Trabajo Fin de Grado

Herramienta de creación y visualización de escenarios de realidad aumentada para videojuegos

Autor: Luis Arenas Rivera
Tutor: Telmo Zarraonandia Ayo

Fecha: Leganés, mayo de 2014

AGRADECIMIENTOS

Dedico este Trabajo Fin de Grado a mis padres José y Maribel, por su apoyo incondicional y por ofrecerme siempre los mejores medios para mi educación.

Quisiera agradecer a mi hermano Carlos su elevado nivel de exigencia académica, el cual ha supuesto siempre un reto a superar a través de una sana competición.

A mi hermano Daniel quiero agradecerle todos sus consejos sobre la carrera y la universidad, las cuales conoce de primera mano y han servido para allanar mi camino.

Por otro lado, quisiera agradecer a todos mis compañeros de la carrera -tanto del grupo bilingüe como del español- la ayuda que me han prestado y los buenos ratos que hemos pasado juntos. De este grupo me gustaría destacar a una persona en particular, mi compañera de prácticas Estefanía, por hacer mi paso por la carrera más ameno y porque sin ella como compañera mis calificaciones seguro habrían sido peores.

A mis compañeros becarios del grupo DEI les agradezco su apoyo y el que me hayan aguantado tanto tiempo. En particular quiero agradecer a Álvaro Montero toda su ayuda técnica para la realización de este proyecto.

Por último quiero dedicar este trabajo a mi tutor Telmo, por todo su esfuerzo y por la paciencia que ha tenido conmigo.

RESUMEN

La tecnología de realidad aumentada es un tema de actualidad que despierta el interés tanto para desarrolladores como para usuarios, especialmente debido a la visibilidad que las gafas de realidad aumentada de Google en la prensa últimamente.

Por otro lado, el campo de la educación siempre ha perseguido nuevas formas de generar interés en el alumnado, pero los educadores se han encontrado barreras de insuficiencia de conocimiento tecnológico para la implantación de las nuevas tecnologías en sus clases. Esto ha tratado de solucionarse con la implementación de herramientas de autoría de videojuegos educativos, una de las cuales es GREP (GameRules scEnario Platform).

Este proyecto consiste en el desarrollo de una herramienta de creación de escenarios de realidad aumentada que no requiere conocimientos técnicos por parte del usuario.

Esta herramienta se complementa con un sistema de visualización que permite la percepción de los elementos virtuales del escenario durante su edición a través del uso de unas gafas de realidad aumentada VUZIX STAR 1200 y el emplazamiento de marcadores de posición en el escenario.

Por último, los escenarios generados se conectan con GREP de manera que puede alterar las propiedades de los elementos virtuales durante la ejecución de una partida, y esto puede observarse con el dispositivo de realidad aumentada.

Palabras clave: Realidad Aumentada, Serious Games, Tecnología para la educación

Índice

Índice de tablas

Índice de ilustraciones

1. Introducción

En este capítulo se presenta el contexto, y los objetivos planteados a la hora de realizar el proyecto.

1.1. Contexto

Con reciente auge en desarrollo de las tecnologías de realidad virtual y realidad aumentada cada vez se habla más de estas tecnologías y en especial de cómo aplicarlas a nuestra vida. En concreto, cabe destacar la expectación por que lleguen al público productos como Google Glass, dispositivo de realidad aumentada de Google y Occulus Rift, dispositivo de realidad virtual de Occulus VR.

Estas tecnologías despiertan el interés de la gente y abren un amplio abanico de posibilidades. Este interés y atractivo podría emplearse para que determinadas tareas, que por norma general no resultan atractivas, sean llevadas a cabo con menor reticencia. Entre estas tareas se encuentra el aprendizaje dentro del ámbito de la educación.

Una línea de investigación reciente consiste en la creación de juegos educativos para mantener el interés del alumno. Por desgracia, en muchas ocasiones el educador no dispone del conocimiento técnico necesario para la generación de juegos educativos. Esto motivó el desarrollo de la herramienta GREP (Game Rules scEnario Player), que permite la definición rápida y sencilla de juegos 3D.

La tecnología de realidad aumentada ofrece múltiples posibilidades de mejorar aún más este escenario, pero desafortunadamente el educador no suele disponer de los conocimientos técnicos que le permitan aprovechar las posibilidades que ofrece esta tecnología y le resulta demasiado costosa de aprender.

1.2. Objetivos

Una vez planteado el contexto y el problema a solucionar se definen los objetivos del proyecto.

1.2.1. Objetivo Principal

El objetivo principal del proyecto es desarrollar una herramienta de generación de escenarios de realidad aumentada que no requiera un conocimiento técnico elevado para su uso.

Estos escenarios de realidad aumentada serán utilizados para el diseño de videojuegos por el sistema de autoría de juegos GREP, de manera que se proporcione el soporte de la realidad

aumentada a este sistema para la generación de juegos principalmente enfocados para la educación.

La herramienta de generación de escenarios deberá ser utilizable desde el entorno físico que representa el escenario, permitiendo al editor moverse por el escenario mientras visualiza los elementos de realidad aumentada que va añadiendo a través de un dispositivo de realidad aumentada equipado sobre la cabeza del editor.

Una vez un escenario ha sido generado, se posibilita la integración de este con el sistema GREP, de manera que un juego GREP pueda manipular elementos de realidad aumentada durante la ejecución de una partida, permitiendo, entre otras acciones, el desplazamiento, la rotación y la aparición/desaparición de estos elementos.

Además, durante la ejecución de una de estas partidas se posibilita la visualización de los elementos de realidad aumentada a través del mismo dispositivo de realidad aumentada equipado sobre la cabeza.

En concreto, los objetivos del proyecto comprenden el desarrollo de tres sistemas interconectados:

1.- **Sistema de edición de escenarios virtuales**. Este sistema permitirá la definición de los escenarios con elementos de realidad aumentada y marcadores que permitan la futura visualización de estos elementos. El sistema debe ser accesible desde dispositivos portátiles que permitan al editor pasearse por el escenario durante la edición.

2.- **Sistema de visualización de escenarios y partidas**. Este sistema permitirá el uso de un dispositivo de realidad aumentada para la visualización de los elementos de realidad aumentada tanto durante la edición del escenario como durante la ejecución la partida que utilice ese escenario.

3.- **Sistema de conexión**. Se complementará el sistema GREP y se generará el servicio que permita la manipulación de los elementos de realidad aumentada por parte de este.

1.3. Estructura del Documento

El presente documento se divide en varios capítulos, los cuales desglosan el desarrollo del proyecto y son los siguientes:

- **Introducción.** Este capítulo comienza detallando el contexto en el que se enmarca este proyecto, describiendo el entorno en el que se inicia. Consecutivamente, una vez en situación, se analizan los problemas latentes que motivan la realización de este proyecto. Tras plantearse estos problemas, se exponen los objetivos propuestos para la subsanación de estos problemas. Para finalizar, se declara la estructura que rige el documento.

- **Estado del arte.** Este capítulo estudia el concepto de Realidad Aumentada, analiza una serie de aplicaciones y clasifica los dispositivos disponibles para uso. A continuación expone las herramientas disponibles para la creación de aplicaciones de realidad aumentada y seguidamente se analiza el sistema de autoría de videojuegos GREP, sistema con el que se integra el proyecto.

- **Análisis.** Este capítulo detalla la fase del proyecto en la que los requisitos de usuario son obtenidos del cliente mediante su colaboración. Adicionalmente se obtienen los casos de uso derivados de los mismos y con esta información recolectada se generan los requisitos de software, los cuales servirán para las siguientes fases del desarrollo. Finalmente se presentan las matrices de trazabilidad pertinentes para la verificación de la cobertura de las funcionalidades requeridas.

- **Diseño.** Este capítulo parte de los requisitos de software generados y detalla la arquitectura del software empleada, descomponiendo el proyecto en subsistemas y estos a su vez en componentes. Estos componentes son enlazados con los requisitos funcionales obtenidos a través de una matriz de trazabilidad que garantice la cobertura de los requisitos. Para finalizar se presentan prototipos de bajo fidelidad que sirven de guía para el diseño del sistema final.

- **Implementación.** Este capítulo se exponen detalles de bajo nivel de los subsistemas implementados: tecnologías concretas utilizadas y algoritmos más significativos. Además se mostrará el diseño final de las interfaces de usuario empleadas con sus correspondientes comentarios.

- **Implantación y Pruebas.** Este capítulo describe el despliegue de los distintos subsistemas del proyecto en los distintos dispositivos hardware empleados para la verificación del correcto funcionamiento del sistema implementado. Además se exponen algunos de los *tests* realizados para esta verificación.

- **Gestión del proyecto.** Este capítulo describe el modelo de ciclo de vida utilizado para el desarrollo del proyecto y presenta la planificación seguida, así como el análisis de los consecuentes costes de la realización de este proyecto.
- **Conclusiones.** Este capítulo expresa las conclusiones obtenidas tras la realización del proyecto, así como las líneas de trabajo futuro.
- **Referencias.** Este capítulo simplemente lista las fuentes bibliográficas consultadas para la elaboración del proyecto.

2. Estado del Arte

En este capítulo se analizará el concepto de Realidad Aumentada y se profundizará en él presentando sus aplicaciones actuales y los tipos de Realidad Aumentada clasificados por los tipos de dispositivo disponibles. Además, se estudiará la creación de aplicaciones de realidad aumentada y se expondrán las librerías existentes. Por último, se detallará el sistema de autoría de videojuegos GREP con el que se ha integrado este Trabajo Fin de Grado, el cual se enfoca a la creación sencilla de videojuegos para la educación desarrollados por educadores sin apenas conocimiento técnico.

2.1. ¿Qué es la Realidad Aumentada?

Realidad Aumentada es un término que define la percepción directa o indirecta de un entorno físico del mundo real que se complementa con elementos virtuales que añaden información o aportan un contexto determinado.

Cabe destacar que, aunque la información aportada por estos elementos virtuales no se restringe estrictamente al ámbito visual, las líneas de investigación y tecnología implementada se han centrado en este ámbito. Este Trabajo Fin de Grado analiza la Realidad Aumentada desde este punto de vista.

Dependiendo del grado de integración de los elementos virtuales en el entorno se presenta la siguiente línea de continuidad de la virtualidad [1].

ILUSTRACIÓN 1 - ESQUEMA DEL CONTINUO DE VIRTUALIDAD

Este diagrama define dos extremos. Por un lado existen los entornos puramente reales en los que la única percepción corresponde con elementos reales. En el otro extremo nos encontramos con entornos puramente virtuales, en los que todo elemento percibido es generado artificialmente. Este último contexto corresponde con el concepto de Realidad Virtual, en la cual

el usuario se sumerge completamente en el mundo virtual perdiendo la percepción de elementos del mundo real.

Entre estos dos extremos nos encontramos con un área en el que elementos virtuales y reales se presentan simultáneamente denominada la Realidad Mixta. Cuando integran elementos virtuales en un entorno real se habla de Realidad Aumentada mientras que si se integran elementos reales en un entorno principalmente virtual se habla de Virtualidad Aumentada.

Cabe destacar que esta clasificación es criticada por algunos autores, quienes opinan que la Realidad Aumentada debe considerarse un caso de Realidad Virtual.

2.1.1. Aplicaciones

A continuación se presentan algunos de los dominios en los que se ha explorado la aplicación de la Realidad Aumentada, analizados por Azuma en su encuesta [3] y su actualización [2].

- **Medicina.** Se podría utilizar entre otras para la 3D de órganos internos y para el entrenamiento de cirujanos.
- **Manufactura y reparación.** Se podría utilizar la realidad aumentada para aumentar los manuales de usuario que indican las posiciones de las piezas con representaciones 3D que guiarán el proceso de ensamblado.
- **Anotación y visualización.** La realidad aumentada podría utilizarse para anotación de información asociada a un objeto o lugar que fuera luego visualizable al volver a visitarlo.
- **Planificación de rutas para robots.** El manejo de un robot puede con llevar problemas en tiempo real como que se pierda la comunicación con este. A través de la realidad aumentada se puede manipular la versión virtual del robot hasta terminar de detallar el plan y una vez definido ejecutarlo cuando fuera necesario en el robot.
- **Aviación militar.** El uso de la superposición de gráficos sobre la vista de los pilotos ha sido utilizada desde hace años.
- **Móvil y *outdoor*.** En concreto se puede utilizar para el turismo, de manera que el turista, utilizando un dispositivo de realdad aumentada, visualice la información de los distintos emplazamientos.

- **Publicidad.** En eventos deportivos se ha utilizado ya la realidad aumentada para la superposición de anuncios en las transmisiones de video de estos encuentros.

2.1.2. Clasificación por tipos de dispositivos de representación

La Realidad Aumentada requiere tres fases: obtención de información del mundo real, procesamiento de la información obtenida y representación de la información virtual.

La fase de obtención de información requiere el uso de sensores u otros medios de adquisición de información sobre el mundo real. El tipo de sensor más común es el dispositivo de captura de vídeo, pero no se limitan a estos exclusivamente. En concreto se pueden utilizar acelerómetros, giroscopios, brújulas, posicionamiento global GPS, radiofrecuencia (RFID) o Near Field Communication (NFC) entre otros.

En concreto la tecnología utilizada en este Trabajo Fin de Grado es la de captura de vídeo, en parte motivada por el dispositivo de realidad aumentada disponible.

La fase de procesamiento de información recopila la información obtenida en la fase anterior y la procesa para el posicionamiento e identificación de objetos. La técnica más empleada pasa por el uso de visión artificial junto con marcadores identificables de posición (*fiducial markers*). Además esta fase se encarga de la generación la de visualización de elementos virtuales que se envía al dispositivo de representación.

La última fase, la representación de la información virtual, utiliza distintas técnicas para la presentación de la información al usuario. Dependiendo de la tecnología de estos dispositivos se clasifica la Realidad Aumentada en la siguiente taxonomía.

- ***Head Mounted Display*.** Se trata de un tipo de dispositivos que se sitúan en la cabeza del usuario y presentan la información en algún tipo de pantalla o cristal situado frente a sus ojos. Normalmente estos dispositivos incluyen una o dos cámaras que se intentan alinear con la visión real del usuario. Esta alineación nunca es perfecta, ya que no se puede posicionar una cámara en la misma posición que el ojo, pero se pueden utilizar aproximaciones como el posicionamiento de la cámara en la frente entre otras. Estas medidas introducen una desviación en cuanto a posición y rotación que puede ser compensada en muchos casos con calibraciones.

  Este tipo se subdivide en *Optical See-Through* y *Video See-Through*.

- ***Optical See-Through.*** Este tipo de dispositivos se basan en el uso de un cristal situado frente a los ojos que permite la proyección de elementos virtuales sobre este. Esto permite la visualización del mundo real directa con el añadido de los elementos virtuales superpuestos. En particular estos elementos virtuales, por motivos tecnológicos del cristal, presentan siempre algún grado de transparencia. Esto puede hacer que los elementos virtuales mostrados se vean distorsionados por la visión real de fondo en cuanto al color percibido.

  Ejemplos:

ILUSTRACIÓN 2 - GOOGLE GLASS Y VUZIX STAR 1200

A la izquierda se observa el dispositivo Google Glass y a la derecha y Vuzix Star 1200, el dispositivo utilizado para la realización de este proyecto.

Cabe destacar que el dispositivo de Google apenas cubre una pequeña porción del campo de visión del usuario mientras que el dispositivo de Vuzix presenta dos cristales sobre los que se proyectan los elementos virtuales con una mayor cobertura del campo de visión del usuario, permitiendo, entre otras, técnicas de visión estereoscópica en las que se proyecten los elementos virtuales desde un punto de vista ligeramente diferente en cada ojo, como compensación de la diferente perspectiva de cada ojo.

- **Video See-Through.** Este segundo tipo de *Head Mounted Display* se diferencia del anterior en que oculta totalmente la visión directa del mundo real interponiendo una pantalla que cubre completamente la visión del usuario. La percepción del mundo real en este caso es indirecta: al usuario se le muestra la imagen captada por una o dos cámaras como fondo y se superponen sobre esta

imagen los elementos virtuales. Esto por un lado evita el efecto de semitransparencia no deseada, pero se experimenta una percepción menos natural del mundo real. Además, esta visualización es acompañada de una mayor fatiga al mirar constantemente una pantalla a escasa distancia del ojo y puede incluso plantear problemas de seguridad física, ya que si el dispositivo por cualquier motivo deja de funcionar el usuario queda desprovisto de toda visión. Esto es especialmente peligroso al conducir.

Ejemplo:

ILUSTRACIÓN 3 - VUZIX WRAP 1200 AR

El dispositivo Vuzix Wrap 1200 AR es un ejemplo de *Video See-Through*. En este dispotivo se apreciar cómo la alineación de las cámaras con respecto a los ojos es mucho mejor ya que pueden situarse directamente frente al ojo.

- ***Handheld* o *Display* Móvil.** Este tipo de dispositivos no se montan frente al ojo como los anteriores, sino que requieren el uso de al menos una mano para sujetarlos. Su uso es también de modalidad *Video See-Through*, debido a que el dispositivo debe interponerse en la línea de visión entre el ojo del usuario y el objeto a aumentar virtualmente utilizando el vídeo capturado ofreciendo una visión indirecta. A pesar de tener los inconvenientes respecto de los *Head Mounted Displays* de ofrecer un campo de visión más reducido e inutilizar una o ambas manos, estos dispositivos ya están

extendidos y son más baratos. En particular se hace referencia a *smartphones* y *tablets*, que disponen de cámara trasera.

Ejemplo:

ILUSTRACIÓN 4 - HANDHELD SEE-THROUGH

En la ilustración se muestra el uso de una *Tablet* para la representación de elementos virtuales a partir del reconocimiento de un marcador.

- **Monitor de ordenador.** Este tipo de dispositivo presentado es un monitor corriente de ordenador, tanto integrado como en el caso de los portátiles como separado en el caso de los equipos de sobremesa. Este dispositivo debe complementarse con el uso de una cámara como puede ser una típica *webcam.* La utilización de este tipo de dispositivos

permite la una percepción de la realidad a modo de “espejo aumentador”. La cámara muestra al propio usuario mirando al monitor, pero superpone elementos virtuales.

Esta modalidad ha tomado fuerza desde el año 2009, cuando la librería FLARToolkit posibilitó el uso de la realidad aumentada directamente desde el navegador web.

Ejemplo:

ILUSTRACIÓN 5 - REAKUDAD AUMENTADA EN EL ORDENADOR

En la imagen se muestra un ejemplo de uso publicitario para la venta de relojes.

- ***Spatial* o *Display Espacial*.** Esta tecnología utiliza proyecciones de imagen sobre cualquier tipo de superficie, lo que permite la visualización virtual de los elementos por varias personas a la vez. Sin embargo, las proyecciones generalmente son de una nitidez

mejorable y las superficies irregulares sobre las que se proyecte pueden crear deformaciones no deseadas.

Ejemplo:

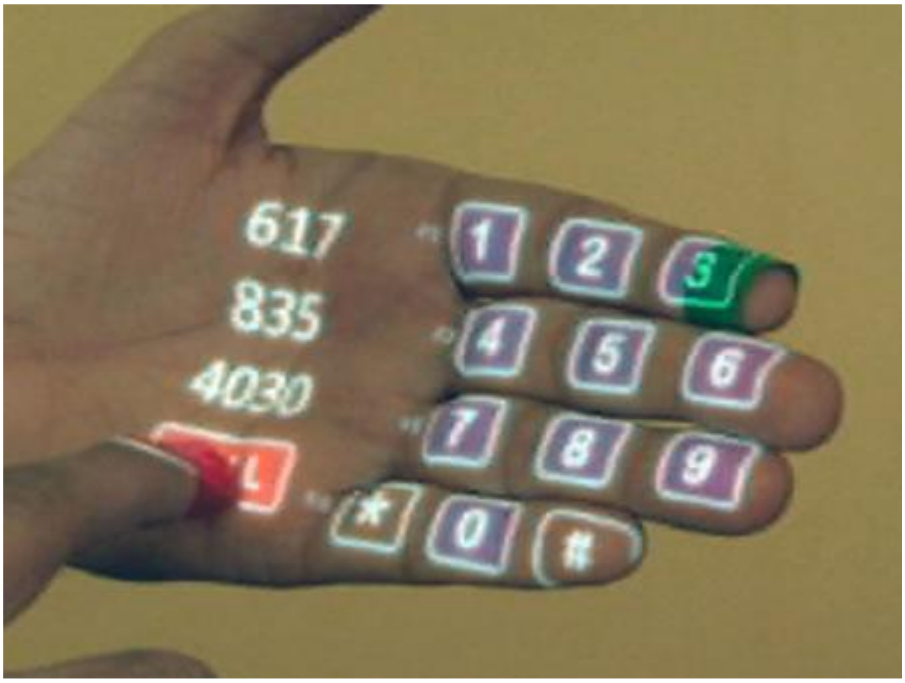


ILUSTRACIÓN 6 - CALCULADORA EN DISPLAY ESPACIAL

Esta ilustración muestra un ejemplo de display físico, la proyección de una calculadora sobre una mano.

Con esto concluye la clasificación de los tipos de Realidad Aumentada según el dispositivo de representación.

2.2. ¿Cómo se crean aplicaciones de Realidad Aumentada?

La creación de aplicaciones de realidad aumentada ha sido un tema de gran interés en los últimos años, pero, como se indica en [4], la investigación se ha centrado en el diseño del hardware y software necesario para el desarrollo de aplicaciones de realidad aumentada por parte de expertos en la materia. Para ello se han desarrollado multitud de librerías, las cuales se detallan a continuación.

2.2.1. Librerías

Desde la creación de la primera librería de código abierto de realidad aumentada en 1999 se han creado multitud de librerías para diversos entornos y lenguajes de programación.

A continuación se exponen las librerías que se han considerado más importantes.

- **OpenCV.** Técnicamente se trata de una librería de visión artificial desarrollada por Intel y cual primera versión fue publicada en 1999 bajo la licencia BSD. Esta licencia permite el uso de la librería con fines comerciales. Esta característica y el hecho de que esta librería tiene un extenso soporte de sistemas operativos (Linux, Mac OS, Windows y Android) han hecho que se extendiera su uso y en la actualidad su uso es omnipresente. En especial destacan las más de 500 funciones que ofrece, que permiten el reconocimiento no sólo de marcadores sino también de caras. La librería soporta los lenguajes de programación C, C++, Python, Java o Matlab.

  A partir de esta librería han surgido otras como EGmuCV, que permite el desarrollo en .Net y ArUco, que se integra con las librerías gráficas OpenGL y OGRE.

- **ARToolKit.** Esta es la primera librería de código abierto de Realidad Aumentada publicada en 1999 por el HIT Lab de la universidad de Washington. Actualmente se mantiene como proyecto de código abierto con licencia comercial. A partir de esta librería se generaron una gran cantidad de variantes, entre las que se destacan dos:

  - ATOMIC Authoring Tool. Software multi-plataforma para la creación de aplicaciones de realidad aumentada bajo la licencia GNU GPL.
  - ATOMIC Web Authoring Tool. Proyecto hijo de ATOMIC Authoring Tool, que permite la creación de aplicaciones de realidad aumentada para exportarlas a cualquier sitio web bajo la Licencia GNU GPL y no requiere apenas conocimiento técnico.

Por su alta compatibilidad y extensión y su licencia no restrictiva se ha optado por la utilización de OpenCV como librería para el reconocimiento de marcadores en el presente proyecto.

2.3. Autoría de juegos enfocados a la educación con GREP

El sistema GREP es una plataforma (GameRules scEnario Platform) que se basa en la definición de escenarios y reglas de juego a través de ficheros XML. Hace uso del motor Unity 3D y se ejecuta en sobre un navegador Web. En concreto, ofrece un modo de edición que permite la generación de los escenarios de juego a través de una interfaz gráfica en la que se pueden añadir entidades a las que se añaden comportamientos y eventos.

ILUSTRACIÓN 7 - EDICIÓN GREP

Tras el diseño del escenario se genera un fichero XML, el cual puede ser modificado para la inclusión de entidades de realidad aumentada o la identificación de entidades presentes como elementos de realidad aumentada.

El diseño detallado del sistema GREP se puede consultar en [5]

3. Análisis

Tras analizar el estado del arte, la siguiente etapa del proyecto es el análisis. En esta etapa se estudia con el cliente el alcance del proyecto a través de la especificación de requisitos de usuario y casos de uso. Para ello se han llevado a cabo una serie de entrevistas con el tutor, quien desempeña en este caso el papel de cliente, y se han acordado los requisitos a satisfacer por el sistema a desarrollar.

A partir de esta información se han elaborado los requisitos de software. Estos constituyen un desglose completo y detallado de los requisitos de usuario desde un punto de vista técnico.

Para asegurar que todo requisito de usuario y caso de uso es cubierto por los requisitos de software, y para la fácil trazabilidad de dependencias a la hora de modificar requisitos se han elaborado las correspondientes matrices de trazabilidad, presentadas al final del capítulo.

El conjunto de requisitos y casos de uso deben de ser confirmados por el cliente y constituyen un contrato escrito vinculante que define el alcance del sistema y delimita las responsabilidades del desarrollador. De esta manera toda capacidad o restricción el sistema no presente en este contrato no es exigible al desarrollador y debe ser renegociada entre ambas partes en caso de que el cliente desee incluirla.

3.1. Resumen de funcionalidad requerida

Para la fácil comprensión de los requisitos se presenta a continuación -en líneas generales- un resumen de la funcionalidad requerida por el cliente durante las diversas entrevistas realizadas.

El cliente desea un sistema de edición de escenarios virtuales que mantengan una correspondencia directa con juegos definidos por el sistema GREP, de manera que los elementos definidos en un escenario puedan ser manipulados desde las partidas de los juegos GREP.

El sistema de edición de escenarios permitirá al usuario definir un escenario utilizando un dispositivo portátil, por ejemplo de tipo *tablet*. El sistema ofrecerá una interfaz sencilla que permita añadir sobre un plano del entorno a aumentar elementos virtuales visualizables a través de un dispositivo de realidad aumentada.

El sistema ofrecerá principalmente dos modos de operación: modo “Edición” y modo “Partida”. En el modo “Edición” el usuario definirá estáticamente el escenario utilizando el dispositivo portátil añadiendo los elementos de realidad aumentada al plano del escenario. Estos elementos

virtuales se definirán a través de una posición sobre el mapa, una altura, una orientación y una escala. Simultáneamente el usuario tendrá la capacidad de visualizar los elementos que añade a través de un dispositivo de realidad aumentada. De esta manera se permite el ajuste y rectificación de los cambios en tiempo real hasta que el escenario quede perfectamente definido y se pase al modo "Partida". Para la detección de la posición y orientación del editor se dispondrán de marcadores reconocibles distribuidos por el escenario y captados por una cámara.

En el modo "Partida" se establecerá una correspondencia entre los elementos de juego definidos en GREP y los elementos virtuales definidos en un escenario. En este modo el usuario será capaz de visualizar en un dispositivo de realidad aumentada cómo los elementos virtuales se mueven, rotan, aparecen y desaparecen según las acciones que vayan teniendo lugar en la partida de GREP.

3.2. Especificación de requisitos de usuario

Mediante la definición de los requisitos de usuario el cliente detalla las funcionalidades demandadas al sistema. Existen dos tipos:

- **Requisitos de capacidad**. Definen las funcionalidades que el sistema debe proporcionar.
- **Requisitos de restricción**. Definen limitaciones sobre las funcionalidades (temporales, recursos, rendimiento, disponibilidad…).

Para la especificación de los requisitos se utilizarán tablas que incluyen los campos explicados a continuación:

- **Identificador**. Permite reconocer de forma unívoca cada requisito. El identificador seguirá la siguiente nomenclatura:

  RU<Tipo>-<Número>

  Donde:

  <Tipo> Tomará uno de los siguientes valores:

  - **C**: Indica requisito de capacidad.
  - **R**: Indica requisito de restricción.

<Número>. Número entero de dos cifras cuyo valor mínimo es 01 y que se irá incrementando en una unidad conforme se requieran más requisitos. Debido a la posible modificación y eliminación de requisitos no se garantiza el orden de este campo con respecto al mismo en los requisitos contiguos.

- **Nombre.** Descripción breve del requisito.
- **Descripción.** Explicación clara y concisa del objetivo del requisito. Además debe ser una descripción completa y consistente con el resto de requisitos, de manera que no exista posibilidad de contradicción entre requisitos.
- **Fuente.** Indicador de la persona que ha solicitado el requisito. Puede tomar los valores *Cliente* o *Analista*.
- **Necesidad.** Grado de importancia del requisito para el correcto funcionamiento del sistema. Puede tomar los valores *Opcional, Deseable* o *Esencial*.
- **Prioridad.** Grado de urgencia con el que el requisito debería cumplirse. Los valores posibles de este campo son: *Alta*, *Media* o *Baja*.
- **Estabilidad.** Indicador de la probabilidad con la que un requisito se espera que sea modificado o eliminado a lo largo del desarrollo del proyecto. Los valores posibles de este campo son: *Alta*, *Media* o *Baja*.
- **Verificabilidad.** Capacidad de comprobación del cumplimiento del requisito. Los valores posibles de este campo son: *Alta*, *Media* o *Baja*.
- **Claridad.** Grado de claridad del requisito. Indica si el requisito es entendible por sí mismo o si precisa explicaciones adicionales. Los valores posibles de este campo son: *Alta*, *Media* o *Baja*.

A continuación se muestra una plantilla de tabla de requisito de usuario.

RUX-YY			
Nombre			
Descripción			
Fuente		**Estabilidad**	
Necesidad		**Verificabilidad**	
Prioridad		**Claridad**	

RU-PLANTILLA

3.2.1. Requisitos de Capacidad

RUC-01			
Nombre	Gestión de modelos 3D		
Descripción	El sistema almacenará de forma persistente los modelos 3D y permitirá al usuario su gestión: • Alta de nuevos modelos 3D en el sistema. • Modificación de propiedades de un modelo 3D almacenado en el sistema. • Eliminación de un modelo 3D del sistema.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RUC- 01

RUC-02			
Nombre	Gestión de escenarios		
Descripción	El sistema almacenará de forma persistente la definición de los escenarios y permitirá al usuario su gestión: • Adición de escenarios al sistema. • Modificación de propiedades de un escenario almacenado en el sistema. • Eliminación de un escenario del sistema.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RUC- 02

RUC-03			
Nombre	Edición de escenarios		
Descripción	El sistema permitirá la edición de un escenario. Para ello posibilitará las siguientes funcionalidades: • Adición, eliminación, posicionamiento 3D, rotación y escalado de elementos virtuales a partir de modelos 3D. • Adición, eliminación, posicionamiento 3D, rotación y escalado de marcadores. La edición del escenario se efectuará de forma persistente sin necesidad de efectuar operaciones de guardado.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RUC- 03

RUC-04			
Nombre	Visualización de escenarios		
Descripción	El sistema permitirá la visualización de un escenario durante la edición. El usuario será capaz de percibir visualmente los cambios realizados sobre un escenario en tiempo real.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RUC- 04

RUC-05			
Nombre	Conexión con la plataforma GREP		
Descripción	El sistema ofrecerá la posibilidad de crear partidas a partir de escenarios definidos en el sistema, de manera que los elementos virtuales del escenario asociados a una determinada partida puedan ser accedidos y modificados desde la plataforma GREP implementada en Unity3D.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Media

RUC- 05

RUC-06			
Nombre	Visualización de partidas		
Descripción	El sistema permitirá la visualización de los elementos virtuales de las partidas durante su ejecución. El usuario será capaz de percibir visualmente los cambios realizados en la partida en tiempo real.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RUC- 06

3.2.2. Requisitos de Restricción

RUR-01			
Nombre	Sistema de edición de escenarios multiplataforma		
Descripción	El sistema de edición de escenarios debe poder ser utilizado mediante *tablets*, portátiles y ordenadores de sobremesa.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RUR- 01

RUR-02			
Nombre	Realidad aumentada		
Descripción	El sistema de visualización hará uso de un dispositivo de realidad aumentada que ofrezca la posibilidad de visualizar elementos virtuales superpuestos a la visión real.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RUR- 02

RUR-03			
Nombre	Portátil		
Descripción	Tanto el sistema de visualización como el sistema de edición deberían ser usables mediante un dispositivo portátil inalámbrico con autonomía de suministro eléctrico.		
Fuente	Cliente	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Alta

RUR- 03

RUR-04			
Nombre	Sistema operativo del sistema de visualización		
Descripción	El sistema operativo en el que se ejecute el sistema de visualización debe ser compatible con el dispositivo de realidad aumentada usado.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RUR- 04

3.3. Especificación de casos de uso

Los casos de uso representan la interacción entre los distintos actores y el sistema a través de escenarios. Esta técnica facilita la obtención de un conjunto de requisitos completo.

