DISEÑO Y DESARROLLO DE UN VIDEOJUEGO EDUCATIVO CON TÉCNICAS DE INTELIGENCIA ARTIFICIAL PARA LA PLATAFORMA ANDROID APLICANDO LA METODOLOGÍA OOHDM. CASO DE ESTUDIO: LABERINTO EN 3D

1 Soledad González, 2 Ing. Margarita Zambrano, 3 Ing. Carlos Prócel.
1 Escuela Politécnica del Ejercito, Ecuador, sgonzalez_est@.espe.edu.ec
2 Escuela Politécnica del Ejercito, Ecuador, mezambrano@.espe.edu.ec
4 Escuela Politécnica del Ejercito, Ecuador , ctprocel@espe.edu.ec

RESUMEN

Los videojuegos lúdicos o educativos es uno de los tipos de videojuegos que más se aplican como parte del proceso de enseñanza-aprendizaje en los niños, para el desarrollo del pensamiento y para el desarrollo psicomotriz de los mismos. Las aplicaciones de entretenimiento en 3D son las más apetecidas por niños y jóvenes cuando de jugar se trata y en los momentos de ocio. No cabe duda que la industria del videojuego es una de las que más ingresos genera a nivel mundial y los consumidores de este tipo de aplicaciones son millones de usuarios en todo el mundo.

Este proyecto de tesis, ha tenido como objetivo principal colaborar con los grupos de desarrollo de videojuegos educativos a través del análisis, diseño y desarrollo de un Laberinto como un juego didáctico en 3D, para ayudar al desarrollo del pensamiento lógico, matemático y espacial de niños entre 6 y 11 años, utilizando el Game Engine Unity con el lenguaje de programación C#.NET y aplicando la metodología de software OOHDM (Metodología de Diseño de Hipermedia Orientado a Objetos - Object Oriented Hypermedia Design Methodology), desarrollada por Daniel Schwabe y Gustavo Rossi, la cual consta de de cuatro etapas: Diseño Conceptual, Diseño Navegacional, Diseño Abstracto de Interfase e Implementación.

El proyecto cumple con una arquitectura cliente – servidor y consta de los siguientes elementos:

- *Un formulario con el menú principal, con las opciones de a) configuración del videojuego; b) resolver el laberinto de forma manual; c) resolver el laberinto de forma automática.*
- *El formulario de configuración, con las opciones de: a) configurar el tamaño de la pantalla para una PC; b) configurar el tamaño de la pantalla para una Tableta.*
- *El formulario resolver el laberinto de forma manual, con las opciones de: a) registro del nombre; b) seleccionar opción de prueba y entrenamiento; c) seleccionar el nivel del juego; d) guardar puntaje del usuario.*
- *El formulario resolver el laberinto de forma automática, con las opciones de: a) seleccionar opción de prueba y entrenamiento; b) seleccionar el nivel del juego; c) guardar puntaje de la PC.*

Palabras Clave:

- *Videojuegos lúdicos*
- *Laberintos con IA*
- *Aplicaciones 3D de entretenimiento*
- *OOHDM*

ABSTRACT

Words Key:

- *Educative Video games*

- *Labyrinths with AI*
- *Applications 3D of entertainment*
- *OOHDM*

1. INTRODUCCIÓN

En la actualidad se vive en un mundo en lo que todo o casi todo está impregnado de tecnología. Los niños viven con total normalidad esta circunstancia. De esta forma conviven y se desenvuelven en ella adaptándola sin dificultad para su uso cotidiano. Sin embargo, aún nos invade la sensación de quererles proteger de estas tecnologías olvidándonos de que quizás nuestra tarea no sea tanto la de aislarles de mucha tecnología, sino más bien la de adecuarles en el uso adecuado de las tecnologías como: Internet, Aulas virtuales, Multimedia, Realidad Virtual, Video Conferencia, Juegos Didácticos en 2D y 3D, etc.

