

Grado en Ingeniería Informática
2021-2022

Trabajo Fin de Grado

“Herramienta para el control y monitorización de entornos inmersivos”

Jesús Albarca Gordillo

Tutor

Alejandro Rey

Leganés, julio 2022



Esta obra se encuentra sujeta a la licencia Creative Commons
Reconocimiento – No Comercial – Sin Obra Derivada

ABSTRACT

In nowadays society, the use of Virtual Reality in aspects of our daily lives is becoming more and more common, mainly due to the recent reduction in the price of these technologies and their application to sectors such as video games or education. The development of new standalone VR devices brings many advantages (e.g. portability) but makes it difficult to use these devices for research purposes, where it is necessary to control and monitor the stimuli presented to users immersed in the virtual world. Achieving this remote control is possible, but it requires specialised programming knowledge and effort to develop a solution that is not very reusable in response to this need for control and monitoring that is so common among VR researchers.

To provide a solution to these problems, this paper presents a web-based tool that allows any non-technical experimenter to define operations that can be executed remotely on a server supporting an immersive environment. In this way, the tool makes it easier to control and monitor users and provides a more flexible solution for each researcher's needs. This document compiles an exhaustive analysis of the tool and compiles the indispensable information for this type of project.

Keywords: tool, control, monitoring, remote, virtual reality, augmented reality, autonomous.

RESUMEN

En la sociedad actual es cada vez más común el uso de la Realidad Virtual en aspectos de nuestro día a día, principalmente dado el reciente abaratamiento de estas tecnologías y su aplicación a sectores como los videojuegos o la educación. El desarrollo de nuevos dispositivos de RV standalone o autónomos traen consigo multitud de ventajas (e.g., portabilidad) pero dificultan la utilización de estos dispositivos para fines de investigación, donde es necesario controlar y monitorizar los estímulos que se presentan a los usuarios sumergidos en el mundo virtual. Lograr este control remoto es posible, pero requiere conocimiento especializado en programación y dedicar esfuerzos en desarrollar una solución poco reutilizable como respuesta a esta necesidad de control y monitorización tan común entre investigadores de RV.

Con el objetivo de proporcionar una solución a estos problemas, en este trabajo se presenta una herramienta web que permite a cualquier experimentador sin conocimientos técnicos definir operaciones que pueden ejecutarse de forma remota en un servidor que de soporte a un entorno inmersivo. De esta manera, la herramienta permite controlar y monitorizar de una manera más sencilla a los usuarios, así como proporciona una solución más flexible de cara a las necesidades de cada investigador. El presente documento recopila un análisis exhaustivo de la herramienta y recopila la información indispensable para este tipo de proyectos.

Palabras clave: herramienta, control, monitorización, remoto, realidad virtual, realidad aumentada, autónomo.

ÍNDICE GENERAL

ABSTRACT	III
RESUMEN	V
ÍNDICE GENERAL	VI
ÍNDICE DE TABLAS	IX
ÍNDICE DE ILUSTRACIONES	XI
1. INTRODUCCIÓN	1
1.1 Motivaciones	1
1.2 Objetivos.....	2
1.3 Estructura del documento	3
2. ESTADO DEL ARTE	4
2.1 Tecnologías que permiten el desarrollo de una aplicación de ejecución de funciones remotas.	4
2.1.1 gRPC.....	4
2.1.2 JAX-RPC	6
2.2 Herramientas similares	7
2.2.1 ImmVis: Bridging Data Analytics and Immersive Visualisation	7
2.2.2 Improved Knowledge from Data: Building an Immersive Data Analysis Platform	8
2.2.3 Welicit: A Wizard of Oz Tool for VR Elicitation Studies	9
2.2.4 Toggle toolkit: A tool for conducting experiments in unity virtual environments	11
2.3 Tecnologías que permiten el desarrollo de una aplicación web.	11
2.3.1 JEE.....	11
2.3.2 JS	13
2.3.3 Express	13
2.3.4 Django	14
2.3.5 PHP.....	14
2.4 Bases de datos.....	14
2.4.1 Bases de datos relacionales	15
2.4.2 Bases de datos no relacionales.....	16
2.5 Conclusiones del estado del arte.....	17
3. ANÁLISIS DEL PROBLEMA	19

3.1 Alcance del proyecto	19
3.1.1 Casos de uso	19
3.1.2 Requisitos	22
3.1.2.1 Requisitos de usuario.....	22
3.1.2.2 Requisitos de sistema	24
3.2 Implementación	26
4. EVALUACIÓN.....	38
5. PLANIFICACIÓN DEL PROYECTO.....	41
5.1 Estudio de metodologías para el desarrollo de proyectos.	41
5.2 Planificación inicial	42
5.3 Planificación real	43
6. ENTORNO SOCIOECONÓMICO	45
6.1 Presupuesto	45
6.1.1 Desglose presupuestario	45
6.2 Impacto Socioeconómico	47
7. MARCO REGULADOR.....	49
7.1 Ley de Protección de datos	49
7.2 Licencias	50
8. CONCLUSIONES	52
8.1 Retrospectiva	52
8.2 Trabajo futuro	52
8.3 Conclusiones personales.....	53
9. SUMMARY	55
9.1 Motivation and objectives	55
9.2 State of the art.....	57
9.2.1 Grpc	57
9.2.2 Express	57
9.2.3 MongoDB	58
9.2.4 Conclusions on the state of the art.....	59
9.3 Problem analysis.....	60
9.3.1 Implementation.....	60
9.4 Project plan	64
9.4.1 Project methodology chosen.....	64

9.4.2 Scheduling	64
9.5 Socio-economic environment	65
9.5.1 Budget.....	65
9.5.2 Socio-economic impact	65
9.6 Legal framework.....	66
9.7 Conclusions	67
9.7.1 Retrospective	67
9.7.2 Future work	67
REFERENCIAS	68

ÍNDICE DE TABLAS

Tabla 1 Formato tabla casos de uso	20
Tabla 2 CU-01	21
Tabla 3 CU-02	21
Tabla 4 CU-03	21
Tabla 5 CU-04	21
Tabla 6 CU-06	21
Tabla 7 Formato tabla requisitos	22
Tabla 8 RU-01	22
Tabla 9 RU-02	22
Tabla 10 RU-03	23
Tabla 11 RU-04	23
Tabla 12 RU-05	23
Tabla 13 RU-06	23
Tabla 14 RU-07	23
Tabla 15 RU-08	23
Tabla 16 RS-01	23
Tabla 17 RS-02	24
Tabla 18 RS-03	24
Tabla 19 RS-04	24
Tabla 20 RS-05	24
Tabla 21 RS-06	24
Tabla 22 RS-07	24
Tabla 23 RS-08	25
Tabla 24 RS-09	25
Tabla 25 RS-10	25
Tabla 26 RS-11	25
Tabla 27 RS-12	25
Tabla 28 RS-13	25
Tabla 29 RS-14	26
Tabla 30 Formato tabla pruebas	38
Tabla 31 P-01	38
Tabla 32 P-02	38
Tabla 33 P-03	39
Tabla 34 P-04	39
Tabla 35 P-05	39
Tabla 36 P-06	39
Tabla 37 P-07	39
Tabla 38 Matriz trazabilidad entre pruebas y requisitos	40
Tabla 39 Planificación inicial	42
Tabla 40 Planificación real	43
Tabla 41 Desviación entre planificación inicial y real	44

Tabla 42 Tiempo total de desarrollo del proyecto	45
Tabla 43 Costes recursos software	46
Tabla 44 Costes recursos hardware	46
Tabla 45 Costes recursos humanos	47
Tabla 46 Costes indirectos	47
Tabla 47 Resumen de costes	47

ÍNDICE DE ILUSTRACIONES

Ilustración 1 gRPC protocol [12]	5
Ilustración 2 gRPC message example [12]	5
Ilustración 3 gRPC file proto example [12]	6
Ilustración 4 ImmVis Dataset vs Time [17]	8
Ilustración 5 Interacción del usuario propuesta [23]	9
Ilustración 6 Arquitectura Welicit [25]	10
Ilustración 7 Ejemplo mongodb documento persona [51]	17
Ilustración 8 Casos de uso genéricos de la herramienta	20
Ilustración 9 Directorio raíz herramienta desarrollada	27
Ilustración 10 Ejemplo mensaje request	29
Ilustración 11 Ejemplo mensaje reply	29
Ilustración 12 Ejemplo mensajes actuales	30
Ilustración 13 Ejemplo crear servicio	30
Ilustración 14 Ejemplo servicios actuales	31
Ilustración 15 Ejemplo ejecutar servicio	32
Ilustración 16 Página home	33
Ilustración 17 Página ejecutar servicios	34
Ilustración 18 Página crear servicio	35
Ilustración 19 Página mostrar servicios	36
Ilustración 20 Página mostrar mensajes	37
Ilustración 21 Diagrama de Gantt (Planificación inicial)	43
Ilustración 22 Diagrama de Gantt (Planificación real)	44

1. INTRODUCCIÓN

La Realidad Virtual consiste en un entorno de escenas y objetos simulados generados mediante tecnología informática los cuales crean al usuario la sensación de estar inmerso en él [1]. Para que el usuario pueda contemplar dicho entorno, lo más común es utilizar un casco de realidad virtual o HMD (Head-mounted display), aunque también existen otras posibilidades como hacer uso de CAVEs, que son salas dónde se proyecta el contenido 3D en las paredes y el suelo [2], aunque este último método resulta tedioso debido a que la mayoría de personas no puede llegar a disponer de los medios necesarios para servirse de una sala exclusivamente reservada para este fin.

1.1 Motivaciones

“La realidad virtual es contenido digital que se puede disfrutar en un espacio 3D totalmente inmersivo mediante el uso de un auricular VR o HMD. El objetivo de los auriculares VR es que los usuarios se sientan completamente inmersos en un entorno simulado, aislados del mundo real” [3]. Los auriculares VR pueden ser independientes (HDM) o necesitar de una conexión con la computadora mediante un cable HDMI o y/o USB. Estos últimos, al estar conectados físicamente al ordenador, cuentan con la ventaja de poder acceder localmente a toda la información disponible en las gafas. Sin embargo, si se utilizan se limita el número de usuarios que pueden interactuar con el entorno inmersivo en base al número de ordenadores disponibles y la realización de experimentos fuera de un entorno específico de trabajo se ve dificultada, debido a que resulta tedioso transportar las máquinas y dispositivos de un lugar a otro.

La tendencia actual entre las empresas e investigadores es usar las gafas standalone, debido a que se pueden utilizar en cualquier entorno, no es necesario que estén conectadas a un ordenador y permiten realizar experimentos con un número mayor de usuarios. Existen numerosos modelos en el mercado. Una de las más utilizadas y avanzadas tecnológicamente para realidad virtual son las Oculus Quest 2 [4], desarrolladas por Meta y con un sistema operativo basado en Android. Las características más relevantes de este dispositivo son que permiten definir una zona de juego que incorpora un sistema guardián cuya función es evitar posibles obstáculos y, además, es posible transmitir el contenido en algunos teléfonos compatibles, además su reducido precio ha logrado acercar al público general la Realidad Virtual [5]. Otro dispositivo para RV con un sistema operativo basado en Android son las pico neo3 pro [6] cuya función de salida DisplayPort permite presentar el contenido de las gafas en cualquier pantalla o televisor con un adaptador USB a HDMI. En el campo de la realidad mixta destaca Hololens 2 [7], un dispositivo holográfico y autónomo desarrollado por Microsoft que cuenta con Windows como sistema operativo.

Sin embargo, los dispositivos inalámbricos necesitan de una capa de software adicional para monitorizar al usuario que se encuentra inmerso en un sistema de RV y, además, la mayoría de los dispositivos inmersivos no permiten acceder a la información que queda registrada en las gafas hasta que el usuario deje de utilizarlas, debido a que este tipo de

lentes tiene su propio sistema de archivos. Otro problema de utilizar este tipo de dispositivos es que se dificulta la realización de pruebas, debido a que es necesario crear una compilación (por ejemplo, un APK) y ejecutarlo en el propio dispositivo [8].

Por tanto, para tratar de mitigar algunos de los inconvenientes que introducen los nuevos modelos de HMD standalone así como obtener, modificar o configurar elementos del entorno en el que se encuentra inmerso el usuario cuando está utilizando este tipo de dispositivos inalámbricos, es necesario desarrollar una herramienta que permita el control y monitorización del entorno inmersivo de forma remota. Además, dado que este tipo de herramienta puede ser potencialmente aplicada a diversos entornos, con objetivos de investigación o prototipado de sistemas inmersivos; esta herramienta debe ser lo suficientemente flexible como para permitir al usuario adaptarla a su caso de uso concreto y minimizar el esfuerzo o conocimiento técnico necesario para su uso.

1.2 Objetivos

El objetivo principal del presente trabajo es desarrollar una herramienta web que permita ejecutar comandos remotos en una aplicación servidor que de soporte a un sistema inmersivo de Realidad Virtual. La aplicación cliente podrá tanto generar nuevos servicios como ejecutar los existentes sin necesidad de conocimientos técnicos. Para llevar a cabo con éxito el objetivo principal es necesario:

- Automatizar la creación de servicios remotos.
- Diseñar y adaptar una interfaz gráfica para que cualquier usuario sin conocimientos técnicos pueda utilizarla.
- Almacenar los servicios en una base de datos.
- Establecer la comunicación entre una aplicación cliente y el servidor para la exitosa ejecución de los servicios remotos.

Además, para la correcta realización del presente documento y la herramienta web, será necesario cumplir los siguientes objetivos:

- Investigar y analizar las tecnologías actuales para el desarrollo de aplicaciones web y aplicaciones remotas.
- Investigar y analizar las metodologías de desarrollo de software actuales para este tipo de proyecto y aplicarlas.
- Exponer las conclusiones y resultados obtenidos.

1.3 Estructura del documento

El presente documento consta de 9 secciones. A continuación, se detalla el contenido de cada una de estas.

Capítulo 1 - Introducción: en este capítulo, inicialmente, se definen algunos conceptos importantes para que los futuros lectores del documento tengan claro el contexto en el que se encuentran. En la subsección 1.1 se plantean los problemas actuales del uso de HMD en entornos inmersivos y se exponen las motivaciones para llevar a cabo el proyecto. En la subsección 1.2 se listan los objetivos a lograr. Por último, en el apartado 1.3 se especifica la estructura del documento.

Capítulo 2 - Estado del arte: este capítulo consta de 4 subsecciones. En la subsección 2.1 se analizan varias tecnologías que permiten el uso de llamadas a procedimiento remoto (RPC) en aplicaciones web. En la 2.2 se detallan herramientas con fines similares a la de la herramienta desarrollada. En la 2.3 se explican algunas tecnologías actuales para el desarrollo de aplicaciones web. En 2.4 se analizan las bases de datos relacionales y no relacionales. Finalmente, en 2.5 se realiza un análisis de todos los apartados anteriores.

Capítulo 3 - Análisis del problema: en este capítulo se realiza un análisis del alcance del proyecto en 3.1 especificando casos de uso y requisitos de usuario y funcionales. Al final del capítulo, en 3.2 se detallan las cuestiones relacionadas con la implementación.

Capítulo 4 - Evaluación: en esta sección se describen las pruebas realizadas para verificar el correcto funcionamiento del sistema desarrollado.

Capítulo 5 - Planificación del proyecto: inicialmente, en el apartado 5.1 se realiza un análisis de varias metodologías para el desarrollo de proyectos y se justifica la elegida. Posteriormente, en el apartado 5.2 se expone la planificación ideal del proyecto antes de ser desarrollado. Finalmente, en 5.3 se muestra la planificación una vez realizado el proyecto y se lleva a cabo una comparación entre ambas planificaciones.

Capítulo 6 - Entorno socioeconómico: en este capítulo, en el apartado 6.1 se detallan los costes totales para la elaboración del presente documento y la herramienta desarrollada. El apartado 6.2 trata de medir el impacto de la herramienta.

Capítulo 7 - Marco regulador: en este capítulo, en el apartado 7.1 se discute sobre la aplicabilidad de la Ley de Protección de datos para el proyecto. En 7.2 se exponen las distintas licencias necesarias para utilizar cada una de las herramientas necesarias para la elaboración de este.

Capítulo 8 - Conclusiones: en este capítulo inicialmente, en el apartado 8.1 se comparan los objetivos logrados con los iniciales. Seguidamente, en 8.2 se detallan posibles mejoras para la herramienta desarrollada. Finalmente, en 8.3 se exponen conclusiones personales del autor.

Capítulo 9 - Summary: este capítulo está formado por un resumen en inglés del presente documento.

2. ESTADO DEL ARTE

En esta sección se analizan, describen e investigan las distintas herramientas y tecnologías utilizadas o que es necesario conocer para realizar el proyecto que describe el presente documento.

2.1 Tecnologías que permiten el desarrollo de una aplicación de ejecución de funciones remotas.

“La llamada a procedimiento remoto (RPC) es un protocolo de comunicación de software que un programa puede utilizar para solicitar un servicio a un programa situado en otro ordenador de otra red sin tener que entender los detalles de la red” [9]. RPC sigue el modelo cliente-servidor [10]. En primer lugar, la aplicación cliente envía un mensaje de solicitud que contiene los parámetros del procedimiento remoto a través de la red a la aplicación servidor y se queda esperando un mensaje de respuesta. En segundo lugar, el servidor, quien siempre está a la espera de una solicitud de mensaje la recibe, ejecuta el procedimiento y envía a la aplicación cliente el mensaje respuesta esperado que contiene el resultado del procedimiento ejecutado [11].

2.1.1 gRPC

gRPC [12] es un software de código abierto de alto rendimiento desarrollado por Google que utiliza un framework de llamadas a procedimientos remotos (RPC) el cual puede ejecutarse en cualquier entorno. Si se utiliza gRPC, una aplicación cliente puede llamar directamente a un método de una aplicación servidor (la cual puede encontrarse en otra máquina) como si fuera un objeto local, facilitando así la creación tanto de aplicaciones como de servicios distribuidos.

Al igual que en muchos sistemas RPC, gRPC está basado en la idea de definir un servicio, especificando antes aquellos métodos que podrán ser llamados remotamente, los cuales se caracterizan por sus parámetros y lo que devuelven. El servidor implementa esta interfaz y ejecuta un servidor gRPC para gestionar las llamadas del cliente. El cliente utiliza un stub [13] que proporciona los mismos métodos que se han definido previamente en el servidor.

