



**ESCUELA DE INGENIERÍA DE FUENLABRADA**

**GRADO EN INGENIERIA EN SISTEMAS  
AUDIOVISUALES Y MULTIMEDIA**

**TRABAJO FIN DE GRADO**

**ENTORNO INMERSIVO 3D CON TECNOLOGÍAS WEB**

Autor : Jorge Luis Grande González

Tutor : Dr. Gregorio Robles

Curso académico 2023/2024



# Trabajo Fin de Grado/Máster

# Entorno Inmersivo 3D con Tecnologías Web

**Autor :** Jorge Luis Grande González

**Tutor : Dr. Gregorio Robles**

La defensa del presente Proyecto Fin de Carrera se realizó el día de 2024, siendo calificada por el siguiente tribunal:

## **Presidente:**

## **Secretario:**

## Vocal:

y habiendo obtenido la siguiente calificación:

## **Calificación:**

Fuenlabrada, a \_\_\_\_\_ de \_\_\_\_\_ de 2024



*Dedicado a  
mi madre*



# **Agradecimientos**

Agradezco haber tenido la oportunidad de estudiar esta carrera.



# Resumen

Este proyecto se centra en el desarrollo de un entorno de realidad virtual que representa un intercambio de paquetes inspirado en la práctica final de la asignatura de Protocolos para la Transmisión de Audio y Vídeo por Internet. La práctica final implica una sesión SIP para el intercambio de audio a través del protocolo RTP. El objetivo principal es construir una representación visual y funcional que permita a los usuarios experimentar de manera tangible los flujos y la gestión de las comunicaciones en tiempo real, utilizando tecnologías de realidad virtual.

Para lograr este objetivo, se ha utilizado una combinación de herramientas y tecnologías avanzadas, incluyendo la biblioteca de JavaScript Three.js para la renderización de gráficos 3D, así como varios módulos adicionales de Three.js que apoyan la interactividad en entornos VR como VRButton.js, XRCControllerModelFactory.js, Stats.js, y OrbitControls.js. El desarrollo se ha llevado a cabo dentro del marco de un proyecto educativo en la universidad, buscando proporcionar una herramienta de enseñanza para estudiantes interesados en la realidad virtual y las telecomunicaciones.

Este Trabajo de Fin de Grado se enmarca en la necesidad de representar la práctica final de PTAVI, la cual suele ser conceptualmente difícil de comprender. Mediante el uso de la realidad virtual, el proyecto no solo demuestra la viabilidad técnica de simular sistemas de comunicación en un entorno inmersivo, sino que también explora nuevas posibilidades para el aprendizaje de conceptos avanzados de telecomunicaciones. En última instancia, busca no solo ofrecer una experiencia educativa más representativa, sino también mejorar la comprensión de esta práctica final.



# Summary

This project focuses on developing a virtual reality environment that represents a packet exchange inspired by the final project of the Internet Audio and Video Transmission Protocols course. The final project involves a SIP session for audio exchange via the RTP protocol. The main objective is to build a visual and functional representation that allows users to tangibly experience the flows and management of real-time communications using virtual reality technologies.

To achieve this objective, a combination of advanced tools and technologies has been used, including the JavaScript library Three.js for 3D graphics rendering, as well as several additional Three.js modules that support interactivity in VR environments such as VRButton.js, XRControllerModelFactory.js, Stats.js, and OrbitControls.js. The development has been carried out within the framework of an educational project at the university, aiming to provide a teaching tool for students interested in virtual reality and telecommunications.

This Bachelor's Thesis is framed within the need to represent the final practice of PTAVI, which is often conceptually challenging to understand. Through the use of virtual reality, the project not only demonstrates the technical feasibility of simulating communication systems in an immersive environment but also explores new possibilities for learning advanced telecommunications concepts. Ultimately, it aims not only to provide a more representative educational experience but also to enhance understanding of this final practice.



# Índice general

<b>1. Introducción</b>	<b>1</b>
1.1. Estructura de la memoria . . . . .	3
<b>2. Objetivos</b>	<b>5</b>
2.1. Objetivo general . . . . .	5
2.2. Objetivos específicos . . . . .	5
<b>3. Estado del arte</b>	<b>7</b>
3.1. Lenguajes de marcado y programación . . . . .	8
3.2. Importaciones de bibliotecas JavaScript . . . . .	9
3.3. Herramientas de desarrollo . . . . .	11
3.4. Protocolos visualizados . . . . .	12
3.5. Herramientas de desarrollo . . . . .	12
<b>4. Diseño e implementación</b>	<b>15</b>
4.1. Arquitectura general . . . . .	15
4.2. Escena . . . . .	18
4.3. Controles de realidad virtual . . . . .	20
4.3.1. Configuración de los controladores VR . . . . .	20
4.3.2. Interacción mediante controladores VR . . . . .	21
4.3.3. Implementación técnica . . . . .	21
<b>5. Resultados</b>	<b>23</b>
5.1. Descripción de las interacciones . . . . .	23
5.1.1. Interacción con UA1 . . . . .	23

5.1.2. Interacción con el proxy . . . . .	24
5.1.3. Interacción con UA2 . . . . .	28
5.1.4. Interacción con paquetes . . . . .	31
<b>6. Conclusiones</b>	<b>45</b>
6.1. Consecución de objetivos . . . . .	45
6.1.1. Objetivo general . . . . .	45
6.2. Aplicación de lo aprendido . . . . .	45
6.3. Lecciones aprendidas . . . . .	46
6.4. Trabajos futuros . . . . .	47
<b>A. Manual de usuario</b>	<b>49</b>
A.1. Requisitos del sistema . . . . .	49
A.2. Uso de la aplicación . . . . .	49
A.2.1. Instalación de la extensión WebXR API Emulator . . . . .	49
A.2.2. Activación de la realidad virtual . . . . .	50
A.2.3. Interacción en la aplicación . . . . .	50
A.2.4. Activación de la secuencia de intercambio de paquetes . . . . .	50
A.3. Modo sin realidad virtual . . . . .	51
<b>Bibliografía</b>	<b>53</b>

# Índice de figuras

1.1.	Secuencia del intercambio de paquetes . . . . .	2
3.1.	Porcentaje de lenguajes usados. . . . .	9
4.1.	Estructura del proyecto . . . . .	18
4.2.	Escena del proyecto . . . . .	20
4.3.	Controlador VR . . . . .	21
4.4.	Botones controlador VR . . . . .	22
5.1.	Estado inicial del UA1 . . . . .	25
5.2.	Atributos UA1 con valores específicos . . . . .	26
5.3.	Estado inicial del Proxy . . . . .	27
5.4.	Atributos Proxy con valores específicos . . . . .	28
5.5.	Estado inicial del UA2 . . . . .	29
5.6.	Atributos UA2 con valores específicos . . . . .	30
5.7.	Secuencia del intercambio de paquetes . . . . .	32
5.8.	Mensaje de registro inicial de UA1 . . . . .	33
5.9.	Respuesta 401 Unauthorized a UA1 . . . . .	34
5.10.	Registro de UA1 exitoso con 200 OK . . . . .	35
5.11.	Mensaje de registro inicial de UA2 . . . . .	36
5.12.	Registro de UA2 exitoso con 200 OK . . . . .	37
5.13.	Inicio de llamada con el mensaje INVITE de UA1 . . . . .	38
5.14.	Respuesta TRYING de UA2 indicando procesamiento . . . . .	39
5.15.	Alerta de llamada con el mensaje RINGING de UA2 . . . . .	39
5.16.	Confirmación de llamada con 200 OK + SDP de UA2 . . . . .	40

5.17. Reconocimiento ACK de UA1 completando el establecimiento de la llamada . . . . .	41
5.18. Transmisión de datos multimedia a través de paquetes RTP . . . . .	42
5.19. Mensaje BYE enviado para terminar la llamada . . . . .	43
5.20. Confirmación del fin de la llamada con un 200 OK . . . . .	44

# **Capítulo 1**

## **Introducción**

En la era de la transformación digital, las comunicaciones interactivas han evolucionado significativamente, incorporando tecnologías inmersivas como la realidad virtual (RV) para ofrecer experiencias más enriquecedoras y colaborativas.

Este Trabajo de Fin de Grado se enfoca en aclarar de manera visual, utilizando un entorno de realidad virtual, la práctica final de la asignatura de Protocolos para la Transmisión de Audio y Vídeo por Internet. Esta práctica implica una sesión SIP para el intercambio de audio a través del protocolo RTP, como se representa en la figura 1.1.

La motivación detrás de este proyecto surge de la necesidad de comprender mejor las complejidades de la señalización y el manejo de medios en las comunicaciones actuales, y cómo estas pueden ser simuladas y visualizadas en un entorno de realidad virtual. A través de la creación de un modelo interactivo, este trabajo pretende proporcionar una herramienta educativa y un medio de investigación que permita a los usuarios experimentar de manera tangible el flujo y la gestión de los datos de comunicación.

Con el uso de la realidad virtual, se espera no solo proporcionar una comprensión más profunda de estos protocolos sino también explorar las posibilidades que la realidad virtual tiene para ofrecer en términos de aprendizaje interactivo, simulación y comprensión.

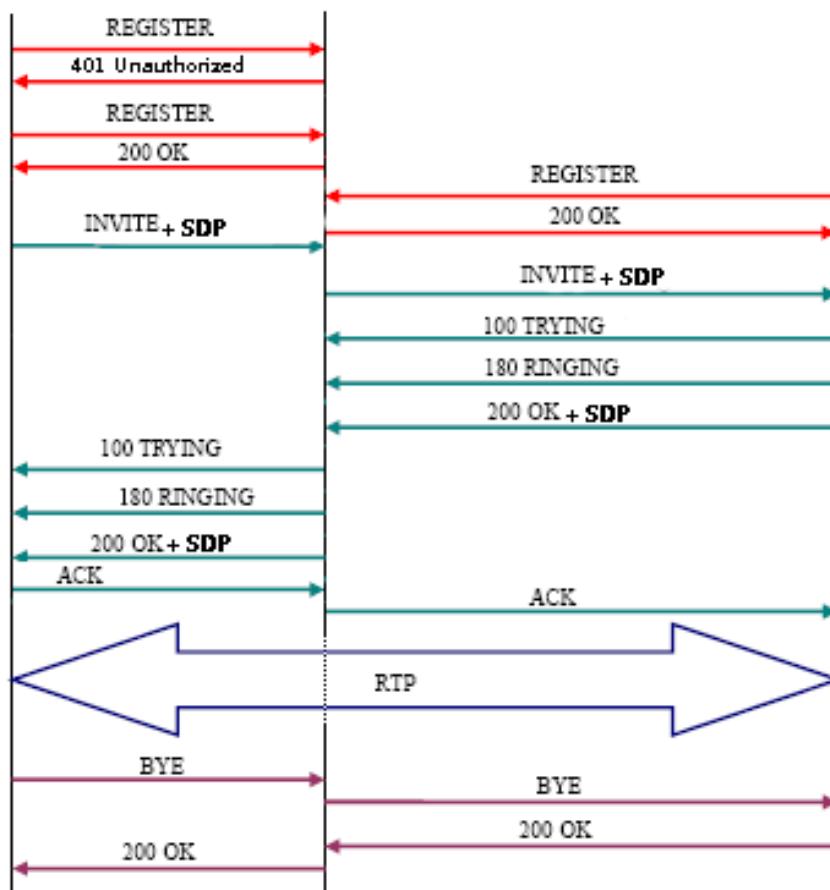


Figura 1.1: Secuencia del intercambio de paquetes

## 1.1. Estructura de la memoria

Esta sección describe la estructura organizativa de la memoria, proporcionando una visión general de cada uno de los capítulos principales que componen este trabajo de fin de grado. La estructura está diseñada para desarrollar gradualmente el tema, desde la introducción y los fundamentos teóricos hasta la implementación práctica y la evaluación de los resultados.

- **Capítulo 1: Introducción** - Se presenta una introducción general al proyecto, estableciendo el contexto y la motivación del estudio.
- **Capítulo 2: Objetivos** - Detalla los objetivos generales y específicos que guían el desarrollo del proyecto.
- **Capítulo 3: Estado del Arte** - Se comentan todas las tecnologías, herramientas y accesorios necesarios para el desarrollo y funcionamiento de este proyecto.
- **Capítulo 4: Diseño e implementación** - Describe los métodos utilizados para desarrollar el sistema de comunicación, incluyendo el diseño, las herramientas y tecnologías empleadas.
- **Capítulo 5: Resultados** - Presenta los resultados obtenidos a través de la implementación del sistema.
- **Capítulo 6: Conclusiones** - Ofrece un desarrollo de las conclusiones extraídas y las recomendaciones para investigaciones futuras.

