DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA

TESIS PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS E INFORMÁTICA.

TEMA:ESTUDIO COMPARATIVO DE LOS MOTORES GRÁFICOS WEBGL 'THREE.JS' Y 'BABYLON.JS' PARA EL DESARROLLO DE UN JUEGO EDUCATIVO.

AUTOR:VACA DE LUCCA, ANDRÉS ENRIQUE

DIRECTORA: ING. ZAMBRANO, MARGARITA

CODIRECTORA: ING. HINOJOSA, CECILIA

SANGOLQUÍ

2015

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por el Sr. ANDRÉS ENRIQUE VACA DE LUCCA como requerimiento parcial a la obtención del título de INGENIERO EN SISTEMAS E INFORMÁTICA.

Sangolquí, Mayo de 2015

ING. MARGARITA ZAMBRANO
DIRECTORA DE TESIS

ING. CECILIA HINOJOSA
CODIRECTORA DE TESIS

AUTORÍA DE RESPONSABILIDAD

Yo, Andrés Enrique Vaca De Lucca declaro que el presente trabajo es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación personal y que he consultado las referencias bibliográficas que se incluyen en el documento.

La Universidad de las Fuerzas Armadas ESPE puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

Sangolquí, Mayo de 2015

Andrés Enrique Vaca De Lucca

AUTORIZACIÓN

Yo, **ANDRÉS ENRIQUE VACA DE LUCCA**, autorizo a la Universidad de las Fuerzas Armadas ESPE la publicación, en la biblioteca virtual de la Institución, el trabajo **"ESTUDIO COMPARATIVO DE LOS MOTORES GRÁFICOS WEBGL 'THREE.JS' Y 'BABYLON.JS' PARA EL DESARROLLO DE UN JUEGO EDUCATIVO"**, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y autoría.

Sangolquí, Mayo de 2015

Andrés Enrique Vaca De Lucca

DEDICATORIA

A Dios.
Por haberme dado la oportunidad de culminar con éxito mis estudios, y darme la fortaleza para llegar hasta aquí.

A mis padres.
Por su apoyo incondicional, y por haberme alentado a seguir adelante en cada instante de mi vida, y saber que siempre puedo contar con ellos.

A mi hermano.
Por impulsarme día a día a ser su ejemplo a seguir, y por apoyarme constantemente, motivándome a cumplir mis metas.

AGRADECIMIENTO

A todas aquellas personas que de alguna manera u otra han sido parte de mi vida, y a las que aún lo son, quiero agradecerles de todo corazón.

Mami, por enseñarme a tener fe en Dios,darme cada día tu bendición, confiar en mí,sembrar en mí el deseo de triunfar y animarme a llegar cada día más lejos.

Papá, quiero darte mil gracias por tu constante apoyo y por motivarme a la realización de mis metas.

Erick mi querido hermano, porque desde el momento en que viniste al mundo, has alegrado mi existencia, sabiendo que siempre puedo contar contigo.

Mis tíos y más familiares, por alentarme a seguir adelante.

A todos mis amigos y compañeros, por esos momentos que compartimos y a la vez por haberme permitido ser parte sus vidas. Por todo eso, ¡Gracias!

De corazón, agradezco a todos y cada uno de mis profesores, por compartir sus conocimientos, pero sobre todo por brindarme su amistad y apoyarme a lo largo de esta etapa tan importante de mi vida.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

A pesar que las aplicaciones web han evolucionado hasta el punto de poseer todas las funcionalidades de una aplicación informática nativa, se han presentado limitaciones respecto al ámbito de gráficos 3D, como lo son la portabilidad y la estandarización de los mismos. La tecnología WebGL pretende solventar estas limitaciones ya que es una API abierta, estándar, multiplataforma y no requiere de complementos para su ejecución. El objetivo del presente proyecto fue el de realizar un estudio comparativo de dos motores gráficos libres, con el fin de determinar cuál tiene mejores prestaciones para desarrollar un juego educativo, utilizando dicha tecnología. Este estudio comparativo se llevó a cabo utilizando un modelo de calidad basado en la norma ISO/IEC 25010, enfocado en las características de: adecuación funcional, eficiencia de desempeño y usabilidad. Para la implementación del juego, se utilizó la metodología OOHDM y la guía de desarrollo de juegos propuesta por Jeannie Novak. El juego educativo que se desarrolló tiene como propósito aportar al proceso constructivista de la atención y memoria de niños preescolares entre 4 a 6 años de edad. Como resultado del presente proyecto, el modelo de calidad permitió determinar el motor gráfico más adecuado al juego que se desarrolló, y contribuye como referencia para destacar los aspectos principales que deben estar presentes en un motor gráfico para plataformas web. La guía anteriormente señalada en conjunto con la metodología, guiaron con éxito la creación del juego, mediante un diseño completo y detallado del mismo.

PALABRAS CLAVE:

- **WEBGL**
- **HTML5**
- **ISO/IEC 25010**
- **JUEGO EDUCATIVO**
- **MOTORES GRÁFICOS**

ABSTRACT

Even though web applications have evolved until having all functionalities of native applications, some limitations in 3D graphics have been encountered, like portability and standardization. The WebGL technology tries to solve these limitations, because it is an open, standard and multiplatform API that does not require add-ons for its execution. The main goal of this project was to perform a comparative study of two free and open source graphics engines in order to determine which one has the best features to develop an educational game using that technology. The comparative study was achieved using a quality model based on the ISO/IEC 25010 standard. The OOHDM methodology and the game development guide purposed by Jeannie Novak were used in the development process of the project. The aim of the educational game was to contribute to the cognitive process of the memory and attention of children aged between 4 and 6 years. The results of this project determined the best graphics engine for development of the game, and a reference of the main features that should be included in graphics engines for the web platform. The game development guide and the OOHDM methodology guided to a detailed and complete design of the game.

KEYWORDS:

- **WEBGL**
- **HTML5**
- **ISO/IEC 25010**
- **EDUCATIONAL GAME**
- **GRAPHICS ENGINES**

CAPÍTULO 1:INTRODUCCIÓN.

1.1 Tema

Estudio comparativo de los motores gráficos WebGL ‘Three.js’ y ‘Babylon.js’ para el desarrollo de un juego educativo.

1.2 Antecedentes

En los inicios de la computación gráfica, los fabricantes de hardware desarrollaban sus propias librerías gráficas, lo cual hacía costoso el soporte de las aplicaciones entre las diferentes plataformas.

La compañía SGI notó que la carencia de una API estándar iba a inhibir el crecimiento del mercado de gráficos 3D, y decidió liderar un grupo de industrias en la creación de dicho estándar. El resultado de esa iniciativa es 'OpenGL'. (Silicon Graphics International Corp., 2009).