Para su representación se utiliza tanto una representación gráfica en forma de diagrama de caso de uso como una representación textual, que detalla las interacciones así como las condiciones requeridas tanto previas como posteriores al caso de uso.

A continuación se muestra el diagrama de casos de uso:

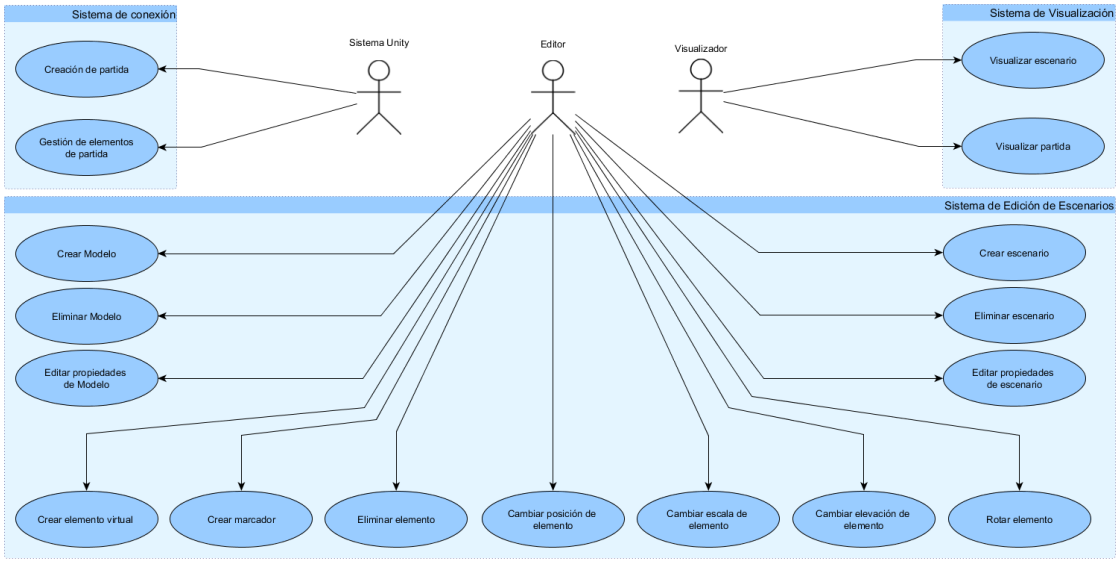


ILUSTRACIÓN 8 – DIAGRAMA DE CASOS DE USO

La representación textual de cada caso de uso se llevará a cabo por medio de tablas con los siguientes campos:

- **Identificador**. Permite reconocer de forma unívoca cada caso de uso. El identificador seguirá la siguiente nomenclatura:

  CU-<Número>

  Donde

  <Número> es un número entero de dos cifras cuyo valor mínimo es 01 y que se irá incrementando en una unidad conforme se elaboren más casos de uso. Debido a la posible modificación y eliminación de casos de uso no se garantiza el orden de este campo con respecto al mismo en los casos de uso contiguos.

- **Nombre.** Descripción breve del caso de uso.
- **Actor.** Todo agente externo al sistema que participa en el caso de uso.
- **Precondiciones.** Condiciones necesarias previas a la iniciación del caso de uso.
- **Poscodiciones.** Condiciones necesarias al finalizar el caso de uso.
- **Flujo normal.** Sucesión de interacciones entre actor y el sistema que culminan con el desarrollo exitoso del caso de uso.
- **Flujo alternativo.** Sucesión de interacciones entre los actores y el sistema que resulta en un flujo distinto al normal y que representa una alternativa a tratar.

CU-00	
Nombre	
Actor	
Descripción	
Precondiciones	
Flujo normal	
Flujo alternativo	
Poscondiciones	

CU-PLANTILLA

CU-01	
Nombre	Crear modelo
Actor	Editor
Descripción	El usuario añade un modelo 3D al sistema especificando nombre, fichero de icono, fichero de modelo, escala inicial y opcionalmente fichero de textura.
Precondiciones	El sistema muestra la página de gestión de modelos y se muestra la lista de modelos actuales en el sistema.
Flujo normal	1. El usuario selecciona la opción de añadir modelo. 2. El sistema presenta un formulario con los siguientes campos a rellenar: - Nombre (obligatorio) - Fichero de icono (obligatorio) - Fichero de modelo (obligatorio) - Escala inicial (obligatorio) - Fichero de textura (opcional) 3. El usuario rellena el formulario y selecciona la opción de guardar. 4. El sistema pide confirmación al usuario. 5. El usuario confirma la creación del modelo. 6. El sistema comprueba que los campos requeridos han sido cumplimentados y que los archivos tienen el formato adecuado. 7. El sistema accede a la base de datos y añade el nuevo modelo. 8. El sistema oculta el formulario y muestra la lista actual de modelos, incluyendo el añadido.
Flujo alternativo	3.a. El usuario selecciona la opción de cancelar. 3.a.1. El sistema oculta el formulario sin añadir el modelo al sistema.
Flujo alternativo	3.b. El usuario no rellena todos los campos obligatorios y selecciona la opción de guardar. 3.b.1. El sistema muestra un mensaje de error especificando los campos obligatorios no cumplimentados y mantiene el formulario abierto sin añadir el modelo al sistema. 3.b.2. Se prosigue con el flujo normal en el punto 3.
Flujo alternativo	3.c. El usuario rellena todos los campos obligatorios seleccionando archivos con formato inválido y selecciona la opción de guardar. 3.c.1. El sistema muestra un mensaje de error especificando los formatos de ficheros requeridos y mantiene el formulario abierto sin añadir el modelo al sistema. 3.c.2. Se prosigue con el flujo normal en el punto 3.

Flujo alternativo	5.a. El usuario rechaza la confirmación. 5.a.1. Se prosigue con el flujo normal en el punto 3.
Poscondiciones	El sistema muestra la página de gestión de modelos y se muestra la lista de modelos actuales en el sistema.

CU- 01

CU-02	
Nombre	Eliminar modelo
Actor	Editor
Descripción	El usuario elimina un modelo 3D del sistema, eliminándose con él todos los elementos virtuales que utilizan el modelo 3D en todos los escenarios previa confirmación del usuario.
Precondiciones	El sistema muestra la página de gestión de modelos y se muestra la lista de modelos actuales en el sistema habiendo al menos un modelo presente.
Flujo normal	1. El usuario selecciona la opción de eliminar del modelo. 2. El sistema informa al usuario de la eliminación de todos los elementos virtuales asociados al modelo al eliminar el modelo y pide la confirmación al usuario. 3. El usuario confirma la eliminación del modelo. 4. El sistema accede a la base de datos y elimina el modelo junto con todos los elementos virtuales asociados. 5. El sistema muestra la lista actualizada de modelos.
Flujo alternativo	3.a. El usuario rechaza la confirmación. 3.a.1. El sistema oculta la petición de confirmación sin eliminar el modelo.
Poscondiciones	El sistema muestra la página de gestión de modelos y se muestra la lista de modelos actuales en el sistema.

CU- 02

CU-03	
Nombre	Editar propiedades de modelo
Actor	Editor
Descripción	El usuario modifica un modelo 3D del sistema cambiando nombre, fichero de icono, fichero de modelo, escala inicial y/o fichero de textura.
Precondiciones	El sistema muestra la página de gestión de modelos y se muestra la lista de modelos actuales en el sistema habiendo al menos un modelo presente.
Flujo normal	1. El usuario selecciona la opción de editar modelo. 2. El sistema presenta un formulario con los siguientes campos rellenos con los valores actuales del modelo: - Nombre - Fichero de icono - Fichero de modelo - Escala inicial - Fichero de textura 3. El usuario modifica los campos que desea cambiar y selecciona la opción de guardar. 4. El sistema comprueba que los archivos tienen el formato adecuado si estos han sido cambiados. 5. El sistema accede a la base de datos y actualiza las propiedades del modelo con los nuevos valores. 6. El sistema oculta el formulario y muestra la lista actual de modelos.
Flujo alternativo	3.a. El usuario selecciona la opción del cancelar. 3.a.1. El sistema oculta el formulario sin aplicar ningún cambio al modelo.
Flujo alternativo	3.b. El usuario modifica algún archivo seleccionando un fichero con formato inválido y selecciona la opción de guardar. 3.b.1. El sistema muestra un mensaje de error especificando los formatos de ficheros requeridos y mantiene el formulario abierto sin actualizar las propiedades del escenario. 3.b.2. Se prosigue con el flujo normal en el punto 3.
Poscondiciones	El sistema muestra la página de gestión de modelos y se muestra la lista de modelos actuales en el sistema.

CU- 03

CU-04	
Nombre	Crear escenario
Actor	Editor
Descripción	El usuario crea un nuevo escenario en el sistema especificando nombre, fichero de plano 2D, y medidas del escenario.
Precondiciones	El sistema muestra la página de gestión de escenarios y se muestra la lista de escenarios actuales en el sistema.
Flujo normal	1. El usuario selecciona la opción de añadir escenario. 2. El sistema presenta un formulario con los siguientes campos a rellenar: - Nombre (obligatorio) - Fichero de plano 2D (obligatorio) - Medida ancho (obligatorio) - Medida largo (obligatorio) - Medida alto (obligatorio) 3. El usuario rellena el formulario y selecciona la opción de guardar. 4. El sistema pide confirmación al usuario. 5. El usuario confirma la creación del escenario. 6. El sistema comprueba que los campos requeridos han sido cumplimentados y que los archivos tienen el formato adecuado. 7. El sistema accede a la base de datos y añade el nuevo escenario. 8. El sistema oculta el formulario y muestra la lista actual de escenarios, incluyendo el añadido.
Flujo alternativo	3.a. El usuario selecciona la opción del cancelar. 3.a.1. El sistema oculta el formulario sin añadir el escenario al sistema.
Flujo alternativo	3.b. El usuario no rellena todos los campos obligatorios y selecciona la opción de guardar. 3.b.1. El sistema muestra un mensaje de error especificando los campos obligatorios no cumplimentados y mantiene el formulario abierto sin añadir el modelo al sistema. 3.b.2. Se prosigue con el flujo normal en el punto 3.
Flujo alternativo	5.a. El usuario rechaza la confirmación. 5.a.1. Se prosigue con el flujo normal en el punto 3.
Poscondiciones	El sistema muestra la página de gestión de escenarios y se muestra la lista de escenarios actuales en el sistema.

CU- 04

CU-05	
Nombre	Eliminar escenario
Actor	Editor
Descripción	El usuario elimina un escenario del sistema, eliminándose con él todos los elementos virtuales así como los marcadores presentes en el escenario.
Precondiciones	El sistema muestra la página de gestión de escenarios y se muestra la lista de escenarios actuales en el sistema habiendo al menos un escenario presente.
Flujo normal	1. El usuario selecciona la opción de eliminar de un escenario. 2. El sistema informa al usuario de la eliminación de todos los elementos virtuales y marcadores asociados al escenario y pide la confirmación al usuario. 3. El usuario confirma la eliminación del escenario. 4. El sistema accede a la base de datos y elimina el escenario junto con todos los elementos virtuales y marcadores asociados. 5. El sistema muestra la lista actualizada de escenarios.
Flujo alternativo	3.a. El usuario rechaza la confirmación. 3.a.1. El sistema oculta la petición de confirmación sin eliminar el escenario.
Poscondiciones	El sistema muestra la página de gestión de escenarios y se muestra la lista de escenarios actuales en el sistema.

CU- 05

CU-06	
Nombre	Editar propiedades de escenario
Actor	Editor
Descripción	El usuario modifica un escenario del sistema cambiando nombre, fichero de plano, medida de ancho, medida de largo y/o medida de alto.
Precondiciones	El sistema muestra la página de gestión de escenarios y se muestra la lista de escenarios actuales en el sistema habiendo al menos un escenario presente.
Flujo normal	1. El usuario selecciona la opción de editar escenario. 2. El sistema presenta un formulario con los siguientes campos rellenos con los valores actuales del escenario: - Nombre - Fichero de plano - Medida de ancho - Medida de largo - Medida de alto 3. El usuario modifica los campos que desea cambiar y selecciona la opción de guardar. 4. El sistema comprueba que tienen el formato adecuado si estos han sido cambiados. 5. El sistema accede a la base de datos y actualiza las propiedades del escenario con los nuevos valores. 6. El sistema oculta el formulario y muestra la lista actual de escenarios.
Flujo alternativo	3.a. El usuario selecciona la opción del cancelar. 3.a.1. El sistema oculta el formulario sin aplicar ningún cambio al escenario.
Flujo alternativo	3.b. El usuario modifica el valor del fichero de plano con formato inválido y selecciona la opción de guardar. 3.b.1. El sistema muestra un mensaje de error especificando los formatos de ficheros requeridos y mantiene el formulario abierto sin actualizar las propiedades del escenario. 3.b.2. Se prosigue con el flujo normal en el punto 3.
Poscondiciones	El sistema muestra la página de gestión de escenarios y se muestra la lista de escenarios actuales en el sistema.

CU- 06

CU-07	
Nombre	Crear elemento virtual
Actor	Editor
Descripción	El usuario añade un elemento virtual a un escenario seleccionando el modelo a usar y la escala.
Precondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.
Flujo normal	1. El usuario selecciona la opción de añadir elemento virtual. 2. El sistema presenta un formulario con una lista de los modelos disponibles y un campo con la escala. 3. El usuario selecciona uno de los modelos de la lista y una escala y selecciona la opción de añadir. 4. El sistema accede a la base de datos y añade el nuevo elemento virtual. 5. El sistema añade una representación visual del nuevo elemento virtual sobre el plano del escenario. 6. El sistema oculta el formulario dejando ver el plano del escenario.
Flujo alternativo	3.a. El usuario selecciona la opción del cancelar. 3.a.1. El sistema oculta el formulario sin añadir el elemento virtual al escenario.
Poscondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.

CU- 07

CU-08	
Nombre	Crear marcador
Actor	Editor
Descripción	El usuario añade un marcador a un escenario seleccionando el número identificador del marcador (número codificado de la cara principal) y el tipo de marcador.
Precondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.
Flujo normal	1. El usuario selecciona la opción de añadir marcador. 2. El sistema presenta un formulario con un campo de identificación de marcador (número codificado de la cara principal) y una lista de tipos de marcadores (planos o cúbicos). 3. El usuario selecciona un número de identificación de marcador y un tipo de marcador y selecciona la opción de añadir. 4. El sistema accede a la base de datos y añade el nuevo marcador. 5. El sistema añade una representación visual del nuevo marcador sobre el plano del escenario. 6. El sistema oculta el formulario dejando ver el plano del escenario.
Flujo alternativo	3.a. El usuario selecciona la opción del cancelar. 3.a.1. El sistema oculta el formulario sin añadir el marcador al escenario.
Poscondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.

CU- 08

CU-09	
Nombre	Eliminar elemento
Actor	Editor
Descripción	El usuario selecciona uno o varios elementos del escenario (marcadores y/o elementos virtuales) y los elimina.
Precondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.
Flujo normal	1. El usuario selecciona los elementos del escenario a eliminar. 2. El usuario selecciona la opción de eliminación. 3. El sistema accede a la base de datos y elimina los elementos seleccionados. 4. El sistema elimina las representaciones visuales de los elementos eliminados.
Poscondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario. Los elementos eliminados no están presentes.

CU- 09

CU-10	
Nombre	Cambiar posición de elemento
Actor	Editor
Descripción	El usuario desplaza marcadores y/o elementos virtuales en un plano horizontal (altura constante).
Precondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.
Flujo normal	1. El usuario selecciona los elementos del escenario a desplazar. 2. El usuario desplaza los elementos seleccionados arrastrando (*drag*) un elemento hasta la posición final deseada manteniéndose las distancias entre todos los elementos seleccionados. 3. El usuario finaliza el arrastre (*drop*). 4. El sistema actualiza las posiciones de los elementos seleccionados en la base de datos.
Poscondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario. Los elementos desplazados se encuentran en las nuevas posiciones.

CU- 10

CU-11	
Nombre	Cambiar escala de elemento
Actor	Editor
Descripción	El usuario selecciona marcadores y/o elementos virtuales y modifica su escala.
Precondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.
Flujo normal	1. El usuario selecciona los elementos del escenario a desplazar. 2. El usuario regula la escala del último elemento seleccionado mientras el sistema regula la escala de los demás elementos seleccionados asignando la misma escala que el elemento regulado por el usuario. 3. El usuario finaliza ajuste de la escala. 4. El sistema actualiza las escalas de los elementos seleccionados en la base de datos.
Poscondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario. Los elementos seleccionados muestran la misma escala.

CU- 11

CU-12	
Nombre	Cambiar elevación de elemento
Actor	Editor
Descripción	El usuario modifica la altura de uno o varios elementos del escenario.
Precondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.
Flujo normal	1. El usuario selecciona los elementos del escenario a desplazar en vertical (modificar altura). 2. El usuario regula la altura del último elemento seleccionado mientras el sistema regula la altura de los demás elementos seleccionados asignando la misma altura que el elemento regulado por el usuario. El sistema muestra visualmente este cambio de altura a través de un gradiente de color. 3. El usuario finaliza ajuste de la altura. 4. El sistema actualiza las alturas de los elementos seleccionados en la base de datos.
Poscondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario. Los elementos seleccionados muestran la misma altura.

CU- 12

CU-13	
Nombre	Rotar elemento
Actor	Editor
Descripción	El usuario modifica la orientación de uno o varios elementos del escenario seleccionándolos y rotándolos respecto a un eje paralelo a la altura del escenario.
Precondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario.
Flujo normal	1. El usuario selecciona los elementos del escenario a rotar. 2. El usuario regula la orientación del último elemento seleccionado mientras el sistema regula la orientación de los demás elementos seleccionados asignando la misma rotación que el elemento regulado por el usuario. El sistema muestra visualmente este cambio de orientación. 3. El usuario finaliza ajuste de la rotación. 4. El sistema actualiza las rotaciones de los elementos seleccionados en la base de datos.
Poscondiciones	El sistema muestra la página de edición de escenario donde se muestran los elementos virtuales y marcadores presentes en el escenario. Los elementos seleccionados muestran la misma orientación respecto al eje vertical del escenario.

CU- 13

CU-14	
Nombre	Visualizar escenario
Actor	Visualizador
Descripción	El usuario selecciona el escenario a visualizar y el sistema muestra los elementos virtuales asociados al escenario en el dispositivo de realidad aumentada teniendo en cuenta la posición y orientación del usuario. La posición del usuario es determinada a través de la presencia de marcadores en el campo de visión del usuario.
Precondiciones	El usuario utiliza el dispositivo de realidad aumentada con entrada de vídeo. El entorno físico del usuario corresponde con el entorno real a aumentar virtualmente. Existen marcadores reales en el entorno del usuario.
Flujo normal	1. El usuario selecciona un escenario a visualizar. 2. El sistema obtiene y almacena los ficheros de modelos necesarios para la visualización en el sistema de ficheros local. 3. El sistema obtiene la información de los elementos virtuales y marcadores presentes en el escenario. 4. El sistema genera un entorno virtual en el que posiciona y orienta los elementos virtuales y marcadores a partir de la información obtenida previamente. 5. El sistema obtiene una imagen de la cámara y la procesa en busca de marcadores. 6. El sistema encuentra un marcador y calcula la posición y orientación relativa del usuario al marcador, siendo la posición y orientación del marcador en el escenario conocida. 7. El sistema calcula la posición de la cámara virtual que simula la visión del usuario en el mundo virtual. 8. El sistema muestra al usuario el mundo virtual desde la posición de la cámara virtual. 9. Se vuelve al punto 3 (bucle hasta que el usuario decida finalizar la visualización)
Poscondiciones	

CU- 14

CU-15	
Nombre	Visualizar partida
Actor	Visualizador
Descripción	El usuario (GREP) selecciona la partida a visualizar y el sistema muestra los elementos de juego virtuales asociados a la partida en el dispositivo de realidad aumentada teniendo en cuenta la posición y orientación del usuario. La posición del usuario es determinada a través de la presencia de marcadores en el campo de visión del usuario.
Precondiciones	El usuario utiliza el dispositivo de realidad aumentada con entrada de vídeo. El entorno físico del usuario corresponde con el entorno real a aumentar virtualmente. Existen marcadores reales en el entorno del usuario.
Flujo normal	1. El usuario selecciona una partida a visualizar. 2. El sistema obtiene y almacena los ficheros de modelos necesarios para la visualización en el sistema de ficheros local. 3. El sistema obtiene la información de los elementos de juego de la partida y la información de los marcadores del escenario en el que se basa el juego. 4. El sistema genera un entorno virtual en el que posiciona y orienta los elementos de juego y marcadores a partir de la información obtenida previamente. 5. El sistema obtiene una imagen de la cámara y la procesa en busca de marcadores. 6. El sistema encuentra un marcador y calcula la posición y orientación relativa del usuario al marcador, siendo la posición y orientación del marcador en el escenario conocida. 7. El sistema calcula la posición de la cámara virtual que simula la visión del usuario en el mundo virtual. 8. El sistema muestra al usuario el mundo virtual desde la posición de la cámara virtual. 9. Se vuelve al punto 3 (bucle hasta que el usuario decida finalizar la visualización)
Poscondiciones	

CU- 15

CU-16	
Nombre	Creación de partida
Actor	Unity (GREP)
Descripción	El usuario solicita la creación de una partida de un juego de realidad mixta asociado a un escenario. El sistema genera la partida creando los elementos de juego en las posiciones definidas por el escenario a usar.
Precondiciones	Existe un juego de realidad mixta definido que utiliza un escenario del sistema.
Flujo normal	1. El usuario indica el juego del cual se quiere crear una partida. 2. El sistema genera los elementos de juego de la partida a partir de los elementos virtuales definidos en el escenario asociado al juego. Los elementos de juego son inicializados con las posiciones, orientaciones y escalas de los elementos virtuales definidos en el escenario.
Poscondiciones	Los elementos de la partida se encuentran en su posición y estado inicial, coincidentes con las posiciones definidas por el escenario

CU- 16

CU-17	
Nombre	Gestión de elementos de partida
Actor	GREP (Unity)
Descripción	El usuario solicita la actualización de la posición, orientación y/o estado de visualización de un elemento de partida y el sistema actualiza los valores.
Precondiciones	El elemento de partida a actualizar debe existir.
Flujo normal	1. El usuario identifica el elemento de juego a actualizar y proporciona una lista de pares atributo-valor con los nuevos valores. 2. El sistema comprueba la existencia de los atributos y la validez de los valores. 3. El sistema efectúa los cambios y responde con un mensaje de éxito.
Flujo alternativo	2.a. El sistema detecta un error en los atributos o en la validez de los valores suministrados. 2.a.1.El sistema responde con un mensaje de error sin efectuar los cambios.
Poscondiciones	Los atributos especificados por el usuario del elemento de partida han sido modificados.

CU- 17

3.4. Especificación de requisitos de software

Una vez analizados los requisitos de usuario se elaboran los requisitos de software. Estos surgen de la revisión de los requisitos de usuario desde un punto de vista técnico a bajo nivel. Como consecuencia surgen dos tipos de requisitos de software:

- **Requisitos funcionales**. Estos requisitos describen el funcionamiento del sistema definiendo las funcionalidades que debe desempeñar.
- **Requisitos no funcionales**. Estos requisitos imponen restricciones de diseño o implementación al sistema. A su vez se clasifican en los siguientes grupos:
    - **De rendimiento**. Definen cuantitativamente los rangos asociados a variables de rendimiento que el sistema debe respetar, como velocidades mínimas de transferencia de datos o tiempos de respuesta.
    - **De interfaz**. Detallan restricciones y protocolos de comunicación del sistema con otros sistemas o componentes.
    - **De portabilidad**. Definen requisitos que restringen la posibilidad de cambiar el entorno en el que se despliega el software.
    - **De usabilidad**. Detallan la facilidad de uso del sistema por parte del usuario.
    - **De seguridad**. Especifican la gestión de amenazas contra la confidencialidad, integridad y disponibilidad del sistema.
    - **De interfaz**. Detalla el software o hardware con el que el sistema debe interactuar
    - **De operación**. Definen cómo se ejecutará el sistema y cómo interactuará con el usuario a través de dispositivos de interfaz de usuario.
    - **De recursos**. Especifican límites superiores de recursos físicos como capacidad de procesamiento, memoria principal o espacio en disco duro.

Los requisitos de software utilizarán la misma representación textual que los requisitos de usuario previamente explicados a excepción del campo *identificador*, que utilizará la siguiente nomenclatura:

RS<Tipo>-<Número>

Donde:

<Tipo> Tomará uno de los siguientes valores:

F: Indica requisito de software funcional.

NF: Indica requisito de software no funcional.

<Número> es un número entero de dos cifras cuyo valor mínimo es 01 y que se irá incrementando en una unidad conforme se elaboren más requisitos. Debido a la posible modificación y eliminación de requisitos no se garantiza el orden de este campo con respecto al mismo en los requisitos contiguos.

La plantilla de la tabla que se utilizará para los requisitos de software será la siguiente:

RSX-YY			
Nombre			
Descripción			
Fuente		**Estabilidad**	
Necesidad		**Verificabilidad**	
Prioridad		**Claridad**	

RS-PLANTILLA

3.4.1. Requisitos de software funcionales

RSF-01			
Nombre	Registro de modelos 3D		
Descripción	El usuario podrá registrar de forma persistente modelos 3D aportando la siguiente información: - Nombre (obligatorio) - Fichero de icono (obligatorio) - Fichero de modelo (obligatorio) - Escala inicial (obligatorio) - Fichero de textura (opcional)		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 01

RSF-02			
Nombre	Modificación de modelos 3D		
Descripción	El usuario podrá modificar los modelos 3D registrados en el sistema. Los campos que el usuario podrá modificar serán: - Nombre - Fichero de icono - Fichero de modelo - Escala inicial - Fichero de textura		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 02

RSF-03			
Nombre	Eliminación de modelos 3D		
Descripción	El usuario podrá eliminar modelos 3D almacenados en el sistema.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 03

RSF-04			
Nombre	Registro de escenarios		
Descripción	El usuario podrá registrar de forma persistente escenarios aportando la siguiente información: - Nombre (obligatorio) - Fichero de plano 2D (obligatorio) - Medida ancho (obligatorio) - Medida largo (obligatorio) - Medida alto (obligatorio)		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 04

RSF-05			
Nombre	Modificación de escenarios		
Descripción	El usuario podrá modificar los escenarios 3D registrados en el sistema. Los campos que el usuario podrá modificar serán: - Nombre - Fichero de plano 2D - Medida ancho - Medida largo - Medida alto		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 05

RSF-06			
Nombre	Eliminación de escenarios		
Descripción	El usuario podrá eliminar escenarios almacenados en el sistema.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 06

RSF-07			
Nombre	Presencia de marcadores		
Descripción	El sistema tendrá la capacidad de detectar la presencia de marcadores a partir de una imagen capturada.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 07

RSF-08			
Nombre	Identificación de marcadores		
Descripción	El sistema tendrá la capacidad de identificar códigos de marcador presentes en una imagen capturada.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 08

RSF-09			
Nombre	Procesamiento de posición de marcadores		
Descripción	El sistema tendrá la capacidad de calcular la posición 3D desde la cámara y la orientación 3D del marcador dada una imagen capturada en la que hay presente un marcador.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Media

RSF- 09

RSF-10			
Nombre	Tipos de marcadores		
Descripción	El sistema tendrá la capacidad de gestionar los siguientes tipos de marcador: - Marcador plano: formado por un único código en una cara. - Marcador cúbico: formado por seis marcadores planos dispuestos en una estructura cúbica. Se define la cara superior como la principal.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Media

RSF- 10

RSF-11			
Nombre	Registro de marcadores		
Descripción	El usuario podrá registrar de forma persistente marcadores asociados a un escenario aportando la siguiente información: - Código de identificación de la cara principal (obligatorio) - Tipo de marcador (obligatorio)		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 11

RSF-12			
Nombre	Eliminación de marcadores		
Descripción	El usuario podrá eliminar los marcadores asociados a un escenario.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 12

RSF-13			
Nombre	Posicionamiento horizontal de marcadores		
Descripción	El sistema permitirá el desplazamiento de marcadores asociados a un escenario sobre el plano horizontal.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 13

RSF-14			
Nombre	Posicionamiento vertical de marcadores		
Descripción	El sistema permitirá el desplazamiento de marcadores asociados a un escenario sobre el eje vertical.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 14

RSF-15			
Nombre	Orientación de marcadores		
Descripción	El sistema permitirá la rotación de marcadores asociados a un escenario respecto del eje vertical.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Media

RSF- 15

RSF-16			
Nombre	Escalado de marcadores		
Descripción	El sistema permitirá el escalado global de marcadores asociados a un escenario. El escalado global afecta a las tres dimensiones simultáneamente.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Media

RSF- 16

RSF-17			
Nombre	Registro de elementos virtuales		
Descripción	El usuario podrá registrar de forma persistente elementos virtuales asociados a un escenario aportando la siguiente información: - Modelo (obligatorio) - Escala (obligatorio)		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 17

RSF-18			
Nombre	Eliminación de elementos virtuales		
Descripción	El usuario podrá eliminar los elementos virtuales asociados a un escenario.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 18

RSF-19			
Nombre	Posicionamiento horizontal de elementos virtuales		
Descripción	El sistema permitirá el desplazamiento de elementos virtuales asociados a un escenario sobre el plano horizontal.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 19

RSF-20			
Nombre	Posicionamiento vertical de elementos virtuales		
Descripción	El sistema permitirá el desplazamiento de elementos virtuales asociados a un escenario sobre el eje vertical.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 20

RSF-21			
Nombre	Orientación de elementos virtuales		
Descripción	El sistema permitirá la rotación de elementos virtuales asociados a un escenario respecto del eje vertical.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 21

RSF-22			
Nombre	Escalado de elementos virtuales		
Descripción	El sistema permitirá el escalado global de elementos virtuales asociados a un escenario. El escalado global afecta a las tres dimensiones simultáneamente.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Media

RSF- 22

RSF-23			
Nombre	Registro de juegos mixtos		
Descripción	El sistema almacenará persistentemente juegos mixtos. Cada juego mixto tiene asociado un escenario y consta de elementos de juego.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Media

RSF- 23

RSF-24			
Nombre	Registro de partidas		
Descripción	El sistema almacenará persistentemente las partidas asociadas a juegos mixtos.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Media

RSF- 24

RSF-25			
Nombre	Registro de elementos de partida		
Descripción	El sistema almacenará persistentemente los elementos de partida asociados a una partida registrándose los siguientes atributos: - posición 3D. - rotación sobre eje vertical. - estado de activación. - escala global. - elemento de juego correspondiente.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Media

RSF- 25

RSF-26			
Nombre	Instanciación de juego mixto		
Descripción	El sistema tendrá la capacidad de instanciar una partida de un juego generando elementos de partida a partir de los elementos virtuales del escenario asociado al juego.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Alta

RSF- 26

RSF-27			
Nombre	Visualización de elementos virtuales de un escenario		
Descripción	El sistema tendrá la capacidad de representar gráficamente elementos virtuales sobre un dispositivo de realidad aumentada.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 27

RSF-28			
Nombre	Visualización de elementos de partida		
Descripción	El sistema tendrá la capacidad de representar gráficamente elementos de partida sobre un dispositivo de realidad aumentada.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 28

RSF-29			
Nombre	Gestión de elementos de partida		
Descripción	El sistema ofrecerá la posibilidad de modificar atributos de los elementos de partida a través de una lista de pares atributo-valor en el que los atributos posibles son: - Posición x - Posición y - Posición z - Rotación z - Escala - Estado de activación		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSF- 29

3.4.2. Requisitos de software no funcionales

3.4.2.1. Requisitos de rendimiento

RSNF-01			
Nombre	Resolución captura de imágenes		
Descripción	El sistema utilizará un dispositivo de captura de imágenes con una resolución mínima de 640x480 píxeles.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 01

RSNF-02			
Nombre	Retardo en detección de cambios.		
Descripción	El sistema de visualización mostrará los cambios con un retardo máximo de 1 segundo.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 02

3.4.2.2. Requisitos de portabilidad

RSNF-03			
Nombre	Navegadores		
Descripción	El sistema de edición de escenarios será compatible con los siguientes navegadores: Escritorio: - Mozilla Firefox - Google Chrome - Opera - Safari Móvil: - Mozilla Firefox - Google chrome - Safari		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Alta

RSNF- 03

RSNF-04			
Nombre	HTML5 y CSS3		
Descripción	El sistema de edición de escenarios será compatible con HTML5 y CSS3.		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 04

RSNF-05			
Nombre	Sistema Operativo visualización		
Descripción	El sistema de visualización de escenarios será compatible con los siguientes sistemas operativos: - Windows 7 - Windows 8		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 05

RSNF-06			
Nombre	Persistencia		
Descripción	La persistencia se gestionará a través del sistema de gestión de base de datos relacional MYSQL.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 06

RSNF-07			
Nombre	Formato archivos de modelos		
Descripción	El sistema será compatible con los siguientes formatos de modelos 3D: - COLLLADA (.dae)		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 07

RSNF-08			
Nombre	Formatos de textura		
Descripción	Los archivos de texturas de los modelos serán compatibles con los siguientes formatos: - PNG - JPG		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 08

3.4.2.3. Requisitos de usabilidad

RSNF-09			
Nombre	Interfaz fluida		
Descripción	El sistema de edición de escenarios dispondrá de una interfaz gráfica de diseño fluido compatible con tamaños de pantallas de 7.9 pulgadas o más (*tablets*, portátiles y ordenadores de sobremesa).		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Media

RSNF- 09

RSNF-10			
Nombre	Confirmación de operaciones		
Descripción	El sistema de edición de escenarios pedirá al usuario la confirmación de operaciones críticas.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Media

RSNF- 10

3.4.2.4. Requisitos de seguridad

RSNF-11			
Nombre	Inyección SQL		
Descripción	Las consultas a la base de datos se construirán de forma que se prevengan ataques de inyección SQL.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Alta

RSNF- 11

3.4.2.5. Requisitos de interfaz

RSNF-12			
Nombre	Mensajes de GREP		
Descripción	El sistema recibirá mensajes de GREP con formato JSON a través de llamadas asíncronas de AJAX.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Alta

RSNF- 12

RSNF-13			
Nombre	Codificación de caracteres		
Descripción	El sistema utilizará la codificación de caracteres UTF-8 en su interacción con sistemas externos.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 13

RSNF-14			
Nombre	Puerto HTTP sistema de edición de escenarios		
Descripción	El sistema de edición de escenarios será accesible desde el puerto HTTP 80.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 14

RSNF-15			
Nombre	HTTP Post		
Descripción	Se utilizará el método HTTP Post para toda petición HTTP que proporcione al sistema información suministrada por el usuario.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 15

3.4.2.6. Requisitos de operación

RSNF-16			
Nombre	Idioma		
Descripción	El sistema ofrecerá una interfaz de usuario en inglés.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Deseable	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 16

RSNF-17			
Nombre	*Drag and drop*		
Descripción	El sistema de edición de escenarios requerirá la funcionalidad *drag and drop*, para lo cual se requerirá de un dispositivo compatible en caso de sistemas no táctiles (e.g. ratón, *touchpad* o *trackball*).		
Fuente	Cliente	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Media	**Claridad**	Alta

RSNF- 17

RSNF-18			
Nombre	Entrada de texto		
Descripción	El sistema de edición de escenarios requerirá la introducción de texto, para lo cual se precisará de un dispositivo de entrada de texto.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 18

RSNF-19			
Nombre	Dispositivo de realidad aumentada		
Descripción	El sistema de visualización requerirá el uso de un dispositivo de realidad aumentada.		
Fuente	Analista	**Estabilidad**	Alta
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 19

3.4.2.7. Requisitos de recursos

RSNF-20			
Nombre	Tamaño archivos de modelado 3D		
Descripción	El tamaño máximo de un archivo de modelo 3D será de 10 Megabytes.		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 20

RSNF-21			
Nombre	Tamaño archivos de imagen.		
Descripción	El tamaño máximo de los archivos de imagen utilizados por el sistema será de 1 MB (e.g. texturas, iconos y planos 2D).		
Fuente	Analista	**Estabilidad**	Media
Necesidad	Esencial	**Verificabilidad**	Alta
Prioridad	Alta	**Claridad**	Alta

RSNF- 21

3.5. Matrices de trazabilidad

A continuación se presentan las matrices de trazabilidad de requisitos. Estas matrices permiten la sencilla visualización de dependencias entre requisitos de software y casos de uso con respecto a los requisitos de usuario. De esta forma, la tarea de detección de requisitos de usuario no cubiertos es directa. Adicionalmente, las matrices de trazabilidad juegan un papel importante a la hora de modificar o eliminar requisitos ofreciendo una clara lista de dependencias entre ellos, lo que permite determinar cómodamente la lista de requisitos que se verían afectados por la modificación o eliminación de uno de ellos.

