Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Título: Implementación de un módulo de edición de escena 3D para una herramienta de creación de Centros Expositivos Virtuales.

Autor: Enier Molina Ramírez.

Tutor: Ing. Minardo Gollún González López.

Cotutor: MsC. Lidiexy Alonso Hernández.

Ciudad de la Habana

2011

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizamos al tutor Minardo Gollún González López y al Centro de Informática Industrial (CEDIN), de la Universidad de las Ciencias Informáticas, para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Enier Molina Ramírez **Minardo G. González López**

_____ _____

Firma del Autor Firma del Tutor

Agradecimientos:

A todos mis amigos que me han ayudado y soportado todos estos años, principalmente a Andy y Manuel, a mi tutor Minardo González por ayudarme en la preparación de y desarrollo de mi tesis y a todos las personas que me han ayudado en mi formación como profesional.

Dedicatoria:

A mi familia entera, principalmente a las cuatro personas más importantes de mi vida: mi madre y mi padre, que siempre me han apoyado muchísimo, se han esforzado y sacrificado para yo sea hoy lo que soy; mi hermano, que ha estado siempre cerca de mí aquí en la universidad; y mi abuelo Capitán, que ha sido mi guía desde pequeño.

Datos de contacto:

Tutor:

Nombre y Apellidos: Minardo Gollún González López

Edad: 28 años

Ciudadanía: cubano

Institución: Universidad de las Ciencias Informáticas (UCI)

Título: Ingeniero en Ciencias Informáticas

Categoría Docente: Instructor

E-mail: mgonzalezl@uci.cu

Graduado de la UCI, con cinco años de experiencia en el tema de Realidad virtual, y jefe del proyecto Paseos Virtuales en el Centro de Desarrollo de Informática Industrial.

Resumen:

Actualmente el proceso de creación de un Paseo Virtual en el CEDIN[1] involucra una gran cantidad de personal, que combinan su trabajo para dar vida al producto final. A través de la realización de este Trabajo de Diploma se logró crear una herramienta de creación de Centros Expositivos Virtuales, que hará innecesaria la intervención del programador en el proceso de creación del software. Dicha herramienta está diseñada para que cualquier persona sea capaz de crear un Paseo Virtual y editar el escenario, incorporándole modelos 3D y modificando la posición, escala, rotación e información y el tipo de dichos modelos. Todo esto a través de herramientas de fácil uso y comprensión. Además, el usuario puede establecer trayectorias para la posterior visualización del Paseo.

Mediante el uso de la solución propuesta en este Trabajo de Diploma se hará más sencillo y fácil de controlar el proceso de desarrollo de un Paseo Virtual. Además, dado que la aplicación está hecha sobre tecnologías de código abierto, los costos de producción se reducirán enormemente, contribuyendo al ahorro de recursos de nuestro centro, y por ende, de nuestro país.

[1] Centro de Informática Industrial, perteneciente a la Universidad de las Ciencias Informáticas (UCI).

Índice:

Índice de Figuras y Tablas:

Introducción

El hombre siempre se ha visto dominado por el impulso de reproducir lo que tiene alrededor, de recrear del modo más fiel posible lo que le rodea. En la era digital, el avance de los gráficos computarizados (y con ellos, de la Realidad Virtual[2]), devino en una herramienta poderosísima que probaría ser de gran ayuda a la hora de simular el mundo real de la forma más fiel posible.

Hoy en día, el motor impulsor de la realidad virtual lo constituyen los videojuegos. Estos fueron los primeros en incorporar escenarios virtuales[3] para suplir sus necesidades, a la vez de implementar módulos que permitían al jugador personalizar un escenario a su gusto, y después desarrollar su juego sobre ese escenario. Esta funcionalidad se hizo muy popular entre el público, y la mayoría de las grandes compañías desarrolladoras de juegos la adoptaron como estándar para darle más libertad de creación al usuario final.

Actualmente, las grandes empresas productoras de videojuegos implementan la funcionalidad de edición de escenarios como elemento imprescindible de sus productos. Ejemplo de ello lo constituyen Ubisoft (Far Cry2(**Ubisoft** 2010)), Electronic Arts (Crysis,) y Microsoft (Halo(**Microsoft** 2010)); dichos editores permiten lo mismo crear una misión nueva, o bien modificar los niveles originales del juego.

En nuestro país la introducción de los primeros equipos de procesamiento de datos se remonta a la década de 1920, aunque no es hasta después del triunfo de la Revolución que la informatización comienza a cobrar auge de manera significativa. A partir de entonces nuestro país ha insertado el uso de la informática en todas las esferas, explotando desde programas de contabilidad hasta diseño asistido por computadoras.

La experiencia en desarrollo de videojuegos en Cuba es prácticamente nula. La Universidad de las Ciencias Informáticas es el centro de desarrollo que hace más hincapié en este tipo de

[2] Simulación de la realidad, realizada con la ayuda de computadoras

[3] Simulación, mediante la Realidad Virtual, de un escenario, real o ficticio.

productos. Dentro de ella, el Grupo de desarrollo de Paseos Virtuales se encuentra desplegando el Paseo Virtual UCI.

Un Paseo Virtual es una simulación en tres dimensiones de un entorno que puede existir o no en el mundo real. El Paseo Virtual UCI es una aplicación que recrea la universidad en la perspectiva de un caminante; ofrece un recorrido virtual combinado con información sobre los elementos más significativos del entorno. Pero tiene la desventaja de que, una vez creado, para poder realizar algún cambio en el entorno, tiene que volverse a pasar por todo el proceso de implementación del software.

Pero, ¿qué pasa cuando el Paseo Virtual es de un lugar que cambia continuamente, digamos, por ejemplo, una galería de arte? Entonces la idea de tener que implementar un nuevo Paseo Virtual cada vez que se cambie la exposición carecería de sentido práctico. Para estos casos es necesario un editor de escenarios que, similar al de los videojuegos, permita la edición de escenarios 3D, simplificando de este modo la tarea de crear un Paseo Virtual.

Es por esto que se plantea como **problema a resolver**: ¿Cómo implementar un módulo que permita la edición de escenas 3D para una herramienta de creación de Centros Virtuales Expositivos?

Para la solución del problema se plantea como **objeto de estudio** “las herramientas de edición de escenarios virtuales” y **como campo de acción** “las herramientas de edición de escenarios virtuales 3D para la creación de Paseos Virtuales y su visualización”; teniendo como **objetivo general** “implementar el módulo de edición de escenas 3D para una herramienta de creación de Centros Expositivos Virtuales”; esperándose como posible resultado “obtener el módulo de edición de escenas 3D para una herramienta de creación de Centros Expositivos Virtuales”

Para el cumplimiento del objetivo planteado anteriormente se trazan las siguientes **tareas a desarrollar**:

- Elaboración del marco teórico relacionado con las técnicas de programación.
- Caracterización de las herramientas de manejo de escenas 3D.
- Caracterización de los Paseos Virtuales.

- Elaboración de un resumen de los modelos de Análisis y Diseño del módulo a Implementar.
- Identificación las herramientas y lenguajes para el desarrollo de la aplicación.
- Realización del Diagrama de Componentes.
- Implementación de los Casos de Usos del Módulo.
- Realización de pruebas de Caja Negra.

Con la realización de este Trabajo de Diploma se dotará a la herramienta de creación de Exposiciones Virtuales con un módulo de edición de escena 3D que permitirá editar un escenario, pudiendo incorporarle a este modelos diseñados en herramientas de diseño 3D, con herramientas de transformación para dichos modelos, de manera que el usuario pueda fácilmente diseñar un Paseo Virtual y personalice el escenario a su gusto, dejándolo listo para su posterior visualización.

Métodos del nivel teórico: posibilitaron descubrir, analizar y sistematizar los resultados obtenidos, para llegar a conclusiones confiables que permitan resolver el problema. En tal sentido se usaron:

- El Analítico – sintético se utilizó al analizar toda la información relacionada con el tema de tesis, ya que permiten la extracción de los elementos más importantes de cada documento analizado. La inducción-deducción se utilizó durante toda la investigación, para llegar a conclusiones y hacer generalizaciones.
- El histórico-lógico se utilizó al estructurar la trayectoria del objeto en el transcurso de su historia y en orden cronológico. La Modelación Analógica se utilizó para la elaboración tanto de proceso actual, como propuestas de solución.

Métodos del nivel empírico: permitieron descubrir y acumular un conjunto de datos, que sirven de base para dar respuesta a las preguntas científicas. Para ello se utilizaron:

- La observación del tamaño resultante en los ficheros de las escenas, exportados desde las herramientas de diseño y la calidad de dichas escenas en tiempo real.

Estructura del trabajo.

El Trabajo de Diploma está estructurado en 3 capítulos:

Capítulo 1. Fundamentación teórica.

Incluye un estado del arte del tema tratado a nivel internacional, nacional y de la Universidad, de las tendencias, técnicas, tecnologías, metodologías y software usados en la actualidad o en las que se apoya para la solución del problema que se enfrenta, sobre los que es necesario profundizar.

Capítulo 2. Resumen del Análisis y Diseño del módulo a Implementar.

Incluye las características del sistema, además de un resumen de los flujos de trabajo de Análisis y Diseño del mismo.

Capítulo 3. Implementación y pruebas.

Incluye los diagramas de despliegue y de componentes del flujo de trabajo de Implementación, y, del flujo de trabajo de Pruebas, los modelos de prueba, con la descripción de los casos de prueba de integración

Bibliografía consultada.

Capítulo 1. Fundamentación teórica.

Introducción al capítulo 1.

En los inicios de la producción de videojuegos, el proceso de desarrollo era muy diferente al que tenemos en la actualidad. Para diseñar el escenario, se utilizaban diferentes herramientas de modelado vinculadas a un motor gráfico[4], pero el peso fundamental recaía en la programación de dicho videojuego. A medida que se desarrolló el mercado, se fueron integrando los diversos componentes que eran necesarios para conformar el producto. Así surgen los editores de escenarios, herramientas mediante las cuales el propio jugador es capaz de crear un escenario nuevo o modificar uno existente. Estos editores de escenarios pronto trascendieron la frontera de los videojuegos, para integrarse a fines más sociales, como los Paseos Virtuales. En este capítulo se hará un estudio de los editores de escenario más usados, motores gráficos y herramientas de diseño 3D más aptas para el desarrollo de Paseos Virtuales. Además se define el ambiente y framework[5] de desarrollo que se utilizarán, y el lenguaje de programación, analizando de ellos características y ventajas que ofrecen.

1.1 Herramientas de diseño 3D utilizadas para el diseño de escenarios virtuales.

Para la modelación de escenarios virtuales se utilizan varias herramientas de diseño 3D, dependiendo del fin que se busca lograr. Por ejemplo, en el mercado de los videojuegos es muy popular el 3ds Max Studio, por sus herramientas para ello. Asimismo, el Maya es de la preferencia de los animadores y productores de cine en general. El Blender, como herramienta basada en software libre, está ganando más adeptos cada día, y se perfila como una herramienta

[4] Serie de rutinas de programación que permiten el diseño, la creación y la representación de un videojuego.

[5] Estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado.

con un futuro prometedor. A continuación se detallan las características fundamentales de dichas herramientas, por ser las más usadas en Centro de Informática Industrial (CEDIN), de la Universidad de las Ciencias Informáticas

1.1.1 Autodesk 3D Studio Max.

Las herramientas de modelado, animación, rendering[6], y composición del 3ds Max (como también se le conoce) están diseñadas para desarrolladores de juegos, artistas de efectos visuales, y diseñadores gráficos que trabajen con juegos, aunque el software es también usado por productores de televisión y cine. Estas son sus características principales:

Modelado 3D:

Tiene uno de los paneles de herramientas más ricos en contenido que hay actualmente en la industria:

Creación eficiente de objetos orgánicos y paramétricos con características de modelado de polígono, spline(línea), y basado en NURBS[7]. Más de 100 herramientas de diseño de modelado poligonal y libre presentes en el set de herramientas Graphite.Control preciso del número de caras y vértices en los objetos con tecnología ProOptimizer y reducción de complejidad de una selección de hasta un 75% sin pérdida de detalle.(**Autodesk.com** 2011)

3ds Max SDK:

El SDK (Software Developer Kit) puede ser usado para extender e implementar prácticamente cada aspecto de la aplicación 3ds Max, incluyendo la geometría de la escena, controladores de animación, efectos de cámara y atmosféricos, crear nuevos componentes de escena, controlar el comportamiento de componentes existentes, y exportar los datos de la escena a un formato personalizado. Los desarrolladores pueden explotar un nuevo cargador administrado de plug-ins

[6] Creación y representación como imagen 2D, de los modelos gráficos en una escena.

[7] Acrónimo inglés de la expresión *Non Uniform Rational B-splines.* Es un modelo matemático muy utilizado en la computación gráfica para generar y representar curvas y superficies.

.NET, haciendo más fácil desarrollar plug-ins en C# u otros lenguajes .NET. (**Autodesk.com** 2011)

Renderizador de Hardware Quicksilver:

Un innovador, nuevo renderizador basado en hardware que puede producir imágenes de alta calidad a altas velocidades. Este nuevo motor de render multi-hilo presente en 3ds Max 2011 usa tanto la unidad de procesamiento de la computadora (CPU, como la unidad de procesamiento de gráficos (GPU), y soporta elementos de render alpha y z-buffer; profundidad de campo, motion blur (desenfoque de movimiento), reflexiones dinámicas, oclusión de área, fotométrica y ambiental, efectos de iluminación indirecta junto a mapas de sombras de precisión adaptativa; y la capacidad de producir imágenes a altas resoluciones.(**Autodesk.com** 2011)

1.1.2 Autodesk Maya.

Maya, aunque es desarrollado también por Autodesk, está más orientado a la producción audiovisual. La diferencia comienza en la interfaz visual, que está optimizada para esos fines, y continúa en las diferentes herramientas que, aunque comparten el mismo productor, tienen equipos de desarrollo que optimizan más en cada versión las características del programa. A continuación se detallan algunas de estas características:

Interfaz de usuario mejorada:

Este software de animación está disponible en versiones para Mac OS X, Windows de 32 y 64 bits, y Linux de 64 bits, con una interfaz que ofrece elementos anclables y editores más flexibles. Además, un nuevo examinador de nodos dentro del Hypershade que muestra los nodos por categoría en una vista de árbol con búsqueda y que provee más fácil acceso a los nodos usados frecuentemente.(**Autodesk.com** 2011)

Modelado 3D:

Ofrece un set poderoso para el modelado 3D, incluyendo NURBS, Superficies de Subdivisión, y un conjunto de herramientas para polígonos.(**Autodesk.com** 2011)

Animación general:

Maya entrega una amplia gama de herramientas especializadas para la animación por keyframes[8], procedural, y a base de scripts, incluyendo:

Un sistema de capas altamente controlable, no destructivo, que funciona con cualquier atributo del Editor de Animaciones No Lineales Trax, para mezcla, conbinación, y edición no destructiva de poses y clips animados.

Una herramienta Establecer Llave Conducida que permite hacer fotogramas clave relaciones complejas entre parámetros animados.

Editores Graph y Dopesheet que proveen funciones precisas de curvas para controlar cómo los atributos animados cambian en el tiempo.

Un conjunto de deformadores para modelado estático o animación.(**Autodesk.com** 2011)

Herramientas de administración de datos y escenas:

A medida que la complejidad de las escenas se incrementa, Maya provee herramientas y flujos de trabajo para administrar más eficientemente grandes conjuntos de datos.

Un grafo de dependencia permite al artista ver y editar las relaciones entre los nodos. Mediante las referencias de valores y archivos el artista puede también segmentar escenas para mejor rendimiento y para administrar flujos de trabajo colaborativos e iterativos.

El historial de construcción editable y animable permite hacer modificaciones amplias de datos modelados sin necesidad de reconstruirlos.

A través del Render Proxy, que está dentro del software de rendering mental ray, los elementos de la escena pueden ser reemplazados por una malla simple, de baja resolución: los datos pretraducidos son cargados sólo cuando se requieran para renderizar.(**Autodesk.com** 2011)

[8] Fotograma clave, término utilizado para animación.

1.1.3 Blender.

Blender es un proyecto de la Blender Foundation, alternativa de software libre que va ganando adeptos a medida que mejora y se le añaden nuevas herramientas. A pesar de ser un software relativamente joven, se perfila como una buena alternativa, avalado por la libre distribución de sus licencias. A continuación se detallan algunas de sus características:

Interfaz:

Disposición de ventanas flexible y completamente configurable con tantas configuraciones de pantalla como el usuario prefiera.

Capacidad de deshacer en todos los niveles.

Cualquier espacio de ventana puede ser cambiado a cualquier tipo de ventana (editor de curvas, NLA, vista 3D, etc.).

Editor de texto interno para anotaciones y edición de scripts[9] de Python.

Interfaz gráfica para scripts de Python.

Interfaz consistente a través de todas las plataformas.(**Blender.org** 2011)

UV Unwrapping:

Métodos de unwrapping conformales y basados en ángulos.

Edición de degradado interactivo de mapas UV para transformaciones suaves.

Unwrapping basado en costuras (seams).

Proyecciones de vista cúbica, cilíndrica y esférica.

Subdivisión Catmull-Clark para menor distorsión de los UVs.

Capas UV múltiples. (**Blender.org** 2011)

Creación de juegos 3D en tiempo real:

Editor gráfico lógico para definir comportamiento interactivo win tener que programar.

[9] Programa usualmente simple, que por lo regular se almacena en un archivo de texto plano.

Detección de colisiones y simulación de dinámicas soportado para Bullet. Bullet es una librería para detección de colisiones y dinámicas de código abierto desarrollada para Play Station 3.
API para scripting de Python con control e inteligencia artificial, lógica de juego avanzada completamente definida.
Soporta todos los modos de iluminación de OpenGLTM, incluyendo transparencias, y texturas animadas y mapeadas por reflexión.
Soporte para multimateriales, multitexturas y modos de fusión de texturas, iluminación per-pixel, dinámica, modos de mapeo, fusión de texturas vertexPaint GLSL, toon shading, materiales animados, mapeo Normal y Parallax.
Audio, usando el toolkit SDL. (**Blender.org** 2011)

Modelado:
Superficies de subdivisión Catmull-Clark con isolíneas óptimas.
Capacidad de modelado multirresolución con pinceles procedurales de mapas 2D y 3D(Paint, Smooth, Pinch, Inflate, Grab), con soporte para simetrías.
Modificadores de deformación: Lattice, Curve, Armature y Displace.
Modificador Mirror con clipping para los vértices del centro y eliminación automática de caras interiores.
Acceso a scripting Python para personalizar las herramientas. (**Blender.org** 2011)

1.2 Motores gráficos.

El motor gráfico (graphic engine, en inglés), es el componente de software principal de un videojuego o de otra aplicación interactiva que se ejecute en tiempo real. Su uso simplifica el desarrollo de la aplicación y a menudo permite que el juego pueda correr en múltiples plataformas, tales como Consolas de videojuegos y Sistemas Operativos de Mac OS, GNU/Linux y Microsoft Windows. La abstracción del hardware es una de las principales ventajas que posee el Motor. Un Motor Gráfico ofrece un conjunto de herramientas de desarrollo, además de componentes de software reutilizables, agrupados en subsistemas que presentan alta cohesión

en relación a sus comportamientos, los subsistemas más comunes son: procesamiento de entradas, gráficos, animación, audio, comportamiento e inteligencia artificial, y conectividad / red, entre otros.

Se toman como objeto de análisis G3D, y Ogre por su popularidad y funcionalidad, adquiridas no sólo en nuestra universidad, sino, también a nivel mundial. También se hace referencia al SceneToolKit, por ser una herramienta nacional, desarrollada y mejorada en nuestro centro.