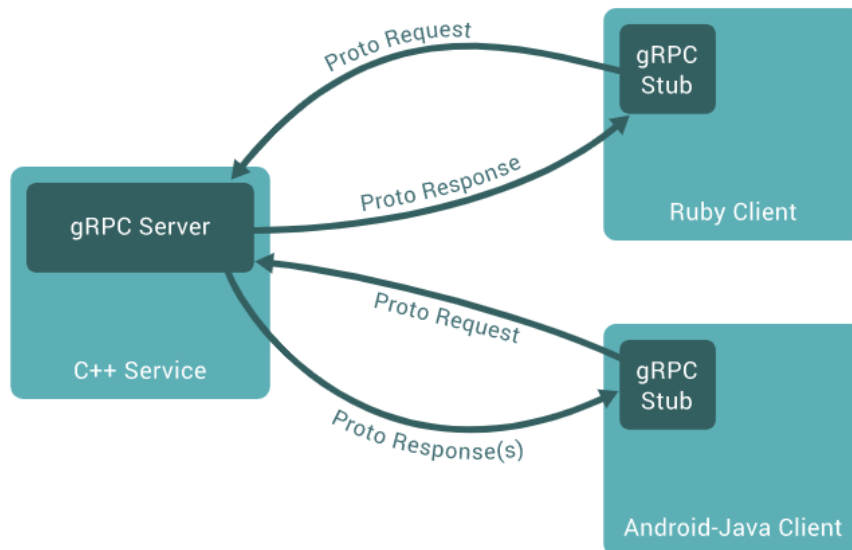


Ilustración 1 gRPC protocol [12]

Tanto los clientes como los servidores que usen gRPC pueden comunicarse entre sí en una gran variedad de entornos, desde cualquier escritorio hasta servidores de Google. Las últimas API de Google tienen versiones de gRPC de sus interfaces, lo que permite incorporar fácilmente funcionalidades de Google a las aplicaciones que utilicen esta tecnología. Cliente y servidor pueden escribirse en cualquiera de los lenguajes compatibles con gRPC: C#, C++, Dart, Go, Java, Kotlin, Node, Objective-C, PHP, Python y Ruby.

Por defecto, gRPC utiliza Protocol Buffers [14], un robusto mecanismo de código abierto desarrollado por Google para serializar datos estructurados, aunque puede utilizarse con otros formatos de datos como JSON. Al trabajar con búferes de protocolo es necesario definir la estructura de datos a serializar en un archivo proto: este es un archivo de texto ordinario con la extensión “.proto”. Los datos del buffer de protocolo se definen como mensajes, donde cada mensaje es un pequeño registro lógico de información que contiene una serie de campos.

```
message Person {
    string name = 1;
    int32 id = 2;
    bool has_ponycopter = 3;
}
```

Ilustración 2 gRPC message example [12]

Una vez se han especificado las estructuras de datos es necesario utilizar el compilador de búferes de protocolo “protoc” para generar clases de acceso a datos en el lenguaje que se haya elegido.

En el archivo proto se definen los servicios gRPC los cuales cuentan con parámetros de métodos de RPC y necesitan de tipos de retorno especificados como mensajes de búfer de protocolo:

```
// The greeter service definition.
service Greeter {
    // Sends a greeting
    rpc SayHello (HelloRequest) returns (HelloReply) {}
}

// The request message containing the user's name.
message HelloRequest {
    string name = 1;
}

// The response message containing the greetings
message HelloReply {
    string message = 1;
}
```

Ilustración 3 gRPC file proto example [12]

gRPC utiliza protoc con un plugin especial de gRPC con el que se genera un código a partir del archivo .proto para el cliente y servidor, además del código del buffer de protocolo para serializar, poblar y recuperar los tipos de mensajes definidos. Este buffer de protocolo cuenta con distintas versiones. La versión actual por defecto es proto2 pero es recomendable usar proto3, debido a que permite utilizar toda la gama de lenguajes soportados por gRPC.

2.1.2 JAX-RPC

JAX-RPC es una API capaz de desarrollar servicios web y clientes que utilizan llamadas a procedimientos remotos (RPC) junto con XML [15]. Una de sus ventajas es la capacidad de que un cliente JAX-RPC pueda comunicarse con otro servicio web aunque este esté programado en otro lenguaje o desarrollado en otra plataforma. Asimismo, un cliente desarrollado en otra plataforma y codificado en otro lenguaje puede comunicarse con un servicio JAX-RPC.

Para desarrollar un servicio [16], es necesario crear dos clases. Una interfaz que contendrá los métodos y otra clase con la implementación de estos. Por otro lado, se requieren dos XML. Un archivo de configuración que usarán tanto cliente como servidor con instrucciones para la creación de artefactos y un web.xml para especificar servlets responsables de enviar solicitudes.

2.2 Herramientas similares

2.2.1 ImmVis: Bridging Data Analytics and Immersive Visualisation

El artículo *ImmVis: Bridging Data Analytics and Immersive Visualisation* [17], expone un entorno de trabajo de código abierto que permite a las aplicaciones de IA [18] beneficiarse de las capacidades de las bibliotecas de Python [19] para el análisis de datos masivos. Los autores de dicho artículo analizan distintas herramientas de visualización inmersiva como iViz [20], arquitecturas para permitir la exploración de datos utilizando la RV y la Realidad Aumentada (RA) como Glance [21], etc. Sin embargo, la mayoría de las herramientas mencionadas, aunque sí que representan un avance significativo en el estudio del entorno inmersivo no ofrecen herramientas o APIs para llevar a cabo tareas como el manejo de datos, el cálculo de la correlación entre las dimensiones o algoritmos de agrupación.

Los autores implementan posibles soluciones para integrar el análisis de datos disponibles en Python con visualizaciones de realidad virtual en Unity [22]. Prueban herramientas como como "IronPython5 " y "pythonnet6" pero provocan que el entorno de Unity se bloquee con frecuencia. También se intenta implementar un sistema de comunicación de red local que comunique Python con Unity, pero hay problemas con que ambos lenguajes de programación entiendan un formato de mensaje común. Para solucionar estos inconvenientes, los autores decidieron utilizar gRPC. Este framework permite que el código sea de código abierto, que sea posible generalizar la creación de servicios y una comunicación fluida entre distintos lenguajes de programación. El framework ImmVis está formado por un servidor escrito en Python y distintas bibliotecas clientes escritas en distintos lenguajes de programación. El servidor fue desarrollado con un enfoque modular con objetivos como mejorar la legibilidad del código, facilitar el proceso de añadir nuevas funciones y permitir la reutilización de componentes.

En el artículo se realiza un análisis sobre la latencia de la comunicación entre cliente y servidor. En la siguiente ilustración se puede observar los resultados de las mediciones basadas en la cantidad de tiempo necesaria para que el cliente obtenga todos los datos del servidor. Para un conjunto de datos compuesto por 50000 filas tarda una media de 21s en cargarse y mostrarse para el usuario. Si se cargan estos mismos datos, utilizando el mismo servidor, pero como cliente un dispositivo Oculus Quest, se reducen el tiempo de carga hasta 14,5 segundos.

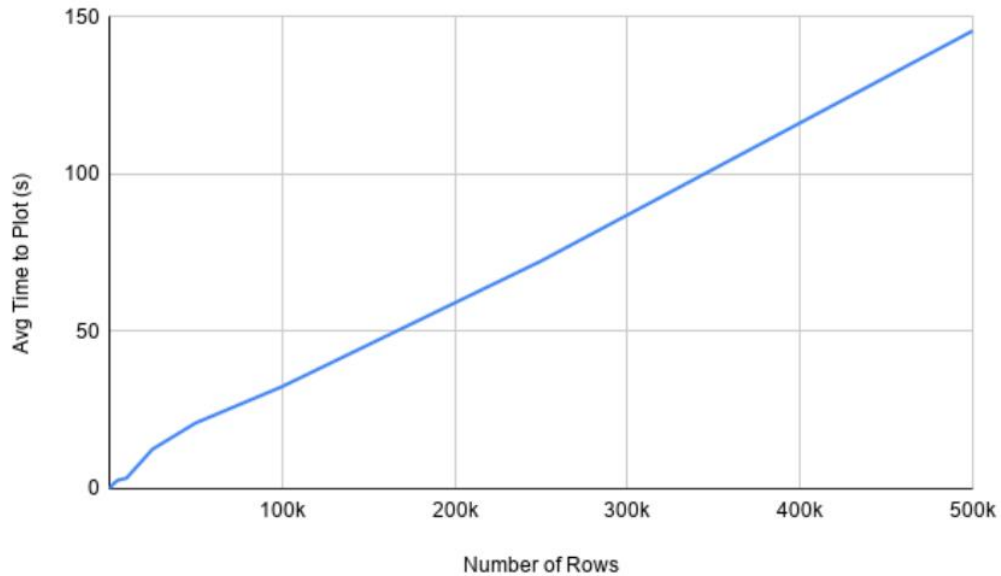


Ilustración 4 ImmVis Dataset vs Time [17]

Como conclusión, ImmVis permite optimizar el análisis de un gran volumen de datos en aplicaciones inmersivas. Para poder analizar el gran volumen de datos los autores, como se explica en los párrafos anteriores, utilizan Python y, para visualizar el entorno inmersivo, Unity. Para conectar estas tecnologías hacen uso del framework de Google Grpc.

2.2.2 Improved Knowledge from Data: Building an Immersive Data Analysis Platform

El artículo presentado por los investigadores de la Universidad de Campinas [23] pretende analizar grandes volúmenes de datos y extraer conocimiento de estos, específicamente en entornos de RV haciendo uso de herramientas y técnicas de Visual Analytics (VA) con el fin de mejorar la experiencia de usuario en entornos inmersivos y la forma de representar todo el conocimiento extraído de este gran volumen de datos.

La solución propuesta por los autores consiste en una plataforma inmersiva que utilice Visual Analytics y que sea capaz de implementar distintos procesos los cuales se explican a continuación. Cuando el usuario se encuentra inmerso en el sistema de RV mediante un HMD debe, en primer lugar, seleccionar el origen de la fuente de datos que admitirá archivos CSV, JSON y documentos de Microsoft Excel. Una vez seleccionada la fuente de datos se procede a cargar un conjunto de datos concretos de la fuente elegida. En este punto, el usuario puede visualizar los datos directamente o realizar un análisis automático. Si el usuario elige la segunda opción, la plataforma permitirá elegir distintas perspectivas y ejecutar distintas operaciones estadísticas, modelar los datos, reconocer patrones, etc. Al visualizar los datos, el usuario podrá manipularlos, hacer zoom, obtener información específica, etc.

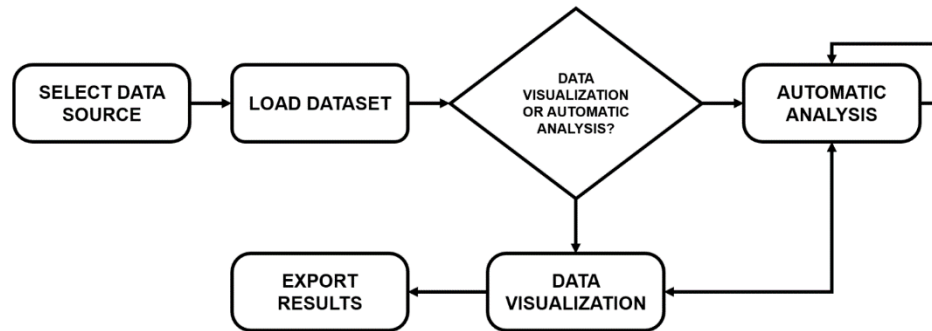


Ilustración 5 Interacción del usuario propuesta [23]

A pesar de que las tecnologías para llevar a cabo los procesos definidos en el párrafo anterior aún están siendo evaluadas, los autores planean utilizar Python para gestionar el análisis automático de los datos y Unity para visualizar el contenido inmersivo. Para establecer la comunicación entre Python y Unity3D se plantean utilizar llamadas a procedimiento remoto (RPC) utilizando el framework Grpc.

2.2.3 Welicit: A Wizard of Oz Tool for VR Elicitation Studies

Las herramientas del mago de Oz (WOz) [24] permiten a los experimentadores simular funcionalidades en sistemas que aún no están implementados, a través de una interfaz previamente definida, proporcionando al usuario la ilusión de que interactúan con un sistema totalmente funcional. Los estudios de elicitación para RV, consisten en una metodología basada en obtener información de un ser humano que se encuentra inmerso en un sistema de RV para potencialmente definir de forma experiencial los requisitos de un sistema. Concretamente, en estos estudios, el experimentador muestra al usuario el resultado de una acción y el usuario que participe en el experimento debe realizar la interacción que provoque ese resultado.

Teniendo en cuenta los conceptos definidos en el párrafo anterior, los autores presentan Welicit [25], una herramienta WOz para facilitar los estudios de elicitación en sistemas de Realidad Virtual, con la finalidad de investigar sobre el modo de interacción, comportamiento y preferencias de los usuarios en este tipo de sistemas. La arquitectura de esta herramienta está construida sobre A-Frame Inspector 1 [26], una herramienta de código abierto para construir escenas 3D mediante widgets que permiten modificar las propiedades de las entidades y ajustar componentes. El sistema cuenta con un servidor local, dónde el experimentador define las configuraciones iniciales del experimento como el conjunto de escenarios, el número de participantes, widgets para controlar los objetos 3D y la interfaz del asistente. Una vez realizadas las configuraciones oportunas en el servidor local, haciendo uso de Node.js [27], se genera automáticamente el código que da soporte a la interfaz del asistente, el cual estará disponible en un servidor remoto o local, además de proporcionar distintas URL's que permiten a los participantes acceder a las escenas virtuales. Una vez el experimentador decide ejecutar el experimento mediante la interfaz del asistente y los participantes se encuentran inmersos en el sistema de RV, el

servidor local recibe el audio y video de los participantes, así como información de las escenas virtuales. Una vez finalizado el experimento, Welicit proporciona una interfaz para observar el audio y video de los usuarios que se encontraban en las escenas virtuales, para analizar la forma en la que estos interactúan con el sistema.

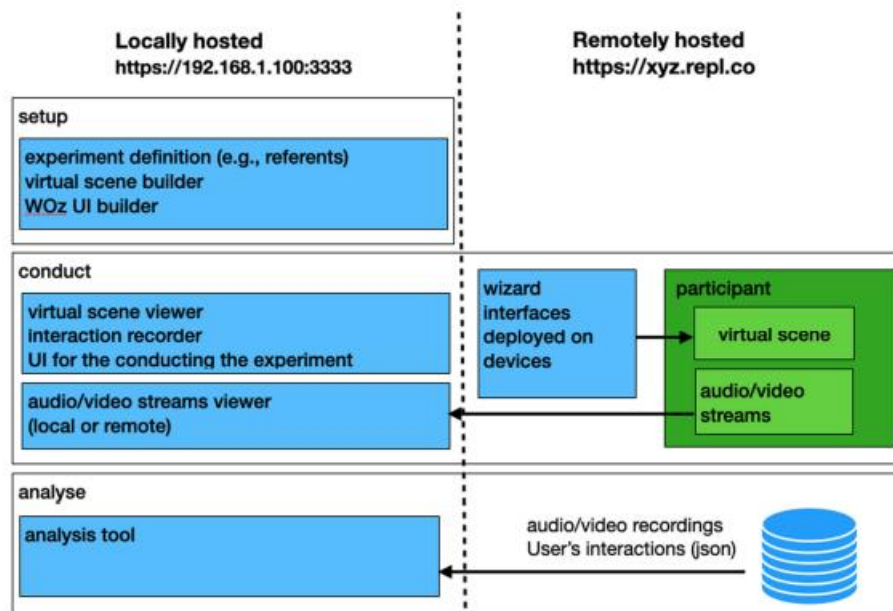


Ilustración 6 Arquitectura Welicit [25]

El sistema Welicit fue desarrollado teniendo en cuenta varios requisitos de diseño como reducir los conocimientos técnicos y la carga cognitiva de los experimentadores, facilitar el análisis de los resultados, realizar estudios a distancia, permitir interacciones que abarquen distintas modalidades y generar datos de interacción en diferentes contextos.

En el artículo propuesto, los investigadores exponen distintos resultados una vez decidieron poner la herramienta a prueba. La fase de configuración inicial facilita la implementación tanto del diseño final del entorno virtual como la interfaz del asistente. Los participantes, una vez se encontraban en el entorno de RV utilizando las gafas Oculus Rift se animaron a proponer distintos modos de interacción para los objetos propuestos. Cuando finalizó el experimento, los investigadores descubrieron que sería útil para el experimentador implementar una interfaz adicional que permita añadir anotaciones en tiempo real.

Como conclusión, los investigadores proponen una herramienta de código abierto para analizar la forma de interacción de usuarios en sistemas de realidad virtual gracias a la herramienta WOz. Para lograr este objetivo, la herramienta cuenta con un servidor local para configurar los experimentos y un servidor remoto con una interfaz para que el experimentador pueda controlar los experimentos y los participantes puedan acceder al entorno inmersivo. El fin de este artículo es poner la herramienta a disposición de otros investigadores para poder contribuir al análisis de los estudios de elicitación en los sistemas de RV.

2.2.4 Toggle toolkit: A tool for conducting experiments in unity virtual environments

El artículo de investigación presenta Toggle Toolkit [28], un conjunto de scripts originales programados en C# diseñados para ser utilizados en el motor de videojuegos Unity. La función principal de este kit de herramientas es permitir a los investigadores realizar experimentos en tiempo real de visualización 3D en entornos inmersivos, concretamente, añadir y modificar funcionalidades, modificar gráficos y visualizar elementos interactivos y dinámicos. Para poder realizar estas operaciones en las escenas y objetos 3D Toggle Toolkit precisa de utilizar triggers y toggles.

Los triggers y toggles son básicamente scripts que se comunican entre sí y permiten cambiar el estado y las propiedades de los objetos que forman parte de las escenas de RV. Cuando se activan los disparadores es posible producir un cambio instantáneo o continuo en el estado de los objetos. La relación entre activadores (triggers) y toggles es una relación N:N, lo que quiere decir que un único activador puede alterar varios toggles y, un único toggle puede ejecutar varios activadores.

Algunas de las funcionalidades que los investigadores han implementado en Toggle Toolkit son el comportamiento autónomo de los scripts, el teletransporte de usuarios, la capacidad de manipular la luz (encender, apagar o variar la intensidad) en los objetos que contengan un componente luminoso, ocultar objetos, resaltar textos, reproducir audios en objetos que posean un componente AudioSource e incluso ejecutar programas externos. Una de las características más notables de Toggle Toolkit es el módulo de registro de datos, el cual está implementado en el script PathScript, quien permite generar nuevos registros en base a la configuración del programa cuando Unity se ejecuta. Esto permite registrar algunos eventos que realiza el usuario inmerso en las escenas 3D una vez la aplicación se ha lanzado. No obstante, los investigadores no pueden determinar con detalle todas las interacciones del usuario con todos los objetos ni cómo esto afecta al espacio virtual, por lo que proponen utilizar una aplicación externa o algún script que incorpore este tipo de funcionalidad.

Después de realizar dos estudios experimentales los autores aconsejan a los futuros investigadores modelar el flujo de los distintos scripts y los respectivos objetos de Toogle Toolkit utilizando autómatas o una herramienta para el modelado de procesos con el fin de documentar la solución del sistema antes de ser implementado y poder modificarla si es necesario después. Por último, advierten que si se añaden implementaciones muy complejas interconectadas Toggle Toolkit no cuenta con la capacidad para implementarlas debido a sus limitaciones de diseño del software.