REQUISITOS DE USUARIO	CASOS DE USO																
REQUISITOS DE CAPACIDAD	CU-01	CU-02	CU-03	CU-04	CU-05	CU-06	CU-07	CU-08	CU-09	CU-10	CU-11	CU-12	CU-13	CU-14	CU-15	CU-16	CU-17
RUC-01																	
RUC-02																	
RUC-03																	
RUC-04																	
RUC-05																	
RUC-06																	
REQUISITOS DE RESTRICCIÓN	CU-01	CU-02	CU-03	CU-04	CU-05	CU-06	CU-07	CU-08	CU-09	CU-10	CU-11	CU-12	CU-13	CU-14	CU-15	CU-16	CU-17
RUR-01																	
RUR-02																	
RUR-03																	
RUR-04																	

TABLA 1 - MATRIZ DE TRAZABILIDAD RU-CU

REQUISITOS DE USUARIO	REQUISITOS DE SOFTWARE FUNCIONALES																											
REQUISITOS DE CAPACIDAD	RSF-01	RSF-02	RSF-03	RSF-04	RSF-05	RSF-06	RSF-07	RSF-08	RSF-09	RSF-10	RSF-11	RSF-12	RSF-13	RSF-14	RSF-15	RSF-16	RSF-17	RSF-18	RSF-19	RSF-20	RSF-21	RSF-22	RSF-23	RSF-24	RSF-25	RSF-26	RSF-27	RSF-28	RSF-29
RUC-01																													
RUC-02																													
RUC-03																													
RUC-04																													
RUC-05																													
RUC-06																													
REQUISITOS DE RESTRICCIÓN	RSF-01	RSF-02	RSF-03	RSF-04	RSF-05	RSF-06	RSF-07	RSF-08	RSF-09	RSF-10	RSF-11	RSF-12	RSF-13	RSF-14	RSF-15	RSF-16	RSF-17	RSF-18	RSF-19	RSF-20	RSF-21	RSF-22	RSF-23	RSF-24	RSF-25	RSF-26	RSF-27	RSF-28	RSF-29
RUR-01																													
RUR-02																													
RUR-03																													
RUR-04																													

TABLA 2 - MATRIZ DE TRAZABILIDAD RU-RSF

REQUISITOS DE USUARIO	REQUISITOS DE SOFTWARE NO FUNCIONALES																			
REQUISITOS DE CAPACIDAD	RSNF-01	RSNF-02	RSNF-03	RSNF-04	RSNF-05	RSNF-06	RSNF-07	RSNF-08	RSNF-09	RSNF-10	RSNF-11	RSNF-12	RSNF-13	RSNF-14	RSNF-15	RSNF-16	RSNF-17	RSNF-18	RSNF-19	RSNF-20	RSNF-21
RUC-01																					
RUC-02																					
RUC-03																					
RUC-04																					
RUC-05																					
RUC-06																					
REQUISITOS DE RESTRICCIÓN	RSNF-01	RSNF-02	RSNF-03	RSNF-04	RSNF-05	RSNF-06	RSNF-07	RSNF-08	RSNF-09	RSNF-10	RSNF-11	RSNF-12	RSNF-13	RSNF-14	RSNF-15	RSNF-16	RSNF-17	RSNF-18	RSNF-19	RSNF-20	RSNF-21
RUR-01																					
RUR-02																					
RUR-03																					
RUR-04																					

TABLA 3 - MATRIZ DE TRAZABILIDAD RU-RSNF

4. Diseño

Tras la comprobación de la coherencia de todos los requisitos obtenidos en la fase de análisis se procede con el diseño del sistema. En primera instancia se ofrece una visión genérica del contexto en el que se integrará el sistema, identificándose los sistemas externos con los que se debe comunicar. Después de esto se continúa con la definición de la arquitectura de software, donde se detalla la división del sistema en componentes y se explica el modelo de datos utilizado. Además, en este apartado se declaran los patrones de diseño empleados. Para finalizar se realiza la comprobación de la cobertura de la funcionalidad requerida a través de una matriz de trazabilidad entre requisitos de software funcionales y componentes.

4.1. Contexto del sistema

Como se ha explicado con anterioridad, este proyecto tiene como objetivo el permitir la creación y visualización de escenarios en los que, a través de realidad aumentada, se permita la percepción de elementos virtuales manejados por la plataforma GREP. La creación y visualización de escenarios no debe precisar un conocimiento técnico elevado por parte del usuario. La plataforma GREP con la que el sistema interactúa fue desarrollada como parte de un proyecto previo y es accesible desde un navegador web en un ordenador de sobremesa.

A continuación se expone un diagrama en el que se muestra la comunicación entre los subsistemas.

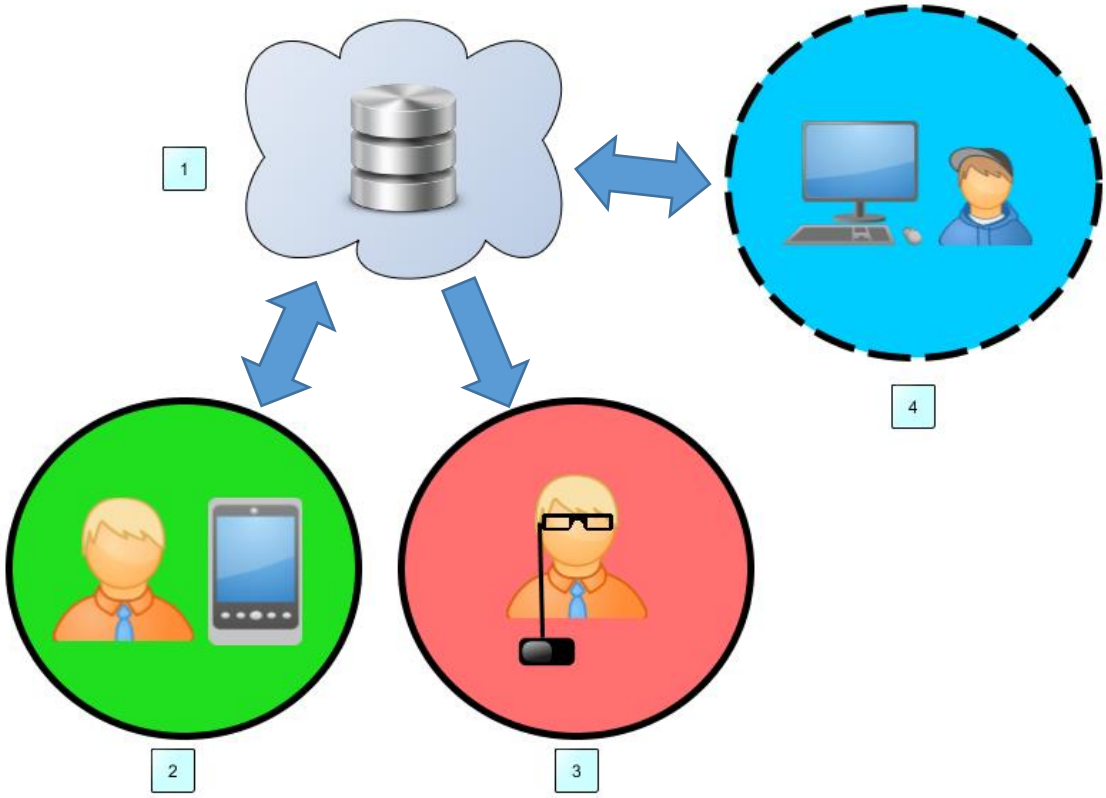


ILUSTRACIÓN 9 - CONTEXTO DEL SISTEMA

En este diagrama se pueden identificar los distintos subsistemas en los que se divide el proyecto. Se identifican cuatro elementos que deben comunicarse, marcados con un número cada uno:

1. Módulo de persistencia. Este elemento del sistema tiene la responsabilidad de almacenar la información gestionada por el resto de elementos del sistema y sirve a modo de comunicación indirecta de información. Los cambios aplicados al modelo de datos almacenado en él son perceptibles por el resto de elementos del sistema en un corto plazo de tiempo.

2. Sistema de edición de escenarios. Este subsistema se encarga de la creación de los escenarios y la gestión de los recursos necesarios para su visualización y es accesible por medio del uso de un dispositivo portátil que permite el posicionamiento de elementos virtuales sobre un plano del escenario a aumentar.

3. Sistema de visualización de escenarios y partidas. Este subsistema permite la visualización tanto de escenarios durante su edición, como de partidas de los juegos derivados de estos. Se precisa el uso de un dispositivo de realidad aumentada para la

visualización de los elementos virtuales así como de un dispositivo de captura de imagen para el posicionamiento del visualizador. Estos periféricos de salida y entrada de vídeo (respectivamente) son conectados a un dispositivo portátil que aporte el procesamiento necesario así como la fuente de energía.

4. Sistema GREP. Este sistema ejecuta un juego en un entorno virtual y se comunica a través de un sistema de conexión. Para ello el proyecto GREP también debe ser modificado ligeramente de para hacer uso de este sistema de conexión.

Cabe destacar que los sistemas de visualización y edición pueden ser ejecutados simultáneamente por la misma persona de manera que se permite la visualización de los elementos virtuales que se gestionan durante la edición de un escenario.

4.2. Arquitectura Software

En este apartado se detallan los componentes de cada uno de los tres subsistemas que forman el proyecto, así como los patrones de diseño aplicados.

Como se ha explicado en el apartado anterior, los diferentes subsistemas utilizarán el mismo módulo de persistencia. Para ello usarán todos ellos el mismo modelo de datos, implementándose el acceso a él de manera independiente por cada subsistema en función al lenguaje de programación y tecnología.

4.2.1. Modelo de datos

Como se puede apreciar en el esquema de la arquitectura de software (Ilustración 2) existirá un único banco de datos compartido por los sistemas de edición, visualización y conexión con GREP. La información principal que se gestiona es:

- los escenarios, con la definición de los elementos virtuales presentes en ellos y los marcadores que permitan la detección de la posición del visualizador.
- los juegos mixtos, con sus elementos de juego y los identificadores de las reglas de juego asociadas.
- las partidas, con sus elementos de partida asociados a los elementos de juego mixto.

Se puede entender un escenario como la definición estática del estado inicial de un juego mixto junto con los recursos necesarios para visualizarlo. Un juego mixto se define por un escenario, la identificación de los elementos del juego con los correspondientes elementos del escenario y

por de las reglas que se aplican al juego y a sus elementos. Tanto los escenarios como los juegos mixtos se consideran estáticos en el sentido de que una vez finalizado el proceso de edición no se espera su modificación, aunque esta está habilitada.

Tras definirse un juego a partir de un escenario se pueden instanciar múltiples partidas de un juego definiendo los elementos de partida correspondientes a los elementos de juego, su estado de activación y su posición, rotación y escala.

A continuación se muestra el diseño del diagrama relacional que se va a implementar.

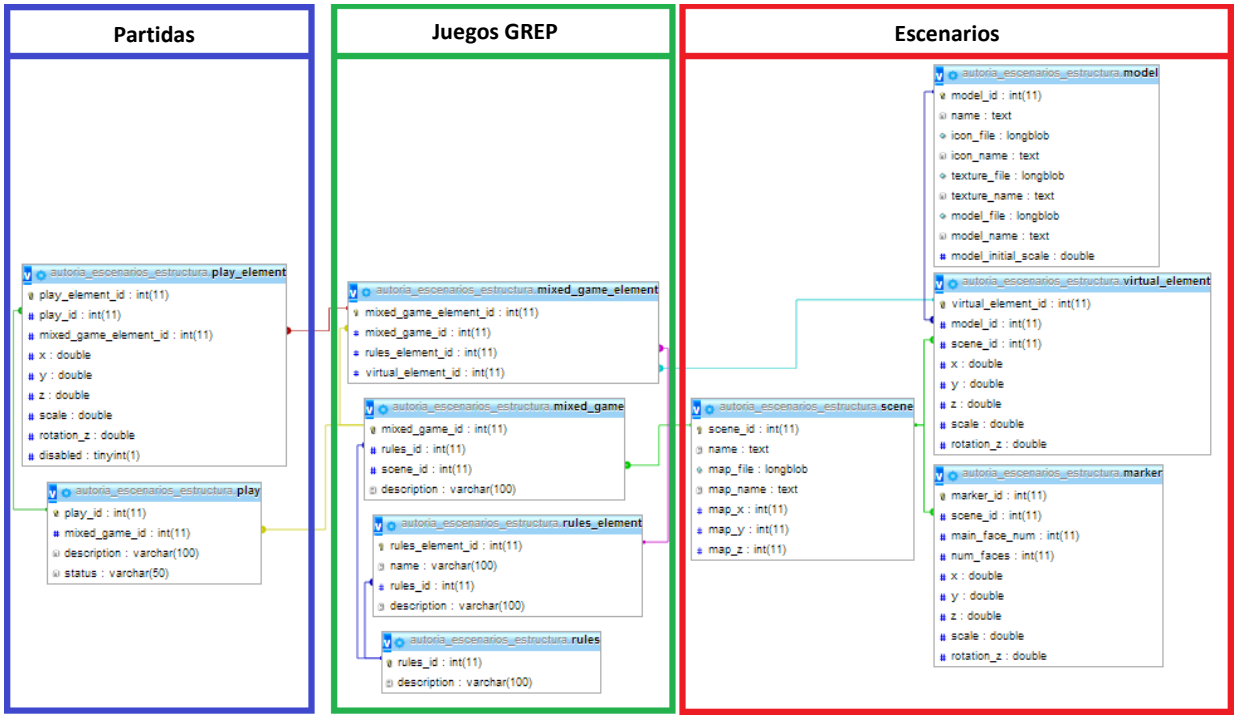


ILUSTRACIÓN 10 – DIAGRAMA DE BASE DE DATOS

Como se puede ver en la ilustración, la base de datos se divide en tres subgrupos: *Escenarios*, *Juegos GREP* y *Partidas*. El subgrupo *Escenarios* almacena la información definida en el modo Edición y corresponde con la definición estática de los escenarios. Por otro lado, *Juegos GREP* contiene una adaptación de las entidades utilizadas en *GREP* para la definición de juegos, ampliada con la asociación entre elemento de juego y elemento virtual (aunque la gestión de los juegos queda fuera del alcance de este proyecto). Por último el subgrupo *Partidas* maneja la información relativa al modo *Partida*.

En los siguientes apartados se desglosan estos subgrupos y se explican las tablas que los componen.

4.2.1.1. Escenarios

Este subgrupo de tablas se encarga de la definición del escenario a través de las tablas *scene*, *model*, *virtual element* y *marker*.

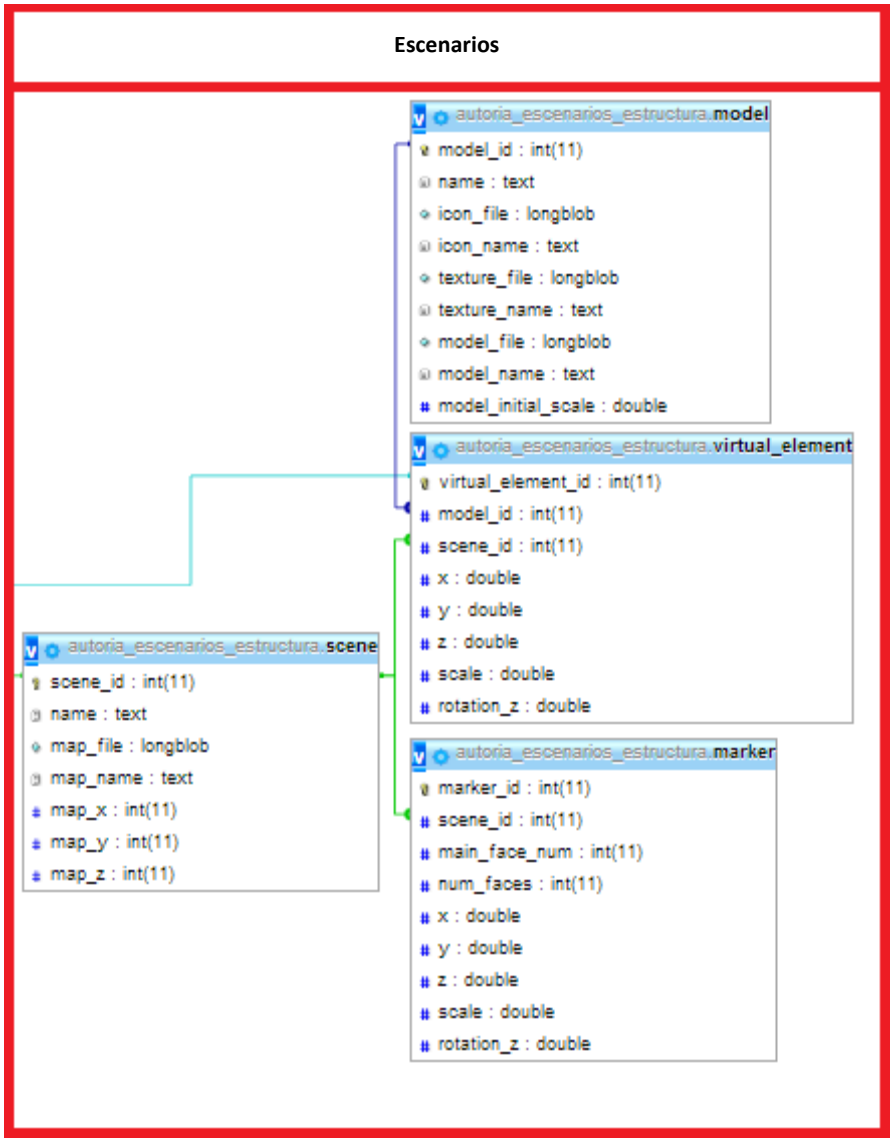


ILUSTRACIÓN 11 - BD - ESCENARIOS

4.2.1.1.1. scene

Esta tabla hace referencia al escenario a aumentar con elementos virtuales. Los campos que contiene son:

- scene_id: Clave primaria numérica entera que identifica unívocamente un escenario.
- name: Campo textual que contiene el nombre del escenario.

- map_file: Campo de datos binarios que almacena la imagen del plano del escenario.
- map_name: Campo de texto que almacena el nombre del fichero del plano.
- map_x: Campo numérico real que almacena una dimensión del plano horizontal, la longitud. Medido en centímetros.
- map_y: Campo numérico real que almacena una dimensión del plano horizontal, la anchura. Medido en centímetros.
- map_z: Campo numérico real que almacena la dimensión vertical, la altura. Medido en centímetros.

4.2.1.1.2. model

Esta tabla almacena los recursos necesarios para visualizar un modelo 3D. Dispone de los siguientes campos:

- model_id: Clave primaria numérica entera que identifica unívocamente un modelo.
- name: Campo de texto que indica el nombre del modelo 3D.
- icon_file: Campo de datos binarios que almacena una imagen icónica representativa del modelo. Codificado en Base64.
- icon_name: Campo de texto que almacena el nombre de la imagen icónica almacenada.
- texture_file: Campo de datos binarios que almacena la imagen de textura que se aplica al modelo. Opcional.
- texture_name: Campo de texto que almacena el nombre del fichero de textura almacenado.
- model_file: Campo de datos binarios que almacena el fichero de modelado 3D que define la estructura tridimensional del modelo así como el esqueleto y las animaciones asociadas.
- model_name: Campo de texto que almacena el nombre del fichero de modelado 3D almacenado.

- model_initial_scale: Campo numérico real que almacena el factor de escala que hay que aplicar al modelo para que se obtenga un tamaño normalizado. Se aplica de forma independiente a cualquier otra escala.

4.2.1.1.3. virtual_element

Esta tabla contiene la definición de un elemento virtual de un escenario. Contiene los siguientes campos:

- virtual_element_id: Clave primaria numérica entera que identifica unívocamente un elemento virtual.
- model_id: Clave foránea numérica entera que identifica unívocamente un modelo.
- scene_id: Clave foránea numérica entera que identifica unívocamente un escenario.
- x: Campo numérico real que almacena una dimensión del plano horizontal, la longitud. Medido en centímetros.
- y: Campo numérico real que almacena una dimensión del plano horizontal, la anchura. Medido en centímetros.
- z: Campo numérico real que almacena la dimensión vertical, la altura. Medido en centímetros.
- scale: Campo numérico real que indica el factor de escala a aplicar respecto del tamaño normal del elemento virtual. Se aplica en todas las dimensiones simultáneamente.
- rotation_z: Campo numérico real que indica el giro sobre el eje vertical positivo según la regla de la mano derecha. Medido en grados.

4.2.1.1.4. marker

Esta tabla define los marcadores que permiten la detección de la posición y orientación del visualizador respecto al marcador. Sabida la posición y orientación del marcador se posibilita el cálculo de la posición del visualizador respecto de todos los elementos del mundo virtual. Contiene los siguientes campos:

- marker_id: Clave primaria numérica entera que identifica unívocamente un marcador.
- scene_id: Clave foránea numérica entera que identifica unívocamente un escenario.

- main_face_num: Campo numérico entero que identifica el código asociado a la cara principal de un marcador.
- num_faces: Campo numérico entero que indica el número de caras que tiene un marcador. Para marcadores planos es 1, para marcadores cúbicos 6.
- x: Campo numérico real que almacena una dimensión del plano horizontal, la longitud. Medido en centímetros.
- y: Campo numérico real que almacena una dimensión del plano horizontal, la anchura. Medido en centímetros.
- z: Campo numérico real que almacena la dimensión vertical, la altura. Medido en centímetros.
- scale: Campo numérico real que indica el factor de escala a aplicar respecto del tamaño normal del marcador. Se aplica en todas las dimensiones simultáneamente.
- rotation_z: Campo numérico real que indica el giro sobre el eje vertical positivo según la regla de la mano derecha. Medido en grados.

4.2.1.2. Juegos GREP

Este conjunto de tablas define un juego GREP y establece la asociación entre elementos de partida y elementos virtuales de escenario a través de los elementos de juego mixto. Tanto los juegos como sus elementos pueden verse regidos por reglas que definen el comportamiento del juego y sus elementos ante los distintos eventos ocurridos (heredado de GREP). Las tablas que componen este subgrupo son *mixed_game*, *mixed_game_element*, *rules* y *rules_element*.

ILUSTRACIÓN 12 - BD - JUEGOS GREP

4.2.1.2.1. mixed_game

Tabla que almacena la información de un juego GREP de realidad mixta.

- mixed_game_id: Clave primaria numérica entera que identifica unívocamente un juego mixto.
- rules_id: Clave foránea numérica entera que identifica unívocamente las reglas asociadas al juego.
- scene_id: Clave foránea numérica entera que identifica unívocamente un escenario.
- description: Campo de texto que describe un juego mixto.

4.2.1.2.2. mixed_game_element

Tabla que define los elementos de un juego mixto a partir de elementos virtuales de un escenario. Los campos que contiene son:

- mixed_game_element_id: Clave primaria numérica entera que identifica unívocamente un elemento de juego mixto.
- mixed_game_id: Clave foránea numérica entera que identifica unívocamente un juego mixto.
- rules_element_id: Clave foránea numérica entera que identifica unívocamente unas reglas de juego.
- virtual_element_id: Clave foránea numérica entera que identifica unívocamente un elemento virtual.

4.2.1.2.3. rules

Esta tabla identifica las reglas de juego mixto que definen propiedades del juego o comportamientos de sus elementos ante determinados eventos. Contiene los siguientes campos:

- rules_id: Clave primaria numérica entera que identifica unívocamente una reglas.
- description: Campo de texto que describe un conjunto de elemento de regla.

4.2.1.2.4. rules _element

La definición completa de las reglas no se guarda en la base de datos, sino en un fichero xml que es interpretado por la plataforma GREP para su aplicación durante el juego. En esta tabla únicamente se recogen los nombres e identificadores de las distintas entidades definidas en las reglas de juego que podrán tener representación en un juego en entorno realidad mixta.

- rules_element_id: Clave primaria numérica entera que identifica unívocamente un elemento de reglas.
- name: Campo de texto que identifica en una o pocas palabras el elemento de reglas.
- rules_id: Clave foránea numérica entera que identifica las reglas a las que pertenece este elemento de reglas.
- description: Campo de texto que describe elemento de reglas.

4.2.1.3. Partidas

Este conjunto de tablas define las partidas, es decir, las instancias de un juego GREP. Las partidas se componen de elementos de partida que tienen una correspondencia directa con los elementos del juego y por ende con los elementos virtuales.

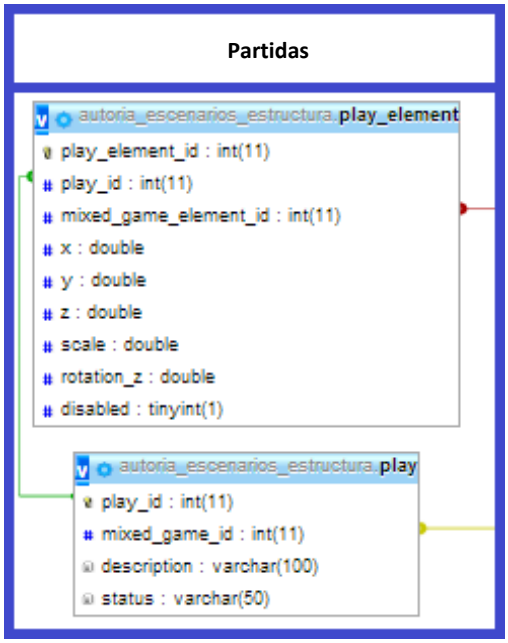


ILUSTRACIÓN 13 - BD - PARTIDAS

4.2.1.3.1. play

Tabla que gestiona las partidas asociadas a un juego junto con una descripción y el estado de la partida (en curso o finalizada).

- play_id: Clave primaria numérica entera que identifica unívocamente una partida.
- mixed_game_id: Clave foránea numérica entera que identifica el juego mixto del que la partida es instancia.
- description: Campo de texto que describe la partida.
- status: Campo de texto que describe el estado de la partida. Actualmente solo admite dos estados correspondientes a si una partida está finalizada o en curso: *ongoing* o *ended*.

4.2.1.3.2. play_element

Tabla que gestiona los elementos de una partida asociándolos con elementos de juego mixto y especificando posición, orientación, escala y estado de visualización.

- play_element_id: Clave primaria numérica entera que identifica unívocamente un elemento de partida.
- play_ id: Clave foránea numérica entera que identifica la partida a la que pertenece el elemento de partida.
- mixed_game_element_id: Clave foránea numérica entera que identifica el elemento de juego al correspondiente al elemento de partida.
- x: Campo numérico real que almacena una dimensión del plano horizontal, la longitud. Medido en centímetros.
- y: Campo numérico real que almacena una dimensión del plano horizontal, la anchura. Medido en centímetros.
- z: Campo numérico real que almacena la dimensión vertical, la altura. Medido en centímetros.
- scale: Campo numérico real que indica el factor de escala a aplicar respecto del tamaño normal del elemento virtual. Se aplica en todas las dimensiones simultáneamente.
- rotation_z: Campo numérico real que indica el giro sobre el eje vertical positivo según la regla de la mano derecha. Medido en grados.

  disabled: Campo numérico entero que indica si el elemento está desactivado o no para su visualización.

4.2.2. Sistema completo y componentes comunes

A continuación se mostrará el diagrama completo de componentes de los sistemas diseñados.