Cada capítulo está diseñado para construir sobre la información presentada en los anteriores, asegurando un flujo lógico y coherente a lo largo de la memoria.



# **Capítulo 2**

## **Objetivos**

### **2.1. Objetivo general**

El objetivo principal es construir una representación visual y funcional de los flujos de señalización y datos utilizando la biblioteca de JavaScript Three.js, junto con tecnologías de RV para facilitar la interacción con la simulación. Se busca demostrar cómo se inicia, se mantiene y se termina una sesión de comunicación, siguiendo los estándares del protocolo SIP, y cómo los paquetes RTP son utilizados para el intercambio efectivo de datos multimedia.

### **2.2. Objetivos específicos**

Para cumplir con el objetivo general de desarrollar una representación visual y funcional de los flujos de señalización y datos en un entorno de realidad virtual, se establecen los siguientes objetivos específicos:

1. Diseñar e implementar un modelo virtual en 3D que simule el entorno de una red de comunicaciones, empleando la biblioteca Three.js.
2. Integrar tecnologías de realidad virtual para facilitar una experiencia inmersiva y la interacción del usuario con el modelo de simulación.
3. Desarrollar una interfaz de usuario que permita la visualización y manipulación de los procesos de señalización y transmisión de datos, acorde con el protocolo SIP.

4. Implementar la lógica para simular el flujo de mensajes SIP, incluyendo el registro de usuarios, la iniciación y recepción de llamadas.
5. Simular la transmisión de datos multimedia a través de paquetes RTP, mostrando la secuencia de envío y recepción en tiempo real.
6. Crear un sistema de retroalimentación visual que indique el estado de las comunicaciones, incluyendo errores y mensajes de confirmación.

# **Capítulo 3**

## **Estado del arte**

En este capítulo se explicarán todas las tecnologías, herramientas y accesorios necesarios para el desarrollo, lo cual es crucial para comprender los fundamentos que hacen posible nuestros marcos de desarrollo. Todo esto se debe a avances significativos en tecnologías como WebVR. Hasta hace poco, las experiencias de VR en la web eran limitadas debido a las restricciones de ancho de banda para la transmisión de datos. Sin embargo, esta situación ha cambiado drásticamente. Proyectos como Google Stadia o Steam VR ejemplifican cómo la ejecución de experiencias de VR desde servidores remotos ha transformado la forma en que interactuamos con la realidad virtual en la web.

WebVR está compuesta por librerías de JavaScript que garantizan la compatibilidad con los navegadores. Además, cada navegador puede implementar modificaciones para adaptarse a sus particularidades.

Una de las fortalezas de WebVR es su naturaleza de código abierto, lo que hace que su desarrollo sea accesible para todos.

Además, WebVR realiza un seguimiento del rendimiento, ya que la realidad virtual puede causar malestar si la tasa de fotogramas es baja. En dispositivos móviles, la frecuencia de actualización es de 60Hz y 60fps, mientras que en computadoras es de 90Hz.

En términos de accesibilidad, WebVR se destaca, ya que no se requieren habilidades técnicas avanzadas para su uso. Con solo tener un ordenador, un navegador compatible y hardware de soporte (como gafas VR), puedes disfrutar de una experiencia de VR con solo un clic.

En cuanto a la compatibilidad, WebVR es compatible con todos los navegadores modernos (Chrome, Firefox, Safari, Edge), y también con la realidad aumentada y mixta.

Finalmente, WebVR se integra con la nube, lo que es una ventaja cada vez más demandada en la actualidad.

### 3.1. Lenguajes de marcado y programación

- **HTML:** Lenguaje estándar para crear páginas web, define la estructura y el contenido utilizando etiquetas <sup>1</sup>.
  - Funcionalidades clave:
    - Definición de la estructura de una página web mediante elementos como encabezados, párrafos, listas, etc.
    - Incorporación de contenido multimedia, como imágenes, videos y audio.
    - Creación de formularios interactivos para la recopilación de datos.
- **JavaScript:** Lenguaje de programación utilizado en desarrollo web para agregar interactividad y dinamismo a las páginas <sup>2</sup>.
  - Funcionalidades clave:
    - Manipulación y modificación del contenido HTML y CSS en tiempo real.
    - Gestión de eventos del navegador, como clics de ratón, pulsaciones de teclas y desplazamientos.
    - Comunicación asincrónica con el servidor mediante peticiones AJAX.
- **TeX:** Sistema de composición tipográfica utilizado principalmente para la creación de documentos científicos y técnicos de calidad <sup>3</sup>.
  - Funcionalidades clave:

---

<sup>1</sup><https://developer.mozilla.org/es/docs/Web/HTML>

<sup>2</sup><https://developer.mozilla.org/en-US/docs/Web/JavaScript>

<sup>3</sup><https://www.latex-project.org/>



Figura 3.1: Porcentaje de lenguajes usados.

- Tipografía de alta calidad para la composición de documentos científicos, libros y artículos académicos.
- Soporte avanzado para notación matemática y fórmulas complejas.
- Personalización detallada de la apariencia del documento mediante la definición de estilos y formatos.

## 3.2. Importaciones de bibliotecas JavaScript

En esta sección, se detallan las bibliotecas JavaScript utilizadas en el proyecto para facilitar la creación y renderización de gráficos en 3D, así como la integración de funcionalidades de realidad virtual (VR) y realidad extendida (XR).

### ■ THREE.js:

- **Descripción:** THREE.js es una biblioteca de código abierto que proporciona una amplia gama de herramientas y funciones para trabajar con gráficos en 3D en el navegador web. Permite la creación y manipulación de escenas 3D, modelos tridimensionales, materiales, luces y cámaras [10].

- **Funcionalidades clave:**

- Renderización de gráficos 3D en tiempo real.
- Creación y manipulación de modelos tridimensionales.
- Implementación de efectos visuales y materiales realistas.
- Gestión de luces y sombras en la escena.
- Control de la cámara para la navegación en entornos 3D.

### ■ VRButton.js:

- **Descripción:** VRButton.js es un módulo de THREE.js diseñado específicamente para simplificar la integración de funcionalidades de realidad virtual en aplicaciones web. Proporciona una interfaz para la activación y desactivación de la visualización en VR, así como para la detección de dispositivos de VR compatibles [9].

- **Funcionalidades clave:**

- Integración de botones para activar/desactivar la visualización en VR.
- Detección automática de dispositivos de realidad virtual compatibles.
- Facilita la creación de experiencias inmersivas en la web.

- **XRCControllerModelFactory.js:**

- **Descripción:** XRCControllerModelFactory.js es otro módulo de THREE.js que ofrece una fábrica para la creación de modelos de controladores de realidad extendida (XR). Estos modelos son representaciones visuales de los controladores utilizados por los usuarios dentro del entorno virtual y pueden personalizarse según las necesidades del proyecto [11].

- **Funcionalidades clave:**

- Creación de modelos de controladores XR para representar dispositivos de VR/AR.
- Personalización de la apariencia y comportamiento de los modelos de controladores.
- Integración fluida con aplicaciones de realidad extendida.

- **Stats.js:**

- **Descripción:** Stats.js es un módulo de THREE.js que proporciona una herramienta para monitorear el rendimiento de aplicaciones web 3D en tiempo real. Permite realizar un seguimiento de métricas importantes, como los cuadros por segundo (FPS), el uso de memoria y el tiempo de procesamiento de cuadros [12].

- **Funcionalidades clave:**

- Monitoreo del rendimiento en tiempo real de aplicaciones web 3D.
- Visualización de métricas como FPS y uso de memoria.

- Ayuda en la identificación y solución de cuellos de botella de rendimiento.

- **OrbitControls.js:**

- **Descripción:** OrbitControls.js es un módulo de THREE.js que proporciona controles de órbita para permitir al usuario manipular la cámara en una escena 3D de manera interactiva. Permite rotar, acercar y alejar la vista de la escena, lo que mejora la experiencia de navegación [8].

- **Funcionalidades clave:**

- Control de la cámara para la navegación interactiva en entornos 3D.
- Rotación, acercamiento y alejamiento suaves de la cámara.
- Personalización de la sensibilidad y restricciones de los controles.

### 3.3. Herramientas de desarrollo

- **FireFox:** Navegador web de código abierto conocido por su enfoque en la privacidad y la personalización. Destaca por su capacidad para bloquear rastreadores de anuncios y proteger la privacidad del usuario. Además, ofrece una amplia gama de complementos y extensiones que permiten personalizar la experiencia de navegación según las preferencias del usuario [7].
- **VisualStudioCode:** Editor de código fuente de Microsoft altamente personalizable y de código abierto, conocido por su rendimiento y amplia gama de extensiones. Es ampliamente utilizado por desarrolladores de software debido a su interfaz intuitiva, funcionalidades avanzadas de edición de código y soporte para una variedad de lenguajes de programación y tecnologías [6].
- **GitHub:** Plataforma de desarrollo colaborativo de software basada en la nube, que permite a los desarrolladores alojar, revisar, colaborar y desplegar proyectos de software, incluidas aplicaciones web, de manera eficiente y transparente. Es ampliamente utilizado en la comunidad de desarrollo de software para gestionar versiones de código, coordinar el trabajo en equipo y facilitar la contribución de la comunidad al código abierto [4].

- **Oculus Quest 2:** Gafas de realidad virtual autónomas producidas por Oculus (una subsidiaria de Facebook), conocidas por su alta calidad y facilidad de uso. Estas gafas fueron proporcionadas por la universidad para el desarrollo del proyecto. Ofrecen una experiencia inmersiva de realidad virtual sin necesidad de estar conectadas a un PC o consola, lo que las hace ideales para el desarrollo y la experiencia de usuario en entornos de realidad virtual [14].
- **Web XR API Emulator:** Herramienta de desarrollo que permite probar y depurar experiencias de realidad virtual y aumentada basadas en WebXR directamente en el navegador web. Facilita a los desarrolladores la creación y prueba de contenido de realidad virtual y aumentada sin necesidad de hardware especializado, lo que agiliza el proceso de desarrollo y depuración de aplicaciones basadas en WebXR [15].

### 3.4. Protocolos visualizados

### 3.5. Herramientas de desarrollo

En esta sección, se detallan los protocolos utilizados en el desarrollo del proyecto, los cuales son esenciales para la comunicación en tiempo real y la transmisión de datos multimedia.

- **SIP (Session Initiation Protocol):**

- **Descripción:** SIP es un protocolo de señalización utilizado para iniciar, mantener y finalizar sesiones de comunicación en tiempo real, como llamadas de voz y video. Es un protocolo de capa de aplicación que puede gestionar la creación, modificación y finalización de sesiones con uno o más participantes [2].
- **Funcionalidades clave:**
  - Establecimiento y finalización de llamadas de voz y video.
  - Modificación de sesiones en tiempo real añadiendo o eliminando participantes.
  - Interoperabilidad con otros protocolos de comunicación.
  - Soporte para autenticación y cifrado de comunicaciones.

- **RTP (Real-time Transport Protocol):**

- **Descripción:** RTP es un protocolo de red diseñado para la entrega de datos en tiempo real, como audio y video, a través de redes IP. Es ampliamente utilizado en aplicaciones que requieren transmisión continua de datos multimedia [1].

- **Funcionalidades clave:**

- Transmisión de datos en tiempo real (audio y video).
- Sincronización de flujos multimedia.
- Detección y corrección de pérdidas de paquetes.
- Extensibilidad para soportar nuevos códigos y funcionalidades.

- **SDP (Session Description Protocol):**

- **Descripción:** SDP es un formato estándar para describir los detalles de las sesiones multimedia con el objetivo de iniciar y gestionar comunicaciones. Este protocolo es utilizado comúnmente junto con SIP para negociar los parámetros de la sesión, como los códigos a utilizar, las direcciones IP y los puertos [3].

- **Funcionalidades clave:**

- Descripción de los parámetros de la sesión multimedia.
- Negociación de capacidades de comunicación entre los participantes.
- Integración con protocolos de señalización como SIP.
- Formato flexible para adaptarse a diferentes configuraciones de red.



# Capítulo 4

## Diseño e implementación

En este capítulo se detalla el desarrollo del proyecto, incluyendo su estructura, los componentes en el entorno inmersivo, su funcionamiento, y las interacciones implementadas.

### 4.1. Arquitectura general

El proyecto se estructura en cinco componentes enlazados entre sí, representados en la figura 4.1. Estos componentes serán explicados a continuación:

- **index.html:** Representamos la estructura básica de una página web para una experiencia de realidad virtual (VR). Se define un documento HTML con metadatos y referencias a archivos externos, incluyendo la biblioteca Three.js para gráficos 3D.

