



Aproximación hacia arquitecturas modulares asistidas por computación en el borde para optimización de comunicaciones holográficas

Genís Castillo, Sergi Fernández, David Rincón, Mario Montagud.
genis.castillo@i2cat.net, sergi.fernandez@i2cat.net, david.rincon@upc.edu, mario.montagud@i2cat.net

Resumen

Este artículo presenta una aproximación inicial hacia arquitecturas modulares asistidas por computación en el borde (Edge Computing) orientadas a una optimización de la prestación de servicios de comunicaciones holográficas multiusuario. Se propone una modularización de componentes y módulos en el lado servidor que, por una parte, permite desacoplar el plano de control del plano de datos de usuario y, por otra parte, facilita una gestión dinámica de conexiones en función de requerimientos específicos y a los recursos disponibles. La arquitectura propuesta, junto con sus mecanismos de orquestación asociados, se adapta y reconfigura mediante la especificación e integración de APIs de red compatibles con iniciativas de estandarización actuales. Los beneficios de esta aproximación se validan preliminarmente a través de pruebas experimentales que evidencian, por un lado, las ventajas de utilizar servidores de borde para la gestión de comunicaciones, y por otro, el impacto positivo del balanceo de carga entre componentes servidor en cuanto a la escalabilidad y estabilidad del servicio.

Palabras Clave—Arquitecturas de Comunicaciones, Comunicaciones Holográficas, Calidad de Servicio, Computación en el Borde, Escalabilidad, Streaming Adaptativo

I. INTRODUCCIÓN

En los últimos años están emergiendo plataformas innovadoras de comunicaciones holográficas que permiten superar limitaciones esenciales de los servicios de videoconferencia 2D tradicionales en cuanto a niveles de calidad de interacción, inmersión y co-presencia [1] [2].

Sin embargo, a pesar de su indudable potencial, las escasas soluciones existentes presentan asimismo limitaciones en cuanto a su capacidad de adaptabilidad y escalabilidad, debido a sus estrictos requisitos en cuanto a latencia, ancho de banda y capacidad de procesado, especialmente cuando están basadas en formatos de vídeo volumétrico capturado en tiempo real [3] [4].

Este artículo presenta una aproximación inicial hacia arquitecturas modulares asistidas por computación en el

borde (Edge Computing) orientadas a una optimización de la prestación de servicios de comunicaciones holográficas multiusuario, en cuanto a su escalabilidad, consumo de recursos y niveles de Quality of Service (QoS) ofrecidos. En particular, se propone una modularización de componentes en el lado servidor que, por una parte, permite desacoplar el plano de control del plano de datos de usuario y, por otra parte, facilita una gestión dinámica de conexiones en función de requerimientos específicos y a los recursos disponibles. La arquitectura propuesta, junto con sus mecanismos de orquestación asociados, se adapta y reconfigura mediante la especificación e integración de APIs de red compatibles con iniciativas de estandarización actuales.

Los beneficios de esta aproximación se validan preliminarmente a través de pruebas experimentales en despliegues a través de entornos de red distribuidos reales y evidencian dos ventajas fundamentales: (i) la capacidad de seleccionar servidores en el borde de la red, cercanos a los clientes, permite reducir significativamente el retardo de las comunicaciones (y el uso de recursos global); y (ii) la capacidad de balancear la carga de tráfico entre componentes servidor distribuidos contribuye a una mayor escalabilidad y estabilidad del servicio.

II. ESTADO DEL ARTE

A. Estudios de revisión

Los artículos en [3] y [4] proporcionan una revisión profunda sobre avances y retos pendientes en el campo del vídeo volumétrico. Por un lado, estos estudios reflejan los altos requerimientos de ancho de banda, procesamiento, y retardo que suelen presentar los servicios de eXtended Reality (XR) interactivos e inmersivos asociados a estos formatos emergentes. Por otro lado, subrayan la relevancia de adoptar funciones de procesamiento asistido por red (p.ej., transcodificación, conversión entre formatos, renderizado remoto) para optimizar el rendimiento, interoperabilidad y minimizar los costes asociados.

B. Plataformas de comunicaciones holográficas

Muchas plataformas de Social Virtual Reality (VR) se han propuesto con tal de superar limitaciones intrínsecas de los sistemas de videoconferencia 2D. No obstante, la mayoría de estas plataformas se basa en avatares, lo que conlleva limitaciones importantes en cuanto a la preservación de la identidad, y calidad de interacción / comunicación. En este contexto, estudios recientes (e.g. [1]) han demostrado que: (i) las plataformas de Social VR basadas en avatares ofrecen una mejor calidad de interacción y niveles de (co-)presencia mayores que las plataformas de videoconferencia 2D; y (ii) estos niveles mejoran aún más cuando se utilizan representaciones realistas de los usuarios mediante video volumétrico (esto es, hologramas), ofreciendo una experiencia de interacción cercana a las reuniones presenciales.

Estos hallazgos son prometedores, especialmente considerando que la resolución y calidad de los hologramas en las plataformas Social VR utilizadas en dichos estudios (e.g., [1], [2], [5]) todavía presentan un amplio margen de mejora.

Se prosigue ahora con las principales plataformas Social VR/XR que admiten representaciones holográficas 3D. En [5] se presenta una plataforma basada en el uso sensores RGB-D, tales como Kinect, y entornos virtuales estáticos basados en vídeo 360°. Los flujos Video + Depth (RGB-D) se codifican mediante codecs de vídeo 2D (como H.264 o VP8) y se transmiten vía Web Real Time Communications (WebRTC). El sistema soporta una resolución de 1080×960 píxeles, que requiere un ancho de banda superior a 5Mbps por cada flujo, y permite sesiones con hasta 3 usuarios simultáneos.

En [2] se presenta una plataforma de comunicaciones holográficas multi-usuario en tiempo real, utilizando también sensores RGB-D, como Kinect. En esta plataforma los usuarios pueden ser capturados por uno o varios sensores, y sus representaciones se transmiten como flujos de Nube de Puntos mediante Socket.io (TCP websockets) a través de una Selective Forwarding Unit (SFU) en la nube. Con una sola cámara, cada flujo requiere entre 5 y 7 Mbps (15 fps y ~50K puntos por fotograma), observándose cuellos de botella a partir de 4 usuarios por sesión. Esta plataforma, denominada HoloMIT, constituye el punto de partida de este trabajo.