'OpenGL' está a cargo del grupo Khronos, que se dedica a: investigar, desarrollar y especificar librerías gráficas abiertas para ser implementadas en diferentes plataformas tecnológicas. El grupo Khronos también está a cargo de ‘WebGL’, que es la adaptación y estandarización de 'OpenGL ES' para aplicaciones web, con la que se obtiene más portabilidad y rendimiento de gráficos 3D, ya que se ejecuta en el navegador y la aceleración gráfica se consigue utilizando los recursos de hardware. (The Khronos Group, 2011).

1.3 Planteamiento del problema

La evolución de la tecnología web ha pasado de ser páginas de contenido estático a sitios web con todas las funcionalidades de una aplicación informática. Sin embargo, se han presentado limitaciones para las aplicaciones que usan esta tecnología en contraste con las aplicaciones nativas. Una de esas limitaciones es que las interfaces de usuario no son tan ricas, y por lo tanto han impactado en el uso y la portabilidad de los gráficos 3D en las aplicaciones web. (Anyuru, 2012) .

Con WebGL, se pretende solventar estas limitaciones ya que es una API: estándar, multiplataforma y abierta. Además, no requiere de complementos adicionales y se ejecuta directamente en navegadores soportados. (The Khronos Group, 2011).

1.4 Justificación

Actualmente existen diversos motores gráficos, librerías y frameworks que utilizan WebGL. Considerando las ventajas que ofrece esta tecnología, se propone realizar un estudio comparativo de dos motores gráficos libres, con el fin de determinar cuál tiene mejores prestaciones para desarrollar un juego educativo, utilizando un modelo de calidad basado en la norma ISO/IEC 25010.

Se escogió a 'Three.js' por su facilidad de aprendizaje y uso, además de estar entre los motores gráficos WebGL más populares. Ha sido utilizado para desarrollar numerosos demos. Según el autor Tony Parisi, 'Three.js' es poderoso, extensible, rápido, robusto, y utiliza las mejores prácticas en técnicas de motores gráficos. Además, recibe soporte y contribución de diversos autores. (Parisi, 2014).

La popularidad de 'Three.js' es evidente debido a que existen libros en los cuales se aprende el desarrollo de aplicaciones WebGL mediante su uso. Además, existen empresas que lo han utilizado para promocionar sus productos o servicios como son:

- Google Inc. en los denominados 'Chrome Experiments'.
- Warner Bros. para promocionar sus películas: 'Gravity' y 'TheHobbit: thedesolation of Smaug'.
- Disney para promocionar su película 'Oz, thegreat and powerful'.
- Samsung, quien auspició un demo WebGL optimizado para dispositivos móviles.
- Nike, que lo utilizó para promocionar su catálogo de zapatos deportivos.
- Nestlé, que lo utilizó para promocionar su té 'Ô Green'.

Los demos anteriormente listados fueron presentados en el sitio web oficial de ‘Three.js’, el día en que se realizó la consulta para el presente proyecto. (Cabello, 2010).

‘Babylon.js’ fue escogido por su facilidad de uso, sus características y sus prestaciones, que son propias de un motor de ‘juegos hardcore’. Estos aspectos son mencionados por el autor Tony Parisi. (Parisi, 2014).

Este motor fue desarrollado por el empleado de Microsoft David Catuhe, y se han publicado numerosos demos WebGL auspiciados por esta empresa. También ha sido utilizado en demos de Galerías de Arte y para promocionar el juego ‘Assassin's Creed: Pirates’ de la empresa Ubisoft.

Fue utilizado para probar la compatibilidad de WebGL en la versión 11 de Internet Explorer. En el blog oficial de ‘Babylon.js’, su autor sostiene que este motor puede ser utilizado en la creación de aplicaciones para la plataforma Windows 8.1. (Catuhe, 2013).

Los demos anteriormente mencionados, aparecieron en el sitio web oficial de ‘Babylon.js’, el día que se realizó la consulta para el presente proyecto.(Babylon.js team, 2013).

1.5 Objetivos

1.5.1 Objetivo General

Realizar el estudio comparativo de los motores gráficos WebGL ‘Three.js’ y ‘Babylon.js’ para la implementación de un juego educativo que aporte en el desarrollo cognitivo de niños de edad preescolar.

1.5.2 Objetivos Específicos

- Realizar el análisis documental de los motores gráficos WebGL ‘Three.js’ y ‘Babylon.js’.
- Realizar el análisis, diseño y desarrollo del juego educativo utilizando la metodología OOHDM junto con la guía de desarrollo de juegos de J. Novak.
- Realizar las pruebas de funcionamiento y eficiencia del juego educativo.

1.6 Alcance

- La finalidad del presente proyecto es el de presentar el resultado del estudio comparativo de los motores gráficos WebGL.
- También se incluye en el proyecto, el desarrollo del juego educativo con el motor gráfico que resulte ganador.
- Por motivos de pruebas el juego se alojará en un host gratuito.
- El desarrollo del juego educativo no incluirá la fase de post producción, debido a que no existirán nuevas características o funcionalidades por agregar después de su versión final.

1.7 Juego Educativo

El juego educativo aportará en el desarrollo cognitivo en niños preescolares de entre 4 a 6 años de edad, ayudando al proceso constructivista de la atención y la memoria. (Morrison, 2005).

Estará conformado por 4 mini juegos, con tres niveles de dificultad: fácil, intermedio y difícil. Los niveles de dificultad permitirán que el juego se adecué a la edad apropiada del jugador.

Los 4 mini juegos son:

- Encontrar las parejas de animales (ayuda a la memoria).
- Recolectar las frutas indicadas (ayuda a la atención).
- Memorizar los animales y encontrarlos al abrirse el telón.
- Memorizar los objetos que se guardan en las mochilas.

CAPÍTULO 2:MARCO TEÓRICO.

En este se capítulo se tratarán fundamentos teóricos de computación gráfica, también se tratará la metodología de desarrollo y la guía del proceso de creación de juegos propuesta por Jeannie Novak para implementar el juego educativo. Al tratarse de un juego, el presente proyecto vendría a ser otro tipo de software, por consiguiente la metodología escogida, OOHDM, debe apoyar al desarrollo del mismo. Esta metodología fue creada para facilitar la creación aplicaciones Web que interactúan con contenido multimedia.

2.1. Aspectos importantes de gráficos computacionales

2.1.1. Efectos especiales

Existen numerosos efectos especiales para juegos en entornos tridimensionales como son:

Luces:

La iluminación tiene un rol esencial en la creación de efectos visuales. A continuación se muestran los diferentes tipos de luces existentes en un motor gráfico:

- **Static Light:** Luz estática. Este tipo de luz se utiliza para reducir el impacto en el rendimiento de los gráficos, ya que requiere de menos cálculos que la luz dinámica.

  Permite pre calcular la iluminación de todos los objetos estáticos que estén en una escena para crear una textura con las sombras que se aplicarán sobre dicha escena.A la técnica anteriormente mencionada se la conoce como “light mapping”.