Además, se importa un módulo de JavaScript que inicializa la aplicación VR, creando una instancia de la clase App y asignándola a la ventana del navegador.

Este archivo proporciona una base para el desarrollo de aplicaciones de realidad virtual en la web.

- **app.js:** Este script es una aplicación que utiliza la biblioteca Three.js para crear una escena 3D interactiva. Que realiza:

1. Importa las bibliotecas necesarias de Three.js para crear la escena.
2. Carga varias texturas de imágenes para aplicarlas a los materiales de los objetos en la escena.

3. Define algunas variables y objetos necesarios para el funcionamiento de la aplicación.
  4. Crea una clase App que inicializa la escena, la cámara, la iluminación y los controles.
  5. Dentro de la clase App, hay métodos para manejar el control de los dispositivos de entrada de VR, como los controladores y sus eventos.
  6. Contiene un método para construir los modelos de los controladores de VR.
  7. Otro método se encarga de manejar la lógica de la animación y la interacción del usuario con la escena.
  8. Y finalmente, tenemos un método de renderizado que se ejecuta en bucle y actualiza la escena en cada fotograma.
- **imágenes:** En la arquitectura del proyecto, tenemos una carpeta llamada imágenes que contiene todas las imágenes utilizadas en la aplicación. Dentro de esta carpeta, tenemos subcarpetas para organizar las imágenes de acuerdo con su propósito.
- Cada imagen se carga en la aplicación utilizando el constructor THREE.TextureLoader().load(), proporcionando la ruta relativa desde el punto donde se está ejecutando el código hacia la ubicación de la imagen en la estructura de carpetas.
- **sceneObjects:** Este archivo es un módulo de funciones que proporciona diferentes objetos tridimensionales (3D) utilizando la biblioteca Three.js. Cada función devuelve un objeto con geometría y material específicos.

1. **Sphere:** Devuelve una esfera roja con una geometría esférica y un material básico.
2. **Side:** Devuelve un objeto rectangular con una textura proporcionada, usado como pared en la escena.
3. **Box:** Devuelve una caja con una textura proporcionada, usado como UA y Proxy en la escena.
4. **Wall:** Devuelve un plano grande con una textura proporcionada, usado como pantalla en la escena.
5. **Logo:** Devuelve un plano con una textura proporcionada, representando logotipo.

6. **Floor:** Devuelve un plano grande con una textura proporcionada, usado para el suelo de la escena.

Cada función utiliza diferentes geometrías y materiales para crear los objetos 3D. También configura propiedades específicas de los materiales, como envoltura de textura y repetición.

- **jsm:** Esta carpeta contiene módulos JavaScript utilizados en el proyecto. Estos módulos son archivos independientes que proporcionan funcionalidades específicas para la aplicación. Aquí está la estructura de la carpeta:

1. **build:** En esta subcarpeta se encuentran los archivos de construcción de la biblioteca Three.js, que proporcionan la funcionalidad principal de renderización en 3D. El archivo principal utilizado es three.module.js, que importa todas las funcionalidades esenciales de Three.js.
2. **webxr:** Aquí están los módulos relacionados con WebXR, una API que permite crear experiencias de realidad virtual (VR) y realidad aumentada (AR) en el navegador. Por ejemplo, VRButton.js proporciona un botón para activar la experiencia de realidad virtual, mientras que XRControllerModelFactory.js permite crear modelos visuales para los controladores de VR.
3. **libs:** Esta subcarpeta contiene bibliotecas adicionales utilizadas en el proyecto. En particular, stats.module.js proporciona una herramienta para medir el rendimiento de la aplicación mediante la visualización de estadísticas, como el número de fotogramas por segundo.
4. **controls:** Aquí están los módulos relacionados con el control de la cámara y la escena en la aplicación. Por ejemplo, OrbitControls.js proporciona controles para permitir al usuario mover y rotar la cámara alrededor de la escena, lo que facilita la navegación en entornos 3D.

Por lo tanto, la carpeta jsm organiza los módulos JavaScript utilizados en el proyecto, proporcionando funcionalidades esenciales, soporte para WebXR, herramientas de medición de rendimiento y controles de cámara. Estos módulos se importan en la aplicación según sea necesario para agregar las características deseadas.

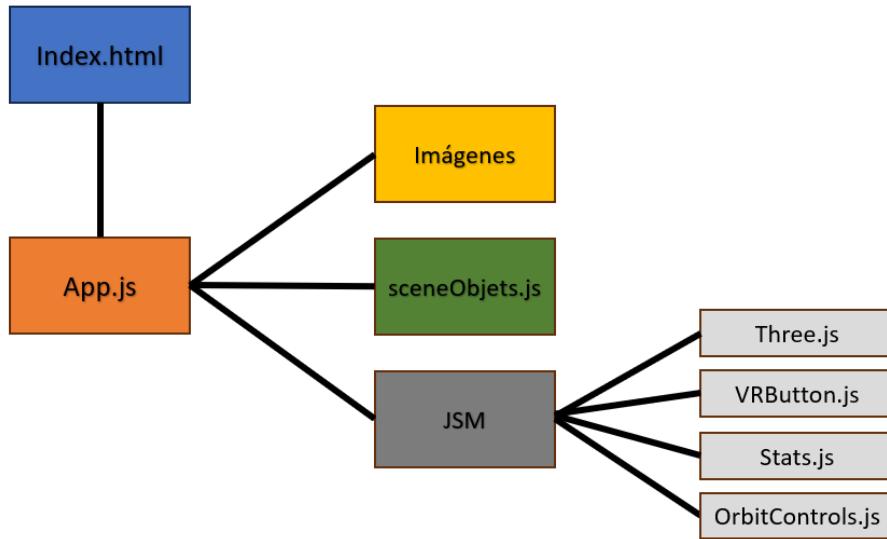


Figura 4.1: Estructura del proyecto

## 4.2. Escena

La escena virtual implementada se desarrolla utilizando la biblioteca Three.js, que permite la creación de gráficos 3D en un entorno web. Este proyecto configura un entorno interactivo que simula un sistema de comunicación, donde se pueden visualizar los intercambios de mensajes entre diferentes usuarios y un proxy. Esta escena es representada en la figura 4.2.

### Estructura básica

- **Cámara y Controles:** La cámara se configura con una perspectiva que permite al usuario tener una vista adecuada del entorno virtual. Se utiliza OrbitControls para permitir al usuario interactuar con la vista, rotando y acercando/alejando la escena.
- **Iluminación:** Se añade iluminación hemisférica para simular una luz suave y direccional para destacar objetos específicos dentro de la escena.
- **Renderizado:** Se establece un renderizado con antialiasing para mejorar la calidad visual de los bordes de los objetos en la escena.

### Objetos en la escena

- **Suelo y paredes:** Se utilizan texturas cargadas para crear un suelo y paredes que encierran la escena, proporcionando un fondo estático.

- **Objetos interactivos:** Se crean varias cajas y esferas que representan a diferentes usuarios (UA1, UA2) y un proxy. Estos objetos se pueden interactuar mediante eventos controlados por los controladores de realidad virtual.
- **Indicadores visuales:** Se utilizan logos y texturas para indicar estados como start y stop, mostrando visualmente el flujo de la simulación.

### Interacción y dinámica

- **Inicio de eventos:** La interacción comienza cuando el usuario activa controles específicos en los controladores de realidad virtual. Esto puede alterar el estado de los objetos (por ejemplo, cambiar el color para indicar actividad) y cambiar texturas que representan diferentes mensajes enviados y recibidos en la comunicación.
- **Simulación de mensajes:** Los objetos esféricos se mueven entre las cajas para simular el envío de mensajes. La ruta y la dirección de estos objetos dependen de las interacciones del usuario y el flujo del protocolo simulado.
- **Animación de paquetes RTP:** En ciertos puntos, esferas adicionales se mueven para simular la transmisión de paquetes RTP, indicando el intercambio de media en una llamada establecida.

### Implementación

- **Controladores XR:** Se configuran controladores para manejar la interacción en un entorno de realidad extendida (XR). Esto incluye la gestión de eventos de selección y conexión de dispositivos.
- **Animación y actualización de la escena:** La animación de la escena se inicia con los controladores, simulando el intercambio de paquetes mediante objetos esféricos. Estos objetos siguen una ruta y dirección determinadas por el estado actual del sistema, representando de manera visual el flujo del protocolo de comunicación.
- **Animación de paquetes RTP:** La lógica para mover objetos y actualizar estados se ejecuta dentro de un bucle de animación que recalcula posiciones y estados basados en la interacción del usuario.

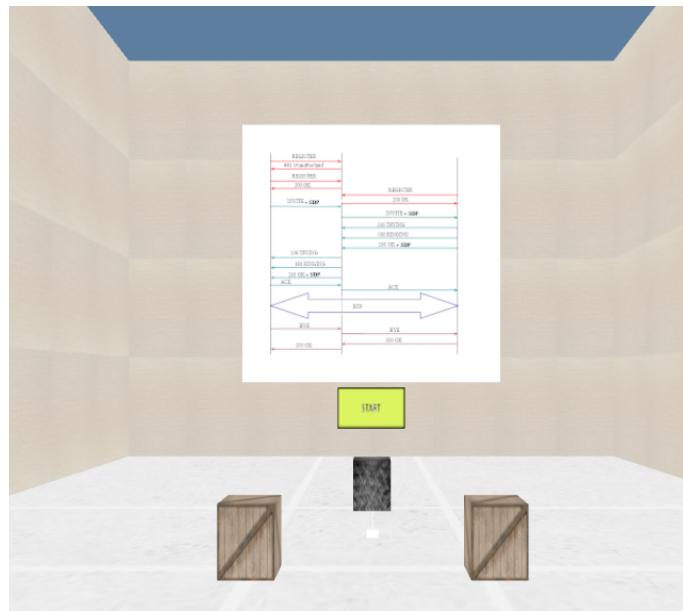


Figura 4.2: Escena del proyecto

#### **4.3. Controles de realidad virtual**

La implementación de controles de realidad virtual en la aplicación proporciona una interfaz interactiva que mejora significativamente la experiencia del usuario, permitiendo una manipulación intuitiva y directa de la escena virtual.

#### **4.3.1. Configuración de los controladores VR**

Los controladores VR son dispositivos físicos que los usuarios sostienen en sus manos y que detectan sus movimientos y gestos en el espacio tridimensional. Estos están equipados con una variedad de sensores y botones que permiten una gama amplia de interacciones:

- **Botones y Gatillos:** Utilizados para realizar selecciones y activar eventos dentro de la aplicación. Por ejemplo, un usuario puede iniciar la transmisión de mensajes al presionar el botón Start.
  - **Sensores de Movimiento:** Capturan el posicionamiento y la orientación de las manos del usuario, permitiendo manipular objetos o navegar por la escena con movimientos naturales.

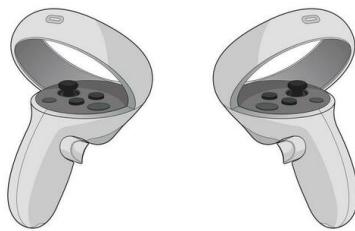


Figura 4.3: Controlador VR

#### 4.3.2. Interacción mediante controladores VR

La interacción con la escena se realiza a través de los controladores VR de la siguiente manera:

- **Selección y Manipulación:** Los usuarios apuntan a objetos virtuales con los controladores y utilizan botones para seleccionarlos. Esto puede incluir activar elementos dentro de la escena.
- **Navegación:** Mediante el uso del gatillo en los controladores, los usuarios pueden desplazarse por la escena virtual, acercándose o alejándose de los objetos o desplazándose lateralmente.
- **Interacción Contextual:** La aplicación cambia la funcionalidad de la escena basándose en el estado del objeto con el que el usuario está interactuando, cambiando modos de visualización o ajustando parámetros específicos de los objetos.

#### 4.3.3. Implementación técnica

La implementación técnica de los controladores VR en la aplicación utiliza la API de WebXR, integrada con Three.js, para gestionar la entrada de los dispositivos de realidad virtual. El código configura cada controlador para responder a eventos como ‘selectstart’ y ‘selectend’, lo que permite detectar interacciones como pulsaciones de botones y liberaciones. Adicionalmente, se emplean rayos virtuales (‘raycasting’) para determinar qué objetos están siendo apuntados por los controladores, facilitando así una interacción precisa [5].