C. Optimización de servicios de comunicaciones holográficas

Aunque las representaciones realistas de usuarios mediante vídeo volumétrico mejoran la experiencia frente al uso de avatares [1], también plantean retos de escalabilidad e interoperabilidad por sus altos requerimientos de procesamiento y ancho de banda [2]. Por ello, la comunidad científica ha centrado esfuerzos en optimizar los servicios de comunicaciones holográficas y XR en general. En este contexto, destacan trabajos recientes orientados a mejorar la escalabilidad y adaptabilidad de estos servicios inmersivos en tiempo real. En primer lugar, se han propuesto estrategias avanzadas de procesamiento

y distribución basadas en campo de visión, en el lado cliente, con tal de mejorar la adaptabilidad y estabilidad de servicios de streaming de nubes de puntos. Por ejemplo, [6] y [7] introdujeron técnicas de transmisión adaptativa basadas en *tiling*, dividiendo los flujos volumétricos en secciones cúbicas codificadas a distintas tasas de bits, y entregando las calidades más apropiadas de estos cubos según la posición y campo de visión del usuario objetivo, así como al ancho de banda disponible, en cada momento. También se han desarrollado optimizaciones gestionadas desde el lado servidor para la transmisión de contenidos XR y servicios de comunicaciones holográfica. En [8] se presentó una versión mejorada de la plataforma presentada en [5], incorporando una Multipoint Control Unit (MCU) en la nube que genera un mosaico fusionado a partir de flujos RGB-D individuales desde cada cliente, usando códecs de vídeo 2D. Evaluada con usuarios simulados, la plataforma soportó hasta 8 participantes (con una resolución de 540×800 píxeles por flujo), disminuyendo la carga de tráfico en los clientes en comparación al uso de una SFU, aunque sin considerar métricas como latencia y fluidez en el estudio presentado. Sin embargo, esta solución no soporta representaciones volumétricas completas, ni aprovecha plenamente las características intrínsecas de los entornos 3D, como son las posiciones y campos de visión variantes. De forma similar, [9] introdujo una MCU para hologramas pre-grabados y entornos 3D, incorporando fusión de escenas, transcodificación dinámica y envío personalizado para cada cliente en función de su posición y campo de visión instantáneo. En [10], se integró una versión evolucionada de esta MCU en la plataforma HoloMIT [2], ejecutando en tiempo real dichas estrategias para generar flujos personalizados según distancias y puntos de vista para cada cliente. Esto permitió prácticamente duplicar el número de usuarios por sesión (desde 7 hasta 13), reduciendo el uso de recursos computacionales (CPU, GPU, RAM) y ancho de banda, así como manteniendo la latencia dentro de límites aceptables.

En un contexto relacionado, [11] propuso técnicas para mejorar la escalabilidad y estabilidad de servicios de videoconferencia multi-usuario basados en web y vídeo 2D, mediante: (i) un esquema de QoS que prioriza flujos según la posición de los usuarios; (ii) filtrado por *frustum culling*; y (iii) un gestor de recursos que distribuye conexiones entre servidores según la proximidad en el entorno virtual. Además, [12] propuso una arquitectura para servicios colaborativos multi-usuario en XR, basada en renderizado remoto y sincronización Cloud-Edge, logrando mejoras significativas en cuanto a escalabilidad (hasta un 50% más de usuarios) y rendimiento (frecuencia de renderizado casi 8 veces superior) frente a soluciones previas. De forma complementaria, [13] mostró que el renderizado remoto para habilitar clientes web ligeros en servicios de comunicaciones holográficas puede añadir unos 200ms de latencia, considerada aceptable para experiencias interactivas en tiempo real.

Finalmente, [14] ofrece una revisión detallada sobre habilitadores tecnológicos para servicios XR en la nube,

incluyendo avances en estandarización como European Telecommunications Standards Institute (ETSI) Multi-access Edge Computing (MEC), nuevas funcionalidades en 3rd Generation Partnership Project (3GPP) para mejorar la QoS en XR, y nuevos formatos y protocolos Moving Picture Experts Group (MPEG).

D. Contribuciones de este trabajo

Este artículo parte de la plataforma HoloMIT [2], y se centra en su extensión y evolución con tal de soportar arquitecturas de despliegue modulares y asistidas por computación en el borde (Edge) que permitan un instanciamiento dinámico de SFUs para sesiones compartidas, mejorando así el rendimiento, la escalabilidad y la sostenibilidad de los servicios de comunicación holográfica. Aunque no se introduce un nuevo componente tecnológico específico para procesamiento XR en la nube, las contribuciones pueden considerarse un habilitante de escalabilidad *per se*, y son compatibles con futuras extensiones que integren funcionalidades de procesado en la nube / borde como, por ejemplo una MCU [10] o un Renderizador Remoto [13], facilitando su incorporación en fases posteriores de investigación.

III. PLATAFORMA EXTENDIDA

A. Plataforma de Partida: HoloMIT

A fecha de inicio de este trabajo, el componente Orquestador (Orchestrator) de HoloMIT se basaba en una implementación monolítica, integrando en un único paquete software las funcionalidades de gestión de usuarios, sesiones, conexiones y reenvío de datos multimedia mediante SFU [2]. Aunque funcional, esta implementación claramente no era la más adecuada ni eficiente, ya que no permitía una escalabilidad efectiva. Además, combinaba funciones del plano de control y del plano de datos en un mismo módulo.

B. Orquestrador Modular

Este trabajo introduce tres mejoras clave para el componente Orquestador, mediante la especificación y adopción de una arquitectura modular y desacoplada (véase Figura 1). Primero, se ha desacoplado el plano de control del plano de datos de usuario, permitiendo instancias únicas y reutilizables de servicios generales (como gestión de usuarios y sesiones) en la nube, así como instancias múltiples bajo demanda de servicios / módulos específicos (como las SFUs). Segundo, todos los nuevos servicios modulares han sido virtualizados, lo que permite su despliegue dinámico en servidores Edge o en la nube, en base a recursos y demandas existentes, por cada sesión. Tercero, se han incorporado nuevos servicios para integrarse con plataformas de orquestación en el Edge y otros elementos / APIs de red para exponer recursos y funcionalidades de interés, como detección de congestión o estrategias de selección de servidores Edge en los que instanciar funcionalidades de procesado

multimedia específicas.