1.2.1 G3D.

G3D (GraphicsThree Dimensional Engine) es un motor gráfico escrito en C++, que ha sido utilizado en una amplia gama de aplicaciones, entre ellas los videojuegos. Es importante destacar que las APIs gráficas de bajo nivel como OpenGLy Direct3D son muy sencillas de utilizar, no significando esto último una pérdida o limitación en el rendimiento del producto final. (**SourceForge.net** 2010)

G3D es compatible con varios sistemas operativos como Windows, Linux y OS X. Además de su diseño orientado a objetos y el soporte de varias extensiones de imágenes como JPG, TGA y PNG; G3D integra dentro de sí el trabajo con shaders y diversas técnicas de dibujado como Shadow Shadow Volumes que aportan sin duda un toque de realismo al entorno virtual.(**SourceForge.net** 2010)

Principales características:

- Soporte para modelos 3DS, IFS, MD2, BSP, PLY2, OFF.
- Soporte para imágenes JPG, PNG, BMP, PPM, PCX, TGA, DDS, e ICO.
- MP4, MPG, MOV, AVI, DV, QT, WMV, video
- Themed GUI and fontrendering
- Administración de memoria automática de forma opcional.
- Red basada en los protocolos TCP y UDP.
- Optimización de cálculos con matrices.
- Compatible con Visual C++, XCode y gcc.
- Amplia documentación que incluye también programas de ejemplo.

1.2.2 Ogre.

OGRE es un motor gráfico de código abierto, multiplataforma escrito en C++. Cuenta con una gran comunidad internacional y una amplia documentación; además de permitir el acople de nuevos plugins incrementando su flexibilidad.(**Ogre3D.org** 2009)

A continuación se expondrán alguna de sus principales características.

Texturas:

- TextureMapping. Cálculo automático de Mip-Maps.
- Texturas multinivel en un solo paso.
- Texturas animadas.

Cálculo procedural de coordenadas de textura.

Iluminación:

- Luces puntuales, direccionales y de foco.
- Nieblas.
- Luces dinámicas, de colores con sombras suaves.
- Radiosidad precalculada sobre los ligthmaps.

Modelado:

- Mallas poligonales con soporte de LODs discretos.
- Superficies de Bezier.
- Grafo jerárquico de escena.
- Edición externa mediante 3D Studio MAX.
- Modelado de terreno.
- Importación de niveles de Quake3.
- Terrains.

Estructura:

La estructura de Ogre, como se puede ver en la figura 1 da al desarrollador un fácil manejo de los plugins, además de ser una estructura bien modulada.

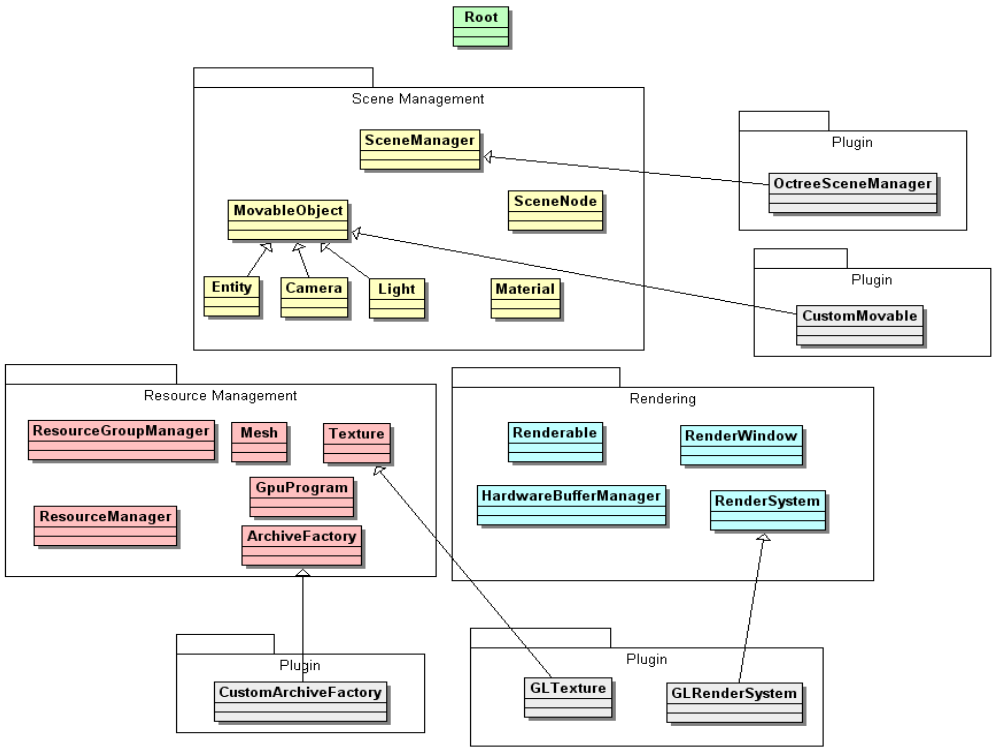


Fig. 1 Estructura de Ogre(Ogre3D.org 2009)

1.2.3 SceneToolKit (STK).

La SceneToolkit[10] es una herramienta que abstrae al desarrollador del trabajo con APIs de bajo nivel como son OpenGL y DirectX, facilitando de esta manera la representación de entornos virtuales y simulaciones. Hasta su última versión publicada, dicha herramienta no contaba con un motor de render que tuviera una estructura bien definida además de carecer de las técnicas más recientes que permiten un incremento considerable del rendimiento.(**Quintana López** and **Alonso Monteagudo** 2010)

El sistema de render actual basa el dibujado de todas sus geometrías en vertexarrays, lo cual representa un paso de avance con respecto al modo inmediato de render (inmediatemode);

[10] Toda la información referente a la SceneToolKit es tomada de documentos del expediente de proyecto.

además el procesamiento innecesario de cambios en los estados de render se traduce en un decremento en cuanto a la eficiencia del sistema. Otra deficiencia lo constituye la no posibilidad de integración de shaders dentro del sistema; que si bien estos se han tratado de incluir en versiones no estables, no han sido establecidos de forma definitiva.(**Quintana López** and **Alonso Monteagudo** 2010)

Por otra parte la SceneToolKit brinda soporte para las APIsOpenGL y DirectX, además de ser compatible con múltiples sistemas operativos (Windows y Linux) facilitando en cada uno de ellos la interacción con ventanas y periféricos (mouse, teclado, joystick). Su arquitectura dividida por módulos permite la integración y/o acoplamiento de nuevos módulos sin necesidad de una restructuración a gran escala del sistema completo.(**Quintana López** and **Alonso Monteagudo** 2010)

Clases más significativas.

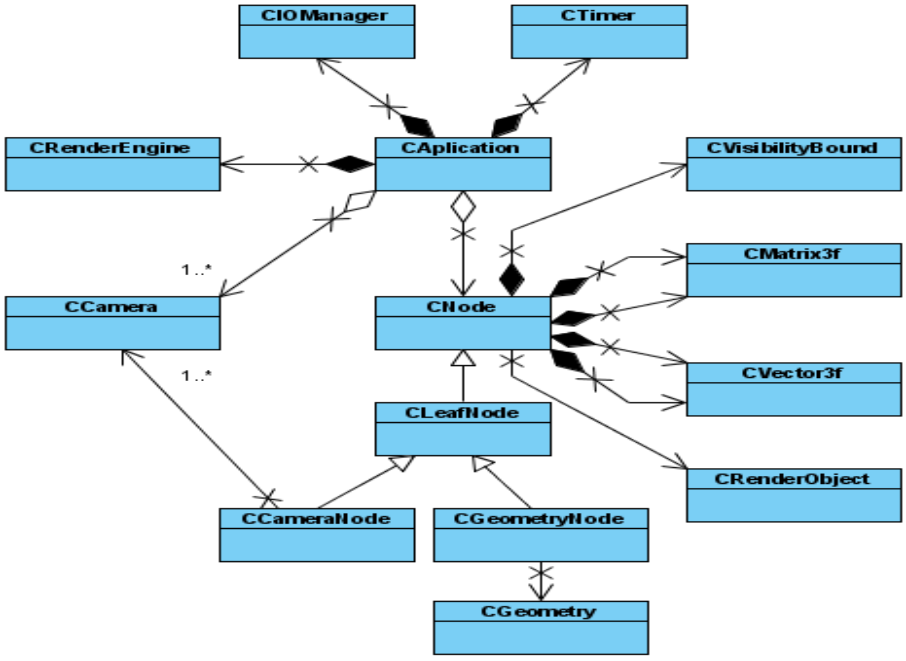


Fig. 2 Principales clases del SceneToolKit.

1.3 Entornos de Desarrollo Integrado.

Un entorno de desarrollo integrado (en inglés Integrated Development Environment o IDE) es un programa compuesto por una serie de herramientas que utilizan los programadores para desarrollar código. Esta herramienta puede estar pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de estos. Las herramientas que normalmente componen un entorno de desarrollo integrado son las siguientes: un editor de texto, un compilador, un intérprete, unas herramientas para la automatización, un depurador, un sistema de ayuda para la construcción de interfaces gráficas de usuario y, opcionalmente, un sistema de control de versiones.(**Escuela Universitaria de Ingeniería Técnica en Informática de Oviedo** 2009)

1.3.1 Microsoft Visual Studio.

Visual Studio es un conjunto completo de herramientas de desarrollo para la generación de aplicaciones Web ASP.NET, Servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Visual Basic, Visual C++, Visual C# y Visual J# utilizan el mismo entorno de desarrollo integrado (IDE), que les permite compartir herramientas y facilita la creación de soluciones en varios lenguajes. Asimismo, dichos lenguajes aprovechan las funciones de .NET Framework, que ofrece acceso a tecnologías clave para simplificar el desarrollo de aplicaciones Web ASP y Servicios Web XML.(**Microsoft**)

Acelera de manera significativa la producción de software y su documentación está entre las mejores. La interfaz es altamente amigable con el usuario, permitiendo que el tiempo en implementar una solución o aplicación determinada sea mucho menor. Los ejecutables desarrollados por esta herramienta son generalmente de menor tamaño, lo que hace que ocupe un lugar cimero en la producción de software al lograr aplicaciones óptimas y de poco volumen.

1.3.2 Qt Creator.

Qt Creator es un entorno de desarrollo multiplataforma de código abierto muy completo, hecho para desarrollar aplicaciones en C++ de manera sencilla y rápida. Como su nombre lo indica, está basado en la librería Qt y cuenta con las siguientes características:

- Editor de código C++ y Javascript.
- Diseñador de interfaz de usuario integrado.
- Herramientas de administración y construcción de proyectos.
- Debuggers DGB y CDB.
- Soporte para control de versiones.
- Simulador para interfaces de dispositivos móviles.
- Soporte para destinos móviles y de escritorio. (**Nokia** 2011)

Qt Creator es distribuido bajo tres tipos de licencias: Qt Commercial Developer License, Qt GNU LGPL v. 2.1, Qt GNU GPL v. 3.0 y está disponible para las plataformas: Linux, Mac OSX; Windows, Windows CE, Symbian y Maemo.

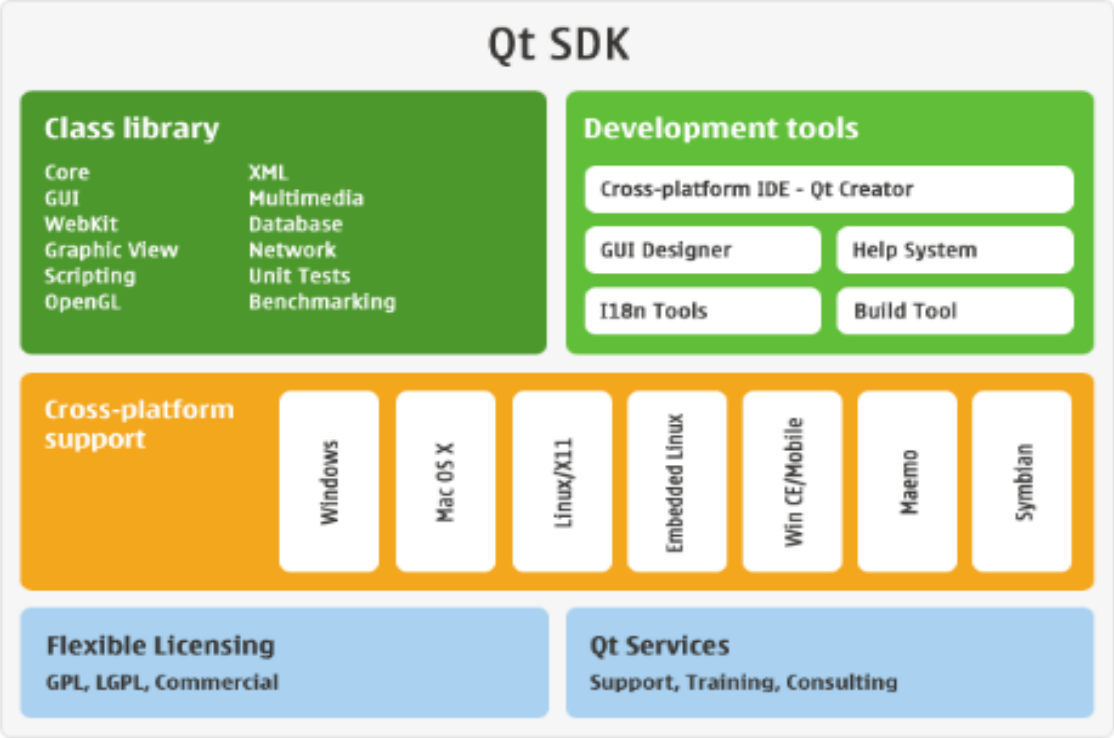


Fig. 3 Características del SDK Qt

1.3.3 Eclipse.

El IDE Eclipse para desarrolladores de C/C++ es de código abierto y corre sobre la Plataforma Eclipse. Provee funcionalidades avanzadas para desarrolladores de C/C++, que incluyen un editor (con realzamiento de sintaxis y completamiento de código), launcher, debugger, un motor de búsquedas y generador de ficheros.(**Eclipse.org** 2011) No incluye compilador ni debugger, es necesario instalarlos de forma independiente.

1.4 Lenguajes de modelado, programación y framework.

1.4.1 Lenguaje Unificado de Modelado (UML).

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified, Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; aun cuando todavía no es un estándar oficial, está respaldado por el OMG (Object Management

Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional, o RUP, por sus siglas en inglés), pero no especifica en sí mismo qué metodología o proceso usar (**RedIRIS.com** 2009). En nuestro caso, es el lenguaje que se utilizó en el flujo de diseño de la herramienta, por eso su obligatoria utilización.

1.4.2 C++.

C++ es un lenguaje orientado a objetos de probada eficacia para generar aplicaciones de alto rendimiento como las aplicaciones gráficas. Aporta un nivel de productividad muy superior a C, sin comprometer la flexibilidad, el rendimiento o el control. Es uno de los lenguajes de sistemas más conocido en el mundo, altamente compatible, y que soporta las bibliotecas gráficas Glide, OpenGL, DirectX y G3D que son muy utilizadas en la industria de los videojuegos. Al hacer uso del paradigma de la programación orientada a objetos, son considerables las ventajas que este lenguaje ofrece.(**Rodríguez Noa** and **Gómez Finalé** 2008)

1.4.3 Framework Qt.

Qt es una aplicación multiplataforma y un framework para interfaces de usuario (UIs, por sus siglas en inglés). A través de ella, se pueden escribir aplicaciones web y desplegarlas para escritorio, sistemas operativos móviles e integrados, sin tener que reescribir el código fuente.

Incluye una librería de clases de C++, portabilidad para sistemas operativos integrados y de escritorio, herramientas para desarrolladores con un IDE multiplataforma, y un alto rendimiento en tiempo real.(**Nokia** 2011)

1.5 Editores de Escenarios.

Los Editores de Escenarios, también conocidos como editores de mapas, son las herramientas que se usan para diseñar los escenarios (niveles, misiones, mapas). Estos surgen para darles más libertad a los usuarios finales que no estén conformes con los escenarios predefinidos que traiga el videojuego o simplemente quieran desarrollar su juego sobre configuraciones de escenario personalizadas. Actualmente los creadores de videojuegos lanzan un editor de escenario (ver figura 4, 5 y 6) del mismo videojuego y este aparece formando parte integral del mismo.

1.5.1 Cryengine Sandbox.

Sandbox es el editor de niveles usado para crear niveles en la línea de juegos CryEngine, producidos por Crytek. Dentro del software se proveen herramientas para facilitar el scripting, animación y creación de objetos. Ha sido incluido en varios juegos de Crytek como Crysis y FarCry y es usado extensivamente con propósitos de modding.

El estilo de edición es un concepto del Sandbox con énfasis en grandes terrenos y estilo libre para programación de misiones. El editor también puede construir escenas de interiores.

De forma opuesta a editores como Unreal Editor que usan un estilo de edición “substractivo” que separa áreas de un escenario lleno, el Sandbox tiene un estilo “aditivo” como el Quake 2. Los objetos son adicionados a un entorno global vacío.

La concentración del Sandbox en un terreno potencialmente enorme (en teoría cientos de kilómetros cuadrados), significa que usa una forma algorítmica de pintar texturas y objetos en el paisaje, usa varios parámetros para definir la distribución de texturas o los tipos de vegetación. Esto está hecho con el propósito de salvar tiempo al hacer la edición de tan grandes terrenos a la vez de que se mantiene el estilo libre del escenario. Esto es diferente de algunos estilos de edición que casi siempre usan falsas panorámicas para dar la ilusión de grandes escenarios.

En una forma de algún modo comparable al renderizador 3D Bender, que puede ser usado para el diseño de juegos, el editor Sandbox tiene la habilidad de que con solo presionar una tecla se puede jugar instantáneamente el escenario en tiempo real(WYSIWYP, "What You See Is What You Play", “Lo que ves es lo que juegas”).