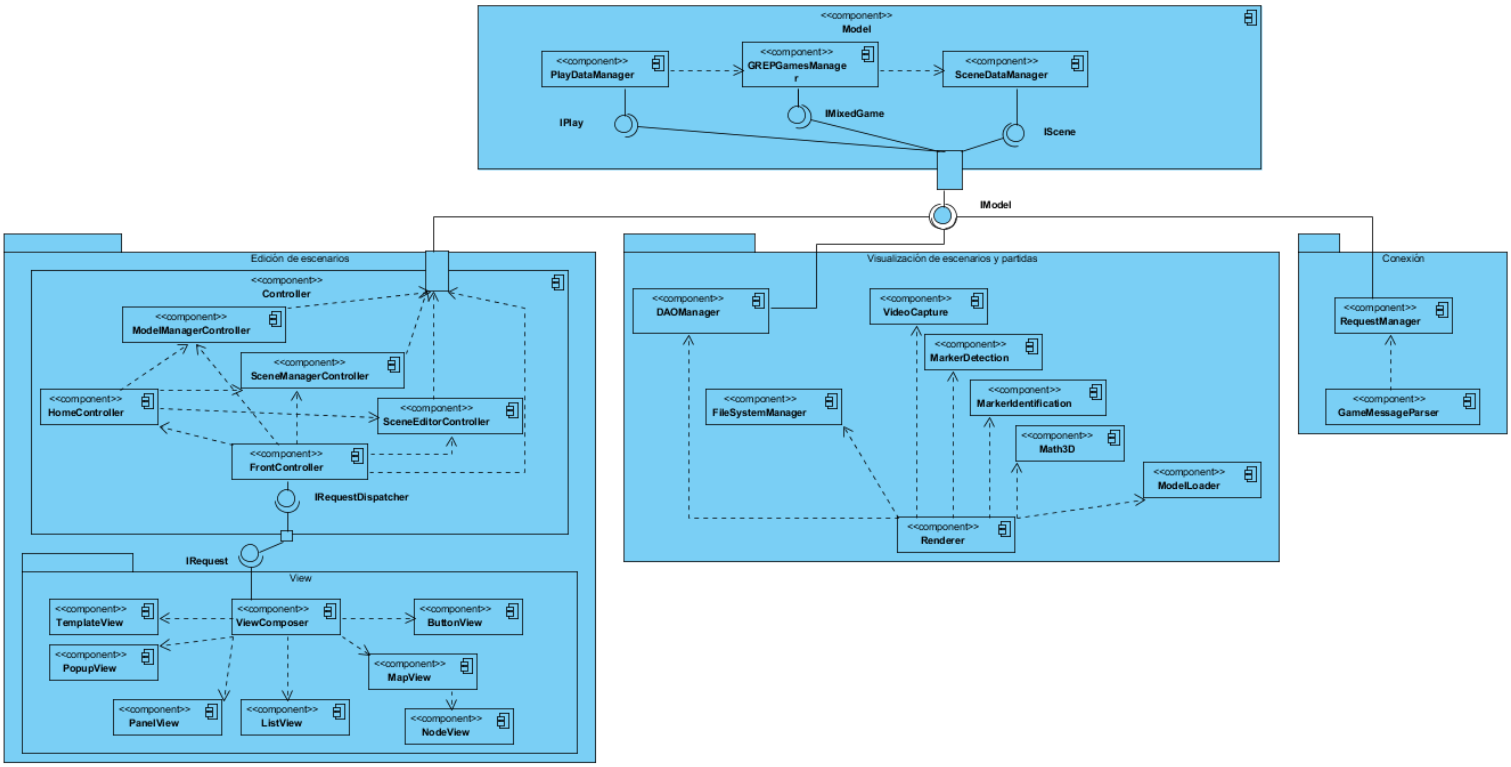


ILUSTRACIÓN 14- DIAGRAMA DE COMPONENTES

En el diagrama expuesto se muestran la división de los distintos subsistemas en componentes. En este apartado se explica el componente del modelo de datos, reutilizado entre sistemas.

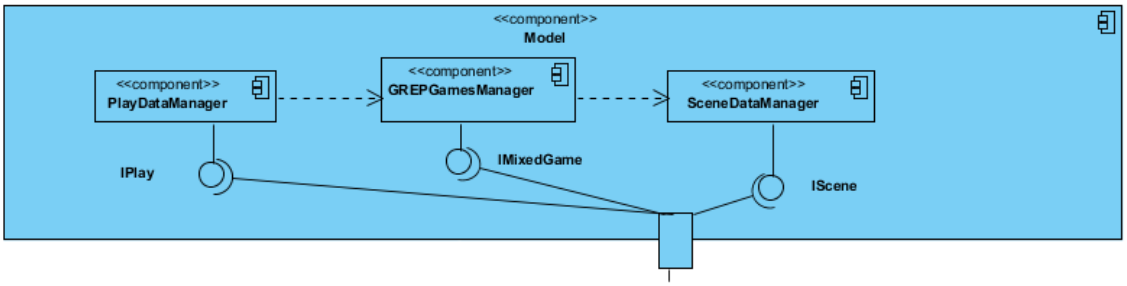


ILUSTRACIÓN 15 – COMPONENTES DE MODELO

Este componente se subdivide en componentes que gestionan las entidades asociadas con los tres subgrupos del modelo de datos presentados.

4.2.3. Sistema de edición de escenarios

El sistema de edición de escenarios clasifica sus componentes según el patrón de diseño *Modelo-Vista-Controlador*.

Este patrón de diseño diferencia claramente tres grupos de componentes, cada uno con una función diferente, detallados a continuación:

- Modelo. Encapsula los datos con los que trabaja el sistema de forma independiente a su representación.
- Vista. Representa la interfaz con la que el usuario interactúa y es responsable de la representación de los datos proporcionados al usuario.
- Controlador. Recibe peticiones suministradas por el usuario a través de su interacción con la vista. Estas peticiones implican un cambio de los datos a visualizar por el usuario o una modificación de los datos almacenados, por lo que actúa de intermediario entre el modelo y la vista. A partir de los datos suministrados por el usuario se decide qué vista mostrar.

Esta división permite separar la lógica y la interfaz, de manera que minimiza la dependencia entre componentes a través de la modularización. Esta separación facilita el mantenimiento y ampliación del sistema.

A continuación se muestra el diagrama de componentes (excluyendo el componente del modelo):

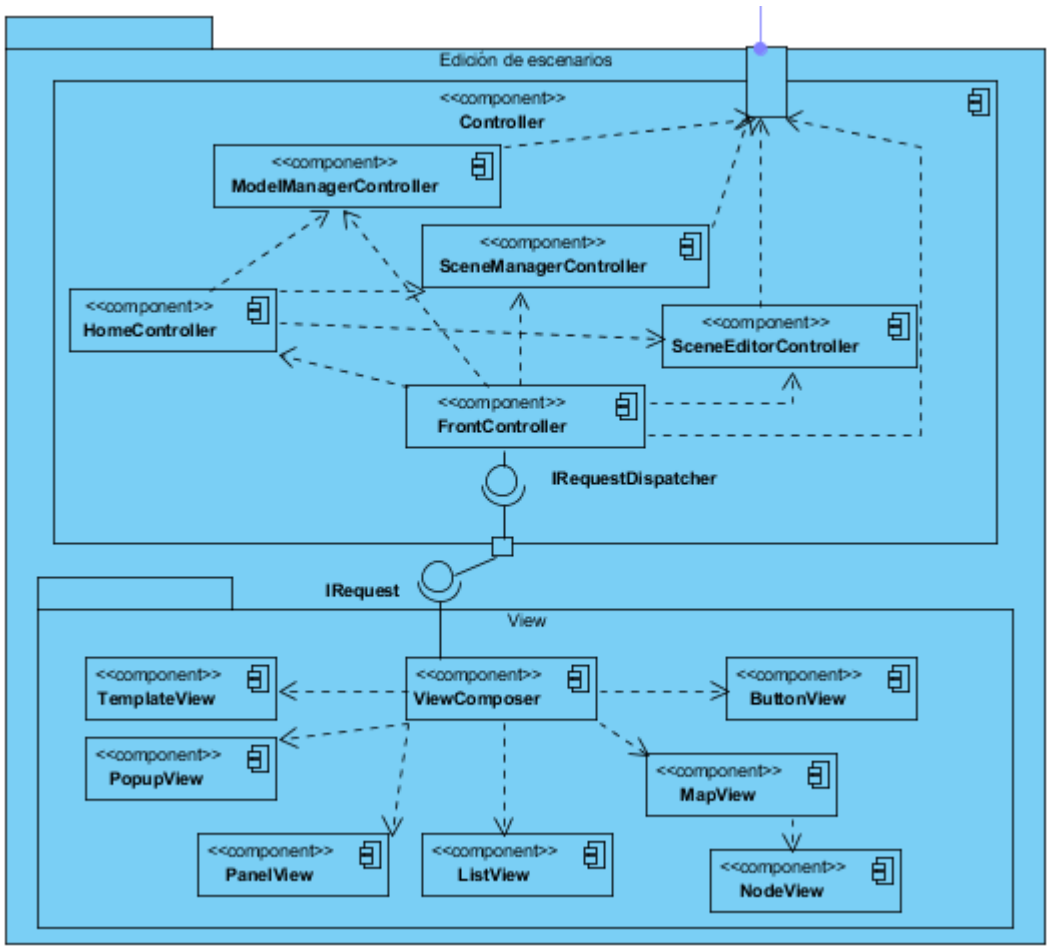


ILUSTRACIÓN 16 - COMPONENTES SISTEMA DE EDICIÓN DE ESCENARIOS

El componente controlador utiliza un patrón de diseño de acceso centralizado a través de un componente que distribuye las peticiones al controlador concreto que la puede gestionar.

Por otro lado, la vista utiliza un componente generador de vistas, el cual utiliza los distintos componentes de vista para ofrecer la generación de vistas compuestas.

4.2.4. Sistema de visualización de escenarios y partidas

El sistema de visualización de escenarios y partidas no utiliza un patrón de diseño estándar. Este sistema se centra en un componente principal, el Renderizador (*Renderer*), que se encarga de la actualización de la imagen proyectada en el dispositivo de realidad aumentada en tiempo real con el uso de un bucle.

Para ello se define una fase de configuración (*setup)* que se ejecuta al iniciar el programa. En esta fase se obtienen los recursos necesarios para la visualización de los elementos virtuales del escenario o de los elementos de partida de la partida a visualizar.

Tras la fase de configuración se ejecuta indefinidamente la secuencia de etapas actualización - dibujado (*update-draw)*.

La etapa de actualización se encarga de obtener una imagen del dispositivo de captura de vídeo y de procesarla en busca de marcadores para el cálculo de la posición y orientación del usuario en el mundo virtual. Para esto se utilizan diversos componentes auxiliares.

Por último la etapa de dibujado muestra los elementos virtuales en el dispositivo de realidad aumentada aplicando las transformaciones 3D necesarias a los modelos.

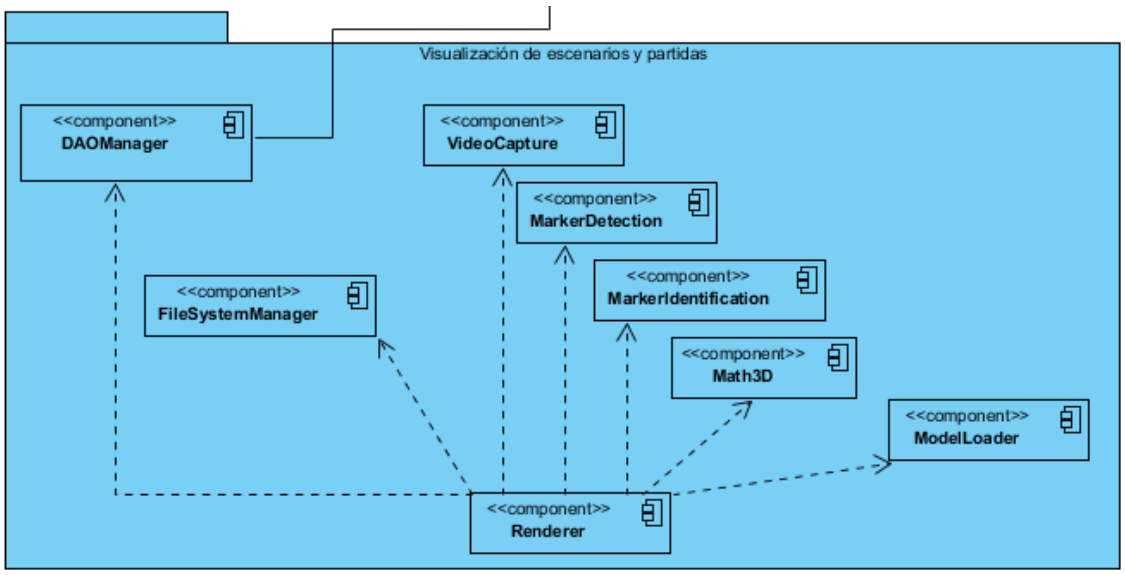


ILUSTRACIÓN 17 - COMPONENTES SISTEMA VISUALIZACIÓN

El diagrama muestra la centralización del diseño en el componente de renderización, el cual hace uso de los demás componente como componentes auxiliares con responsabilidades concretas.

4.2.5. Sistema de conexión

Este sistema es consta de dos simples componentes, uno que accede a los datos relacionados con las partidas permitiendo su modificación y otro que recibe mensajes desde el sistema GREP, los interpreta y solicita la actualización del modelo correspondiente.

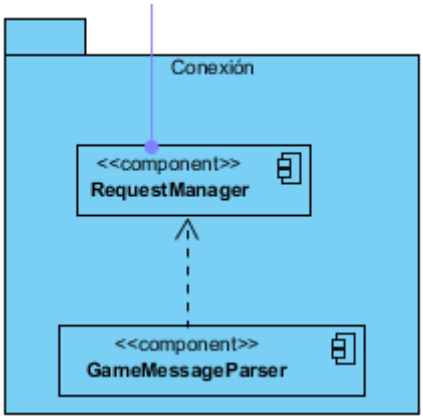


ILUSTRACIÓN 18 - COMPONENTES SISTEMA CONEXIÓN

4.3. Especificación de componentes

Para sintetizar la información de los componentes de los subsistemas presentados y facilitar la comprobación de los requisitos obtenidos en la fase de análisis, se presenta a continuación la especificación de estos en forma de tablas. Las tablas de dispondrán de los siguientes campos:

- Identificador. Permite reconocer unívocamente cada componente. La nomenclatura que sigue es la siguiente:

  CO-XX

  Donde XX es un número entero de dos dígitos que identifica el componente. Se asignará de manera incremental desde el valor 01 sin garantizarse la existencia de valores intermedios debido a la posibilidad de modificar la cantidad de los componentes.

- Nombre: especifica el nombre del componente.
- Nivel: especifica el nivel al que se encuentra el componente.
- Propósito: especifica los identificadores de los requisitos de software cuya funcionalidad es tratada por el componente.
- Función: describe las funcionalidades que lleva a cabo el componente.
- Dependencias: especifica las interfaces y componentes que precisa para su correcto funcionamiento.
- Interfaces: especifica las interfaces proporcionadas por el componente.

- Subordinados: especifica los componentes hijo, es decir, los componentes en un nivel inmediatamente inferior encapsulados dentro de este componente.

A continuación se muestra la plantilla de tabla de especificación de componentes.

CO-XX			
Nombre		**Nivel**	
Propósito			
Función			
Dependencias			
Interfaces			
Subordinados			

PLANTILLA ESPECIFICACIÓN DE COMPONENTES 1

Una vez especificado el formato de las tablas a utilizar se pasa a listar las tablas.

CO-01			
Nombre	Model	**Nivel**	1
Propósito	Todos los requisitos de software funcionales.		
Función	Ofrece las entidades del modelo de datos para su modificación persistente y consulta.		
Dependencias	Ninguna.		
Interfaces	IModel: ofrece el acceso y modificación del modelo de datos reuniendo las interfaces IPlay, IMixeGame e IScene.		
Subordinados	CO-04, CO-05, CO-06.		

CO- 01. COMPONENTE MODEL

CO-02			
Nombre	Controller	**Nivel**	1
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Recibe peticiones desde la vista y las gestiona accediendo al modelo de daos para su actualización o consulta y actualiza la vista.		
Dependencias	IModel: requiere acceso a las operaciones de IScene, que permiten el acceso al modelo de datos asociado con los escenarios.		
Interfaces	IRequest: ofrece una interfaz de recepción de peticiones para la vista.		
Subordinados	CO-06, CO-07, CO-08, CO-09, CO-10, CO-11.		

CO- 02. COMPONENTE CONTROLLER

CO-03			
Nombre	View	**Nivel**	1
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Permite la interacción del usuario con el sistema de edición de escenarios realizando peticiones al controlador y visualizando los datos.		
Dependencias	IRequest: realiza peticiones al controlador.		
Interfaces	Ninguna.		
Subordinados	CO-12, CO-13, CO-14, CO-15, CO-16, CO-17, CO-18, CO-19.		

CO- 03. COMPONENTE VIEW

CO-04

Nombre	PlayDataManager	**Nivel**	2
Propósito	RSF-23, RSF-24, RSF-25, RSF-26, RSF-28, RSF-29.		
Función	Ofrece acceso a las entidades de partida y elemento de partida permitiendo su consulta y modificación.		
Dependencias	CO-05.		
Interfaces	IPlay: permite la modificación y consulta de los elementos de partida y de las partidas.		
Subordinados	Ninguno.		

CO- 04. COMPONENTE PLAYDATAMANAGER

CO-05			
Nombre	GREPGamesManager	**Nivel**	2
Propósito	RSF-23, RSF-24, RSF-25, RSF-26, RSF-28, RSF-29.		
Función	Ofrece acceso a las entidades de elementos de juego mixto, juegos mixtos, elementos de regla y reglas de los juegos GREP permitiendo su consulta y modificación.		
Dependencias	CO-06.		
Interfaces	IMixedGame: permite la modificación y consulta de los elementos de juego mixto, juegos mixtos, reglas y elementos de regla.		
Subordinados	Ninguno.		

CO- 05. COMPONENTE GREPGAMESMANAGER

CO-06

Nombre	SceneDataManager	**Nivel**	2
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22, RSF-26, RSF-27, RSF-28.		
Función	Ofrece acceso a las entidades asociadas con los escenarios, marcadores, elementos virtuales y modelos permiendo su consulta y modificación.		
Dependencias	Ninguna.		
Interfaces	IScene: permite la modificación y consulta de los escenarios, marcadores, elementos virtuales		
Subordinados	Ninguno.		

CO- 06. COMPONENTE SCENEDATAMANAGER

CO-07			
Nombre	ModelManagerController	**Nivel**	2
Propósito	RSF-01, RSF-02, RSF-03, RSF-07, RSF-08, RSF-09,		
Función	Controla las peticiones asociadas a los modelos 3D de los escenarios permitiendo su gestión.		
Dependencias	IModel: acceso a modelos 3D.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 07. COMPONENTE MODERMANGERCONTROLLER

CO-08

Nombre	SceneManagerController	**Nivel**	2
Propósito	RSF-04, RSF-05, RSF-06,		
Función	Controla las peticiones asociadas a los escenarios permitiendo su gestión.		
Dependencias	IModel: acceso a los escenarios.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 08. COMPONENTE SCENEMANAGERCONTROLLER

CO-09			
Nombre	SceneEditorController	**Nivel**	2
Propósito	RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Controla las peticiones asociadas a los elementos virtuales, marcadores y modelos de un escenario permitiendo la edición de este.		
Dependencias	IModel: acceso a elementos virtuales, marcadores y modelos de los escenarios.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 09. COMPONENTE SCENEEDITORCONTROLLER

CO-10

Nombre	HomeController	**Nivel**	2
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Sirve de punto inicial para el uso del sistema de edición de escenarios.		
Dependencias	CO-07, CO-08, CO-09.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 10. COMPONENTE HOMECONTROLLER

CO-11			
Nombre	FrontController	**Nivel**	2
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Recibe las peticiones de la vista y las redirecciona al controlador que puede gestionarla.		
Dependencias	CO-07, CO-08, CO-09, CO-10.		
Interfaces	IRequestDispatcher: ofrece una interfaz de distribución de peticiones.		
Subordinados	Ninguno.		

CO- 11. COMPONENTE FRONTCONTROLLER

CO-12

Nombre	ViewComposer	Nivel	2
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Genera vistas complejas a partir de elementos de vista simples como listas, botones, diálogos…etc. Genera las principales interfaces del sistema de edición de escenarios.		
Dependencias	IRequest: acceso a peticiones al controlador. CO-13, CO-14, CO-15, CO-16, CO-17, CO-18, CO-19.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 12. COMPONENTE VIEWCOMPOSER

CO-13			
Nombre	TemplateView	Nivel	2
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Vistas que manejan la distribución de los componentes en la interfaz permitiendo una consistencia en las interfaces del sistema de edición de escenarios.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 13. COMPONENTE TEMPLATEVIEW

CO-14

Nombre	PopupView	**Nivel**	2
Propósito	RSF-01, RSF-02, RSF-04, RSF-05, RSF-10, RSF-11, RSF-17.		
Función	Vistas que permiten la presentación modal de información y formularios.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 14. COMPONENTE POPUPVIEW

CO-15			
Nombre	PanelView	**Nivel**	2
Propósito	RSF-11, RSF-12, RSF-17, RSF-18.		
Función	Vistas que permiten la presentación de componentes visuales en menús colapsables.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 15. COMPONENTE PANELVIEW

CO-16			
Nombre	ListView	**Nivel**	2
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-10, RSF-11, RSF-17.		
Función	Vistas que presentan la información de una sucesión de elementos iguales.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 16. COMPONENTE LISTVIEW

CO-17			
Nombre	ButtonView	**Nivel**	2
Propósito	RSF-01, RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-11, RSF-12, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-20, RSF-21, RSF-22.		
Función	Vistas que permiten la interacción del usuario para la ejecución de una acción como por ejemplo el uso de un *slider* para ejecutar una rotación.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 17. COMPONENTE BUTTONVIEW

CO-18			
Nombre	MapView	**Nivel**	2
Propósito	RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Vistas que permiten añadir nodos sobre una imagen y su manipulación con interacción de arrastre (*drag* y *drag-and-drop*).		
Dependencias	CO-19.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 18. COMPONENTE MAPVIEW

CO-19			
Nombre	NodeView	**Nivel**	2
Propósito	RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-17, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22.		
Función	Vista que encapsula un elemento manipulable y anidable.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 19. COMPONENTE NODEVIEW

CO-20			
Nombre	DAOManager	**Nivel**	1
Propósito	RSF-07, RSF-08, RSF-09, RSF-27, RSF-28.		
Función	Componente que encapsula el acceso al modelo de datos para la obtención de marcadores, elementos virtuales, modelos y elementos de partida del sistema de visualización de escenarios y partidas.		
Dependencias	IModel: acceso a marcadores, elementos virtuales, modelos y elementos de partida.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 20. COMPONENTE DAOMANAGER

CO-21			
Nombre	FileSystemManager	**Nivel**	1
Propósito	RSF-01, RSF-02, RSF-03, RSF-27, RSF-28.		
Función	Accede al sistema de ficheros subyacente para la gestión de los ficheros de modelos y texturas.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 21. COMPONENTE FILESYSTEMMANAGER

CO-22			
Nombre	VideoCapture	**Nivel**	1
Propósito	RSF-07, RSF-08, RSF-09, RSF-10, RSF-27, RSF-28.		
Función	Gestiona el acceso a la cámara de captura de vídeo para la obtención de imágenes a procesar en busca de marcadores.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 22. COMPONENTE VIDEOCAPTURE

CO-23			
Nombre	MarkerDetection	**Nivel**	1
Propósito	RSF-07, RSF-08, RSF-09, RSF-10, RSF-27, RSF-28.		
Función	Detecta la presencia de posibles marcadores en imágenes.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 23. COMPONENTE MARKERDECTECTION

CO-24			
Nombre	MarkerIdentification	**Nivel**	1
Propósito	RSF-08, RSF-09, RSF-10, RSF-27, RSF-28.		
Función	Determina si los marcadores que se han detectado son realmente marcadores o no y decodifica el número asociado.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 24. COMPONENTE MARKERIDENTIFICATION

CO-25			
Nombre	Math3D	**Nivel**	1
Propósito	RSF-27, RSF-28.		
Función	Calcula la posición de elementos virtuales y marcadores en el mundo virtual además de posicionar al usuario respecto de un marcador identificado.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 25. COMPONENTE MATH3D

CO-26			
Nombre	ModelLoader	**Nivel**	1
Propósito	RSF-01, RSF-02, RSF-03, RSF-27, RSF-28.		
Función	Gestiona los ficheros de modelado 3D y sus texturas para la renderización de estos.		
Dependencias	Ninguna.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 26. COMPONENTE MODELLOADER

CO-27			
Nombre	Renderer	**Nivel**	1
Propósito	RSF-01, RSF-02, RSF-03, RSF-07, RSF-08, RSF-09, RSF-10, RSF-27, RSF-28.		
Función	Gestiona las etapas de la visualización (*setup, update, draw*) y maneja las iteraciones para la actualización. Utiliza el resto de componentes como auxiliares para el desempeño de la función del sistema de visualización de escenarios y partidas. Es el componente principal del sistema.		
Dependencias	CO-20, CO-21, CO-22, CO-23, CO-24, CO-25, CO-26.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 27. COMPONENTE RENDERER

CO-28			
Nombre	RequestManager	**Nivel**	1
Propósito	RSF-23, RSF-24, RSF-25, RSF-26, RSF-29.		
Función	Recibe las peticiones de actualización de partidas y accede al modelo de datos que lo permite.		
Dependencias	IModel: acceso a las partidas, elementos de partida y escenarios para la instanciación y actualización de las partidas.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 28. COMPONENTEREQUESTMANAGER

CO-29			
Nombre	GameMessageParser	**Nivel**	1
Propósito	RSF-23, RSF-24, RSF-25, RSF-26, RSF-29.		
Función	Interpreta los mensajes recibidos del sistema GREP y los encapsula en estructuras de datos serializables para el envío de los parámetros actualizados al componente de gestión de peticiones.		
Dependencias	CO-28.		
Interfaces	Ninguna.		
Subordinados	Ninguno.		

CO- 29. COMPONENTE GAMEMESSAGEPARSER

4.4. Matriz de trazabilidad

En este apartado se presenta la matriz de trazabilidad que permite visualizar cobertura de los requisitos de software funcionales por parte de los componentes de los distintos subsistemas.

	REQUISITOS DE SOFTWARE FUNCIONALES																												
	RSF-01	RSF-02	RSF-03	RSF-04	RSF-05	RSF-06	RSF-07	RSF-08	RSF-09	RSF-10	RSF-11	RSF-12	RSF-13	RSF-14	RSF-15	RSF-16	RSF-17	RSF-18	RSF-19	RSF-20	RSF-21	RSF-22	RSF-23	RSF-24	RSF-25	RSF-26	RSF-27	RSF-28	RSF-29
C0-01																													
C0-02																													
C0-03																													
C0-04																													
C0-05																													
C0-06																													
C0-07																													
C0-08																													
C0-09																													
C0-10																													
C0-11																													
C0-12																													
C0-13																													
C0-14																													
C0-15																													
C0-16																													
C0-17																													
C0-18																													
C0-19																													
C0-20																													
C0-21																													
C0-22																													
C0-23																													
C0-24																													
C0-25																													
C0-26																													
C0-27																													
C0-28																													
C0-29																													

4.5. Prototipos de interfaz

La utilización de prototipos de interfaz permite obtener una visión general de cómo será el sistema final en términos de apariencia y funcionalidad ofrecida al usuario. Gracias al uso de esta técnica, el cliente podrá tener una toma de contacto con el sistema a desarrollar. A través de la evaluación del prototipo se posibilita la detección de disconformidades entre cliente y desarrollador, de manera que se eviten cambios en el sistema una vez implementado.

En este proyecto se han utilizado prototipos de baja fidelidad, que se caracterizan por su velocidad de desarrollo y simpleza, lo que alienta al cliente a expresar sin barreras sus opiniones sobre el prototipo. Esto contrasta con los prototipos de alta fidelidad, que dan una visión más completa y detallada del sistema final, lo que en ocasiones cohíbe al cliente a expresar su opinión.

Dicho esto y dado que el sistema de conexión no precisa interfaz gráfica y que la interfaz del sistema de visualización de escenarios y partidas consiste exclusivamente en la superposición de elementos virtuales sobre la visión del usuario, se han desarrollado prototipos de interfaz exclusivamente para el sistema de edición de escenarios y se detallan a continuación.

4.5.1. Plantilla y menú principal

A continuación se muestra la plantilla que utilizarán las diferentes interfaces de usuario del sistema.

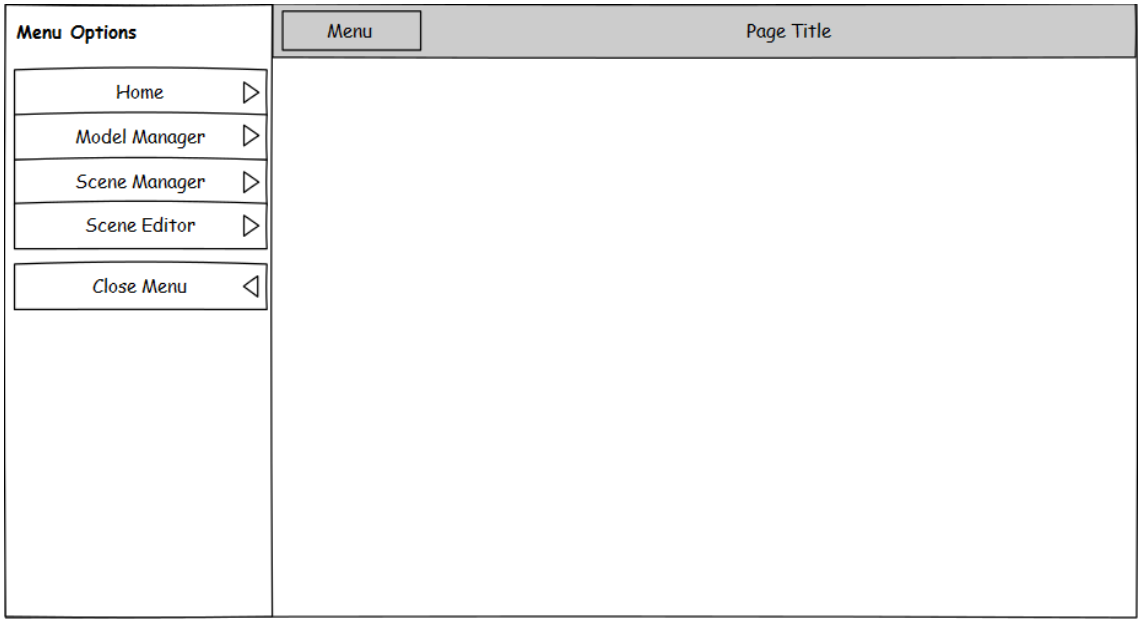


ILUSTRACIÓN 19 - PLANTILLA Y MENÚ PRINCIPAL

En la ilustración se puede visualizar una barra de título con un botón que permite el despliegue de un panel lateral que lista las diversas secciones en las que se subdivide el sistema de edición.

Este panel lateral aparece desde la izquierda desplazando el resto de la interfaz hacia la derecha, de manera que la parte situada a la derecha de la interfaz queda inaccesible al usuario cuando despliega el menú.

Se ha optado por este diseño teniendo en cuenta que el sistema debe ser accesible desde dispositivos táctiles de tamaño medio como pueden ser las *tablets* además de desde ordenadores de sobremesa y portátiles con tamaños de pantalla superiores. Esto influyó a la hora de decidir que el menú no estuviera visible permanentemente para tener así mayor espacio para el contenido. Esta necesidad es especialmente importante en dispositivos táctiles debido a la baja precisión de los toques sobre la pantalla, lo que se puede resolver aumentando el tamaño de los elementos interactivos como los botones y elementos de menús.

4.5.2. Gestor de modelos

A continuación se muestra la interfaz del sistema que permite la gestión de los modelos 3D.