Una gran desventaja en todo ámbito es el crecimiento acelerado de la tecnología y todos debemos ser conscientes en preparar a nuestros alumnos para el presente y el mañana, por tanto, si reflexionamos sobre las nuevas demandas sociales es necesario formar más acorde con los tiempos actuales. La tecnología 3D y la Realidad Virtual están revolucionando el proceso de enseñanza-aprendizaje de los niños en los países desarrollados como juegos en 3D, laboratorios virtuales, simuladores 2D y 3D, kits de robótica, etc., que ayudan al desarrollo psicomotriz y de la inteligencia de los niños.

Lastimosamente aun esta tecnología es muy costosa y no está al alcance de todos, especialmente de los países en vías de desarrollo donde los centros educativos no cuentan todavía con el soporte de estos tipos de tecnología en especial para el desarrollo psicomotriz de los niños donde las aplicaciones de juegos lúdicos en 2D y en 3D ayudan al desarrollo del pensamiento. Muchas escuelas de nuestro país, principalmente los de bajos recursos económicos, no cuentan con medios tecnológicos que se pueden aplicar en la educación como: Internet, Aplicaciones de entornos 3D, Aplicaciones de Multimedia, Aulas Virtuales, Video Conferencia, etc. En este sentido se va a desarrollar una aplicación de 3D y realidad virtual como una herramienta de apoyo para los profesores y estudiantes en el proceso de enseñanza-aprendizaje y estar acorde con la tecnología actual en la materia de desarrollo del pensamiento para niños de Educación Básica.

1.1 Objetivos

1.1.1 Objetivo General

Desarrollar un videojuego educativo con técnicas de inteligencia artificial para la plataforma Android aplicando la Metodología OOHDM. Caso de Estudio: Laberinto en 3D.

1.1.2 Objetivos Específicos

- Fundamentar el marco teórico acerca de aplicativos de software 3D, juegos didácticos y motores de juegos.
- Documentar las distintas fases de la metodología OOHDM.
- Realizar el análisis y el diseño de la aplicación 3D utilizando la metodología OOHDM con UML.
- Construir el modelo arquitectónico 3D de la aplicación con una herramienta de diseño 3D.
- Desarrollar la aplicación 3D utilizando la metodología OOHDM con UML.
- Utilizar las técnicas de inteligencia artificial de planificación y sistemas de reacción basados en reglas para resolver un laberinto en 3D.

2. METODOLOGÍA

Para el desarrollo de esta Aplicación Multimedia se adoptará la Metodología de Diseño de Hipermedia Orientado a Objetos (OOHDM), desarrollado por Daniel Schwabe y Gustavo Rossi. Esta metodología básicamente

consta de cuatro etapas: Diseño Conceptual, Diseño Navegacional, Diseño Abstracto de Interfase e Implementación. Cada etapa de la concepción define un esquema objeto específico en el que se introducen nuevos elementos o clases.

En la primera etapa se construye un esquema conceptual representado por los objetos de dominio o clases y las relaciones entre dichos objetos. Se puede usar un modelo de datos semántico estructural, como el modelo de entidades y relaciones. El modelo OOHDM propone como esquema conceptual basado en clases, relaciones y subsistemas.

En la segunda etapa, se define la estructura de navegación a través del hiperdocumento mediante la realización de modelos navegacionales que representen diferentes vistas del esquema conceptual de la fase anterior. El Diseño Navegacional se expresa, también con un enfoque orientado a objetos, a través de dos tipos de esquemas o modelos: el denominado esquema de clases navegacionales, con las posibles vistas del hiperdocumento a través de unos tipos predefinidos de clases, llamadas navegacionales, como son los "nodos", los "enlaces", y otras clases que representan estructuras o formas alternativas de acceso a los nodos, como los "Índices" y los "recorridos guiados"; y el esquema de contexto navegacional, que permite la estructuración del hiperespacio de navegación en subespacios para los que se indica la información que será mostrada al usuario y los enlaces que estarán disponibles cuando se acceda a un objeto u nodo en un contexto determinado.