En la implementación de la aplicación, hacemos un uso específico de los botones trigger de los controladores VR para facilitar una interacción intuitiva y efectiva dentro del entorno

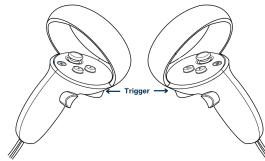


Figura 4.4: Botones controlador VR

virtual. Cada trigger tiene un propósito bien definido que mejora la experiencia del usuario y la funcionalidad de la aplicación:

- **Trigger del controlador izquierdo:** Este botón se utiliza para la navegación dentro de la escena. Al presionar el trigger del controlador izquierdo, el usuario puede desplazarse hacia donde esté mirando con las gafas de realidad virtual, lo que permite una navegación intuitiva y natural. Esta funcionalidad es fundamental para explorar diferentes áreas de la escena de manera fluida.
- **Trigger del controlador derecho:** El trigger del controlador derecho está configurado para interactuar con objetos específicos de la escena. Cuando se presiona este botón, se puede seleccionar o activar elementos dentro de la escena, como iniciar una simulación, modificar parámetros de un objeto, o ejecutar acciones específicas relacionadas con los objetos con los que se interactúa. Esta interacción es esencial para manipular elementos de la aplicación y para participar activamente en las simulaciones y demostraciones que se presentan en el entorno virtual.

La programación de estos triggers se realiza a través del manejo de eventos en JavaScript, utilizando la API de WebXR para detectar y responder a las acciones del usuario. Este diseño permite un control preciso sobre la aplicación, ofreciendo a los usuarios una manera clara y directa de interactuar con la interfaz y con los elementos virtuales de la escena.

# **Capítulo 5**

## **Resultados**

En este capítulo se presentan los resultados derivados de las interacciones con los objetos de la escena en la aplicación de realidad virtual desarrollada. Se describirá cómo distintas selecciones de objetos generan resultados visuales y funcionales diferentes, lo que demuestra la dinámica y la reactividad de la aplicación ante las acciones del usuario.

### **5.1. Descripción de las interacciones**

Las interacciones en la aplicación permiten a los usuarios seleccionar objetos virtuales, los cuales alteran el estado de la escena y desencadenan eventos específicos.

Como se mencionó en la Sección 4.3.3, para lograr estos resultados es necesario utilizar controladores VR. Estos dispositivos permiten a los usuarios moverse por la escena e interactuar con los objetos de manera intuitiva y efectiva. La selección se realiza apuntando hacia el objeto con el controlador derecho y activando el trigger.

Cada objeto tiene asociadas consecuencias específicas que se detallan a continuación:

#### **5.1.1. Interacción con UA1**

Este objeto representa un usuario dentro del entorno virtual y está visualizado como una caja de madera ubicada en la parte izquierda de la escena.

Al interactuar con UA1, este cambiará de color y podrán obtener dos posibles resultados, los cuales dependen de si el intercambio de paquetes ha sido inicializado o no.

## Intercambio de paquetes no iniciado

Si el intercambio de paquetes no ha sido inicializado, en la pantalla situada en la pared central se mostrarán los atributos del usuario en estado vacío, indicando los valores que deberían ser completados, mostrados en la figura 5.1. Esta visualización es esencial para comprender los requisitos iniciales de configuración del usuario en la red. Estos tributos son:

- **account:** [username, passwd]
- **uaserver:** [ip, puerto]
- **rtpaudio:** [puerto]
- **regproxy:** [ip, puerto]
- **log:** [path]
- **audio:** [path]

## Intercambio de paquetes iniciado

Una vez que el intercambio de paquetes ha sido iniciado, en la misma pantalla se actualizarán los atributos del usuario con valores específicos y correctos para su funcionamiento mostrados en la figura 5.2. Este cambio refleja cómo el sistema procesa y responde a las interacciones, ofreciendo un feedback visual del estado operativo del usuario.

### 5.1.2. Interacción con el proxy

Este objeto representa un servidor proxy en el entorno virtual y está visualizado como una caja con textura oscura, ubicada en la parte central de la escena.

Al interactuar con el Proxy, este cambiará de color y dependiendo del estado de la interacción con los usuarios (UA1 y UA2), se pueden obtener diferentes resultados, influenciados por si el intercambio de paquetes ha sido iniciado o no.

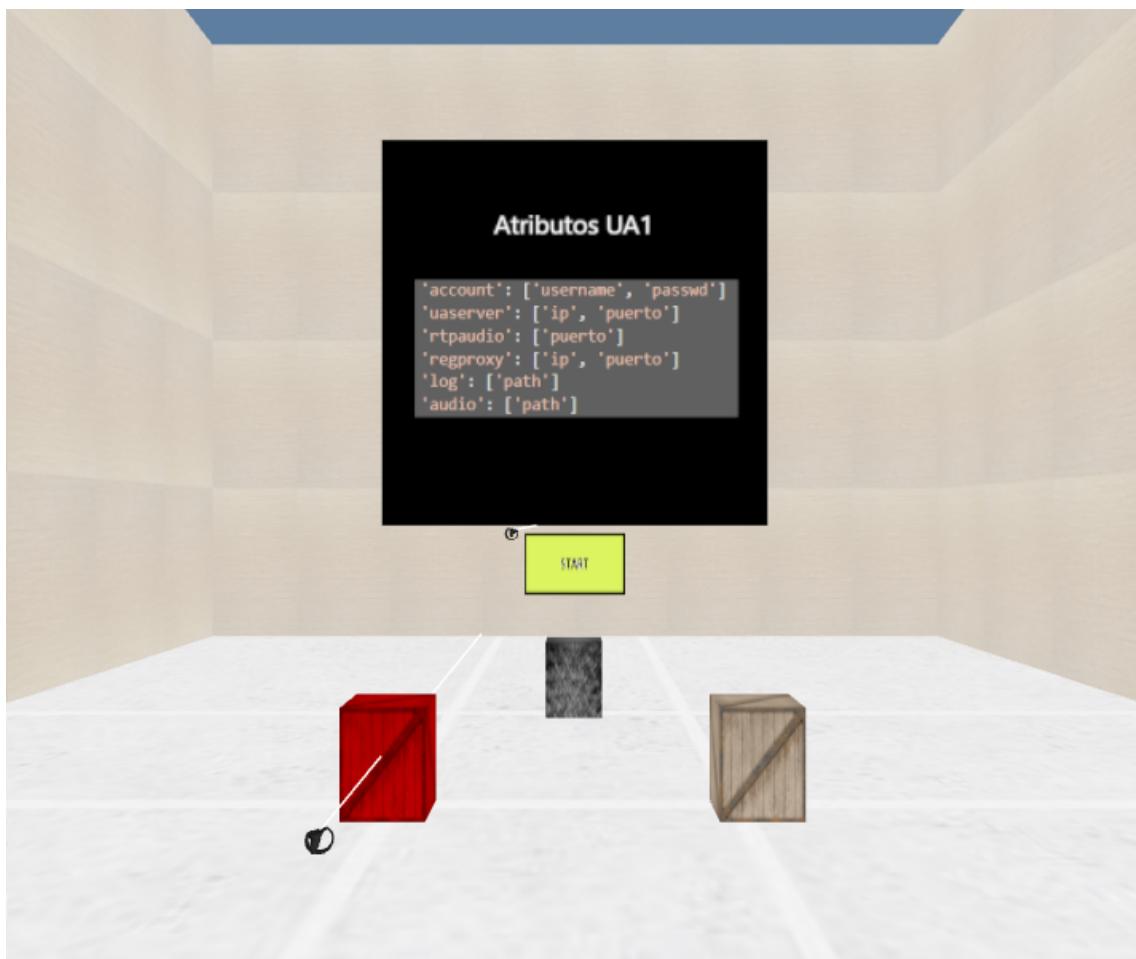


Figura 5.1: Estado inicial del UA1



Figura 5.2: Atributos UA1 con valores específicos

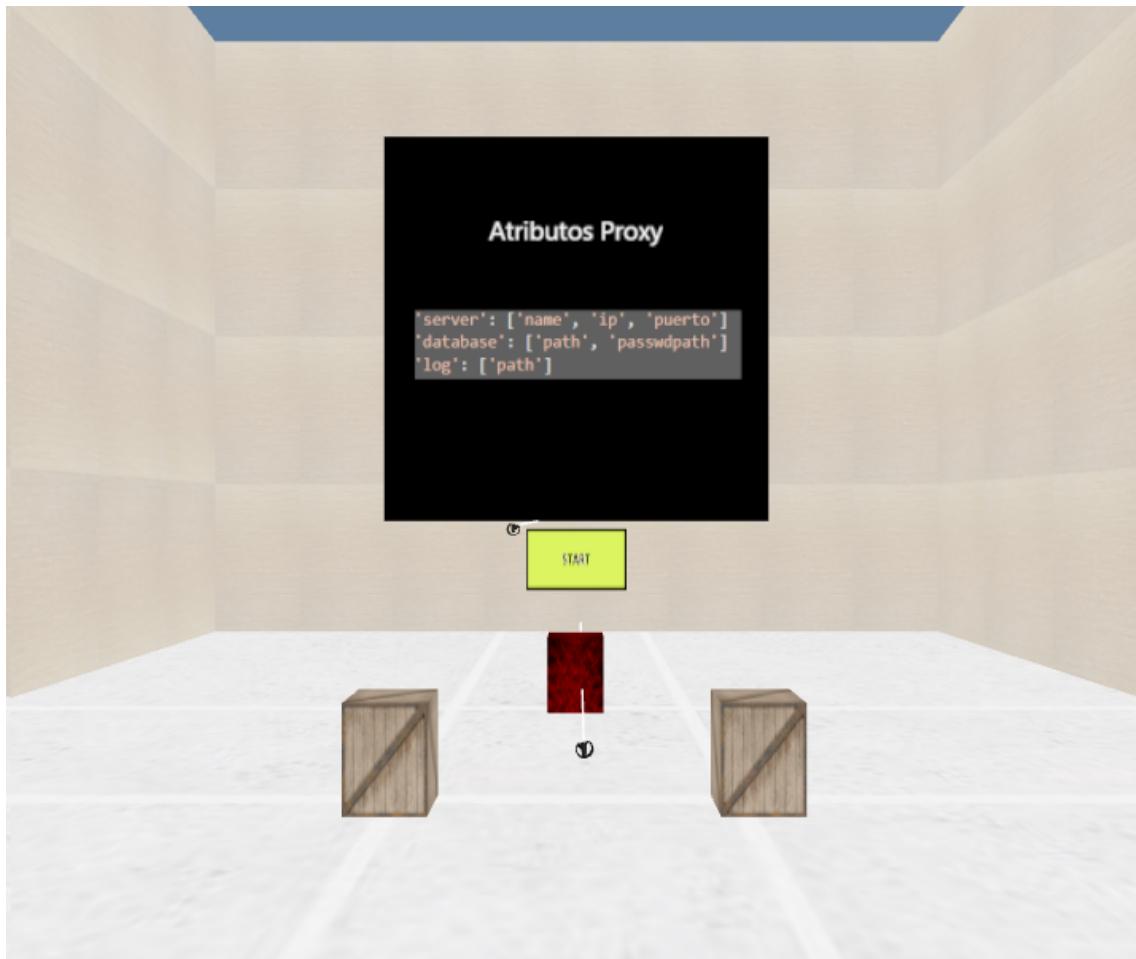


Figura 5.3: Estado inicial del Proxy

### Intercambio de paquetes no iniciado

Si el intercambio de paquetes no ha sido inicializado, la interacción con el Proxy mostrará en la pantalla sus atributos con la información que deben ser completados, donde no se procesan aún los paquetes, mostrados en la figura 5.3. Estos campos son:

- **server:** [name, ip, puerto]
- **database:** [path, passwdpath]
- **log:** [path]