El editor es compatible con todas las funcionalidades que brinda el motor gráfico CryEngine como los vehículos, la física, la secuencias de comandos, iluminación avanzada, sombras en movimiento, cinemáticas, shaders, audio 3d, animaciones, partículas, mapa de normales, oclusión de mapas, e inteligencia artificial, entre otras; todo esto en tiempo real de edición del escenario. (**Crytek** 2008)

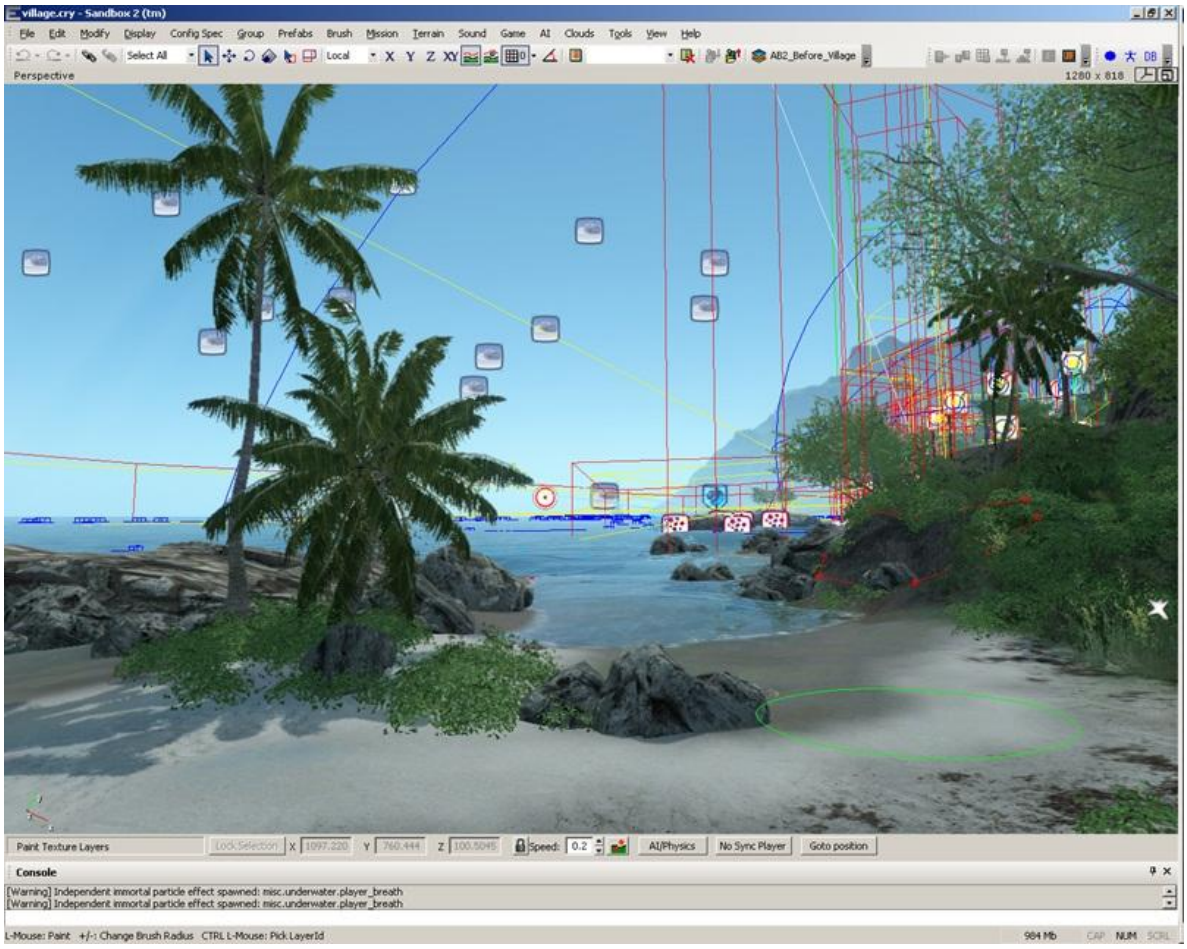


Fig. 4 Interfaz del CryEngine SandBox

1.5.2 Warcraft III World Editor.

El Warcraft III World Editor es el editor de niveles del juego de estrategia Warcraft III: Reign of Chaos y su expansión Warcraft III: The Frozen Throne, hechos por la compañía Blizzard. Basado en el editor de niveles de su anterior juego Starcraft, permite a los jugadores crear y personalizar sus propios escenarios y campañas (incluso mapas personalizados) con un alto nivel de detalle y flexibilidad. El editor ha sido usado en la creación de mapas personalizados como el popular DoTA(Defense of the Ancients). Ha sido sustancialmente mejorado en la expansión del juego, y permite crear escenarios con soporte de voz. Características avanzadas permiten el acceso a modelos personalizados, texturas e iconos. El editor también soporta el lenguaje de scripting JASS de Blizzard para programar acciones complejas que no son accesibles mediante la interfaz gráfica.

El editor está dividido en diferentes módulos que permiten al usuario personalizar diferentes características de un mapa.

Editor de Terreno:

Permite al usuario editar el terreno, facilitando la edición mediante una paleta de edición provista de islas, agua, árboles, luces avanzadas, pueblos, etc.

Editor de Objetos:

El Editor de Objetos permite al usuario modificar la mayoría de los atributos de los objetos ubicables en el juego, como la apariencia y habilidades de las unidades, edificios e ítems. También permite crear objetos personalizados basados en otros ya existentes.

Editor de disparadores:

El Trigger Editor, como también se le conoce, permite crear triggers (disparadores), que se disparan cuando ciertos eventos ocurren en el juego e importan su propio código JASS, lo cual es más fácil para la mayoría de las personas. Construido sobre JASS, mediante él se pueden añadir poderosos aditamentos al juego.

Administrador de importaciones:

Permite a los usuarios importar modelos personalizados, texturas, mini mapas, pantallas de carga y sonidos a un mapa.

Editor de IA (inteligencia artificial):

Permite al usuario crear scripts de IA para jugadores de la computadora. Aunque es recomendable usarGUI o JASS para hacerlo.

Editor de campañas:

Permite crear campañas personalizadas combinando mapas salvados en la computadora. Incluye también un editor de objetos, para que múltiples mapas puedan usar las mismas unidades.(**Hiveworkshop.com** 2010)

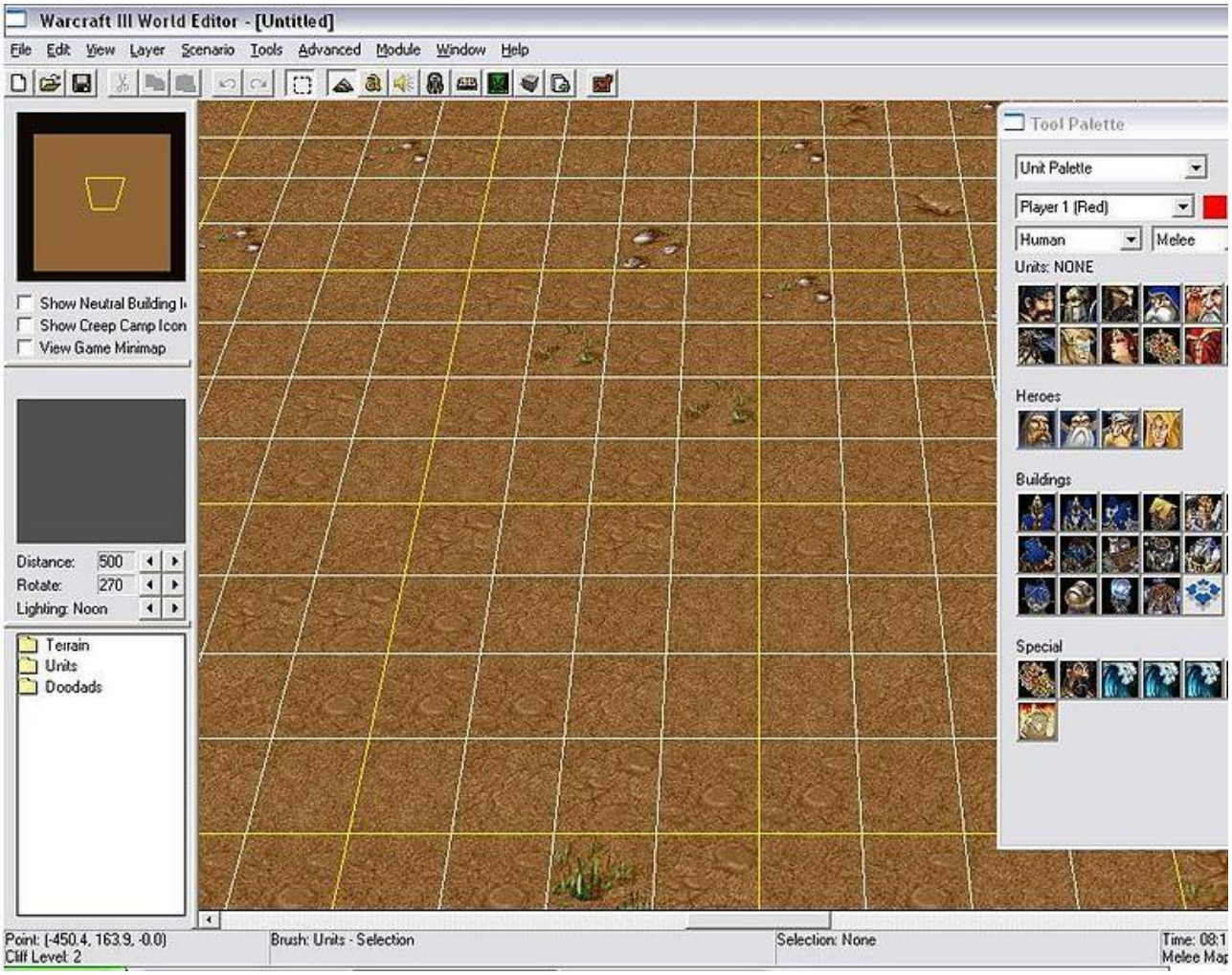


Fig. 5 Vista de la interfaz del Warcraft III World Editor

1.5.3 Unreal Level Editor.

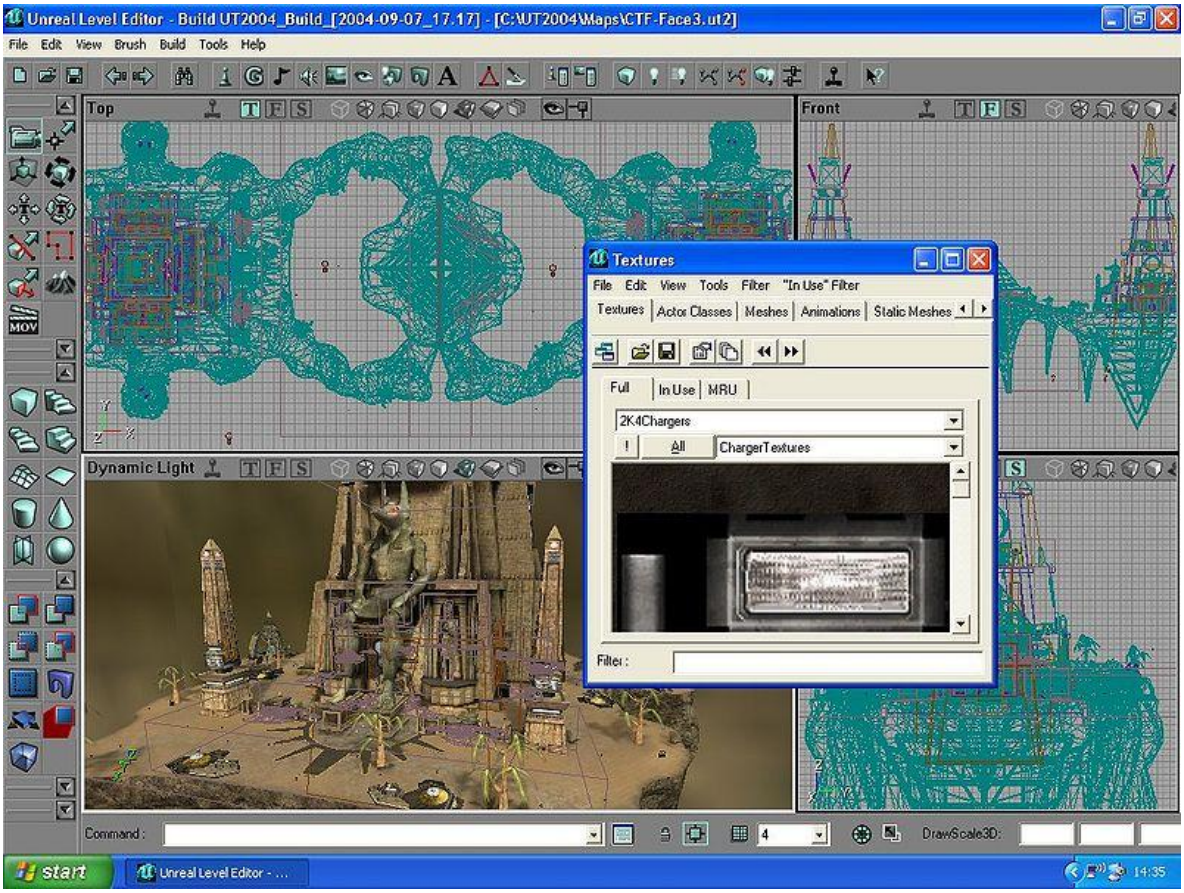
El Unreal Editor (o UnrealEd) es una herramienta de creación de contenidos del tipo “Lo que ves es lo que obtienes”, llenando el vacío entre herramientas como 3ds Max, Maya, y XSI, y el contenido de juegos. El editor es una suite de variadas herramientas para creación contenidas en el Unreal Engine. Sus **características** incluyen:

- Diseño visual de los niveles, colocación, edición y organización de los objetos en el escenario, los artículos de inventario, NPCs (non-player character, carácter no jugador), rutas de acceso y luces.
- Rendering en tiempo real interactivo de los niveles durante la edición, incluida la iluminación dinámica y sombras. Todo el contenido de nivel en el editor se presenta en consistencia con el motor del juego.
- Construido sobre un framework de edición de propiedades guiado por datos, permite a los diseñadores de niveles personalizar fácilmente cualquier objeto del juego, y a los programadores exponer nuevas propiedades personalizables a través de scripts.
- Cargar y editar grandes escenarios compuestos por varios subniveles.
- Herramienta compatible con 64 y 32 bits, permitiendo cargar muy grandes escenarios haciendo mejor uso del hardware.
- En el editor, mediante el botón “Play Here”, el diseñador puede probar lo que está haciendo en un editor, a la vez que modifica objetos y reacomoda la geometría en otro.
- Diseño arquitectónico del sistema basado en pinceles para creación rápida y shelling de escenarios.
- Usa múltiples núcleos de procesamiento para acelerar el trabajo, como generación de luces estáticas e interacción de sombras.
- Vistas previas del juego, con gráficos precisos, vistas previas instantáneas de iluminación

También incluye un set completo de **herramientas** para facilitar el trabajo **de edición**, tales como:

- **Content Browser:** organizador/administrador de objetos.
- **Unreal Kismet**, un lenguaje visual de scripting para juegos.
- **Unreal Matinee**, que permite entre otras cosas crear cinemáticas dentro del juego.
- **Terrain Editor,** herramienta de creación de terrenos.
- **Material Editor**, herramienta visual para diseño de materiales y shaders.
- **Mesh Editor**, para ver previamente las mallas y ajustar las propiedades físicas.

- **Unreal Facade**, herramienta de diseño visual para creación de edificios y arquitecturas de otros tipos.
- **Unreal PhAT(Physics Assets Tool),** herramienta para la creación de sistemas de física para caracteres y objetos.
- **FaceFX Studio**, conjunto de herramientas para animación facial.
- **Importador de archivos FBX** de Autodesk, capaz de importar mallas, animaciones, morphings , materiales, etc.
- **Editor de post-procesamiento**, utilizado para encadenar efectos de post-procesamiento tales como desenfoques de movimiento y profundidad de campo.(**UnrealTechnology** 2008)



1.6 Fig. 6 Vista de la interfaz del Unreal Level Editor

Consideraciones del capítulo.

A través del estudio de este capítulo es posible obtener los conocimientos necesarios para comprender la esencia de la solución que se da en este trabajo al problema de la falta de una herramienta de edición de escena 3D para una herramienta de creación de Centros Expositivos Virtuales.

Se hizo un estudio del estado del arte referente a nuestra solución: las herramientas de diseño 3D más utilizadas actualmente para el diseño de escenarios virtuales, los principales motores gráficos utilizados para la visualización de escenarios, así como las herramientas de programación y modelado del negocio necesarias para llevar a cabo una aplicación funcional. Asimismo se estudiaron los principales editores de escenario de juegos presentes en la industria, para sacar de ellos las herramientas que podían ser utilizadas para la creación de un Centro Expositivo Virtual.

Capítulo 2. Resumen del Análisis y Diseño del módulo a Implementar.

Introducción al capítulo 2.

En el presente capítulo se hará un resumen de la fase de Análisis y Diseño, cuyos resultados le fueron entregados al autor para la implementación del módulo. Se definirán los requisitos, tanto funcionales como no funcionales, que deberá cumplir el módulo, así como los Casos de Uso correspondientes al diseño del mismo, cuyos diagramas se presentan. Por último, se presentarán los modelos conceptuales, dígase diagramas y descripción de clases, realización de Casos de Uso y prototipo de interfaz.

2.1 Especificación de los requisitos de software.

Los requisitos son una especificación de lo que el sistema debe cumplir para satisfacer al cliente. Estos pueden dividirse en requisitos funcionales y requisitos no funcionales. En los próximos epígrafes se especifican los requisitos tanto funcionales como no funcionales para cada uno de los módulos que conforman la herramienta que se propone.

2.1.1 Requisitos funcionales.

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir

Tabla 1. Requisitos funcionales del módulo Editor.

N^{o}	Nombre	Descripción
RF 1	Administrar proyecto.	Maneja las funcionalidades Crear, Salvar Eliminar y Cargar un paseo virtual.
RF 1.1	Crear proyecto.	Crea una carpeta con el nombre del paseo virtual a diseñar, dentro de esta se crea una carpeta que contendrá los recursos (modelos gráficos, cámaras, recorridos, luces) del paseo virtual y el fichero principal del paseo. El fichero y la carpeta de los recursos tendrán el mismo nombre.

RF 1.2	Salvar proyecto.	Guarda los cambios realizados al paseo cargado en la carpeta creada según la distribución descrita en el requisito RF 1.1.
RF 1.3	Eliminar proyecto.	Elimina el proyecto, la carpeta del paseo con el fichero principal y todos sus modelos gráficos.
RF 1.4	Cargar proyecto.	Carga el proyecto seleccionado en el editor para luego realizar cambios.
RF 2	Gestionar modelos del sistema.	Adiciona o elimina modelos al sistema.
RF 2.1	Adicionar modelo al sistema.	Adiciona un modelo al sistema.
RF 2.2	Eliminar modelo del sistema.	Elimina un modelo del sistema. Antes de eliminarlo se debe preguntar al usuario si realmente está seguro de eliminarlo definitivamente del sistema.
RF 3	Administrar objetos del entorno.	Maneja las funcionalidades relacionadas con los objetos decorativos o expositivos que se mostrarán dentro del entorno.
RF 3.1	Cargar objeto en el entorno.	Carga un objeto y lo muestra en el área de visualización.
RF 3.2	Mover objeto del entorno.	Mover un objeto hacia cualquier área válida en el entorno.
RF 3.3	Alinear objetos.	Al mover un objeto se mostrará una línea que indica la posición del objeto seleccionado con respecto a otro objeto cercano, permitiendo alinearlo con otro objeto del escenario.
RF 3.4	Activar sombra.	Muestra u oculta la sombra de un objeto en el entorno.
RF 3.5	Cambiar vista del objeto.	Cambia el ángulo de vista de todos los objetos. Las vistas pueden ser Frente, Arriba, Derecha, Izquierda y Libre en Perspectiva.
RF 3.6	Cambiar modo de	Cambia la forma de visualizar los objetos gráficos de la

	visualización.	escena. La forma de visualización puede ser: Solido, Puntos o Malla.
RF 3.7	Eliminar objeto del entorno.	Elimina el objeto seleccionado del entorno y del área de visualización.
RF 4	Modificar objeto decorativo.	Modifica el tamaño de un objeto decorativo y la cantidad que puede existir en el paseo.
RF 4.1	Escalar objetos decorativos.	Aumenta o disminuye el tamaño de un objeto decorativo.
RF 4.2	Duplicar objetos decorativos.	Realiza una copia de un objeto decorativo en el entorno.
RF 5	Modificar información de los objetos.	Crea y modifica la información referente a un objeto y permite clasificarlo en decorativo o expositivo.
RF 5.1	Actualizar información de objeto.	Modifica la información asociada a un objeto.
RF5.2	Definir tipo de objeto.	Clasifica el objeto en decorativo o expositivo.
RF 6	Administrar sonido	Adiciona un sonido al entorno, permitir la reproducción recursiva, asociarlo a un objeto o escenario y da la posibilidad de activarlo o desactivarlo.
RF 6.1	Adicionar sonido.	Adiciona un sonido al sistema.
RF 6.2	Eliminar sonido.	Elimina un sonido del sistema.
RF 6.3	Repetir sonido.	Realiza la reproducción recursiva de un sonido.
RF 6.4	Asociar sonido a la escena.	Asigna un sonido a un entorno.
RF 6.5	Quitar sonido de la escena.	Quita un sonido asociado a un entorno.
RF 7	Administrar cámara	Adiciona una cámara al paseo, permite modificar su altura y

		velocidad así como definir una trayectoria y eliminarla del paseo.
RF 7.1	Adicionar cámara.	Adiciona una cámara al paseo.
RF 7.2	Modificar altura.	Aumenta o disminuye la altura de una cámara.
RF 7.3	Definir trayectoria.	Define el recorrido que realizara la cámara en dentro del paseo.
RF 7.4	Eliminar cámara.	Elimina la cámara del paseo.
RF 7.5	Eliminar Trayectoria.	Elimina la trayectoria definida por el actor.
RF 8	Detectar colisiones.	Detecta la cercanía entre objetos.
RF 9	Exportar paseo.	Guarda todos los cambios realizados al paseo y se guarda en la carpeta del visualizador.
RF 10	Listar recursos.	Lista los nombres de los proyectos y los nombres de los modelos que se encuentren en el sistema.