La tercera etapa está dedicada a la especificación de la interfaz abstracta. Así, se define la forma en la cual deben aparecer los contextos navegacionales. También se incluye aquí el modo en que dichos objetos de interfaz activarán la navegación y el resto de funcionalidades de la aplicación, esto es, se describirán los objetos de interfaz y se los asociará con objetos de navegación. La separación entre el diseño navegacional y el diseño de interfaz abstracta permitirá construir diferentes interfaces para el mismo modelo navegacional.

En la cuarta etapa, es en si la implementación del hiperdocumento o sistema hipermedial diseñado, es decir, la concreción de los modelos navegacionales y de interfase en objetos particulares con sus correspondientes contenidos y sus posibilidades de navegación. Aunque, al utilizar un enfoque de orientación a objetos podría parecer conveniente que la implementación se hiciera en un entorno de construcción de hiperdocumentos también orientado a objetos, debido al carácter abstracto del diseño, sin embargo ésta puede hacerse fácilmente en otros entornos hipermediales que permitan trabajar con el lenguaje HTML.

Para el desarrollo de este proyecto en lo que respecta al Marco Teórico se va a utilizar una metodología de trabajo basada en la investigación bibliográfica de fuentes de Información, y consultas en Internet. Mediante esta etapa del proyecto se pretende obtener la información necesaria y válida que permita establecer el marco teórico referencial, el cual proporcione el soporte teórico – técnico necesario para la consecución del proyecto.

Se cumplirán las siguientes actividades como parte de una metodología de desarrollo estándar:

- Análisis
- Diseño
- Desarrollo
- Pruebas

El sistema a ser desarrollado se compone de un componente hipermedial para la Web y otro de tipo desktop que correrá sobre la plataforma Windows. La Metodología OOHDM se aplicará en conjunto con la Ingeniería de Software y UML (Lenguaje de Modelamiento Unificado).

3. DISEÑO E IMPLEMENTACION DEL JUEGO LÚDICO (LABERINTO3D)

3.1 Diseño

3.1.1 Identificación de Roles y Tareas

A) Roles

- Jugador-Gamer

  Es el usuario que tiene acceso al juego para poder interactuar con la aplicación en 3D, para lo cual debe registrarse como un gamer del juego y puede programar el tiempo que estará vigente dicho usuario en el juego y poder completar los retos del juego.

  En la Figura 3.1 se muestra gráficamente al actor del LABERINTO3D, que como se mencionó anteriormente es el jugador.

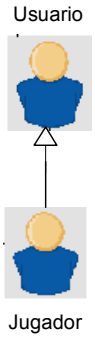

Figura 3.1. Actores - LABERINTO3D.

B) Tareas

- Jugador - Gamer

  1. Configurar el videojuego
  2. Resolver el laberinto de forma manual
  3. Resolver el laberinto de forma automática

3.1.2 Especificación de Escenarios

Rol Jugador

- ***Configurar el videojuego*:** El jugador podrá configurar las diferentes opciones que tiene el videojuego como son controlar los niveles de acceso al mismo a través de un nivel manual y de un nivel automático.

- ***Resolver el laberinto de forma manual*:** El jugador podrá resolver los retos del videojuego que consiste en resolver el laberinto en 3D manualmente, utilizando las flechas del teclado y luego almacenar en un archivo plano el nombre del usuario y el tiempo que se demora en resolver el laberinto, en base a los tres niveles de dificultad (principiante, intermedio y avanzado) que tiene el juego.

- ***Resolver el laberinto de forma automática*:** El jugador podrá visualizar la resolución del laberinto en 3D, para lo cual puede seleccionar cualquiera de los niveles de dificultad que tiene el juego y la computadora resolverá los retos del mismo, utilizando técnicas de IA como lo es la regla de la mano izquierda. En esta opción se almacenará en un archivo plano el nombre de la PC y el tiempo que se demora en resolver el laberinto, en base a los tres niveles de dificultad (principiante, intermedio y avanzado) que tiene el juego.