2.3 Tecnologías que permiten el desarrollo de una aplicación web.

2.3.1 JEE

JEE (Java Enterprise Edition) [29] es una tecnología creada por Sun propuesta para el desarrollo de aplicaciones distribuidas usada actualmente por numerosas organizaciones en el entorno empresarial. JEE puede definirse como una normativa para

aplicaciones distribuidas encargada de describir los distintos elementos que intervienen y contribuyen en el funcionamiento de la aplicación. El lenguaje de programación que sustenta la base para esta solución es Java, también conocido como JSE (Java Standard Edition). Algunas de las ventajas de utilizar este lenguaje es que puede ser ejecutado en cualquier arquitectura debido a que hace uso de una máquina virtual, permite organizar los objetos en clases con sus respectivos métodos, es sencillo de aprender, posee potentes plataformas para el desarrollo de proyectos como Eclipse [30] o Netbeans [31], cuenta con una amplia colección de librerías de código abierto, etc.

Algunos de los elementos definidos por JEE son cómo desarrollar los distintos componentes que forman parte de la aplicación, por ejemplo, mediante páginas JSP o servlets. También define cómo los distintos componentes que forman parte de la aplicación distribuida se comunican tanto con otras aplicaciones como entre ellos utilizando, por ejemplo, jndi, JavaMail o JDBC. JEE también se encarga de definir cómo todos los componentes de la aplicación deben organizarse mediante un descriptor de despliegue y especifica las limitaciones de los servidores que dan soporte a las distintas aplicaciones distribuidas. El aplicar esta normativa en un proyecto permite que varias sociedades se dediquen a desarrollar servidores capaces de hospedar las aplicaciones de cualquier otro proyecto que cumpla los estándares instaurados por JEE.

La arquitectura de JEE está compuesta por las siguientes capas [32]:

- Capa cliente: está formada por la interfaz a la que accede el usuario final. Esta capa contiene la lógica necesaria para procesar los componentes de negocio mediante un navegador cuando se utiliza un cliente web y, en el caso de que se precise de una aplicación cliente contendrá la lógica para comunicarse con la capa web mediante servlets gracias a HTTP [33].
- Capa web: también conocida como la capa de presentación esta contiene la lógica encargada de recibir solicitudes por parte del usuario (capa cliente) y generar respuestas que serán procesadas por capa de negocio. Esta capa puede incluir servlets o páginas JSP.
- Capa de negocio: esta capa contiene la lógica de negocio y es la encargada de coordinar varios usuarios, gestionar la BBDD, procesar datos, etc. Para implementar esta capa que se comunica con la capa de negocio se suele utilizar la API de EJB (Enterprise JavaBeans).
- Capa de conectividad y recursos: esta capa está formada por el software e infraestructura para gestionar los datos persistentes de la aplicación distribuida, es decir, la base de datos.

Las aplicaciones JEE precisan de un marco de desarrollo para poder ser ejecutadas. El framework actualmente más utilizado es Spring [34] debido a que permite crear aplicaciones basadas en la arquitectura MVC [35], proporciona consistencia a la hora de gestionar las transacciones, está especializado en servicios RESTful [36], utiliza POJO [37], etc.

2.3.2 JS

JavaScript es un lenguaje de programación de un solo hilo, dinámico, multiparadigma, basado en prototipos, que admite programación orientada a objetos y cuyo estándar es ECMAScript (ECMA-402) [38]. Este lenguaje precisa de HTML y CSS para mostrar el contenido web y se ejecuta en el lado del cliente con la capacidad de gestionar los eventos del navegador. Su sintaxis es similar a C++ y Java con la ventaja de que permite implementar funciones complejas de una manera más rápida y con menos líneas de código.

2.3.3 Express

Express [39] es un framework utilizado para desarrollar proyectos de servidores web en el entorno de ejecución Node.js. Está formado por una capa de enrutamiento y se despliega sobre el servidor HTTP de node.js. Este framework ofrece un patrón básico de software intermedio y un enrutamiento declarativo.

La estructura de express está compuesta por tres capas que se comunican entre sí. En primer lugar, tenemos la capa superior `expressjs`, que cuenta con los `middlewares` encargados de gestionar tanto las peticiones como la pila. La segunda capa hacia abajo la forma `pillarjs` la cual cuenta con el enrutador Express, el módulo de envío que contiene la lógica para leer archivos del sistema para posteriormente agruparlos y enviarlos y, por último, el controlador final donde se definen las diferentes funciones y es el encargado de serializar los errores. La última capa es `jshttp` la cual tiene tres módulos para gestionar el módulo de envío de la capa superior `pillarjs` que son el módulo `etag` quien envía información de solicitudes y respuestas mediante etiquetas, el módulo `fresh` quien decide cómo enviar esas etiquetas y, el módulo `range-passer` quien necesita analizar los rangos. La capa `jshttp` también cuenta con un módulo final que ayuda a gestionar las solicitudes del controlador final de la capa `pillarjs`.

Para instalar express en nuestra aplicación node.js simplemente se ha de ejecutar ‘`npm install express`’ y declarar una variable en el código Javascript haciendo uso del módulo ‘`require`’. Una vez realizado los pasos anteriores es posible manejar las rutas de la aplicación creada y los distintos métodos de solicitud como ‘`GET`’, ‘`PUT`’, ‘`POST`’, ‘`USE`’, etc.

Express se caracteriza por su gran rendimiento en aplicaciones web ya que permite integrar motores de renderización de “vistas”, como EJS [40], para mostrar las respuestas que son tratadas como objetos. Ayuda a manejar de manera sencilla las rutas HTTP, cuenta con muchos paquetes de `middleware` compatibles, permite su fácil integración en proyectos Javascript, etc.

2.3.4 Django

Django [41] es un framework de alto nivel para aplicaciones web que serán desarrolladas en el lenguaje de programación Python. Características como que es gratuito, de código abierto, que evita problemas de seguridad, que es escalable y versátil, hace que sea uno de los frameworks más populares para realizar aplicaciones web.

Esta herramienta permite definir una base de datos sin utilizar ningún gestor de este tipo. Para ello, es necesario diseñar un modelo programado en Python con las entidades, instalar la API gratuita y con escasas líneas de código ya se podrán realizar de forma minimalista todas las operaciones necesarias para administrar las entidades del proyecto. También permite definir las rutas URL de la aplicación de forma limpia y sencilla, codificar plantillas en html con la ventaja de poder utilizar Python con etiquetas, etc.

2.3.5 PHP

PHP (Hypertext Preprocessor) [42] es un lenguaje de programación de código abierto cuyas características hacen que sea uno de los lenguajes más utilizados actualmente para el desarrollo web. Este puede ser embebido en un html utilizando unas etiquetas especiales permitiendo cargar los elementos de una página web antes de mostrarlos al usuario, con la ventaja de que el cliente no podrá acceder al código subyacente generado por php.

Este lenguaje de programación de alto nivel fue desarrollado en C en 1994 por Rasmus Lerdorf [43]. Su primera versión permitía interpretar un número limitado de comandos, aunque adquirió éxito y produjo que nuevos desarrolladores y programadores investigaran y contribuyeran para mejorarlo. La versión PHP 4 implementó características para trabajar con objetos, aunque eran muy rudimentarias hasta PHP 5, el cual estuvo 11 años hasta que quedó obsoleto debido a que las variables no eran tipificadas, hasta que en 2015 salió al mercado PHP 7 subsanando los errores de las versiones anteriores y con mejoras de rendimiento.

Algunos de los motivos de su actual popularidad es su simplicidad para implementar funciones complejas, su naturaleza para disminuir el tiempo de carga de las páginas o su facilidad para conectarse con bases de datos [44].

2.4 Bases de datos

En esta sección, se explican en diferentes apartados los modelos de bases de datos relacionales y no relacionales. Además, se dedica un subapartado para cada uno de los modelos descritos que contiene un ejemplo concreto de un sistema gestor.

2.4.1 Bases de datos relacionales

Las bases de datos relaciones son un tipo de base de datos que satisfacen el modelo relacional, el cual consiste en agrupar la información en bloques y establecer vínculos o nexos con otros bloques cuando sea necesario relacionar información entre estos [38]. Concretamente, este modelo utiliza tablas que representan entidades y permiten de una forma intuitiva y eficiente almacenar y acceder a la información estructurada. Para poder comunicarse con la base de datos se precisa de SQL (Structured Query Language) [45], un lenguaje de consultas basado en álgebra relacional que permite tanto la manipulación como el acceso a los datos concretos de las tablas que forman el sistema.

Este modelo garantiza la integridad de los datos gracias al uso de restricciones que ayudan a aplicar reglas en los datos de las tablas para lograr fiabilidad y precisión. Las restricciones pueden establecer un dominio de valores para un campo, evitar falta de datos, identificar tablas y sus relaciones con otras, etc. Todas las transacciones, es decir, una instrucción o un conjunto de instrucciones en SQL deben satisfacer ACID para asegurar la integridad de los datos [47]:

- Atomicidad: la instrucción o conjunto de instrucciones en SQL debe ejecutarse por completo o ser invalidada.
- Consistencia: es imprescindible que los datos de la transacción cumplan con todas las reglas y el formato definido en la base de datos.
- Aislamiento: consiste en realizar las transacciones de forma independiente
- Durabilidad: requiere que las transacciones sean persistentes.

2.4.1.1 MySQL

MySQL [48] es un sistema de gestión de bases de datos relacional o RDBMS (Relational Database Management System) de código abierto que emplea un modelo cliente-servidor. Desarrollado por Oracle Corporation en 1995 es uno de los gestores de bases de datos más populares que cumplen con el estándar SQL.

Los gestores MySQL organizan la información en forma de tablas, las cuales están formadas por atributos y contienen la información real dividida en filas y columnas. Cada tabla debe tener como mínimo una clave única (PRIMARY KEY) para poder identificar cada elemento, pero también puede tener claves foráneas para establecer relación con otras tablas.

Algunas de las ventajas de usar MySQL es su alto rendimiento y escalabilidad ya que permite acceder a la información de forma rápida y consumiendo pocos recursos. Es un servicio gratuito y fácil de usar con una amplia documentación e incorpora un sistema de administración de cuentas de usuario, privilegios de acceso, sistema de encriptación de contraseñas, etc. Por lo que todas estas características lo convierten en un sistema confiable.

2.4.2 Bases de datos no relacionales

Las bases de datos no relacionales o NoSQL [49] cuentan con la capacidad de gestionar tanto datos estructurados como semiestructurados debido a que no utilizan el modelo relacional. Tampoco hacen uso de SQL ya que para acceder y manipular los datos es necesario utilizar alguna API basada en objetos, que es como se representan las distintas entidades existentes en la base de datos. Las características más relevantes de las bases de datos NoSQL es su alto rendimiento y funcionalidad, el permitir esquemas flexibles que favorecen una mejor organización de la información y su alta escalabilidad gracias al uso de clústeres.

Existen distintos tipos de bases de datos no relacionales [50]:

- **Clave-valor:** facilitan el escalado horizontal proporcionando una baja latencia lo cual lo convierte en un modelo óptimo para ser utilizado en juegos, publicidad o el ámbito de IoT.
- **Documentos:** es utilizado en sistemas de administración, catálogos, aplicaciones web, etc. En vez de usar tablas para representar las entidades hace uso de objetos o documentos JSON que permiten a los desarrolladores consultar y almacenar la información de una manera muy cómoda sencilla y eficiente.
- **Gráficos:** es muy común en redes sociales, detección de fraude y motores de recomendación debido a que simplifica las aplicaciones que utilizan datos altamente conectados.
- **En memoria:** es una buena opción para análisis en tiempo real, juegos o aplicaciones que utilicen sesiones debido a que ofrece un tiempo de respuesta muy rápido.

Cualquier tipo de base de datos NoSQL no satisface las propiedades ACID, por lo que es una buena opción si se desea un sistema con un modelo de datos flexible, de alto rendimiento, con baja latencia y escalable horizontalmente.

2.4.2.1 MongoDB

MongoDB [51] es un sistema gestor de bases de datos no relacional de código abierto. Se caracteriza porque almacena los datos en documentos, los cuales tienen un formato BSON, es decir, una representación binaria de JSON. Los documentos se almacenan en colecciones, lo que sería una tabla en MySQL, pero con la ventaja de que los documentos pueden tener distintos atributos e incluso ser de distinto tipo.

```
{
  _id: ObjectId("5099803df3f4948bd2f98391"),
  name: { first: "Alan", last: "Turing" },
  birth: new Date('Jun 23, 1912'),
  death: new Date('Jun 07, 1954'),
  contribs: [ "Turing machine", "Turing test", "Turingery" ],
  views : NumberLong(1250000)
}
```

Ilustración 7 Ejemplo mongodb documento persona [51]

Características como la adaptabilidad de los documentos para convertirse en objetos, la facilidad para realizar consultas e inserciones, la indexación, que es de uso gratuito, etc. provocan que sea uno de los sistemas más utilizados en los proyectos que precisan de una base de datos.

2.5 Conclusiones del estado del arte.

Una vez analizadas las tecnologías que permiten la ejecución de funciones remotas en aplicaciones web y las herramientas o marcos de trabajo que tienen como fin examinar el comportamiento y/o la forma de interactuar de los usuarios que se encuentran inmersos en un entorno de realidad virtual es posible llegar a varias conclusiones:

- Para visualizar y desarrollar el entorno inmersivo una buena opción es hacer uso del motor de videojuegos multiplataforma Unity, debido a que permite mostrar y programar el contenido 3D de una forma sencilla y minimalista, incorpora una API para interactuar con varios dispositivos de RV, no es necesario implementar plugins externos para cada dispositivo, etc.
- Utilizar un número elevado de scripts interconectados entre sí no es una buena práctica porque puede resultar complicado y engorroso para el investigador capturar las distintas interacciones de un usuario en un entorno inmersivo.
- Al trabajar con un gran volumen de datos es conveniente utilizar Python ya que es un lenguaje que incorpora potentes y veloces librerías para el análisis y visualización de datos, estadísticas o aprendizaje automático.
- Es conveniente utilizar Grpc en sistemas que requieran algún tipo de comunicación entre aplicaciones multilenguaje debido a la facilidad para enviar y recibir RPC's.

La herramienta por desarrollar se trata de una aplicación web del lado cliente con la capacidad de enviar RPC's a un servidor y recibir una respuesta asociada por parte de este. Además, la herramienta debe de simplificar la creación y ejecución de servicios que son los que definen las llamadas a procedimiento remoto. Después de estudiar en el apartado 2.3 todas las tecnologías actuales para el desarrollo de aplicaciones web se plantea la opción de utilizar Express en Node.js dado que se requiere de una interfaz gráfica potente y fácil de usar y estas tecnologías lo permiten satisfactoriamente.

Respecto a la decisión de usar una base de datos relacional o no relacional se plantea el uso de esta última, porque, aunque las bases de datos relacionales son más robustas y aseguran la integridad de los datos, para desarrollar la aplicación que se define en el presente documento se requiere de una mayor flexibilidad ya que la información y estructura de los datos es variable debido al uso de funciones remotas.

3. ANÁLISIS DEL PROBLEMA

En esta sección se describen las capacidades principales de la aplicación desarrollada, se detallan tanto los requisitos de usuario como los del sistema, se exponen distintos casos de uso y, por último, se detallan las tecnologías utilizadas para la implementación del sistema.

3.1 Alcance del proyecto

La herramienta por desarrollar podrá:

- Generar y eliminar tipos de mensaje necesarios para la creación de servicios.
- Generar y eliminar servicios.
- Establecer una comunicación con el servidor para enviar comandos remotos definidos previamente en forma de servicios.

La herramienta, en primera instancia, está desarrollada para investigadores interesados en la comunicación entre usuarios que estén utilizando un dispositivo inalámbrico de realidad virtual y un servidor encargado de crear el entorno inmersivo.

Su uso puede expandirse a sujetos que quieran controlar/monitorear a varios usuarios inmersos en un sistema de RV mediante un dispositivo inalámbrico. El sujeto o administrador podría gestionar de forma remota todos los eventos de cada usuario inmerso.

3.1.1 Casos de uso

En este apartado se analizan las posibles interacciones que puede realizar el usuario al utilizar la herramienta desarrollada. La siguiente ilustración representa un diagrama de casos de uso que especifica las acciones que puede realizar el usuario (Actor) en la herramienta web detallando el orden lógico de estas. Para una mejor comprensión se utilizan las relaciones ‘Include’ y ‘Extend’. Por un lado, cuando se emplea ‘Include’ quiere decir que el caso de uso al que apunta la cabeza de la flecha es necesario para poder llevar a cabo el caso de uso principal, por lo que podrían ser tratados como subprocesos. Por otro lado, ‘Extend’ implica la opcionalidad del caso de uso y la cabeza de la flecha apunta al caso de uso del que depende para poder ser llevado a cabo.

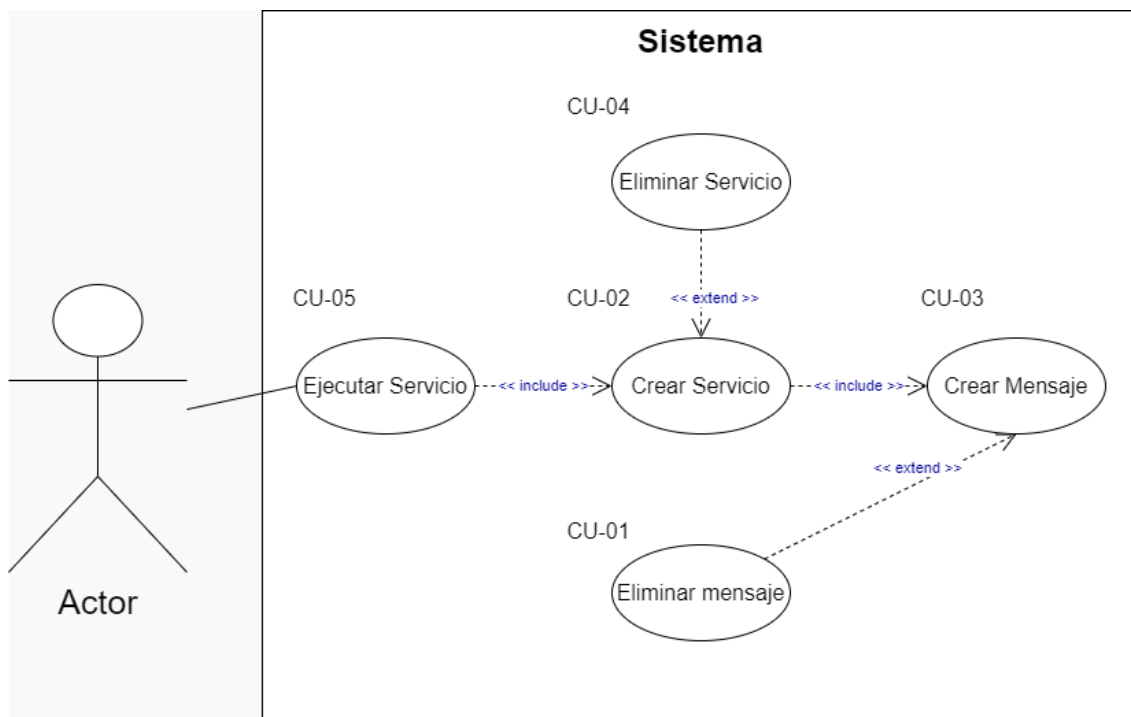


Ilustración 8 Casos de uso genéricos de la herramienta

El objetivo de la *Ilustración 8* es quedar claro que la finalidad de la aplicación es ejecutar servicios remotos y las acciones para lograrlo. Para poder llevar a cabo esta tarea es necesario que el usuario tenga claro algunos conceptos. Los servicios pueden estar formados desde uno hasta N RPC's, siendo N un número entero positivo mayor que uno. Cada RPC tiene que estar formado por dos mensajes. Por un lado, un mensaje de tipo "Request" que consiste en un objeto JSON que será enviado al servidor que de soporte al entorno inmersivo. Por otro lado, un mensaje de tipo "Response" que también será un objeto JSON, pero en este caso será la respuesta del servidor que será recibida por el cliente.