2.1.2 Requisitos no funcionales.

Los requisitos no funcionales son propiedades o cualidades que el producto debe cumplir. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Para la herramienta a desarrollar se especifican los siguientes requisitos no funcionales:

Requisitos no funcionales del módulo Editor.

1. Usabilidad.

RnF 1.1 Utilización del sistema. Los usuarios del sistema serán diseñadores de Paseos Virtuales con conocimiento de cómo rotar, mover y escalar modelos gráficos en alguna de las herramientas de diseño de modelos 3D.

2. Portabilidad.

RnF 3.1 Ejecución de la herramienta. La herramienta permitirá ser compilada y ejecutada sobre la plataforma Windows y Linux.

3. Restricciones de diseño y la implementación.

RnF 3.1 Lenguajes de programación. Se utilizará el lenguaje de programación C++.

RnF 3.2 Herramientas de desarrollo. Para la implantación del sistema se utilizará la herramienta de desarrollo QT Creator.

RnF 3.3 Herramientas de diseño gráfico. Se utilizará la herramienta Blender para la modelación de gráficos.

RnF 3.4 Implementación del sistema. Se regirá por la filosofía de Programación Orientada a Objetos y se utilizará el motor gráfico Ogre.

4. Hardware

RnF 4.1 Instalación de la herramienta. Para la instalación del software la computadora deberá cumplir con los siguientes requisitos:

- Memoria RAM de 1 Gb o Superior.
- Velocidad de Microprocesador a 3.0 GHz o superior.
- Recursos de Video de 256 Mb o superior.

5. Interfaz.

RnF 5.1 Interfaz de Usuario. Las opciones de servicios además de tener su icono identificador tendrá el texto que muestre la opción en cuestión con su función descrita en una frase breve para un reconocimiento rápido por el usuario.

6. Ayuda y documentación en línea.

RnF 6.1 Manual de ayuda. Deberá contar con un manual de ayuda para comprensión de los servicios que brinda la aplicación.

2.2 Definición de los casos de uso.

Una vez recopilados los requisitos se pueden definir los casos de uso los cuales facilitaran una descripción de cómo el sistema se usará. El caso de uso describe la manera en que los actores interactúan con el sistema.

Definición de los actores.

Un actor representa papeles que las personas (o dispositivos) juegan como impulsores del sistema.

Tabla 2. Descripción de los actores que interactúan con la herramienta.

Actores	Justificación
Diseñador del Paseo.	Es la persona que interactúa con el módulo Editor de la herramienta para crear o editar un paseo virtual a un centro expositivo.

Listados de casos de uso (CU):

CU-1. Administrar proyecto.
CU-2. Gestionar modelos del sistema.
CU-3. Administrar objetos del entorno.
CU-4. Modificar objeto decorativo.
CU-5 Modificar información de los objetos.
CU-6. Administrar sonido.
CU-7. Administrar cámara.
CU-8. Detectar colisiones.
CU-9. Exportar paseo.
CU-10. Listar recursos.

2.2.1 Diagrama de casos de uso.

La representación de los diagramas de casos de uso permite especificar la funcionalidad y el comportamiento de un sistema mediante su interacción con los actores del sistema.

En las siguientes figuras se muestra la vista de casos de uso para el módulo Visualizador que representan los casos de usos arquitectónicamente significativos. En el Anexo 1 del documento se podrán observar los diagramas de casos de uso representando todos los casos de uso.

Fig. 7. Diagrama de casos de uso arquitectónicamente significativos del módulo Editor.

2.2.2 Descripción textual de los casos de usos.

Mediante la descripción textual de un caso de uso se describe paso a paso la secuencia de eventos que los actores realizan para completar un proceso a través del sistema. Se especifican las acciones que realizan los actores sobre el sistema y la respuesta que surge como consecuencia de cada evento. Seguidamente se muestran las descripciones textuales de los casos de usos arquitectónicamente significativos. Pueden verse en el Anexo 2 las descripciones textuales de todos los casos de uso definidos para la herramienta.

Tabla 3. Casos de uso expandidos para el módulo Editor.

Caso de uso	
CU-1	Administrar proyecto.
Propósito	El objetivo del caso de uso es que el diseñador del paseo pueda crear un proyecto, salvar un proyecto, cargar un proyecto en escena para luego editarlo o eliminar un proyecto de la carpeta del sistema.
Prioridad	Crítico.
Actores: Diseñador del Paseo.	
Resumen: El diseñador del paseo inicializa el caso de uso cuando selecciona una de las acciones siguientes: Crear Proyecto, Salvar Proyecto, Cargar Proyecto o Eliminar Proyecto. Al seleccionar la opción Crear Proyecto se crea una carpeta con el nombre del proyecto en la cual se salvará el fichero principal del paseo y una carpeta que contendrá los recursos (cámara, luces, recorridos, modelos) del paseo virtual. El fichero y la carpeta de recursos tendrán el mismo nombre. Los cambios efectuados al proyecto serán guardados en la capeta especificada. El caso de uso permite además eliminar el paseo virtual diseñado.	
Referencias	RF 1, RF 1.1, RF 1.2, RF 1.3, RF 1.4.
Precondiciones	No puede existir un proyecto con el mismo nombre.

Poscondiciones	Se crea una carpeta con el nombre del proyecto en la cual se salva el fichero principal del paseo y una carpeta que contendrá los modelos del paseo virtual. El fichero y la carpeta que almacena los modelos son salvados con el mismo nombre. Se muestra en pantalla el área de diseño y los objetos almacenados en el sistema. En caso de haber seleccionado la opción eliminar proyecto, se elimina el proyecto seleccionado de la carpeta del sistema.
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador selecciona unas de las opciones siguientes: Crear Proyecto, Salvar Proyecto o Eliminar Proyecto.	1.1 Si el diseñador selecciona la opción Crear Proyecto ir al Escenario 1 Crear Proyecto. 1.2 Si el diseñador selecciona la opción Salvar Proyecto ir al Escenario 2 Salvar Proyecto. 1.3 Si el diseñador selecciona la opción Eliminar Proyecto ir al Escenario 3 Eliminar Proyecto.
Escenario 1 Crear Proyecto	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**

1. El diseñador elige la opción Crear Proyecto. 2. El diseñador teclea el nombre del proyecto. 3. El diseñador selecciona la opción aceptar.	1.1 El sistema muestra una venta al diseñador para que teclee el nombre del proyecto a crear. 3.1 El sistema verifica la existencia de un proyecto con igual nombre al que introduce el diseñador. 3.2 Crea una carpeta con el nombre del proyecto, dentro de esta carpeta crea el fichero principal del paseo y una carpeta que contendrá los modelos del paseo virtual. El fichero y las carpetas son creados con el mismo nombre del proyecto. Finaliza el caso de uso.
Curso Alterno de Eventos	
Acción del actor	**Respuesta del sistema**
3.3 El diseñador inserta un nombre de proyecto que ya existe en el sistema. 3.4 El diseñador selecciona la opción cancelar.	3.3 El sistema muestra el mensaje “Ya existe un Proyecto con este nombre...” indicando al diseñador que ya el proyecto existe en el sistema. 3.5 El sistema cancela la operación, culmina así el caso de uso.
Escenario 2 Salvar Proyecto	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador elige la opción Salvar Proyecto.	1.1 Guarda los cambios realizados al paseo en la carpeta creada según la distribución descrita en el Escenario 1. Finaliza así el caso de uso.
Escenario 3 Eliminar Proyecto	

Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador elige la opción Eliminar Proyecto. 2. El diseñador selecciona la opción aceptar.	1.1 El sistema muestra el siguiente mensaje de alerta: ¿Desea eliminar el proyecto del sistema? 2.1 El sistema elimina el proyecto, la carpeta del paseo con el fichero principal y todos sus modelos, cámara, recorridos luces. Finaliza el caso de uso.
Curso Alterno de Eventos	
Acción del actor	**Respuesta del sistema**
2. El diseñador elige sobre la opción cancelar.	2.1 El sistema cancela la operación y culmina así el caso de uso.
Escenario 4 Cargar Proyecto	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador selecciona un paseo virtual de la lista de paseo y elige la opción Cargar Paseo.	1.1 El sistema muestra en el editor el paseo seleccionado con todos sus recursos (cámara, modelos, recorridos, luces).

2.3 Modelos conceptuales de la herramienta.

El modelo conceptual que se muestra a continuación es una forma representar los conceptos significativos asociados a la propuesta de solución para su mejor comprensión.

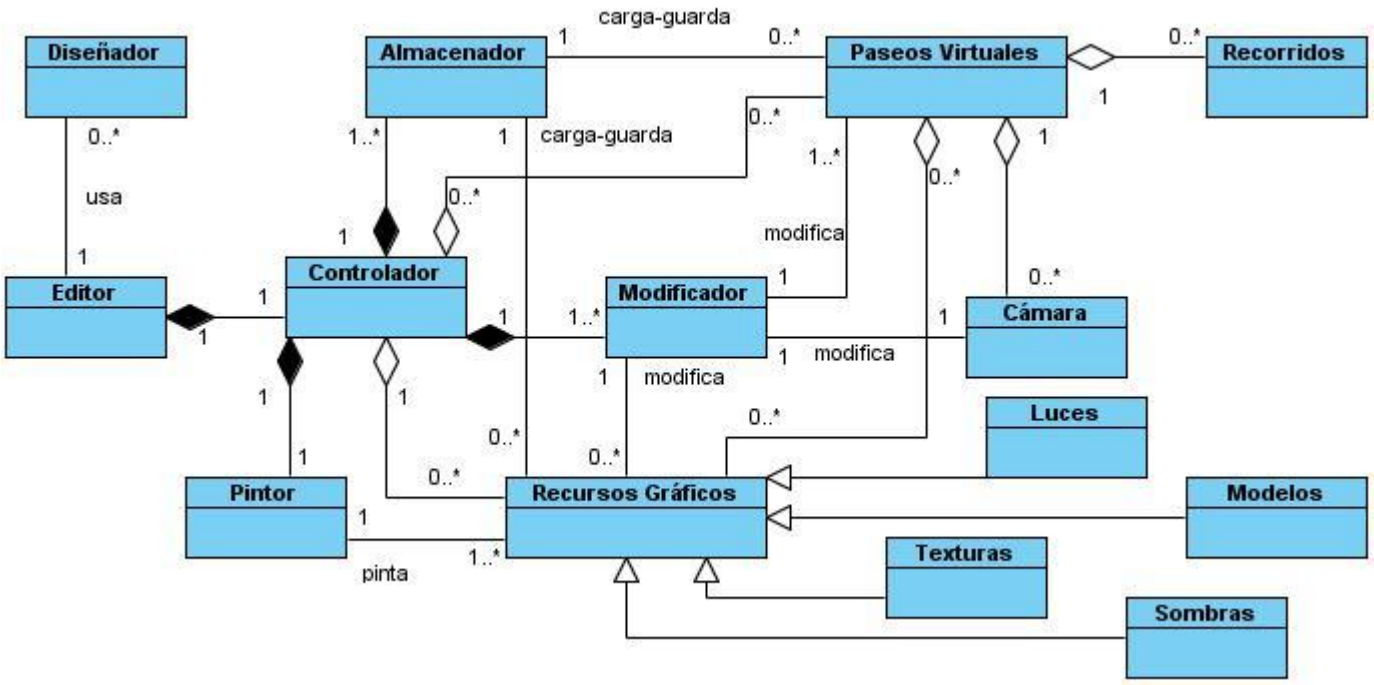


Fig. 8. Modelo conceptual del módulo Editor.

2.3.1 Diagramas de clases.

Los diagramas de clases se utilizan para mostrar clases y sus relaciones, para saber las clases que participaran en la realización de un caso de uso y las responsabilidades que deberán cumplir. El diagrama de clases del diseño incluye más detalles que el diagrama de clases del modelo de análisis lo cual es necesario para la adaptación del modelo de diseño al entorno de la implementación. Las clases que intervienen en la realización de los casos de uso de los módulos de la herramienta estarán estructuradas según el patrón tres capas, ya que permite simplificar y comprender el desarrollo del sistema reduciendo la dependencia de forma que las capas más bajas no sean conscientes de ningún detalle de las superiores.

Clases que se encuentran en la capa Presentación.

Clases que se encuentran en la capa Aplicación.

Clases que se encuentran en la capa Acceso a Datos.

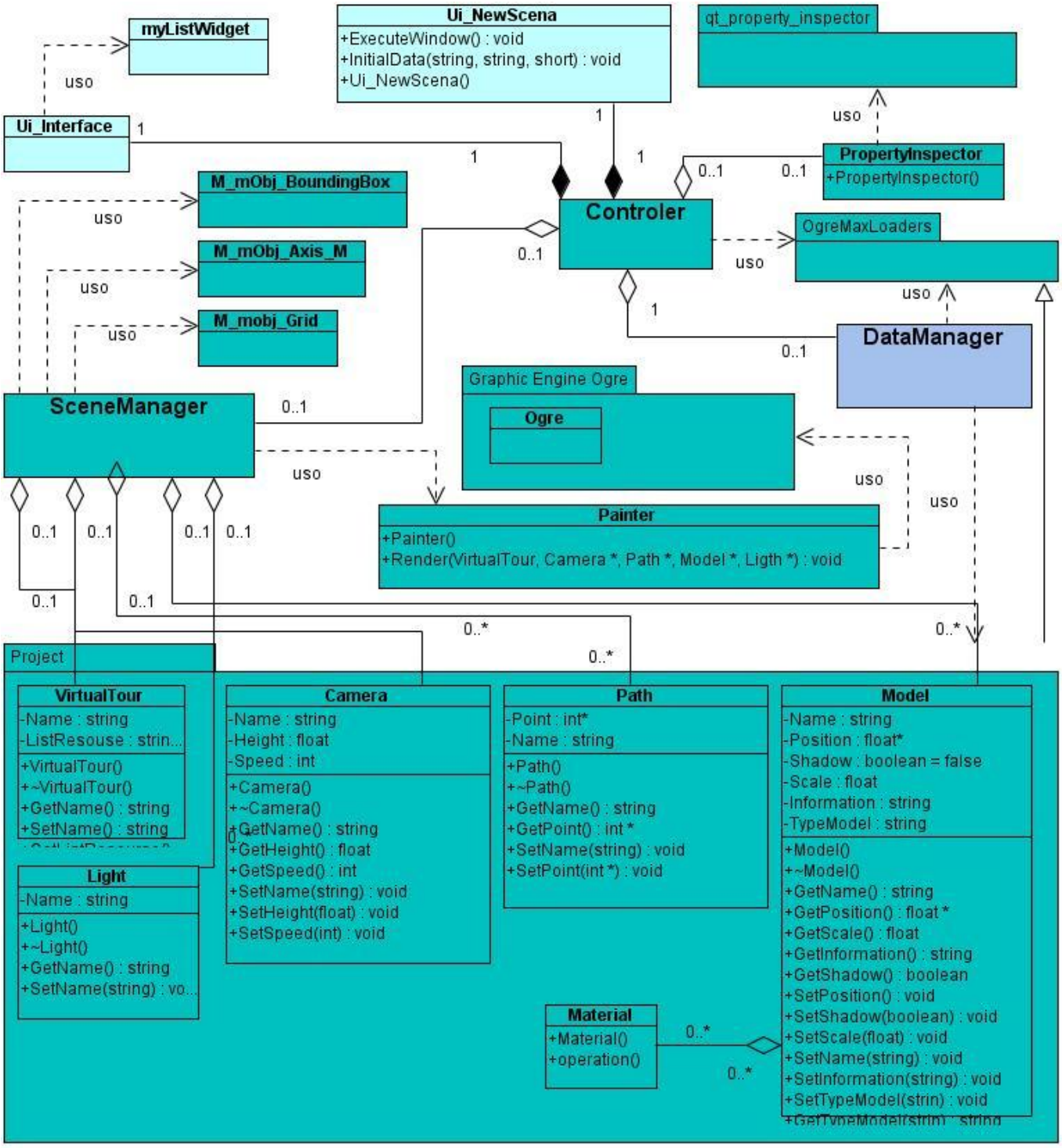


Fig. 9. Diagrama de clases del diseño del módulo Editor.

Los diagramas de clases del diseño que se muestran en la Figura 9 representan las clases, sus relaciones y sus responsabilidades.

2.3.2 Descripción de las clases.

A continuación se realiza una breve descripción de las clases arquitectónicamente significativas para el diseño de la herramienta. Clases arquitectónicamente significativas del módulo Editor.
Clases arquitectónicamente significativas del módulo Editor.
“Controler”. Clase controladora que tiene la responsabilidad de enviar las respuestas a las peticiones que realiza el usuario hacia la clase Interfaz. Contiene todas las funcionalidades que dan respuesta a los casos de uso del sistema.
“SceneManager”.Clase controladora que tiene la responsabilidad de administrar los datos de las entidades que maneja el sistema.
“DataManager”. Clase controladora que tiene la responsabilidad de administrar los ficheros que almacenan los datos persistentes del sistema.

2.3.3 Realización de casos de uso.

La realización de los casos de uso se describe durante el análisis y el diseño a través de los diagramas de colaboración y los diagramas de secuencia. Los diagramas de colaboración se utilizan en el modelo de análisis mostrando cómo el control pasa de un objeto a otro a medida que se lleva a cabo el caso de uso, y los mensajes que se envían entre los objetos. El nombre del mensaje indica el motivo del objeto que realiza la llamada en su interacción con el objeto invocado. La realización de un caso de uso en el modelo de diseño describe como se realiza el caso de uso en términos de las clases de diseño correspondientes. Para modelar las interacciones entre los objetos del diseño se utiliza el diagrama de secuencia. En las siguientes figuras se representan los diagramas de secuencia de los casos de uso arquitectónicamente significativos que fueron realizados durante el diseño. Para cada escenario de un caso de uso se tiene un diagrama de secuencia.

