



**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**

Aquí vienen los agradecimientos... Aunque está bien acordarse de la pareja, no hay que olvidarse de dar las gracias a tu madre, que aunque a veces no lo parezca disfrutará tanto de tus logros como tú... Además, la pareja quizás no sea para siempre, pero tu madre sí.



# Resumen

Aquí viene un resumen del proyecto. Ha de constar de tres o cuatro párrafos, donde se presente de manera clara y concisa de qué va el proyecto. Han de quedar respondidas las siguientes preguntas:

- ¿De qué va este proyecto? ¿Cuál es su objetivo principal?
- ¿Cómo se ha realizado? ¿Qué tecnologías están involucradas?
- ¿En qué contexto se ha realizado el proyecto? ¿Es un proyecto dentro de un marco general?

Lo mejor es escribir el resumen al final.



# **Summary**

Here comes a translation of the “Resumen” into English. Please, double check it for correct grammar and spelling. As it is the translation of the “Resumen”, which is supposed to be written at the end, this as well should be filled out just before submitting.



# Índice general

<b>1. Introducción</b>	<b>1</b>
1.1. Estructura de la memoria . . . . .	2
<b>2. Objetivos</b>	<b>3</b>
2.1. Objetivo general . . . . .	3
2.2. Objetivos específicos . . . . .	3
2.3. Planificación temporal . . . . .	4
<b>3. Estado del arte</b>	<b>5</b>
3.1. Lenguajes de Marcado y Programación . . . . .	5
3.2. Importaciones de Bibliotecas JavaScript . . . . .	6
3.3. Herramientas de Desarrollo . . . . .	6
<b>4. Diseño e implementación</b>	<b>7</b>
4.1. Arquitectura general . . . . .	7
4.2. Escena . . . . .	10
4.3. Controles de Realidad Virtual . . . . .	12
4.3.1. Configuración de los Controladores VR . . . . .	12
4.3.2. Interacción Mediante Controladores VR . . . . .	13
4.3.3. Implementación Técnica . . . . .	13
<b>5. Resultados</b>	<b>15</b>
5.1. Descripción de las Interacciones . . . . .	15
5.1.1. Interacción con UA1 . . . . .	15
5.1.2. Interacción con el Proxy . . . . .	16

5.1.3. Interacción con UA2 . . . . .	20
5.1.4. Interacción con paquetes . . . . .	21
<b>6. Conclusiones</b>	<b>33</b>
6.1. Consecución de objetivos . . . . .	33
6.2. Aplicación de lo aprendido . . . . .	33
6.3. Lecciones aprendidas . . . . .	34
6.4. Trabajos futuros . . . . .	34
<b>A. Manual de usuario</b>	<b>35</b>

# Índice de figuras

3.1. Porcentaje de lenguajes usados. . . . .	5
4.1. Estructura del proyecto . . . . .	10
4.2. Escena del proyecto . . . . .	12
4.3. Controlador VR . . . . .	13
4.4. Botones controlador VR . . . . .	14
5.1. Estado inicial del UA1 . . . . .	17
5.2. Atributos UA1 con valores específicos . . . . .	18
5.3. Estado inicial del Proxy . . . . .	19
5.4. Atributos Proxy con valores específicos . . . . .	20
5.5. Estado inicial del UA2 . . . . .	22
5.6. Atributos UA2 con valores específicos . . . . .	23
5.7. Secuencia del intercambio de paquetes . . . . .	24
5.8. Mensaje de Registro Inicial de UA1 . . . . .	26
5.9. Respuesta 401 Unauthorized a UA1 . . . . .	26
5.10. Registro de UA1 Exitoso con 200 OK . . . . .	27
5.11. Mensaje de Registro Inicial de UA2 . . . . .	27
5.12. Registro de UA2 Exitoso con 200 OK . . . . .	28
5.13. Inicio de Llamada con el Mensaje INVITE de UA1 . . . . .	28
5.14. Respuesta TRYING de UA2 Indicando Procesamiento . . . . .	29
5.15. Alerta de Llamada con el Mensaje RINGING de UA2 . . . . .	29
5.16. Confirmación de Llamada con 200 OK + SDP de UA2 . . . . .	30
5.17. Reconocimiento ACK de UA1 Completando el Establecimiento de la Llamada . . . . .	30

5.18. Transmisión de Datos Multimedia a través de Paquetes RTP . . . . .	31
5.19. Mensaje BYE Enviado para Terminar la Llamada . . . . .	31
5.20. Confirmación del Fin de la Llamada con un 200 OK . . . . .	32

# **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 centra en el desarrollo e implementación de un sistema de comunicación dentro de un entorno de realidad virtual, destacando la aplicación práctica y la comprensión de los protocolos de comunicación estándar, como el Protocolo de Inicio de Sesión (SIP) y el Protocolo de Transporte en Tiempo Real (RTP).

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 y simulación.

## 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.

### **2.3. Planificación temporal**

A mí me gusta que aquí pongáis una descripción de lo que os ha llevado realizar el trabajo. Hay gente que añade un diagrama de GANTT. Lo importante es que quede claro cuánto tiempo llevas (tiempo natural, p.ej., 6 meses) y a qué nivel de esfuerzo (p.ej., principalmente los fines de semana).

# Capítulo 3

## Estado del arte

En este capítulo se explicarán todas las tecnologías, herramientas y accesorios necesarios para el desarrollo.

### 3.1. Lenguajes de Marcado y Programación

- **HTML:** Lenguaje estándar para crear páginas web, define la estructura y el contenido utilizando etiquetas.
- **JavaScript:** Lenguaje de programación utilizado en desarrollo web para agregar interactividad y dinamismo a las páginas.
- **TeX:** Sistema de composición tipográfica utilizado principalmente para la creación de documentos científicos y técnicos de alta calidad.

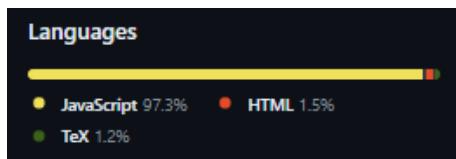


Figura 3.1: Porcentaje de lenguajes usados.

## 3.2. Importaciones de Bibliotecas JavaScript

- **THREE.js:** Biblioteca JavaScript de código abierto para crear y renderizar gráficos en 3D en el navegador web.
- **VRButton.js:** Módulo de Three.js que facilita la integración de botones para activar la funcionalidad de realidad virtual (VR) en aplicaciones web.
- **XRControllerModelFactory.js:** Módulo de Three.js que proporciona una fábrica para crear modelos de controladores de realidad extendida (XR) para su uso en aplicaciones de realidad virtual y aumentada.
- **Stats.js:** Módulo de Three.js que ofrece una utilidad para monitorear el rendimiento (FPS, uso de memoria) de aplicaciones web 3D en tiempo real.
- **OrbitControls.js:** Módulo de Three.js que proporciona controles de órbita para permitir al usuario rotar, acercar y alejar la cámara en una escena 3D de manera interactiva.

## 3.3. Herramientas de Desarrollo

- **FireFox:** Navegador web de código abierto conocido por su enfoque en la privacidad y la personalización.
- **VisualStudioCode:** Editor de código fuente de Microsoft altamente personalizable y de código abierto, conocido por su rendimiento y amplia gama de extensiones.
- **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.
- **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.
- **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.