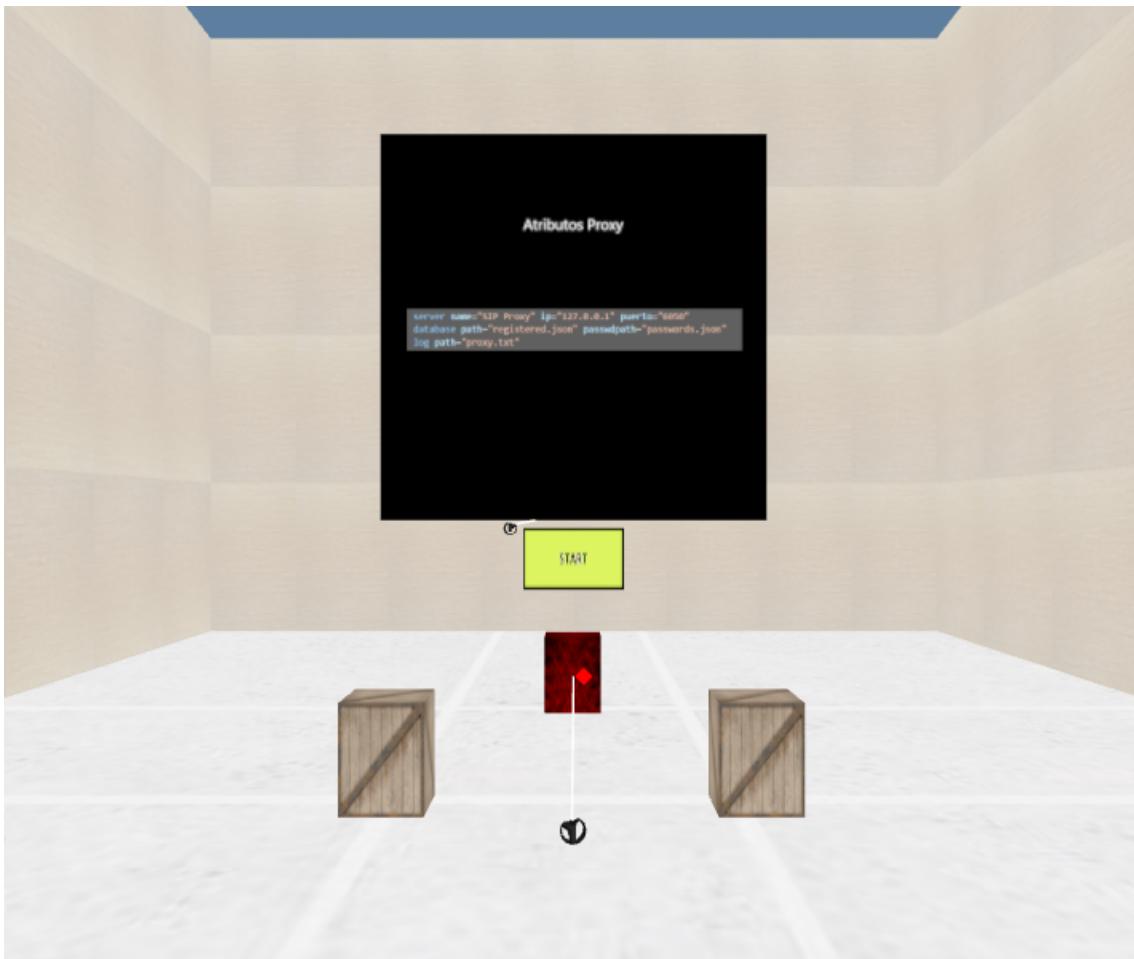


Figura 5.4: Atributos Proxy con valores específicos

### Intercambio de paquetes iniciado

Una vez que el intercambio de paquetes ha sido iniciado, el Proxy procesará y dirigirá los paquetes entre los usuarios correspondientes, reflejando este flujo en la interfaz con actualizaciones dinámicas. Al seleccionar el Proxy se mostrarán sus atributos con valores específicos y correctos para su funcionamiento, mostrados en la figura 5.4.

#### 5.1.3. Interacción con UA2

Este objeto representa un usuario dentro del entorno virtual y está visualizado como una caja de madera ubicada en la parte derecha de la escena.

Al interactuar con UA2, este cambiará de color y podrán obtener dos posibles resultados, los cuales dependen de si el intercambio de paquetes ha sido inicializado o no.

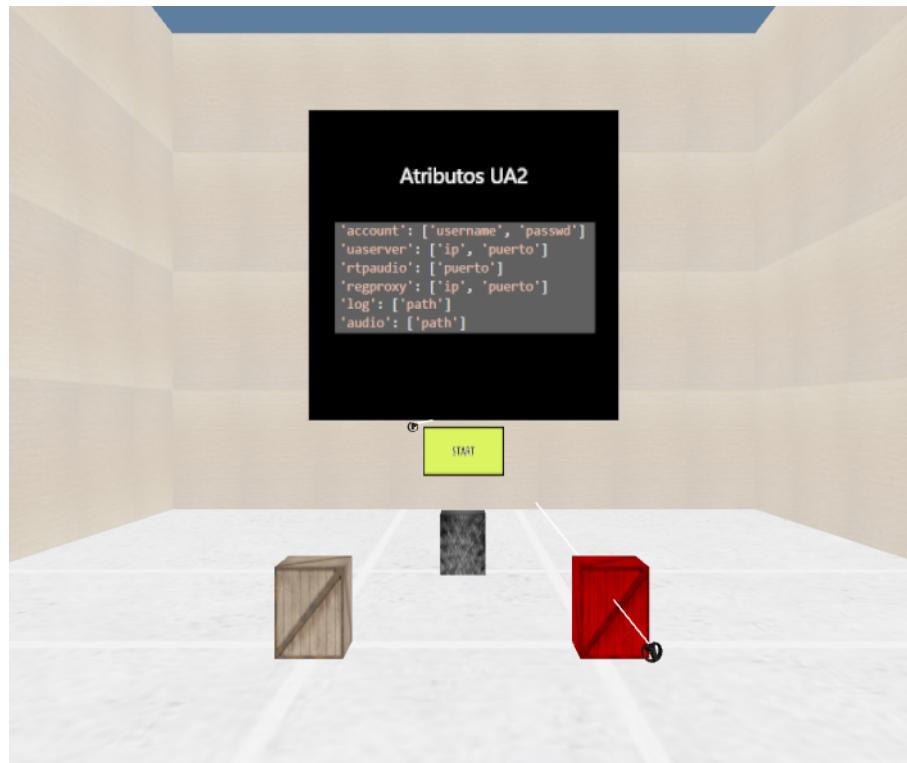


Figura 5.5: Estado inicial del UA2

### Intercambio de paquetes no iniciado

Si el intercambio de paquetes no ha sido inicializado, en la pantalla situada en la pared central se mostrarán los atributos del usuario en estado vacío, indicando los valores que deberían ser completados, mostrados en la figura 5.5. Esta visualización es esencial para comprender los requisitos iniciales de configuración del usuario en la red. Estos tributos son:

- **account:** [username, passwd]
- **uaserver:** [ip, puerto]
- **rtpaudio:** [puerto]
- **regproxy:** [ip, puerto]
- **log:** [path]
- **audio:** [path]



Figura 5.6: Atributos UA2 con valores específicos

### Intercambio de paquetes iniciado

Una vez que el intercambio de paquetes ha sido iniciado, en la misma pantalla se actualizarán los atributos del usuario con valores específicos y correctos para su funcionamiento mostrados en la figura 5.6. Este cambio refleja cómo el sistema procesa y responde a las interacciones, ofreciendo un feedback visual del estado operativo del usuario.

### 5.1.4. Interacción con paquetes

En el entorno de realidad virtual, los paquetes se representan en forma de esferas que simulan la transmisión de datos en una red. Estas esferas visualizan claramente el flujo de información, con un origen, un destino y un color claramente definidos, dependiendo del tipo de intercambio en el que participan.

Al seleccionar uno de estos paquetes, el sistema mostrará información detallada contenida en el paquete, incluyendo una representación visual del flujo de transmisión que indica su origen, destino y la dirección del flujo.

El proceso y secuencia de intercambio de paquetes en este proyecto se ilustra en la siguiente figura 5.7, donde se puede observar el orden específico seguido durante las simulaciones.

Esta figura 5.7 ilustra un intercambio de mensajes de protocolo típico en una comunicación de red utilizando el Protocolo de Inicio de Sesión (SIP). Este diagrama de flujo representa la secuencia de mensajes entre dos usuarios que intentan establecer una comunicación, pasando por un proceso de registro y luego de llamada.

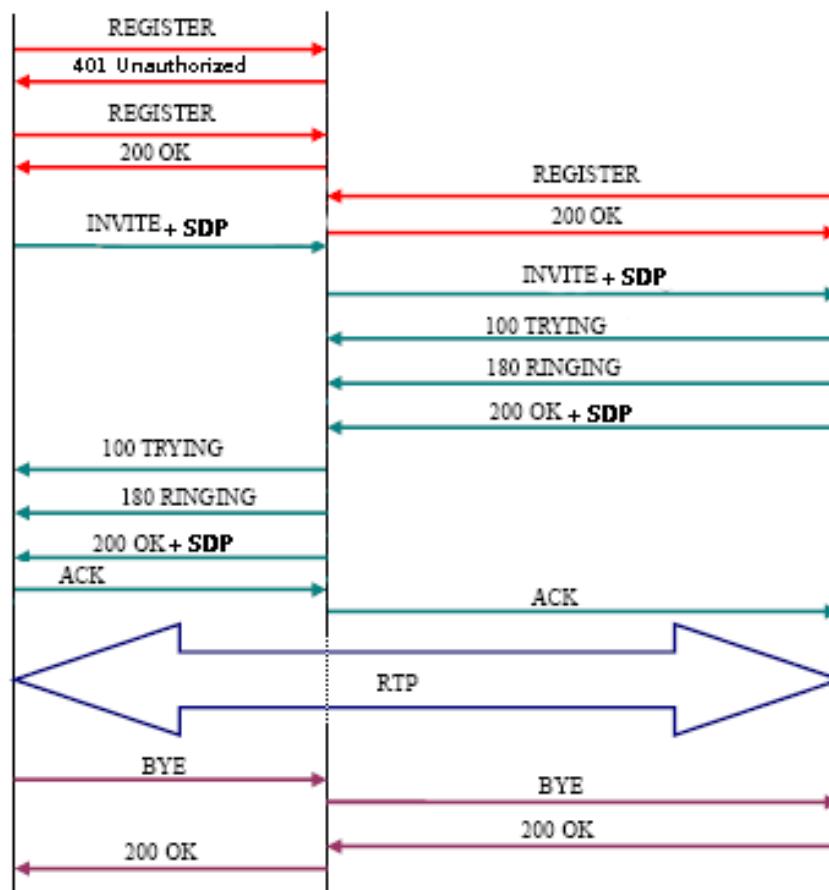


Figura 5.7: Secuencia del intercambio de paquetes

**Desglose de la secuencia de comunicación**

Inicialmente, el proceso comienza con un intento de registro, donde el usuario A (UA1) envía un mensaje REGISTER al servidor, representado en la figura 5.8.

Este mensaje contiene información necesaria para que el servidor registre al usuario en el sistema. Esta información generalmente incluye el identificador del usuario, su dirección de red y otros parámetros necesarios para la autenticación y la configuración de la sesión.

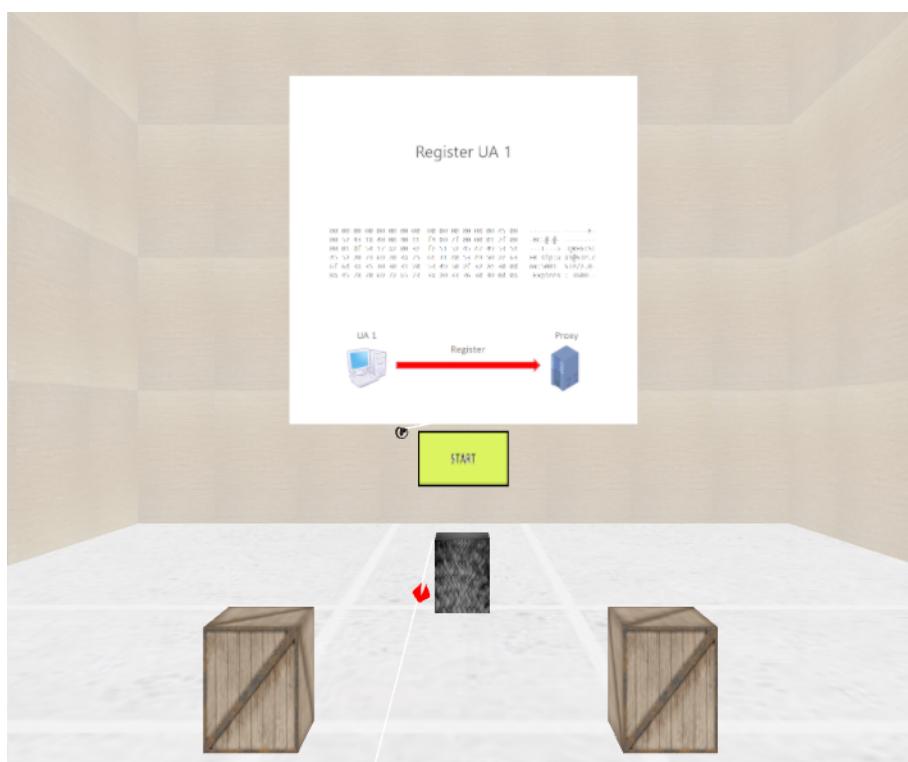


Figura 5.8: Mensaje de registro inicial de UA1

Si el usuario no está autorizado, recibe un mensaje 401 Unauthorized, mostrado en la figura 5.9.