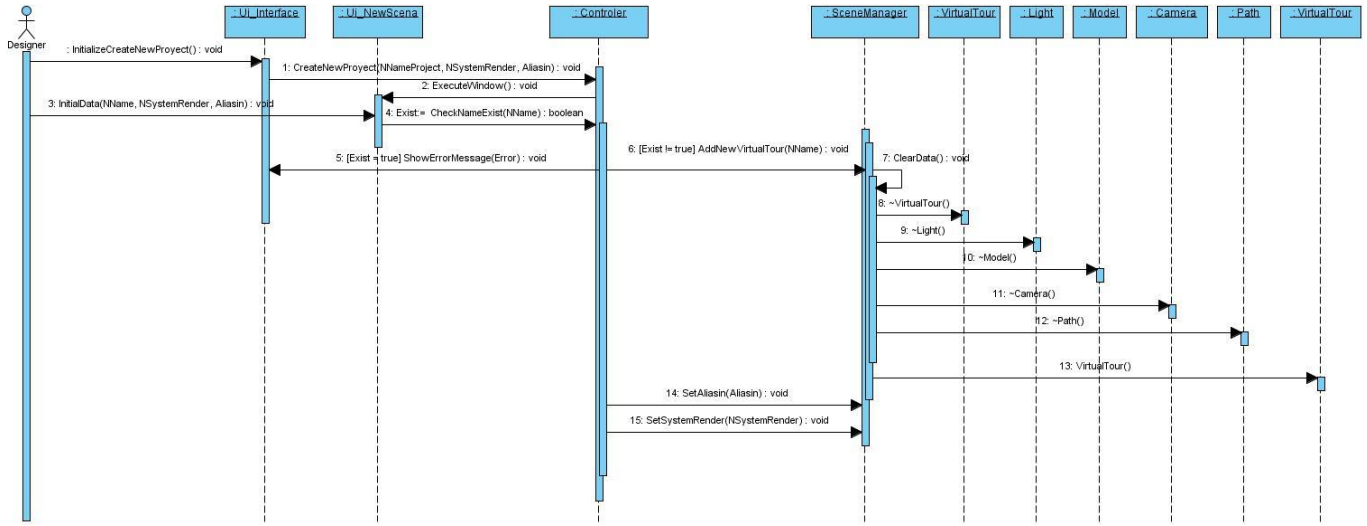


Fig. 10. Diagrama de secuencia CU-1 Administrar Proyecto Escenario Crear Proyecto.

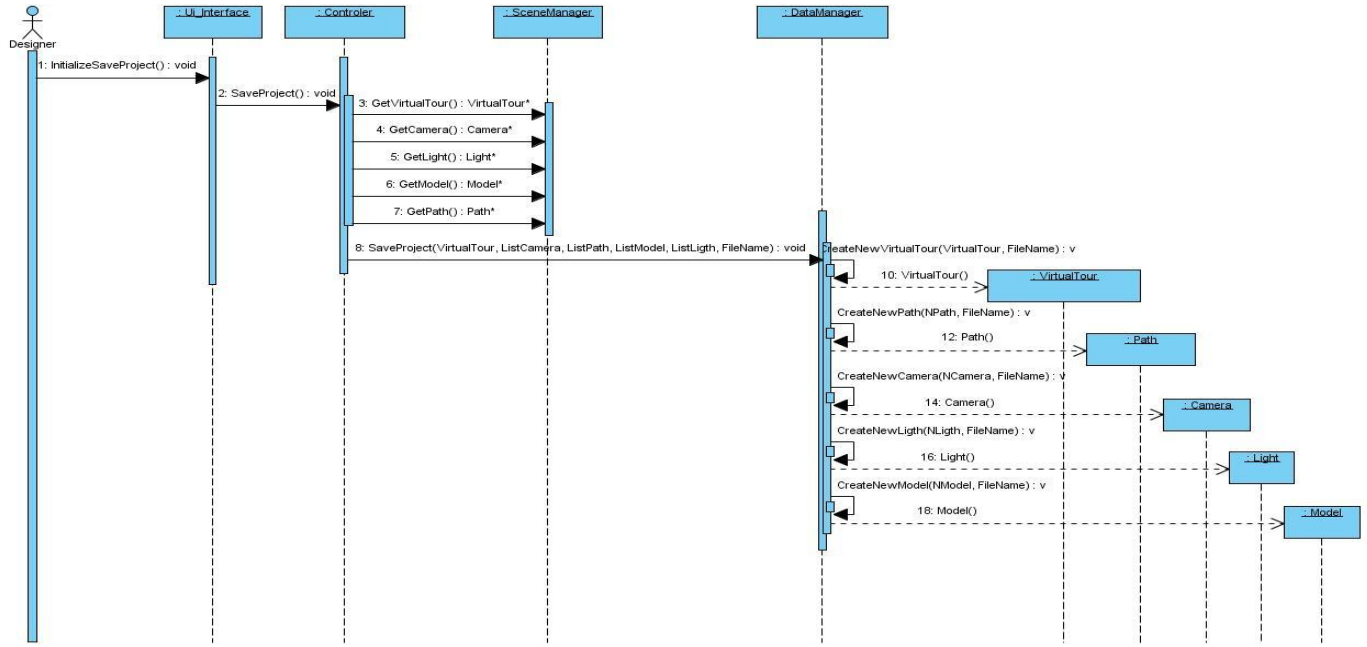


Fig. 11. Diagrama de secuencia CU-1 Administrar Proyecto Escenario Salvar Proyecto.

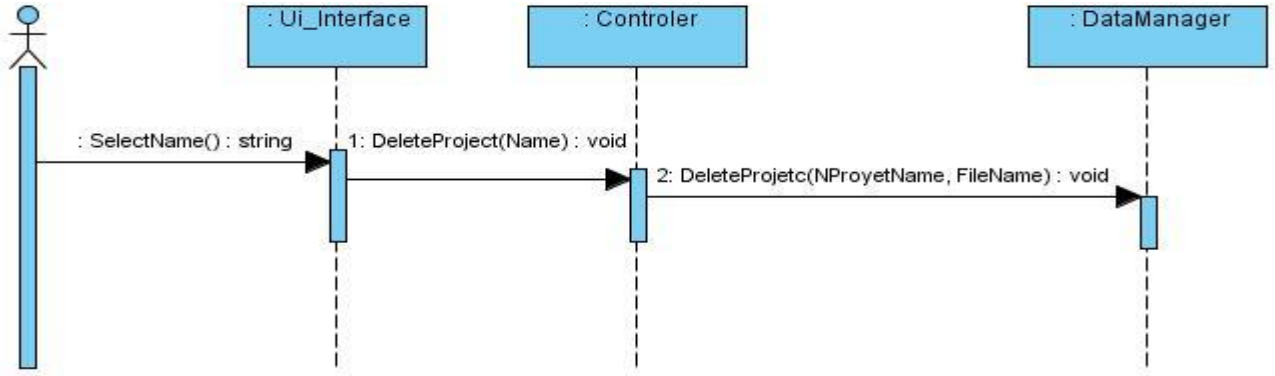


Fig. 12. Diagrama de secuencia CU-1 Administrar Proyecto Escenario Eliminar Proyecto.

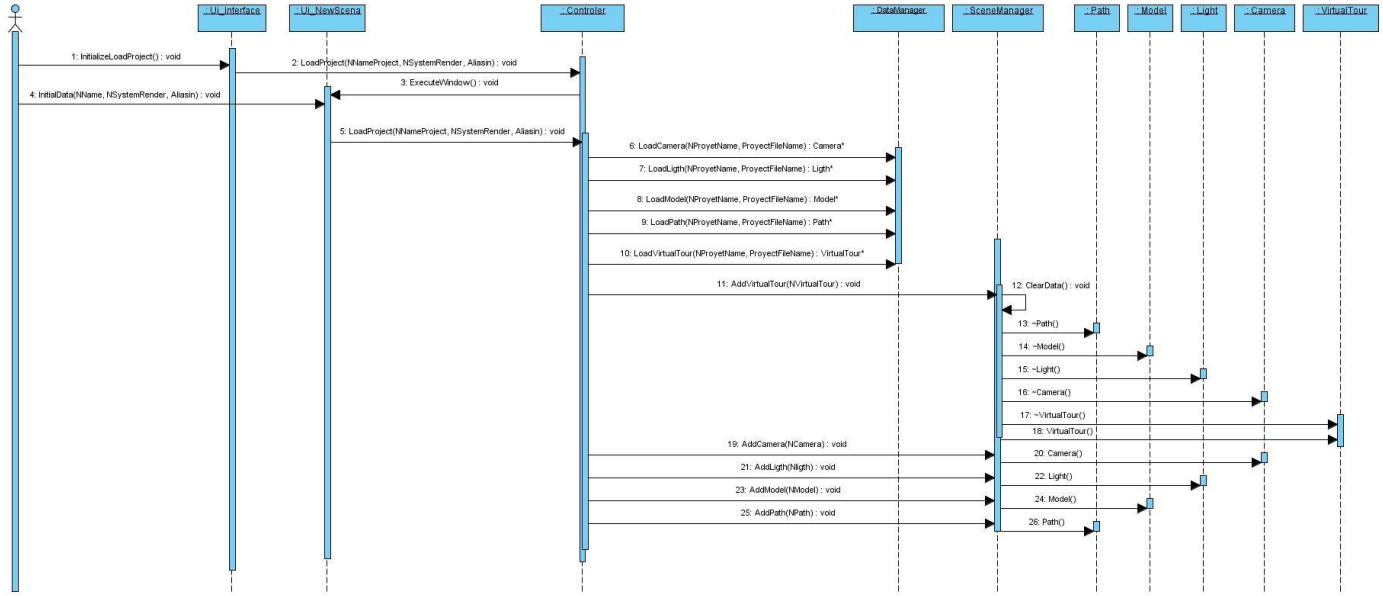


Fig. 13. Diagrama de secuencia CU-1 Administrar Proyecto Escenario Cargar Proyecto.

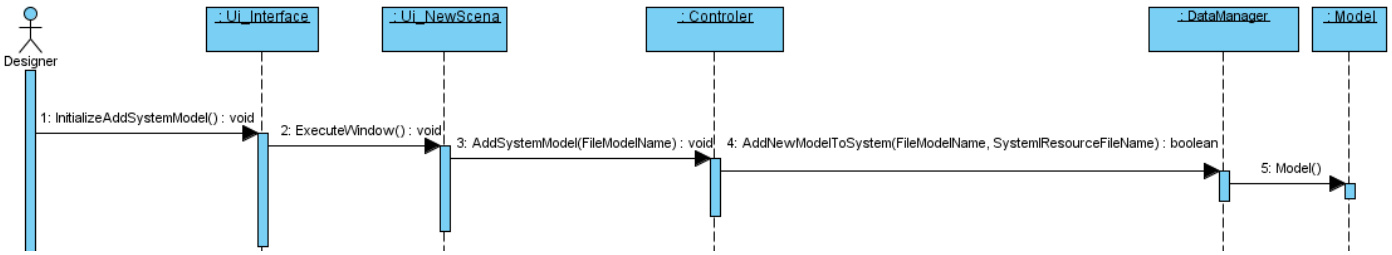


Fig. 14. Diagrama de secuencia CU-2 Gestionar Modelos del Sistema Escenario Adicionar Modelo.

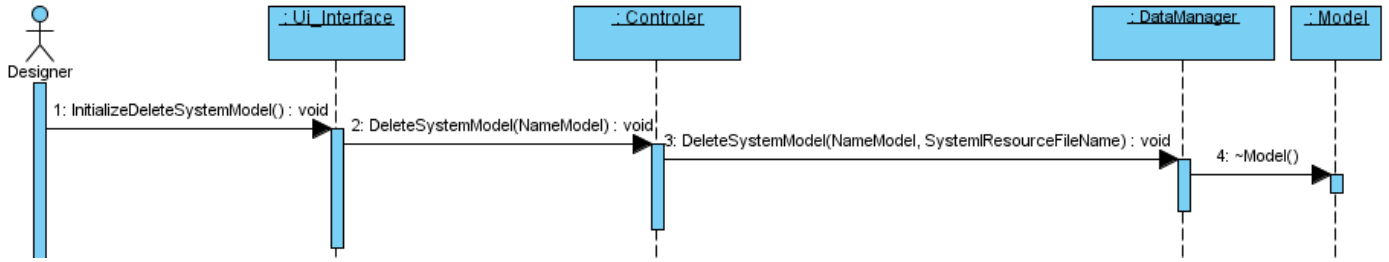


Fig. 15. Diagrama de secuencia CU-2 Gestionar Modelos del Sistema Escenario Eliminar Modelo.

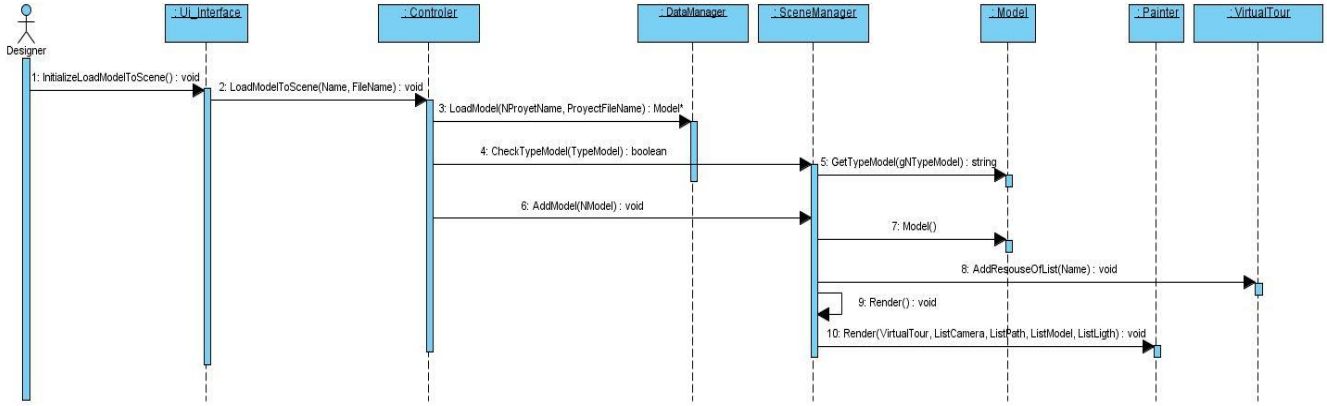


Fig. 16. Diagrama de secuencia CU-3 Administrar Objetos del Entorno Escenario Cargar Objeto en Entorno.

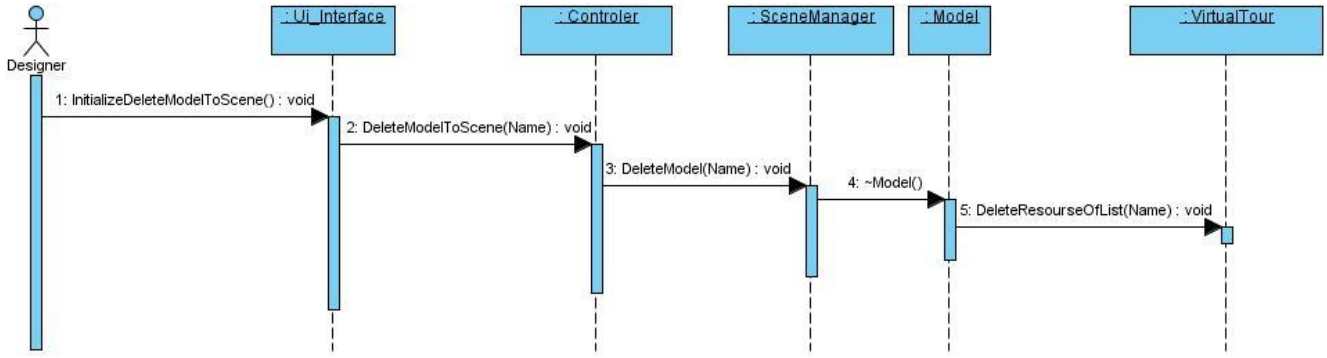


Fig. 17. Diagrama de secuencia CU-3 Administrar Objetos del Entorno Escenario Eliminar Objeto de Entorno.

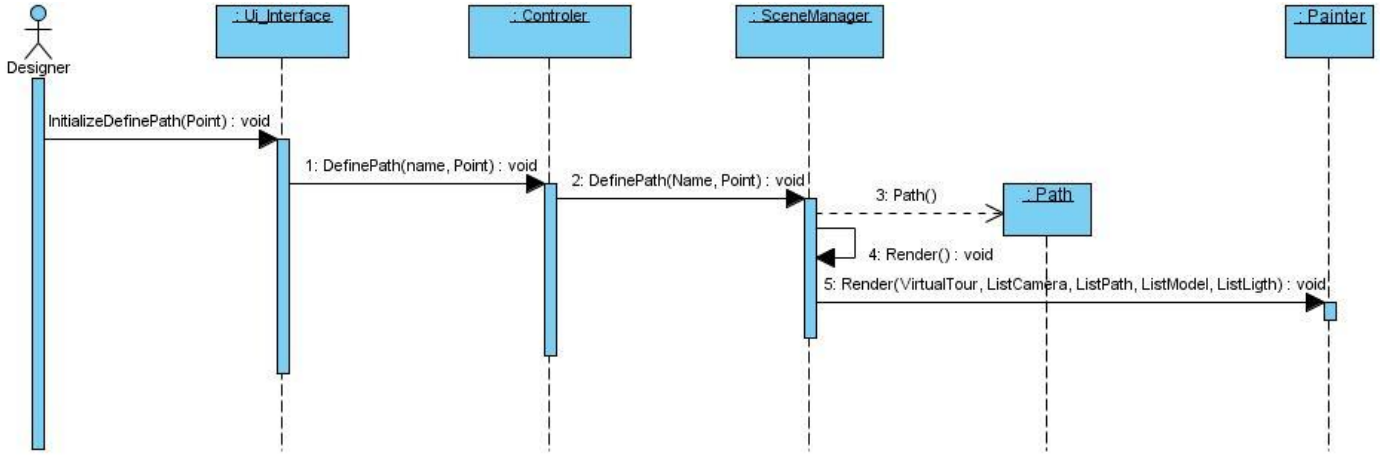


Fig. 18. Diagrama de secuencia CU-7 Administrar Cámara Escenario Definir Trayectoria.

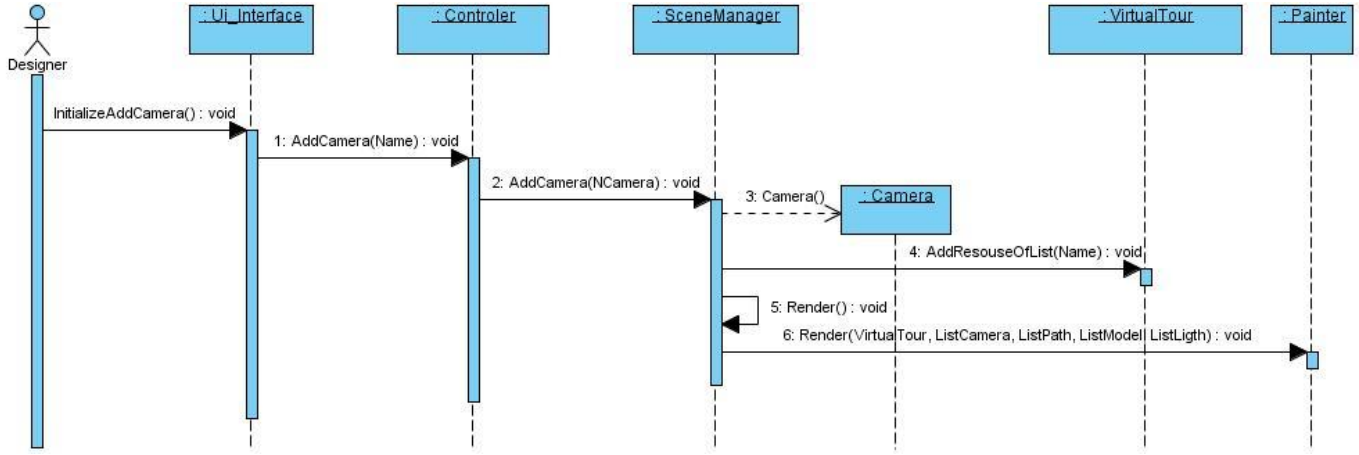


Fig. 19. Diagrama de secuencia CU-7 Administrar Cámara Escenario Adicionar Cámara.