# 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 4.2. 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 `ppz` 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 "íimagenes" 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.

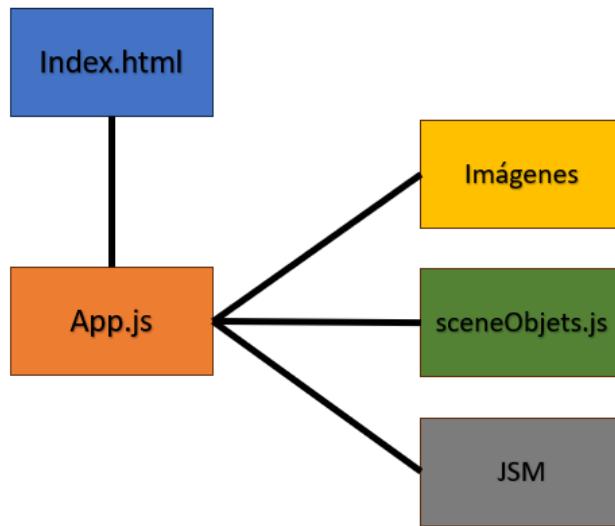


Figura 4.1: Estructura del proyecto

## 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.

### 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" "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 Técnica

- **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:** 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:** 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.

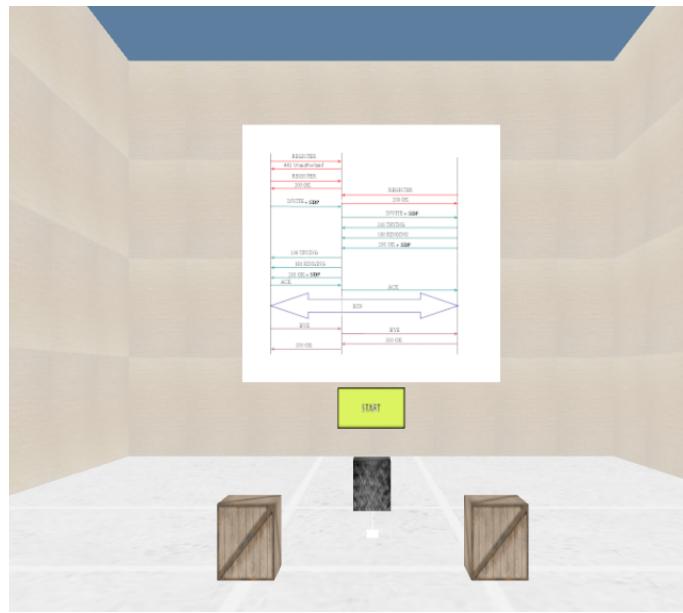


Figura 4.2: Escena del proyecto

### 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.

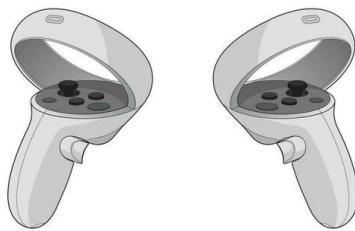


Figura 4.3: Controlador VR

#### 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.

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

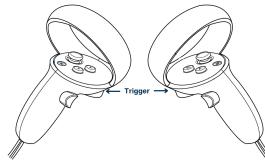


Figura 4.4: Botones controlador VR

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.