A continuación, se presentan los diferentes módulos que forman parte de los servicios que lo componen:

- **User Manager:** Responsable del registro, gestión y provisión de datos de los clientes, escenarios y otros componentes en la nube, utilizando MongoDB [15].
- **Session Manager:** Encargado de gestionar el ciclo de vida de las sesiones multi-usuario (creación, unión, abandono y destrucción), tanto para cada cliente como para cada escenario virtual seleccionado, almacenando la información correspondiente en MongoDB.
- **Clock Manager:** Responsable de garantizar sincronización entre todas las entidades involucradas en la sesión multimedia. En caso de no disponer de acceso a un servidor Network Time Protocol (NTP), permite proporcionar sincronización mediante un protocolo ad-hoc de intercambio y alineamiento de marcas de temporización, sin necesidad de disponer de conexión a Internet.
- **Index Manager:** Nuevo manager que decide y orquesta los servidores en el Edge o en la nube donde se desplegarán managers del plano de usuario (como SFUs, MCUs), y gestiona su ciclo de vida en coordinación con una plataforma de orquestación de recursos Edge. También actúa como interfaz de acceso para explotar funcionalidades / APIs de red específicas (por ejemplo, detección de congestión, identificación de ubicación, descubrimiento y selección de nodos Edge) [16].

C. Arquitectura Multi-SFU

En este trabajo, la SFU ha sido evolucionada e integrada como un conjunto de módulos que actúan como managers o servicios:

- **Media Manager:** Gestiona el intercambio de flujos audiovisuales entre los clientes (es decir, actúa como una SFU para audio y vídeo). Puede interactuar con otras SFUs si es necesario e identificar tuplas [*dirección Internet Protocol (IP), puerto, protocolo*] para cada conexión activa, permitiendo así que otros elementos y APIs de red apliquen ciertas optimizaciones (por ejemplo, Access Traffic Steering, Switching and Splitting (ATSSS), slicing, etc.).
- **Events Manager:** Encargado de reenviar eventos o metadatos relevantes entre los clientes, como sus posiciones en el espacio virtual o información de control relacionada con la sesión multimedia (p.ej., interacciones con el entorno, actualizaciones de estado, etc.).

D. Cliente web sin interfaz gráfica (headless)

La disponibilidad de clientes / reproductores completos en plataformas nativas y dispositivos con disponibilidad de GPU es esencial para ofrecer servicios de comunicación holográfica multi-usuario de manera eficiente, debido a los exigentes requisitos de ancho de banda y procesamiento [2], [10]. En la plataforma HoloMIT, el player nativo se

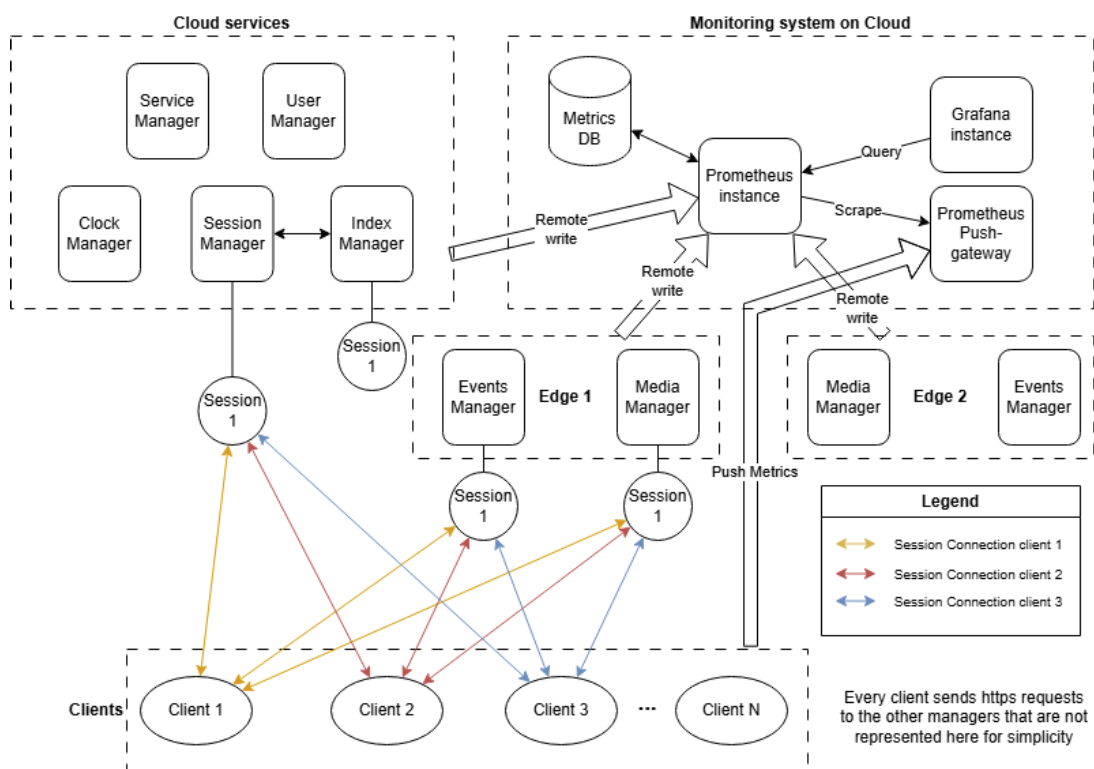


Figura 1: Nueva versión del orquestador modular con un subsistema de métricas integrado

ha implementado en Unity y está diseñado para sistemas operativos Windows, debido a dependencias específicas de ciertas librerías y módulos software. Sin embargo, utilizar reproductores nativos completos no resulta eficiente en términos de esfuerzo y coste para realizar pruebas extensivas de escalabilidad y rendimiento. Por este motivo, en este trabajo se ha desarrollado un nuevo reproductor ligero sin interfaz gráfica (*headless*) basado en tecnologías web. El objetivo es disponer de un reproductor que requiera menor consumo de recursos (espacio, GPU y memoria), y sea más fácil y ágil de desplegar para pruebas experimentales. Además, dado que este trabajo se centra principalmente en pruebas objetivas de rendimiento y uso de recursos, no es necesario que los hologramas se capturen en origen ni se visualicen en destino, lo que permite omitir funciones de procesamiento intensivo como la captura, codificación, decodificación y el renderizado, en línea con las características de un cliente *headless*.