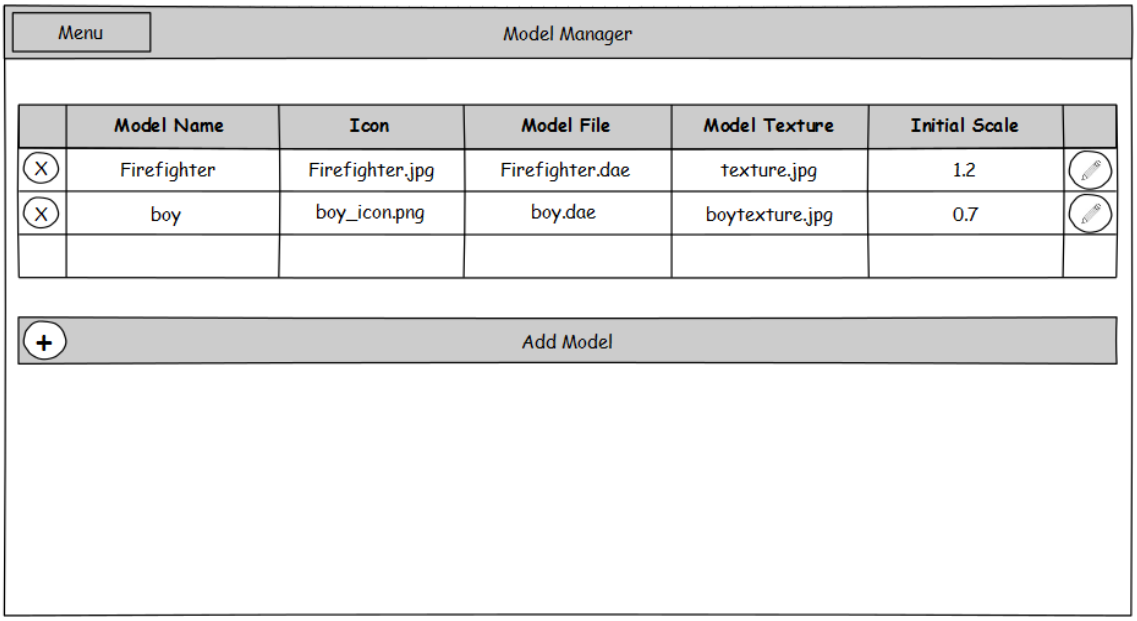


ILUSTRACIÓN 20 - GESTOR DE MODELOS

Esta interfaz muestra los modelos presentes en el sistema en una tabla especificando el nombre, la escala inicial y los nombres de los archivos asociados (icono, modelo y textura).

Se disponen de dos botones por cada modelo presente en el sistema, separados los máximo posible (uno a cada lado de la fila de la tabla) para evitar las posibles activaciones accidentales si estuvieran juntos. El botón de la izquierda permite la eliminación del modelo mientras que el de la derecha permite la modificación de sus campos. Adicionalmente se presenta un botón bajo la tabla cuya finalidad es la inserción de un modelo en el sistema. Cabe destacar que la eliminación debe realizarse con confirmación por parte del usuario.

Tanto la modificación de un modelo como la inserción se realizan a través de una ventana emergente o *popup* posicionado sobre la interfaz actual de manera modal, es decir, que no permite la interacción con la interfaz subyacente hasta que se cancele el *popup* o se complete el formulario que presenta. Los *popups* utilizados son los siguientes:

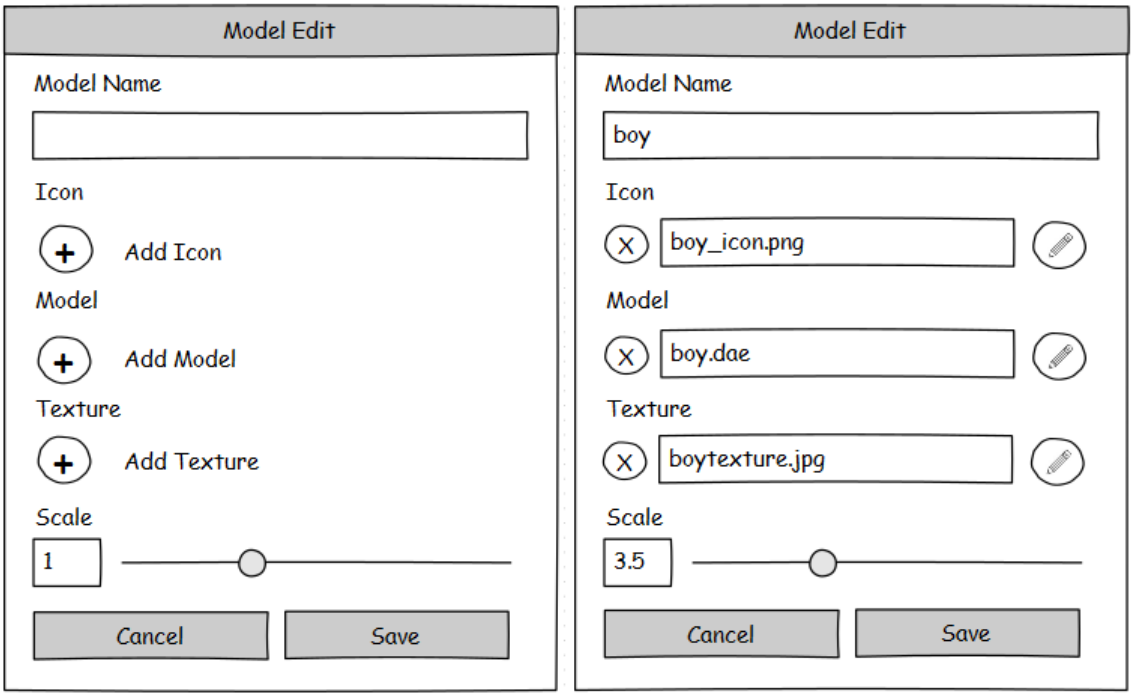


ILUSTRACIÓN 21 - GESTOR DE MODELOS – POPUPS

La ilustración muestra a la izquierda el *popup* de inserción de modelos, que se diferencia del de modificación a la derecha en que los campos se encuentran con valores por defecto o vacíos.

El nombre se introducirá por medio de un teclado (virtual o físico), la escala se podrá introducir tanto por medio de un teclado como utilizando un *slider* o deslizador. Para la especificación de los ficheros de iconos, modelos y texturas se accionará la selección de fichero que dependerá del sistema operativo subyacente.

4.5.3. Gestor de escenarios

A continuación se muestra la interfaz que permite la gestión de los escenarios y sus campos principales.

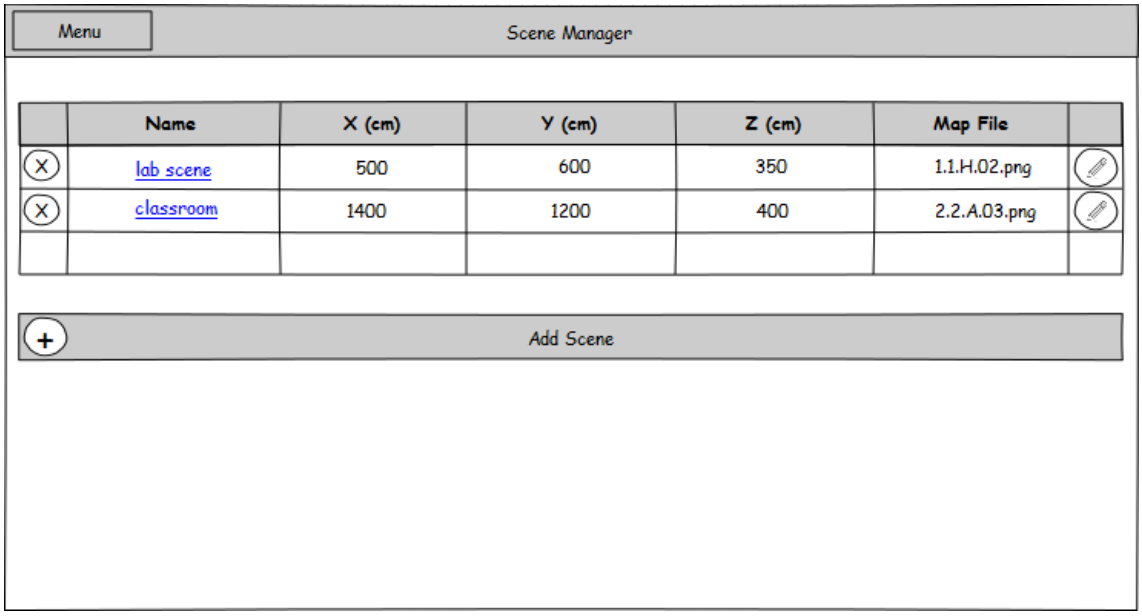


ILUSTRACIÓN 22 - GESTOR DE ESCENARIOS

Esta interfaz muestra los escenarios presentes en el sistema en una tabla especificando el nombre, las medidas de las tres dimensiones en centímetros escala el nombre del fichero de plano del escenario.

De la misma manera que con en la interfaz de gestión de modelos se disponen de dos botones por cada escenario presente en el sistema. El botón de la izquierda permite la eliminación del escenario mientras que el de la derecha permite la modificación de sus campos. Adicionalmente se presenta un botón bajo la tabla cuya finalidad es la inserción de un modelo en el sistema. Cabe destacar que la eliminación debe realizarse con confirmación por parte del usuario.

El nombre del escenario actúa a modo de enlace a la sección de edición de escenarios seleccionando ese escenario concreto.

Tanto la modificación de un escenario como la inserción se realizan a través de un *popup* posicionado sobre la interfaz actual de manera modal. Los *popups* utilizados son los siguientes:

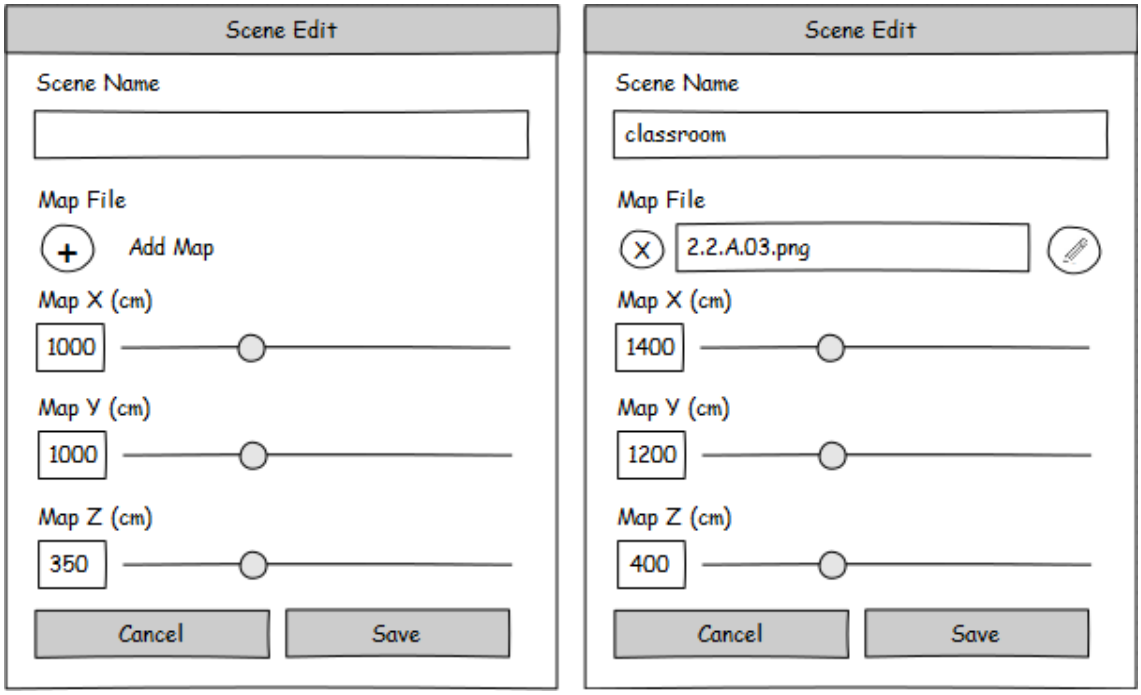


ILUSTRACIÓN 23 – GESTOR DE ESCENARIOS – POPUPS

La ilustración muestra a la izquierda el *popup* de inserción de escenarios, que se diferencia del de modificación a la derecha en que los campos se encuentran con valores por defecto o vacíos.

El nombre se introducirá por medio de un teclado (virtual o físico), las dimensiones se podrán introducir tanto por medio de un teclado como utilizando los *sliders*. Para la especificación del fichero del plano del escenario se accionará la selección de fichero que dependerá del sistema operativo subyacente.

4.5.4. Editor de escenario

A continuación se muestra la interfaz que permite el posicionamiento de los elementos virtuales y marcadores en el escenario sobre el plano del escenario.

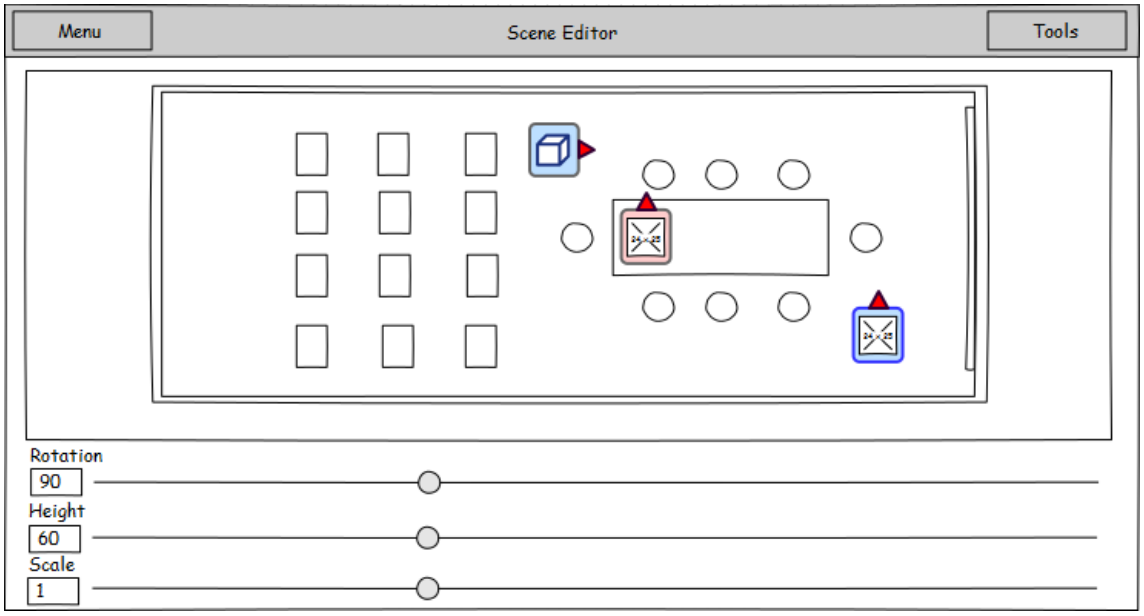


ILUSTRACIÓN 24 - EDITOR DE ESCENARIO

La interfaz presenta una zona de manipulación de elementos virtuales y marcadores en el parte superior y unos *sliders* en la parte inferior. En la zona de manipulación se muestra el plano del escenario como imagen de fondo de manera que se pueda añadir encima los elementos virtuales y marcadores del escenario a editar. Los elementos añadidos al mapa podrán seleccionarse por medio de *click* (o *tap* en caso de sistemas táctiles) ofreciendo una representación distinta, como por ejemplo un bordeado de distinto color.

Los elementos posicionados en el escenario indican la orientación por medio de una flecha y el modelo que usan mostrando el icono asociado al modelo. En el caso de ser marcadores se utiliza el icono del tipo de marcador usado.

Dado de que un plano la altura no se puede apreciar por la posición, se utiliza un gradiente de color para la especificación de la altura.

La modificación de la posición de un elemento o marcador se realiza seleccionándolo y utilizando *drag and drop* acto seguido. Por otro lado, la modificación del resto de propiedades como la orientación, altura ya escala se lleva a cabo a través de los *sliders* presentes en la parte inferior.

Esta interfaz además incluye un panel lateral al lado derecho que proporciona las operaciones de inserción de un nuevo elemento o marcador sobre el plano que se muestra del mismo modo que se muestra el menú principal, utilizando el correspondiente botón en la barra de título y es requerido por motivos de necesidad de espacio para contenido.

Además, el panel ofrecerá la opción de borrado de elementos (marcadores o elementos virtuales) seleccionados por el usuario.

A continuación se presentan los formularios presentados modalmente en forma de *popups* para añadir elementos virtuales y marcadores respectivamente junto con el panel lateral:

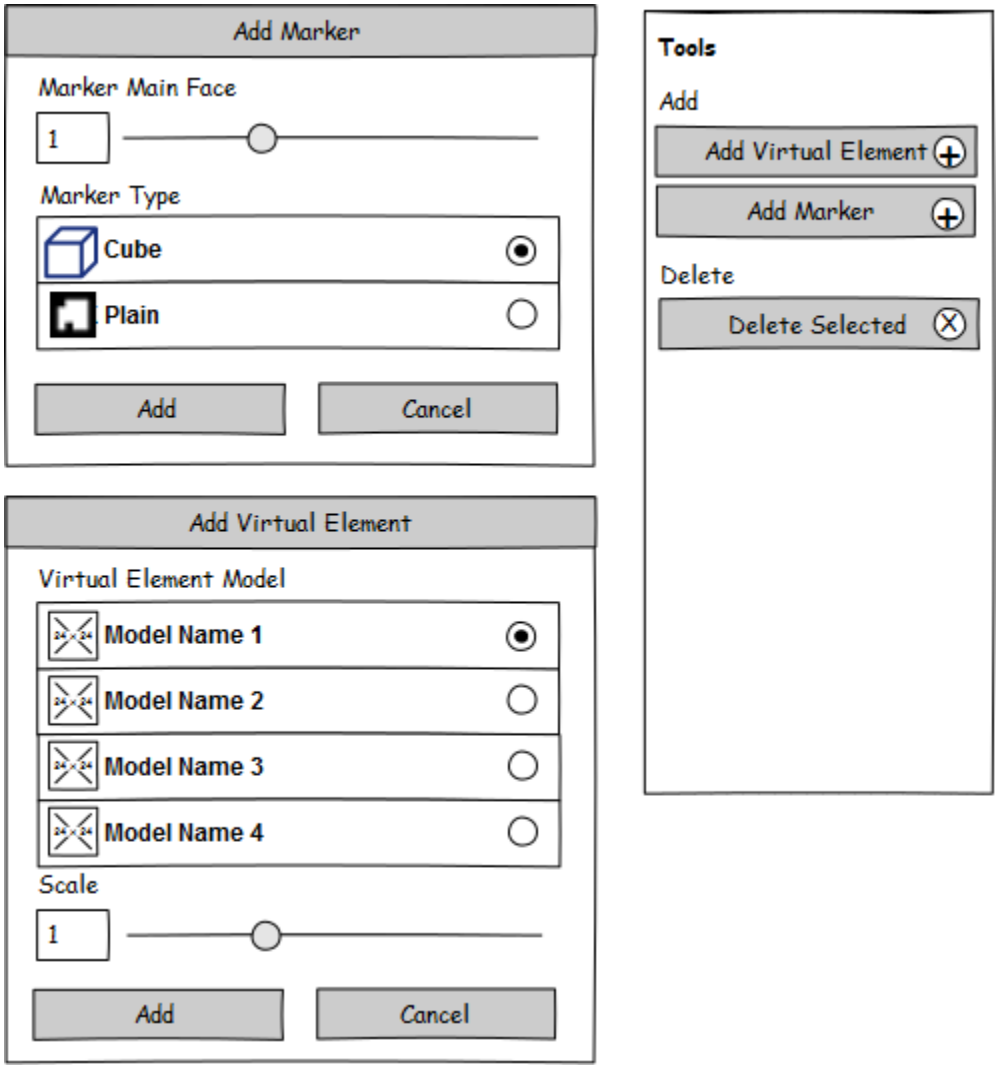


ILUSTRACIÓN 25 - EDITOR DE ESCENARIOS – POPUPS Y PANEL LATERAL

La ventana emergente que permite añadir marcadores requerirá la selección de un número mediante el uso de un *slider* o introduciéndolo por teclado. Este número pedido corresponde 1con el número decodificado del marcador. En el caso de marcadores compuestos de varios códigos como el cúbico, se especifica el número asociado a la cara principal, que en este caso sería la cara superior.

Adicionalmente se requiere la selección del tipo marcador, cúbico o plano antes de habilitar la activación del botón de añadir.

Asimismo, el *popup* de inserción de elementos virtuales también requerirá la selección del tipo de elemento virtual ofreciendo utilizar los modelos presentes en el sistema. Para ello se muestra el icono asociado con el modelo al lado de su nombre y se permite la selección de un solo modelo. Por último se ofrece la posibilidad de modificar la escala del elemento virtual a añadir a través de otro *slider*.

5. Implementación

En este capítulo se describirá en detalle la implementación del proyecto especificando las distintas tecnologías y herramientas utilizadas y los algoritmos más importantes empleados ofreciendo así una visión del funcionamiento de cada sistema.

Para la elección de las tecnologías y herramientas de desarrollo se ha tenido en especial consideración el uso de software libre frente al software propietario, principalmente por sus beneficios económicos (se evita el pago de licencias) y de seguridad y adaptabilidad (la publicación del código fuente evita la mala práctica de la seguridad por oscuridad y permite la exploración y modificación del código para adaptarse a los requisitos del proyecto).

Otro de los principales criterios de selección de tecnologías han sido los requisitos materiales del sistema, como la velocidad de procesamiento y memoria principal. El objetivo ha sido la minimización de la necesidad de estos requisitos de sistema debido a la baja disponibilidad de ellos en los dispositivos portátiles.

Por último, a la hora de seleccionar tecnologías y herramientas se ha primado la portabilidad de ellas a otros sistemas operativos centrándose principalmente en Linux y Windows. Las versiones de software a continuación especificadas corresponden con las versiones para Windows 8, el sistema operativo finalmente utilizado en el dispositivo portátil.

A continuación se desglosa la implementación de cada subsistema, incluyendo el motor de persistencia común a los sistemas de edición de escenarios, de visualización de escenarios y partidas y de conexión.

5.1. Persistencia y servidor web

La persistencia del proyecto ha sido tratada a través del sistema de gestión de bases de datos relacional MySQL (versión 5.6.14). El puerto de escucha utilizado por MySQL por este servicio es el 3306.

Para la fácil exploración de la base de datos se ha utilizado la herramienta web phpMyAdmin (versión 4.0.9), que ofrece por medio de una interfaz gráfica, entre otras funcionalidades, la generación de consultas SQL, la visualización de las tablas y el diseño de la base de datos sin necesidad de utilizar la consola de comandos.

Debido a la necesidad de utilizar un servidor web para el uso de phpMyAdmin y para los sistemas de edición de escenarios y de conexión (como se detallará a continuación), se ha utilizado un servidor Apache con el puerto de escucha 80.

Todas las herramientas expuestas se encuentran convenientemente disponibles a través de la distribución de Apache gratuita XAMPP (versión 1.8.3). Esta distribución tiene como objetivo, entre otros, la fácil instalación de un servidor para el desarrollo web en PHP con soporte de múltiples sistemas operativos como Windows, Linux y Mac OS X.

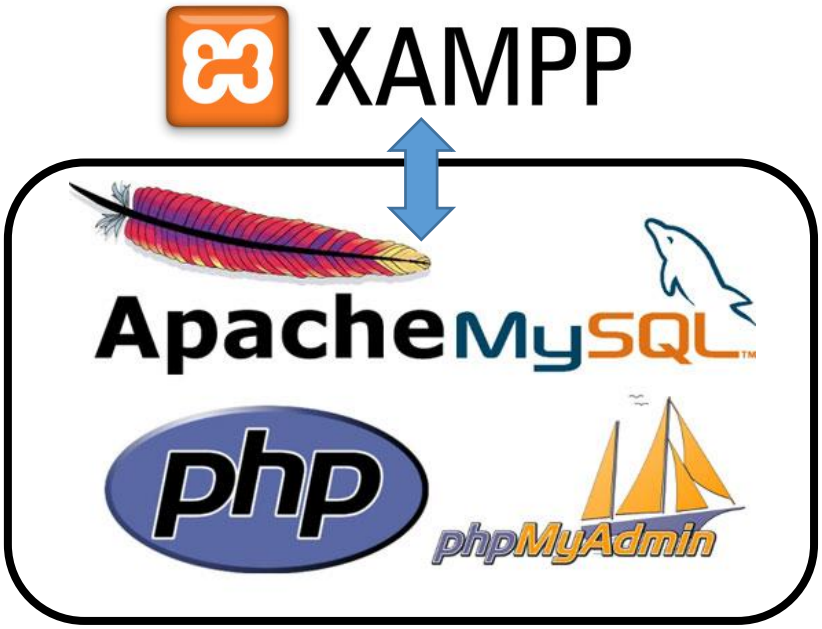


ILUSTRACIÓN 26 - DIAGRAMA XAMPP

Con la instalación de XAMPP y por medio del uso de su panel de control se pueden configurar las herramientas anteriormente mencionadas.

5.2. Sistema de edición de escenarios

El sistema de edición de escenarios fue diseñado con los requerimientos de ser multiplataforma y portátil, de forma que además de ser accesible desde un dispositivo táctil de tamaño medio sea posible su uso desde un ordenador de sobremesa o portátil.

Debido a estos requisitos, el sistema ha sido implementado como una aplicación web, lo que permite el acceso al sistema desde navegadores web en múltiples sistemas operativos tanto móviles como de escritorio.

A continuación se detallan las tecnologías utilizadas para el desarrollo de aplicación, tanto en el lado del servidor web como en el cliente.

5.2.1. Servidor web: PHP

Para el desarrollo de parte del servidor de la aplicación web se optó por el uso de PHP (versión 5.5.6) por su facilidad de aprendizaje, agilidad de implementación y alto rendimiento bajo la disponibilidad de pocos recursos.

Se consideró seriamente el uso de Java EE, pero se acabó prefiriendo PHP debido a no requerir una máquina virtual con el fin de reducir la necesidad de recursos. Esto corresponde con el requisito de portabilidad y la escasa disponibilidad de recursos en los dispositivos portátiles.

5.2.2. Cliente web: HTML5, CSS3 y JavaScript

Por el lado del cliente se utilizó la combinación de HTML5, CSS3 y JavaScript.

ILUSTRACIÓN 27 - HTML5, CSS3 Y JAVASCRIPT

Dada la necesidad de posicionar elementos sobre un plano y gracias a las posibilidades que ofrece la nueva etiqueta <canvas> presente en HTML5, se decidió utilizar esta versión de HTML para el desarrollo del sistema de edición de escenarios. Otras alternativas pasan por la utilización de Adobe Flash o la propuesta de Microsoft para la sustitución de Flash: Microsoft Silverlight. A pesar de no estar consolidado el estándar de HTML5, su desarrollo está en auge y los principales navegadores web están centrando sus esfuerzos en su suporte. Por contra, Adobe Flash ha dejado de tener soporte y Silverlight no cumple con los requerimientos de multiplataforma.

El estilo fue gestionado con CSS3, que permitió el diseño fluido de la interfaz de manera que se adaptara a distintos tamaños de pantalla. Al mismo tiempo evitó el uso de tablas para la disposición de los elementos de la interfaz.

Para el diseño de los distintos elementos de la interfaz gráfica como botones, menús, paneles y listas se recurrió a la librería JavaScript jQuery Mobile (versión 1.4.2), la cual proporciona una potenciación de las etiquetas html ampliando sus funcionalidades y ofreciendo un estilo

optimizado para la interacción táctil en pantallas de diversos tamaños. Esta librería ofrece asimismo un sistema de navegación basado en llamadas asíncronas AJAX y ofrece el almacenamiento de las páginas visitadas en caché para una rápida transición entre páginas visitadas.

jQuery Mobile hace uso a su vez de la librería jQuery (se empleó la versión 2.1.0), que consiste en una librería de JavaScript que ofrece una API de manipulación, gestión de eventos, animación y uso de llamadas asíncronas con soporte multiplataforma.

La utilización del *canvas* de HTML5 se apoyó en una librería de JavaScript denominada KineticJS (versión 5.1.0), que ofrece la anidación de nodos, su distribución en grupos y capas y la escucha de eventos asociados a ellos. En particular posibilita la rotación, escalado y

desplazamiento de nodos sobre el *canvas*.

ILUSTRACIÓN 28 - LIBRERÍAS JAVASCRIPT

5.2.3. Prototipo de interfaz

En este apartado se muestran capturas de las tres principales páginas del prototipo final que componen la aplicación web asociadas con:

- La gestión de los modelos 3D del sistema (*Model Manager*).
- La gestión de escenarios del sistema (*Scene Manager*).
- La edición de un escenario concreto (*Scene Editor*).

5.2.4. Model Manager

En esta página se listan los modelos 3D presentes en el sistema y se proporciona la posibilidad de añadir nuevos modelos, eliminar modelos existentes y editar las características de un modelo concreto.

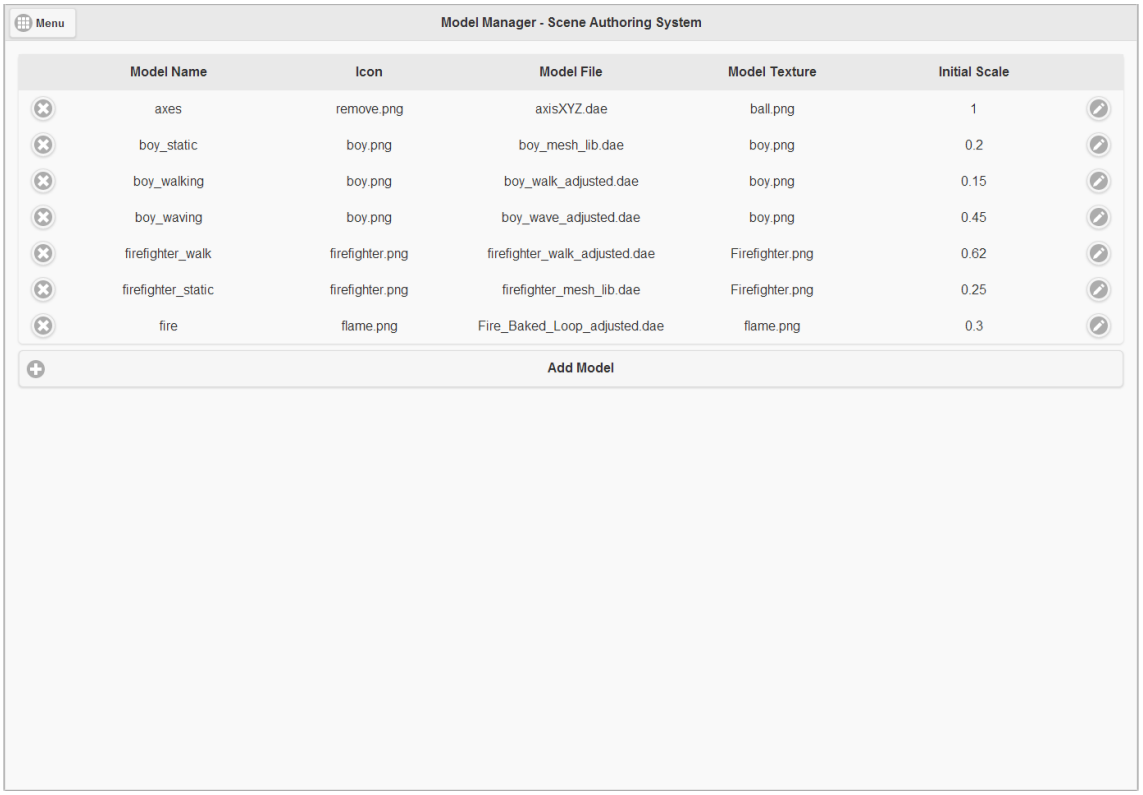


ILUSTRACIÓN 29 - PROTOTIPO DE GESTOR DE MODELOS

Como se puede ver en la ilustración, el contenido de la página consiste en una representación en formato tabla que lista los modelos actualmente presentes indicando los valores de sus campos.

Al lado izquierdo de cada modelo se ofrece un botón de eliminación, el cual pide confirmación al usuario, mientras que al derecho se ofrece un botón de edición de los atributos del modelo.

A su vez en inmediatamente debajo de la tabla se ofrece la posibilidad de añadir un modelo nuevo.