El usuario podrá ejecutar los servicios por defecto que incorpora la herramienta y crear nuevos servicios utilizando los mensajes tanto de tipo 'Request' como de tipo 'Response' ya existentes. No obstante, si la base de datos no ha sido poblada es importante tener en cuenta que no será posible crear un servicio debido a que la aplicación solo deja seleccionar los mensajes existentes, por lo que para llevar a cabo esta tarea la herramienta debe disponer de al menos un mensaje.

Para representar los casos de uso se usará la *Tabla 1* que describe el formato a utilizar:

ID	
Título	
Precondición	
Postcondición	

Tabla 1 Formato tabla casos de uso

- **ID (Identificador):** Cada caso de uso posee un identificador único con el siguiente formato “CU-XX” donde XX es un número único.
- **Título:** breve descripción del caso de uso específico.
- **Precondición:** condiciones necesarias para poder llevar a cabo la realización del caso de uso.
- **Postcondiciones:** resultados que surgen a través de la realización del caso de uso.

ID	CU-01
Título	El usuario crea un mensaje
Precondición	El usuario ha accedido a la url '/insertar_servicio' y ha rellenado todos los campos necesarios para la creación del mensaje.
Postcondición	El servidor recibe la petición, guarda el mensaje y lo inserta en la base de datos si no existe un mensaje con el mismo nombre.

Tabla 2 CU-01

ID	CU-02
Título	El usuario crea un servicio
Precondición	El usuario ha accedido a la url '/insertar_servicio' y ha rellenado todos los campos necesarios para la creación del servicio. Por cada grpc que forme parte del servicio el sistema solo dejará seleccionar aquellos mensajes que ya hayan sido creados anteriormente.
Postcondición	El servidor recibe la petición, guarda el servicio y lo inserta en la base de datos si no existe un servicio con el mismo nombre.

Tabla 3 CU-02

ID	CU-03
Título	El usuario elimina un mensaje
Precondición	El usuario ha accedido a la url '/insertar_servicio' o '/mostrar_mensajes' y ha hecho click sobre el icono con forma de basura.
Postcondición	El servidor recibe la petición y elimina el mensaje de la base de datos.

Tabla 4 CU-03

ID	CU-04
Título	El usuario elimina un servicio
Precondición	El usuario ha accedido a la url '/mostrar_servicios' y ha hecho click sobre el icono con forma de basura.
Postcondición	El servidor recibe la petición y elimina el servicio de la base de datos.

Tabla 5 CU-04

ID	CU-05
Título	El usuario ejecuta un servicio remoto
Precondición	El usuario ha accedido a la url '/ejecutar_servicio' y ha seleccionado un servicio, un Grpc asociado a ese servicio, ha rellenado todos los campos del/los atributo/s asociados al mensaje request del Grpc y ha hecho click en el botón de enviar.
Postcondición	El servidor recibe la petición y envía el RPC asociado al servicio elegido y lo envía al servidor que da soporte al entorno inmersivo.

Tabla 6 CU-06

3.1.2 Requisitos

En este apartado se definen los requisitos de la herramienta desarrollada. Debemos diferenciar entre requisitos de usuario y requisitos de sistema. Por un lado, los requisitos de usuario describen qué puede hacer el sujeto que utilice la aplicación. Por otro lado, los requisitos de sistema formalizan el comportamiento de la aplicación desarrollada, es decir, como se comporta el sistema ante la interacción del usuario. Para mostrar cada uno de ellos se usará una tabla con el siguiente formato:

ID	
Nombre	
Descripción	
Prioridad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
RU Involucrado(s)*	

Tabla 7 Formato tabla requisitos

- **ID (Identificador):** Cada requisito de usuario posee un identificador único. En función del tipo de requisito los identificadores tendrán un formato específico:
 - Requisitos de usuario: El identificador tendrá el formato “RU-XX” donde “XX” es el número único del requisito.
 - Requisitos de sistema: El identificador tendrá el formato “RS-XX” donde “XX” es el número único del requisito.
- **Nombre:** definición textual del requisito que permita conocer de que trata.
- **Descripción:** explicación del requisito.
- **Prioridad:** nivel de preferencia del requisito durante el desarrollo de la herramienta. Puede tomar el valor “alta”, “media”, o “baja”.
- **Estabilidad:** mide la capacidad de que el requisito pueda ser alterado en un futuro. Puedo tomar el valor de “alta”, “media”, o “baja”.
- **RU Involucrado*:** cuando se trate de un requisito de sistema, se especificará el requisito o requisitos asociados.

3.1.2.1 Requisitos de usuario

En este subapartado se listan y detallan los requisitos de usuario.

ID	RU-01
Nombre	Navegación
Descripción	El usuario podrá acceder a las distintas funcionalidades de la aplicación mediante una interfaz gráfica.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja

Tabla 8 RU-01

ID	RU-02
Nombre	Ejecutar Comandos Remotos
Descripción	El usuario podrá ejecutar los distintos comandos definidos en los servicios desde la aplicación cliente obteniendo feedback por parte del servidor.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja

Tabla 9 RU-02

ID	RU-03
Nombre	Crear Servicio
Descripción	El usuario podrá generar nuevos servicios a través de una interfaz gráfica.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja

Tabla 10 RU-03

ID	RU-04
Nombre	Crear Mensaje
Descripción	El usuario podrá generar nuevos mensajes a través de una interfaz gráfica.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja

Tabla 11 RU-04

ID	RU-05
Nombre	Consultar Servicios
Descripción	El usuario podrá consultar todos los servicios previamente creados en la aplicación.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja

Tabla 12 RU-05

ID	RU-06
Nombre	Consultar Mensajes
Descripción	El usuario podrá consultar todos los mensajes previamente creados en la aplicación.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja

Tabla 13 RU-06

ID	RU-07
Nombre	Eliminar Servicio
Descripción	El usuario podrá eliminar cualquier servicio existente a través de una interfaz gráfica.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja

Tabla 14 RU-07

ID	RU-08
Nombre	Eliminar Mensaje
Descripción	El usuario podrá eliminar cualquier mensaje existente a través de la interfaz gráfica.
Prioridad	<input type="checkbox"/> Alta <input checked="" type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja

Tabla 15 RU-08

3.1.2.2 Requisitos de sistema

Este subapartado se destina a listar y detallar los requisitos del sistema.

ID	RS-01
Nombre	Establecer comunicación con entorno inmersivo.
Descripción	El sistema debe comunicarse con un servidor que de soporte a un entorno inmersivo para el envío de llamadas a procedimiento remoto.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-02

Tabla 16 RS-01

ID	RS-02
Nombre	Persistencia de servicios descritos por el usuario.
Descripción	El sistema debe permitir almacenar de forma permanente los servicios creados por el usuario por medio de una base de datos NoSQL.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-03, RU-04, RU-07, RU-08.

Tabla 17 RS-02

ID	RS-03
Nombre	Automatizar creación de archivo de definición de servicios.
Descripción	El sistema debe generar un fichero que contenga los elementos necesarios para permitir ejecutar comandos remotos en la aplicación cliente que tengan impacto en el servidor asociado.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-03, RU-04, RU-07, RU-08.

Tabla 18 RS-03

ID	RS-04
Nombre	Actualizar Servicios
Descripción	El sistema debe actualizar los servicios en la base de datos cada vez que el usuario inserte o elimine alguno de estos.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-03, RU-07.

Tabla 19 RS-04

ID	RS-05
Nombre	Actualizar Mensajes
Descripción	El sistema debe actualizar los mensajes en la base de datos cada vez que el usuario inserte o elimine alguno de estos.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-04, RU-08.

Tabla 20 RS-05

ID	RS-06
Nombre	Mostrar Servicios
Descripción	El sistema debe mostrar todos los servicios que hay actualmente en la base de datos.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-05

Tabla 21 RS-06

ID	RS-07
Nombre	Mostrar mensajes
Descripción	El sistema debe mostrar todos los mensajes que hay actualmente en la base de datos.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-06

Tabla 22 RS-07

ID	RS-08
Nombre	Comprobar formato servicio
Descripción	El sistema debe comprobar que el usuario ha rellenado todos los campos del formulario para evitar crear servicios con algún campo del formulario vacío. Si falta algún campo, el sistema deberá avisar al usuario del campo/s que falta/n.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-03

Tabla 23 RS-08

ID	RS-09
Nombre	Comprobar formato mensaje
Descripción	El sistema debe comprobar que el usuario ha rellenado todos los campos del formulario para evitar crear mensaje con algún campo del formulario vacío. Si falta algún campo, el sistema deberá avisar al usuario del campo/s que falta/n.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-04

Tabla 24 RS-09

ID	RS-10
Nombre	Evitar servicios duplicados
Descripción	El sistema debe comprobar que el nombre del servicio introducido por el usuario es único. Si el nombre del servicio introducido no es único, el sistema debe indicar al usuario que el servicio introducido ya existe y evitar la inserción.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-03

Tabla 25 RS-10

ID	RS-11
Nombre	Evitar mensajes duplicados
Descripción	El sistema debe comprobar que el nombre del mensaje introducido por el usuario es único. Si el nombre del mensaje introducido no es único, el sistema debe indicar al usuario que el mensaje introducido ya existe y evitar la inserción.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-04

Tabla 26 RS-11

ID	RS-12
Nombre	Confirmar inserción
Descripción	El sistema debe dar feedback al usuario cuando un servicio o mensaje se inserte correctamente en la base de datos
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-03, RU-04.

Tabla 27 RS-12

ID	RS-13
Nombre	Confirmar borrado
Descripción	El sistema debe dar feedback al usuario cuando un servicio o mensaje se elimine correctamente en la base de datos
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-07, RU-08.

Tabla 28 RS-13

ID	RS-14
Nombre	Manejo de errores
Descripción	El sistema debe responder con el código 500 de HTTP cuando se produzca cualquier error.
Prioridad	<input checked="" type="checkbox"/> Alta <input type="checkbox"/> Media <input type="checkbox"/> Baja
Estabilidad	<input type="checkbox"/> Alta <input type="checkbox"/> Media <input checked="" type="checkbox"/> Baja
RU Involucrado(s)	RU-01, RU-02, RU-03, RU-04, RU-05, RU-06, RU-07, RU-08.

Tabla 29 RS-14

3.2 Implementación

En este apartado se detallan las tecnologías y herramientas que se han utilizado para implementar el proyecto y la forma en que lo hacen.

Para establecer la comunicación entre la aplicación cliente y servidor y poder enviar y recibir RPC's se ha utilizado el framework de Google GRPC, el cual estará escuchando en el puerto 50051 de la aplicación servidor para recibir las llamadas que serán enviadas utilizando el mismo puerto desde la aplicación cliente. La aplicación servidor, la cual dará soporte al entorno inmersivo idealmente debe ser desarrollada en el entorno Unity, aunque los investigadores pueden utilizar cualquier otro entorno disponible con los lenguajes que admite GRPC [52]. La aplicación cliente ha sido desarrollada en el entorno Node.js utilizando el framework Express. Esta inicia un servidor HTTP desplegado de forma local (localhost) utilizando el puerto 3000. Para llevar a cabo la comunicación entre el cliente y el servidor con GRPC es necesario utilizar un fichero “.proto” que esté formado por los servicios y sus respectivos RPC's. Este fichero es generado dinámicamente en la aplicación cliente en base a los servicios y mensajes actuales en la base de datos y para escribir en él se hace uso del módulo de sistema de archivos fs para Node.js [27]. Para poder recibir los comandos remotos enviados por parte de la herramienta web al servidor de Unity actualmente es necesario copiar el fichero generado en la aplicación cliente y pegarlo en el repositorio correspondiente del servidor.

Respecto a la base de datos, se ha decidido utilizar el sistema de base de datos NoSQL MongoDB y, la herramienta elegida para gestionar los datos ha sido MongoDB Compass [53], que permite visualizar y gestionar de una manera gráfica las colecciones y sus respectivos documentos. La base de datos ha sido alojada localmente (localhost) en el puerto 27017. Es importante mencionar el uso de la biblioteca de programación Mongoose [54], la cual está desarrollada para ser usada en el entorno Node.js y es la que ha permitido realizar de forma muy sencilla y eficaz la conexión y todas las operaciones de la base de datos.

La arquitectura del sistema desarrollado sigue el modelo MVC. Este patrón de diseño del software consiste en dividir en tres la lógica de la aplicación: el modelo, quien se encargará de manejar los datos y la lógica de negocios; la vista, que se ocupa del diseño y presentación y, por último; el controlador, quien se ocupa de gestionar las rutas de la aplicación y dirigir la vista y el modelo. En primer lugar, para el modelo, se han representado las entidades ‘Servicio’ y ‘Mensajes’ mediante los esquemas (Schema) codificados en JavaScript que proporciona la librería Mongoose. En segundo lugar, para poder mostrar los elementos del modelo de forma rápida, simple y eficaz se ha hecho uso de la plantilla Javascript para renderizar vistas EJS, quien ha permitido tratar las entidades del modelo como objetos y, por lo tanto, generar de una manera más cómoda todo el HTML necesario. Por último, el controlador, quien se encarga de redirigir todas las peticiones HTTP, realizar todas las operaciones de la base de datos, enviar los comandos al servidor y todo lo relacionado con la lógica de la aplicación, el cual se encuentra codificado en el archivo ‘index.js’ del directorio raíz de la aplicación cliente.

La mayoría de los estilos que el cliente puede observar en el navegador han sido implementados utilizando la biblioteca multiplataforma Bootstrap [55], concretamente la versión 5. El resto de los estilos han sido implementados por el autor del presente documento.

A continuación, la *Ilustración 9* muestra el directorio raíz de la aplicación cliente con el fin de visualizar de una manera más gráfica la estructura del proyecto:

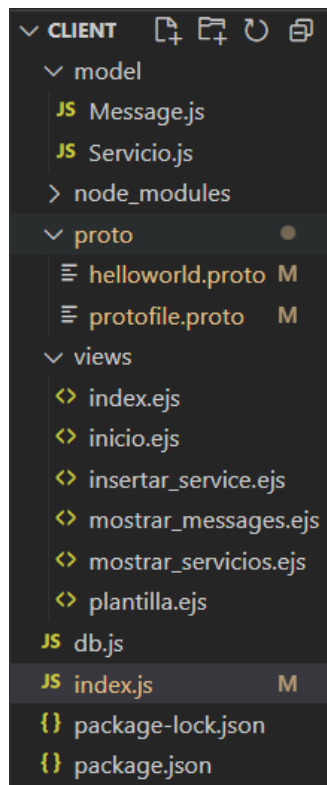


Ilustración 9 Directorio raíz herramienta desarrollada

- **Directorio model:** en esta carpeta del proyecto se definen los modelos de la base de datos MongoDB haciendo uso de la estructura Schema proporcionada por la API moongoose. Concretamente, se codificaban modelos para los mensajes y servicios con el fin de crear y leer documentos de la base de datos subyacente.
- **Node_modules:** contiene todos las librerías y paquetes necesarios instalados en el proyecto junto con sus respectivas dependencias.
- **Proto:** en este directorio se encuentra el archivo .proto quien está formado por el conjunto de servicios y mensajes necesarios para ejecutar comandos remotos en el servidor.
- **Views:** contiene todas las vistas de la aplicación desarrollada.
- **Db.js:** este fichero javascript es el encargado de establecer la comunicación con la base de datos en MongoDB.
- **Index.js:** en este fichero se encuentra el código para generar dinámicamente el archivo .proto, se manejan las rutas de la aplicación, se inicia el servidor HTTP y se establece la comunicación con el servidor de Unity mediante gRPC.
- **Package-lock.json:** este fichero es generado automáticamente en base a las operaciones npm y trata de un árbol con todas las dependencias del proyecto.
- **Package.json:** este fichero está formado por todos los metadatos del proyecto como las dependencias, descripción, licencias y scripts.

Con el fin de lograr una sencilla utilización de la herramienta por parte de futuros investigadores se ha realizado la siguiente guía que resume de forma detallada todas las capacidades del sistema y los pasos a realizar para el correcto funcionamiento de este.

El código de la herramienta web está disponible en el siguiente repositorio: <https://github.com/jesusalbarca/TFG-Herramienta-Control-y-Monitorizacion-Entornos-Inmersivos>.

Configuraciones Iniciales

Para evitar errores al iniciar la herramienta hay que tener en cuenta las siguientes consideraciones:

- En el equipo que se esté utilizando la herramienta debe existir una base de datos NoSQL en el host local con el puerto 27017. Esta base de datos debe contener una database con el nombre 'grpcDB' y dos colecciones llamadas 'messages' y 'services'.
- Los puertos 3000 y 50051 del equipo dónde se esté utilizando la herramienta no deben tener ningún servicio en ejecución.

Teniendo en cuenta las observaciones anteriores, para poder ejecutar la herramienta debemos acceder al directorio raíz que contiene la aplicación, abrir un terminal y lanzar el comando 'node index.js'. A partir de este momento, si se accede a la dirección web 'http://localhost:3000/' se podrá hacer uso de la herramienta.

Crea tu primer servicio

Es importante recordar en este punto que un servicio debe estar formado por dos mensajes. El primer mensaje será de tipo 'request' y será recibido por el servidor. El segundo mensaje será de tipo 'reply' y contendrá la respuesta del servidor.

En primer lugar, accedemos a la página crear servicio y observamos el apartado 'Crea tu nuevo tipo de mensaje!'. Como se puede apreciar, inicialmente hay dos campos, uno para el nombre del mensaje y otro para el nombre del atributo que es de tipo 'string' por defecto, aunque el usuario puede seleccionar cualquiera de los tipos admitidos por Grpc: 'string', 'int32' y 'boolean'. Entre los dos campos se aprecia un botón verde con un símbolo '+' que si es pulsado aparecerá un input debajo del primer atributo reservado para un atributo extra, debido a que el mensaje puede estar compuesto por tantos atributos como el usuario desee.