Concretamente, el cliente web *headless* se ha implementado mediante Node.js [17], emulando el comportamiento y las funcionalidades de transmisión y recepción del reproductor nativo, interactuando con los managers necesarios del Orchestrator y comunicándose con el resto de clientes de la sesión a través de los managers de comunicación seleccionados (p.ej. Media o Events Manager).

E. Subsistema de métricas

Se ha desarrollado y adoptado un subsistema de métricas (véase Figura 1) basado en Prometheus [18] y Grafana [19], con el fin de medir, registrar y monitorizar métricas

de rendimiento y uso de recursos de distintos componentes de HoloMIT, durante sesiones de prueba.

Se emplean distintos exportadores de Prometheus para recopilar métricas de los módulos de interés. Por ejemplo, Node Exporter [20] proporciona métricas como uso de CPU, memoria y ancho de banda, mientras que Cadvisor [21] ofrece métricas relacionadas con contenedores Docker, como el ancho de banda utilizado por componentes como la SFU. Además, se ha integrado Prometheus Push Gateway [22] para exportar métricas desde el cliente nativo Unity y el cliente web *headless*. Adicionalmente se han adoptado dos exportadores adicionales para obtener métricas de los clientes nativos Unity: (i) Windows Exporter [23], que resulta más adecuado para sistemas Windows que Node Exporter; y (ii) dcm-exporter [24], útil para recopilar métricas relacionadas con la GPU.

Dado que Prometheus funciona como un sistema de métricas por extracción (*pull*), se ha desplegado una instancia de Prometheus como agente en cada máquina. Esto permite realizar escrituras remotas (esto es, *remote writing*) hacia una instancia general de Prometheus y visualizar todas las métricas en un único lugar, sin necesidad de conocer la IP de cada máquina, lo cual representa una ventaja clara para la ejecución de pruebas.

A continuación, se enumeran las principales métricas que se soportan:

- **SFU:** latencia de procesado por fotograma (ms), desviación de sincronización con el Clock Manager (ms), uso de CPU (%), uso de memoria (MB), uso

de ancho de banda (Mbps).

- **Cliente web headless:** latencia desde transmisión hasta recepción (ms), desviación de sincronización con el Clock Manager (ms), uso de CPU (%), uso de memoria (MB), uso de ancho de banda (Mbps).
- **Cliente nativos Unity:** tiempo desde captura hasta renderizado (ms), tiempo de codificación (ms), fps de codificación, tiempo de decodificación (ms), fps de decodificación, desviación de sincronización con el Clock Manager (ms), uso de CPU (%), uso de GPU (%), uso de memoria (MB), uso de ancho de banda (Mbps).

IV. RESULTADOS

A continuación se presentan resultados para 2 tipos de tests realizados: (i) comparación entre el uso de una SFU cercana vs una SFU más lejana; (ii) comparación de límites de escalabilidad cuando se usa una sola SFU vs cuando se utilizan múltiples SFUs para una sesión dada.

A. Test 1. SFU cercana vs SFU lejana

A1. Descripción y Objetivos: Se llevan a cabo sesiones de 2 usuarios cuando se usa una SFU cercana (p.ej. en la misma red local) vs una SFU más lejana (esto es, en una red remota).

El objetivo es evaluar la existencia de diferencias en términos técnicos y beneficios potenciales (p.ej., latencia, estabilidad, consumo de recursos) cuando se posibilita el uso de SFUs en servidores Edge o en la nube. Para ello, se recrean y evalúan dos escenarios análogos con 2 usuarios, uno con una SFU cercana y otro con una SFU remota. En ambas condiciones de prueba se miden y comparan métricas relevantes como latencia, fps y consumo de ancho de banda, durante sesiones de aproximadamente 5 minutos.

A2. Configuración y Despliegue: Para ambas condiciones de test se configura una sesión de HoloMIT con 2 clientes remotos (web sin interfaz gráfica) y una SFU (Media / Events Manager). Para la condición de test de SFU cercana tanto los clientes como la SFU se despliegan en la misma infraestructura de Azure en el Norte de Europa (Irlanda). Para la condición de test de SFU lejana, los clientes se trasladan a la región Oeste de Europa.

A3. Resultados:

A3a. Test 1.1 - SFU cercana: Como era de esperar, los niveles de latencia obtenidos fueron bajos, aunque con ligeras fluctuaciones (véase Figura 2a) posiblemente por desviaciones en el proceso de sincronización de relojes. En particular, la latencia media fue de aproximadamente 4 ms, y la latencia de procesamiento introducida por la SFU fue de 1.2 ms, un valor muy bajo pero relevante para esta prueba. Este retardo de procesamiento se midió mediante Wireshark [25], aunque debe tenerse en cuenta que las mediciones asociadas pueden verse afectadas por el reloj interno de la máquina, interrupciones de la tarjeta de red o multitarea del sistema operativo. Aun así, ofrecen una buena estimación del procesamiento de la SFU.

Name	Mean	Max	Min
latency_from_2269406695_to_Player_1-351537849	5.19 ms	25 ms	2 ms
latency_from_351537849_to_Player_0-2269406695	3.71 ms	23 ms	1 ms

(a) Latencia en el Test 1.1 (SFU cercana)

Name	Mean	Max	Min
latency_from_1162162909_to_Player_0-1640133122	19.9 ms	35 ms	18 ms
latency_from_1640133122_to_Player_1-1162162909	19.8 ms	35 ms	18 ms

(b) Latencia en el Test 1.2 (SFU lejana)

Figura 2: Resultados de latencia experimentada cuando se usa SFU cercana vs lejana

A3b. Test 1.2 - SFU lejana: La Figura 2b muestra los valores de latencia experimentados por los clientes al conectarse a una SFU lejana (clientes en Europa Occidental y SFU en Europa del Norte). Como era previsible, la latencia aumentó con respecto al Test 1.1, situándose en torno a los 18 ms, con ligeras fluctuaciones atribuibles también a la sincronización de relojes.