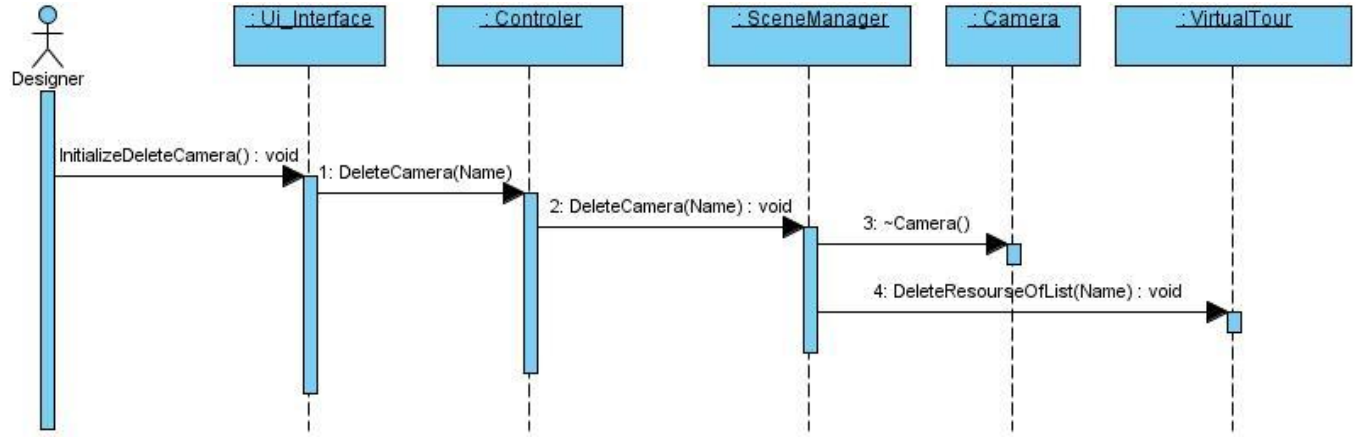


Fig. 20. Diagrama de secuencia CU-7 Administrar Cámara Escenario Eliminar Cámara.

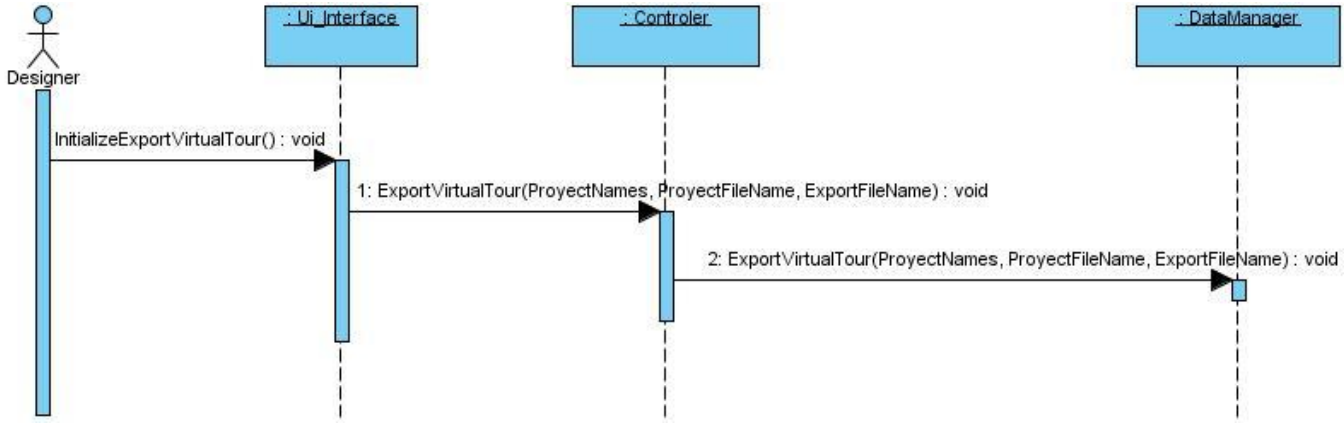


Fig. 21. Diagrama de secuencia CU-9 Exportar Paseo.

2.3.4 Prototipo de interfaz.

El diseño de interfaces de usuario es una tarea que ha adquirido relevancia en el desarrollo de un sistema. Un prototipo de interfaz representa un conjunto de elementos que le permiten al usuario interactuar con el sistema. Durante el presente epígrafe se muestra la interfaz desarrollada para el módulo de edición teniendo en cuenta los requisitos y funcionalidades entregadas como parte de la fase de Análisis y Diseño y se describen los elementos de la interfaz que le dan respuesta a los casos de uso arquitectónicamente significativos. En las figuras se resaltan mediante un

recuadro los elementos que dan respuesta a los casos de uso y pueden identificarse con una letra.

En la Figura 22 se representa la interfaz de usuario del módulo editor la cual de forma general está conformada por 7 áreas que se describen a continuación.

Barra de Títulos: Se muestra el nombre de la herramienta, el proyecto que esté cargado en escena y los botones para maximizar, minimizar o cerrar la herramienta.

Área “A”: Se resalta en un recuadro la barra de tarea donde se encuentra elementos de acceso rápido que le dan respuesta algunas de las funcionalidades del sistema.

Área “B”: Se resalta en un recuadro el componente Explorer, permite listar los nombres de los objetos que se encuentren en la escena (área D).

Área “C”: Proyectos, componente que permite listar los nombres de los proyectos que se encuentran en la carpeta del sistema.

Área “D”: Render, área de edición y visualización de la escena.

Área “E”: Propiedades, componente que muestra los atributos de los recursos (modelos, cámara, recorridos, luces) de la escena y permite editarlos.

Área “F”: Mallas, componente que permite listar los modelos gráficos que se encuentren en el sistema.

Área “G”: Cargar Proyecto, ventana que permite introducir los datos necesarios para crear o cargar un proyecto.

A continuación se mencionan los componentes de la propuesta de interfaz de usuario que dan respuesta a los casos de uso arquitectónicamente significativos del módulo Editor:

CU-1: Administrar proyecto.

La aplicación da respuesta mediante los componentes de la interfaz situados en las áreas “A”, “B” y “G” de la Figura 22.

CU-2: Gestionar modelos del sistema.

La aplicación da respuesta mediante uno de los componentes que se encuentra en la barra de herramienta. Ver Figura 22.

CU-3: Administrar objetos del entorno.

La aplicación da respuesta mediante uno de los componentes que se encuentra en el área “F”. Ver Figura 22.

CU-7: Administrar cámara.

La aplicación da respuesta mediante uno de los componentes que se encuentra en la barra de herramienta y el componente Propiedades del área “E”. Ver Figura 22.

CU-9: Exportar paseo.

La aplicación da respuesta mediante los componentes que se encuentra en el área “B”, y “G”. Ver Figura 22.

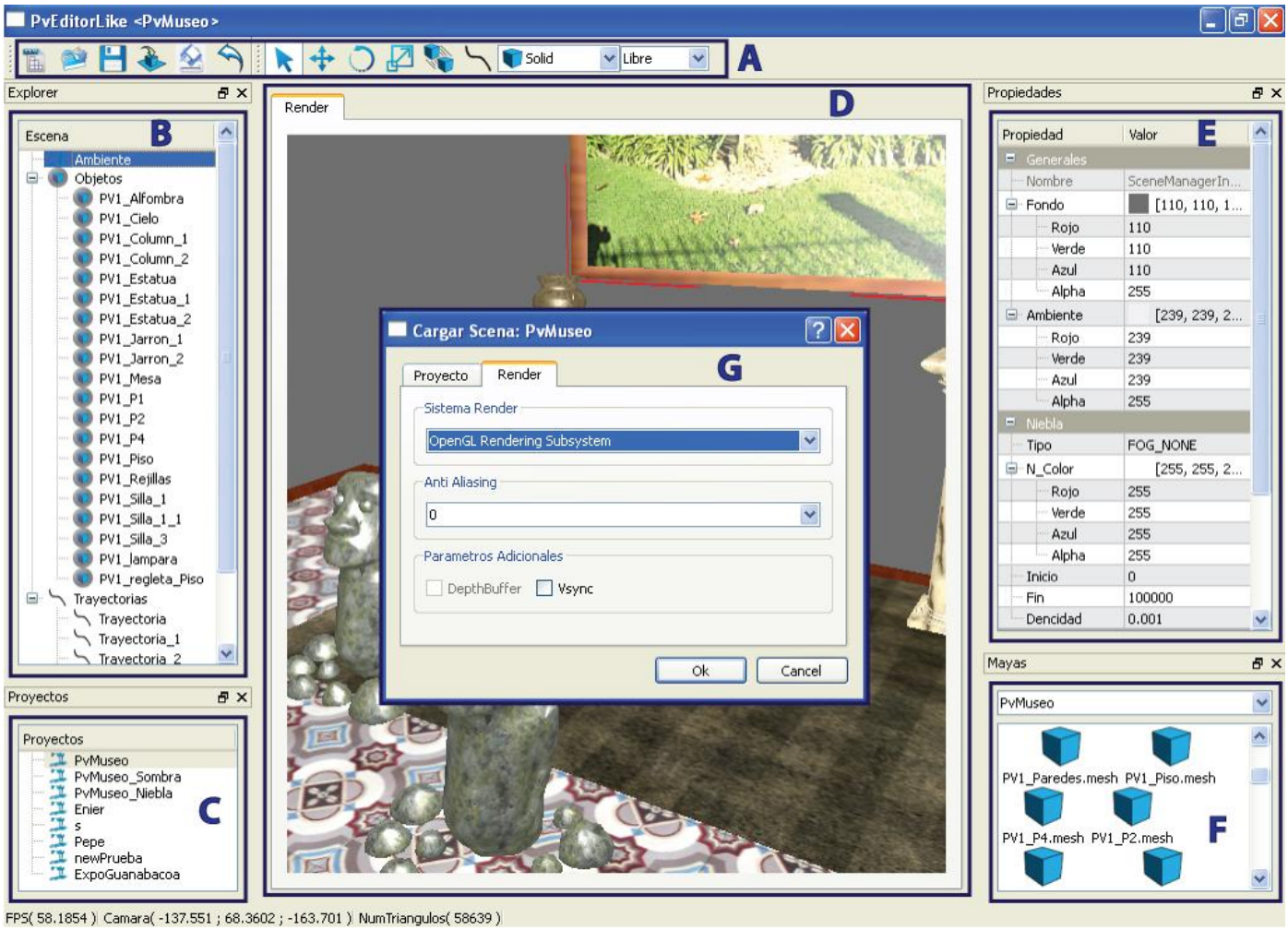


Fig. 22. Interfaz Gráfica del Editor.

Consideraciones del capítulo.

En el presente capítulo se resumió la fase de Análisis y Diseño, cuyos resultados le fueron entregados al autor para la implementación del módulo. Se definieron los requisitos, tanto funcionales como no funcionales, que deberá cumplir el módulo, así como los Casos de Uso correspondientes al diseño del mismo, cuyos diagramas se presentaron. Por último, se presentaron los modelos conceptuales, diagramas y descripción de clases, realización de Casos de Uso y finalmente se diseñó una propuesta de interfaz agradable a la vista y sencilla para facilitar su uso y dar respuesta a las funcionalidades del módulo.

Capítulo 3. Implementación y Prueba.

Introducción al capítulo 3.

En la fase de Implementación se traduce el diseño de clases, elaborado en la fase de Análisis y Diseño, en componentes físicos, los cuales se convierten en los ficheros correspondientes a la implementación en C++. También se harán las pruebas correspondientes a los Casos de Uso descritos en el flujo de trabajo anterior.

3.1 Lenguajes y herramientas a utilizar.

- Para la implementación del módulo se utilizaron las herramientas y lenguajes que más ventajas nos ofrecían para el trabajo con motores gráficos e interfaces visuales, así como para la modelación de la herramienta. Tales fueron:
- Lenguaje de modelado: UML.
- Herramienta de diseño de software: Visual Paradigm para UML.
- Lenguaje de programación: C++.
- Entorno integrado de desarrollo (IDE): QT Creator 4.7.
- Motor Gráfico: Object Oriented Graphics Engine (Ogre). OgreSDK MinGW v1-7-1.

3.2 El modelo de implementación.

El modelo de implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes. Describiendo además cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados y cómo dependen los componentes unos de otros.(**Rodríguez Noa** and **Gómez Finalé** 2008)

3.2.1 Diagrama de Componentes.

El diagrama de componentes muestra las relaciones de dependencia entre las partes modulares de un sistema y encapsula la implementación. A continuación se muestra el diagrama de componentes del módulo desarrollado en el presente trabajo.

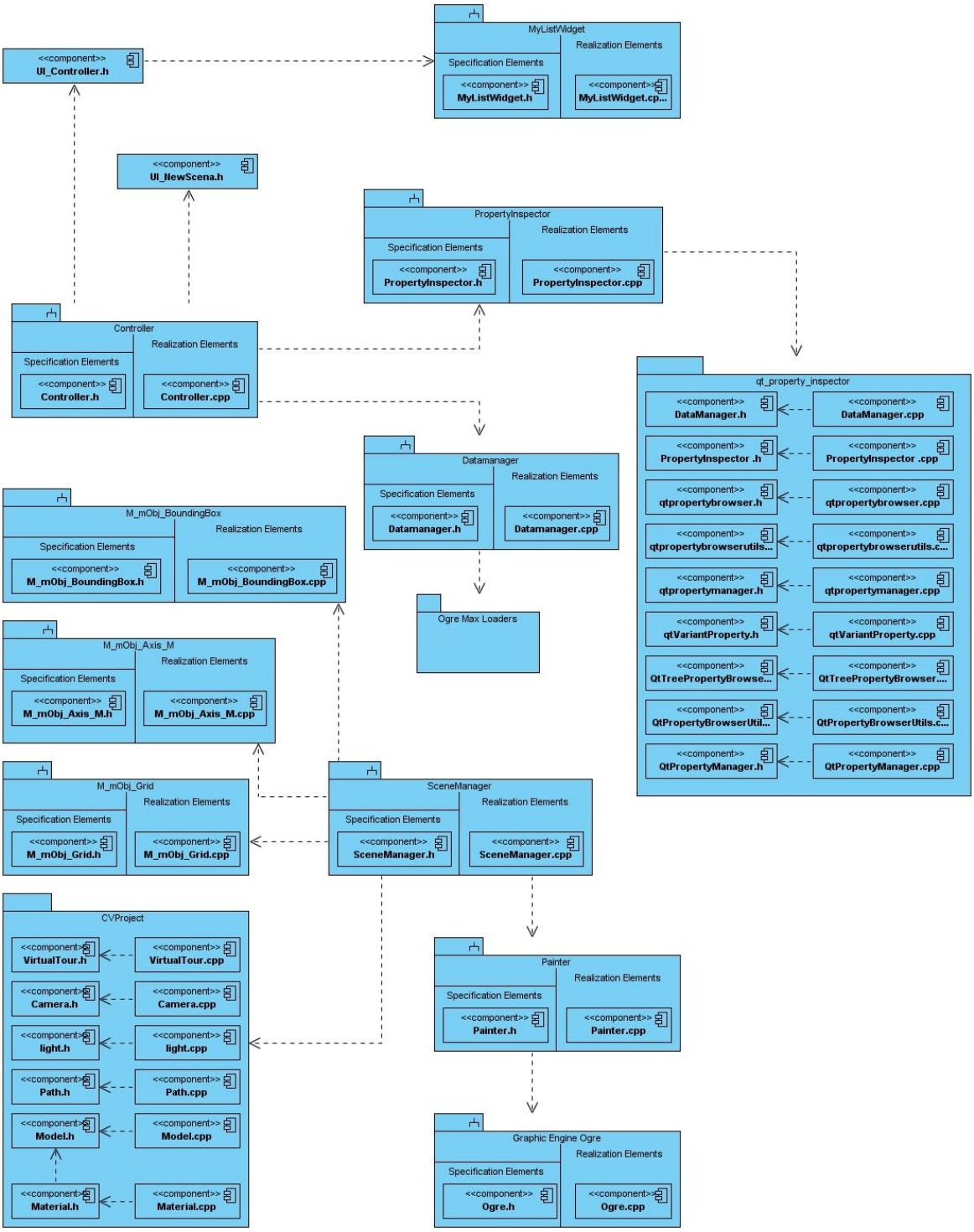


Fig. 23. Diagrama de Componentes

3.2.2 Diagrama de Despliegue.

El modelo de despliegue es un modelo de objetos que describe la distribución física de un sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. En nuestro caso, al ser desplegado en una misma estación de trabajo, no consideramos necesario mostrar el diagrama.

3.2.3 Implementación del CU Mostrar ayuda.

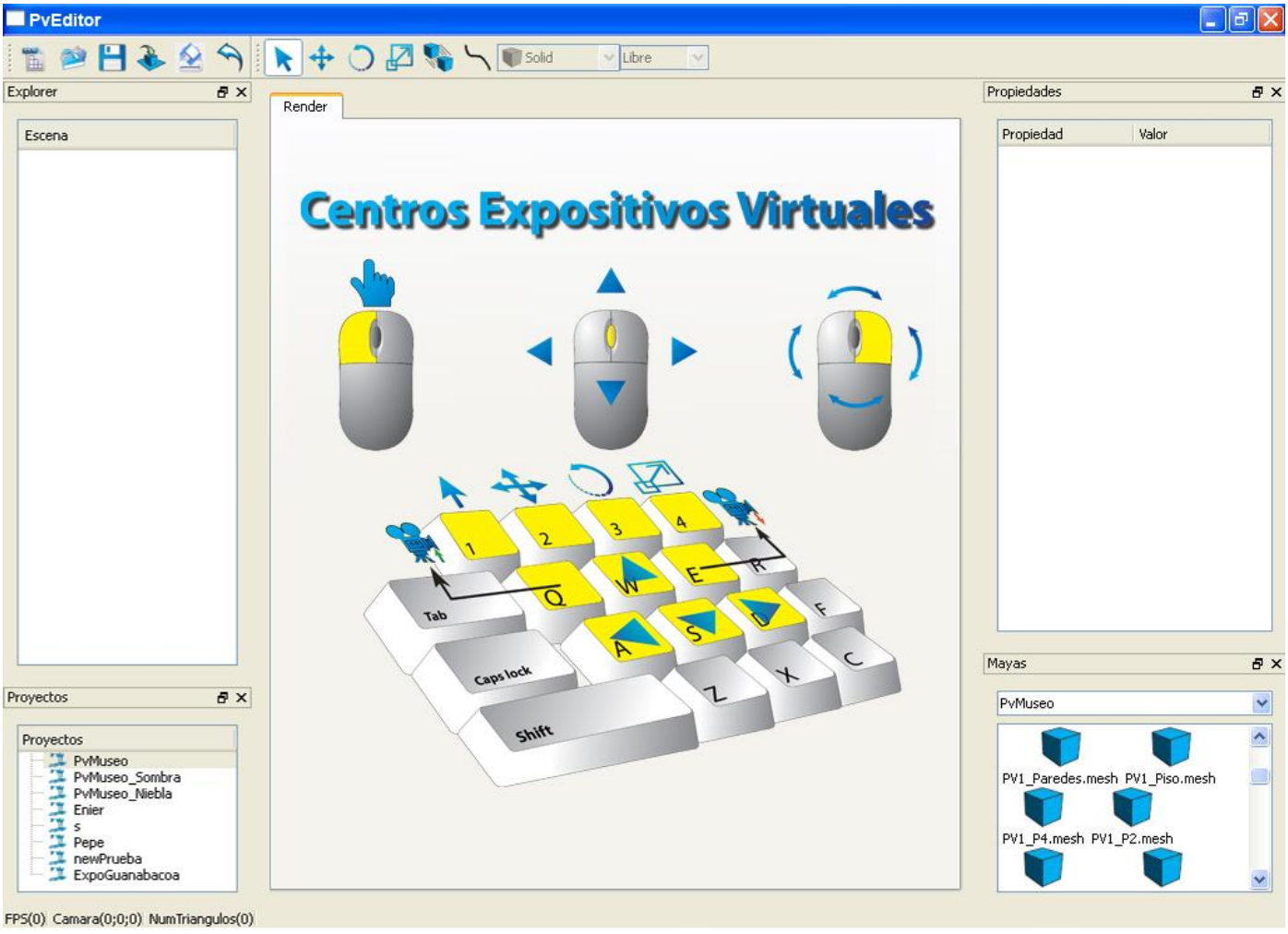


Fig. 24. Implementación del CU Mostrar ayuda.