El mensaje 401 Unauthorized es una respuesta del servidor que indica que el intento de registro ha fallado debido a la falta de credenciales adecuadas. Como resultado, el usuario (UA1) debe realizar otro intento de REGISTER con la información de autenticación adecuada, como un nombre de usuario y una contraseña válidos.

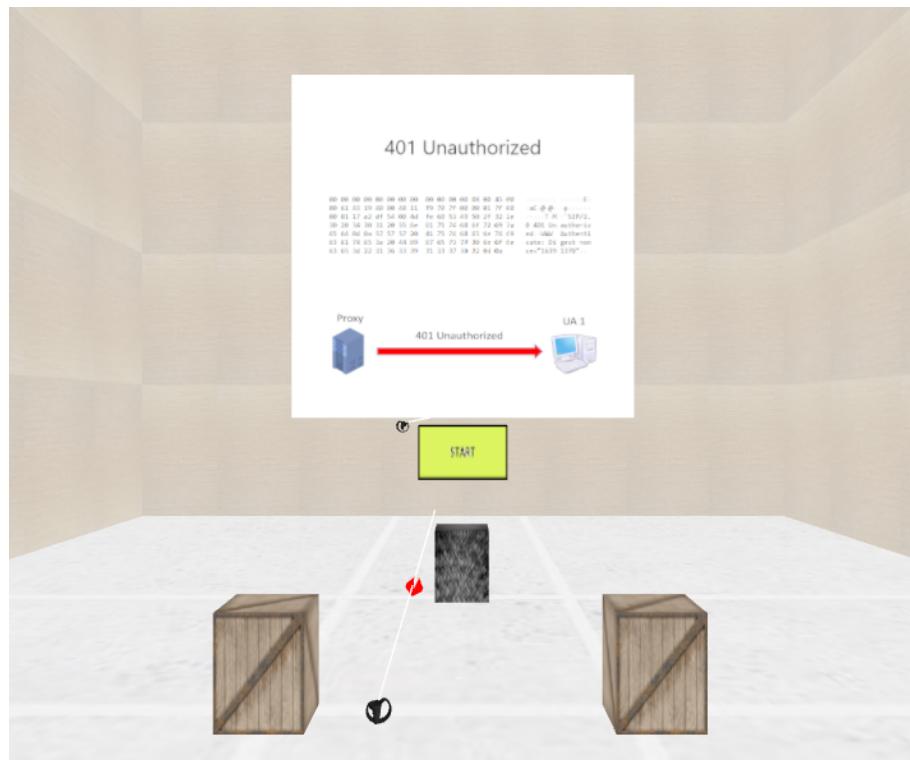


Figura 5.9: Respuesta 401 Unauthorized a UA1

El segundo intento de REGISTER es aceptado y confirmado con una respuesta 200 OK, ilustrado en la figura 5.10.

El mensaje 200 OK indica que el servidor ha aceptado el registro del usuario y que el usuario está ahora autenticado y registrado en el sistema. Este mensaje puede incluir información adicional de los parámetros de configuración de la sesión.

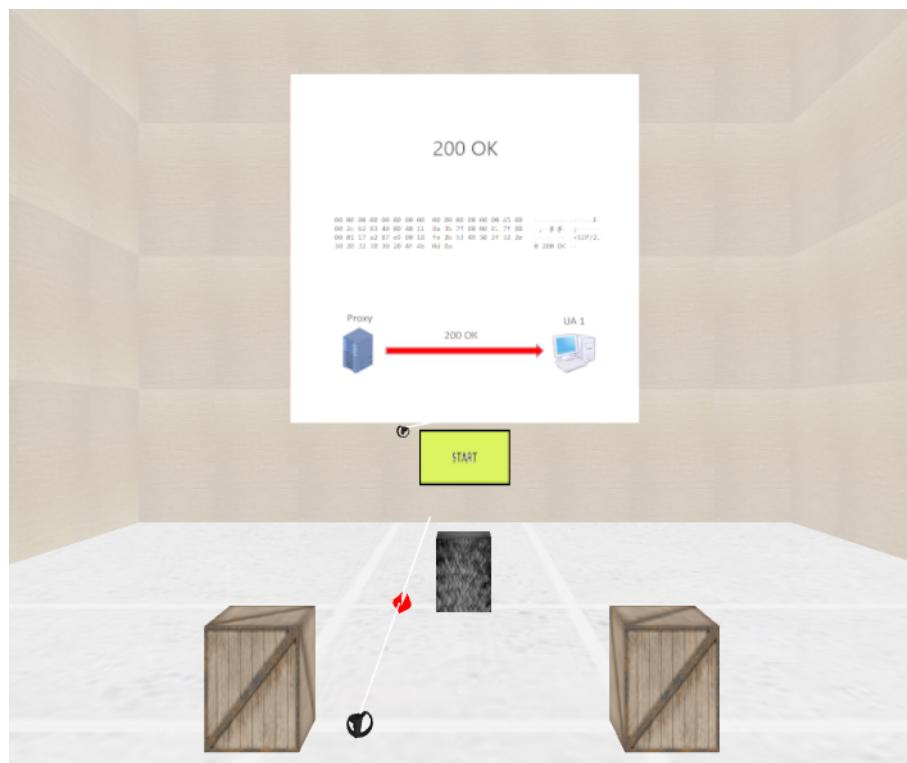


Figura 5.10: Registro de UA1 exitoso con 200 OK

A continuación, el usuario B (UA2) se registra enviando un mensaje REGISTER al servidor, representado en la figura 5.11. Este mensaje sigue el mismo proceso de autenticación y registro.

La figura muestra cómo UA2 inicia el proceso de registro proporcionando su información de autenticación al servidor.

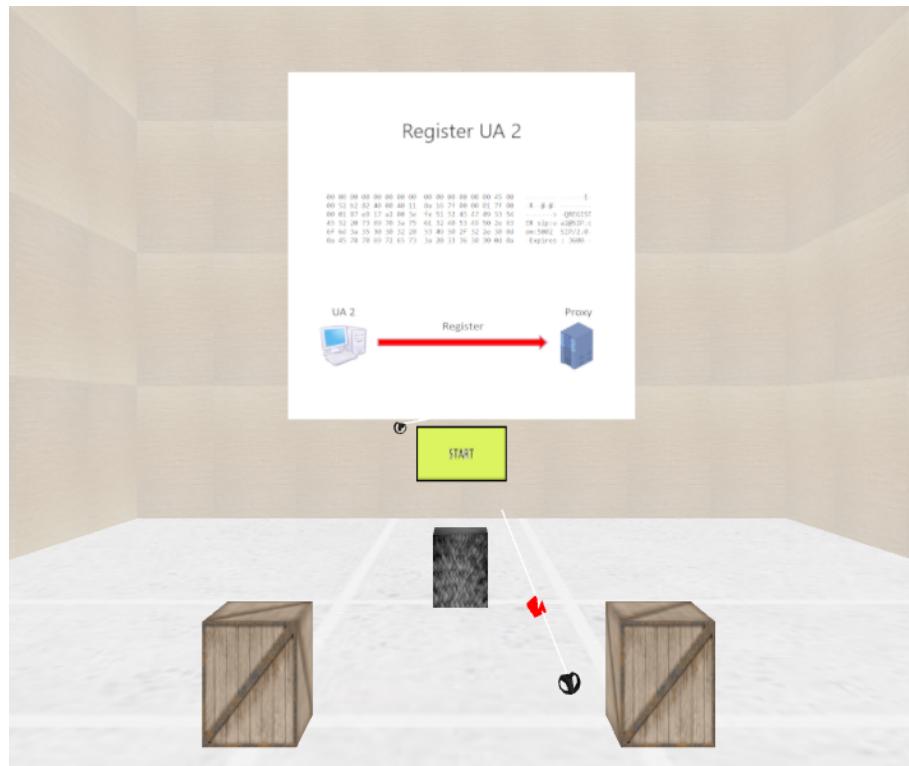


Figura 5.11: Mensaje de registro inicial de UA2

El registro de UA2 es aceptado y confirmado con una respuesta 200 OK, ilustrado en la figura 5.12.

De igual forma que en registro de UA1, el mensaje 200 OK confirma que UA2 ha sido autenticado y registrado exitosamente en el sistema.

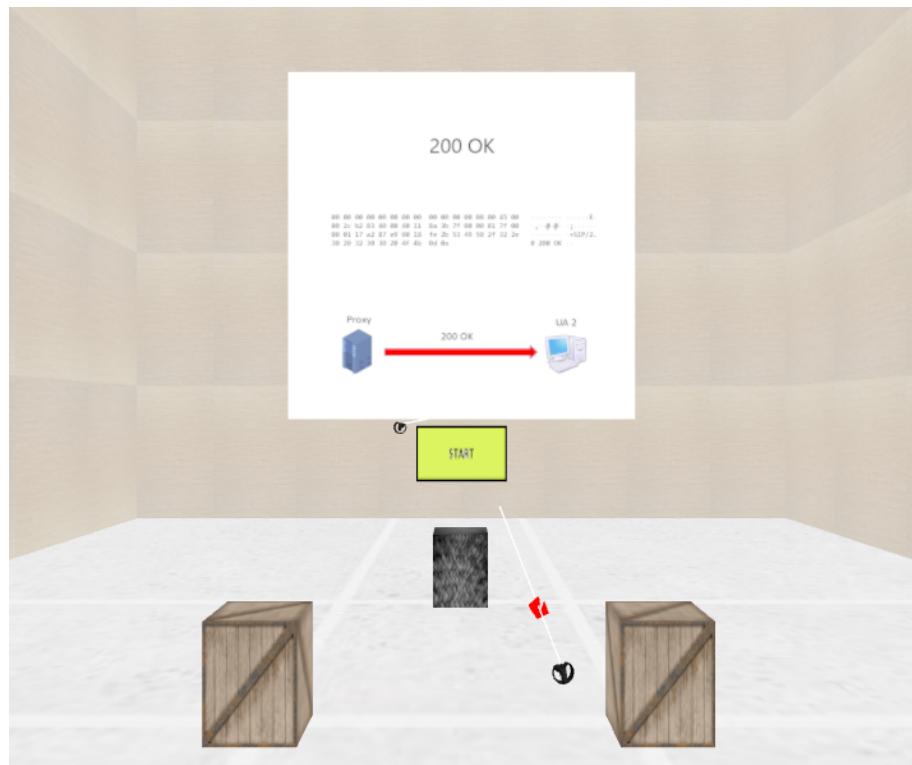


Figura 5.12: Registro de UA2 exitoso con 200 OK

Ahora que ambos usuarios están registrados en el servidor proxy, la fase de llamada comienza con un mensaje INVITE acompañado de una descripción de la sesión (SDP), escenificado en la figura 5.13.

El mensaje INVITE es enviado por UA1 a UA2 a través del servidor y contiene detalles sobre la sesión propuesta, incluyendo los parámetros de conexión.

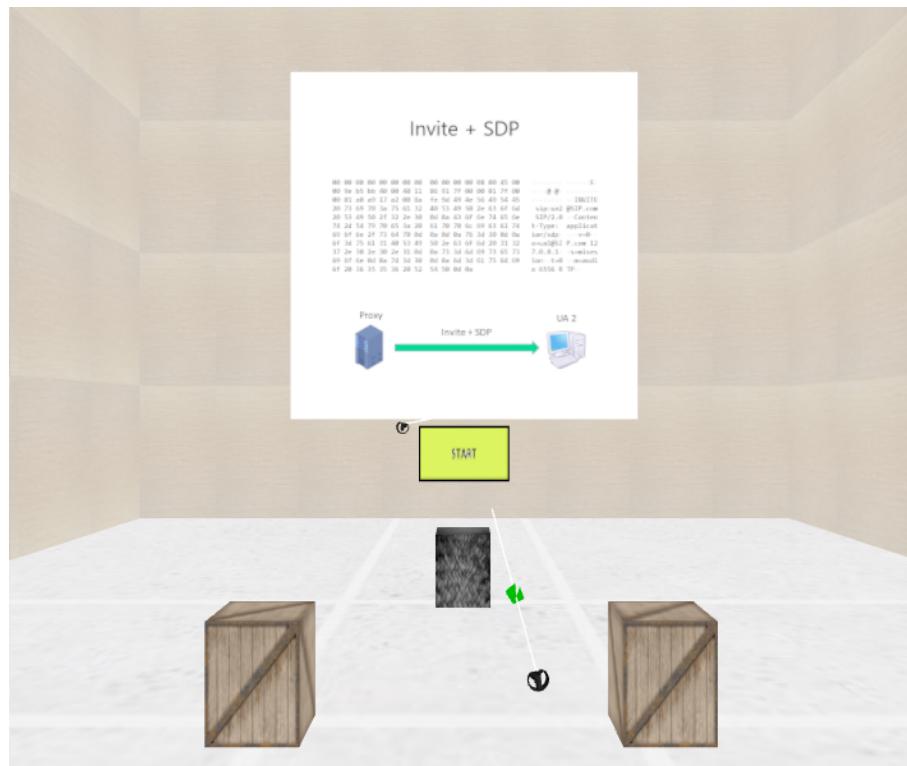


Figura 5.13: Inicio de llamada con el mensaje INVITE de UA1

Este mensaje es respondido con un 100 TRYING, evidenciado en la figura 5.14. El mensaje 100 TRYING indica que el servidor ha recibido el mensaje INVITE y está intentando localizar al usuario de destino (UA2).