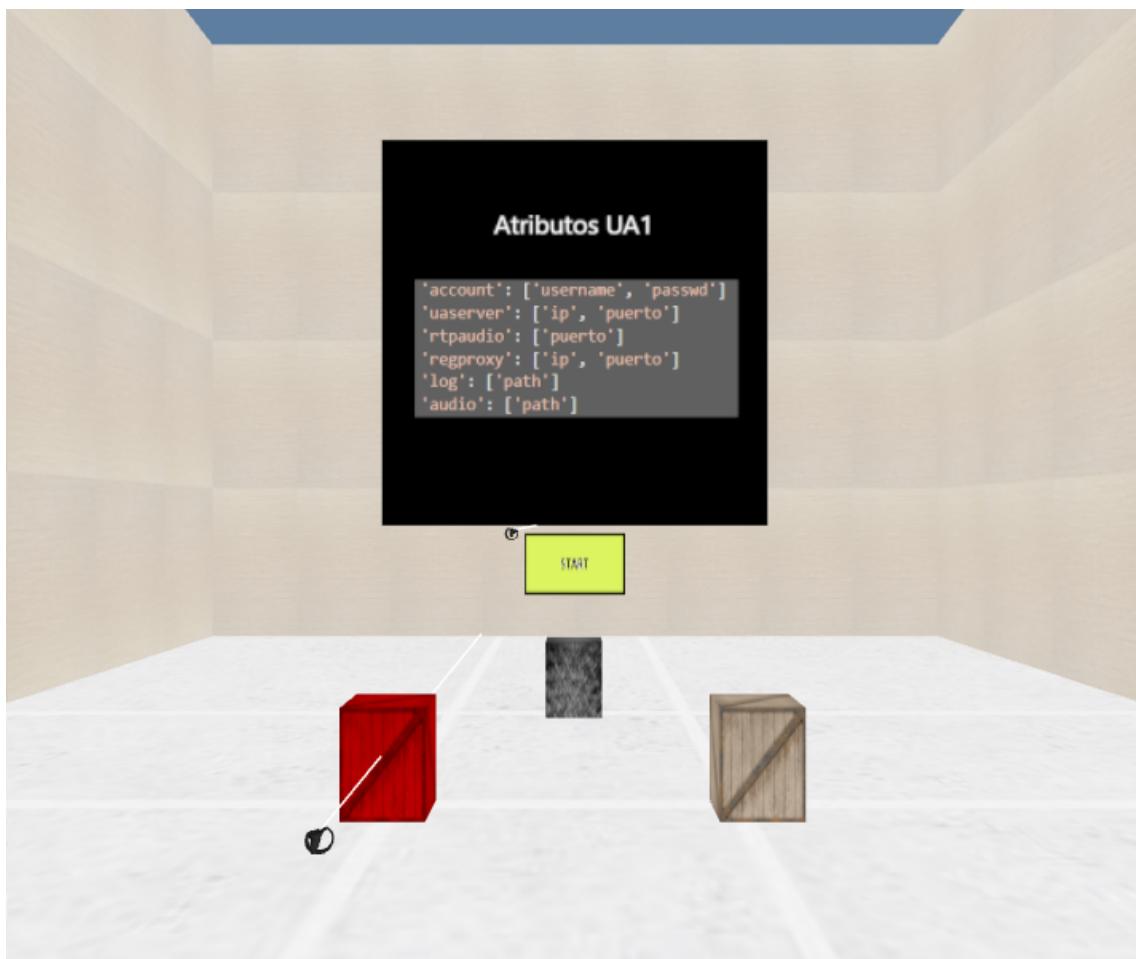


Figura 5.1: Estado inicial del UA1



Figura 5.2: Atributos UA1 con valores específicos

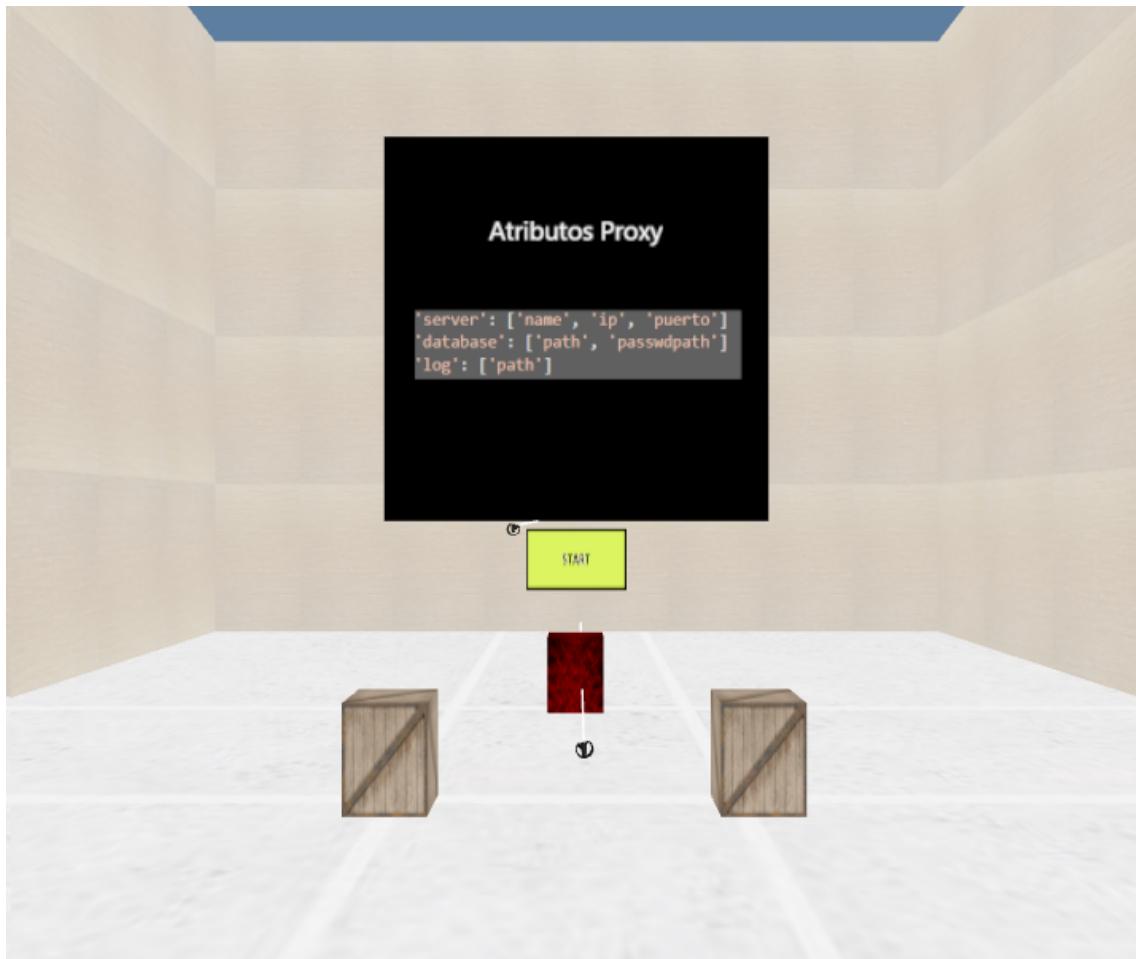


Figura 5.3: Estado inicial del Proxy

### 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']

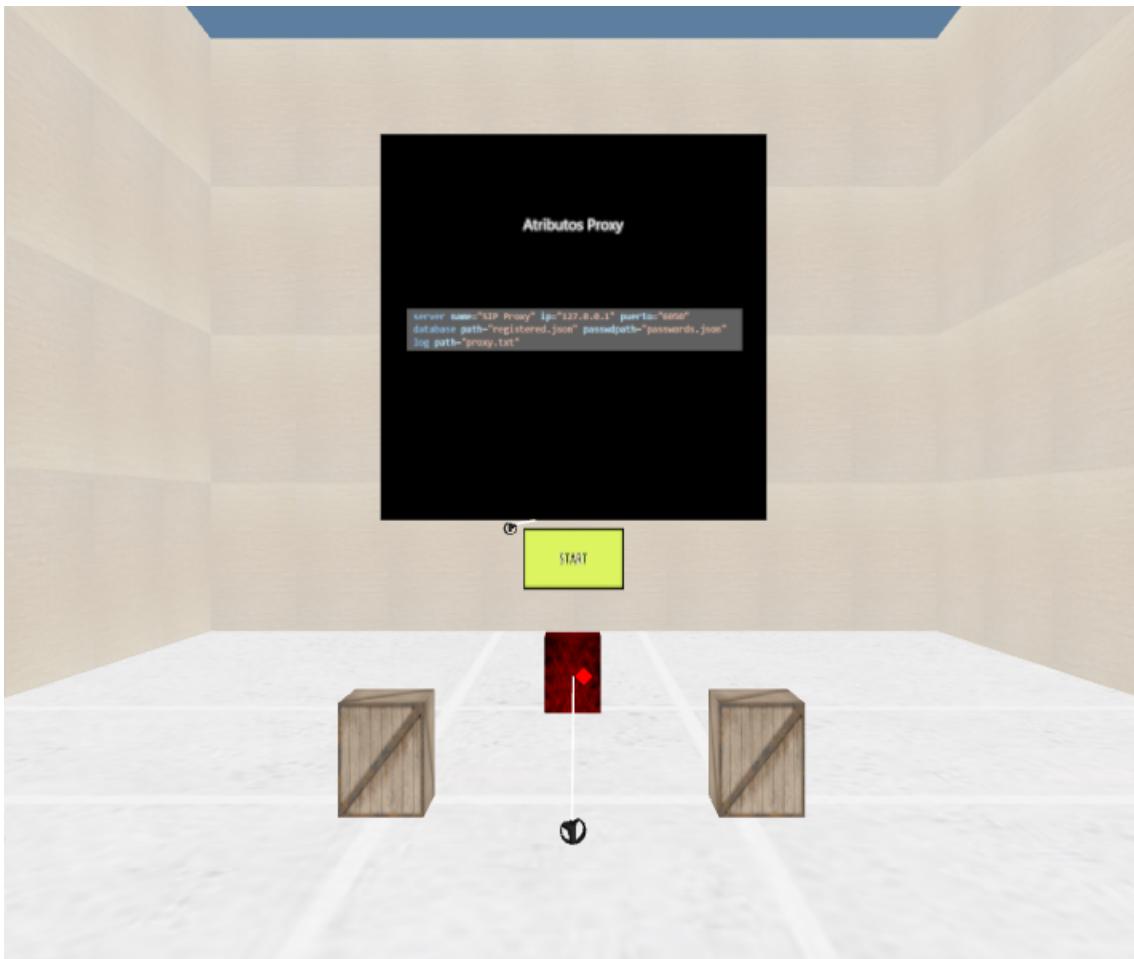


Figura 5.4: Atributos Proxy con valores específicos

### 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.

### 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']

### 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.



Figura 5.5: Estado inicial del UA2



Figura 5.6: Atributos UA2 con valores específicos

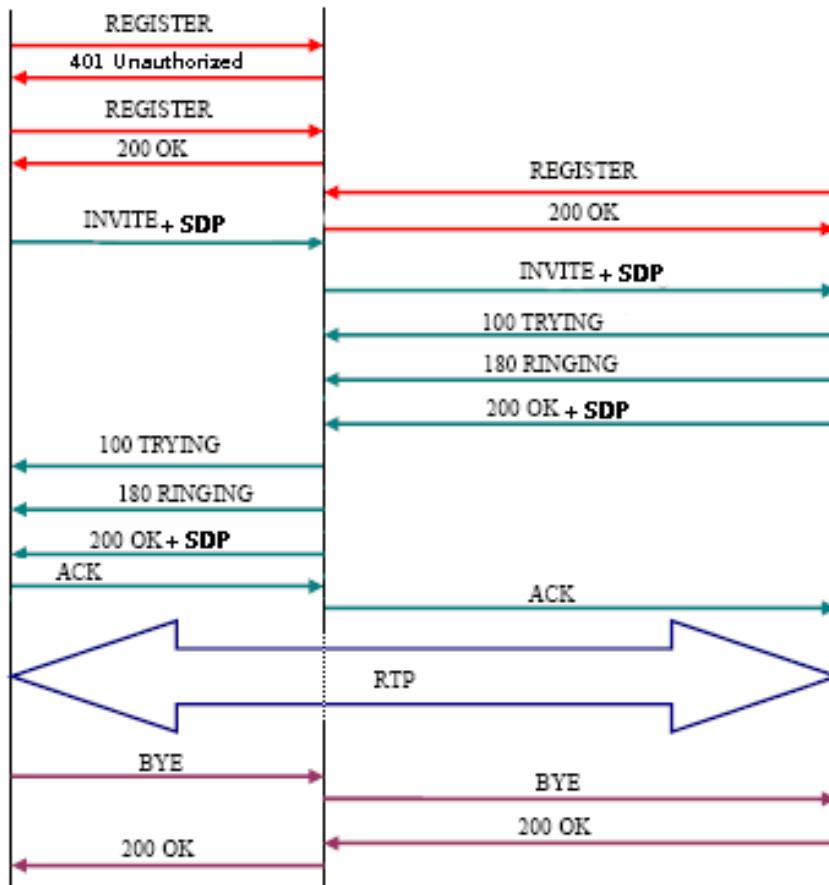


Figura 5.7: Secuencia del intercambio de paquetes

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.

### 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. Si no está autorizado, recibe un 401 Unauthorized, mostrado en la figura 5.9, lo que obliga al usuario (UA1) a realizar otro intento de REGISTER, el cual, es aceptado, y se confirma con una respuesta 200 OK, ilustrado en la figura 5.10, indicando que el registro se ha completado con éxito.

A continuación, se registra el usuario B (UA2) enviando un mensaje REGISTER al servidor, representado en la figura 5.11. Siendo este aceptado y confirmado con una respuesta de 200 OK, ilustrado en la figura 5.12.

La fase de llamada comienza con un INVITE acompañado de una descripción de la sesión (SDP), escenificado en la figura 5.13, que contiene los detalles para establecer la comunicación. Este mensaje es respondido con un 100 TRYING, evidenciado en la figura 5.14, seguido de 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. Cuando el usuario B (UA2) acepta la llamada, envía una respuesta 200 OK + SDP, representado en la figura 5.16, que contiene la confirmación y los detalles necesarios para establecer la comunicación. El usuario A (UA1) confirma recibir esta información con un ACK, mostrado en la figura 5.17.