4. IMPLEMENTACIÓN Y PRUEBAS DEL JUEGO LÚDICO (JDRA3D)

4.1 Unity

Unity es un motor gráfico 3D para PC y Mac que viene empaquetado como una herramientapara crear juegos, aplicaciones interactivas, visualizaciones y animaciones en 3D y tiemporeal. Unity puede publicar contenido para múltiples plataformas como PC, Mac, NintendoWii y iPhone. El motor también puede pu-

blicar juegos basados en la web usando el plugin Unityweb player, que es un componente que permite interpretar código y los mapas de archivos 3D.

Características

a) Fácil instalación en ambientes Windows y Mac.
b) Interfaz de usuario amigable, compuesta por cinco partes: 1) vista de escena; 2) vista de juego; 3) vista de proyecto; 4) vista de jerarquía; 5) vista de inspector.
c) Modos de visualización 3D en perspectiva.
d) Fácil configuración de la visualización utilizando componentes: 1) Render Mode; 2) Color Modes; 3) Interruptor de luces; 4) Interruptor de skybox, lense flare y niebla.
e) Manejo de botones de control para comandos de transformación.

4.2 Librerías y Componentes del Unity

Las principales librerías que maneja el Unity son seis como se explican a continuación:

1) Creación de escenas: Unity tiene un DLL que maneja la creación de escenas en 3D donde se ubican todos los elementos u objetos del juego en 3D como planos, edificios, terrenos, cielo, personajes, etc.
2) Generador de terrenos: Unity tiene un DLL para generar terrenos los mismos que son generados como una malla plana que se puede texturizar y esculpir sin salir del editor. Los terrenos tienen algunas propiedades importantes, como la longitud del terreno y algunas propiedades que controlar el nivel de detalle del terreno.
3) Renderizado: Unity tiene una DLL para setear el renderizado de los objetos 3D de un juego y añadir efectos especiales a un juego como: a) Niebla (Fog); b) Color de la Niebla (Fog Color); c) Luz de Ambiente (Ambient Light); d) Material de la Caja del Cielo (Skybox Material); e) Fuerza de la Luz (Halo Strenght); f) Fuerza del Fuego (Flare Strenght); g) Textura de la Luz (Halo Texture); h) Mancha de Galleta (Spot Cookie)
4) Controlador de Primera Persona: Unity incluye un DLL para el Controlador Estándar en Primera Persona que hace que los juegos con vista en primera persona sean realmente sencillos de configurar. Para manejar este controlador en la escena, se debe debe ir a la vista de proyecto y seleccionar “Standard Assets -> Prefabs”. Dentro de esa carpeta hay un prefabricado (prefab) llamado “First Person Controller”. Se arrastra este objeto a la vista de escena y se lo posiciona de forma que el cilindro toque el terreno.
5) El componente MonoBehaviour: Utilizado pata la programación de scripts de Unity, en dos lenguajes de programación que son el C# y el Java.
6) El componente GameApp del Juego: Este componente maneja toda la interfaz de usuario y la computación gráfica del juego y se lo puede integrar a otras clases creadas por el usuario, como en el caso del presente proyecto que consta de más de 20 clases.

4.3 Construcción del Videojuego

4.3.1. Creación del Modelo 3D en Maya

Maya de Autodesk se utilizó para la creación de los gráficos y objetos, edición de materiales en 3D y la configuración de la iluminación del juego. Los elementos que se diseñaron en Maya para Unity fueron (ver Figura 4.1):

- El plano del laberinto.
- Los cubos del laberinto que representan las paredes.

El ratón que representa al usuario en modo manual y a la PC en el modo automático.