3.2.4 Implementación del CU Gestionar proyecto.

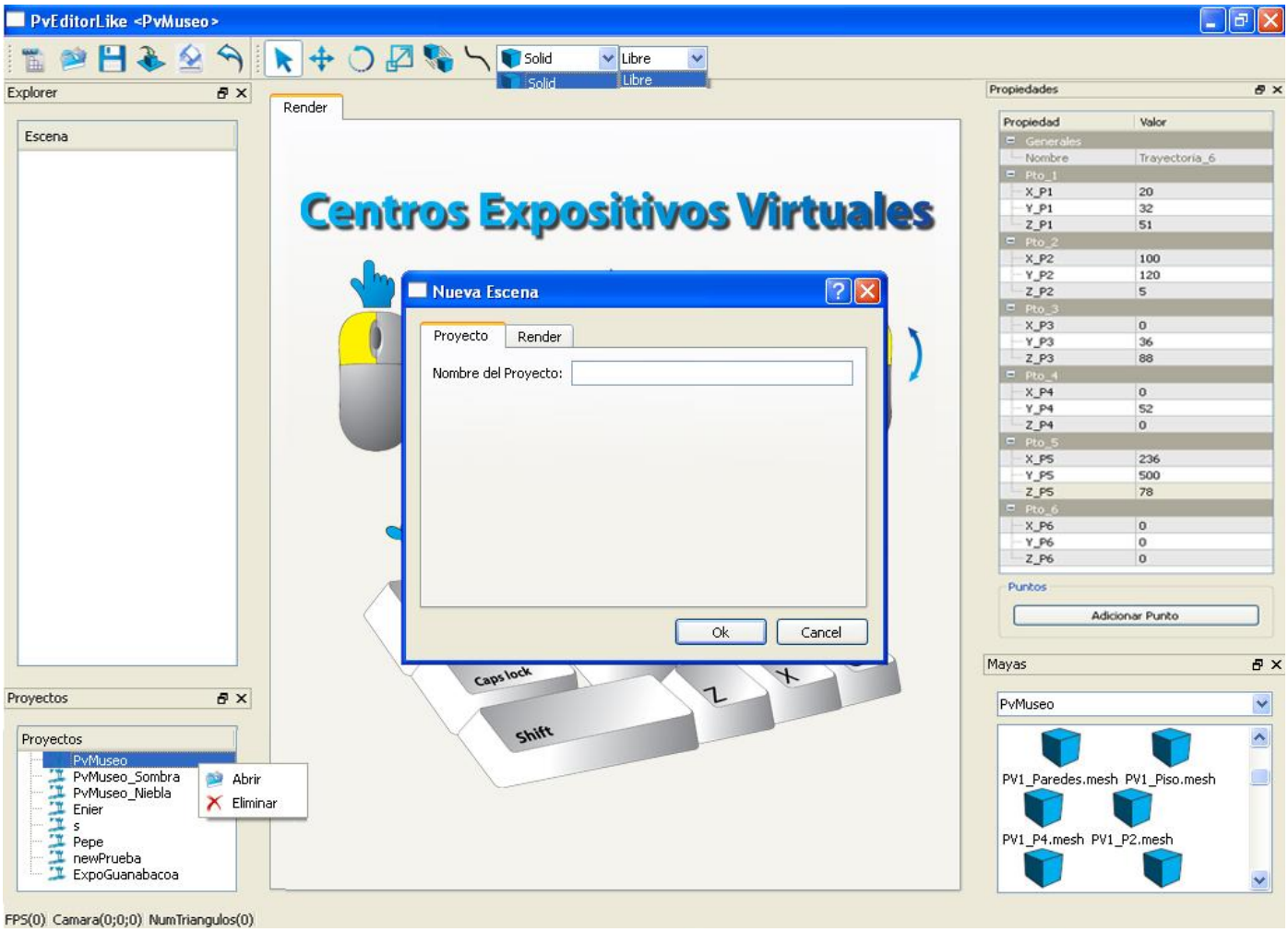


Fig. 25. Implementación del CU Gestionar proyecto.

3.2.5 Implementación de los CU relacionados con la edición de la escena.

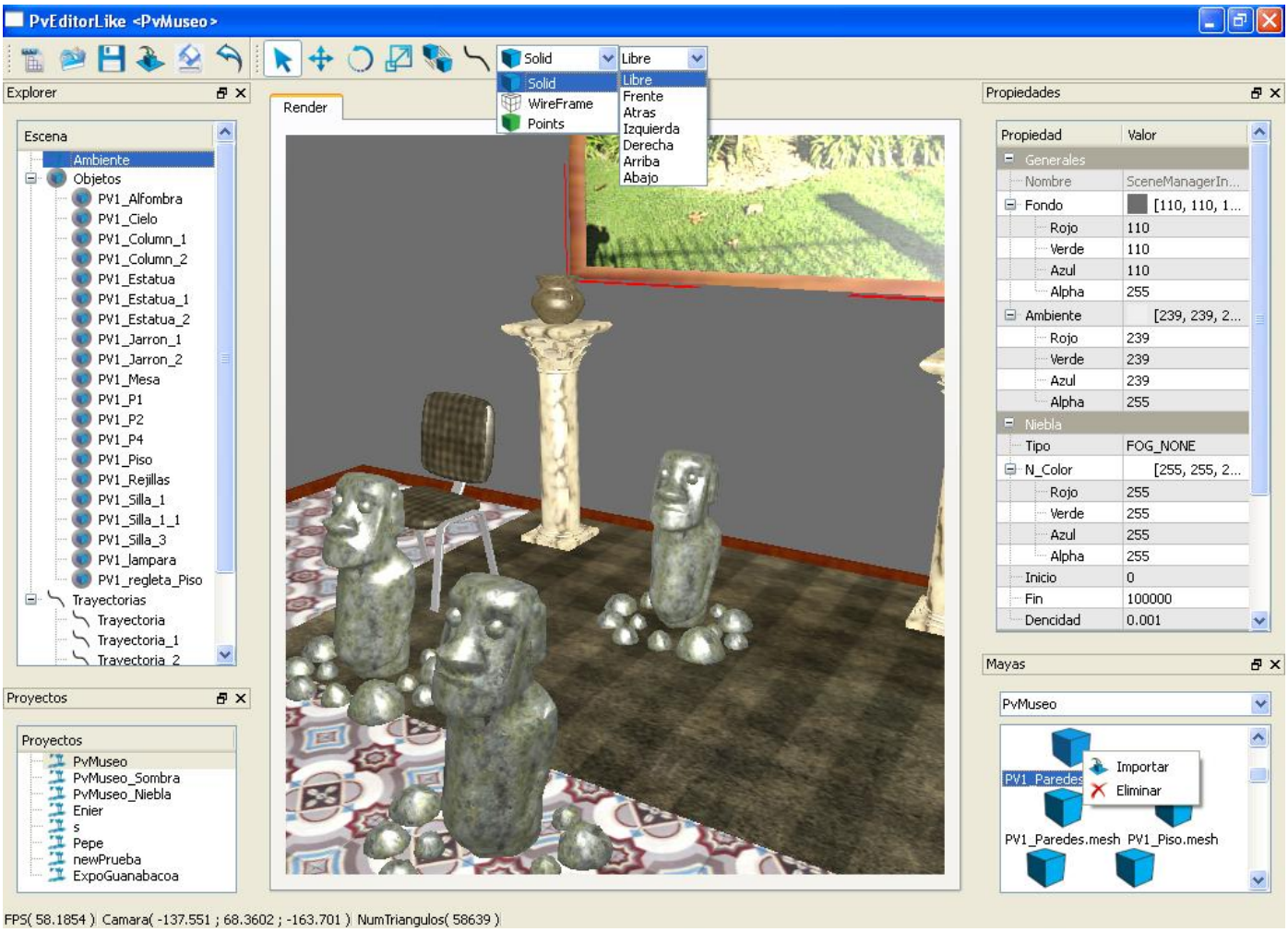


Fig. 26. Implementación de los CU de edición.

3.2.6 Implementación del CU Gestionar Cámara.

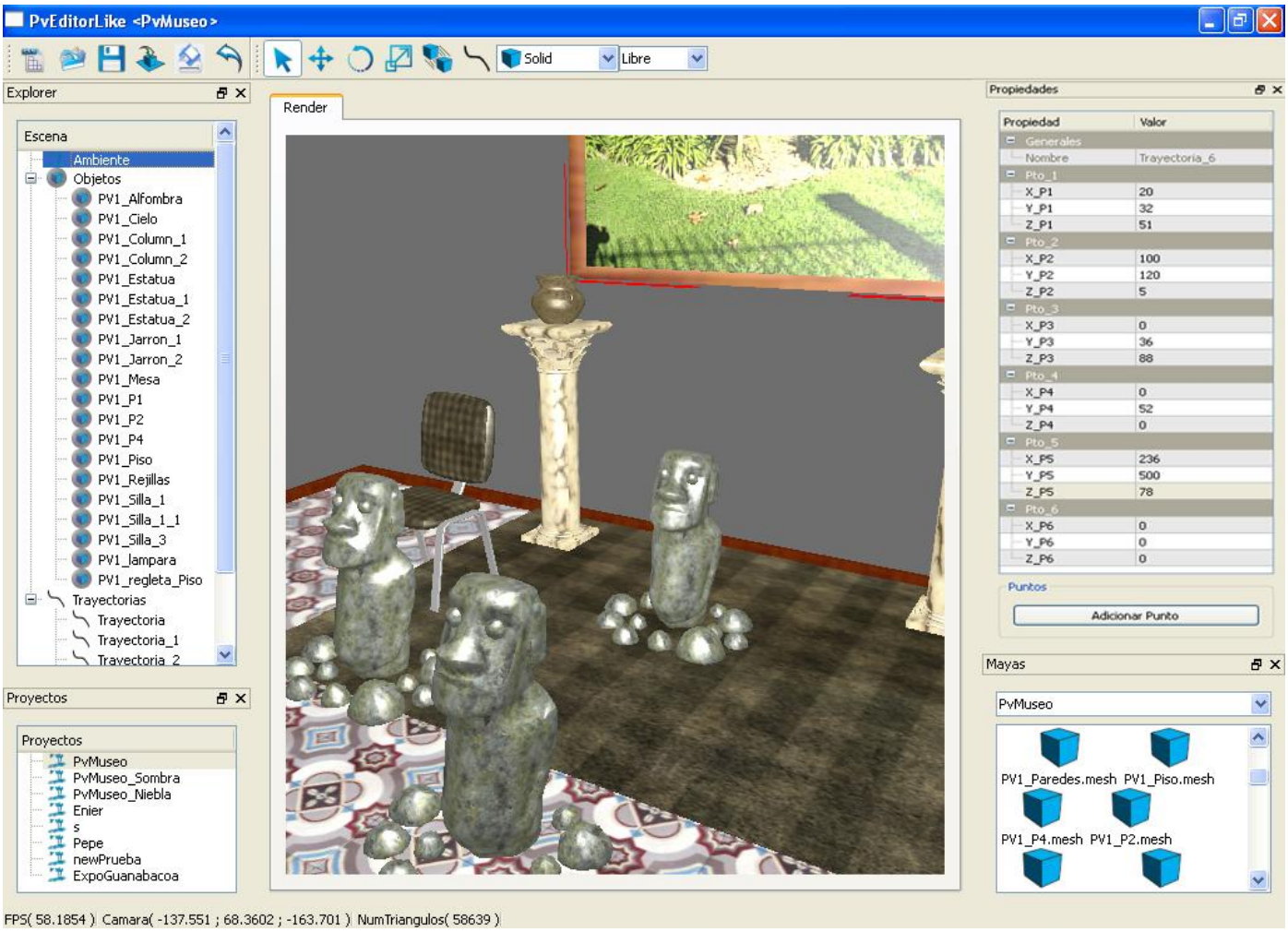


Fig. 27. Implementación del CU Gestionar Cámara.

3.3 Modelo de prueba.

Los modelos de pruebas responden a la implementación de los Casos de Uso más importantes de la aplicación. En este caso, se utilizó para estructurarlos el formato utilizado por los documentos oficiales del Centro de Informática Industrial, al que pertenece el proyecto Paseos Virtuales. El tipo de pruebas realizadas son de caja negra, lo que significa que las pruebas de caja negra se centran en lo que se espera de un módulo, es decir, intentan encontrar casos en

que el módulo no se atiene a su especificación, y el probador se limita a suministrarle datos como entrada y estudiar la salida.

Las pruebas de caja negra están especialmente indicadas en aquellos módulos que van a ser interfaz con el usuario (en sentido general: teclado, pantalla, ficheros, canales de comunicaciones, etc.). No obstante, pueden ser útiles en cualquier módulo del sistema.

3.3.1 Diseño de caso de prueba. CU Gestionar proyecto.

Descripción general:

El diseñador del paseo inicializa el caso de uso cuando selecciona la opción Crear un proyecto. Se crea una carpeta con el nombre del proyecto en la cual se salvará el fichero principal del paseo y una carpeta que contendrá los recursos del paseo virtual. El fichero y la carpeta de recursos tendrán el mismo nombre. Los cambios efectuados al proyecto serán guardados en la capeta especificada. El caso de uso permite además eliminar el paseo virtual diseñado

Condiciones de ejecución:

No puede existir un proyecto con el mismo nombre.

Secciones a probar en el caso de uso:

Tabla 4. Diseño de caso de prueba. CU Gestionar proyecto.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC 1: Crear Proyecto.	**EC 1.1:** Crear Proyecto exitosamente.	El sistema pide al usuario un nombre para el nuevo proyecto. Este escribe un nombre que no está siendo utilizado por más ningún proyecto, y el sistema crea la carpeta del proyecto, que contendrá los recursos del paseo virtual. Ambos (carpeta de proyecto y	Seleccionar opción nuevo proyecto. Escribir un nombre que no esté en uso. Se crea la carpeta y el fichero del proyecto.

		fichero) son creados con el nombre del proyecto.	
	EC 1.2: El usuario introduce nombre de proyecto existente	Se muestra un mensaje indicando que un proyecto con ese nombre existe ya. El usuario da clic sobre la opción cancelar y el sistema cancela la operación.	Se introduce un nombre ya existente. Se selecciona la opción cancelar. Se cancela la operación.
SC 2: Salvar Proyecto.	**EC 2.1:** Salvar Proyecto.	Cuando el usuario da clic sobre la opción de salvar, el sistema guarda la información del proyecto actual.	Seleccionar opción guardar. El sistema guarda la información del proyecto.
SC 3: Eliminar proyecto	**EC 3.1:** Eliminar proyecto exitosamente.	El usuario selecciona la opción Eliminar Proyecto. El sistema pregunta al actor si realmente está seguro de eliminar el proyecto del sistema. El actor da clic sobre el botón aceptar. El sistema elimina el proyecto, la carpeta del paseo con el fichero principal y todos sus recursos. Finaliza el caso de uso.	Seleccionar opción Eliminar. El sistema pide confirmación. Se elimina la carpeta del proyecto.
	EC 3.2: Cancelar eliminación de proyecto.	El usuario selecciona la opción Eliminar Proyecto. El sistema pregunta al actor si realmente está seguro de eliminar el proyecto del	Seleccionar opción Eliminar. El sistema pide confirmación.

		sistema. El actor da clic sobre el botón cancelar. El sistema cancela la operación.	Se cancela la operación.

Descripción de la variable:

Tabla 5. Descripción de variables. CU Gestionar proyecto.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Nombre del proyecto	Campo de texto	No	Cadena de caracteres.

Matriz de Datos

SC #1: Autenticar usuario

Tabla 6. Matriz de Datos. CU Gestionar proyecto.

ID del escenario	Escenario	Nombre del proyecto	Respuesta del sistema	Resultado de la prueba
EC 1.1	El usuario introduce nombre de proyecto válido.	Museo	Se crea el fichero y la carpeta del proyecto.	Se crean la carpeta y el fichero de proyecto exitosamente.
EC 1.2	El usuario introduce nombre de proyecto existente.	Museo	Se muestra el mensaje de error “El nombre de proyecto está en uso”.	Se cancela la operación de creación de un nuevo proyecto exitosamente.

3.3.2 Diseño de caso de prueba. CU Gestionar modelos del sistema.

Descripción general:

El diseñador del paseo inicializa el caso de uso cuando selecciona la opción adicionar modelo al sistema. Antes de eliminar un modelo del sistema se pregunta al usuario si realmente está seguro de eliminarlo definitivamente del sistema.

Condiciones de ejecución:

No puede existir ningún modelo en el sistema con el mismo nombre del modelo a adicionar.

Secciones a probar en el caso de uso:

Tabla 7. Diseño de caso de prueba. CU Gestionar modelos del sistema.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC 1: Adicionar modelo al proyecto.	**EC 1.1:** Adicionar modelo nuevo al proyecto.	El usuario da clic sobre la opción de adicionar modelo al proyecto. El sistema abre un explorador para que el usuario busque el modelo. Da clic sobre el botón aceptar y el sistema agrega el modelo como un recurso gráfico del proyecto.	Seleccionar opción adicionar modelo. Buscar el modelo. Adicionar modelo al proyecto.
	EC 1.2: Adicionar modelo existente al proyecto.	Al seleccionar un modelo que ya existe en el proyecto, el sistema dará la opción de adicionarlo al sistema con un nuevo nombre, que será el nombre del modelo seguido de un "_" y un número para diferenciarlo del ya existente.	Seleccionar modelo ya existente en el proyecto. Adicionar modelo con nombre nuevo.
SC 2: Eliminar	**EC 2.1:** Eliminar	El usuario selecciona la opción	Seleccionar opción

modelo del proyecto.	modelo del proyecto.	eliminar modelo, el sistema pide confirmación de la eliminación, y elimina el modelo del proyecto.	eliminar modelo. El sistema elimina el modelo del proyecto.

3.3.3 Diseño de caso de prueba. CU Gestionar objetos del entorno.

Descripción general:

El diseñador del paseo inicializa el caso de uso cuando selecciona una de las opciones para gestionar los objetos decorativos o expositivos que se mostrarán dentro del entorno. El caso de uso permite cargar modelos en escena, moverlos, alinearlo respecto a otro objeto, activar sombra y modificar la forma de verlos.

Condiciones de ejecución:

Los objetos deben estar añadidos al sistema.