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

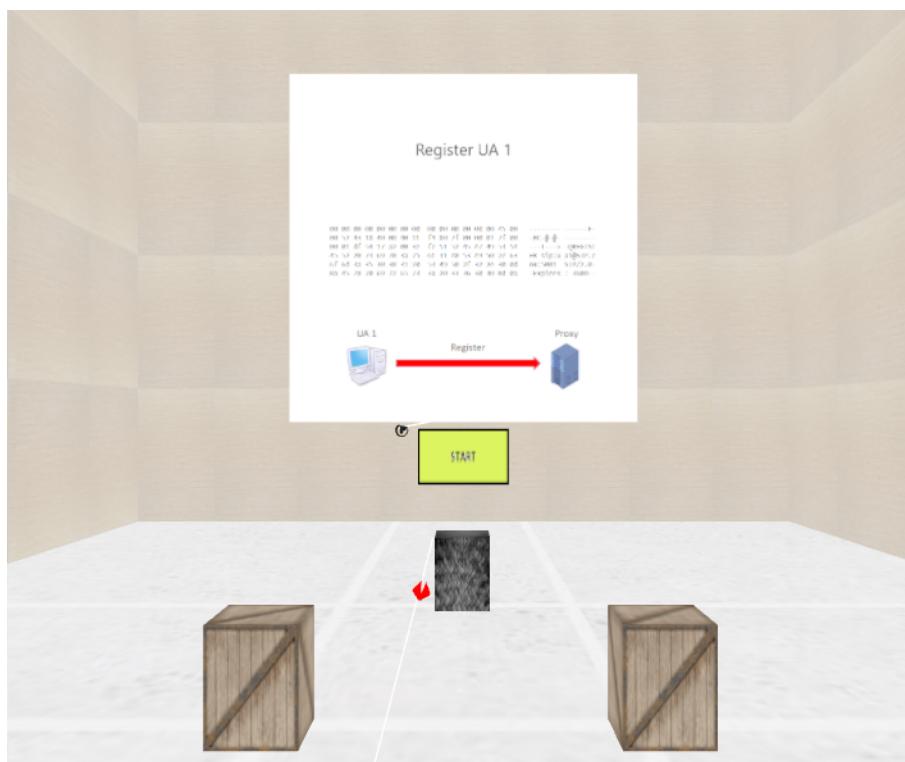


Figura 5.8: Mensaje de Registro Inicial de UA1

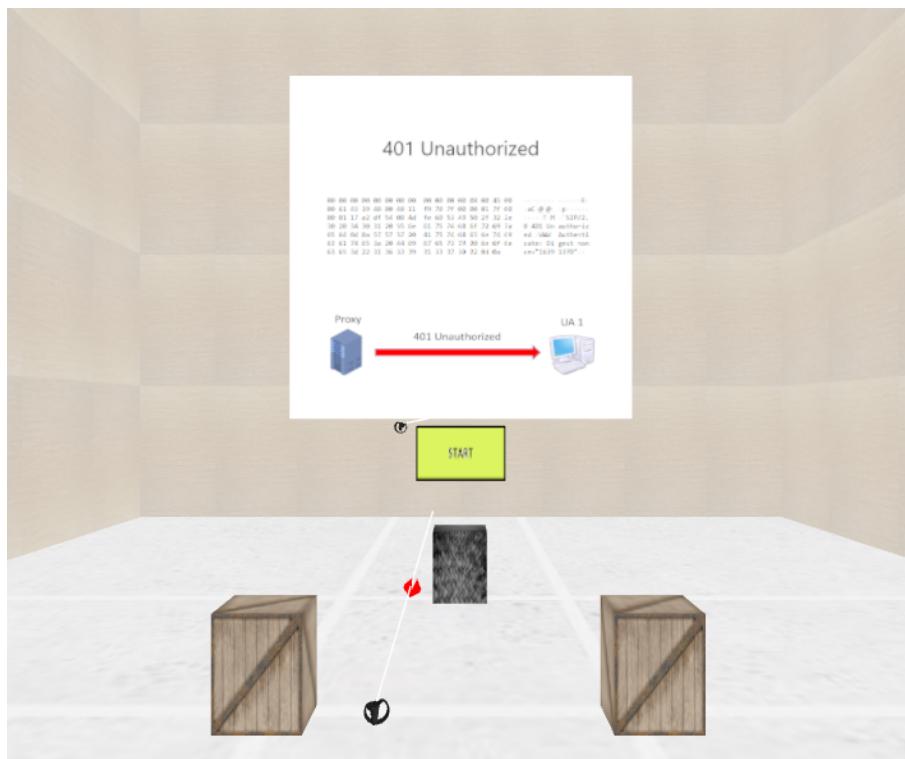


Figura 5.9: Respuesta 401 Unauthorized a UA1

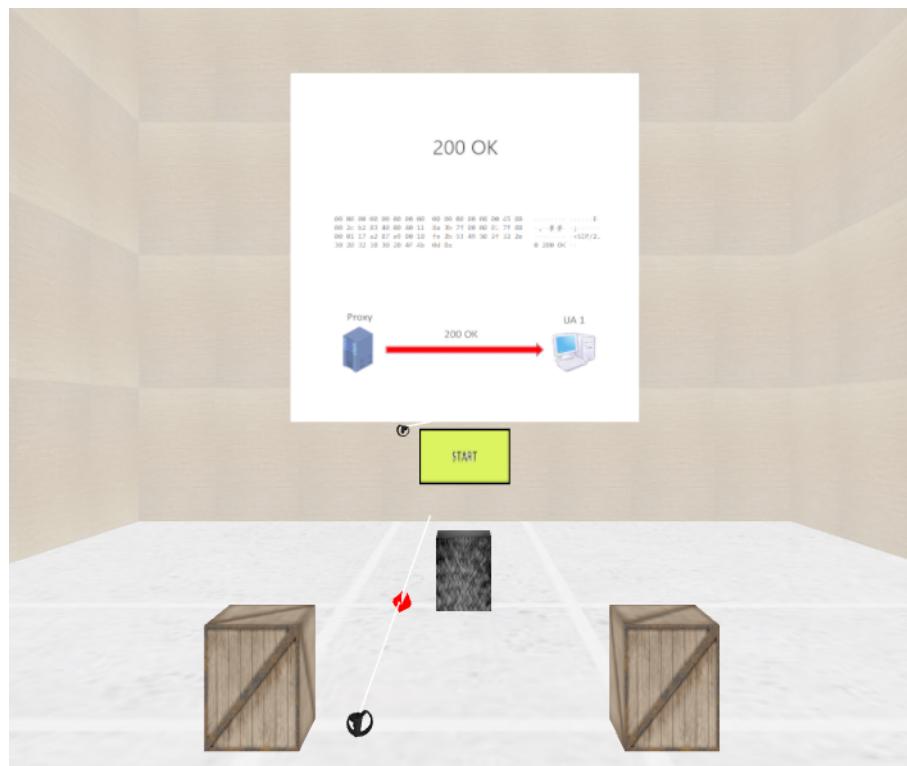


Figura 5.10: Registro de UA1 Exitoso con 200 OK

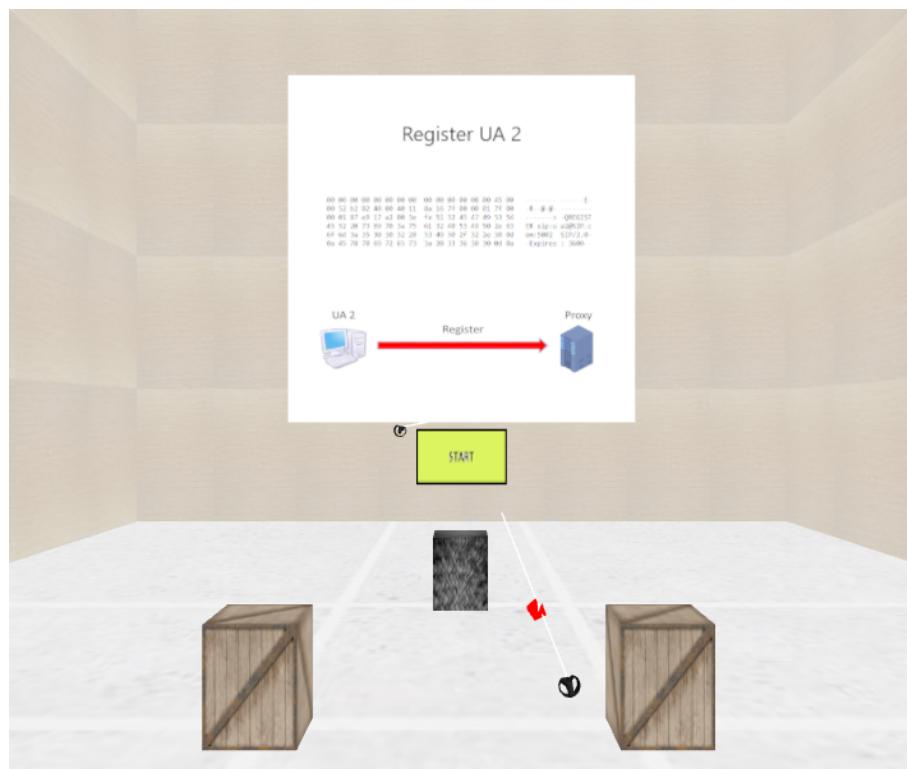