4.3.2. Creación del Videojuego en Unity

El videojuego fue desarrollado enteramente con el motor de juegos Unity y con la herramienta de programación Mono C# para la creación de las clases del juego del componente GameApp creado, las mismas que son:

- ***Menu:*** Esta clase se utiliza para manejar todos los elementos que conforman el menú principal, como son las opciones de: a) Trial; b) Principiante; c) Intermedio; d) Avanzado; e) Puntajes.
- ***Scores:*** Esta clase se utiliza para controlar el puntaje del juego obtenido por los usuarios y la PC.

- ***GameControl:*** Esta clase controla la ejecución del juego independientemente de las opciones de: a) Trial; b) Principiante; c) Intermedio; d) Avanzado.
- ***FarCamera:*** Esta clase permite visualizar todo el laberinto a través de una cámara.
- ***FollowCamera:*** Esta clase permite que una de las cámaras siga al ratón que está en el laberinto.
- ***Loading:*** Esta clase permite manejar la ventana que carga el juego y opera un buffer de comunicación que calcula el porcentaje de carga del mismo.
- ***GameMode:*** Esta clase controla el tipo de juego que puede ser manual, donde el que opera el juego es un usuario; o también puede ser automático, donde el que opera el juego es la PC con IA. Además esta clase se encarga de almacenar los estados del juego como son el nombre del usuario, el nivel de dificultad (principiante, intermedio, avanzado) y el tiempo que se demora en completar el laberinto.
- ***MazeAlgorithm:*** Esta clase se encarga de realizar todas las operaciones matemáticas y generar con IA el mapa del laberinto, lo cual permite obtener una matriz de unos y ceros, donde los ceros representan el camino a seguir y los unos representan las paredes del laberinto.
- ***MazeCreator:*** Esta clase se encarga de pasar del entorno matemático a un entorno virtual en 3D, para lo cual utiliza la clase ***MazeAlgorithm*** para operar el laberinto.
- ***MouseBrain:*** Esta clase se encarga de controlar el movimiento del ratón en modo automático, donde se utiliza IA para analizar la ruta a seguir y llegar a la meta que consiste en encontrar el queso. Esta clase utiliza parte de los algoritmos de la clase ***MazeAlgorithm*** para moverse por el camino representado por un grupo de ceros y almacena en una lista enlazada todas las posiciones recorridas basada en la regla de la mano izquierda.
- ***PlayerController:*** Esta clase controla manualmente el movimiento del ratón por el laberinto.
- ***ParticleTester:*** Esta clase se encarga de manejar un sistema de partículas para generar los efectos de explosión que tiene el juego y que se produce cuando el usuario representado por un pequeño ratón, encuentra el queso en el laberinto.
- ***GUICamera:*** Esta clase se encarga de controlar la interfaz gráfica del usuario en 2D, donde consta el menú y sus opciones.
- ***CameraController:*** Esta clase se encarga de administrar las dos cámaras del juego.
- ***XMLDataBase:*** Esta clase se utiliza para manejar los archivos planos del juego donde se almacena los datos del juego.