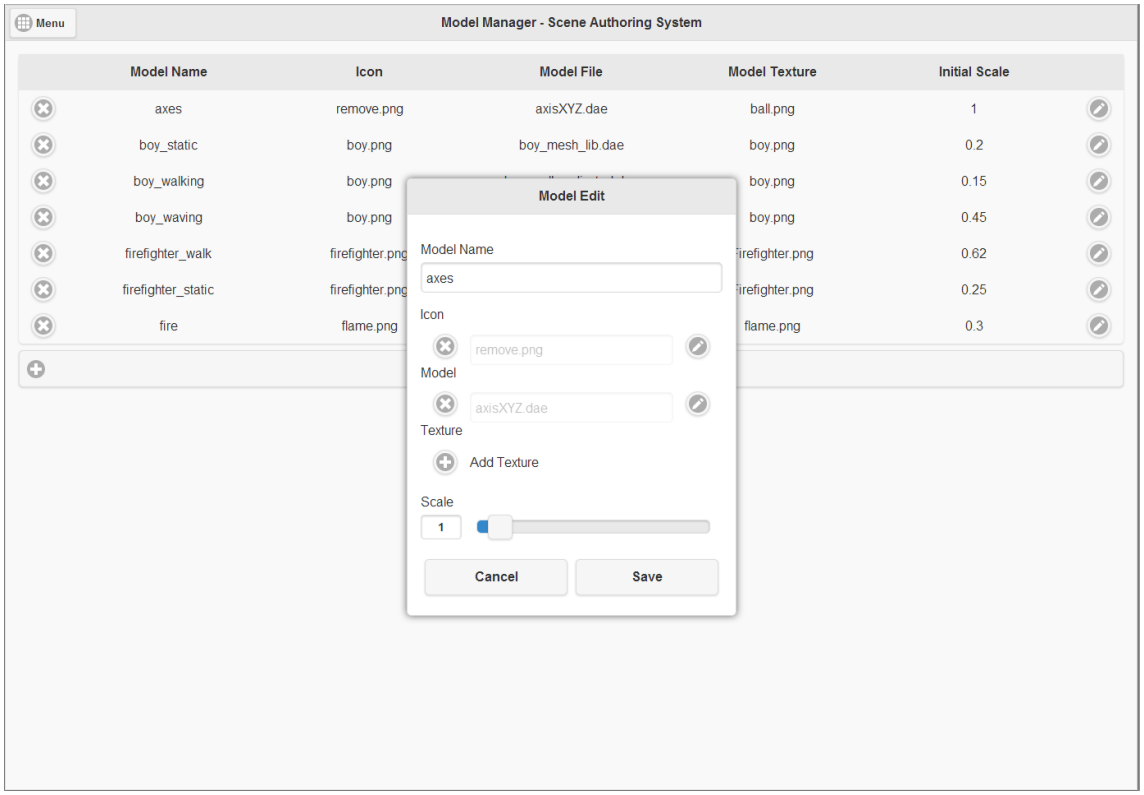


ILUSTRACIÓN 30 - PROTOTIPO DE GESTOR DE MODELOS – POPUP

Tras la selección tanto de la opción de editar como de añadir se muestra un *popup* modal con campos de entrada para los atributos del modelo. En el caso de la edición los campos se encontrarán rellenos mientras que en el caso de la inserción de modelo deberán ser cumplimentados por el usuario.

En particular se pide el nombre del modelo con un campo de texto, los ficheros de icono, modelo 3D y textura por medio de botones de selección de ficheros y el factor de escala requerido por el modelo para su normalización con un *slider*.

Para finalizar se ofrece la confirmación de los cambios por medio de un botón de guardado y la cancelación sin modificación sobre el modelo tratado.

5.2.5. Scene Manager

En esta página se listan los escenarios presentes en el sistema y se proporciona la posibilidad de añadir nuevos escenarios, eliminar escenarios existentes y editar las propiedades de un escenario concreto además de seleccionar el escenario para su edición con la página de edición de escenarios (*Scene Editor*).

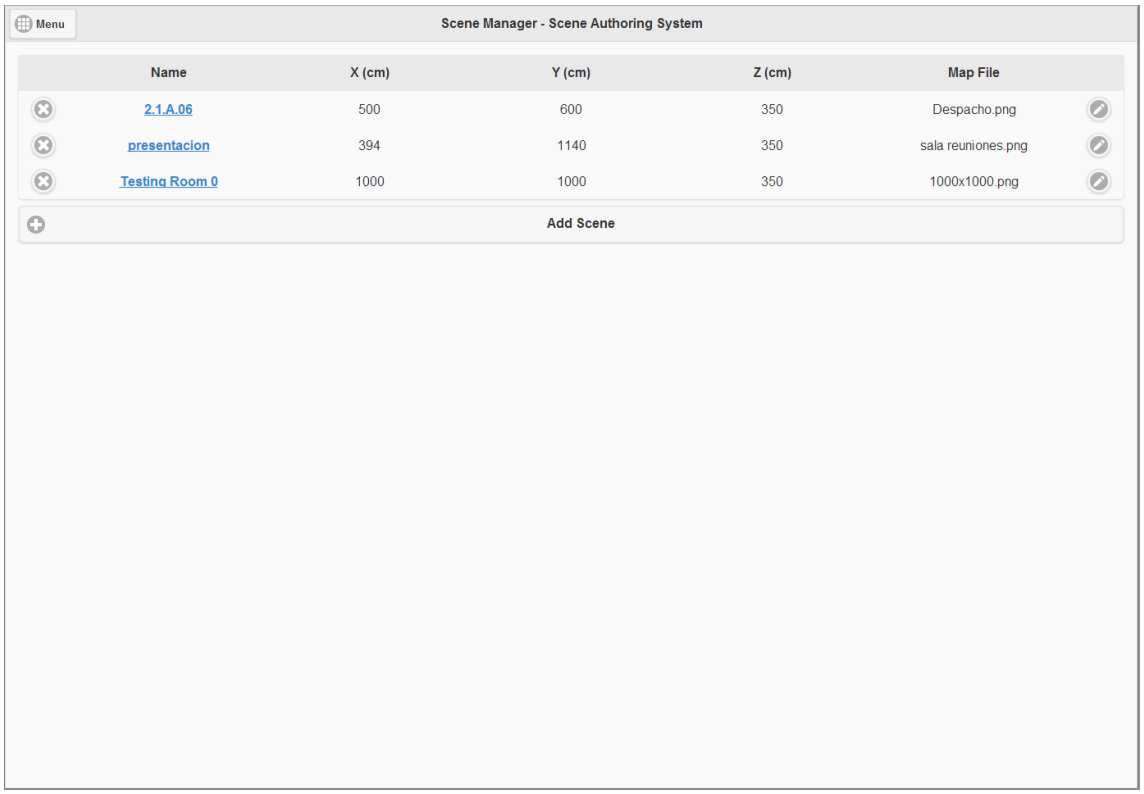


ILUSTRACIÓN 31 - PROTOTIPO DE GESTOR DE ESCENARIOS

La interfaz de esta página sigue la misma estructura que la del gestor de modelos: el contenido es principalmente una tabla en la que cada file muestra un escenario presente en el sistema, ofreciendo a los lados las opciones de eliminación o edición mientras que bajo la tabla se ofrece la inserción de nuevos escenarios.

Cabe destacar que se ofrece un enlace a través del nombre del escenario, el cual redirecciona al editor de escenarios para la edición de los elementos presentes en ese escenario concreto.

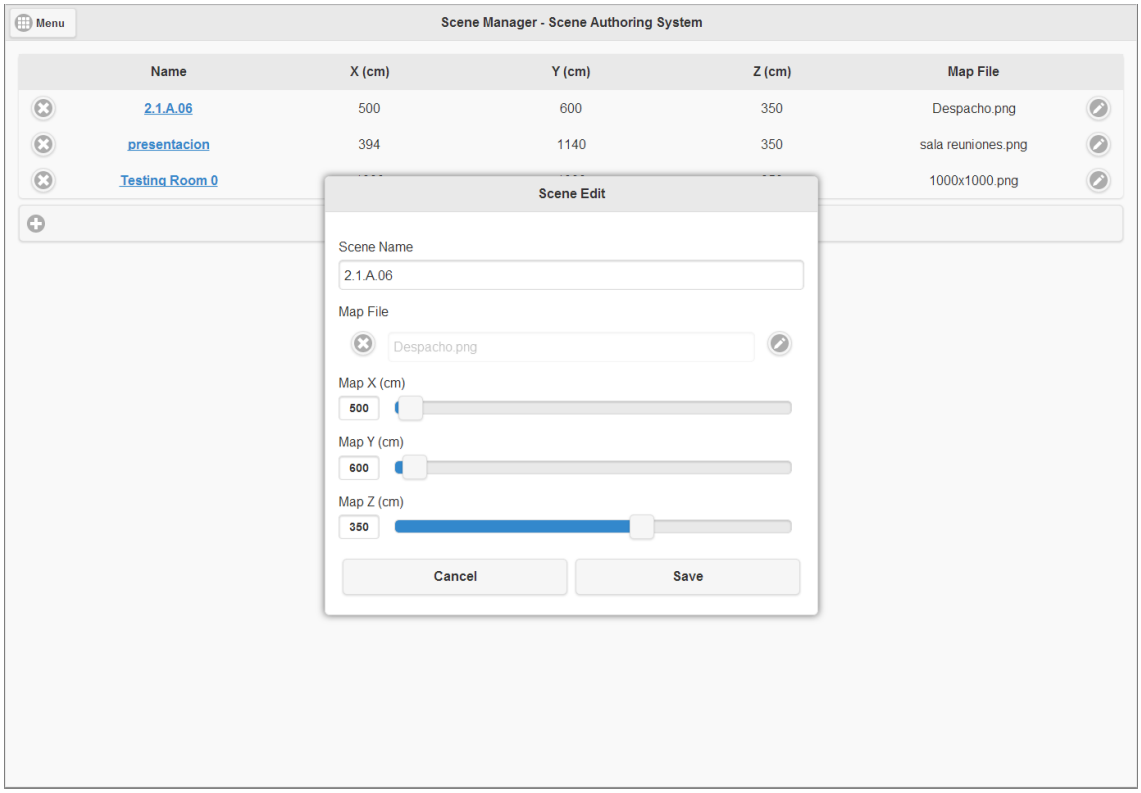


ILUSTRACIÓN 32 - PROTOTIPO DE GESTOR DE ESCENARIOS – POPUP

Del mismo modo que en el gestor de modelos, al seleccionar las opciones de inserción o de edición de un modelo por medio de los botones correspondientes se presenta modalmente el *popup* que permite la introducción de los distintos campos del escenario.

En particular se utiliza un campo de texto para la introducción del nombre, un selector de ficheros para incluir el plano del escenario a utilizar y tres *sliders*, uno para cada una de las tres dimensiones que caracterizan al escenario.

De la misma manera se ofrecen las opciones de cancelación y guardado.

5.2.6. Scene Editor

Tras la selección de un escenario a editar se redirecciona a la página de edición de escenarios, la cual permite la manipulación de los elementos virtuales de un escenario así como de los marcadores que servirán para la referencia sobre el mundo virtual del visualizador.

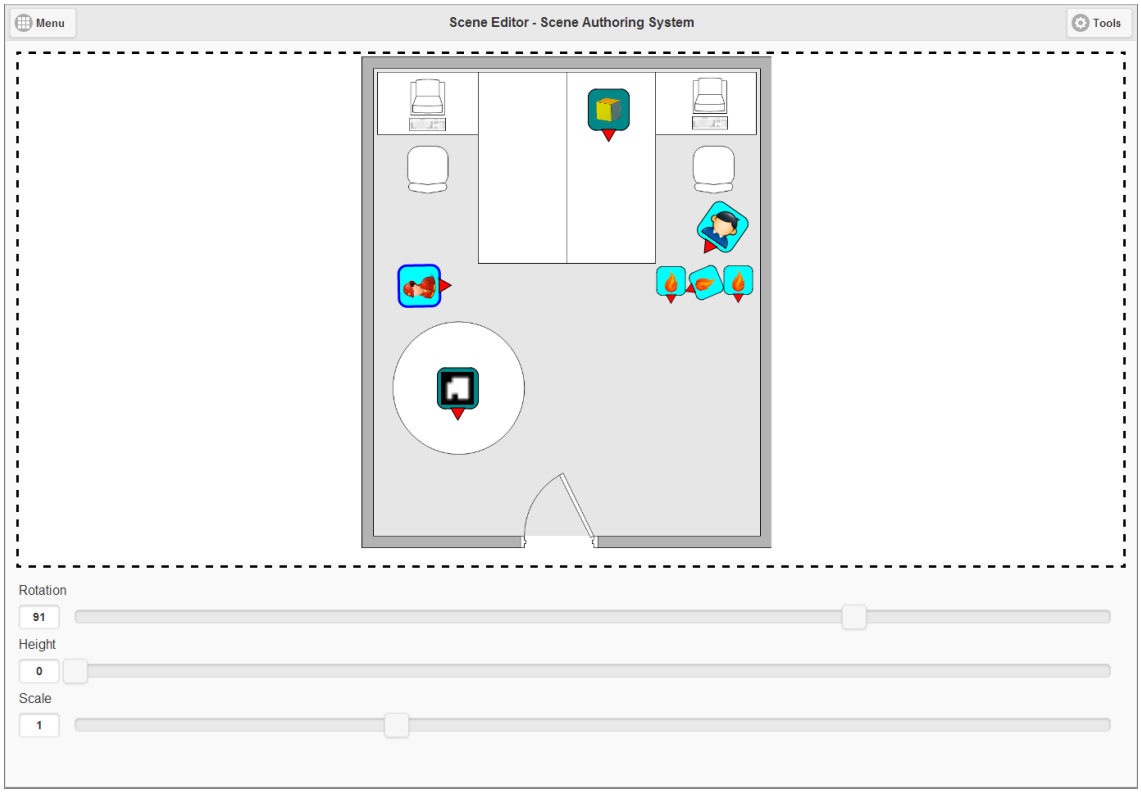


ILUSTRACIÓN 33 - PROTOTIPO DE EDITOR DE ESCENARIOS

La interfaz se encuentra ocupada en prácticamente su totalidad por *canvas* en el que se presenta el plano del escenario y sobre el cual añaden y manipulan los elementos virtuales y marcadores.

Los elementos virtuales y los marcadores son representados por nodos con un indicador de orientación, el icono asociado al modelo 3D o icono del tipo de marcador. Estos nodos utilizan un color de fondo que sirve de indicador de la altura a la que se encuentran desde el suelo y que utiliza un gradiente de color.

A la vez, se permite la selección de los elementos virtuales y marcadores, los cuales son visibles por medio de un bordeado de color azul.

La posición de estos elementos sobre el plano se modifica seleccionando y arrastrando los nodos de manera que al dejar de arrastrar se produce una llamada asíncrona por medio de AJAX, que proporciona al servidor las nuevas coordenadas para su persistencia.

Asimismo, la orientación, altura y escalado de los elementos virtuales y marcadores es modificable por medio del uso de tres *sliders*. Al modificar uno de estos *sliders* se modifican las propiedades de todos los elementos seleccionados enviándose al servidor a través de llamadas asíncronas de AJAX los nuevos valores.

La función de los *sliders* no se acaba en la modificación. Estos sirven a su vez para determinar los valores concretos de orientación, altura y escalado de elemento seleccionado.

La utilización de llamadas asíncronas evita la recarga de la página y la consulta periódica a la base de datos de las posiciones de los elementos del escenario. Esto permite la actualización en tiempo real de la base de datos para la visualización simultánea del escenario a través del sistema de visualización.

Cabe destacar que debido a la posibilidad de un plano de gran tamaño y/o el uso de un dispositivo de pantalla reducida, el plano podría no ser abarcado en su totalidad por el *canvas* de manera que se mostraría solamente una porción de este. Para solucionar el problema de no poder acceder determinadas partes del plano se ofrece el arrastre de este, de manera que el *canvas* puede situarse en cualquier parte del plano.

Por último se ofrece un botón de herramientas específico para esta página en la parte superior derecha de la interfaz, que despliega un panel lateral con las distintas acciones sobre el escenario que se ofrecen.

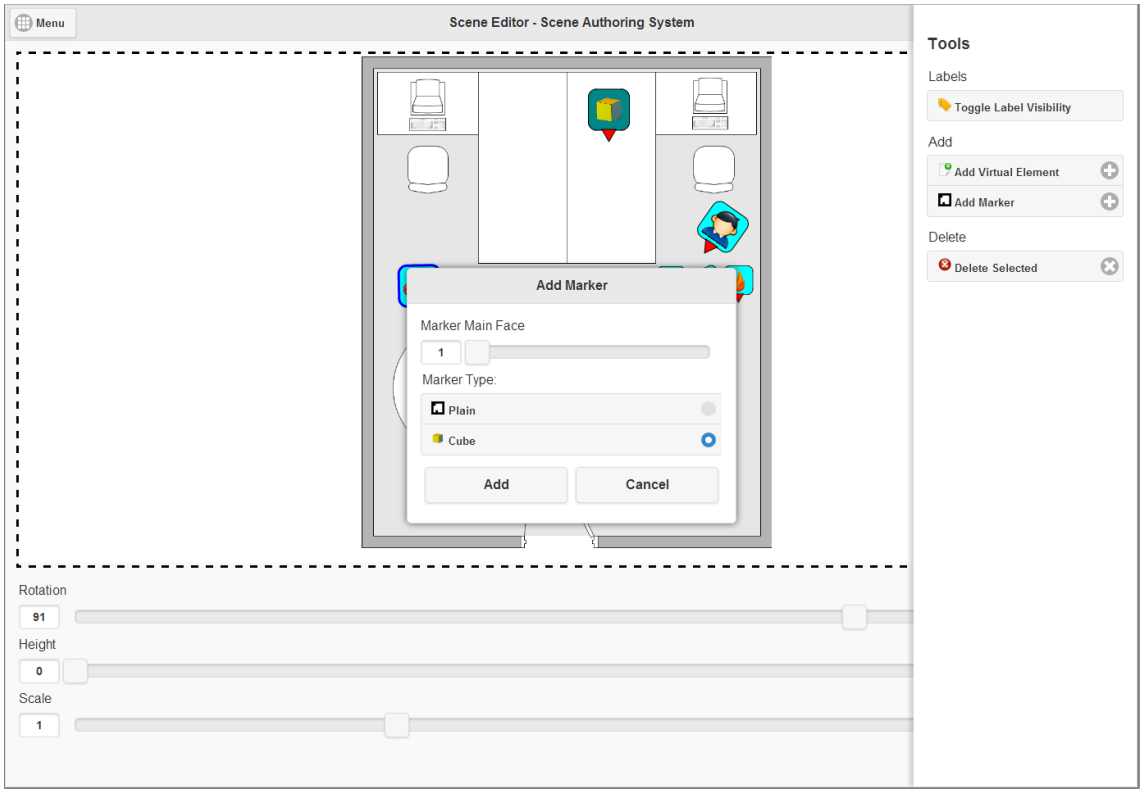


ILUSTRACIÓN 34 - PROTOTIPO DE EDITOR DE ESCENARIOS - INSERCIÓN DE MARCADOR

Las acciones disponibles sobre el escenario son la posibilidad de visualizar etiquetas con información de cada nodo, la inserción de un elemento virtual al escenario, la inserción de un marcador al escenario y el borrado de marcadores y/o elementos virtuales del escenario previa selección de estos.

En la ilustración anterior se muestra el *popup* que emerge al seleccionar la opción de inserción de marcadores y pide al usuario la introducción del código de la cara principal del marcador que se usará, además de la selección del tipo de marcador: marcador de plano (de una sola cara) o marcador cúbico (con seis caras, cada una de ellas con un marcador).

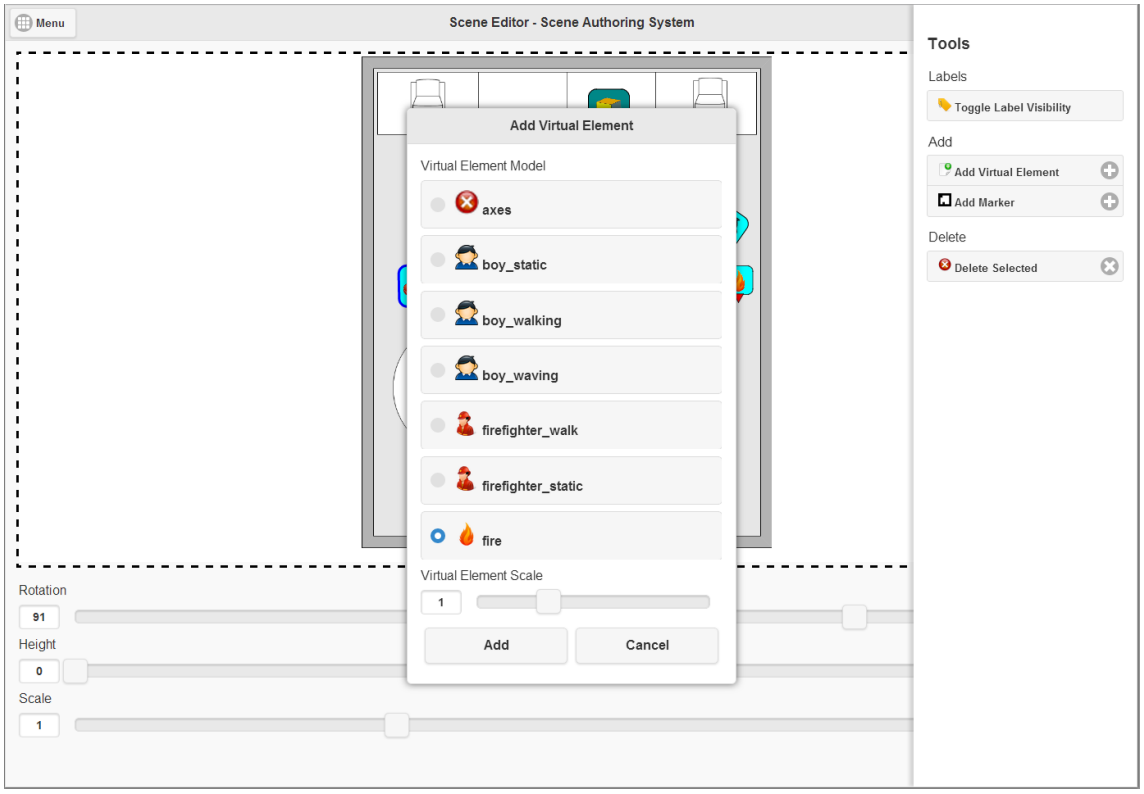


ILUSTRACIÓN 35 - PROTOTIPO DE EDITOR DE ESCENARIOS - INSERCIÓN DE ELEMENTO VIRTUAL

Por otro lado, la opción de inserción de elemento virtual ofrece al usuario una lista de los modelos disponibles entre los que seleccionar el tipo de elemento virtual y la escala a aplicar al elemento virtual. Una vez el elemento es creado y añadido al plano se puede modificar su posición, orientación, altura y escala manipulando el nodo en el *canvas*. Esto ocurre también para los marcadores.

5.3. Sistema de visualización de escenarios y partidas

El sistema de visualización de escenarios y partidas ha sido implementado haciendo uso del entorno de desarrollo openFrameworks. Esta es una herramienta de código abierto escrita en C++ distribuida bajo la licencia MIT. Esta licencia permite el uso de openFrameworks manteniendo la libertad de su utilización para fines comerciales y sin la obligación de la publicación del código. Además, openFrameworks ofrece la encapsulación de diversas librerías gráficas, de audio, de captura y reproducción de vídeo, de visión artificial y de renderización de modelos 3D bajo una sencilla API de alto nivel. A su vez, la herramienta está codificada para ser compatible con múltiples sistemas operativos, en concreto Windows, Mac OS X, Linux, iOS y

Android y diversos entornos de desarrollo integrados o *IDEs*, como XCode, Code::Blocks, Visual Studio y Eclipse.

El sistema a implementar requería el procesamiento de imagen con visión artificial y la renderización de modelos 3D manteniendo un alto grado de portabilidad entre sistemas operativos. Asimismo, el desarrollo del proyecto debía cumplir con unos plazos de implementación exigentes que no permitían el entrenamiento del equipo de desarrollo en el uso de librerías gráficas de bajo nivel como OpenGL. Debido a estos requisitos y dado que openFrameworks cumple con todos ellos ofreciendo una abstracción a un nivel elevado con una corta curva de aprendizaje, se optó por su utilización.

En particular, de las múltiples librerías encapsuladas se hace uso principalmente de dos: OpenCV para la utilización de visión artificial y Assimp para la carga de modelos 3D para su renderización.

Para la carga de ficheros 3D se decidió el uso del formato de fichero COLLADA 1.4 (extensión .dae). COLLADA es un estándar abierto de esquema XML diseñado para el intercambio de recursos digitales entre aplicaciones y su almacenamiento gestionado por el Grupo Khronos, el cual también gestiona el desarrollo de OpenGL.

Pese a la existencia de otros formatos para el modelado 3D como DirectX X (.x), 3ds Max 3DS (.3ds), Wavefront Object (.obj) y Blender 3D (.blend), se decidió el uso de COLLADA por su compatibilidad, por no necesitar software propietario para su generación y por soportar la animación de los modelos y el uso de texturas y materiales.

El software empleado para la generación de los modelos 3D fue Blender. Blender es un proyecto de código abierto que permite la generación de modelos 3D con formato .blend con soporte de animación, uso de texturas y materiales. La principal razón para su utilización al margen de por ser software libre, es que incluye módulos de exportación a otros formatos, entre ellos COLLADA, formato usado en el proyecto.

5.3.1. Diagrama de flujo

A continuación se presentará el diagrama de flujo del sistema de visualización de escenarios y partidas.

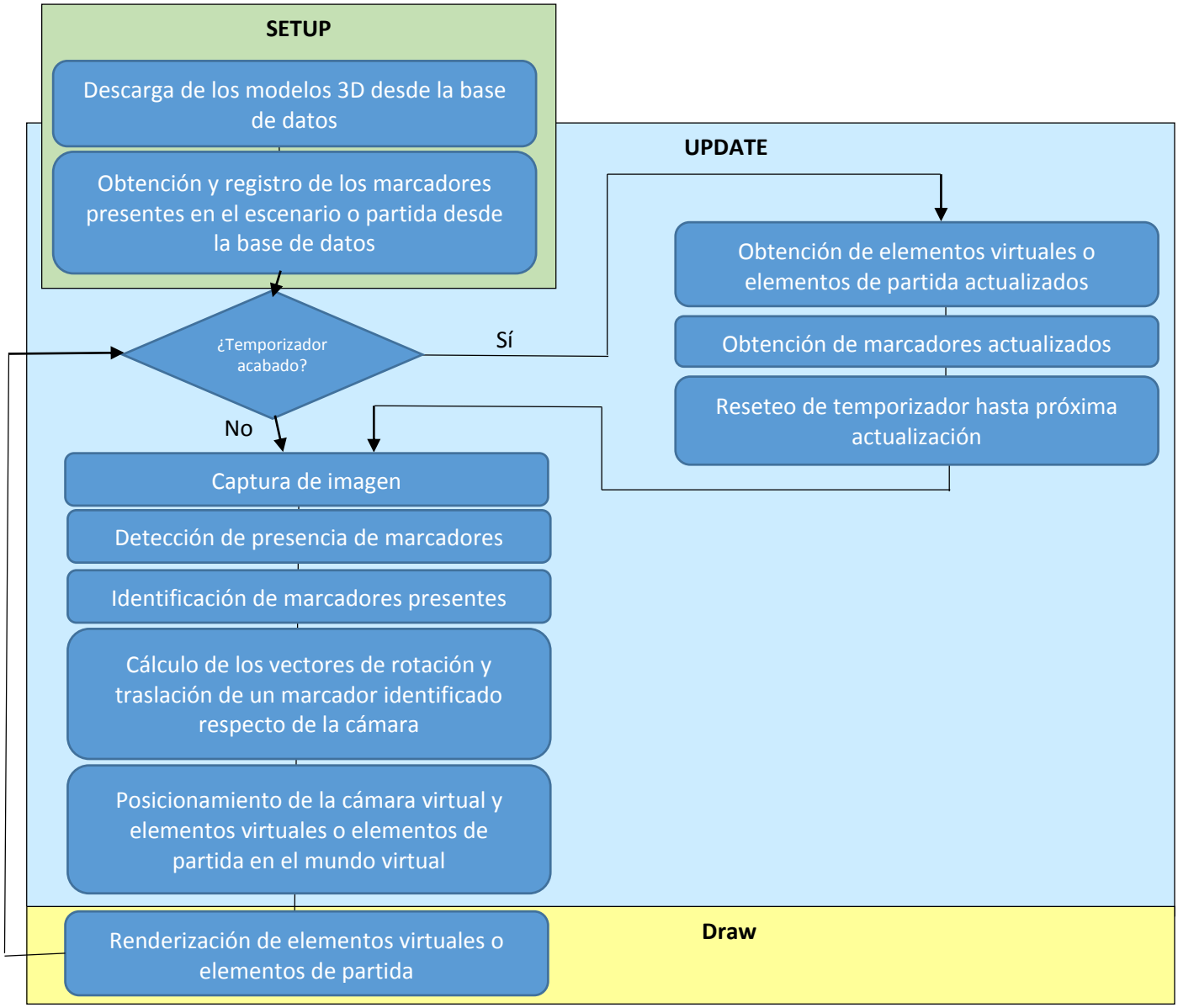


ILUSTRACIÓN 36 - FLUJO DE SISTEMA DE VISUALIZACIÓN

Como se observa en el diagrama, la ejecución consta de tres etapas: una etapa de configuración o *setup*, una etapa de actualización o *update* y una etapa de renderizado o *draw*.

La fase de configuración se ejecuta una vez, como fase inicial y se encarga de preparar los recursos necesarios para la visualización del escenario o la partida.

Tras la fase inicial se ejecuta indefinidamente un bucle que alterna las etapas de actualización y renderización, el cual termina cuando el usuario decide finalizar el programa.

La fase inmediatamente siguiente a la fase de configuración es la actualización y esta se encarga de mantener la información de los elementos virtuales y de la posición del usuario respecto del mundo virtual actualizada. Para ello se ejecutan periódicamente consultas a la base de datos para obtener la nueva información del escenario o partida a visualizar.

Una vez actualizada la información disponible desde la base de datos se procede con la actualización de la posición del usuario visualizador. Esto se lleva a cabo obteniendo imágenes desde la cámara y buscando marcadores presentes en la imagen con técnicas de visión artificial que se explicarán más adelante.

Una vez encontrados marcadores en la imagen capturada se pasa a su identificación, de manera que se comparan con los marcadores registrados en el escenario en busca de coincidencias. El objetivo es encontrar al menos un marcador en la imagen que esté presente en el escenario para poder así calcular la posición del usuario respecto del marcador y, sabiendo la posición del marcador, calcular la posición y orientación absolutas del usuario sobre el mundo virtual.

Una vez calculadas estas coordenadas y orientación se posiciona una cámara virtual en ese punto del espacio virtual, de manera que se obtiene una perspectiva de los elementos virtuales a visualizar desde el mundo virtual en el que estaría el usuario.

Los elementos virtuales a visualizar son entonces posicionados según la información obtenida de la base de datos concluyendo la fase de actualización: el mundo virtual gestionado por el sistema de visualización se encuentra en este momento actualizado respecto a la información proporcionada por la base de datos y la cámara que permite el posicionamiento del usuario.

Como última etapa del bucle se ejecuta la renderización. En esta etapa se realiza la representación gráfica de los elementos virtuales y la obtención de una captura de la imagen del mundo virtual desde la cámara virtual, es decir, la visión que tendría el usuario en el mundo

virtual. Esta imagen es la que se proyecta en el dispositivo de realidad aumentada, que se superpone a la visión del mundo real.

Hay que destacar que la imagen proyectada debería mostrar los elementos de virtuales ofreciendo un fondo transparente, sin embargo esto no es posible con la tecnología empleada y se ofrece un fondo negro. La tecnología del dispositivo de realidad aumentada utilizado es óptica (*optical see-through* en concreto) y la visión virtual superpuesta no bloquea totalmente la visión del mundo real sino que se ofrece con cierta semitransparencia. Esto por un lado es un inconveniente a la hora de visualizar los elementos virtuales, ya que cada pixel virtual se ve modificado por la imagen real que oculta. Por otro lado esto permite la visión del mundo real a través del fondo negro proyectado, siendo la visión humana capaz de acomodarse a esto de manera que el fondo es ignorando.

5.3.2. Reconocimiento de marcadores

En esta sección se detallará el diseño de los marcadores y el algoritmo de detección e identificación de estos.

Los marcadores utilizados en el sistema se utilizan para el posicionamiento del usuario visualizador en el mundo virtual. Esto se consigue en dos pasos: identificando dónde se encuentra el marcador y dónde se encuentra el usuario respecto del marcador.

La posición y orientación del marcador se registra en la base de datos a la hora de editar el escenario. Para ello cada marcador utilizado en el escenario debe ser distinto de cualquier otro. Es por esta razón que los marcadores se codifican un número.

Por otro lado la posición y orientación del usuario respecto del marcador se calcula aplicando cálculos de deformación debidos a la perspectiva.

