I/es

[16] Lederer, S. (2023, 6 enero). Optimal Adaptive Streaming Formats MPEG-DASH & HLS Segment Length - Bitmovin. Bitmovin. https://bitmovin.com/mpeg-dash-hls-segment-length/

[17] Docker. (s. f.). https://hub.docker.com/_/httpd/

[18] Barakabitze, A. A., Barman, N., Ahmad, A., Zadtootaghaj, S., Sun, L., Martini, M. G., & Atzori, L. (2020). QOE Management of Multimedia Streaming Services in Future Networks: A Tutorial and survey. IEEE Communications Surveys and Tutorials, 22(1), 526-565. https://doi.org/10.1109/comst.2019.2958784

[19a] Itu-P. (2023). GitHub - ITU-P1203/ITU-P1203: ITU-T Rec. P.1203 Implementation. GitHub. https://github.com/itu-p1203/itu-p1203/

[19b] Raake, A., Garcia, M.-N., Robitza, W., List, P., Göring, S., Feiten, B. (2017). A bitstream-based, scalable video-quality model for HTTP adaptive streaming: ITU-T P.1203.1. In 2017 Ninth International Conference on Quality of Multimedia Experience (QoMEX). Erfurt.

[19c] Robitza, W., Göring, S., Raake, A., Lindegren, D., Heikkilä, G., Gustafsson, J., List, P., Feiten, B., Wüstenhagen, U., Garcia, M.-N., Yamagishi, K., Broom, S. (2018). HTTP Adaptive Streaming QoE Estimation with ITU-T Rec. P.1203 – Open Databases and Software. In 9th ACM Multimedia Systems Conference. Amsterdam.

[20] MySQL :: MySQL Workbench. (s. f.). https://www.mysql.com/products/workbench/

[21] RTVE.es. (2023). Noticias de última hora, programas y series de televisión - RTVE.es. https://www.rtve.es/

[22] Tsbmail. (2022). P.910 : Métodos subjetivos de evaluación de la calidad vídeo para aplicaciones multimedios. https://www.itu.int/rec/T-REC-P.910-202207-I/es

[23] ITU Telecommunication Standardization Sector, "ITU-T Rec P 1204.3 (Amendment 1). Models and tools for quality assessment of streamed media," 2021

[24] Juluri, P., Tamarapalli, V., & Medhi, D. (2016). Measurement of Quality of experience of Video-on-Demand Services: a survey. IEEE Communications Surveys and Tutorials, 18(1), 401-418. https://doi.org/10.1109/comst.2015.2401424

[25] Telecommunication-Telemedia-Assessment. (2023). GitHub. Telecommunication Telemedia Assessment/bitstream_Mode3_P1204_3: Open Source Reference Implementation of ITU-T P.1204.3. GitHub. https://github.com/Telecommunication-Telemedia-Assessment/bitstream_mode3_p1204_3#introduction