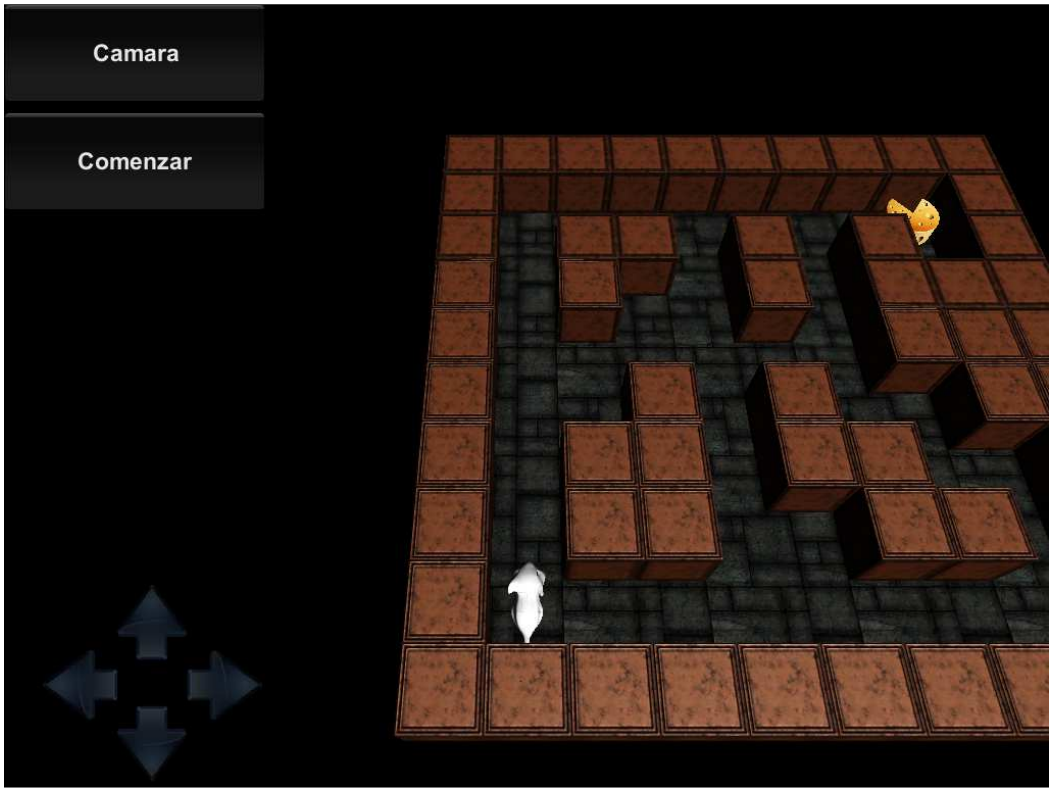

Figura 4.1. Elementos del Videojuego.

Adicionalmente se utilizó tres componentes básicos del motor de juegos Unity, los mismos que son:

- ***Componente CoreUnity:*** Este componente es el centro de creación de aplicativos del Unity, ya que se constituye en el alma del Game Engine del Unity, donde se encuentran todas las clases que manejan toda la computación gráfica del juego como: vectores, matrices, imágenes, texturas, color, fuentes, objetos 2D y 3D, luces, cámaras, animaciones, física y transformación de objetos.
- ***Componente MonoBehaviour:*** Este componente es un conjunto de DLLs implementados por el proyecto Mono C# para Unity que es compatible con C#.NET y es el que se encarga de compilar el programa con las librerías del Unity y manipular los objetos mediante scripts de programación.
- ***Componente GUI:*** Este componente es el que permite crear las interfaces gráficas del usuario en 2D como los componentes para formularios tales como: botones, etiquetas, sliders, etc.

4.4. Pruebas de la Aplicación

Se estableció un proceso de pruebas que inicio con la verificación de navegación, estándar de colores, tamaño y tipo de letra; para finalizar con la verificación de infraestructura y seguridad de la aplicación. Se realizaron las siguientes pruebas con la finalidad de resolver todos los posibles conflictos de conectividad, desempeño y navegabilidad de la Aplicación:

4.4.1 Prueba de Contenido

Se ha realizado con varios niños (usuarios) una revisión minuciosa de los siguientes tipos de contenido:

- Estático: Referente a la información estática que se muestra en la aplicación.
- Dinámico: Referente a la información encontrada en los archivos planos de datos que está integrada a la aplicación tanto para Windows como para la Web; que en este caso es un 100% por el hecho de utilizar el Game Engine Unity.

A través de la técnica de observación y lectura de todos los elementos y enlaces dela aplicación y sin olvidar el correcto agrupamiento de los temas a ser mostrados; se ha logrado corregir los 4 aspectos más relevantes de los contenidos que son:

- Errores tipográficos y/o equívocos gramaticales.
- Errores semánticos (información incompleta o ambigua)
- Errores en la organización de la información para ser mostrada al usuario final.
- Errores de obtención de información incorrecta de los archivos planos de datos, al momento de ser desplegada en las diferentes ventanas de la aplicación.