  La calidad de las sombras y la iluminación del “light map” resultante depende de la resolución del mismo. **(Gregory, 2009)**.

  En la siguiente figura se muestra un ejemplo de “light mapping” con luz estática. La resolución de la textura resultante es baja, con la finalidad de demostrar cómo funciona dicha técnica.

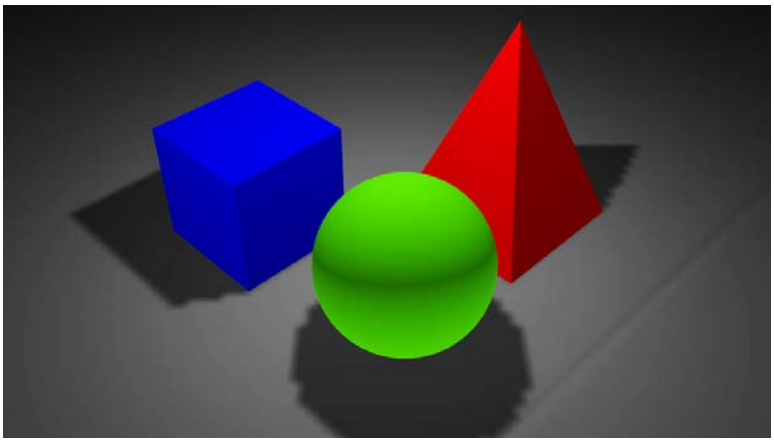


Figura 1. “Light mapping” con luz estática.

- **Point Light:**Punto de luz. Este tipo de luz es omnidireccional, es decir que, ilumina uniformemente en todas la direcciones.
  La luz tiene atributos de: intensidad, color, posición y radio máximo (el cual determina la atenuación).(Gregory, 2009).
  En la siguiente figura se muestra un ejemplo un punto de luz ubicado en el centro de tres objetos.

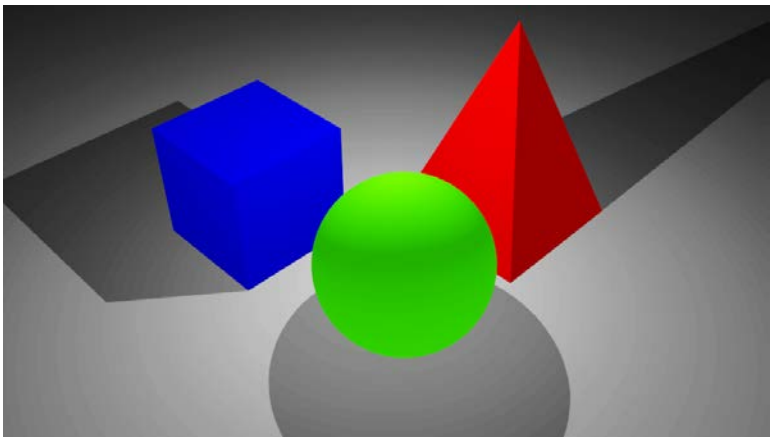


Figura 2. Ejemplo de punto de luz.

- **Directional Light:**Luz direccional. Se comporta como una fuente de luz que se ubica a una distancia infinita de la superficie a ser iluminada.Los rayos que provienen de ésta son paralelos, y no requiere de ninguna ubicación particular dentro de la escena.

Este tipo de luz tiene atributos de color y de vector dirección.(Gregory, 2009).

En la siguiente figura se muestra un ejemplo de luz direccional.

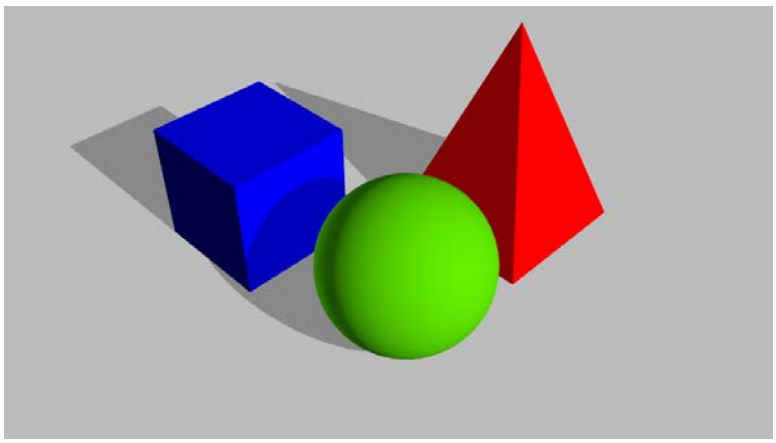


Figura 3. Ejemplo de luz direccional.

- **Spot Light:** Luz focal. Actúa como un punto de luz, con la diferencia de que sus rayos se limitan a una región cónica. Usualmente se utilizan dos conos para especificar los parámetros de este tipo de luz.

  En el cono interno la luz tiene intensidad completa.La intensidad de la luz se va opacando en el espacio existente entre el cono interno y el externo.

  Los atributos de este tipo de luz son: vector de posición, color, vector de dirección central, radio máximo y los ángulos del cono interno y externo.(Gregory, 2009).

  En la siguiente figura se muestra un ejemplo de luz focal.

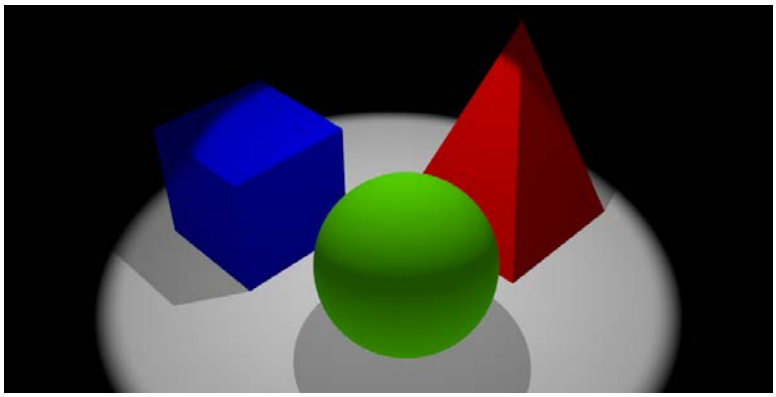


Figura 4. Ejemplo de luz focal.

- **Ambient Light:**Luz ambiental. Consiste en utilizar un color específico de luz para dar más realismo a las escenas donde no hay luz directa. Este tipo de luz es independiente del ángulo y no tiene dirección. En la siguiente figura se muestra un ejemplo de luz ambiental. (Gregory, 2009).