Imaginemos que un investigador necesita controlar las luces de varias salas que aparecen en un entorno de realidad virtual. Una buena práctica sería crear un servicio con un nombre identificativo que contenga varios RPC's. Uno, por ejemplo, para controlar la intensidad y duración de la luz en una determinada sala, otro para apagar o encender las luces de todas las salas disponibles en el entorno, etc. Existen tantas posibilidades como el experimentador desee. Veamos cómo crear desde cero un servicio basado en el primer RPC de ejemplo. El primer paso es crear los dos mensajes asociados al servicio.

La *Ilustración 10* muestra un ejemplo de un mensaje ‘Request’. El servidor recibirá un objeto JSON con los valores introducidos cuando se quiera ejecutar.

Crea tu nuevo tipo de message!

Name of message

Insert your attribute(s) +

int32 ▼	<input type="text" value="intensity"/>
int32 ▼	<input type="text" value="duration"/>
string ▼	<input type="text" value="room"/>

Submit

Ilustración 10 Ejemplo mensaje request

La *Ilustración 11* muestra un ejemplo de un mensaje ‘Reply’. El servidor enviará al usuario que esté utilizando la herramienta un ‘true’ en caso de que se ejecute correctamente el comando remoto o ‘false’ en caso contrario. Sin embargo, esta funcionalidad debe ser implementada en el servidor.

Crea tu nuevo tipo de message!

Name of message

Insert your attribute(s) +

bool ▼	<input type="text" value="itsOkey"/>
---	--------------------------------------

Submit

Ilustración 11 Ejemplo mensaje reply

Para confirmar que se han creado correctamente los mensajes, el usuario puede deslizar la página y comprobarlo rápidamente.

Request	<ul style="list-style-type: none">• Nombre Atrib1: intensity (int32)• Nombre Atrib2: duration (int32)• Nombre Atrib3: room (string)
Reply	<ul style="list-style-type: none">• Nombre Atrib1: itsOkey (bool)

Ilustración 12 Ejemplo mensajes actuales

Sin salir de la propia página el usuario ya puede crear su primer servicio. El primer paso es ir al formulario ‘Crea tu nuevo servicio!’. Aquí se observan dos campos para rellenar con texto, uno reservado para el nombre del servicio y otro para el nombre del primer RPC y dos campos para seleccionar los mensajes correspondientes al ‘Request type’ y al ‘Reply Type’ del primer grpc, los cuales permiten seleccionar cualquier mensaje existente en la base de datos. Al igual que para el formulario destinado a los mensajes, se aprecia un botón verde que si es pulsado creará debajo del primer RPC tres campos destinados al

nombre, ‘Request type’ y al ‘Reply Type’, respectivamente del segundo RPC, debido a que un Servicio puede estar formado por tantos RPC’s como quiera el experimentador.

La *Ilustración 13* muestra un ejemplo de un servicio con nombre de servicio ‘ServiceLights’, nombre de RPC ‘SetLightsByRoom’ y seleccionados los tipos ‘Request’ y ‘Reply’ creados en esta guía siguiendo el ejemplo descrito al inicio de este subapartado.



Crea tu nuevo servicio!

Nombre Servicio: ServiceLights

GRPc's: +

Nombre GRPc 1: SetLightsByRoom

Request type GRPc 1: Request

Reply type GRPc 1: Reply

Submit

Ilustración 13 Ejemplo crear servicio

Si accedemos a la página ‘mostrar servicios’ podremos ver que se ha insertado correctamente.



ServiceLights

- Grpc1: SetLightsByRoom
 - Mensaje Request: Request
 - Mensaje Reply: Reply

Ilustración 14 Ejemplo servicios actuales

La herramienta web no tiene límites en cuanto al número de servicios y mensajes que puede almacenar. El usuario podrá eliminar cualquiera de ellos en cualquier momento haciendo click en el icono con forma de basura que siempre está disponible cuando se muestra cualquier mensaje o servicio.

Configurar Servidor.

Una vez creados los servicios a probar en el servidor se debe reiniciar la herramienta. Cuando esta se inicia de nuevo, recupera los servicios actuales de la base de datos y los escribe en el archivo ya anteriormente detallado 'protofile.proto'. Entonces, el usuario debe copiar manualmente el contenido del fichero 'protofile.proto' y pegarlo en el fichero 'helloworld.proto', que es el utilizado para establecer la comunicación con el servidor, dónde se deberán realizar las siguientes acciones:

- Emplear como fichero '.proto' el fichero 'helloworld.proto' de la herramienta web.
- Añadir al objeto 'Server' de GRPC el servicio a implementar junto con sus respectivos RPC's.
- Implementar funciones de tipo callback [56] por cada RPC.

Cuando esté implementada la funcionalidad en el servidor en base al objeto recibido por el usuario se podrá ejecutar el servicio correctamente.

Ejecutar Servicio

Para ejecutar el servicio es necesario acceder a la página 'Ejecutar servicio' y seguir los siguientes pasos. El primer paso es seleccionar el servicio a utilizar. Una vez elegido, el sistema mostrará un desplegable con todos los RPC's asociados a ese servicio en concreto y el usuario deberá elegir uno. El segundo paso consiste en rellenar los campos de los atributos pertenecientes al mensaje 'request' según las preferencias del experimentador. El tercer y último paso es pulsar el botón 'submit'. La *Ilustración 15* muestra un ejemplo de comando remoto a enviar basado en el servicio creado en los ejemplos proporcionados anteriormente en esta guía.

Ejecuta tus Servicios!

Elige el servicio a utilizar

ServiceLights

Elige el grpc deseado

SetLightsByRoom

Configura los atributos a tu gusto (Request)

intensity	0.4
duration	20
room	main

Submit

Ilustración 15 Ejemplo ejecutar servicio

El servidor recibirá los valores personalizados del usuario sobre los atributos intensidad, duración y habitación introducidos pertenecientes al grpc 'SetLightsByRoom' del servicio 'ServiceLights' y realizará las acciones en base a lo implementado con anterioridad.

Por último, se muestran capturas de pantalla de la herramienta web desarrollada para cada una de las páginas accesibles para el usuario.

- **Página Home:** aquí se da una breve manual de como ejecutar por primera vez un comando remoto.



Ilustración 16 Página home

- **Página Ejecutar Servicio:** en esta página el usuario puede ejecutar de forma remota los servicios que hay actualmente escritos en el fichero .proto y recibir feedback por parte del servidor.

[Home](#) [Ejecutar Servicios](#) [Crear Servicio](#) [Servicios](#) [Mensajes](#)

Ejecuta tus Servicios!

Elige el servicio a utilizar

Command

Elige el grpc deseado

LightsOn

Configura los atributos a tu gusto (Request)

MessageType	Type: string
OpCode	Type: int32
Data	Type: string
Options	Type: string

Submit

Ilustración 17 Página ejecutar servicios

- **Página Crear Servicio:** en esta página el usuario puede crear servicios y mensajes.

[Home](#) [Ejecutar Servicios](#) [Crear Servicio](#) [Servicios](#) [Mensajes](#)

Crea tu nuevo servicio!

Nombre Servicio

GRPC's

+

Nombre GRPC 1

Request type GRPC 1

Request_message

▼

Reply type GRPC 1

Request_message

▼

Submit

Crea tu nuevo tipo de message!

Name of message

Insert your attribute(s)

+

string

▼

input name of attribute type

Submit

Ilustración 18 Página crear servicio

- **Página Mostrar Servicios:** en esta página el usuario puede consultar y eliminar los servicios actuales en la base de datos.

[Home](#)
[Ejecutar Servicios](#)
[Crear Servicio](#)
[Servicios](#)
[Mensajes](#)

Tus servicios!




Nombre Servicio	Grpc's Asociados	
Greeter	<ul style="list-style-type: none"> Grpc1: SayHello <ul style="list-style-type: none"> Mensaje Request: HelloRequest Mensaje Reply: HelloReply Grpc2: SayHelloAgain <ul style="list-style-type: none"> Mensaje Request: HelloRequest Mensaje Reply: HelloReply Grpc3: AddMessage <ul style="list-style-type: none"> Mensaje Request: Request_message Mensaje Reply: Response_message 	
Command	<ul style="list-style-type: none"> Grpc1: LightsOn <ul style="list-style-type: none"> Mensaje Request: Request Mensaje Reply: Reply Grpc2: LightsOff <ul style="list-style-type: none"> Mensaje Request: Request Mensaje Reply: Reply Grpc3: MoveHome <ul style="list-style-type: none"> Mensaje Request: Request Mensaje Reply: Reply 	
GestionarNodos	<ul style="list-style-type: none"> Grpc1: MostrarNodos <ul style="list-style-type: none"> Mensaje Request: request_node Mensaje Reply: reply_node 	

Ilustración 19 Página mostrar servicios

- **Página Mostrar Mensajes:** en esta página el usuario puede consultar y eliminar los mensajes actuales de la base de datos.

[Home](#)
[Ejecutar Servicios](#)
[Crear Servicio](#)
[Servicios](#)
[Mensajes](#)

Tus mensajes!









Nombre Mensaje	Atributos Asociados	
Request_message	<ul style="list-style-type: none"> Nombre Atrib1: MessageType (string) Nombre Atrib2: OpCode (string) Nombre Atrib3: Dataa (string) Nombre Atrib4: Options (string) 	
HelloRequest	<ul style="list-style-type: none"> Nombre Atrib1: name (string) 	
HelloReply	<ul style="list-style-type: none"> Nombre Atrib1: message (string) 	
request_node	<ul style="list-style-type: none"> Nombre Atrib1: numer_nodos (int32) 	
reply_node	<ul style="list-style-type: none"> Nombre Atrib1: isOkey (bool) 	
Response_message	<ul style="list-style-type: none"> Nombre Atrib1: message (string) 	
Request	<ul style="list-style-type: none"> Nombre Atrib1: MessageType (string) Nombre Atrib2: OpCode (int32) Nombre Atrib3: Data (string) Nombre Atrib4: Options (string) 	
reply	<ul style="list-style-type: none"> Nombre Atrib1: itsOkey (bool) 	

Ilustración 20 Página mostrar mensajes

4. EVALUACIÓN

En esta sección se exponen las pruebas realizadas en la herramienta web con el fin de verificar el correcto funcionamiento del sistema. Para organizar de una manera óptima el conjunto de pruebas realizadas se utilizará la siguiente tabla como formato:

ID	
Nombre	
Requisitos Relacionados	
Descripción	
Resultados	
Estado	

Tabla 30 Formato tabla pruebas

- **ID (Identificador):** Cada prueba posee un identificador único con el siguiente formato “P-XX” donde XX es un número único.
- **Nombre:** nombre descriptivo de la prueba en cuestión.
- **Requisitos relacionados:** requisitos del sistema soportados para la realización de la prueba.
- **Descripción:** explicación de los pasos realizados para realizar la prueba.
- **Resultado:** hace referencia a la respuesta del sistema ante la acción que soporta la prueba.
- **Resultado:** respuesta del sistema ante una determinada prueba/acción.

A continuación, se exponen las pruebas realizadas en el sistema.

ID	P-01
Nombre	Crear servicio
Requisitos Relacionados	RS-02, RS-04, RS-08, RS-10, RS-12, RS-14
Descripción	1. El usuario accede a la página ‘crear_servicio’ mediante el menú de navegación. 2. El usuario rellena los campos del formulario 'Crear tu nuevo Servicio', selecciona los mensajes asociados a los grpc's que desee y pulsa el botón 'submit'.
Resultados	Si el nombre del servicio insertado no está en la base de datos el sistema informa al usuario mediante un pop-up de que el servicio ha sido creado correctamente. En caso contrario se informará al usuario mediante un pop-up de que ya existe un servicio con ese mismo nombre.
Estado	Verificado

Tabla 31 P-01

ID	P-02
Nombre	Crear mensaje
Requisitos Relacionados	RS-02, RS-05, RS-09, RS-11, RS-12, RS-14
Descripción	1. El usuario accede a la página 'crear_servicio' mediante el menú de navegación 2. El usuario rellena los campos del formulario 'Crear tu tipo de Mensaje', selecciona el tipo deseado de atributo y pulsa el botón 'submit'.
Resultados	Si el nombre del mensaje insertado no está en la base de datos el sistema informa al usuario mediante un pop-up de que el mensaje ha sido creado correctamente. En caso contrario se informará al usuario mediante un pop-up de que ya existe un mensaje con ese mismo nombre.
Estado	Verificado

Tabla 32 P-02

ID	P-03
Nombre	Visualizar servicios
Requisitos Relacionados	RS-02, RS-06, RS-14
Descripción	1. El usuario accede a la página 'Servicios' mediante el menú de navegación.
Resultados	El sistema muestra todos los servicios actuales en la base de datos.
Estado	Verificado

Tabla 33 P-03

ID	P-04
Nombre	Visualizar mensajes
Requisitos Relacionados	RS-02, RS-07, RS-14
Descripción	1. El usuario accede a la página 'Crear Servicio' o 'Mensajes' mediante el menú de navegación.
Resultados	El sistema muestra todos los mensajes actuales en la base de datos.
Estado	Verificado

Tabla 34 P-04

ID	P-05
Nombre	Eliminar Servicio
Requisitos Relacionados	RS-02, RS-04, RS-13, RS-14
Descripción	1. El usuario accede a la página 'Servicios' mediante el menú de navegación. 2. El usuario pulsa el botón 'basura' asociado al servicio que quiere eliminar.
Resultados	El sistema muestra un pop-up confirmando que el servicio ha sido eliminado correctamente.
Estado	Verificado

Tabla 35 P-05

ID	P-06
Nombre	Eliminar Mensaje
Requisitos Relacionados	RS-02, RS-05, RS-13, RS-14
Descripción	1. El usuario accede a la página 'Mensajes' o 'Crear Servicio' mediante el menú de navegación. 2. El usuario pulsa el botón 'basura' asociado al mensaje que quiere eliminar.
Resultados	El sistema muestra un pop-up confirmando que el mensaje ha sido eliminado correctamente.
Estado	Verificado

Tabla 36 P-06

ID	P-07
Nombre	Ejecutar Servicio
Requisitos Relacionados	RS-01, RS-02, RS-03, RS-14
Descripción	1. El usuario accede a la página 'Ejecutar Servicio' mediante el menú de navegación. 2. El usuario selecciona el servicio y el grpc a utilizar, rellena todos los campos del formulario y pulsa el botón submit.
Resultados	Si el servidor tiene implementado el servicio el usuario recibirá un mensaje indicando los cambios producidos en el entorno inmersivo. Si se produce algún error el sistema mostrará el mensaje de error correspondiente.
Estado	Verificado

Tabla 37 P-07

A continuación, se muestra la matriz de trazabilidad entre los requisitos del sistema y las pruebas realizadas.

	P-01	P-02	P-03	P-04	P-05	P-06	P-07
RS-01							
RS-02							
RS-03							
RS-04							
RS-05							
RS-06							
RS-07							
RS-08							
RS-09							
RS-10							
RS-11							
RS-12							
RS-13							
RS-14							

Tabla 38 Matriz trazabilidad entre pruebas y requisitos

5. PLANIFICACIÓN DEL PROYECTO

5.1 Estudio de metodologías para el desarrollo de proyectos.

En este apartado se nombran y explican brevemente las metodologías actuales existentes más comunes para el desarrollo de proyectos, con el fin de analizar cada una de ellas para finalmente escoger una que pueda ser utilizada para el proyecto que describe el presente documento.

- **Scrum:** se trata de una metodología ágil en el que se aplican distintas buenas prácticas para poder abordar problemas complejos. Realmente, se trata de un marco de trabajo fácil de entender, liviano, pero difícil de llegar a dominar. Esta metodología está basada en la teoría de control de procesos empírica y cuenta con varios eventos para evitar numerosas reuniones y crear regularidad. El evento más importante de Scrum es el Sprint, que consiste en bloques de tiempo de duración aproximada de 1 mes donde se mejora un producto hasta un determinado objetivo. Cada Sprint se planifica, se revisa y se da una retrospectiva (todos estos procesos quedan recogidos por la metodología Scrum). Por otro lado, los artefactos de Scrum son los elementos que representan un valor o trabajo y que van evolucionando a medida que avanzan los distintos Sprints [57].
- **Waterfall:** la metodología Waterfall (o metodología en cascada) se trata de un enfoque para la gestión de proyectos que proporciona una progresión lineal desde el inicio del proyecto hasta que este finaliza [58]. Este tipo de metodología divide el proyecto en varias fases como análisis, diseño, desarrollo, pruebas, documentación, etc. La característica principal de Waterfall es que se trata de un proceso de desarrollo secuencial, lo que significa que no se puede iniciar una fase hasta que se complete la fase anterior. Algunas de las ventajas de usar esta metodología es el hecho de poder detectar errores de diseño en las primeras fases del proyecto (diseño y análisis) antes de llegar a la implementación, evitando así errores de codificación provocados por una mala planificación. Sin embargo, esta metodología es muy poco flexible y su uso está relegado a proyectos cuyos requisitos son invariables a lo largo del proceso de desarrollo.
- **Modelo Incremental:** se trata de una metodología para el proceso de desarrollo del software donde se dividen todas las fases del proyecto en módulos independientes. Este modelo también es conocido como modelo de versión sucesiva, esto quiere decir que, en primer lugar, se desarrolla una versión inicial del sistema que incorpora todos los módulos que van a formar parte del proyecto, pero solo implantando características básicas. Una vez realizada una primera versión del sistema minimalista con algunas características básicas y se entrega al cliente, se procede a realizar otras versiones que van incorporando nuevas características y funciones hasta, finalmente, lograr el sistema deseado. Esto

permite poder modificar el sistema sin muchas complicaciones si se produce algún error ya que solo hay que volver a la versión anterior si el cliente no queda satisfecho con la versión actual [59].

Para el desarrollo del presente documento se ha optado por utilizar la metodología incremental, debido a que, es posible dividir todas las tareas del proyecto en módulos independientes e ir realizando distintas versiones de estos, para ir mejorando la versión anterior, con el fin de obtener un proyecto final que cumpla todas las expectativas. Gracias a utilizar este modelo es más sencillo detectar errores, adaptarse a una fecha específica para la entrega del proyecto y poder gestionar todas las tareas de una forma más eficiente y eficaz.

5.2 Planificación inicial

En este apartado se encuentra la planificación inicial del proyecto antes de ser llevado a cabo. Para ello se han especificado distintas tareas principales que a su vez pueden estar formadas por subtareas. El proyecto tiene como fecha de inicio el 20 de enero de 2022 y como fecha de fin el 2 de junio de 2022 con una duración total de 133 días.