4.4.2 Prueba de Interfaz de Usuario

Se ha tomado en cuenta las siguientes consideraciones:

4.4.2.1 Prueba de Mecanismos de la Interfaz

A) Enlaces: Cada uno de los enlaces de la aplicación, sean estos enlaces internos o externos a la aplicación, se enlacen al objeto deseado.

B) Formato: El computador recibe toda la información y no existe pérdida de datos en la ejecución de la aplicación, los campos del formato tienen el ancho y tipos de datos adecuados.

C) Ventanas Dinámicas: La navegación en cada una de las ventanas de la aplicación maneja memoria dinámica para construir y destruir los objetos que tiene cada una de estas con los scripts de programación hechos en Mono C# para garantizar su correcto despliegue.

4.4.2.2 Prueba de Facilidad de Uso

Para este tipo de pruebas se ha tomado en consideración aspectos como: grado de usabilidad (es fácil en-

contrar lo que se busca), interacción con el usuario (menús desplegables, botones, estética (colores), despliegue (resolución de la pantalla).

4.4.3. Prueba de Navegación

Para la realización de esta prueba se ha establecido una verificación de todos los enlaces de la aplicación. Se ha analizado junto a los usuarios que los enlaces creados lleven hacia el contenido o la funcionalidad adecuada y sobretodo que estos enlaces sean comprensibles conforme se realiza la navegación.

Además se ha verificado que los nombres de los nodos sean significativos para los usuarios, como por ejemplo: a) Menú Principal; b) Modo Manual; c) Modo Automático; d) Trial; e) Nivel Principiante; f) Nivel Intermedio; g) Nivel Avanzado; h) Puntajes del Juego. Con esta prueba se ha logrado ejercitar ampliamente la navegación de la aplicación por parte de los desarrolladores y de los usuarios finales.

4.4.4. Prueba de Componentes

Las pruebas a nivel de componente se las ha realizado primeramente integrando el componente desarrollado en el juego que es el GameApp sin ningún inconveniente en su creación. Se ha realizado pruebas en función de la entrada de datos en formularios, definiendo los tipos de datos permitidos para cada uno de los campos de entrada, verificando que exista una correcta validación de la información, que la información se envíe y se reciba desde los archivos planos hasta la aplicación y que no exista perdida de información durante la ejecución del programa.

4.4.5. Prueba de Configuración

Se ha analizado la arquitectura (Cliente / Servidor) que maneja Unity para crear aplicaciones de tipo desktop para Windows y se ha especificado lo siguiente:

- Se ha instalado la versión de Unity y configurado para que funcione en modo local y no distribuido.
- Se ha generado un archivo ejecutable que corra bajo Windows y un applet para que corra en una tableta.

4.4.6. Prueba de Seguridad

La aplicación no implementa ninguna seguridad de acceso, por lo que puede cualquier usuario acceder a la aplicación. Además la aplicación puede ser copiada e instalada en cualquier computadora con el sistema operativo Windows y no tendrá ningún problema de funcionamiento.

5. CONCLUSIONES

- Para la creación del juego se utilizó el game engine Unity el cual utiliza un conjunto de componentes para manejo de gráficos y de la física de objetos en tres dimensiones.
- Las pruebas de contenido, función, estructura, facilidad de uso, navegabilidad, y desempeño; ayudaron a detectar y corregir los errores antes de la puesta en producción del aplicativo en 3D.
- Para el presente trabajo en función de los requerimientos del videojuego, el game engine Unity cubrió por sí solo el 65%, el 35% fue adaptación y desarrollo propio. En esta conclusión se considera como medida del software el número de componentes que se utilizaron para desarrollar esta aplicación. Unity aportó con 3 componentes (CoreUnity, MonoBehaviour, GUI). Se adaptó y se creó un componente (GameApp).
- Al final del presente trabajo se puede apreciar que la correcta utilización del game engine Unity, fue posible crear un videojuego con excelentes características de funcionalidad, navegabilidad, desempeño y compatibilidad.
- En la presente aplicación de un videojuego en 3D se utilizó la metodología OOHDM con UML, debido a que es una metodología orientada al diseño y desarrollo de aplicaciones multimedia tipo desktop, web y

móviles, aportando con diagramas útiles y prácticos que permiten llevar un proceso de desarrollo organizado y eficiente.