Figura 5.11: Mensaje de Registro Inicial de UA2

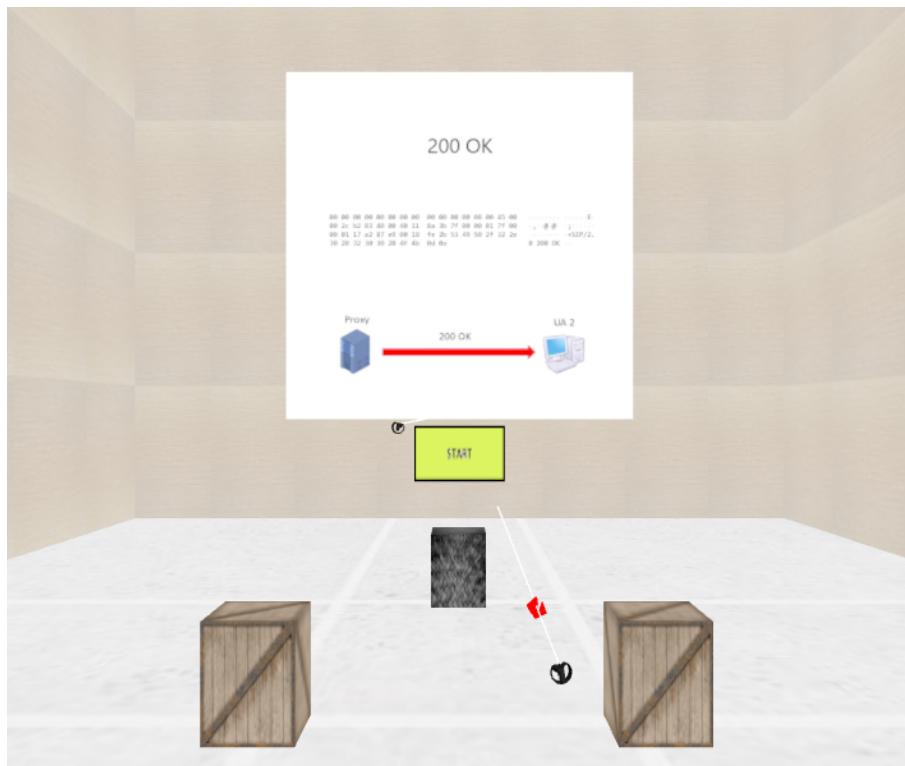


Figura 5.12: Registro de UA2 Exitoso con 200 OK

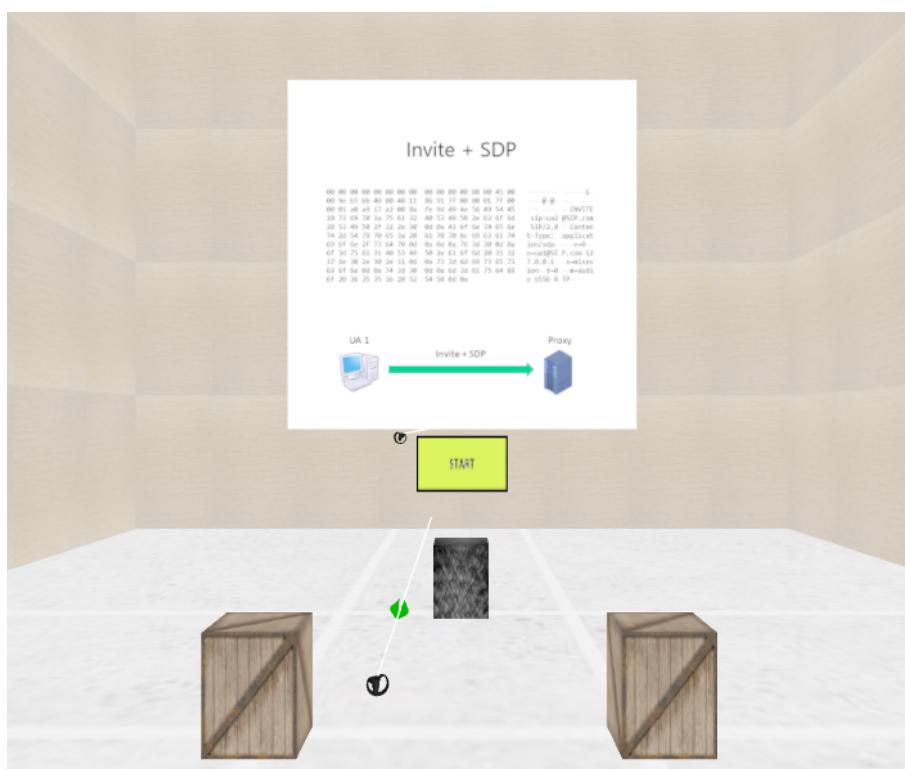


Figura 5.13: Inicio de Llamada con el Mensaje INVITE de UA1

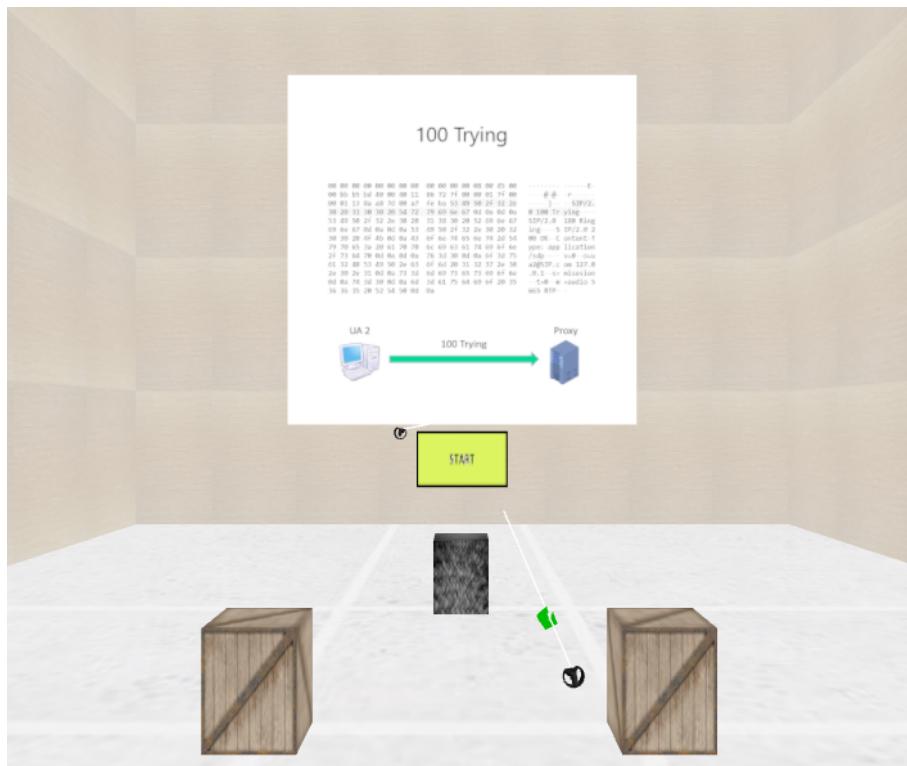


Figura 5.14: Respuesta TRYING de UA2 Indicando Procesamiento