Nombre de la tarea	Fecha Inicio	Fecha Fin	Duración (días)
Planificación inicial	20/01/2022	22/01/2022	2
Desarrollo del documento	23/01/2022	11/03/2022	47
Introducción	23/01/2022	29/01/2022	6
Estado del arte	30/01/2022	15/02/2022	16
Análisis del problema	16/02/2022	02/03/2022	14
Evaluación	03/03/2022	11/03/2022	8
Desarrollo herramienta	12/03/2022	10/05/2022	59
Configuración inicial	12/03/2022	15/03/2022	3
Conexión base datos	15/03/2022	16/03/2022	1
Implementación funcionalidades	17/03/2022	01/05/2022	45
Estilos y retroalimentación	02/05/2022	10/05/2022	8
Documentación	11/05/2022	02/06/2022	22
Desarrollo total del proyecto	20/01/2022	02/06/2022	133

Tabla 39 Planificación inicial

A continuación, se muestra el diagrama de Gantt asociado a la planificación inicial:

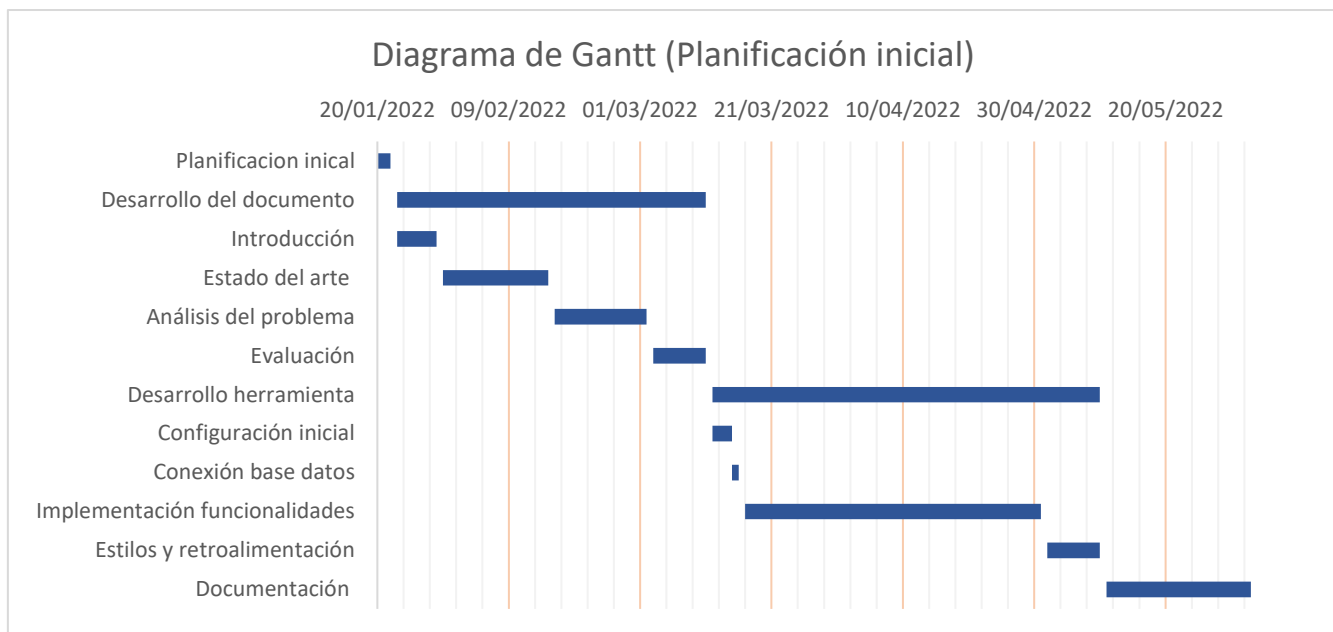


Ilustración 21 Diagrama de Gantt (Planificación inicial)

5.3 Planificación real

Una vez realizado el proyecto es posible mostrar la cronología y duración real de las tareas y subtareas realizadas para este trabajo. El proyecto se inició el 20 de enero de 2022 como fue previsto, pero finalizó el 10 de junio de 2022.

Nombre de la tarea	Fecha Inicio	Fecha Fin	Duración (días)
Planificación inicial	20/01/2022	22/01/2022	2
Desarrollo del documento	23/01/2022	15/03/2022	51
Introducción	23/01/2022	29/01/2022	6
Estado del arte	30/01/2022	19/02/2022	20
Análisis del problema	20/02/2022	06/03/2022	14
Evaluación	07/03/2022	15/03/2022	8
Desarrollo herramienta	16/03/2022	10/05/2022	55
Configuración inicial	12/03/2022	15/03/2022	3
Conexión base datos	15/03/2022	16/03/2022	1
Implementación funcionalidades	17/03/2022	04/05/2022	48
Estilos y retroalimentación	05/05/2022	10/05/2022	5
Documentación	11/05/2022	10/06/2022	30
Desarrollo total del proyecto	20/01/2022	10/06/2022	141

Tabla 40 Planificación real

Como se puede observar en la tabla anterior la duración total del proyecto para la planificación real es de 141 días frente a los 133 días de la planificación inicial. Por tanto, el proyecto finalizó 8 días después de lo esperado.

En la *Tabla 41* se hace una comparación entre la planificación inicial y final para poder observar de forma más concisa las diferencias entre ambas planificaciones:

Nombre de la tarea	Días (Inicial)	Días (Real)	Diferencia	Desviación
Planificación inicial	2	2	0	0,00%
Desarrollo del documento	47	51	4	8,51%
Introducción	6	6	0	0,00%
Estado del arte	16	20	4	25,00%
Análisis del problema	14	14	0	0,00%
Evaluación	8	8	0	0,00%
Desarrollo herramienta	59	55	-4	-6,78%
Configuración inicial	3	3	0	0,00%
Conexión base datos	1	1	0	0,00%
Implementación funcionalidades	45	48	3	6,67%
Estilos y retroalimentación	8	5	-3	-37,50%
Documentación	22	30	8	36,36%
Desarrollo total del proyecto	133	141	8	6,02%

Tabla 41 Desviación entre planificación inicial y real

La siguiente ilustración contiene el diagrama de Gantt para la planificación real:

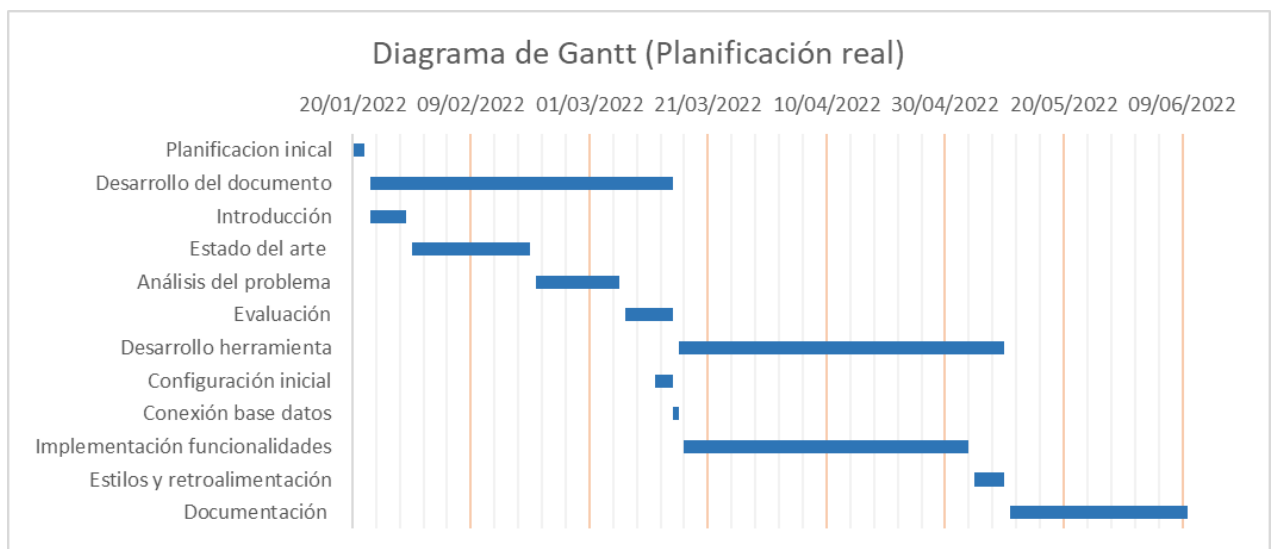


Ilustración 22 Diagrama de Gantt (Planificación real)

6. ENTORNO SOCIOECONÓMICO

En esta sección se expone el presupuesto tenido en cuenta para la elaboración del proyecto y algunas consideraciones sobre aspectos económicos.

6.1 Presupuesto

Para realizar el presupuesto del presente trabajo es importante tener en cuenta las siguientes cuestiones:

- La unidad monetaria utilizada es el Euro (€) con dos cifras decimales.
- El impuesto aplicado para el coste total del proyecto será el IVA (Impuesto Valor Añadido) cuyo valor es actualmente del 21%.
- El porcentaje de amortización por defecto es del 33% como nivel máximo anual. Esto ha sido decidido en base a lo establecido por la Agencia Tributaria española para “Sistemas y programas informáticos” [60].

Para calcular correctamente los costes imputables es necesario tener en cuenta la amortización de los activos, cuya fórmula de cálculo es:

$$\text{Amortización} = (\text{precio activo}) * \% \text{ de amortización anual} * \left(\frac{\text{Meses de uso}}{\text{Meses año}} \right)$$

El tiempo dedicado al proyecto queda reflejado y detallado en la sección 5.2. En total, han sido necesarios 5 meses comprendidos desde mediados de enero de 2022 y principios de junio de ese mismo año, en los que se han trabajado 141 días teniendo en cuenta días de descanso. La media diaria de horas trabajadas ha sido de 3 horas, por lo que se obtienen un total de 423 horas. A continuación, se muestra una tabla que resumen esta información.

Tiempo total de desarrollo del proyecto	
Meses	5
Días	143
Horas diarias trabajadas	3 horas
Horas totales	423 horas

Tabla 42 Tiempo total de desarrollo del proyecto

6.1.1 Desglose presupuestario

En este apartado se detallan los costes de los recursos software, recursos hardware, recursos humanos, costes indirectos y se hace un resumen de los costes totales del proyecto.

Recursos software

En la siguiente tabla se detallan los costes en base al software utilizado para la realización del proyecto:

Costes Recursos Software		
Recurso Software	Precio	Amortización
Microsoft Windows 10 – Education Edition	0,00 €	0,00 €
Microsoft Word y Microsoft Excel	0,00 €	0,00 €
Visual Studio Code	0,00 €	0,00 €
Node.js	0,00 €	0,00 €
GitHub	0,00 €	0,00 €
Draw.io	0,00 €	0,00 €
Javascript	0,00 €	0,00 €
Mongodb	0,00 €	0,00 €
Mongodb Compass	0,00 €	0,00 €
Zotero	0,00 €	0,00 €
Total:	0,00 €	

Tabla 43 Costes recursos software

Recursos hardware

Este subapartado se encarga de detallar el coste del hardware utilizado para la realización del proyecto. Aquí se utiliza un porcentaje de amortización anual de un 20% debido a que es lo que establece la Agencia Tributaria española para los “Equipos electrónicos”.

Coste Recursos Hardware		
Recurso hardware	Precio Compra	Amortización
OMEN by HP Laptop	1100,00 €	91,66 €
Total	91,66 €	

Tabla 44 Costes recursos hardware

Recursos humanos

El coste de los recursos humanos se ha llevado a cabo como si una empresa hubiera realizado el proyecto. Para la correcta puesta en marcha de la herramienta desarrollada es necesario contar con al menos cuatro profesionales especializados en ingeniería informática, como un analista, un diseñador web, un programador backend y un programador encargado de realizar las pruebas del sistema. Como salario de referencia para los trabajadores se han utilizado los valores que ofrece la página web profesional talent.com [61]. La *Tabla 45* refleja esta información desglosando el coste total de los recursos humanos del proyecto en base a los trabajadores.

Coste Recursos Humanos				
Rol	Analista	Diseñador web	Programador	Pruebas
Días de trabajo	130	50	130	13
Horas/día	1	2	1	2
Salario anual medio	27.001,00 €	24.239,00 €	27.500,00 €	27.500,00 €
Salario/hora	13,85 €	12,43 €	14,10 €	14,10 €
Total por trabajador	1800,50 €	1243,00 €	1833,00 €	366,60 €
Total	5343,1 €			

Tabla 45 Costes recursos humanos

Costes indirectos

La siguiente tabla refleja los costes indirectos del proyecto en base a la duración total de este (5 meses).

Costes Indirectos		
Nombre	Coste Mensual	Total
Luz	15,00 €	75,00 €
Internet	10,00 €	50,00 €
Total	25,00 €	125,00€

Tabla 46 Costes indirectos

Resumen de costes

La *Tabla 47* muestra un resumen de los costes del proyecto incluyendo la suma de los costes parciales, impuestos y un beneficio del 18%.

Concepto	Coste
Coste recursos software	0,00 €
Coste recursos hardware	91,66 €
Coste recursos humanos	5343,10 €
Costes indirectos	125,00€
Total Costes	5559,10 €
Beneficio (20%)	1111,82 €
Total (sin IVA)	6670,90 €
Total (IVA)	8071.81 €

Tabla 47 Resumen de costes

6.2 Impacto Socioeconómico

Este apartado trata de analizar el impacto esperado de la herramienta web desarrollada y descrita en el presente documento en términos sociales, éticos, económicos, medioambientales, etc.

En primer lugar, la realización de esta herramienta web ha supuesto un gran impacto en el autor del presente documento, debido a que le ha brindado numerosos conocimientos de desarrollo web tanto backend como frontend, habilidades para realizar mejoras de código o planificar con éxito un proyecto de software. Todas estas competencias adquiridas propiciarán la inserción al mundo laboral del autor. Además, el desarrollo de

la herramienta web induce a la elaboración del presente documento el cual proporciona un análisis de tecnologías y herramientas que pueden tener un impacto propicio para futuros investigadores interesados en esta materia.

Por otro lado, si la herramienta web se desplegara en Internet supondría un impacto social y económico favorable para aquellos investigadores o desarrolladores que quieran gestionar de una manera más cómoda, fácil e intuitiva los servicios definidos por el framework de Google Grpc, debido a que podría ser utilizada por cualquier experimentador de forma gratuita bajo la licencia de la UC3M y su uso supondría un ahorro de esfuerzo considerable a los investigadores de cara a administrar un número elevado de servicios lo que permitirá llevar a cabo experimentos que requieran el control remoto de entornos inmersivos.

Por último, si el uso de la herramienta web desarrollada llegara a expandirse esto supondría un impacto medioambiental positivo debido a que para realizar experimentos con un alto número de usuarios no será necesario el desplazamiento de estos a un mismo lugar e igualmente se podrá controlar y monitorizar a los usuarios que se encuentren en el entorno inmersivo salvaguardando el medio ambiente.

7. MARCO REGULADOR

En esta sección, en primer lugar, se hace una explicación de porqué para este proyecto no se aplican Leyes de protección de datos y, en segundo lugar, se exponen las distintas licencias necesarias para todas las herramientas software utilizadas en el proyecto.

7.1 Ley de Protección de datos

Para el Proyecto desarrollado no es necesario ninguna Ley de protección de datos, debido a que se trata de un proyecto cuya finalidad es desarrollar una herramienta que ayude a los investigadores a controlar y monitorizar usuarios que se encuentren en un entorno inmersivo, por lo que las Leyes de protección de datos deben ser especificadas en base a los experimentos de los futuros investigadores y no en el presente documento.

No obstante, para simplificar el esfuerzo de futuros experimentadores a continuación se detallan las Leyes que se deben aplicar si la aplicación incorporase varios usuarios:

- **REGLAMENTO (UE) 2016/679 DEL PARLAMENTO EUROPEO Y DEL CONSEJO de 27 de abril de 2016 (Reglamento general de protección de datos) [62]:** esta ley que debe ser aplicada por todos los países de la Unión Europea trata de abordar la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos. Cuando este reglamento entró en vigor se derogó la Directiva 95/46/CE del Parlamento Europeo y del Consejo, de 24 de octubre de 1995, relativa al mismo fin, pero desactualizada debido a que Internet en esa época no manejaba tantos usuarios como en la actualidad o en 2016. Esta ley frente a la directiva derogada proporciona nuevos principios que tener en cuenta para velar por la protección de datos como el principio de responsabilidad que define que las organizaciones deben implementar mecanismos que permitan tratar los datos como exige la norma, principio de transparencia que consiste en mostrar de una manera más clara y concisa tanto los avisos legales como las políticas de privacidad y principios de protección de datos por defecto que se basan en utilizar medidas desde que se inicie el diseño del proyecto para cumplir la norma. También se incorporan nuevas obligaciones para las administraciones o empresas como realizar periódicamente evaluaciones del impacto sobre la privacidad, utilizar sellos y certificaciones, comunicar tan pronto como ocurran brechas de seguridad, etc. Por último, es importante mencionar algunos de los nuevos derechos para los ciudadanos como el brindar consentimiento a las organizaciones para tratar los datos de carácter personal con la posibilidad de ser revocados, la posibilidad de denunciar o exigir indemnizaciones, transferir datos personales entre varios proveedores o el derecho a revocar el consentimiento del préstamo para el tratamiento de datos personales [63]. Todas estas nuevas medidas que recoge la normativa europea permiten velar por la seguridad de todos los ciudadanos en lo que a protección de datos se refiere y es muy importante tenerlas presente y aplicar la norma para el correcto uso de Internet y evitar problemas de filtración de datos.

- **Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales (LOPDGDD)** [64]: esta ley española publicada en el BOE (BOE-A-2018-16673) es la encargada de sustituir, debido al aumento del uso de internet en la sociedad, a la desfasada Ley Orgánica 15/1999 de Protección de Datos de Carácter Personal. La finalidad de esta Ley es adaptar la legislación española en base al reglamento europeo definido en el apartado anterior y velar por el artículo 18.4 de la Constitución española de 1978 donde se estipula que el uso de la informática será limitado por la ley con el fin de garantizar el honor y la intimidad personal y familiar de los ciudadanos y el pleno ejercicio de sus derechos [65].

El uso de estas Leyes es más que necesario cuando se incorporan varios usuarios a cualquier aplicación web para garantizar la confidencialidad, integridad y disponibilidad de los recursos de cualquier usuario, por lo que el autor de este documento anima a los futuros experimentadores a considerar las cuestiones planteadas en este apartado.

7.2 Licencias

A continuación, se listan todas las herramientas software utilizadas para desarrollar la herramienta anteriormente detallada en el presente documento y la realización de este, junto con sus respectivas licencias.

- **Microsoft Word y Microsoft Excel:** para estos programas se ha utilizado para redactar el presente documento y realizar gráficos, respectivamente. Ambos haciendo uso de la licencia ‘Office 365 A1 Plus for students’ que proporciona la UC3M a todos sus estudiantes.
- **Visual Studio Code:** este editor de código fuente precisa de la licencia MIT (Massachusetts Institute of Technology) [66], aunque los ejecutables se encuentran bajo una licencia no libre y gratuita.
- **Node.js** [27]: el entorno utilizado para desarrollar la herramienta cuenta también con la licencia MIT.
- **GitHub** [67]: esta herramienta ha permitido alojar las distintas versiones de la herramienta desarrollada con el fin de seguir la metodología elegida para la realización del proyecto (Incremental) y mantener varias copias en Internet por si se perdía o dañaba el código en local.
- **Draw.io** [68]: esta página web ha permitido realizar esquemas para representar la arquitectura del sistema, casos de usos, etc. La licencia que permite hacer uso de la herramienta es Apache v2 [69].
- **Javascript:** este ha sido el lenguaje mayormente utilizado para codificar la herramienta web. La licencia asociada a su uso es Creative Commons Attribution [70].
- **Mongodb:** es la base de datos utilizada para almacenar los servicios y mensajes. La licencia utilizada para este gestor de base de datos es la licencia AGPL [71].
- **Mongodb Compass:** la herramienta para visualizar y administrar el conjunto de datos gráficamente precisa de la misma licencia que Mongodb: licencia AGPL.