6. RECOMENDACIONES

- Realizar una adecuada configuración del motor de juegos (game engine) Unity, ya que existe una gran variedad de módulos para implementar nuevas funcionalidades a los juegos en 3D como la física y el movimiento de objetos utilizando inteligencia artificial y redes neuronales entre algunos de los módulos que se pueden encontrar en el mercado para Unity.
- Para la adecuada selección de componentes adicionales a ser instalados en el motor de juegos Unity, se recomienda escoger los que tengan mayores comentarios positivos en los foros y con mayores votaciones, ya que serían los más calificados por su estabilidad y funcionalidad.
- En todo proyecto desktop, Web o móvil, se debería definir una fase de pruebas que podría ser integrada con el desarrollo del proyecto o como un documento separado para controlar la calidad del producto desde sus inicios de creación.
- Utilizar el motor de juegos Unity cuando cubra hasta más de un 50% de los requerimientos solicitados por los usuarios, ya que con su gran variedad de extensiones, facilidad de uso y documentación actualizada se pueden adaptar y crear las funcionalidades adicionales.
- Que se formen grupos de trabajo de al menos dos personas o de pares para que uno de ellos se centre al diseño y modelado en 3D, mientras que el otro se dedique al proceso de programación para acortar los tiempos de desarrollo e implementación de los videojuegos. Además se recomienda que tengan conocimientos básicos de programación orientada a objetos, de animación e IA, para que en el proceso de creación de videojuegos adquieran los conocimientos extras necesarios para cumplir con este objetivo.
- Utilizar el motor de juegos Unity para la creación y mantenimiento de videojuegos, ya que no solo se ahorra tiempo y dinero, sino que se logra una verdadera concepción de las tareas de un desarrollador de aplicativos de simulación 3D tal como ciertos autores lo consideran al videojuego, el cual tendrá todas las facilidades para: expandir y mejorar la estética, navegabilidad y servicios de entretenimiento que es el meollo de un videojuego.

7. REFERENCIAS

Libros:

- PRESSMAN Roger, Ingeniería de Software Un Enfoque Práctico, Mc. Graw Hill, Madrid – España, 2002.
- LAMARCA María Jesús, Hipermedia/Multimedia. Universidad Complutense de Madrid – 2006.
- GARCÍA RUBIO Ramón, QUIRÓS SUÁREZ Javier, GONZÁLEZ Santiago Martín, SANTOS GALLEGO Ramón, FERNÁNDEZ MORÁN Samuel;Diseño Gráfico de Contenidos para Internet, Pearson Prentice Hall
- BLACKMAN Sue, Beginning 3D Game Development with Unity, Apress, New York – USA, 2011.
- SCHNEIDER Electric Instituto, Manual de Formación Unity Pro, Instituto Schneider Electric, Barcelona – España, 2008.
- KELLER Eric, PALAMAR Todd, HONN Anthony, Mastering Autodesk Maya 2011, Sybex, USA, 2010.

Web:

- Página Oficial de Unity (2012)
  Disponible: http://unity3d.com/
- Arocena, F. (Abril 2003), Crea tu Página Web
  Disponible: http://wmaestro.com/webmaestro/docs/portada.html

- Consejos de estilo Gráfico para WWW (2006)
  Disponible: http://dmi.uib.es/people/acoca/estilo/index.htmml
- Página Oficial de Maya (2012)
  Disponible: http://usa.autodesk.com/maya/