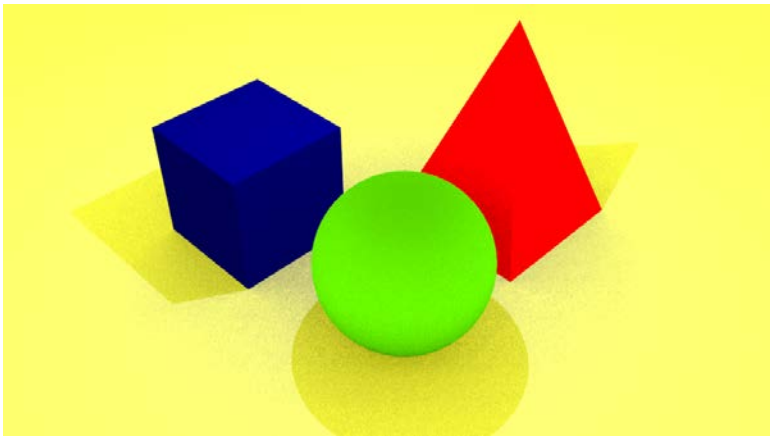


Figura 5. Ejemplo de luz ambiental.

Materiales:

Un material es una descripción completa de las propiedades visuales de una malla tridimensional. Esto incluye la especificación de las texturas asignadas a su superficie, así como también propiedades de alto nivel tales como: los shaders a utilizarse, los parámetros de configuración para dichos shaders, y los parámetros de entrada para el control de las funcionalidades gráficas.

Un modelo tridimensional generalmente se compone de más de un material, por esta razón una malla se puede dividir en sub mallas, cada una asignada a un material en particular.(Gregory, 2009).

2.1.2. Sistemas de partículas

Los sistemas de partículas son ampliamente usados para generar efectos de fenómenos naturales tales como: fuego, humo, explosiones, chispas, diferentes tipos de clima (lluvia, nieve, etc.), y efectos visuales imaginarios como hechizos y otros tipos de poderes.

Estos efectos visuales son muy difíciles de crear manualmente y si fuesen realizables, no se verían realistas. Por esta razón, se concibió la idea de programar la simulación de estos efectos de una forma más realista usando conceptos de física.

Básicamente, consiste en un conjunto de partículas que tienen propiedades tales como peso y tiempo de vida, y que son afectadas por las propiedades físicas del mundo virtual como lo son la gravedad y la dirección del viento. Los sistemas de partículas también contienen emisores cuyas propiedades son la tasa de emisión y la fuerza. (Kyaw & Stein, 2011).

2.2. Shaders

Un shader es un programa diseñado para ejecutarse en una determinada etapa del procesamiento gráfico. Su propósito es ejecutar una de las etapas programables del canal de renderizado.(The Khronos Group, 2013). Los shaders mejoran los efectos visuales de una imagen renderizada. Agregan realismo a las escenas o le dan efectos especiales.

2.3. Cámaras

Existen 2 tipos de cámara para renderizar una escena 3D: ortogonal o isométrica, y perspectiva. A continuación se detallan cada una de ellas:

- **Ortogonal:**

  Este tipo de cámara proyecta a la escena de forma isométrica, y permite obtener vistas precisas de los tamaños de los objetos que aparezcan en dicha escena. La percepción de las escenas proyectadas por esta cámara se asemeja a las de un plano.(Sukin, 2013).

- **Perspectiva:**

  Este tipo de cámara muestra a la escena desde un solo punto en el espacio, asemejándose a la realidad, ya que se produce un efecto de distorsión debido a la distancia.(Sukin, 2013).

En la siguiente figura se muestra una comparación de los tipos de cámara anteriormente expuestos.

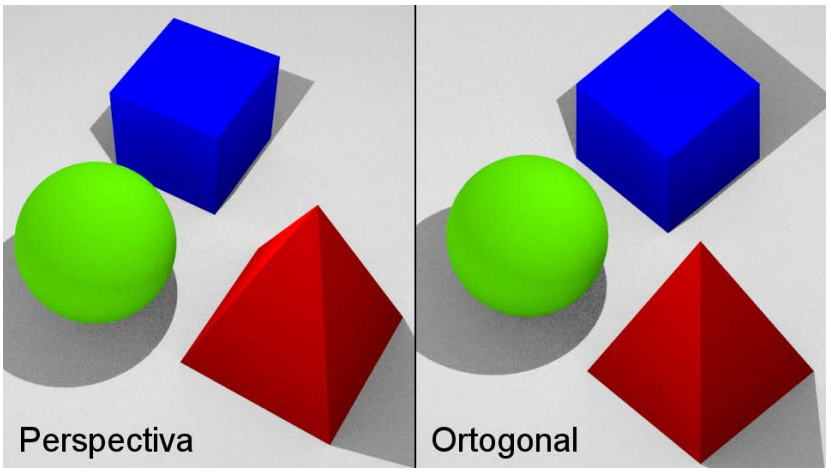


Figura 6. Cámaras de perspectiva y ortogonal.

2.4. Mallas tridimensionales y animaciones

Se puede definir a una malla como el conjunto de vértices, bordes y caras que definen la forma de un objeto poligonal en un espacio tridimensional. Sobre las mallas se aplican los materiales, los mismos que contienen las propiedades visuales de éstas.(Kyaw & Stein, 2011).

En la siguiente figura se muestra un ejemplo de una malla tridimensional representando a una ballena.

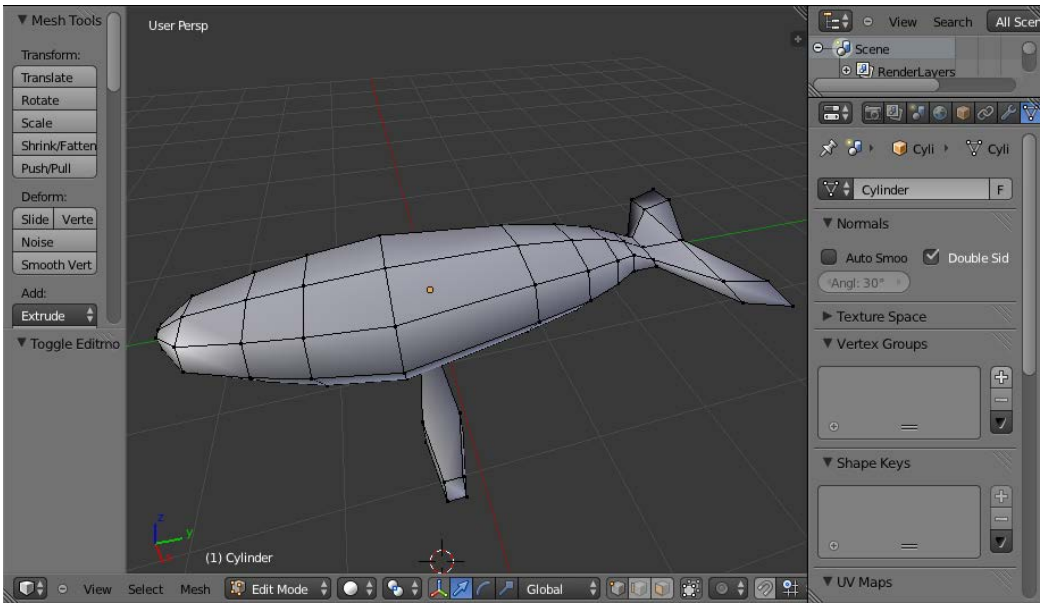


Figura 7. Ejemplo de una malla tridimensional.