Posteriormente, el servidor envía un 180 RINGING, ilustrado en la figura 5.15, que indica que la llamada está siendo procesada y que el otro usuario está siendo alertado de la llamada entrante.

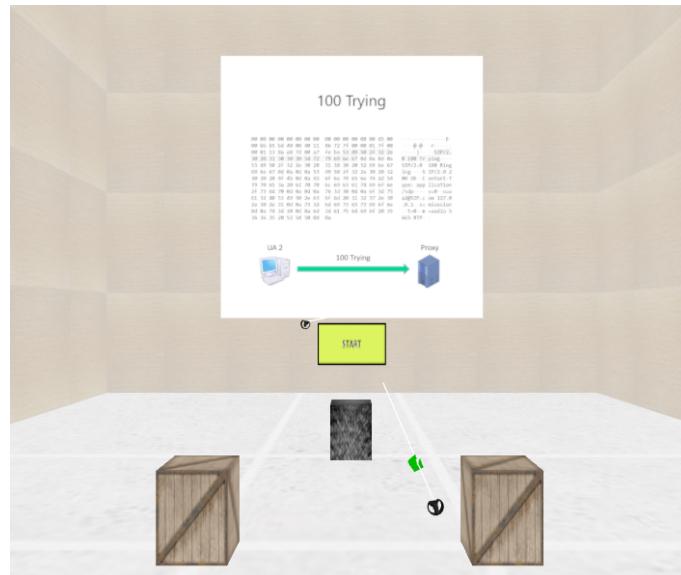


Figura 5.14: Respuesta TRYING de UA2 indicando procesamiento

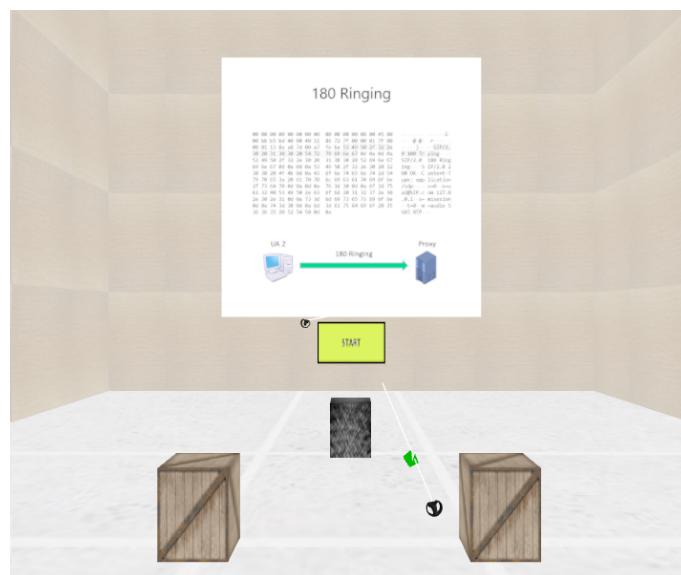


Figura 5.15: Alerta de llamada con el mensaje RINGING de UA2

Cuando el usuario B (UA2) acepta la llamada, envía una respuesta 200 OK + SDP, representado en la figura 5.16.

Este mensaje contiene la confirmación de que UA2 está listo para establecer la sesión de comunicación y proporciona los detalles necesarios para la transmisión.

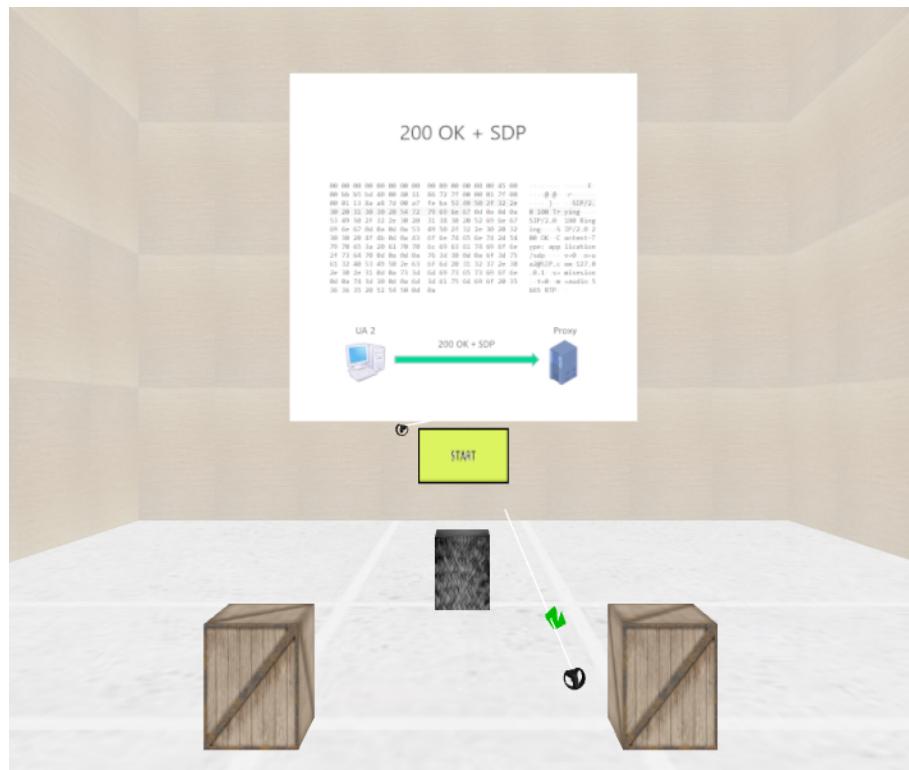


Figura 5.16: Confirmación de llamada con 200 OK + SDP de UA2

El usuario A (UA1) confirma recibir esta información con un ACK, mostrado en la figura 5.17. El mensaje ACK es una confirmación final que completa el establecimiento de la llamada y permite el inicio de la transmisión.

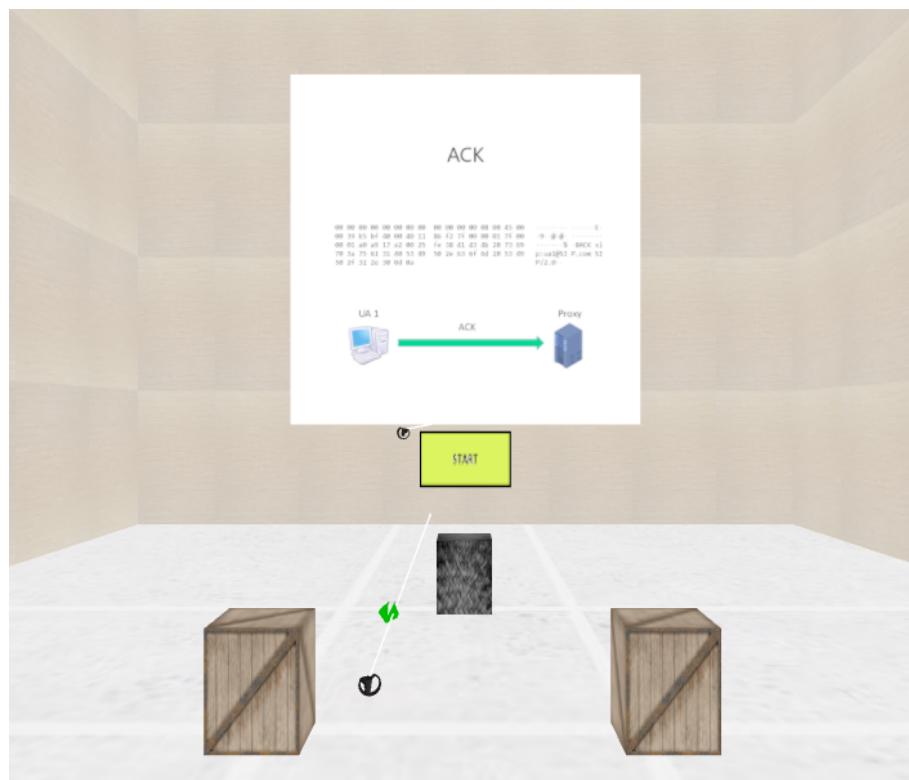


Figura 5.17: Reconocimiento ACK de UA1 completando el establecimiento de la llamada

Una vez establecida la llamada, los datos multimedia se transmiten a través de paquetes RTP (Protocolo de Transporte en Tiempo Real), escenificado en la figura 5.18. Los paquetes RTP transportan la información de audio entre UA1 y UA2, permitiendo la comunicación en tiempo real.

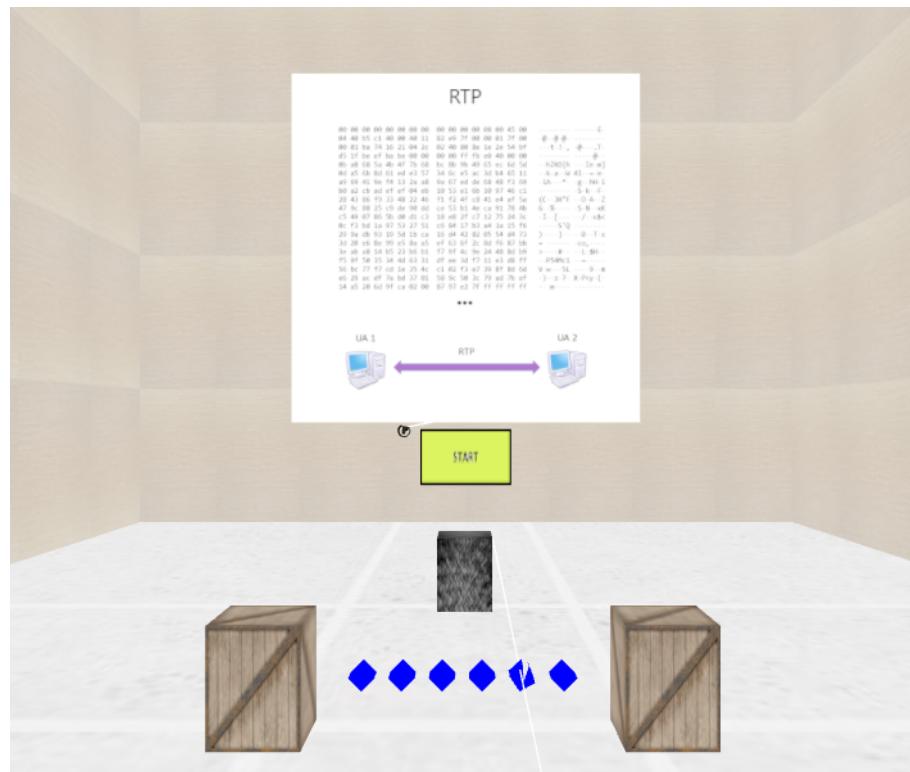


Figura 5.18: Transmisión de datos multimedia a través de paquetes RTP

Este intercambio continúa hasta que uno de los usuarios decide terminar la llamada, enviando un mensaje BYE, ilustrado en la figura 5.19.

El mensaje BYE indica la intención de finalizar la llamada y cerrar la sesión de comunicación.