5.3.2.1. Identificación y detección de marcadores

En esta sección se explican la codificación de los marcadores utilizados así como el proceso para la detección de estos desde una imagen capturada.

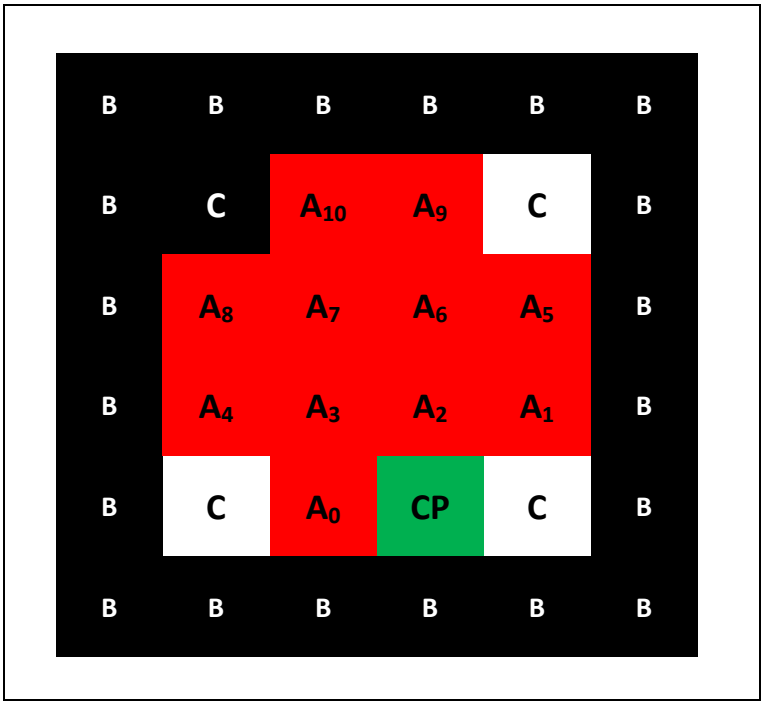


ILUSTRACIÓN 37 - DISEÑO DEL MARCADOR

El marcador está formado por un cuadrado de 14 centímetros de lado rodeado de un marco de color blanco que permita la detección de las esquinas de este cuadrado.

A su vez el marcador constituye una matriz de 6x6 celdas cuadradas. Estas celdas son blancas o negras (podrían haberse utilizado otros colores distintos siempre y cuando el contraste fuera elevado). Las celdas de las primeras y últimas filas y columnas (celdas B) forman un marco interior negro que contrasta con el marco blanco para la identificación de las esquinas.

A su vez, dentro del marco formado por las celdas B se utiliza un sistema de identificación de rotación: de las cuatro esquinas (celdas C), la superior izquierda se diferencia de las otras. Esto define una identificación de las demás celdas aunque el marcador obtenido se encuentre girado respecto al eje perpendicular al marcador.

Las celdas marcadas en rojo en el diagrama contienen la información del número codificado. Esta codificación corresponde a un número binario entero positivo de 11 bits. Cada una de las casillas rojas corresponde a un bit, siendo A_{10} el bit más significativo y A_0 el menos significativo, de manera que si la casilla es negra se considera el valor 1 mientras que si es blanca se considera

valor 0. Esto implica que la cantidad de números codificables en este tipo de marcadores es 2^{11}, del 0 al 2047, cantidad que supera con creces la cantidad de marcadores de posición presentes en un mismo escenario simultáneamente.

Finalmente se utiliza una celda para control de paridad, la celda CP. Esta paridad se comprueba sobre las celdas interiores al marco negro de la siguiente forma: la cantidad de celdas negras interiores al marco negro (celdas C, celdas A y CP) debe ser siempre un número par.

Ejemplo:

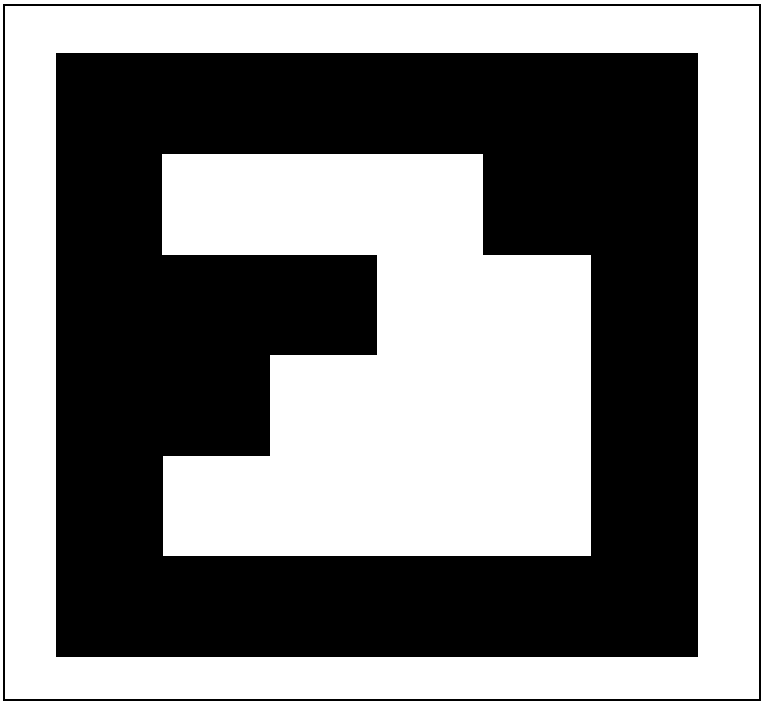


ILUSTRACIÓN 38 - EJEMPLO DE MARCADOR

Este marcador muestra un giro de 90º en el sentido de las agujas de reloj, deducido por la posición de la celda C.

Paralelamente, se puede observar cómo las celdas A_0 y A_1 están en negro, por lo que el número codificado es: $N = 2^0 + 2^3 = 1 + 8 = 9$.

Además, se comprueba la paridad, ya que la cantidad de celdas dentro del marco negro es par: una esquina, dos celdas A y, para que se cumpla esta condición, la celda CP.

Hasta aquí se ha explicado el diseño del marcador y el algoritmo de identificación, pero esto parte de la base de que se ha podido aislar desde una imagen capturada desde la cámara una matriz de 6x6 que representa el marcador.

A continuación se explica cómo se obtiene esta matriz de 6x6 en blanco y negro.

El proceso es el siguiente:

1. **Conversión de la imagen RGB a escala de grises**. Dada una imagen en RGB de 640x480 píxeles obtenida desde la cámara, se aplica una media ponderada para la obtención de la imagen en escala de grises equivalente. Los canales rojo, verde y azul no contribuyen por igual a la imagen en escala de grises, por lo que hay que aplicar un factor distinto para cada canal según la siguiente fórmula:

$$\begin{bmatrix} M_{1,1} & \dots & M_{1,640} \\ \vdots & \ddots & \vdots \\ M_{480,1} & \dots & M_{480,640} \end{bmatrix} = 0.2989 \times \begin{bmatrix} R_{1,1} & \dots & R_{1,640} \\ \vdots & \ddots & \vdots \\ R_{480,1} & \dots & R_{480,640} \end{bmatrix} + 0.5870 \times \begin{bmatrix} G_{1,1} & \dots & G_{1,640} \\ \vdots & \ddots & \vdots \\ G_{480,1} & \dots & G_{480,640} \end{bmatrix} + 0.1140 \times \begin{bmatrix} B_{1,1} & \dots & B_{1,640} \\ \vdots & \ddots & \vdots \\ B_{480,1} & \dots & B_{480,640} \end{bmatrix}$$

   Se puede apreciar que el canal que más contribuye a la generación de la escala de grises es el verde, seguido del rojo y terminando por el azul.

2. **Conversión de la imagen en escala de grises a binaria (blanco o negro absolutos)**. Obtenida la matriz en escala de grises (valores entre 0 y 255) se define un valor límite por encima del cual se considere que el valor será negro y por debajo del cual será blanco. El número que parece más lógico utilizar es el punto medio entre 0 y 255, el valor 128. Sin embargo hay que tener en cuenta que es posible que toda la imagen presente una alta iluminación u oscuridad. En esos casos, aunque hubiera presente un marcador no se detectaría ya que toda la imagen se reduciría al valor 0 (negro) en imágenes oscuras y a 1 en imágenes luminosas. Para arreglar esto se define el valor límite como la media entre el valor máximo y mínimo de los píxeles de la imagen y a continuación se aplica la clasificación según el valor límite obteniendo una imagen en blanco y negro.

3. **Obtención de polígonos**. Una vez obtenida la imagen en blanco y negro se dispone de una alternancia de regiones anidadas de distinto color. Una región blanca contiene subregiones negras, que a su vez contienen subregiones blancas...etc.

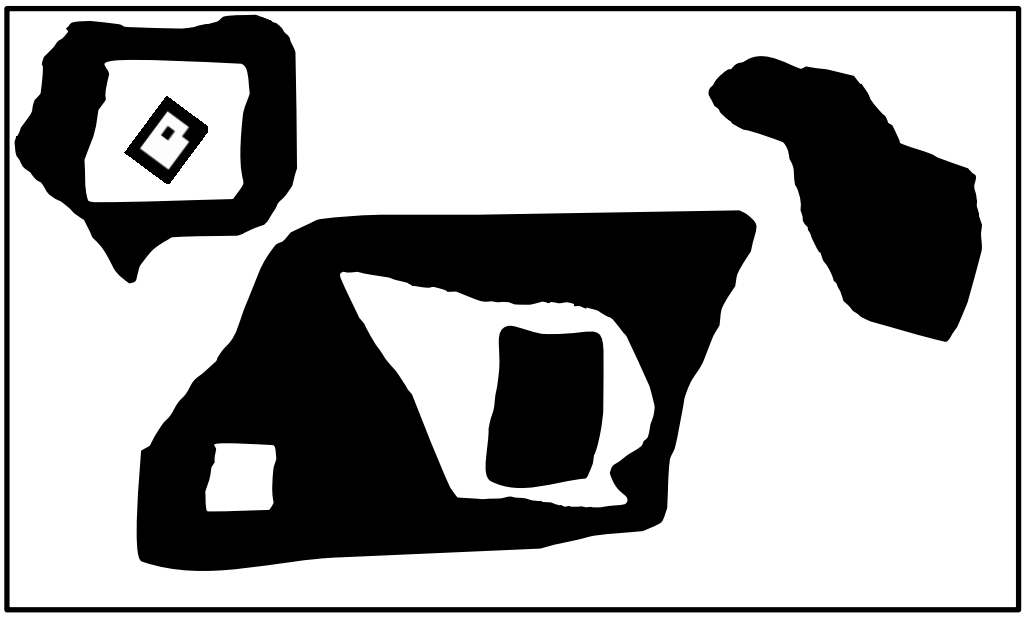


ILUSTRACIÓN 39 - REGIONES ANIDADAS

Los vértices que delimitan estas regiones se pueden obtener a través de la librería de visión artificial OpenCV y, organizándolas en un árbol, se pueden clasificar las regiones de la siguiente manera:

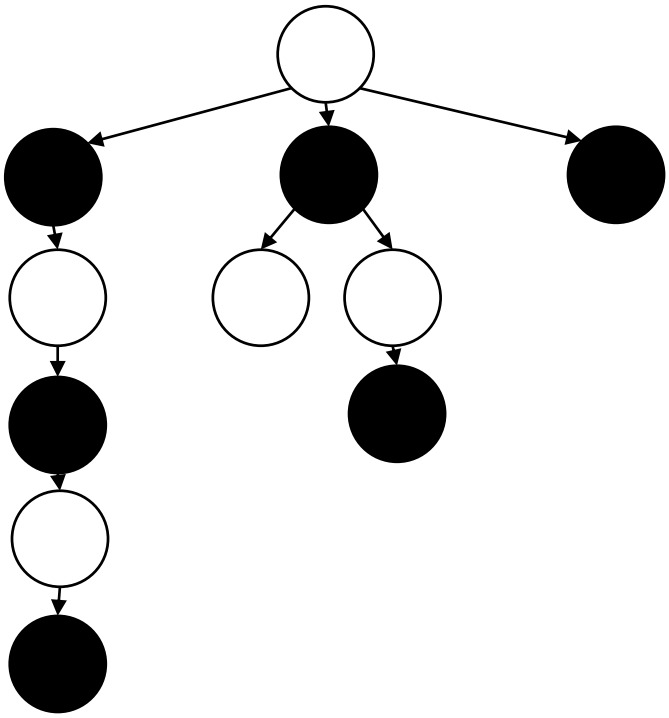


ILUSTRACIÓN 40 - ÁRBOL DE REGIONES

Analizando el color de los elementos del árbol se seleccionan las regiones negras como candidatas a albergar un marcador.

4. **Reducción de polígonos a cuadriláteros**. En esta fase se combinan los vértices de las regiones negras que se encuentran próximos hasta su reducción a cuadriláteros con el fin de corregir posibles errores debidos a ruido en la captura. Una vez hecho esto se dispone de una lista de cuadriláteros que podrían contener un marcador.

5. **Cálculo de transformación de cuadriláteros a cuadrados.** Los cuadriláteros detectados son el resultado de la transformación del cuadrado exterior del marcador como consecuencia del punto de vista desde donde se observa. Se busca revertir el proceso a través del cálculo de la matriz de transformación 2D que, al aplicar a los vértices del marcador, resulte en un cuadrado.

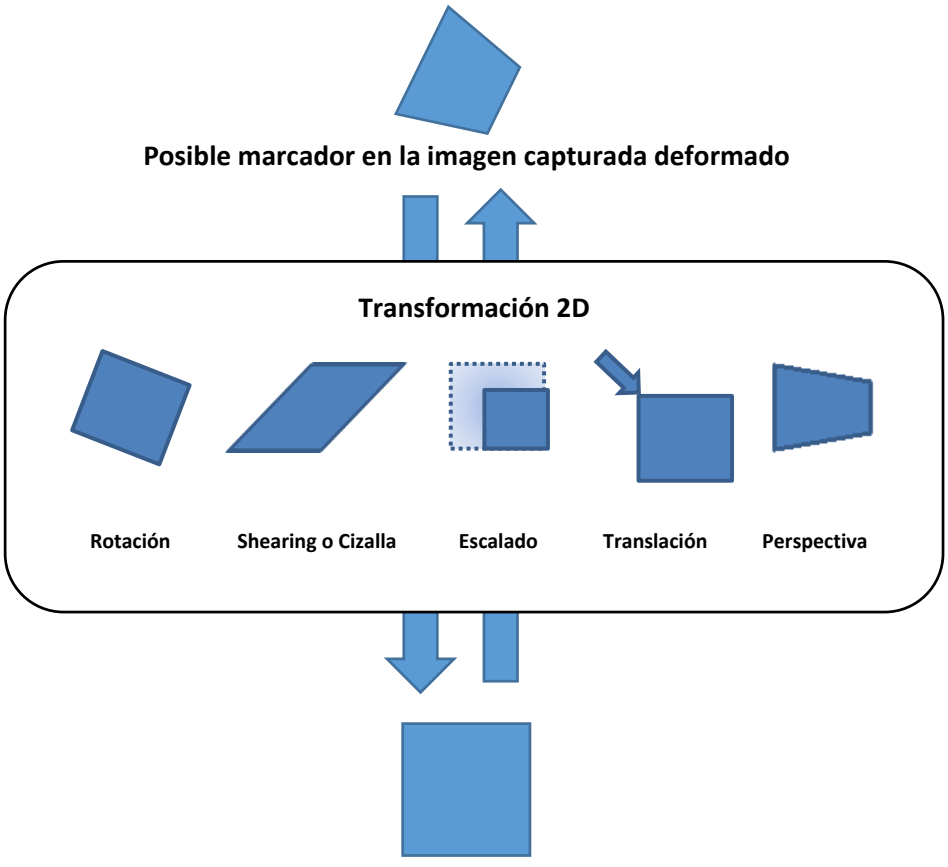


ILUSTRACIÓN 41 - TRANFORMACIONES 2D

Esta matriz de transformación es una matriz de 3x3 y presenta la siguiente estructura:

$$Matriz\ de\ Transformación\ 2D = \begin{bmatrix} ScaleX * Rotate & Shearing * Rotate & TrasnlationX \\ Shearing * Rotate & ScaleY * Rotate & TrasnlationY \\ Perspective & Perspective & 1 \end{bmatrix}$$

Por cada vértice del cuadrilátero al que se aplique esta transformación se debería obtener un punto del cuadrado final, es decir, uno de los puntos (0,0),(0,6),(6,0) o (6,6).

Se tiene que para cada punto (x_i,y_i) se quiere obtener un punto del cuadrado (x_i',y_i') según la siguiente ecuación:

$$\begin{bmatrix} t_i x_i' \\ t_i y_i' \\ t_i \end{bmatrix} = \texttt{map\_matrix} \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Siendo map_matrix la matriz de transformación 2D y t_i una constante.

Dado que existen cuatro puntos esto plantea un sistema de ecuaciones de 4*3=12 ecuaciones a resolver con 12 incógnitas. De estas incógnitas, 8 provienen de la matriz de transformación 2D correspondientes a cada elemento de la matriz exceptuando el último, que se sabe que es 1. Las otras 4 incógnitas provienen de las constantes t_i.

Una vez resuelto el sistema se dispone de la matriz de transformación 2D. y se prosigue con el mapeo de cada pixel del cuadrilátero en la matriz final de 6x6.

6. **Conversión a matriz 6x6 binaria**.Teniendo ahora en cuenta cada pixel de interior del cuadrilátero y no sólo los vértices, se calcula a qué elemento de la matriz 6x6 contribuye cada pixel a través de la siguiente fórmula:

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & 1 \end{bmatrix} \times \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} t \times X' \\ t \times Y' \\ t \end{bmatrix}$$

El punto final sobre el que se proyectará cada pixel de la imagen en la matriz de 6x6 será el valor de X' e Y' cuando t=1.

Haciendo la media aritmética entre las contribuciones de todos los puntos a cada celda se obtiene una matriz 6x6 en escala de grises que se pasará a blanco y negro definiendo un valor límite o umbral de la misma forma que se hizo previamente, mediante el cálculo de la media aritmética entre los valores máximo y mínimo presentes en la matriz.

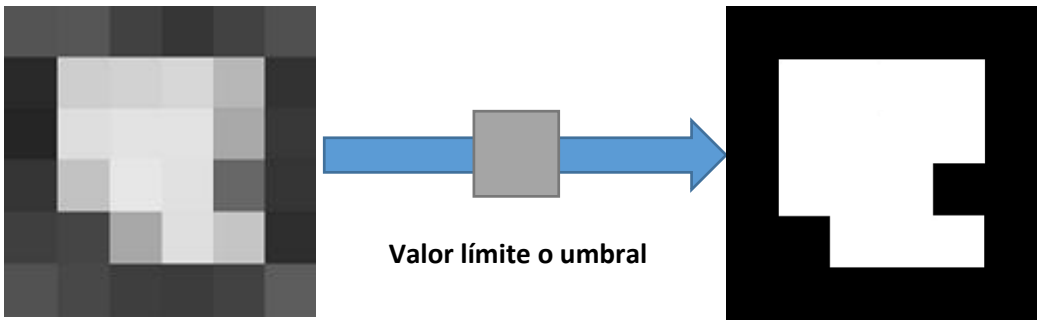


ILUSTRACIÓN 42 - APLICACIÓN DE UMBRAL

Una vez obtenida esta matriz binaria de 6x6 se prosigue con la identificación del marcador como se ha explicado previamente.

5.3.2.2. Posicionamiento del usuario respecto del marcador

El cálculo de la posición del usuario pasa por la obtención de los vectores de traslación y rotación una vez comprobado que el cuadrilátero corresponde con un marcador y que el marcador se encuentra presente en el escenario (contrastados con la información del a base de datos).

Estos cálculos permiten conocer la distancia a la que se encuentra la cámara del marcador así como el ángulo desde el cual se está observando. La ecuación utilizada es la siguiente:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} x \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \times \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

La proyección de un punto del espacio tridimensional (X,Y,Z) sobre el punto de la imagen (u,v) se relaciona a través de la multiplicación de dos matrices tal como se presenta en la fórmula. La primera matriz corresponde con la matriz intrínseca de la cámara. Esta matriz indica la referencia en la imagen para la traslación, mediante los valores c_x y c_y, y el ángulo de visión, mediante la introducción de la distancia focal horizontal y vertical medida en píxeles en los valores focal f_x y f_y. Si la distancia focal cambiara por la utilización de algún tipo de lente, se debería aplicar un factor a estos valores para su compensación. La segunda matriz del producto contiene el valor de los tres vectores de rotación, que representarán los ejes de coordenadas, y el vector de traslación. Dado que se conoce la primera matriz, las incógnitas a resolver son las 12 pertenecientes a la segunda matriz. Para ello se plantean 12 ecuaciones utilizando 4 puntos, en concreto los puntos de los vértices del cuadrilátero.

5.4. Sistema de conexión

La implementación del sistema de conexión se inició con la modificación de los XML que utiliza el sistema GREP para la definición de juegos. En particular se requirió la identificación unívoca entre una entidad de GREP y un elemento de partida definido en la base de datos.

Para ello se añadió un atributo a la entidad GREP en el XML de definición de juego, el cual indicara el *mixed game play element id* (codificado como "mgplayel-id" en el XML), que corresponde con el *play_element_id* de la tabla play_element en la base de datos.

Como ejemplo, una entidad de juego que representara un fuego en GREP se vería modificada de la siguiente manera:

```
<entity id="Fuego" name="Fuego" pos="-3.993738;2.624559;2.4684">
    <attribute type="health" value="3500"/>
    <attribute type="hpregen" value="2"/>
    <attribute type="enabled" value="OFF"/>
    <attribute type="instanceName" value="Fuego1"/>
    <attribute type="mgplayel-id" value="55"/>
</entity>
```

Siendo en este caso 55 el *play_element_id* correspondiente al elemento de partida asociado a la entidad de juego GREP.

Conocida la identidad del elemento de partida por el sistema GREP y dado que este debe ser capaz de gestionar el elemento de partida (rotar, desplazar, mostar/esconder…), se necesitó la generación de un servicio de modificación de elementos de partida accesible desde el navegador web sobre el que se ejecuta GREP.

Para ello se decidió la generación de una pequeña librería de JavaScript que generara llamadas asíncronas con AJAX a un servidor web, el cual se encargaría de la modificación de la base de datos.

La modificación de los elementos de partida se efectúa con la generación de un mensaje de modificación con la siguiente estructura:

<play_element_id> - <attribute> : <value>[/ <attribute> : <value>]*

Donde <play_element_id> es número entero identificador del elemento de partida, <attribute> es el nombre del campo a modificar utilizando la nomenclatura de GREP y <value> es el valor actualizado del campo a modificar.

Este mensaje es interpretado por la librería de JavaScript implementada, quien se encarga de su envío asíncrono al servicio web de modificación.

Este servicio web fue implementado en PHP aprovechando la experiencia obtenida con el desarrollo del sistema de edición de escenarios en la gestión de llamadas asíncronas.

A continuación se detalla a través de un diagrama los pasos llevados a cabo para actualización de un elemento de partida junto con un ejemplo simplificado.

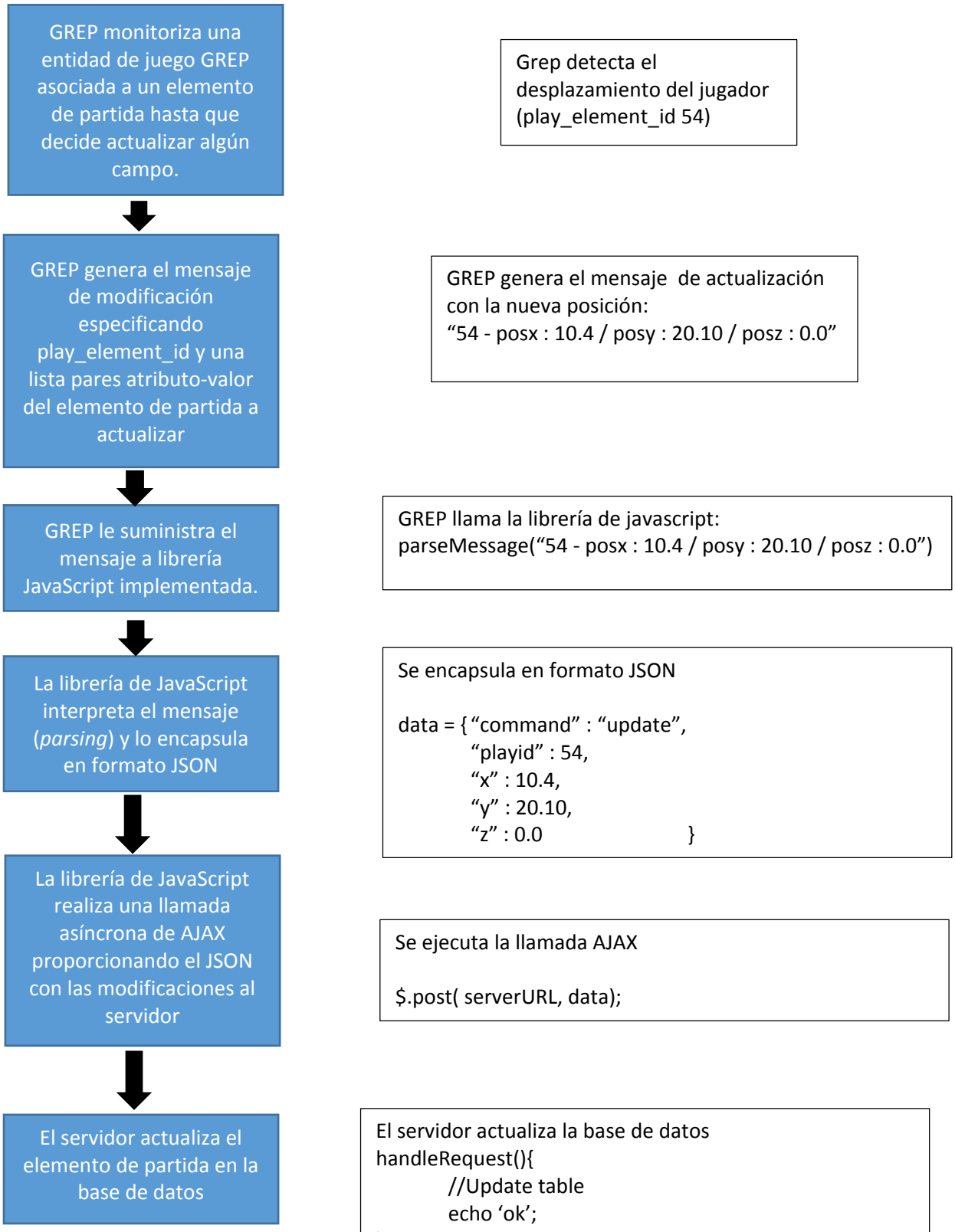


ILUSTRACIÓN 43 - DIAGRAMA SISTEMA DE CONEXIÓN

6. Implantación y Pruebas

En esta sección del documento se detallará el despliegue del sistema en los diferentes componentes hardware para la realización de las pruebas. Seguidamente se presentarán una serie de pruebas realizadas sobre la edición de un escenario asistida por el sistema de visualización.

6.1. Implantación

Se presentará un diagrama de despliegue del sistema incluyendo los distintos componentes hardware y software. Para ello se presenta inicialmente el hardware utilizado:

6.1.1. Hardware empleado

Los elementos hardware utilizados son los siguientes:

- **Vuzix STAR 1200.** Este es el dispositivo de realidad aumentada utilizado que permite la superposición de los elementos virtuales sobre la visión real del usuario. Se utiliza para el sistema de visualización.

ILUSTRACIÓN 44 - GAFAS AR

- **Cámara de vídeo Logitech C910**. Dispositivo de captura de vídeo montado sobre el dispositivo de realidad aumentada. Configurado para la captura de imágenes en formato VGA 640x480 píxeles.

ILUSTRACIÓN 45 - CÁMARA DE VÍDEO

- **Dispositivo portátil de procesamiento, servidor web y gestor de persistencia.** Este dispositivo tiene la responsabilidad de ejecutar el sistema de visualización de escenarios

y partidas, albergar la base de datos y ofrecer la parte del servidor tanto del sistema de edición de escenarios como del sistema de conexión. La decisión de implantar los sistemas de esta manera surge de la necesidad de actualización continua de los datos de visualización, lo que genera consultas a la base de datos que si se encontraran en otro dispositivo tardarían más en ser visibles, aumentando así el retraso o *lag* entre la modificación de un elemento virtual y su reflejo en el sistema de visualización.

Este dispositivo ha sido la mayor fuente de problemas del proyecto y lo que ha causado mayores retrasos en la planificación, ya que tuvo que ser cambiado a mitad de proyecto.

- **Dispositivo inicial: ODROID U2**. Este dispositivo se utilizó como dispositivo inicial sobre el que conectar las gafas de realidad aumentada por su tamaño reducido (48x52 mm), por su capacidad de cómputo (procesador Samsung Exynos4412 Prime Cortex-A9 Quad Core 1.7Ghz con caché L2 de 1MB), su tarjeta gráfica (Mali-400 Quad Core 440MHz) y por su memoria de 2 GB. Los sistema operativos que soporta son principalmente Ubuntu 13.04 (con ciertas variantes) y Android (ahora actualizado hasta la versión 4.3).

ILUSTRACIÓN 46 - ODROID U2

El principal problema por el cual no se llegó a poder utilizar el dispositivo fue el soporte de las librerías gráficas y de renderización de modelos 3D junto con los controladores de la tarjeta gráfica incorporados. En concreto openFrameworks ofrecía una distribución para dispositivos con sistema operativo Linux basado en procesador ARM y se conocía el uso de openFrameworks con este dispositivo. Sin embargo, a la hora de efectuar la renderización de modelos 3D la librería ASSIMP, que encapsulaba openFrameworks a través del *addon* ofxAssmipModelLoader, no podía compilarse para soportar procesadores ARM y la versión de OpenGL para sistemas Embebidos OpenGL ES.

Adicionalmente, la actualización del sistema operativo y parcheado de controladores dependía de imágenes del sistema proporcionadas por una comunidad de usuarios de dispositivos ODROID, a cuyo foro principal se accede desde la página web del fabricante de ODROID, Hardkernel. En este foro se anunciaron actualizaciones que nunca llegaron. De esta manera, aunque se comenzó con el desarrollo del sistema de visualización para este dispositivo, se terminó aplazando su uso hasta que se actualizara el soporte de las librerías requeridas.

Este dispositivo se pensó utilizar conjuntamente con una *tablet*: mientras en el ODROID se ejecuta el sistema de visualización utilizando las gafas de realidad aumentada, en la *tablet* se ejecutaría el sistema de edición de escenarios o lo que es lo mismo, se accede a la página web de edición de escenarios servida desde el ODROID.