- **Zotero:** el siguiente programa se ha usado para realizar las referencias del presente documento. La licencia necesaria es, al igual que para MongoDB y MongoDB Compass la licencia AGPL.

8. CONCLUSIONES

En esta sección se exponen las conclusiones obtenidas una vez desarrollada la herramienta y escrito el presente documento. En primer lugar, se recapitulan los objetivos para compararlos con el resultado obtenido. En segundo lugar, se expondrán posibles mejoras para futuras versiones. Por último, se explican algunas consideraciones personales por parte del autor.

8.1 Retrospectiva

En este apartado se describen los resultados del proyecto en base a los objetivos iniciales que se definieron en la sección 1.2 Objetivos.

El objetivo principal del presente documento que consistía en desarrollar una herramienta web con la capacidad de ejecutar comandos remotos en servidores inmersivos ha sido cumplido con éxito. Esto no hubiera sido posible sin lograr los objetivos secundarios. En primer lugar, el automatizar la creación de servicios se consigue autogenerando código en base a la información de estos. En segundo lugar, la interfaz gráfica ha sido probada y no es necesario conocimientos técnicos para utilizarla debido a que está diseñada de una forma intuitiva y minimalista. En tercer lugar, los servicios y los mensajes asociados se almacenan en una base de datos NoSQL. Por último, el establecer la comunicación entre la aplicación cliente desarrollada y el servidor ha sido posible gracias al framework de Google Grpc. Es importante mencionar que todas las funcionalidades asociadas a estos objetivos y la forma en que han sido implementadas se detallan en el apartado 3.2 Implementación.

Por otro lado, la redacción de este documento ha supuesto cumplir otros objetivos como realizar investigaciones sobre tecnologías basadas en el desarrollo tanto de aplicaciones web como aplicaciones remotas y analizar, investigar y aplicar metodologías de desarrollo de software actuales.

El hecho de cumplir con éxito todos estos objetivos ha supuesto un reto en términos de esfuerzo y planificación por parte del autor.

8.2 Trabajo futuro

En este apartado se listan y detallan algunas funcionalidades que no han sido implementadas en la herramienta web desarrollada por falta de tiempo. Todas estas permitirían disminuir la carga cognitiva del usuario que la utilice y/o evitar errores internos en el sistema.

- **Permitir el registro de usuarios.** Desarrollar un ‘Sign in’ con Google para que los distintos investigadores que utilicen la aplicación puedan tener sus propios servicios.
- **Función Editar Servicios.** Implementar la funcionalidad de poder editar los servicios actuales en la herramienta web, es decir, tener la posibilidad de eliminar o añadir grpc’s y poder asignar otros mensajes tanto de solicitud como de respuesta de parte del servidor en cada uno de los grpc’s actuales del servicio.

- **Cargar fichero .proto en el servidor.** Implementar una funcionalidad que envíe el fichero .proto actual al servidor encargado de recibir las llamadas a procedimiento remoto.
- **Filtrar Servicios y Mensajes.** Consiste en aplicar un filtro para registrar los mensajes y servicios con la finalidad de que independientemente de lo que introduzca el usuario en el input, recoja todos los datos y los ponga en minúscula para evitar posibles fallos de sintaxis.
- **Multisesiones.** Permitir al usuario usar varias sesiones de trabajo ayudaría a organizar de una manera óptima y eficiente experimentos independientes.

8.3 Conclusiones personales

El haber desarrollado esta herramienta web ha supuesto un reto que ha sido posible lograr gracias a los numerosos conocimientos y habilidades adquiridas a lo largo del Grado. Por un lado, asignaturas de primer y segundo curso como Programación, Estructura de Datos y Algoritmos, Sistemas Operativos, Teoría de Autómatas y Lenguajes Formales o Lógica establecieron las bases para un pensamiento crítico de cara a la resolución de problemas o retos y varios principios fundamental sobre distintas ramas de la informática. Por otro lado, las asignaturas finales del Grado como Dirección de Proyectos de Desarrollo del Software, Computación Ubicua, Metodología de desarrollo visual, Interfaces de Usuario o Tecnologías Informáticas de la Web ayudaron a afianzar conocimiento y desarrollar sistemas o prototipos funcionales como el detallado en el presente documento utilizando las herramientas y metodologías necesarias en cada caso.

Codificando la herramienta web, al principio, surgieron problemas con la conexión de la base de datos en MongoDB debido a que se utilizaba un clúster gratuito y este denegaba la conexión si no era utilizado constantemente, por lo que se decidió utilizar una conexión local. Por otro lado, diseñando la arquitectura del sistema, al comienzo se utilizó un enrutador para gestionar todas las peticiones HTTP de la aplicación de forma independiente en base a los modelos establecidos en la base de datos, pero esto provocaba errores de renderizado si se quería acceder a los recursos de ambos modelos en una misma página, por lo que para gestionar todas las rutas de la aplicación se optó por utilizar el fichero index.js como se detalla en el apartado 3.2. Por último, como el proyecto ha sido desarrollado en la plataforma Node.js utilizando Javascript esto implica que solo hay un único hilo de ejecución y ya que la escritura en el fichero .proto se hace de forma asíncrona puede ocurrir que el objeto que contiene todos los servicios no esté disponible al inicializarse antes de que el fichero esté listo, por lo que hay se han utilizado temporizadores y promesas para una correcta ejecución del programa.

Realizar la herramienta web ha supuesto un evidente aprendizaje para el autor de este documento. Se ha codificado desde cero una interfaz gráfica ‘responsive’ navegable y minimalista. Se han manejado rutas HTTP con sus respectivos métodos para orientar la aplicación en base a lo requerido. Se ha profundizado sobre el uso de módulos javascript para iniciar servidores, escribir o modificar ficheros, crear, acceder y eliminar directorios, hacer uso de alertas, resolver promesas y establecer la comunicación con servidores remotos o entre la misma aplicación con el navegador. También se han estudiado nuevos frameworks no conocidos hasta entonces para simplificar el esfuerzo a la hora de realizar consultas o modificaciones en la base de datos de una forma muy simple, eficiente y potente con escasas líneas de código o frameworks para iniciar servicios con diferentes

métodos que envían y reciben RPC's. Por último, se ha enriquecido el conocimiento adquirido en el lenguaje de programación Javascript, desde la autogeneración de código html hasta el renderizado de páginas web.

La redacción del presente documento también ha aportado beneficios en el autor como el entender el dilema del uso de dispositivos inalámbricos de RV y proponer una herramienta que ayude a solucionar la cuestión; analizar y juzgar diversas tecnologías actuales para desarrollo web, bases de datos y comunicación entre aplicaciones remotas muchas de ellas novedosas; y también ha constituido una oportunidad para investigar y comprender artículos científicos con aportes realmente atractivos que solucionan problemas reales que presenta la RV. Asimismo, este documento ha requerido describir de una manera formal las capacidades y limitaciones del sistema desarrollado, así como la implementación de este; analizar y opinar sobre distintas metodologías de desarrollo del Software y elegir y aplicar la más conveniente para este proyecto, e Informarse de las licencias necesarias y deducir futuras leyes susceptibles de ser aplicadas a la presente aplicación.

En conclusión, estos meses de trabajo han aportado un gran número de conocimientos y nuevas competencias que no hubieran sido posible sin mantener una sólida planificación para lograr los objetivos propuestos en el tiempo previsto. El adquirir toda esta sabiduría y habilidades y realizar un proyecto de estas características con una duración predefinida favorecerá la inminente incorporación al mundo laboral del autor, lo que deriva a un resultado satisfactorio.

9. SUMMARY

9.1 Motivation and objectives

Nowadays, virtual reality goggles or glasses can be wireless or connected by cable. If the virtual reality goggles are connected by cable, they have the advantage of being able to access locally all the information available on the glasses. However, if they are used, the number of users who can interact with the immersive environment is limited based on the number of computers available and conducting experiments outside a specific working environment is hampered by the tediousness of transporting machines and devices from one location to another.

The current trend among companies and researchers is to use standalone glasses, because they can be used in any environment, do not need to be connected to a computer and allow experiments with a larger number of users. However, wireless devices require an additional software layer to monitor the user who is immersed in a VR or MRI system and, in addition, most holographic devices do not allow access to the information recorded on the glasses until the user stops using them, because this type of lens has its own file system and does not have a connection to a computer. Another problem of using this type of device is that it is difficult to carry out tests, because it is necessary to create a compilation (for example, an APK) and run it on the device itself [7].

Therefore, to try to mitigate some of the drawbacks introduced by the new standalone HMD models as well as to obtain, modify or configure elements of the environment in which the user is immersed when using this type of wireless devices, it is necessary to develop a tool that allows the control and monitoring of the immersive environment remotely. Furthermore, given that this type of tool can potentially be applied to different environments, for research or prototyping purposes of immersive systems, this tool must be flexible enough to allow the user to adapt it to their specific use case and minimize the effort or technical knowledge required for its use.

The main objective of this work is to develop a web tool that allows remote commands to be executed in a server application that supports an immersive Virtual Reality system. The client application will be able to both generate new services and execute existing ones without the need for technical knowledge. To successfully carry out the main objective it needs.

- Automate the creation of remote services.
- Design and adapt a graphical interface so that any non-technical user can use it.
- Store the services in a database.
- Establish communication between a client application and the server for the successful execution of remote services.

In addition, the following objectives will be necessary for the correct implementation of this document and the web tool:

- Research and analyze current technologies for the development of web applications and remote applications.
- Research and analyze the current software development methodologies for this type of project and apply them.
- To present the conclusions and results obtained.

9.2 State of the art

9.2.1 Grpc

gRPC [12] is a high-performance open-source software developed by Google that uses a remote procedure call (RPC) framework that can run in any environment. By using gRPC, a client application can directly call a method of a server application (which may be located on another machine) as if it were a local object, thus facilitating the creation of both applications and distributed services. As in many RPC systems, gRPC is based on the idea of defining a service, specifying in advance those methods that can be called remotely, which are characterized by their parameters and what they return. The server implements this interface and runs a gRPC server to handle calls from the client. The client uses a stub [13] that provides the same methods that have been previously defined on the server.

Both clients and servers using gRPC can communicate with each other in a wide variety of environments, from any desktop to Google servers. The latest Google APIs have gRPC versions of their interfaces, allowing Google functionality to be easily incorporated into applications using this technology. Client and server can be written in any of the languages supported by gRPC: C#, C++, Dart, Go, Java, Kotlin, Node, Objective-C, PHP, Python and Ruby.

By default, gRPC uses Protocol Buffers [14], a robust open-source mechanism developed by Google for serializing structured data, although it can be used with other data formats such as JSON. When working with protocol buffers it is necessary to define the data structure to be serialized in a proto file: this is an ordinary text file with the extension ".proto". The data in the protocol buffer is defined as messages, where each message is a small logical record of information containing a few fields. In the proto file, gRPC services are defined which have RPC method parameters and need return types specified as protocol buffer messages.

9.2.2 Express

Express [39] is a framework used to develop web server projects in the Node.js runtime environment. It consists of a routing layer and is deployed on top of the node.js HTTP server. This framework provides a basic middleware pattern and declarative routing.

The structure of express is composed of three layers that communicate with each other. First, we have the top layer expressjs, which has the middlewares in charge of managing both the requests and the stack. The second layer downwards is pillarjs, which has the

express router, the dispatch module that contains the logic to read files from the system to later group and send them and, finally, the final controller where the different functions are defined and is in charge of serialising errors. The last layer is jshttp which has three modules to manage the top pillarjs layer sending module which are the etag module which sends request and response information via tags, the fresh module which decides how to send those tags and the range-passer module which needs to parse the ranges. The jshttp layer also has an end module that helps manage requests from the pillarjs layer's end controller.

To install express in your node.js application, simply run 'npm install express' and declare a variable in the Javascript code using the 'require' module. Once the previous steps have been carried out, it is possible to manage the routes of the application created and the different request methods such as 'GET', 'PUT', 'POST', 'USE', etc.

Express is characterized by its high performance in web applications as it allows the integration of "view" rendering engines, such as EJS [29], to display responses that are treated as objects. It helps to easily handle HTTP routes, has many supported middleware packages, allows easy integration into JavaScript projects, etc.

9.2.3 MongoDB

Non-relational or NoSQL databases such as MongoDB [51] have the capacity to manage both structured and semi-structured data because they do not use the relational model. They do not use SQL either, since to access and manipulate the data, they use an object-based API, which is how the different entities in the database are represented. The most relevant characteristics of NoSQL databases are their high performance and functionality, the fact that they allow flexible schemas that favors a better organization of the information and their high scalability thanks to the use of clusters.

MongoDB [38] is an open-source non-relational database management system. It is characterized by the fact that it stores data in documents, which have a BSON format, i.e., a binary representation of JSON. The documents are stored in collections, which would be a table in MySQL, but with the advantage that the documents can have different attributes and even be of different types.

Characteristics such as the adaptability of documents to become objects, the ease of queries and insertions, indexing, the fact that it is free to use, etc. make it one of the most widely used systems in projects that require a database.

9.2.4 Conclusions on the state of the art

To reach solid conclusions, it was decided to analyze tools or frameworks that aim to examine the behavior and/or interaction of users who are immersed in a virtual reality environment. The articles analyzed were ImmVis: Bridging Data Analytics and Immersive Visualization [17], Improved Knowledge from Data: Building an Immersive Data Analysis Platform [23], Welicit: A Wizard of Oz Tool for VR Elicitation Studies [25] and Toggle toolkit: A tool for conducting experiments in unity virtual environments [28].

Having analysed the technologies that enable the execution of remote functions in web applications and the different tools or frameworks described in the previous paragraph, it was possible to draw the following conclusions:

- To visualize and develop the immersive environment, a good option is to make use of the Unity multiplatform videogame engine, as it allows the 3D content to be displayed and programmed in a simple and minimalist way, it incorporates an API to interact with various VR devices, it is not necessary to implement external plugins for each device, etc.
- Using a large number of interconnected scripts is not good practice because it can be complicated and cumbersome for the researcher to capture the various interactions of a user in an immersive environment.
- When working with a large volume of data, it is convenient to use Python as it is a language that incorporates powerful and fast libraries for data analysis and visualization, statistics or machine learning.
- It is convenient to use Grpc in systems that require some form of communication between multilingual applications due to the ease of sending and receiving RPCs.

The tool to be developed is a client-side web application with the ability to send RPCs to a server and receive an associated response from the server. In addition, the tool must simplify the creation and execution of services that define the remote procedure calls. After studying in section 2.3 all the current technologies for the development of web applications, the option of using Express in Node.js is considered, given that a powerful and easy to use graphical interface is required and these technologies allow it satisfactorily.

Regarding the database, it has been decided to use MongoDB, because, although relational databases are more robust and ensure data integrity, in order to develop the application defined in this document, greater flexibility is required since the information and structure of the data is variable due to the use of remote functions.

9.3 Problem analysis

The tool to be developed will be able to:

- Generate and delete message types necessary for the creation of services.
- Generate and delete services.
- Establish communication with the server to send previously defined remote commands in the form of services.

The tool, in the first instance, is developed for researchers interested in communication between users who are using a wireless virtual reality device and a server in charge of creating the immersive environment.

Its use can be expanded to subjects who want to control/monitor multiple immersed users in a VR system via a wireless device. The subject or administrator could remotely manage all events of each immersed user.

Services can consist of from one to N RPCs, where N is a positive integer greater than one. Each RPC must consist of two messages. On the one hand, a message of type "Request" that consists of a JSON object that will be sent to the server that supports the immersive environment. On the other hand, a message of type "Response" that will also be a JSON object, but in this case, it will be the response from the server that will be received by the client

The user will be able to run the default services provided by the tool and create new services using both existing 'Request' and 'Response' type messages. However, if the database has not been populated, it is important to take into account that it will not be possible to create a service because the application only allows the selection of existing messages, so in order to carry out this task the tool must have at least one message.

9.3.1 Implementation

To establish communication between the client and server application and to be able to send and receive RPCs, the Google GRPC framework has been used, which will be listening on port 50051 of the server application to receive the calls that will be sent using the same port from the client application. The server application, which will support the immersive environment, should ideally be developed in the Unity environment, although researchers can use any other environment available with the languages supported by GRPC [52].

The client application has been developed in the Node.js environment using the Express framework. It starts an HTTP server deployed locally (localhost) using port 3000. To carry out the communication between the client and the server with GRPC it is necessary to use a ".proto" file that is formed by the services and their respective RPCs. This file is generated dynamically in the client application based on the current services and

messages in the database and to write to it, the fs file system module for Node.js [40] is used. To receive the remote commands sent by the web tool to the Unity server, it is currently necessary to copy the file generated in the client application and paste it into the corresponding repository on the server.

Regarding the database, it has been decided to use the NoSQL database system MongoDB and the tool chosen to manage the data has been MongoDB Compass [53], which allows to visualize and manage in a graphical way the collections and their respective documents. The database has been hosted locally (localhost) on port 27017. It is important to mention the use of the Mongoose programming library [54], which is developed to be used in the Node.js environment and is the one that has allowed the connection and all the database operations to be carried out in a very simple and efficient way.

The architecture of the developed system follows the MVC model. This software design pattern consists of dividing the application logic into three: the model, which will oversee managing the data and business logic; the view, which deals with the design and presentation and, finally, the controller, which is responsible for managing the application routes and directing the view and the model. Firstly, for the model, the entities 'Service' and 'Messages' have been represented using the schemas (Schema) coded in JavaScript provided by the Mongoose library. Secondly, to show the elements of the model in a fast, simple, and efficient way, we have made use of the JavaScript template for rendering EJS views, which has allowed us to treat the entities of the model as objects and, therefore, to generate all the necessary HTML in a more comfortable way. Finally, the controller, which oversees redirecting all HTTP requests, performing all database operations, sending commands to the server and everything related to the application logic, which is coded in the 'index.js' file in the root directory of the client application.

Most of the styles that the client can see in the browser have been implemented using the cross-platform Bootstrap library [55], specifically version 5. The rest of the styles have been implemented by the author of this document.

Initial Settings

To avoid errors when starting the tool, the following considerations must be considered:

- A NoSQL database must exist on the local host with port 27017 on the machine where the tool is being used. This database must contain a database with the name 'grpcDB' and two collections called 'messages' and 'services'.
- Ports 3000 and 50051 of the machine where the tool is being used must not have any services running.

Considering the previous observations, to run the tool, we must access the root directory containing the application, open a terminal and launch the command 'node index.js'. From this moment on, if you access the web address 'http://localhost:3000/' you will be able to use the tool.

Create your first service

It is important to remember at this point that a service must consist of two messages. The first message will be of type 'request' and will be received by the server. The second message will be of type 'reply' and will contain the server's response.