Adicionalmente, también se comprobó experimentalmente que la diferencia de retardos entre la selección de diferentes servidores Azure desde una localización dada para un cliente local externo puede alcanzar niveles en torno a 200ms. Ello refleja la relevancia de poder seleccionar el servidor de interés para ofrecer servicios de comunicaciones en tiempo real, sensibles al retardo.

B. Test 2. Escalabilidad con una sola SFU

B1. Descripción y Objetivos: Se persigue determinar el límite de escalabilidad cuando se adopta una única SFU por sesión, dada una configuración específica de despliegue (véase Figura 3), mediante el aumento progresivo del número de usuarios por sesión. Se empieza con sesiones de 2 usuarios, y se va incrementando el número de usuarios de dos en dos en cada iteración del test. Una vez alcanzado el límite superior, se puede reducir el número en una unidad para identificar el número máximo de usuarios que aún permite un rendimiento estable.

B2. Configuración y Despliegue: Dado que las especificaciones y recursos del servidor que ejecuta la SFU pueden desempeñar un papel clave, se han llevado a cabo condiciones de prueba utilizando dos máquinas virtuales de Azure con diferentes configuraciones.

Los clientes sin interfaz gráfica se ejecutaron en una máquina virtual de Azure con especificaciones Standard_DS1_v2, desplegada en Europa Occidental. Los subcomponentes del Orquestador y los módulos/managers de la SFU también se desplegaron en una máquina virtual de Azure, en este caso ubicada en Europa del Norte. En una de las condiciones de prueba se utilizó una máquina virtual Standard_DS1_v2, mientras que en otra se empleó una Standard_DS2_v2 para ejecutar la SFU, con tal de evaluar el impacto de la diferencia de recursos y capacidades del

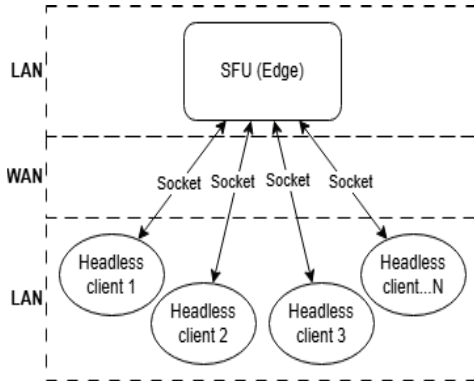


Figura 3: Configuración de test para el límite de escalabilidad con SFU única

servidor en la nube en términos de escalabilidad de la sesión.

En cada una de las condiciones de test, los clientes sin interfaz gráfica se han añadido progresivamente a una sesión compartida. Al alcanzarse el límite de escalabilidad (es decir, cuando se observan fallos), se reduce el número de clientes en una unidad. Si la sesión resulta estable, ese es el número de clientes que determina el límite de escalabilidad en régimen estable; en caso contrario, se considera el número de clientes antes de los fallos. En cada condición de test, se miden y comparan métricas clave como los retardos, el consumo de ancho de banda y el uso de recursos computacionales, durante sesiones de aproximadamente 5 minutos.

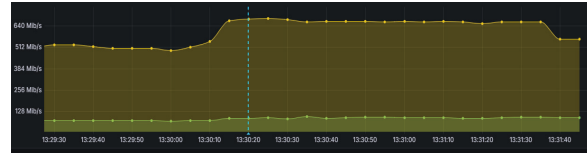
Además, se contempla otra condición de test adicional, que consiste en el uso de sockets volátiles (sin retransmisión) para determinar si bajo esta configuración de streaming mejoran las prestaciones; en caso de ser así, se adoptará esta configuración para cada una de las sesiones de test.

B3. Resultados:

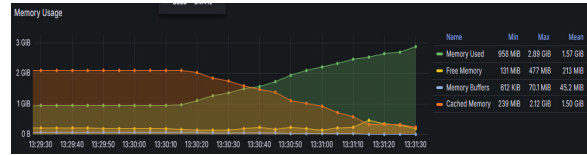
B3a. Test 2.1: En esta condición de test, la diferencia entre el ancho de banda recibido y transmitido por la SFU, así como la cantidad total de datos intercambiados, comenzó a ser significativa al llegar a 8 clientes por sesión, lo que podría representar una barrera en ciertos escenarios de despliegues. En esta situación, mientras que la SFU recibe $N_{Clients} = 8$ flujos (aproximadamente 75,5 Mbps), debe transmitir $N_{Clients} * (N_{Clients} - 1) = 56$ flujos (más de 500 Mbps) a todos los clientes remotos. A pesar de ello, la sesión se ejecutó sin problemas.

Al llegar a 10 clientes, ya se empezaron a percibir problemas de estabilidad y escalabilidad. En este caso, la SFU debería recibir $N_{Clients} = 10$ flujos (alrededor de 100 Mbps) y transmitir $N_{Clients} * (N_{Clients} - 1) = 90$ flujos (aproximadamente 1 Gbps) a todos los participantes. Sin embargo, como se observa en la Figura 4a, el ancho de banda transmitido no superó los ~682 Mbps cuando se unieron los clientes 9 y 10, ya que el límite de la máquina virtual adoptada para desplegar la SFU era de 750 Mbps, y ya se estaban recibiendo ~105 Mbps en el canal de subida. En este caso, la SFU experimentó sobrecargas

de memoria debido a su incapacidad para gestionar el volumen total de datos (véase Figura 4b), lo que provocó el apagado automático del contenedor Docker para preservar los recursos de la máquina virtual. Este fue el punto de fallo para el Test 2.1.