Una animación consiste en una serie de imágenes con ligeros cambios que, cuando se ven en una secuencia rápida, crean la ilusión de movimiento.

La animación de una malla se obtiene a través de fotogramas claves que indican la posición, tamaño y rotación en un determinado instante del tiempo que dura dicha animación.

Para facilitar la creación y manipulación de animaciones en una malla, se puede utilizar un esqueleto asociado a los vértices que la componen. Este proceso se conoce como "Rigging".

El proceso de manipular las deformaciones de la malla con el esqueleto asociado se conoce como "Skinning", y permite configurar los fotogramas claves del esqueleto para conseguir la animación.(Mullen, 2011). En la siguiente figura se muestra un ejemplo de "Skinning":

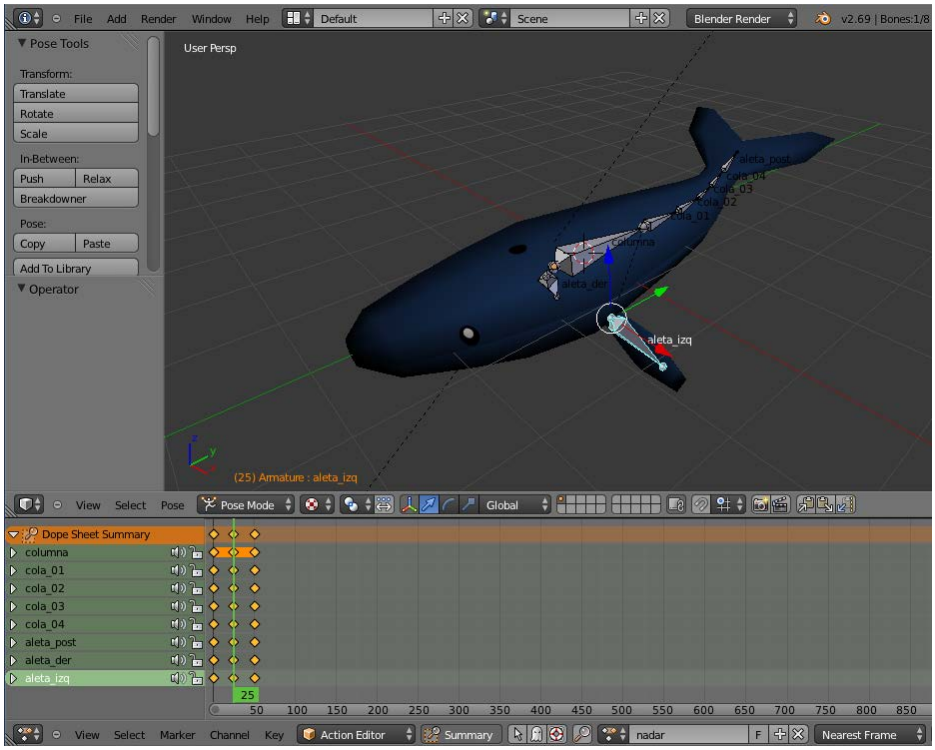


Figura 8. Ejemplo de “Skinning” sobre una malla texturizada.

2.5. Software para modelado de objetos 3D

Suites de animación 3D, como lo son: Blender, 3DS Max o Maya, proveen las herramientas necesarias para modelar, animar y exportar

objetos y escenas 3D en múltiples formatos, para que puedan ser utilizados en otros productos software.

Para el presente proyecto se escogió a Blender, debido a que es una suite de modelado y animación 3D de código abierto, multiplataforma y con la posibilidad de integrar los plugins que proveen los motores gráficos del estudio comparativo a realizar. (Blender Foundation, 2011).

2.6. Software para manipulación de imágenes

Para el presente proyecto se requiere de una herramienta capaz de editar y exportar mapas de bits.La herramienta de manipulación de imágenes escogida es GIMP, debido a que es de código abierto y posee las funcionalidades necesarias para editar el tamaño, calidad, y gama de colores de las imágenes que conformarán las texturas del juego y los componentes de la interfaz gráfica. (The GIMP Team, 2012).

2.7. Three.js

Está licenciado como un producto software de código abierto bajo las condiciones de la licencia “MIT”.(Cabello, 2012). Las principales características de Three.js son:

- Renders usando tecnologías: WebGL, canvas, svg, CSS3D, DOM y por Software. Incluye efectos de render en 3D anaglífico.
- Manipulación de escenas con la posibilidad de añadir y remover objetos en tiempo de ejecución.
- Cámaras de perspectiva y ortográfica. Incluye controladores de cámara como: trackball, FPS y de trayectoria.
- Animaciones: mallas animadas a través de fotogramas (keyframe) y deformación de vértices (morph).
- Luces: ambientales, direccionales, de punto y hemisféricas. También permite la proyección de sombras.
- Permite la ejecución de shaders con acceso a las prestaciones de WebGL y efectos pos procesamiento.

- Manipulación de objetos como: mallas, partículas, imágenes 2D, esqueletos, etc.
- Geometrías primitivas del motor como: planos, cubos, esferas, torus y textos 3D.
- Cargadores de: archivos binarios, imágenes, archivos JSON y de escenas.
- Utilidades para: funcionalidades de tiempo, y funciones matemáticas.
- Utilidades para importar y exportar archivos al formato JSON compatible con Three.js.

2.8. Babylon.js

Es de código abierto, su uso y distribución están sujetos bajo la licencia Apache 2.0. (Chenard, Barreau, & Lepère, 2014).Babylon.js posee las siguientes características:

- Render completo de escenas con luces, cámaras, materiales y mallas.
- Motor de colisiones.
- Motor de físicas (mediante cannon.js)
- Selección y manipulación de escenas.
- Antialiasing.
- Motor de animaciones.
- Sistemas de partículas.
- Capas 2D.
- Motores de optimización.
- Materiales estándares que usan shaders por pixel y con soporte de animaciones con esqueleto.
- Efectos especiales como: niebla, multi vistas y capas de render.
- Efectos para texturas como: videos sobre texturas, texturas dinámicas y texturas comprimidas.
- Manipulación de mallas.
- Permite la exportación de escenas y objetos a través de complementos para programas de modelamiento 3D.