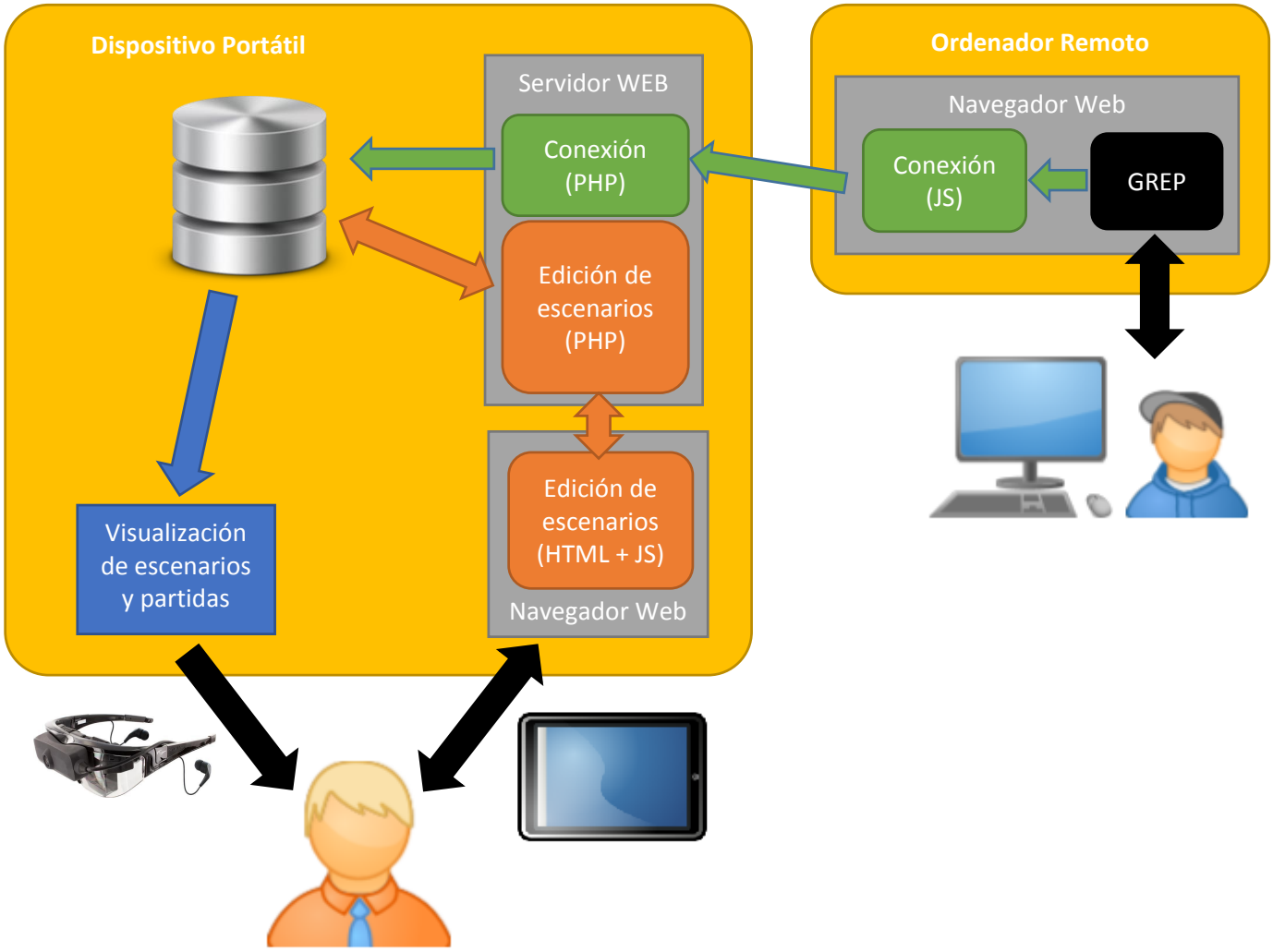
- **Dispositivo final: Asus Transformer Book T100**. Este dispositivo cumple a la vez con la responsabilidad esperada del ODROID y de la *tablet*. La primera ventaja que plantea este dispositivo es que utiliza un procesador de arquitectura x86, en contraposición con la ARM del ODROID. Esto supone una ventaja debido al problema de compatibilidad de librerías del ODROID, el cual se evita de este modo.

ILUSTRACIÓN 47 - ASUS TRANSFORMER BOOK T1

En particular, el sistema operativo con el que se vende el dispositivo es Windows (8.1), sistema operativo con soporte para las librerías necesarias.

Este dispositivo permite además la separación del teclado de la pantalla, convirtiéndose así en un dispositivo táctil de pantalla reducida como lo son las *tablets*.

Expuestos los componentes hardware, se procede a mostrar visualmente el despliegue de los sistemas implementados.



6.1.2. Diagrama de despliegue de sistemas

Como se expone en el diagrama, el sistema se centra en un dispositivo principal (el dispositivo portátil), en el que se despliega: la base de datos, el sistema de visualización de escenarios y partidas, el lado del servidor del sistema de conexión y el lado del servidor del sistema de edición de escenarios. Además, en este dispositivo se puede ejecutar la parte cliente del sistema de edición de escenarios a través de un navegador web.

Por otro lado, en ordenadores remotos los usuarios de los juegos GREP juegan desde su navegador. Es aquí donde GREP hace uso de la librería JavaScript de conexión, que sincroniza los cambios realizados en el juego GREP con la base de datos a través de las llamadas a la parte del servidor del sistema de conexión.

Las flechas en el diagrama indican el flujo de información.

6.2. Pruebas

En esta sección se detallan una serie de pruebas que demuestran el correcto funcionamiento del sistema de visualización ejecutado de manera simultánea con el sistema de edición de escenarios.

Para la realización de las pruebas se ha creado un escenario vacío con una cuadrícula como plano. A continuación se ha añadido al sistema un modelo, el cual representa un bombero.

Una vez preparado el sistema se ha procedido a añadir un marcador que permita el posicionamiento del visualizador respecto del mundo virtual y un elemento virtual del tipo de modelo recientemente añadido: el bombero.

A partir de ese punto se han realizado cambios simples sobre el escenario de manera que se visualizarán los cambios en el sistema de visualización.

Los resultados se muestran enseñando una captura de pantalla del sistema de edición de escenarios al mismo tiempo que una adaptación de lo que se vería en el dispositivo de realidad aumentada inmediatamente debajo. Debido a la práctica imposibilidad de mostrar una foto de lo que se vería a través de las gafas de realidad aumentada, se ha optado por superponer la visión virtual sobre la captura de video hecha por la cámara para los tests, de manera que se vea lo que había detrás de los elementos virtuales en cada momento.

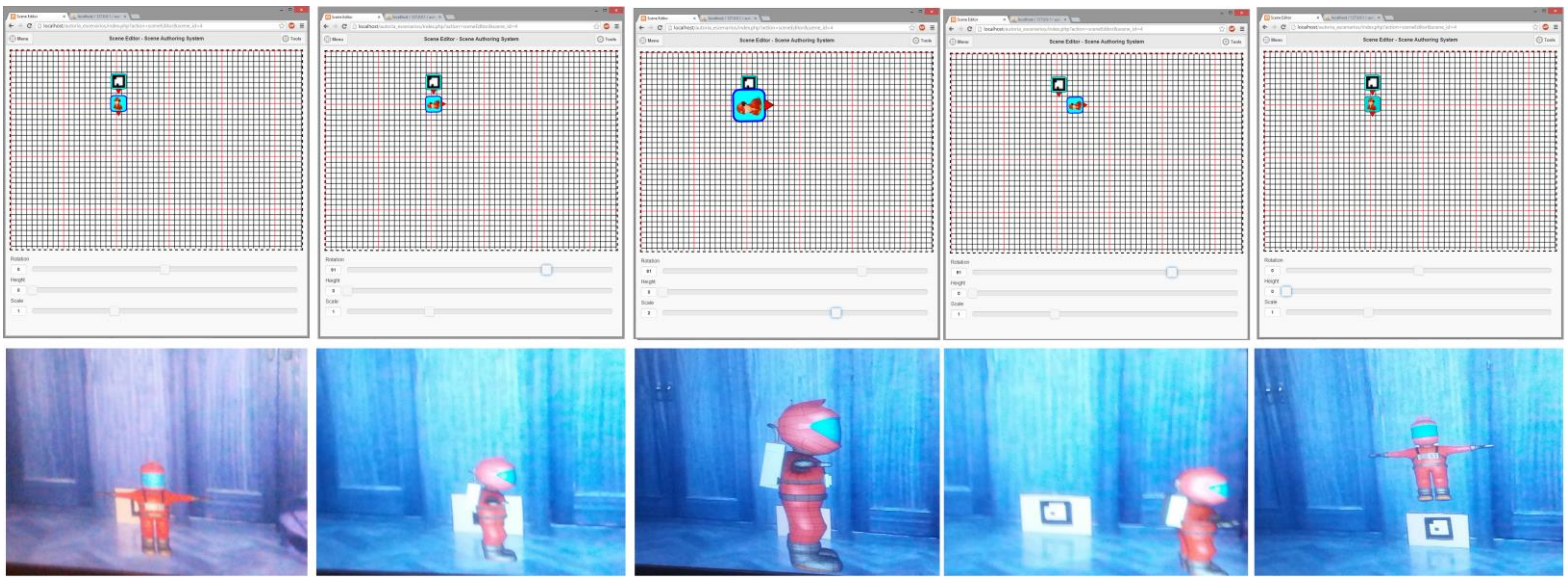


ILUSTRACIÓN 49 - PRUEBAS

Como se muestra en la ilustración de pruebas, se han realizado cinco *tests*. El primero de ellos simplemente comprueba que el marcador se detecta correctamente y se muestra el contenido virtual en la posición correcta.

El segundo test comprueba la rotación respecto del eje z. En concreto se giran 91º en sentido contrario a las agujas del reloj al mirar desde arriba. En la imagen capturada del sistema de visualización se observa como el elemento virtual ha girado en la misma medida en que el nodo ha girado en el sistema de edición de escenarios.

El tercer test comprueba la correcta transformación del elemento virtual modificando su escala. Se puede apreciar un incremento en el tamaño del objeto virtual.

El cuarto test comprueba la traslación del elemento virtual. Como se muestra, el elemento virtual no ha mantenido la posición, desplazándose en la medida y dirección especificadas.

Por último el quinto test comprueba la elevación del elemento virtual desde el sistema de edición de escenarios.

7. Gestión del Proyecto

En este apartado se especifican las fases de desarrollo del proyecto exponiendo el modelo de ciclo de vida empleado. Tras esto se presentara la planificación inicial que se verá contrastada con el tiempo que finalmente fue necesario para el desarrollo de cada una de las tareas. A demás se presentará el desglose de los costes asociados al proyecto.

7.1. Modelo de ciclo de vida

El modelo de desarrollo de software empleado en este proyecto corresponde con el modelo en cascada con retroalimentación. Este modelo define una secuencia ordenada de fases de desarrollo. En el modelo de cascada original el orden de estas fases es estricto: tras terminar una fase se comienza con la siguiente. La modificación de este modelo añadiendo retroalimentación permite volver a una fase anterior para realizar cambios.

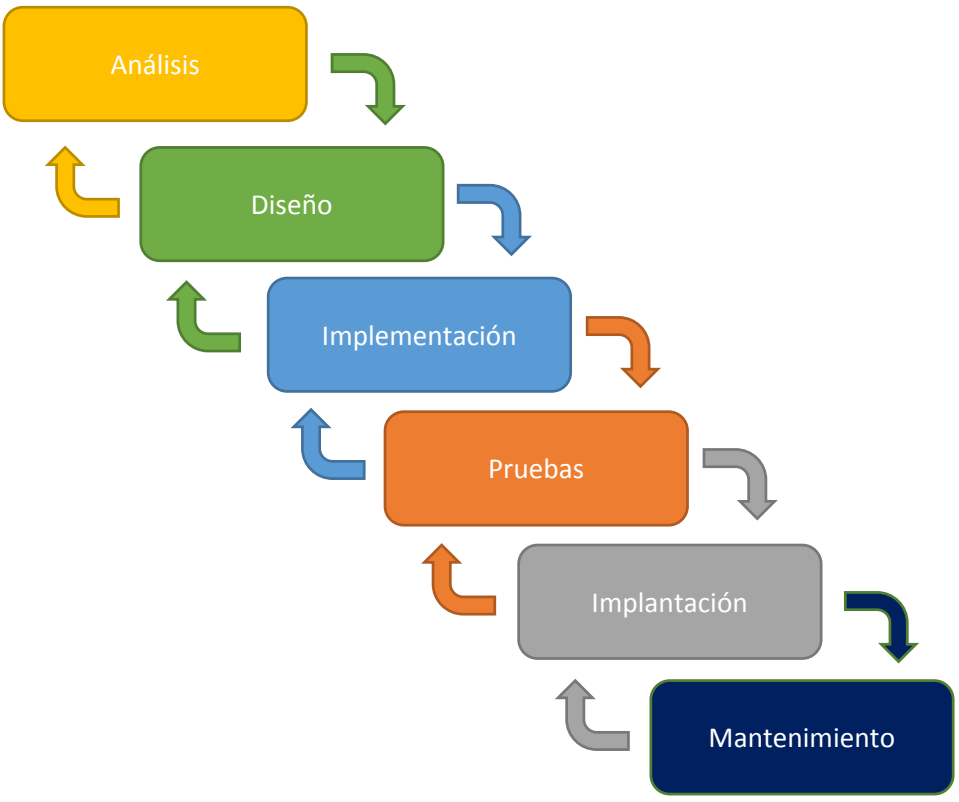


ILUSTRACIÓN 50 - CICLO DE VIDA EN CASCADA

Este modelo de ciclo de vida tiene la ventaja de permitir una planificación sencilla y permite el seguimiento del proyecto. Sin embargo, también tiene inconvenientes. En particular, este

modelo es adecuado cuando los requisitos son estables y la concepción del sistema es clara y prácticamente fija desde la fase del análisis.

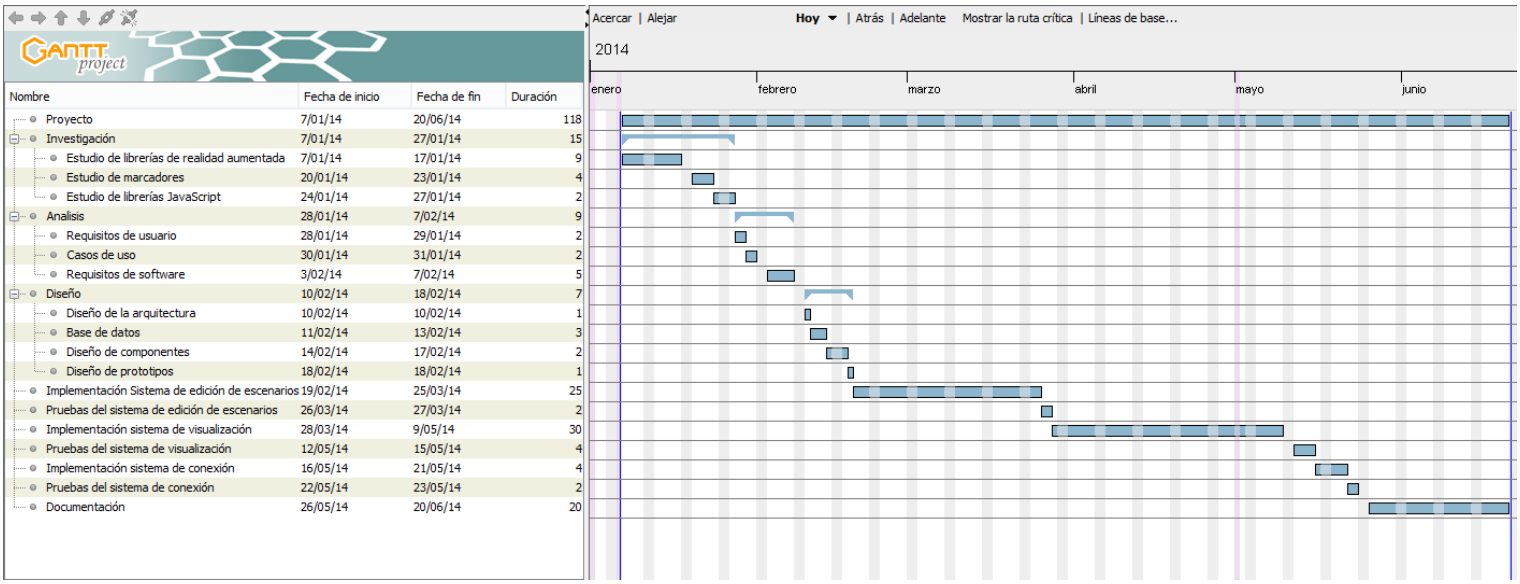
7.2. Planificación

La planificación permite la obtención de una aproximación de la cantidad de tiempo necesario para el desarrollo del proyecto. Además, el uso de herramientas de planificación como los diagramas Gantt permite el seguimiento del proyecto, de manera que se puede estimar si el proyecto se encuentra en situación de cumplir con los requisitos temporales. Si la desviación temporal respecto de la planificación inicial es excesivamente alta se podría incluso plantear el abandono del proyecto, con las posibles consecuencias económicas.

7.2.1. Planificación inicial

Para la planificación inicial se ha asumido una jornada laboral de 4 horas diarias. Esta es una estimación de dedicación realista que se debe a que el único desarrollador del proyecto es el autor de este documento, del que se sabe que se encuentra todavía finalizando sus estudios y no puede dedicarse a tiempo completo.

Las tareas a realizar junto con su estimación se presentan a través de un diagrama Gantt generado con la herramienta de software libre GanttProject.



Nombre	Fecha de inicio	Fecha de fin	Duración
Proyecto	7/01/14	20/06/14	118
Investigación	7/01/14	27/01/14	15
Estudio de librerías de realidad aumentada	7/01/14	17/01/14	9
Estudio de marcadores	20/01/14	23/01/14	4
Estudio de librerías JavaScript	24/01/14	27/01/14	2
Analisis	28/01/14	7/02/14	9
Requisitos de usuario	28/01/14	29/01/14	2
Casos de uso	30/01/14	31/01/14	2
Requisitos de software	3/02/14	7/02/14	5
Diseño	10/02/14	18/02/14	7
Diseño de la arquitectura	10/02/14	10/02/14	1
Base de datos	11/02/14	13/02/14	3
Diseño de componentes	14/02/14	17/02/14	2
Diseño de prototipos	18/02/14	18/02/14	1
Implementación Sistema de edición de escenarios	19/02/14	25/03/14	25
Pruebas del sistema de edición de escenarios	26/03/14	27/03/14	2
Implementación sistema de visualización	28/03/14	9/05/14	30
Pruebas del sistema de visualización	12/05/14	15/05/14	4
Implementación sistema de conexión	16/05/14	21/05/14	4
Pruebas del sistema de conexión	22/05/14	23/05/14	2
Documentación	26/05/14	20/06/14	20

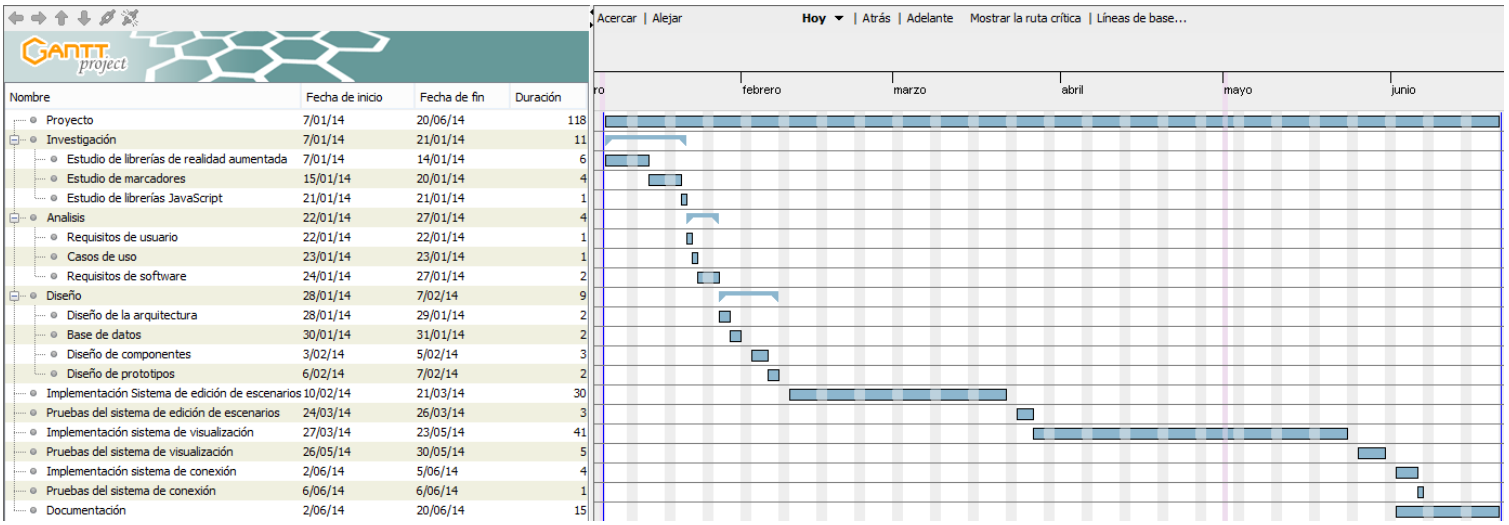
ILUSTRACIÓN 51 - PLANIFICACIÓN INICIAL

En concreto, la planificación de inicio del proyecto se produce el día 7 de enero de 2014 y finaliza el día 20 de junio. Teniendo en cuenta que no se trabaja los fines de semana esto resulta en 118 jornadas de trabajo, o lo que es lo mismo 472 horas totales de desarrollo.

Si se analizan las tareas expuestas en el diagrama Gantt se puede ver cómo se produce un retroceso en ciclo de vida del desarrollo. En particular se tiene un paso de las fases de pruebas a fases de implementación. Esto se debe a que el proyecto ha sido divido en tres subsistemas y, aunque las fases previas se han planificado conjuntamente para los tres sistemas, las secuencia implementación-pruebas ha decidido llevarse a cabo de manera independiente, aprovechando la retroalimentación permitida en el ciclo de vida.

7.2.2. Planificación final

Tras el desarrollo del proyecto se presenta la planificación final tal como se ha llevado a cabo realmente y se analizan las diferencias con respecto a la planificación inicial.



Nombre	Fecha de inicio	Fecha de fin	Duración
Proyecto	7/01/14	20/06/14	118
Investigación	7/01/14	21/01/14	11
Estudio de librerías de realidad aumentada	7/01/14	14/01/14	6
Estudio de marcadores	15/01/14	20/01/14	4
Estudio de librerías JavaScript	21/01/14	21/01/14	1
Analisis	22/01/14	27/01/14	4
Requisitos de usuario	22/01/14	22/01/14	1
Casos de uso	23/01/14	23/01/14	1
Requisitos de software	24/01/14	27/01/14	2
Diseño	28/01/14	7/02/14	9
Diseño de la arquitectura	28/01/14	29/01/14	2
Base de datos	30/01/14	31/01/14	2
Diseño de componentes	3/02/14	5/02/14	3
Diseño de prototipos	6/02/14	7/02/14	2
Implementación Sistema de edición de escenarios	10/02/14	21/03/14	30
Pruebas del sistema de edición de escenarios	24/03/14	26/03/14	3
Implementación sistema de visualización	27/03/14	23/05/14	41
Pruebas del sistema de visualización	26/05/14	30/05/14	5
Implementación sistema de conexión	2/06/14	5/06/14	4
Pruebas del sistema de conexión	6/06/14	6/06/14	1
Documentación	2/06/14	20/06/14	15

ILUSTRACIÓN 52 - PLANIFICACIÓN FINAL

Expuesto el diagrama Gantt real se analizan las diferencias con la planificación inicial y se justifican sus causas.

Tarea	Duración planificada	Duración real	Desviación de duración	Desviación acumulada
Investigación	**15**	**11**	**-4**	**-4**
Análisis	**9**	**4**	**-5**	**-9**
Diseño	**7**	**9**	**2**	**-7**
Implementación edición de escenarios	**25**	**30**	**5**	**-2**
Pruebas edición de escenarios	**2**	**3**	**1**	**-1**
Implementación de visualización	**30**	**41**	**11**	**10**
Pruebas de visualización	**4**	**5**	**1**	**11**
Implementación de conexión	**4**	**4**	**0**	**11**
Pruebas de conexión	**2**	**1**	**1**	**10**
Documentación	**20**	**(15+10) 25**	**5**	**15**
Proyecto total	**118**	**133**	**15**	**15**

TABLA 4 - COMPARACIÓN DE PLANIFICACIONES

En la tabla anterior se calcula utilizando como unidad la jornada de trabajo (equivalente a 4 horas), la desviación sufrida durante el proyecto, tanto por tarea como acumulada a la finalización de la tarea. En color verde se destacan las desviaciones más deseables, aquellas que suponen una sobreestimación inicial de tiempo realmente requerido. En rojo, por contraposición, se marca la desviación de subestimación, o lo que es lo mismo, los retrasos en las tareas.

Como se puede ver en la tabla, el desarrollo del proyecto comenzó más rápido de lo esperado inicialmente. En concreto la investigación y análisis duraron menos de lo esperado mientras que prácticamente el resto de tareas duró más de lo esperado.

El primer momento en el que globalmente se comenzó a entrar en retraso fue en la fase de implementación del sistema de visualización de escenarios y partidas, tarea que tomó 11 jornadas de trabajo más de lo esperado. Esto se debe principalmente al cambio de dispositivo sobre el que se ejecutaba el sistema, comentado en el capítulo de pruebas e implantación. Resumidamente, el dispositivo inicial (ODROID U2) no pudo desempeñar la funcionalidad requerida debido principalmente a la falta de soporte de librerías de renderización de modelos 3D, a pesar de existir la rama de desarrollo de estas librerías para dispositivos con la misma

arquitectura ARM. Esto forzó a la modificación de los requisitos y del sistema de visualización para el soporte de un sistema operativo distinto (se pasó de Ubuntu 13.04 a Windows 8.1).

Debido a que la fecha de finalización del proyecto no podía ser aplazada, se llevó a cabo un incremento del horario de la jornada laboral durante las tres últimas semanas del proyecto (de 4 a 8 horas diarias).

En resumen, el proyecto sufrió una desviación de 15 jornadas de trabajo sobre un total estimado de 118, lo que supone un 12,71% de desviación en cuanto a esfuerzo de desarrollador o, expresado en horas, un sobreesfuerzo de 60 horas sobre 472. Se insiste en el hecho de que esta desviación no llegó a afectar a la consecución de los objetivos del proyecto.

7.3. Presupuesto

Esta sección detalla el coste total del proyecto a través de un desglose en personal, y materiales.

7.3.1. Recursos humanos

Las personas que han participado en el desarrollo del proyecto son únicamente dos, el autor del proyecto y su tutor. Sin embargo, para el cálculo del presupuesto se separa la actividad de desarrollo en distintos roles simulando el desarrollo por un equipo más amplio donde los distintos roles tienen una remuneración diferente.

Persona	Rol	Horas dedicadas	Coste por hora (€/hora)	Coste total (€)
Luis Arenas Rivera	Analista	60	25	1.500,00
	Diseñador	36	30	1.080,00
	Programador	300	20	6.000,00
	Tester	36	20	720,00
	Documentador	100	16	1.600,00
Telmo Zarraonandia Ayo	Ingeniero senior	10	40	400,00
			Total (€)	11.300,00

TABLA 5 - SALARIOS BRUTOS

	Cuantía (€)	Cuota seguridad social (%)	Coste total (€)

Salarios brutos	11.300,00	35	3.955,00

TABLA 6 - CUOTA SEGURIDAD SOCIAL

	Coste total (€)
Cuota seguridad social	3.955,00
Salarios brutos	11.300,00
Total (€)	**15.255,00**

TABLA 7 - COSTES PERSONALES

Tras aplicar el 35% de la cuota empresarial a la seguridad social correspondiente con el Grupo J de la Clasificación Nacional de Actividades Económicas (actividades de información y comunicaciones), el total asciende a QUINCE MIL DOSCIENTOS CINCUENTA Y CINCO EUROS (**15.255,00 €**).

7.3.2. Recursos materiales

7.3.2.1. Inmueble

Inmueble	Alquiler mensual	Duración del proyecto (meses)	Coste total (€)
Oficina	600	5,5	3.300,00
		Total (€)	**3.300,00**

TABLA 8 – INMUEBLE

7.3.2.2. Equipo amortizable y licencias de software

En este apartado se define el material

Material	Precio unitario (€)	Vida útil (meses)	Coste mensual por amortización	Duración del proyecto (meses)	Coste total (€)
Equipo portátil de desarrollo	1.200	36	33,33	5,5	183,33
Gafas de realidad aumentada Vuzix STAR 1200	3.680	48	76,66	5,5	421,66
ASUS Transformer Book T100	370	36	10,28	5,5	56,54
ODROID U2	66	12	5,5	5,5	30,25
Microsoft Office 2010	139	36	3,86	5,5	21,24

Microsoft Visual Studio 2012 Professional	1.283,65	12	106,97	5,5	588,34
				Total (€)	**1.301,37**

TABLA 9 - EQUIPO AMORTIZABLE Y LICENCIAS DE SOFTWARE

7.3.2.3. Material fungible

Material	Coste total (€)
Material de oficina y fotocopias	100,00
Total (€)	**100,00**

TABLA 10 - MATERIAL FUNGIBLE

7.3.2.4. Gastos corrientes

	Coste mensual (€)	Meses	Coste total (€)
Internet, luz, agua y gas	90	5,5	495,00
		Total (€)	**495,00**

TABLA 11 – GASTOS CORRIENTES

7.3.3. Resumen de costes

A continuación se calcula el coste (sin IVA) del proyecto.

	Coste total (€)
Costes Personales	**15.255,00**
Inmueble	**3.300,00**
Equipo amortizable y licencias de software	**1.301,37**
Material fungible	**100,00**
Gastos corrientes	**495,00**
TOTAL (€)	**20.451,37**

TABLA 12 - RESUMEN DE COSTES

El coste asociado al desarrollo del proyecto asciende a VEINTE MIL CUATROCIENTOS CINCUENTA Y UN EUROS CON TREINTA Y SIETE CÉNTIMOS (20.451,37 €) IVA no incluido.

8. Conclusiones y trabajo futuro

8.1. Conclusiones

Los cambios en la sociedad y en la tecnología nos llevan a la posibilidad de explorar nuevas técnicas educativas haciendo uso de los nuevos recursos tecnológicos. Para esto es vital proporcionar a los educadores las herramientas necesarias para la creación de contenidos educativos, como pueden ser los juegos basados en la Realidad Aumentada.

Aunque se han cumplido con los objetivos, no ha sido una tarea fácil y la principal conclusión sacada es que un ingeniero informático siempre debe adaptarse a las nuevas tecnologías.

El desarrollo de este proyecto ha requerido un uso de múltiples tecnologías y ha supuesto un enorme esfuerzo de aprendizaje constante. A continuación se enumeran las tecnologías que se han tenido estudiar: Unity, HTML5, JavaScript, AJAX, PHP, CSS3, JQuery Mobile, KineticJS, JSON, XML, MYSQL, openFrameWorks, OpenCV, C++, makefiles, compilación cruzada, gestión de sistemas Linux, modelado 3D y algoritmos de visión artificial. Es por esto que pienso que este proyecto es lo suficientemente completo.

En cuanto a la Realidad Aumentada, durante el desarrollo del proyecto se han podido apreciar las limitaciones actuales de esta tecnología, asociadas principalmente al hardware de visualización de realidad aumentada.

Las gafas de RA utilizadas son aún demasiado grandes y pesadas y se calientan con facilidad. Además, cubren un ángulo de visión todavía muy pequeño, de manera que los elementos virtuales a visualizar deben ser mirados casi frontalmente.

El uso de las nuevas gafas de Google podría reducir el problema de tamaño, peso y posiblemente calor, pero plantearía otros retos asociados a la reducida pantalla de que dispone.

8.2. Trabajo futuro

Este proyecto plantea la interacción con elementos virtuales simplemente a través de la visualización de ellos o a través de la interacción con un sistema virtual (el juego GREP). En ningún momento las acciones que se realizan en el mundo real afectan al mundo virtual.

En un trabajo futuro se podrían soportar dos modalidades de jugadores que coexistieran y pudieran interactuar entre ellos: los jugadores virtuales utilizarían el juego GREP desde un ordenador de sobremesa y percibirían a los jugadores reales como avatares en su juego.

Mientras tanto, los jugadores reales (aquellos con dispositivos de realidad aumentada) podrían percibir la presencia de los jugadores virtuales a través del dispositivo de realidad aumentada. Además, las acciones ejecutadas por un jugador virtual tendrían consecuencias físicas sobre el mundo real como por ejemplo el accionamiento de un interruptor virtual podría provocar la apertura de una puerta.

Si este sistema se implementara se podría utilizar en el entrenamiento de personal, en concreto con el cuerpo de bomberos como ejemplo claro.

9. Referencias

A continuación se detallan las referencias bibliográficas utilizadas.

[1] MILGRAM, Paul; KISHINO, Fumio. A taxonomy of mixed reality visual displays.*IEICE TRANSACTIONS on Information and Systems*, 1994, vol. 77, no 12, p. 1321-1329.

[2] AZUMA, Ronald, et al. Recent advances in augmented reality. *Computer Graphics and Applications, IEEE*, 2001, vol. 21, no 6, p. 34-47.

[3] AZUMA, Ronald T., et al. A survey of augmented reality. *Presence*, 1997, vol. 6, no 4, p. 355-385.

[4] JEE, Hyung-Keun, et al. An immersive authoring tool for augmented reality-based e-learning applications. En *Information Science and Applications (ICISA), 2011 International Conference on*. IEEE, 2011. p. 1-5.

[5] A Scenario-Based Virtual Environment for Supporting Emergency Training
Telmo Zarraonandia, Victor Bañuls, Ignacio Aedo, Paloma Díaz, Murray Turoff
11th International ISCRAM Conference, State College, PA, USA; 05/2014