(a) Fallo de Test 2.1 con 10 usuarios – valores de ancho de banda



(b) Fallo de Test 2.1 con 10 usuarios – valores de memoria

Figura 4: Resultados del Test 2.1

Tras analizar los resultados de este primer Test 2.1, se observó que el desbordamiento de memoria (Figura 4b) fue potencialmente provocado por la retransmisión de segmentos TCP (socket.io). Así pues, se consideró el uso de sockets volátiles con el fin de evaluar posibles mejoras. En esta condición de test, aunque no se logró aumentar el número de clientes con dicha configuración, no se produjo desbordamiento de memoria, aunque sí se registró una pérdida significativa de paquetes. Ello también puede considerarse un límite operativo, aunque resulta mucho más estable y conveniente que la inhabilitación del contenedor por exceso de uso de memoria.

B3b. Test 2.2: En este caso, el uso de una máquina virtual con especificaciones más altas (Standard_DS2_v2) para el despliegue de la SFU resultó en el soporte de un mayor número de usuarios por sesión, llegando hasta 12 usuarios sin pérdidas ni estabilidades significativas. Esto de alguna manera confirma que el límite en Test 2.1 fue más debido a la tarjeta de red que no a la CPU o la memoria. Sin el uso de sockets volátiles, ocurrió una situación similar con 13 usuarios a la ocurrida con 10 usuarios en el Test 2.1. Esto valida que las fórmulas analíticas dan una idea aproximada adecuada del límite de escalabilidad según el ancho de banda máximo disponible, además de que el límite siempre se encuentra en el ancho de banda transmitido.

C. Test 3. SFU única vs SFU múltiples

C1. Descripción y Objetivos: Tras identificar los límites de escalabilidad con una única SFU, se preparan escenarios con múltiples SFUs por sesión para evaluar si se obtienen mejoras en cuanto a escalabilidad y, en tal caso, en qué medida. También se considera el despliegue de las SFUs en distintas regiones continentales. En cada una de las condiciones experimentales, al igual que para los tests anteriores, los clientes sin interfaz gráfica se

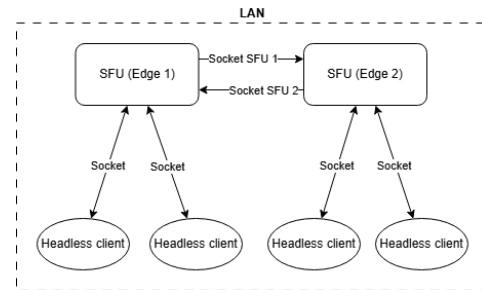
incorporan progresivamente en grupos de dos a una sesión compartida, distribuyéndolos de manera equilibrada entre las distintas SFUs. Para cada condición de prueba, se miden y analizan métricas relevantes como la latencia, el consumo de ancho de banda y el uso de recursos computacionales, durante sesiones de aproximadamente 5 minutos.

C2. Configuración y Despliegue: Se consideran despliegues de la SFU en distintas regiones geográficas. Como se muestra en la Figura 5a, se lleva a cabo una primera prueba (Test 3.1) interconectando dos SFUs que atienden a los clientes de una misma sesión multi-usuario. Posteriormente, se realiza una prueba similar (Test 3.2), sin interconectar las SFUs, pero modificando el comportamiento de los clientes para que puedan recibir datos desde más de una SFU en el canal de bajada, como se ilustra en la Figura 5b. En este segundo caso, los clientes siguen enviando sus datos a una única SFU en el canal de subida, pero reciben datos de otros clientes desde múltiples SFUs. Finalmente, se realiza una tercera prueba (Test 3.3) desplegando tres SFUs en distintas regiones de Azure. Esta prueba tiene un doble objetivo: (i) evaluar si el uso de tres SFUs mejora la escalabilidad en comparación con escenarios con dos SFUs; y (ii) comprobar si el efecto multiplicativo en el canal de bajada puede mitigarse al desplegar las SFUs en diferentes regiones y, por tanto, en redes distintas. En este caso, las conexiones entre SFUs seguirán el patrón del Test 3.2, ya que tanto empíricamente como analíticamente se comprobó que se reduce la sobrecarga de red en comparación con el Test 3.1.

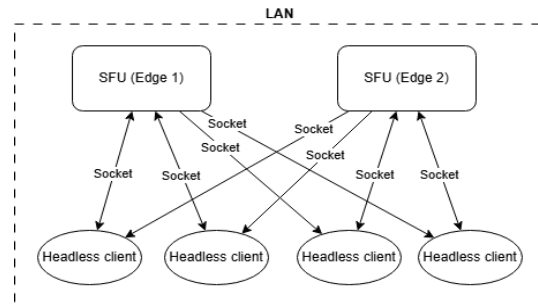
En cuanto al despliegue específico, los subcomponentes del Orquestador y los módulos o managers de SFU fueron desplegados en una máquina virtual Azure Standard_DS1_v2. En los Test 3.1 y 3.2, las SFUs se desplegaron en la región de Europa del Norte, mientras que en el Test 3.3 se distribuyeron entre las regiones de Europa del Norte, Europa Occidental e Italia del Norte, con el objetivo de evaluar sesiones distribuidas geográficamente. Los clientes sin interfaz gráfica también se desplegaron en Azure (en Europa Occidental para los Test 3.1 y 3.2, y distribuidos entre regiones en el Test 3.3).

C3. Resultados:

C3a. Test 3.1 - 2 SFUs interconectadas: En una condición de prueba con 4 clientes (2 por SFU), los valores analíticos esperados de ancho de banda son 76.8 Mbps para la transmisión y 38.4 Mbps para la recepción. Estos valores teóricos presentan una alta correlación con los valores obtenidos experimentalmente, que fueron de 76.2 Mbps y 38.65 Mbps para el ancho de banda transmitido y recibido, respectivamente. En una condición de prueba con 12 clientes (6 por SFU) se está cerca del límite superior. Mientras que el ancho de banda recibido coincide con el valor analítico (alrededor de 115 Mbps), el ancho de banda transmitido real (aproximadamente 600 Mbps) fue ligeramente inferior al valor teórico (691.2 Mbps). Esto indica que se alcanzó el límite de escalabilidad. No obstante, se realizó una prueba adicional con 13 usuarios



(a) SFUs interconectadas



(b) SFUs no interconectadas

Figura 5: Configuración de Test 3

para determinar con mayor precisión el límite operativo de la SFU. Los resultados se muestran en la Figura 6, en la que puede observarse que una de las SFUs alcanza su límite de capacidad, lo cual se evidencia en la línea casi recta del gráfico correspondiente al ancho de banda transmitido por la SFU en cuestión, indicando que no es posible enviar más datos y que se produce una pérdida de paquetes.