2.9. Metodología de desarrollo: OOHDM

OOHDM (ObjectOrientedHypermediaDesignMethod) será aplicada como metodología de desarrollo para el juego educativo.Los diagramas de esta metodología estarán presentes en el documento de diseño técnico. En la siguiente figura se muestran las fases que la componen con sus respectivos diagramas. (Rossi, Pastor, Schwabe, & Olsina, 2007).

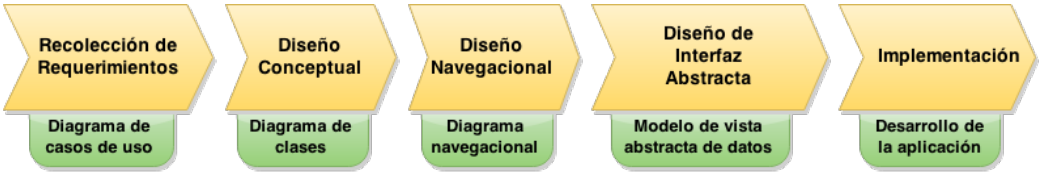


Figura 9. Fases de OOHDM

2.9.1. Recolección de Requerimientos.

Este es el primer paso de la metodología, consiste en identificar a los stakeholders (parte interesada) y las tareas que deben realizar.

Luego se reúnen los escenarios para cada tipo de tarea y actor, para formar casos de uso, los cuales son representados a través de los Diagramas de Interacción con Usuarios (UID por sus siglas en inglés).

2.9.2. Diseño Conceptual.

En este paso se construye un modelo conceptual usando principios de modelado orientado a objetos. Lo que realmente concierne en esta fase es la semántica del dominio de la aplicación. OOHDM utiliza UML para construir dichos diseños conceptuales.

2.9.3. Diseño Navegacional.

En OOHDM, una aplicación es tratada como una vista navegacional sobre el modelo conceptual. Para cada perfil de usuario se puede definir una estructura navegacional que refleje a los objetos y sus relaciones en el esquema conceptual acorde a las tareas que el usuario debe realizar.

2.9.4. Diseño de Interfaz Abstracta

El modelo de interfaz abstracta es construido definiendo los objetos perceptibles que contengan información (contenido multimedia) en términos de interfaces. Las interfaces son definidas como agregaciones recursivas de clases primitivas, o como otras clases de interfaces.

2.9.5. Implementación

La implementación abarca la integración de la interfaz y objetos de navegación con los objetos de tiempo de ejecución y puede involucrar arquitecturas elaboradas.

2.10. Guía de desarrollo de juegos

En la siguiente figura, se ilustran las fases de la guía de desarrollo propuesta por Jeannie Novak, con sus respectivos entregables. (Novak, 2012).

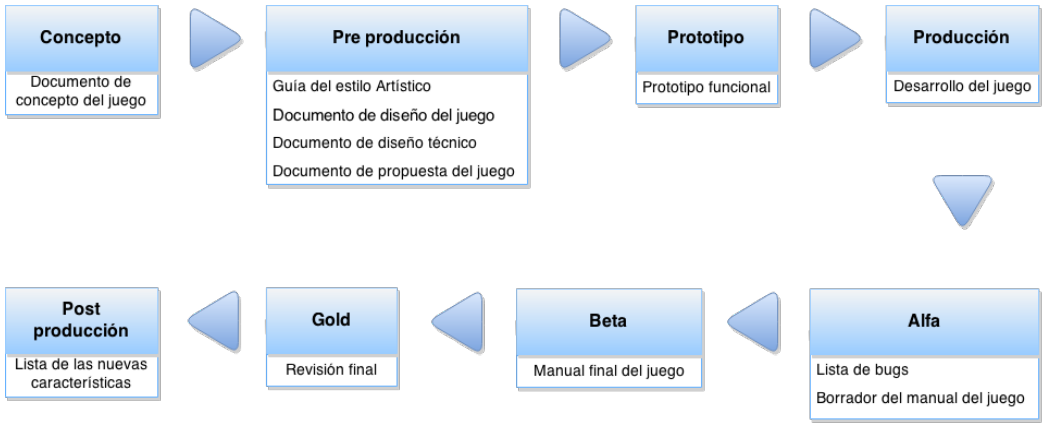


Figura 10. Fases de la guía de desarrollo de juegos

2.10.1. Concepto

Esta fase comienza conla concepción inicial de la idea del juego, y finaliza con las decisiones finalessobre dicha idea para empezar a planificar el proyecto. En esta fase se realizan las siguientes actividades:

- Definición de la temática del juego.
- Identificación de la audiencia a quién está dirigida.

- Cálculo de los recursos necesarios, tales como: costos, personal, herramientas y plataforma tecnológica.

El entregable de esta fase es el documento de concepto del juego. Este documento incluye los siguientes puntos:

- Premisa: Abarca la idea básica del juego. Consiste en un resumen del propósito del juego y su temática.
- Motivación del jugador: Resume brevemente la condición de victoria del juego, y responde a las preguntas: ¿Cómo el jugador gana?, ¿Qué motivaría al jugador completar el juego?
- Propuesta única de venta: Resume las características que hacen único y especial al juego.
- Audiencia objetivo: Especifica la audiencia a la cual está dirigida el juego, indicando el rango de edad y datos importantes como demográficos y psicológicos.
- Género: Consiste en identificar el estilo al cual pertenece el juego, mediante su modo de juego.
- Clasificación objetivo: Clasifica al juego acorde al nivel de madurez de la audiencia objetivo.
- Plataforma objetivo y requerimientos de hardware:Especifica la plataforma tecnológica en la cual se ejecutará el juego e incluye los requerimientos técnicos mínimos y recomendados para ejecutarse en dicha plataforma.(Novak, 2012).

2.10.2. Pre-producción

La fase de pre-producción es la concerniente a la planificación del juego. La documentación de esta fase abarca: el documento de propuesta deljuego, la guía del estilo artístico, el documento de diseño del juego, y el documento de diseño técnico. A continuación se detalla el contenido de cada uno de dichos documentos:

Documento de Propuesta del juego:

El propósito de este documento es el de presentar los detalles del juegocon el fin de explicar al equipo de trabajo cómo empezar a planificarlo. Este documento debe contener las secciones:

- Atracción: Consiste en determinar los elementos característicos que volverán atractivo al juego.
- Modo de juego: Describe cómo será el estilo deljuego, qué mecánica utilizará y cómo debe desenvolverse el jugador a medida que avanza en el mismo.
- Características en línea: Describe las funcionalidades del juego en el ámbito de interacción en línea y características multi jugador.
- Tecnología: Esta sección es opcional, y debe ser incluida solo si se planea incorporar alguna tecnología especial de software o hardware para el funcionamiento del juego.
- Características de Arte y Audio: Consiste en describir los atributos del estilo gráfico, la música y los sonidos que harán que el juego sea único.
- Detalles de producción: Esta sección debe incluir detalles tales como: el equipo de desarrollo, el presupuesto, y cronograma. También se debe incluir costos estimados y roles en los equipos de trabajo.
- Historia de fondo: Consiste en resumir los precedentes a la historia del juego. Esto debe ser incluido solo si el juego sigue una trama.
- Sinopsis de la historia: Esta sección solo es aplicable si el juego tiene trama. Describe la misma sin entrar en mucho detalle y enfocándose en la idea principal de la historia.
- Descripción de personajes: Resume las características de los personajes.