First, we access the create service page and look at the section 'Create your new message type!'. As can be seen, initially there are two fields, one for the name of the message and the other for the name of the attribute, which is of type 'string' by default, although the user can select any of the types supported by Grpc: 'string', 'int32' and 'boolean'. Between the two fields there is a green button with a '+' symbol which, if pressed, will show an input below the first attribute reserved for an extra attribute, because the message can be composed of as many attributes as the user wishes.

Once the two necessary message types have been created, the user can create their first service without leaving the page itself. The first step is to go to the 'Create your new service!' form. Here there are two fields to fill in with text, one reserved for the name of the service and the other for the name of the first RPC and two fields to select the messages corresponding to the 'Request type' and 'Reply Type' of the first grpc, which allow you to select any existing message in the database. As with the form for messages, there is a green button which, if pressed, will create three fields under the first RPC for the name, 'Request type' and 'Reply Type', respectively, of the second RPC, since a Service can be made up of as many RPCs as the experimenter wishes.

The web tool has no limit to the number of services and messages it can store. The user can delete any of them at any time by clicking on the trash icon that is always available when any message or service is displayed.

Configure Server.

Once the services to be tested have been created on the server, the tool must be restarted. When the tool starts up again, it retrieves the current services from the database and writes them to the previously detailed 'protofile.proto' file. Then, the user must manually copy the content of the file 'protofile.proto' and paste it into the file 'helloworld.proto', which is the one used to establish communication with the server, where the following actions must be performed:

- Use as '.proto' file the 'helloworld.proto' file of the web tool.
- Add to the 'Server' object of GRPC the service to be implemented together with its respective RPCs.
- Implement callback type functions [56] for each RPC.

When the functionality is implemented on the server based on the object received by the user, the service can be executed correctly.

Run Service

To run the service it is necessary to access the 'Run service' page and follow the steps below. The first step is to select the service to be used. Once selected, the system will display a drop-down list with all the RPCs associated with that service and the user must choose one. The second step consists of filling in the fields of the attributes belonging to the 'request' message according to the experimenter's preferences. The third and last step is to press the 'submit' button.

9.4 Project plan

9.4.1 Project methodology chosen

For the development of this document, we have chosen to use the incremental methodology, because it is possible to divide all the tasks of the project into independent modules and make different versions of these, to improve the previous version, to obtain a final project that meets all expectations. Thanks to using this model it is easier to detect errors, to adapt to a specific date for the delivery of the project and to be able to manage all the tasks in a more efficient and effective way.

9.4.2 Scheduling

The total project duration for the actual planning is 141 days compared to 133 days for the initial planning. Therefore, the project ended 8 days later than expected.

9.5 Socio-economic environment

9.5.1 Budget

The total cost of the project considering the costs of human resources, hardware, software, indirect costs and a profit of 18% amounts to € 8071.81 assuming that the project has been developed by a team of professionals.

9.5.2 Socio-economic impact

First, the creation of this web tool has had a great impact on the author of this document, because it has given him a lot of knowledge of web development, both backend and frontend, skills to make code improvements or successfully plan a software project. All these acquired competences will help the author's insertion into the world of work. In addition, the development of the web tool induces the elaboration of the present document which provides an analysis of technologies and tools that can have a favorable impact for future researchers interested in this subject.

On the other hand, if the web tool were deployed on the Internet, it would have a favorable social and economic impact for those researchers or developers who want to manage the services defined by the Google Grpc framework in a more comfortable, easy and intuitive way, as it could be used by any experimenter free of charge under the UC3M license and its use would mean a considerable saving of effort for researchers in terms of managing a large number of services, which would allow experiments requiring remote control of immersive environments to be carried out.

Finally, if the use of the developed web tool were to expand, this would have a positive environmental impact, since it would not be necessary to move the users to the same place to carry out experiments with many users, and it would also be possible to control and monitor the users in the immersive environment, thus safeguarding the environment.

9.6 Legal framework

No data protection law is necessary for the developed project because it is a project whose purpose is to develop a tool that helps researchers to control and monitor users in an immersive environment, so data protection laws should be specified on the basis of the experiments of future researchers and not in this document.

Most of the licenses used to produce the document and the development of the web tool are either open source or require the MIT license.

9.7 Conclusions

9.7.1 Retrospective

The main objective of this paper, which was to develop a web tool with the ability to execute remote commands on immersive servers, has been successfully achieved. This would not have been possible without achieving the secondary objectives. Firstly, automating the creation of services is achieved by auto-generating code based on service information. Secondly, the graphical interface has been tested and no technical knowledge is required to use it because it is designed in an intuitive and minimalistic way. Thirdly, the services and associated messages are stored in a NoSQL database. Finally, establishing communication between the developed client application and the server has been possible thanks to the Google Grpc framework. It is important to mention that all the functionalities associated with these objectives and the way in which they have been implemented are detailed in section 3.2 of this document.

On the other hand, the drafting of this document has meant fulfilling other objectives such as carrying out research on technologies based on the development of both web applications and remote applications and analyzing, researching and applying current software development methodologies.

9.7.2 Future work

This section lists and details some functionalities that have not been implemented in the web tool developed due to lack of time. All of these would allow to reduce the cognitive load of the user and/or avoid internal errors in the system.

- **Allow user registration.** Develop a 'Sign in' with Google so that different researchers using the application can have their own services.
- **Edit Services function.** Implement the functionality of being able to edit the current services in the web tool, i.e. having the possibility of deleting or adding grpc's and being able to assign other messages both request and response from the server in each of the current grpc's of the service.
- **Load .proto file on the server.** Implement a functionality that sends the current .proto file to the server in charge of receiving remote procedure calls.
- **Filter Services and Menages.** It consists of applying a filter to register the messages and services so that regardless of what the user enters in the input, it collects all the data and puts them in lower case to avoid possible syntax errors.
- **Multisession.** Allowing the user to use several working sessions would help to organize independent experiments in an optimal and efficient way.

REFERENCIAS

- [1] R. Skarbez, M. Smith, y M. C. Whitton, «Revisiting Milgram and Kishino's Reality-Virtuality Continuum», *Front. Virtual Real.*, vol. 2, 2021, Accedido: 3 de mayo de 2022. [En línea]. Disponible en: <https://www.frontiersin.org/article/10.3389/frvir.2021.647997>
- [2] «Dossier_CAVE.pdf_2063069239.pdf». Accedido: 3 de junio de 2022. [En línea]. Disponible en: https://www.udc.es/export/sites/udc/campus_innova/_galeria_down/Dossier_CAVE.pdf_2063069239.pdf
- [3] «Types of VR headsets - PC VR, standalone VR, and smartphone VR», *Aniwaa*. <https://www.aniwaa.com/guide/vr-ar/types-of-vr-headsets/> (accedido 19 de junio de 2022).
- [4] «Quest 2: nuestras gafas de VR todo en uno más avanzadas | Meta Quest». <https://store.facebook.com/es/quest/products/quest-2/> (accedido 3 de mayo de 2022).
- [5] «Os damos la bienvenida a Meta | Meta». <https://about.facebook.com/es/meta/> (accedido 3 de junio de 2022).
- [6] «Pico Neo 3 | Empowering Leaders in Enterprise VR | 6DoF All-In-One VR headset». <https://www.pico-interactive.com/eu/en/neo3.html> (accedido 13 de mayo de 2022).
- [7] «Microsoft HoloLens | Tecnología de realidad mixta para empresas». <https://www.microsoft.com/es-es/hololens> (accedido 13 de mayo de 2022).
- [8] A. R. López, «A Tool for Monitoring and Controlling Standalone Immersive HCI Experiments», p. 3.
- [9] «What Is Remote Procedure Call (RPC)? Definition from SearchAppArchitecture», *SearchAppArchitecture*. <https://www.techtarget.com/searchapparchitecture/definition/Remote-Procedure-Call-RPC> (accedido 19 de junio de 2022).
- [10] «Client-Server Model», *GeeksforGeeks*, 23 de octubre de 2019. <https://www.geeksforgeeks.org/client-server-model/> (accedido 19 de junio de 2022).
- [11] «Remote Procedure Call (RPC) in Operating System», *GeeksforGeeks*, 30 de agosto de 2017. <https://www.geeksforgeeks.org/remote-procedure-call-rpc-in-operating-system/> (accedido 19 de junio de 2022).
- [12] «gRPC», *gRPC*. <https://grpc.io/> (accedido 5 de abril de 2022).
- [13] hmong.wiki, «Stub (informática distribuida)». [https://hmong.es/wiki/Stub_\(distributed_computing\)](https://hmong.es/wiki/Stub_(distributed_computing)) (accedido 5 de abril de 2022).
- [14] «Protocol Buffers», *Google Developers*. <https://developers.google.com/protocol-buffers> (accedido 5 de abril de 2022).
- [15] «Getting Started with JAX-RPC». <https://www.oracle.com/technical-resources/articles/javase/getstartjaxrpc.html> (accedido 19 de junio de 2022).
- [16] «Primeros pasos con JAX-RPC». <https://www.oracle.com/technical-resources/articles/javase/getstartjaxrpc.html> (accedido 22 de abril de 2022).
- [17] «ImmVis: Bridging Data Analytics and Immersive Visualisation - VISIGRAPP 2021». <https://www.insticc.org/node/TechnicalProgram/visigrapp/2021/presentationDetails/102560> (accedido 5 de abril de 2022).
- [18] I. CORPORATIVA, «¿Somos conscientes de los retos y principales aplicaciones de la Inteligencia Artificial?», *Iberdrola*.

- <https://www.iberdrola.com/innovacion/que-es-inteligencia-artificial> (accedido 5 de abril de 2022).
- [19] «Welcome to Python.org», *Python.org*. <https://www.python.org/> (accedido 5 de abril de 2022).
- [20] M. Cordeil *et al.*, «Immersive Analytics», sep. 2015. doi: 10.1109/BDVA.2015.7314296.
- [21] D. Filonik, T. Bednarz, M. Rittenbruch, y M. Foth, «Glance: generalized geometric primitives and transformations for information visualization in AR/VR environments», en *Proceedings of the 15th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry - Volume 1*, New York, NY, USA, dic. 2016, pp. 461-468. doi: 10.1145/3013971.3014006.
- [22] U. Technologies, «Plataforma de desarrollo en tiempo real de Unity | Motor de VR y AR en 3D y 2D». <https://unity.com/es> (accedido 6 de abril de 2022).
- [23] F. Pedroso y P. Costa, «Improved Knowledge from Data: Building an Immersive Data Analysis Platform», oct. 2018.
- [24] C. Ardito, P. Buono, M. F. Costabile, R. Lanzilotti, y A. Piccinno, «A tool for Wizard of Oz studies of multimodal mobile systems», en *2009 2nd Conference on Human System Interactions*, may 2009, pp. 344-347. doi: 10.1109/HSI.2009.5091003.
- [25] A. Bellucci, T. Zarraonandia, P. Díaz, y I. Aedo, «Welicit: A Wizard of Oz Tool for VR Elicitation Studies», en *Human-Computer Interaction – INTERACT 2021*, Cham, 2021, pp. 82-91. doi: 10.1007/978-3-030-85607-6_6.
- [26] «Visual Inspector & Dev Tools», *A-Frame*. <https://aframe.io> (accedido 3 de junio de 2022).
- [27] «File system | Node.js v18.3.0 Documentation». <https://nodejs.org/api/fs.html> (accedido 3 de junio de 2022).
- [28] P. Ugwitz, A. Šašinková, Č. Šašinka, Z. Stachoň, y V. Juřík, «Toggle toolkit: A tool for conducting experiments in unity virtual environments», *Behav. Res. Methods*, vol. 53, n.º 4, pp. 1581-1591, ago. 2021, doi: 10.3758/s13428-020-01510-4.
- [29] T. Groussard, *Java Enterprise Edition: Desarrollo de aplicaciones web con JEE* 6. Ediciones ENI, 2010.
- [30] «Eclipse IDE for Java Developers | Eclipse Packages». <https://www.eclipse.org/downloads/packages/release/kepler/sr1/eclipse-ide-java-developers> (accedido 3 de junio de 2022).
- [31] «Welcome to Apache NetBeans». <https://netbeans.apache.org/> (accedido 19 de junio de 2022).
- [32] A. C. N. Lemus, «ARQUITECTURA POR COMPONENTES JEE, UN CASO PRÁCTICO», *Rev. GTI*, vol. 14, n.º 38, Art. n.º 38, 2015.
- [33] «Generalidades del protocolo HTTP - HTTP | MDN». <https://developer.mozilla.org/es/docs/Web/HTTP/Overview> (accedido 19 de junio de 2022).
- [34] «Spring | Home». <https://spring.io/> (accedido 3 de junio de 2022).
- [35] «MVC - Glosario | MDN». <https://developer.mozilla.org/es/docs/Glossary/MVC> (accedido 1 de junio de 2022).
- [36] «Que es REST y que Es RESTful», 16 de enero de 2018. <https://codigonaranja.com/restful-web-service> (accedido 3 de junio de 2022).
- [37] C. Á. Caules, «¿Qué es un POJO en Java?», *Arquitectura Java*, 15 de septiembre de 2021. <https://www.arquitecturajava.com/que-es-un-pojo-en-java/> (accedido 3 de junio de 2022).

- [38] «JavaScript | MDN». <https://developer.mozilla.org/es/docs/Web/JavaScript> (accedido 23 de abril de 2022).
- [39] «Express - Infraestructura de aplicaciones web Node.js». <https://expressjs.com/es/> (accedido 23 de mayo de 2022).
- [40] «EJS -- Embedded JavaScript templates». <https://ejs.co/> (accedido 1 de junio de 2022).
- [41] «The web framework for perfectionists with deadlines | Django». <https://www.djangoproject.com/> (accedido 3 de junio de 2022).
- [42] «PHP: ¿Qué es PHP? - Manual». <https://www.php.net/manual/es/intro-whatis.php> (accedido 23 de abril de 2022).
- [43] «Breve historia de PHP». <https://desarrolloweb.com/articulos/436.php> (accedido 23 de abril de 2022).
- [44] «PHP: ¿qué es, para qué sirve y cuáles son sus características?», *Rock Content - ES*, 9 de marzo de 2020. <https://rockcontent.com/es/blog/php/> (accedido 23 de abril de 2022).
- [45] «SQL Introduction». https://www.w3schools.com/sql/sql_intro.asp (accedido 19 de junio de 2022).
- [46] «¿Qué es una base de datos relacional?». <https://www.oracle.com/es/database/what-is-a-relational-database/> (accedido 6 de junio de 2022).
- [47] «Bases de datos SQL | AWS», *Amazon Web Services, Inc.* <https://aws.amazon.com/es/relational-database/> (accedido 6 de junio de 2022).
- [48] «¿Qué es MySQL?», *Ayuda | dinahosting*, 23 de enero de 2019. <https://dinahosting.com/ayuda/que-es-mysql/> (accedido 23 de abril de 2022).
- [49] «Explicación Sobre Las Bases De Datos NoSQL», *MongoDB*. <https://www.mongodb.com/es/nosql-explained> (accedido 19 de junio de 2022).
- [50] «Bases de datos no relacionales | Bases de datos de gráficos | AWS», *Amazon Web Services, Inc.* <https://aws.amazon.com/es/nosql/> (accedido 6 de junio de 2022).
- [51] «Qué es MongoDB y características | OpenWebinars». <https://openwebinars.net/blog/que-es-mongodb/> (accedido 31 de marzo de 2022).
- [52] «Supported languages», *gRPC*. <https://grpc.io/docs/languages/> (accedido 16 de junio de 2022).
- [53] «MongoDB Compass», *MongoDB*. <https://www.mongodb.com/es/products/compass> (accedido 16 de junio de 2022).
- [54] «Mongoose ODM v6.3.6». <https://mongoosejs.com/> (accedido 13 de junio de 2022).
- [55] M. O. contributors Jacob Thornton, and Bootstrap, «Bootstrap». <https://getbootstrap.com/> (accedido 1 de junio de 2022).
- [56] Node.js, «What are callbacks?», *Node.js*. <https://nodejs.org/en/knowledge/getting-started/control-flow/what-are-callbacks/> (accedido 15 de junio de 2022).
- [57] Atlassian, «Scrum: qué es, cómo funciona y por qué es excelente», *Atlassian*. <https://www.atlassian.com/es/agile/scrum> (accedido 6 de junio de 2022).
- [58] «Gestión de proyectos: ¿Qué es el modelo secuencial Waterfall y cómo funciona este método en cascada?», *Alaimo Labs*. <https://alaimolabs.com/es/self-learning/scrum/gestion-de-proyectos-que-es-el-modelo-secuencial-waterfall-y-como-funciona-este-metodo-en-cascada> (accedido 6 de junio de 2022).
- [59] «Qué es el modelo incremental», *Blog - ComparaSoftware*, 13 de mayo de 2021. <https://blog.comparasoftware.com/que-es-el-modelo-incremental/> (accedido 6 de junio de 2022).

- [60] «Agencia Tributaria: Amortizaciones». <https://sede.agenciatributaria.gob.es/Sede/impuesto-sobre-sociedades/que-base-imponible-se-determina-sociedades/amortizaciones.html> (accedido 10 de junio de 2022).
- [61] «Salario en España - Salario Medio», *Talent.com*. <https://es.talent.com/salary> (accedido 10 de junio de 2022).
- [62] *Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE (Reglamento general de protección de datos) (Texto pertinente a efectos del EEE)*, vol. 119. 2016. Accedido: 11 de junio de 2022. [En línea]. Disponible en: <http://data.europa.eu/eli/reg/2016/679/oj/spa>
- [63] «RGPD - Reglamento General de Protección de datos». <https://rgpd.es/> (accedido 11 de junio de 2022).
- [64] M. B. Samper, *Protección de datos personales: Esquemas*, 1.^a ed. Dykinson, 2020. doi: 10.2307/j.ctv17hm980.
- [65] «Título I. De los derechos y deberes fundamentales - Constitución Española». <https://app.congreso.es/consti/constitucion/indice/titulos/articulos.jsp?ini=18&tipo=2> (accedido 9 de junio de 2022).
- [66] «The Massachusetts Institute of Technology (MIT)», *Massachusetts Institute of Technology*. <http://web.mit.edu> (accedido 31 de mayo de 2022).
- [67] «GitHub: Where the world builds software», *GitHub*. <https://github.com/> (accedido 19 de junio de 2022).
- [68] *jgraph/drawio*. JGraph, 2022. Accedido: 19 de junio de 2022. [En línea]. Disponible en: <https://github.com/jgraph/drawio>
- [69] «Apache License, Version 2.0». <https://www.apache.org/licenses/LICENSE-2.0.html> (accedido 19 de junio de 2022).
- [70] «Creative Commons — Reconocimiento 3.0 España — CC BY 3.0 ES». <https://creativecommons.org/licenses/by/3.0/es/> (accedido 1 de junio de 2022).
- [71] iText PDF, «Licencia AGPL», *iText PDF*, 12 de octubre de 2018. <https://itextpdf.com/es/how-buy/agpl-license> (accedido 1 de junio de 2022).