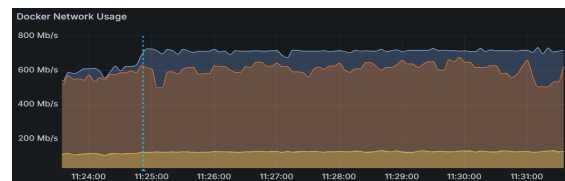


Figura 6: Ancho de banda del Test 3.1 con 13 clientes

C3b. Test 3.2 - Dos SFU no interconectadas por sesión: En esta test, los clientes se añadieron progresivamente a la sesión y se asignaron de manera balanceada a una de las dos SFUs para la transmisión de datos, aunque recibían datos de ambas SFUs. En una condición de prueba con 4 clientes (2 por SFU), los valores experimentales de ancho de banda transmitido y recibido mostraron una alta correlación con los valores analíticos (19.2 Mbps y 57.6 Mbps, respectivamente), lo que evidencia un rendimiento satisfactorio. Con 12 clientes (6 por SFU), la prueba se mantuvo estable, con valores de ancho de banda recibido muy próximos a los analíticos (57 Mbps). No obstante, ya se observaron desviaciones en el ancho de banda transmitido respecto al valor teórico (633 Mbps). Se añadió

un cliente adicional para evaluar el rendimiento en dicha condición de test. Con 13 clientes, las desviaciones en el ancho de banda transmitido fueron más notables que con 12 clientes, aunque la evolución del ancho de banda se mantuvo relativamente estable. Con 14 clientes, el ancho de banda no aumentó en comparación con las condiciones de test anteriores, lo que indica que se alcanzaron los límites de capacidad de la SFU. Por tanto, se puede considerar que el límite de escalabilidad y estabilidad para este Test 3.2 se sitúa en 12 clientes.

C3c. Test 3.3 - Tres SFUs no interconectadas por sesión en distintas regiones: Como primera condición de test, se ejecutó una sesión con 12 usuarios (4 conectados a cada SFU). En la Figura 7 se observa que se obtuvieron tres grupos principales de valores de latencia, uno por región, como era de esperarse. No obstante, estas diferencias de retardo se mantienen dentro de los márgenes aceptables para comunicaciones en tiempo real exitosas [10]. Posteriormente, se realizó una condición de test con 15 clientes (5+5+5), la cual también mostró un rendimiento estable y consistente de las SFUs en términos de transmisión y recepción de ancho de banda. Los valores de latencia fueron similares a los observados con 12 usuarios (véase Figura 7). Finalmente, se ejecutó una condición de test con 18 clientes (6+6+6), la cual evidenció limitaciones de rendimiento tanto en la gestión del ancho de banda por parte de las SFUs como en los niveles de latencia experimentados por los clientes.

Así pues, se confirma que el número de clientes conectados puede incrementarse mediante el despliegue de más SFUs en distintas regiones geográficas, y que los niveles de latencia asimétricos experimentados siguen siendo aceptables en este tipo de escenarios.

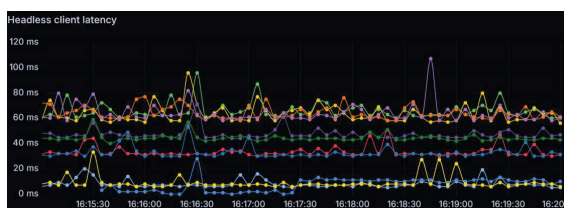


Figura 7: Resultados para el Test 3.3 con 12 clientes

V. CONCLUSIONES

En este trabajo se ha evolucionado una plataforma de comunicaciones holográficas multiusuario pionera con nuevas contribuciones que contribuyen a mejorar su rendimiento en cuanto a escalabilidad, estabilidad, retardos y estabilidad, mediante la adopción adaptativa de arquitecturas modulares asistidas por computación en el borde y APIs de red. Las pruebas de validación realizadas aportan evidencias claras sobre los beneficios aportados, tanto por la posibilidad de desplegar y seleccionar módulos de comunicaciones en los servidores de interés (ej. minimizar retardos, garantizar recursos necesarios...), como por la posibilidad de instanciar nuevos módulos de comunicaciones en diferentes servidores para conseguir un mejor balanceo de carga y mayor escalabilidad por cada sesión activa.

VI. TRABAJOS FUTUROS

En primer lugar, se pretende extender las contribuciones de este trabajo mediante la integración de nuevos mecanismos de distribución multimedia basado en posiciones relativas y campos de visión de cada usuario. En segundo lugar, se pretende incorporar servicios de transcodificación virtualizados, instanciables bajo demanda para cada flujo de vídeo volumétrico entrante y/o cada cliente objetivo. Estos dos nuevos habilitadores tecnológicos permitirán incrementar la adaptabilidad y escalabilidad de los servicios de comunicaciones holográficas, de una manera más ligera, modular y escalable que cuando se adaptan estrategias de transcodificación en la nube, mediante el uso de MCUs monolíticas desplegadas para cada sesión. Finalmente, se tratará de explotar el potencial de APIs de red, siendo especificadas en el seno de iniciativas de estandarización como CAMARA (<https://camaraproject.org/>).

AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por: el Programa Horizonte Europa de la Unión Europea, bajo el acuerdo nº 101135637 (proyecto PRESENCE), y por el Programa Smart Networks and Services Joint Undertaking (SNS JU), bajo el acuerdo nº 101096838 (proyecto 6G-XR) y nº 101139257 (proyecto SUNRISE-6G); por la Agencia Estatal de Investigación (AEI), en el marco de Proyectos Generación de Conocimiento 2021 (PID2021-126551OB-C21) y 2022 (PID2022-140749OB-I00 y PID2022-137329OB-C41, financiados por MICIU/AEI/10.13039/501100011033 y FEDER, UE); y por el Ministerio para la Transformación Digital y la de Función Pública, en el marco del programa UNICO 5G I+D 2021, bajo el proyecto con Ref. TSI-063000-2021-4 (6G-OPENVERSO-HOLO). El trabajo de Mario Montagud ha sido financiado por MCIN/AEI/10.13039/501100011033, bajo la subvención RYC2020-030679-I.