- Análisis de riesgos: Esta sección describe los posibles problemas que puedan presentarse para la creación del juegoy sus correspondientes soluciones.
- Presupuesto de Desarrollo: Esta sección se enfoca en identificar los costos necesarios para lanzar el juego al mercado, estimando también los ingresos que éste generará.
- Arte conceptual: Esta sección describe el estilo gráfico de los personajes y escenarios relacionados al juego. Puede incluir dibujos y bosquejos de los mismos.

Documento de Diseño del Juego:

El propósito de este documento es el de servir como una guía de referencia para el proceso de desarrollo del juego.

Se enfoca en: el modo de juego, la historia, los personajes, la interfaz y las reglas del juego.A continuación se muestran las secciones que componen a este documento:

- Interfaz del juego: En esta sección se describen: los elementos a ser incluidos en cada interfaz, el tiempo de producción requerido, el costo de producción, viabilidad de la interfaz para la audiencia objetivo y las características de usabilidad.
- Mundo del juego: Esta sección debe describir: los elementos presentes en cada nivel (incluyendo cinemáticos), el arte, el modo de juego, animaciones, personajes e ítems del juego.
- Habilidades de personajes: Esta sección describe las acciones que puede realizar o no personajes controlados por el jugador, como también los no controlables (Non- player character, NPC).
- Motor de Juego: El propósito de esta sección delimitar el alcance del motor gráfico a utilizarse, e incluye los siguientes elementos: número de personajes que pueden ser presentados en pantalla al mismo tiempo, número de animaciones por personaje y demás detalles de rendimiento.

Guía del estilo artístico:

El propósito de este documento es el de establecer el aspecto visual del juego (Look and feel). Este documento referencia a la creación de una librería visual que direccionará todo el arte gráfico del juego.

Documento de diseño técnico:

Está basado en el documento de diseño de juego. Describe las especificaciones del motor gráfico a utilizarse y la documentación necesaria para trascender desde el concepto del juego a su posterior codificación.

2.10.3. Prototipo

Esta fase consiste en la creación de un prototipo funcional que se enfoque más en el modo de juego que en el estilo visual del mismo. El prototipo debe capturar la esencia del juego presentando las principales características que tendrá la versión completa.

2.10.4. Producción

En esta fase se desarrolla el juego completo, asignando tareas por cada equipo de trabajo y acorde a un cronograma. Cabe recalcar que esta es la fase más larga dentro del proceso de producción de un juego.

2.10.5. Alfa

En esta fase el juego se encuentra desarrollado en su totalidad, y por tanto, se puede jugar de principio a fin. Aquí es donde comienza el proceso de depuración para detectar fallos y corregirlos. La documentación de esta fase abarca:

- Lista de bugs detectados en el juego.
- Borrador del manual del juego.

2.10.6. Beta

Esta fase se enfoca en la corrección de fallos, y su objetivo es el de estabilizar el proyecto eliminando cuantos bugs sean posibles antes del lanzamiento del producto.Documentación de esta fase:

- Manual final del juego.

2.10.7. Gold

En esta fase se revisa el juego para su posterior despliegue en la plataforma para la cual fue desarrollado.

2.10.8. Post Producción

También conocida como post lanzamiento. Durante esta fase se desarrollan versiones del juego en los que se añaden características y mejoras para aumentar su tiempo de vida. Si el juego aplica para una fase de post producción se debe documentar las funcionalidades y características agregadas.

2.11. Características de la NormaISO/IEC 25010

El estudio comparativo de los motores gráficos se basará en las siguientes características del modelo de calidad de software de la norma ISO/IEC 25010 (International Organization for Standardization, 2011):

2.11.1. Adecuación funcional

Consiste en la capacidad del producto software para ajustarse a las necesidades y los requisitos funcionales del usuario. Se divide en:

- Completitud funcional: se refiere al grado en el que las funcionalidades del producto cubren los objetivos especificados.
- Corrección funcional: se refiere al nivel de precisión de los resultados correctos del producto.
- Pertinencia funcional: se refiere a las funciones proporcionadas por el producto, para cumplir los objetivos especificados.

2.11.2. Eficiencia de desempeño

Describe cómo el producto responde a los requerimientos de los usuarios y qué tan eficiente es en su ejecución. Se divide en:

- Comportamiento en el tiempo: se mide el tiempo de respuesta bajo las condiciones especificadas en un conjunto de pruebas establecido.
- Utilización de recursos: la cantidad de recursos utilizados cuando el producto se ejecuta bajo condiciones específicas.
- Capacidad: el grado de cumplimiento de los requisitos dentro de los límites máximos del producto.

2.11.1. Usabilidad

Comprende los aspectos referentes a la facilidad de utilizar el producto. Esta característica se divide en:

- Facilidad para reconocer si resulta apropiado acorde a los requerimientos.
- Facilidad de aprendizaje del producto software.
- Operabilidad: facilidad de ser operado y contralado.
- Protección contra errores de usuario: facilidad del producto para proteger a sus usuarios de cometer errores.
- Estética de interfaz de usuario: grado en el que la interfaz permite una interacción agradable y satisfactoria.
- Accesibilidad: grado en el que el producto puede ser utilizado por la mayor cantidad de personas con diferentes características y capacidades, dentro de un uso específico.

CAPÍTULO 3:ESTUDIO COMPARATIVO

3.1. Modelo de evaluación.

Para construir el modelo de evaluación, se tomaron en cuenta las características de la norma ISO/IEC 25010 más acordes a los requerimientos del juego a desarrollar y las prestaciones de los motores gráficos. Estas características son: adecuación funcional, eficiencia de desempeño y usabilidad.

La característica de adecuación funcional permitirá determinar si los motores gráficos se ajustan a los requerimientos del juego.

La eficiencia de desempeño determinará el rendimiento de los motores yla cantidad de recursos que necesitan para cumplir con los requerimientos.

La característica de usabilidad ayudará a precisar el motor con la mayor facilidad de aprendizaje.

En la siguiente tabla se detalla el criterio de ponderación para las características con sus correspondientes justificaciones:

Tabla 1.