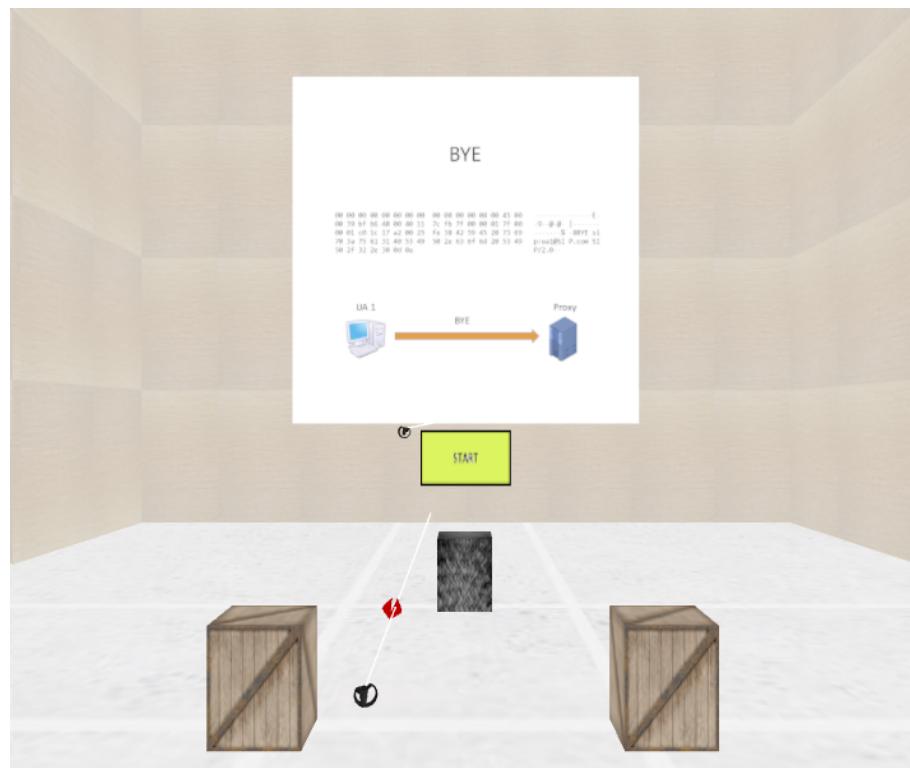


Figura 5.19: Mensaje BYE enviado para terminar la llamada

El fin de la llamada es confirmado por el otro usuario con un 200 OK, representado en la figura 5.20, cerrando así la sesión de comunicación.

Este mensaje final confirma que ambos usuarios han terminado la sesión de manera ordenada.

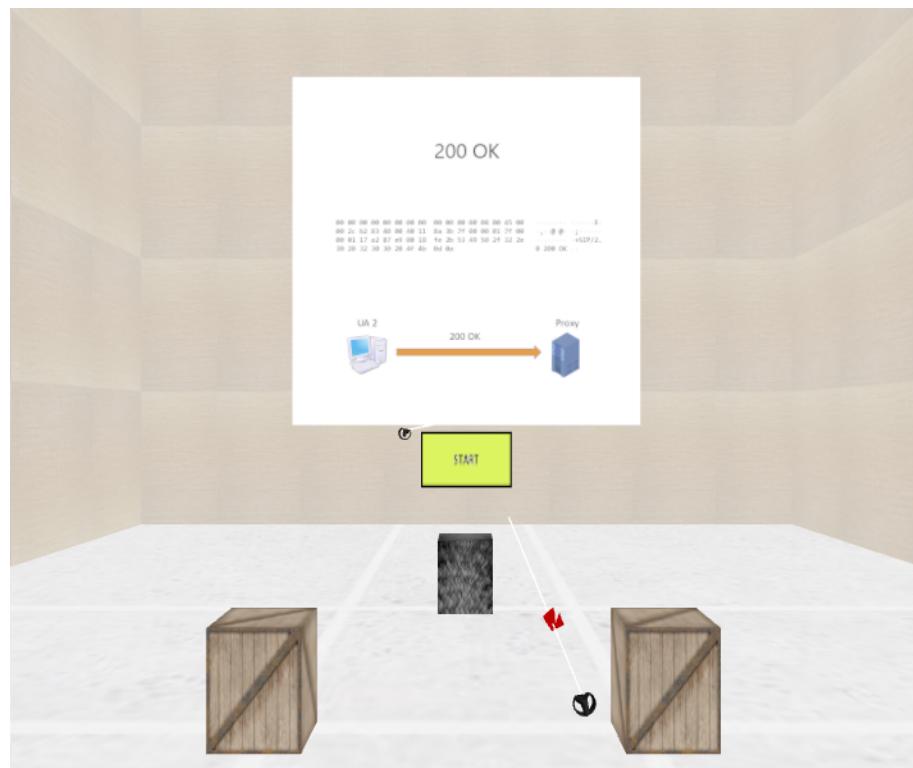


Figura 5.20: Confirmación del fin de la llamada con un 200 OK

# **Capítulo 6**

## **Conclusiones**

### **6.1. Consecución de objetivos**

#### **6.1.1. Objetivo general**

El objetivo general del proyecto era construir una representación visual y funcional de los flujos de señalización y datos en un entorno de realidad virtual, utilizando tecnologías avanzadas como Three.js y módulos adicionales para facilitar la interacción en entornos VR. Este objetivo se ha cumplido satisfactoriamente, como se demostró en los resultados de funcionamiento del sistema, donde los usuarios pueden interactuar con la simulación de comunicaciones y experimentar de forma inmersiva el manejo de las comunicaciones en tiempo real.

### **6.2. Aplicación de lo aprendido**

A lo largo de mi formación en el grado de Ingeniería en Sistemas Audiovisuales y Multimedia, adquirí una serie de conocimientos y habilidades que han sido fundamentales para el desarrollo de mi proyecto de fin de grado. A continuación, detallo cómo se aplicaron específicamente algunas de las asignaturas más relevantes:

- 1. Informática I e Informática II:** Estas fueron las primeras asignaturas relacionadas con la programación, las cuales sentaron las bases y aportaron los primeros pasos para desarrollar habilidades en el área de las telecomunicaciones.

2. **Protocolos para la Transmisión de Audio y Vídeo por Internet:** Esta asignatura profundizó en la programación de manera más específica, realizando prácticas que combinaban conocimientos técnicos con la utilización de protocolos de comunicación.
3. **Construcción de Servicios y Aplicaciones Audiovisuales en Internet:** En esta asignatura se asentaron las bases del desarrollo web utilizando lenguajes como JavaScript, HTML5 y CSS, elementos cruciales para la interfaz de usuario de mi proyecto.
4. **Gráficos y Visualización en 3D:** Aprendí a desarrollar una parte significativa de mi proyecto, adquiriendo conocimientos en gráficos 3D, conceptos básicos de WebGL y la creación de gráficos utilizando Three.js.

En general, todas las asignaturas del grado han aportado valiosos conocimientos y fomentado el desarrollo del pensamiento crítico y la resolución de problemas, habilidades que he aplicado tanto en el TFG como en el entorno laboral.

### 6.3. Lecciones aprendidas

Durante el desarrollo de este proyecto, he adquirido una serie de conocimientos tanto técnicos como personales que han enriquecido mi experiencia y han contribuido a mi desarrollo profesional.

1. **Importancia de la planificación detallada:** Una de las lecciones más valiosas ha sido comprender la importancia de una planificación y organización en proyectos de desarrollo tecnológico.
2. **Profundización en tecnologías de realidad virtual:** El proyecto me permitió profundizar en el uso de tecnologías de realidad virtual, especialmente en la programación con JavaScript y el uso de bibliotecas como Three.js. Aprendí no solo a implementar estas tecnologías sino también a resolver problemas específicos relacionados con la renderización de gráficos 3D y la interactividad en entornos virtuales [13].
3. **Capacidad de adaptación y solución de problemas:** Desarrollo de la capacidad de buscar soluciones alternativas y adaptar el enfoque del proyecto según las circunstancias fue importante para el éxito del proyecto.

4. **Comprendión de los protocolos de comunicación:** El proyecto profundizó mi comprensión de los protocolos de comunicación como SIP y RTP. A través de la implementación práctica, pude ver cómo la teoría se aplica en situaciones reales.

## 6.4. Trabajos futuros

Los resultados obtenidos implican que la aplicación de realidad virtual en el campo de las telecomunicaciones puede mejorar significativamente la comprensión de sistemas complejos de comunicación. Se recomienda que investigaciones futuras exploren ampliar la funcionalidad del sistema para incluir más escenarios de comunicación y evaluar su aplicabilidad en entornos educativos y profesionales.



# **Apéndice A**

## **Manual de usuario**

Este apéndice proporciona un manual de usuario para la aplicación desarrollada en este proyecto. Aquí se explican los pasos para instalar, configurar y utilizar la aplicación.

### **A.1. Requisitos del sistema**

Para utilizar esta aplicación, asegúrate de que tu sistema cumple con los siguientes requisitos:

- Sistema operativo: Windows.
- Navegador web compatible: Google Chrome.
- Hardware: Gafas de realidad virtual (por ejemplo, Oculus Quest 2) o en su defecto, descargar la extensión Web XR API Emulator.
- Conexión a Internet.

### **A.2. Uso de la aplicación**

#### **A.2.1. Instalación de la extensión WebXR API Emulator**

Lo primero que debemos hacer es descargarnos e instalar la extensión Web XR API Emulator.

Una vez realizado este paso, accedemos a la URL donde se encuentra desplegada la aplicación. La URL es la siguiente:

<https://jorgegragon.github.io/TFGJorge/VRM/index.html>

Al acceder al enlace, veremos lo siguiente.

### A.2.2. Activación de la realidad virtual

A continuación pulsamos F12 para abrir la herramienta de desarrollador, donde seleccionaremos en la parte superior de esta ventana, la opción de WebXR.

Una vez realizado este paso, en la parte inferior de la escena, tenemos un botón que pone Enter VR. Lo pulsamos para activar la realidad virtual.

### A.2.3. Interacción en la aplicación

Ya tenemos la aplicación funcionando, podremos movernos por la escena, accionar botones y obtener información acerca de los usuarios y del proxy.

Tenemos dos controles, que producen interacción con la aplicación:

- **Select Button del controlador izquierdo:** Con este botón nos movemos por la escena en la dirección frontal hacia donde están mirando las gafas virtuales.
- **Select Button del controlador derecho:** Con este botón interactuamos con los objetos de la escena. Apuntamos hacia el objeto y pulsamos el botón. Esto hará que, dependiendo del objeto seleccionado, se produzca o no un cambio en la escena.

### A.2.4. Activación de la secuencia de intercambio de paquetes

Para activar la secuencia del intercambio de paquetes, con el controlador derecho, apuntar hacia el botón de START y pulsamos el Select Button.

De esta forma, veremos cómo se ejecuta el intercambio, pudiendo pararlo en todo momento para obtener información de los paquetes, usuarios o proxy, de la misma forma que hemos hecho anteriormente, ya que el botón en la escena que indicaba START ahora indicará STOP.

## A.3. Modo sin realidad virtual

Adicionalmente, si no se desea ejecutar la aplicación en realidad virtual, no se quiere descargar la extensión o no se poseen las gafas VR. Podremos visualizar la escena y el intercambio de paquetes en 2D y con los controles del teclado, en la siguiente URL:

<https://jorgegragon.github.io/TFGJorge/Original/AreaEstado.html>

Donde los controles serán:

- **Movimiento:** W, A, S, D, Q y E
- **Iniciar secuencia:** Usar la barra espaciadora para comenzar el intercambio.
- **Parar secuencia:** Clicar en los usuarios, proxy o paquete.
- **Reanudar secuencia:** Usar la barra espaciadora para reanudar.



# Bibliografía

- [1] I. como RFC 1889. Rtp: A transport protocol for real-time applications.  
<https://www.3cx.es/voip-sip/rtp/>.
- [2] I. como RFC 3261. Session initiation protocol. <https://www.3cx.es/voip-sip/sip/>.
- [3] I. como RFC 4566. Session description protocol. <https://www.3cx.es/voip-sip/sdp/>.
- [4] GitHub. Github. <https://github.com/>.
- [5] N. Lever. Curso vr, Oct. 2021. <https://youtu.be/juwRSr-J2rg?si=Secs5epkMgLtRCPZ>.
- [6] Microsoft. Visual studio code. <https://code.visualstudio.com/>.
- [7] Mozilla. Mozilla firefox. <https://www.mozilla.org/es-ES/firefox/browsers/what-is-a-browser/>.
- [8] Mrdoob. Documentación controles. <https://threejs.org/docs/#examples/en/controls/OrbitControls>.
- [9] Mrdoob. Documentación de como crear contenido de realidad virtual con three.js.  
<https://threejs.org/docs/#manual/en/introduction/How-to-create-VR-content>.
- [10] Mrdoob. Documentación three.js. <https://threejs.org/docs/index.html>.
- [11] Mrdoob. Documentación webxr. <https://threejs.org/docs/#api/en/renderers/webxr/WebXRManager>.
- [12] Mrdoob. Stats.js. <https://github.com/mrdoob/stats.js/blob/master/README.md>.
- [13] Mrdoob. Ejemplos three.js, June 2023. <https://threejs.org/examples/>.
- [14] Oculus. Oculus quest 2. <https://www.meta.com/es/quest/products/quest-2>.

- [15] M. M. Reality. Web xr api emulator. <https://addons.mozilla.org/en-GB/firefox/addon/webxr-api-emulator/>.