REFERENCIAS

- [1] M. Montagud, J. Li, G. Cernigliaro, A. El Ali, S. Fernández, and P. Cesar, "Towards socialvr: evaluating a novel technology for watching videos together," *Virtual Real.*, vol. 26, no. 4, p. 1593–1613, Dec. 2022. [Online]. Available: <https://doi.org/10.1007/s10055-022-00651-5>
- [2] S. F. Langa, M. Montagud, G. Cernigliaro, and D. R. Rivera, "Multiparty holomeetings: Toward a new era of low-cost volumetric holographic meetings in virtual reality," *IEEE Access*, vol. 10, pp. 81 856–81 876, 2022.
- [3] Y. Jin, K. Hu, J. Liu, F. Wang, and X. Liu, "From capture to display: A survey on volumetric video," 2023. [Online]. Available: <https://arxiv.org/abs/2309.05658>
- [4] J. van der Hooft, H. Amirpour, M. T. Vega, Y. Sanchez, R. Schatz, T. Schierl, and C. Timmerer, "A tutorial on immersive video delivery: From omnidirectional video to holography," *IEEE Commun. Surv. Tutor.*, vol. 25, no. 2, pp. 1336–1375, 2023.
- [5] S. N. B. Gunkel, H. M. Stokking, N. Prins, Martin Jand van der Stap, F. B. T. Haar, and O. A. Niamut, "Virtual reality conferencing," in *Proceedings of the 9th ACM Multimedia Systems Conference*. New York, NY, USA: ACM, Jun. 2018.
- [6] J. Park, P. A. Chou, and J.-N. Hwang, "Rate-utility optimized streaming of volumetric media for augmented reality," *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 9, no. 1, pp. 149–162, Mar. 2019.

- [7] S. Subramanyam, I. Viola, A. Hanjalic, and P. Cesar, "User centered adaptive streaming of dynamic point clouds with low complexity tiling," in *Proceedings of the 28th ACM International Conference on Multimedia*. New York, NY, USA: ACM, Oct. 2020.
- [8] S. N. B. Gunkel, R. Hindriks, K. M. E. Assal, H. M. Stokking, S. Dijkstra-Soudarissanane, F. t. Haar, and O. Niamut, "Vrcomm: an end-to-end web system for real-time photorealistic social vr communication," in *Proceedings of the 12th Multimedia Systems Conference*, ser. MMSys '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 65–79. [Online]. Available: <https://doi.org/10.1145/3458305.3459595>
- [9] G. Cernigliaro, M. Martos, M. Montagud, A. Ansari, and S. Fernandez, "Pc-mcu: point cloud multipoint control unit for multi-user holoconferencing systems," in *Proceedings of the 30th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, ser. NOSSDAV '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 47–53. [Online]. Available: <https://doi.org/10.1145/3386290.3396936>
- [10] S. Fernandez, M. Montagud, D. Rincón, J. Moragues, and G. Cernigliaro, "Addressing scalability for real-time multiuser holoportation: Introducing and assessing a multipoint control unit (mcu) for volumetric video," in *Proceedings of the 31st ACM International Conference on Multimedia*, ser. MM '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 9243–9251. [Online]. Available: <https://doi.org/10.1145/3581783.3613777>
- [11] M. Dasari, E. Lu, M. W. Farb, N. Pereira, I. Liang, and A. Rowe, "Scaling VR video conferencing," in *2023 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*. IEEE, Mar. 2023.
- [12] Y. Huang, H. Wang, X. Qiao, X. Su, Y. Li, S. Dustdar, and P. Zhang, "SCAXR: Empowering scalable Multi-User interaction for heterogeneous XR devices," *IEEE Netw.*, vol. 38, no. 4, pp. 250–258, Jul. 2024.
- [13] I. Yeregui, D. Mejías, G. Pacho, R. Viola, J. Astorga, and M. Montagud, "Edge rendering architecture for multiuser XR experiences and E2E performance assessment," in *2024 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*. IEEE, Jun. 2024, pp. 1–7.
- [14] T. Theodoropoulos, A. Makris, A. Boudi, T. Taleb, U. Herzog, L. Rosa, L. Cordeiro, K. Tserpes, E. Spatafora, A. Romussi, E. Zschau, M. Kamarianakis, A. Protopsaltis, G. Papagiannakis, and P. Dazzi, "Cloud-based XR services: A survey on relevant challenges and enabling technologies," *J-NaNA*, vol. 2, no. 1, pp. 1–22, 2022.
- [15] MongoDB, "Mongodb," last Access in: August 2024. [Online]. Available: <https://www.mongodb.com/>
- [16] e. a. M. Montagud, "Awarexr: A naas architecture to enhance xr services over beyond 5g networks," , IEEE Network, To Appear in 2025.
- [17] nodejs.org, "Node.js," last Access in: August 2024. [Online]. Available: <https://nodejs.org/en>
- [18] Prometheus, "Prometheus," last Access in: August 2024. [Online]. Available: <https://prometheus.io/>
- [19] Grafana, "Grafana," last Access in: August 2024. [Online]. Available: <https://grafana.com/>
- [20] Prometheus, "Node exporter," last Access in: August 2024. [Online]. Available: https://github.com/prometheus/node_exporter
- [21] Google, "Cadvisor," last Access in: August 2024. [Online]. Available: <https://github.com/google/cadvisor>
- [22] Prometheus, "Prometheus push gateway," last Access in: August 2024. [Online]. Available: <https://github.com/prometheus/pushgateway>
- [23] P. Community, "Windows exporter," last Access in: August 2024. [Online]. Available: https://github.com/prometheus-community/windows_exporter
- [24] Nvidia, "Dcgm-exporter," last Access in: August 2024. [Online]. Available: <https://github.com/NVIDIA/dcgmexporter>
- [25] Wireshark, "Wireshark," last Access in: September 2024. [Online]. Available: <https://www.wireshark.org/download.html>