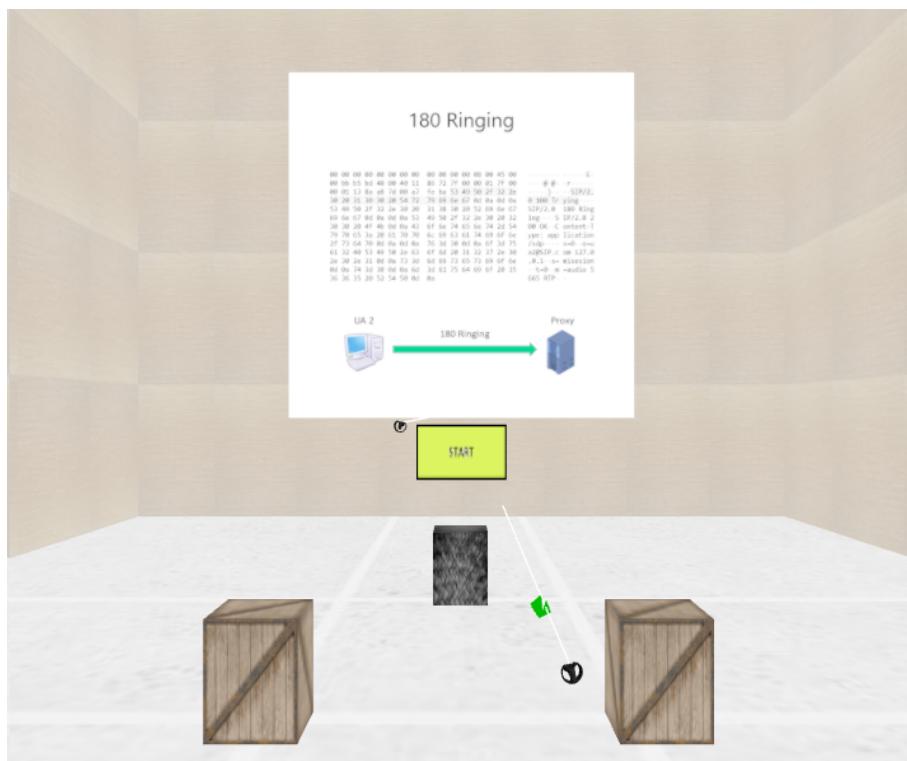


Figura 5.15: Alerta de Llamada con el Mensaje RINGING de UA2

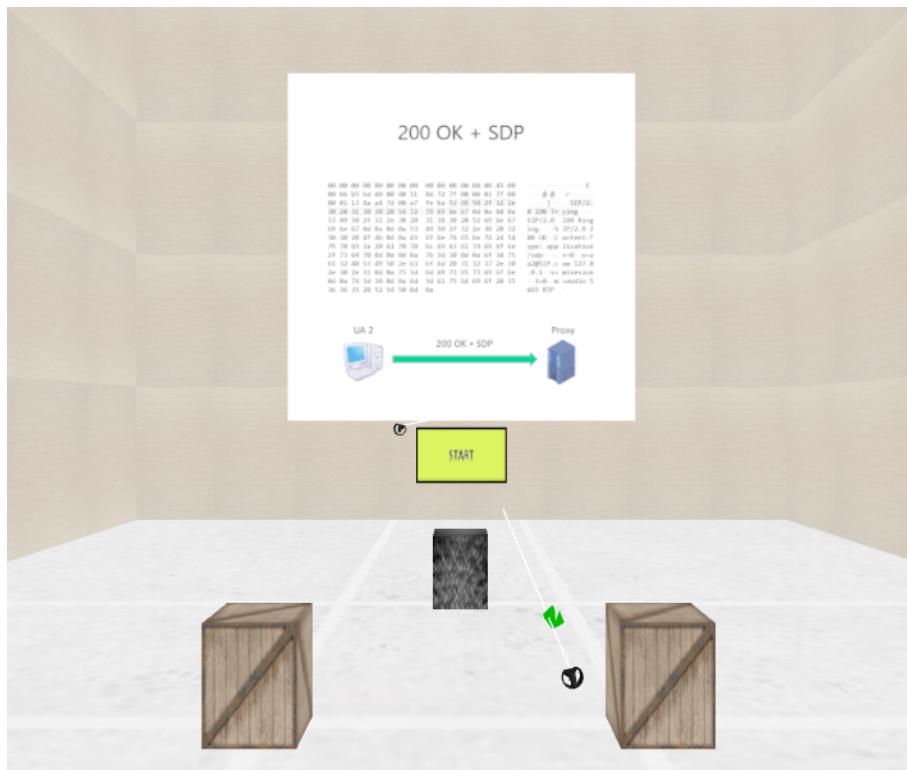


Figura 5.16: Confirmación de Llamada con 200 OK + SDP de UA2

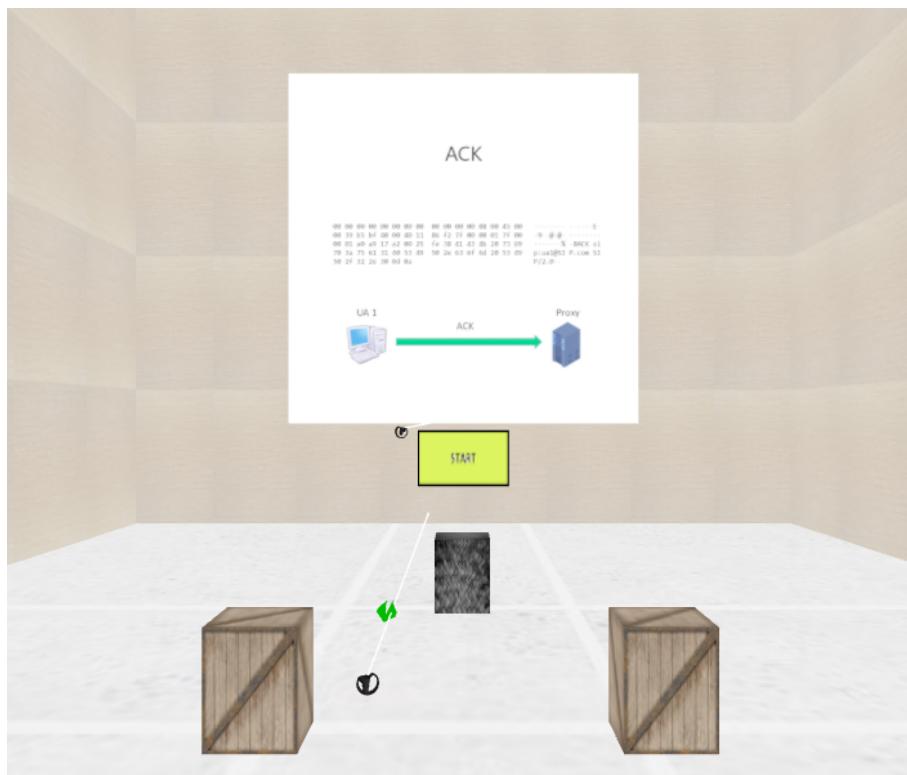


Figura 5.17: Reconocimiento ACK de UA1 Completando el Establecimiento de la Llamada