Secciones a probar en el caso de uso:

Tabla 8. Diseño de caso de prueba. CU Gestionar objetos del entorno.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC 1: Cargar modelo en la escena.	**EC 1.1:** Cargar modelo nuevo en la escena.	Al añadir el objeto a la escena, el sistema la dibuja en el área de visualización.	Se añade el objeto a la escena.
	EC 1.2: Cargar modelo existente en la escena.	Si el objeto ya se encuentra en la escena, se renombra con un "_" al final de su nombre, seguido por una numeración.	Se añade el objeto a la escena.
SC 2: Mover objeto.	**EC 2.1:** Mover objeto.	Se selecciona la opción Mover, se dibujan los ejes x, y, z, y el usuario	Se posiciona el objeto en la escena.

		posiciona el objeto en la escena.	
SC 3: Alinear Objeto	**EC 3.1:** Alinear Objeto	El sistema muestra una rejilla que indica la posible alineación con os otros objetos. El usuario alinea el objeto.	Se alinean los objetos de acuerdo a un eje.
SC 4: Activar Sombra	**EC 4.1:** Activar Sombra	Una vez seleccionado un objeto, la opción activar/desactivar sombra activará o desactivará la sombra del objeto en el área de visualización.	Se activan o desactivan las sombras de un objeto.
SC 5: Cambiar vista del objeto	**EC 5.1:** Cambiar vista del objeto	El usuario selecciona una opción de visualización para un objeto: Frente, Arriba, Derecha, Izquierda y Libre en Perspectiva.	Cambia la vista del objeto
SC 6: Cambiar modo de visualización	**EC 6.1:** Cambiar modo de visualización	El usuario puede seleccionar un modo de visualización de la escena: Solido, Puntos, o Malla	Se cambia el modo de visualización de la escena.
SC 7: Eliminar objeto de la escena	**EC 7.1:** Eliminar objeto de la escena	Teniendo un objeto seleccionado, el usuario selecciona la opción Eliminar Objeto	Se elimina el objeto de la escena.

3.3.4 Diseño de caso de prueba. CU Modificar objeto decorativo.

Descripción general:

El diseñador del paseo inicializa el caso de uso cuando selecciona una de las opciones para modificar el tamaño y la cantidad de objetos decorativos de un mismo tipo que puede existir en el paseo.

Condiciones de ejecución:

El objeto debe estar añadido al sistema.

Secciones a probar en el caso de uso:

Tabla 9. Diseño de caso de prueba. CU Modificar objeto decorativo.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC 1: Escalar objeto decorativo	**EC 1.1:** Escalar objeto decorativo	El actor modifica las dimensiones del objeto seleccionando la opción escalar y arrastrando el mouse sobre el (los) eje(s) de coordenada que se desea escalar el objeto, o modificando los atributos correspondientes a las dimensiones del objeto ubicado en el inspector de propiedades.	Se escala el objeto
SC 2: Duplicar objeto decorativo	**EC 2.1:** Duplicar objeto decorativo	El sistema duplica el objeto, nombrando al nuevo objeto igual que el objeto original, con un "_" al final y una numeración.	Se duplica el objeto

3.3.5 Diseño de caso de prueba. CU Gestionar información de los objetos.

Descripción general:

El diseñador del paseo inicializa el caso de uso cuando Crea y modifica la información referente a un objeto y lo clasifica en decorativo o expositivo.

Condiciones de ejecución:

El objeto debe estar añadido al sistema.

Secciones a probar en el caso de uso:

Tabla 10. Diseño de caso de prueba. CU Gestionar información de los objetos.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC 1: Crear texto	**EC 1.1:** Crear texto	El usuario selecciona un objeto y escribe un texto en la propiedad del objeto Texto.	Se añade información al objeto
SC 2: Eliminar texto	**EC 2.1:** Eliminar texto	El usuario selecciona un objeto y borra el texto en la propiedad del objeto Texto.	Se elimina la información del objeto
SC 3: Modificar texto	**EC 3.1:** Modificar texto	El usuario selecciona un objeto y modifica el texto en la propiedad del objeto Texto.	Se modifica la información del objeto
SC 4: Definir tipo de objeto	**EC 4.1:** Definir tipo de objeto	El usuario selecciona un objeto y selecciona una de las opciones de la propiedad Tipo del objeto. Las opciones de la propiedad son Decorativo, Expositivo o Escenario.	Se asigna la opción de la propiedad seleccionada al objeto.

3.3.6 Diseño de caso de prueba. CU Gestionar cámara.

Descripción general:

El diseñador del paseo inicializa el caso de uso cuando adiciona un recorrido al paseo, adiciona una cámara y modifica su altura, velocidad o la elimina del paseo.

Condiciones de ejecución:

Debe estar definido el recorrido a realizar por el paseo.

Secciones a probar en el caso de uso:

Tabla 11. Diseño de caso de prueba. CU Gestionar cámara.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC 1: Definir trayectoria	**EC 1.1:** Definir trayectoria	El usuario selecciona la opción adicionar un recorrido y define mediante el mouse puntos que van definiendo el recorrido (spline).	Se crea el spline de recorrido.
SC 2: Adicionar cámara	**EC 2.1:** Adicionar cámara en el recorrido.	El usuario da clic sobre la opción adicionar cámara y la posiciona en el recorrido previamente creado.	Se crea la cámara.
	EC 2.2: Adicionar cámara fuera del recorrido.	El usuario da clic sobre la opción adicionar cámara y la posiciona fuera del recorrido previamente creado. El sistema notifica al usuario que el área no está permitida cancelando la acción.	Se cancela la acción.
SC 3: Eliminar cámara	**EC 3.1:** Eliminar cámara	El usuario selecciona la opción eliminar cámara.	Se elimina la cámara de la escena.
SC 4: Modificar altura de cámara	**EC 4.1:** Modificar altura de cámara	El usuario selecciona la cámara y modifica la altura desplazándola con el mouse o modificando el atributo altura del objeto.	Se modifica la altura de la cámara.
SC 5: Eliminar trayectoria.	**SC 5.1:** Eliminar trayectoria.	El usuario selecciona la opción eliminar trayectoria.	Se elimina la trayectoria creada.

Consideraciones del capítulo.

En este capítulo se describió cómo los elementos entregados como resultado de la fase de Análisis y Diseño son implementados. Se describió además cómo se organizan los componentes de acuerdo con los mecanismos disponibles en el entorno de implementación y en el lenguaje de programación utilizado y cómo dependen los componentes unos de otros.

Se muestran también los casos de pruebas diseñados para realizar el flujo de Pruebas, asegurando que las deficiencias puedan ser solucionadas rápidamente por el desarrollador.

Conclusiones.

Terminado el trabajo se puede concluir que se logró desarrollar un módulo de edición de escena 3D para una herramienta de creación de Centros Expositivos Virtuales, se elaboró el marco teórico relacionado con las técnicas de programación, se caracterizaron e identificaron las herramientas de manejo de escenas 3D, las posibles metodologías y lenguajes a utilizar y los Paseos Virtuales, se resumieron los resultados del Análisis y Diseño recibidos para el desarrollo del módulo, se diseñó y graficó el Diagrama de Componentes, se implementaron los Casos de Usos del módulo, y se diseñaron, describieron y aplicaron las pruebas de Caja Negra.

Todos estos resultados obtenidos demuestran que el objetivo de la investigación se alcanzó satisfactoriamente ya que de manera general se obtuvo un módulo de edición de escena 3D para una herramienta de creación de Centros Expositivos Virtuales; que permite crear un Paseo Virtual, incorporarle a este modelos diseñados en herramientas de diseño 3D, con herramientas de transformación para dichos modelos, de manera que el usuario pueda fácilmente diseñar un Paseo Virtual y personalice el escenario a su gusto, dejándolo listo para su posterior visualización.

Recomendaciones.

Recomendamos como parte del soporte del módulo:

- Implementar un módulo de detección de colisiones.
- Implementar la adición de sonido y video como información a los objetos.
- Estudiar la posibilidad de realizar paseos virtuales de entornos abiertos.

Referencias Bibliográficas.

Autodesk.com. (2011, Ene 2011). "Autodesk 3ds Max Features." from http://usa.autodesk.com/adsk/servlet/pc/index?siteID=123112&id=13567426.

Autodesk.com. (2011, Ene 2011). "Autodesk Maya Features." from http://usa.autodesk.com/adsk/servlet/pc/index?siteID=123112&id=13583239.

Blender.org. (2011, Jan 2011). "Blender Features." from http://www.blender.org/features-gallery/features/.

Crytek. (2008). "SandBox 2 Manual." Retrieved Dic 19, 2011, from http://doc.crymod.com/SandboxManual/frames.html?frmname=topic&frmfile=index.html.

Eclipse.org. (2011). "Eclipse IDE for C/C++ Developers." from http://www.eclipse.org/downloads/moreinfo/c.php.

Escuela Universitaria de Ingeniería Técnica en Informática de Oviedo, E. U. I. T. I. O. (2009). Entornos de Desarrollo Integrado.

Hiveworkshop.com. (2010). "The Hive Workshop tutorials." Retrieved Dic 20, 2010, from http://www.hiveworkshop.com/forums/tutorials/.

Microsoft. "Introducción a Visual Studio." from http://msdn.microsoft.com/es-es/library/fx6bk1f4(v=VS.80).aspx.

Microsoft. (2010). "Halo Official Site." Retrieved Diciembre 12, 2010, from www.halo.xbox.com.

Nokia. (2011). "Qt framework, Qt Creator IDE and tools." from http://qt.nokia.com/products/developer-tools/.

Ogre3D.org. (2009, Dic 22, 2010). "The Core Objects." from http://www.ogre3d.org/docs/manual/manual_4.html#SEC4.

Ogre3D.org. (2009). "Ogre3D Homepage." from http://www.ogre3d.org/.

Quintana López, A. and **Y. Alonso Monteagudo** (2010). Motor de Render Genérico, Universidad de las Ciencias Informáticas.

RedIRIS.com. (2009). "Modelado UML." from https://forja.rediris.es/docman/view.php/282/444/uml20.pdf.

Rodríguez Noa, C. A. and **Y. Gómez Finalé** (2008). Arquitectura de red para la confección de videojuegos multijugadores.

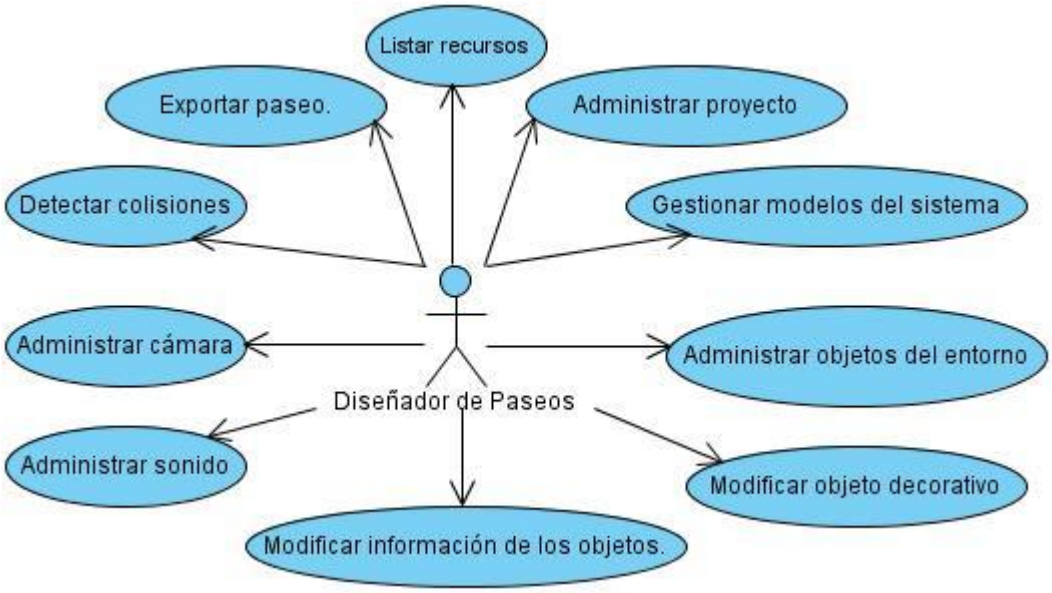
SourceForge.net. (2010). "G3D Project Home ", from http://g3d.sourceforge.net/.

Ubisoft. (2010). "FarCry2 Homepage." Retrieved Diciembre 12, 2010, from www.farcry2.com.

UnrealTechnology. (2008, 2011). "Unreal Editor Features." Retrieved Dic 20, 2010, from http://www.unrealengine.com/features/editor/.

Anexos.

Anexo 1. Diagrama de casos de uso.



Anexo 1. Diagrama de casos de uso del módulo Editor.

Anexo 2. Descripción textual de los casos de uso.

Casos de uso expandidos para el módulo Editor.

Caso de uso	
CU-4	Modificar objeto decorativo.
Propósito	El objetivo del caso de uso es que el diseñador del paseo pueda modificar el tamaño de un objeto decorativo y la cantidad que puede existir en el paseo.
Prioridad	Secundario.
Actores: Diseñador del Paseo.	
Resumen: El diseñador del paseo inicializa el caso de uso cuando selecciona una de las opciones para modificar el tamaño y la cantidad de objetos decorativos de un mismo tipo que puede existir en el paseo.	
Referencias	RF 4, RF 4.1, RF 4.2.
Precondiciones	El objeto debe estar añadido al sistema.

Poscondiciones	Se modifica el tamaño o la cantidad de objetos del entorno.
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador selecciona un objeto decorativo y le aplica una de las opciones siguientes: Escalar objeto decorativo, Duplicar objeto decorativo.	1.1 Si el diseñador selecciona la opción Escalar objetos decorativo ir al Escenario 1. 1.2 Si el diseñador selecciona la opción Duplicar objeto decorativo ir al Escenario 2.
Escenario 1 Escalar objeto decorativo	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador modifica las dimensiones del objeto seleccionando la opción escalar y arrastrando el mouse sobre el (los) eje(s) de coordenada que se desea escalar el objeto, o modificando los atributos correspondientes a las dimensiones del objeto ubicado en el inspector de propiedades.	1.1 El sistema modifica las dimensiones del objeto y lo muestra con las nuevas dimensiones en el espacio de visualización.
Escenario 2 Duplicar objeto decorativo	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador elige la opción duplicar objeto.	1.1 El sistema adiciona un nuevo modelo en el área de visualización con las mismas características del modelo seleccionado. El nombre del nuevo modelo tendrá el siguiente formato: “nombre del modelo duplicado_número”.

Caso de uso	
CU-5	Modificar información de los objetos.

Propósito	El objetivo del caso de uso es que el diseñador del paseo pueda modificar la información asociada a un objeto que se encuentre en el entorno o cambiarle el atributo “Tipo de objeto”.
Prioridad	Secundario.
Actores: Diseñador del Paseo.	
Resumen: El diseñador del paseo inicializa el caso de uso cuando Crea y modifica la información referente a un objeto y lo clasifica en decorativo o expositivo.	
Referencias	RF 5, RF 5.1, RF 5.2, RF 5.3, RF 5.4, RF 5.5.
Precondiciones	El objeto debe estar añadido al sistema.
Poscondiciones	Se asocia una información a un objeto y se define el tipo de objeto que se encuentre en el diccionario.

Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicializa cuando el diseñador selecciona una de las opciones siguientes: Actualizar información de objeto, Definir tipo de objeto.	1.1 Si el diseñador selecciona la opción Actualizar información de objeto ir al Escenario 1. 1.2 Si el diseñador selecciona la opción Definir tipo de objeto ir al Escenario 2.
Escenario 1 Actualizar información de objeto	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador selecciona un objeto y modifica el texto en la propiedad del objeto Texto.	1.1 El sistema guarda la actualización realizada en la propiedad del objeto.
Escenario 4 Definir tipo de Objeto	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador selecciona un objeto y elige una de las opciones de la propiedad Tipo del objeto. Las opciones de la propiedad son Decorativo, Expositivo o Escenario.	1.1 El sistema asigna la opción de la propiedad seleccionada al objeto.

Caso de uso	
CU-6	Administrar sonido.
Propósito	El objetivo del caso de uso es que el diseñador del paseo pueda adicionar y modificar un sonido al entorno.
Prioridad	Opcional.
Actores: Diseñador del Paseo.	
Resumen: El diseñador del paseo inicializa el caso de uso cuando adiciona un sonido al entorno y realiza modificaciones al mismo.	
Referencias	RF 6, RF 6.1, RF 6.2, RF 6.3, RF 6.4, RF 6.5.
Precondiciones	El sonido debe estar añadido al sistema.
Poscondiciones	Se asocia un sonido al entorno y se modifica el atributo que permite repetir sonido. Se adicionar sonido o eliminar un sonido del sistema.
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador elige una de las opciones siguientes: Adicionar sonido, Eliminar sonido, Repetir sonido, Asociar sonido.	1.3 Si el diseñador selecciona la opción Adicionar sonido ir al Escenario 1. 1.4 Si el diseñador selecciona la opción Eliminar sonido ir al Escenario 2. 1.5 Si el diseñador selecciona la opción Repetir sonido ir al Escenario 3. 1.6 Si el diseñador selecciona la opción Asociar sonido ir al Escenario 4.
Escenario 1 Adicionar Sonido	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**

1. El diseñador elige la opción Adicionar Sonido. 2. El diseñador selecciona el sonido desde la ubicación donde se encuentre. 3. El diseñador da clic en la opción aceptar.	1.1 El sistema abre una venta como el explorador de Windows para que el usuario busque el sonido. 3.1 El sistema agrega el sonido como un recurso del sistema disponible para cargarlo en los proyectos.
Curso Alterno de Eventos	
Acción del actor	**Respuesta del sistema**
2.1 El diseñador selecciona un sonido que ya existe en el sistema. 3.4 El diseñador da clic en la opción cancelar.	2.2 El sistema muestra el mensaje “El sonido “nombre” ya existe”. 3.5 El sistema cancela la operación culmina así el caso de uso.
Escenario 2 Eliminar Sonido	
Curso Normal de Eventos	
1. El diseñador elige la opción Eliminar sonido. 2. El diseñador da clic en la opción aceptar.	1.1 El sistema muestra el siguiente mensaje: ¿Desea eliminar el sonido: “nombre” del sistema? 2.1 El sistema elimina el sonido del sistema y finaliza el caso de uso.
Escenario 3 Repetir Sonido	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador selecciona la opción repetir sonido.	1.1 El sistema indica que el sonido se reproducirá cíclicamente.
Escenario 4 Asociar Sonido	
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador selecciona una escena y le asocia un sonido.	1.1 El sistema asocia el sonido seleccionado al entorno.

Caso de uso

CU-8	Detectar colisiones.
Propósito	El objetivo del caso de uso es que el diseñador del paseo no pueda ubicar más de un objeto en la misma posición del espacio.
Prioridad	Opcional.
Actores: Diseñador del Paseo.	
Resumen: El diseñador del paseo inicializa el caso de uso cuando intenta ubicar un objeto en una posición ocupada por otro objeto.	
Referencias	RF 8.
Precondiciones	Debe haber más de un objeto en la escena.
Poscondiciones	El paseo se crea con detección de colisiones.

Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador selecciona un objeto y activa la propiedad detectar colisiones.	1.1 El sistema adiciona al objeto la características “colisiona”, y verifica la posición de él relativa al resto de los objetos del entorno con esta característica, impidiendo que ocupe el mismo espacio físico de otros objetos con esta propiedad.

Caso de uso	
CU-10	Listar recursos.
Propósito	El objetivo del caso de uso es listar los nombres de los proyectos y los nombres de los modelos que se encuentren en el sistema.
Prioridad	Secundario
Actores: Diseñador del Paseo.	
Resumen: El diseñador del paseo inicializa el caso de uso cuando ejecuta el módulo Editor. El sistema muestra el listado de los nombres de los proyectos y el listado de nombres de los modelos que se encuentren en el sistema.	
Referencias	RF 10.

Poscondiciones	Se muestra el listado de los nombres de los proyectos y el listado de nombres de los modelos que se encuentren en el sistema.
Curso Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El diseñador ejecuta el módulo Editor.	1.1 El sistema muestra el listado de los nombres de los proyectos y los nombres de los modelos que se encuentren en el sistema.