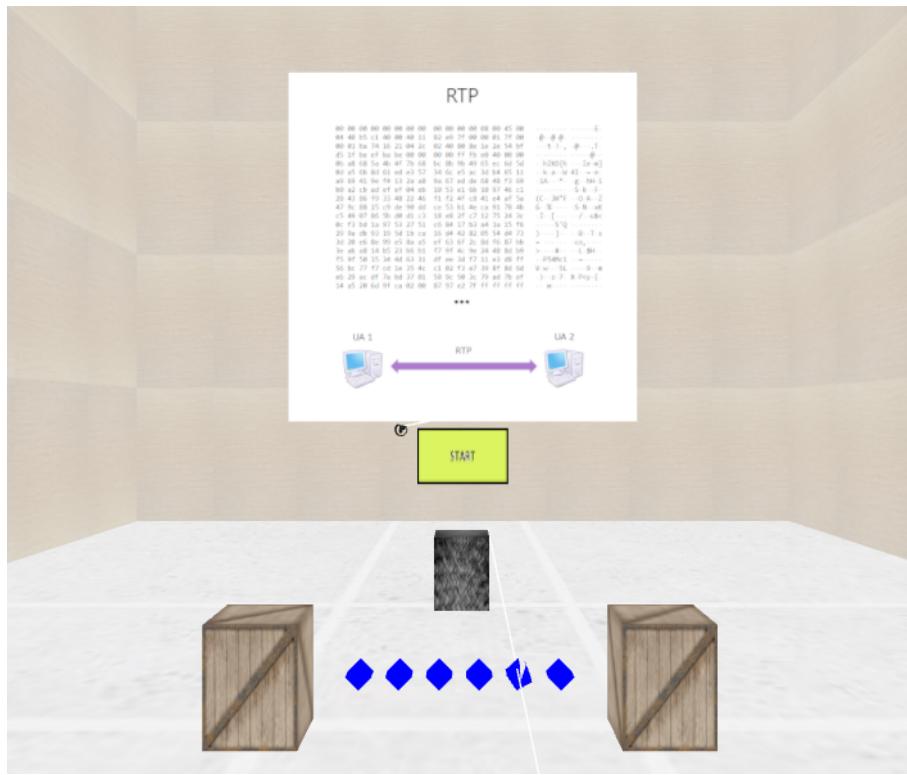


Figura 5.18: Transmisión de Datos Multimedia a través de Paquetes RTP

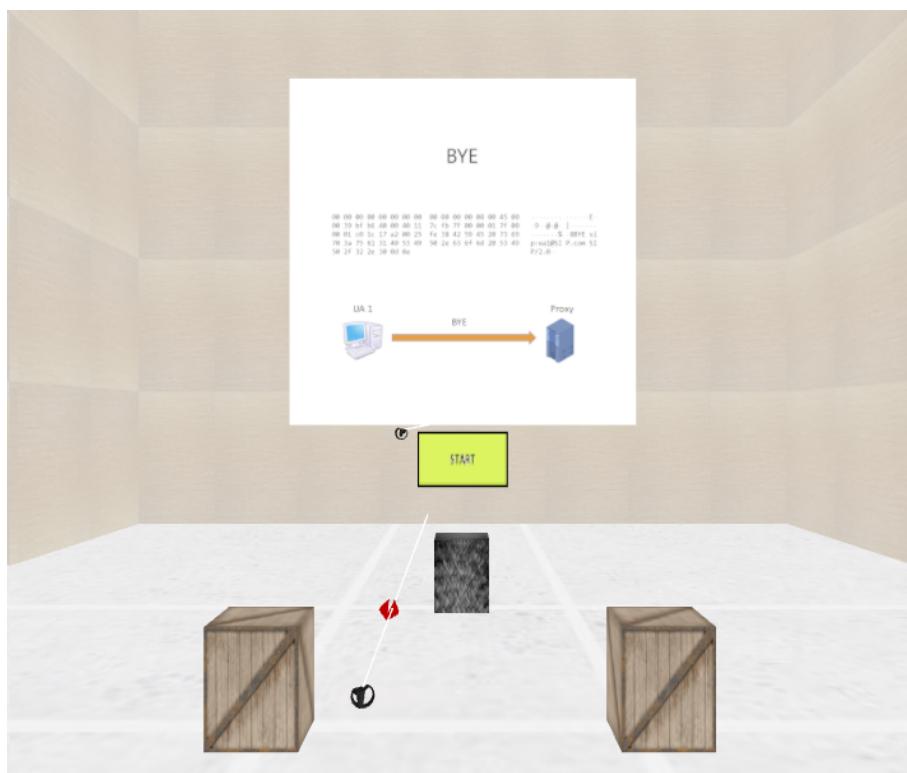


Figura 5.19: Mensaje BYE Enviado para Terminar la Llamada

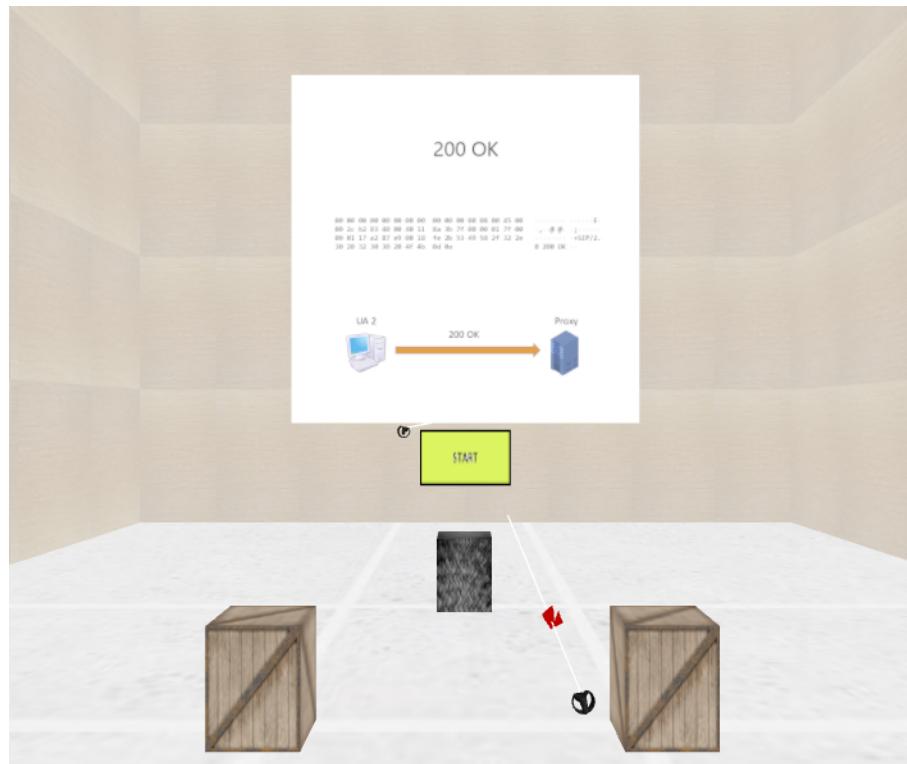


Figura 5.20: Confirmación del Fin de la Llamada con un 200 OK

# **Capítulo 6**

## **Conclusiones**

### **6.1. Consecución de objetivos**

Esta sección es la sección espejo de las dos primeras del capítulo de objetivos, donde se planteaba el objetivo general y se elaboraban los específicos.

Es aquí donde hay que debatir qué se ha conseguido y qué no. Cuando algo no se ha conseguido, se ha de justificar, en términos de qué problemas se han encontrado y qué medidas se han tomado para mitigar esos problemas.

Y si has llegado hasta aquí, siempre es bueno pasarle el corrector ortográfico, que las erratas quedan fatal en la memoria final. Para eso, en Linux tenemos aspell, que se ejecuta de la siguiente manera desde la línea de *shell*:

```
aspell --lang=es_ES -c memoria.tex
```

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

Aquí viene lo que has aprendido durante el Grado/Máster y que has aplicado en el TFG/TFM. Una buena idea es poner las asignaturas más relacionadas y comentar en un párrafo los conocimientos y habilidades puestos en práctica.

1. a

2. b

### **6.3. Lecciones aprendidas**

Aquí viene lo que has aprendido en el Trabajo Fin de Grado/Máster.

1. Aquí viene uno.
2. Aquí viene otro.

### **6.4. Trabajos futuros**

Ningún proyecto ni software se termina, así que aquí vienen ideas y funcionalidades que estaría bien tener implementadas en el futuro.

Es un apartado que sirve para dar ideas de cara a futuros TFGs/TFMs.

# **Apéndice A**

## **Manual de usuario**

Esto es un apéndice. Si has creado una aplicación, siempre viene bien tener un manual de usuario. Pues ponlo aquí.