Criterio de ponderación

Característica	Ponderación	Justificación
Adecuación funcional	30%	Es importante determinar las características y prestaciones de los motores gráficos con la finalidad de utilizar el que más se adecúe a las necesidades y requerimientos para el desarrollo del juego
Eficiencia de desempeño	40%	El desempeño de un motor gráfico es crucial para una aplicación que requiere interacción en tiempo real, como es el caso del juego
Usabilidad	30%	La facilidad de aprendizaje y uso del motor gráfico es un factor determinante que ayuda al rápido desarrollo del juego

Las versiones de los motores gráficos con los que se realizó el estudio comparativo son:

- Babylon.js 1.14
- Three.js r69

3.1.1. Adecuación funcional.

A continuación se muestra el modelo de evaluación de la adecuación funcional consus correspondientes subcaracterísticas.

Tabla 2.

Adecuación funcional

Subcaracterística	Completitud funcional	
Atributos	**Métrica**	**Descripción**
Animación por esqueleto (SkeletalAnimation)	Sí = 1 No = 0	El motor gráfico permite animar una malla 3D a través de los fotogramas clave que se le asignaron a su esqueleto
Animación por deformación de vértices (Morph)	Sí = 1 No = 0	El motor gráfico permite animar una malla 3D mediante la deformación de sus vértices
Funcionalidades extensibles	Sí = 1 No = 0	Las funcionalidades del motor gráfico se pueden extender y complementar junto con librerías adicionales
Subcaracterística	Corrección funcional	
Atributos	**Métrica**	**Descripción**
Búsqueda e interacción de objetos instanciados en escena	Sí = 1 No = 0	El motor gráfico permite buscar un objeto ya instanciado en una escena e interactuar con el mismo
Añadir o remover objetos de la escena en tiempo de ejecución	Sí = 1 No = 0	El motor gráfico permite añadir nuevos objetos a la escena, así también como removerlos, mientras la aplicación está ejecutándose
Subcaracterística	Pertinencia funcional	
Atributos	**Métrica**	**Descripción**
Formato propio para escenas y mallas 3D	Sí = 1 No = 0	El motor gráfico tiene su propio formato de escenas y mallas 3D (sean estáticas o animadas)
Permite cargar otros formatos de mallas 3D	Sí = 1 No = 0	Además de trabajar con el formato propio del motor gráfico, se puede cargar mallas 3D en otros formatos
Permite configurar animaciones	Sí = 1 No = 0	Permite configurar propiedades de las animaciones tales como: velocidad, ciclo repetitivo o no, reproducir, pausar, etc.
Existen exportadores para software de modelado 3D	Sí = 1 No = 0	Se disponen de complementos o extensiones para software de modelado 3D que permitan exportar al formato propio del motor gráfico
Permite el manejo de cámaras de perspectiva y ortogonal	Sí = 1 No = 0	El motor gráfico permite parametrizar el tipo de cámara ya sea esta de perspectiva u ortogonal
Contiene materiales predeterminados para las mallas 3D	Sí = 1 No = 0	El motor gráfico contiene materiales predeterminados y parametrizables
Permite utilizar shaders personalizados GLSL	Sí = 1 No = 0	Además de los shaders predeterminados en el motor gráfico, también se pueden usar shaders externos escritos en GLSL.
Permite efectos post-procesamiento	Sí = 1 No = 0	Se pueden agregar efectos visuales como por ejemplo: desenfoque, refracción, etc.
Incorpora un sistema de partículas	Sí = 1 No = 0	El motor gráfico incluye un sistema de partículas parametrizable
Incluye diferentes tipos de luces parametrizables	Sí = 1 No = 0	El motor gráfico incluye diferentes tipos de luces parametrizables para iluminar la escena conforme a las necesidades requeridas
El render soporta antialiasing	Sí = 1 No = 0	El render del motor gráfico soporta antialiasing (filtro correctivo para la distorsión visual)

3.1.2. Eficiencia de desempeño.

El siguiente modelo de evaluación corresponde a la eficiencia de desempeño con sus respectivas subcaracterísticas.

Tabla 3.

Eficiencia de desempeño

Subcaracterística	Comportamiento en el tiempo	
Atributos	**Métrica**	**Descripción**
Cantidad de FPS (fotogramas por segundo)	FPS Mayor cantidad de FPS = 1, Menor cantidad de FPS = 0	Eficiencia para renderizar una escena 3D durante el tiempo establecido que durará la prueba (el resultado es el promedio de los FPS registrados)
Subcaracterística	Utilización de recursos	
Atributos	**Métrica**	**Descripción**
Uso de GPU (Tarjeta Gráfica)	MB Menor cantidad utilizada = 1, Mayor cantidad utilizada = 0	Cantidad de memoria de la tarjeta gráfica que se utilizó para realizar la prueba (el resultado es el promedio de la memoria utilizada)
Espacio que ocupa las mallas 3D exportadas	KB Menor cantidad utilizada = 1, Mayor cantidad utilizada = 0	Cantidad de KB que pesan las mallas 3D exportadas al formato propio del motor gráfico (el resultado es el peso en conjunto de dichas mallas)
Subcaracterística	Capacidad	
Atributos	**Métrica**	**Descripción**
Busca cargar los contenidos multimedia directamente desde el indexedDB	Sí = 1 / No = 0	Por defecto, el motor gráfico busca almacenar los contenidos multimedia al indexedDB (API de almacenamiento del navegador), para así cargarlos más rápido e inclusive permitir trabajar offline.

3.1.3. Usabilidad.

El modelo de evaluación que se presenta a continuación corresponde a las diferentes subcaracterísticas de la usabilidad de los motores gráficos.

Tabla 4.

Usabilidad

Atributos	Métrica	Descripción
Subcaracterística	Facilidad para reconocer si resulta apropiado	
Su licencia permite su uso en diversos tipos de proyectos	Especificar la licencia Sí = 1 No = 0	Su licencia permite su uso en proyectos académicos u otros tipos de proyectos, ya sean estos con o sin fines lucrativos
En constante soporte y desarrollo	Sí = 1 No = 0	Recibe el soporte de una comunidad de desarrolladores para su constante mejora
Repositorio histórico de versiones	Sí = 1 No = 0	Existe un repositorio histórico de las diferentes versiones existentes
Subcaracterística	Facilidad de aprendizaje	
Documentación oficial	Sí = 1 No = 0	Cuenta con documentación oficial
Wiki Oficial	Sí = 1 No = 0	Se dispone de una wiki oficial cuyos contenidos estén relacionado al uso de las herramientas y aprendizaje del motor gráfico
Videotutoriales	Sí = 1 No = 0	Existen videotutoriales ya sean oficiales o difundidos por una comunidad
Ejemplos en línea	Sí = 1 No = 0	Se disponen de ejemplos en el sitio web oficial
Libros	Sí = 1 No = 0	Utilizado como herramienta principal para el aprendizaje de WebGL, y no solamente citado en los textos
Subcaracterística	Operabilidad	
Controles de cámara integrados	Sí = 1 No = 0	Incorpora funcionalidades para controlar el movimiento de la cámara, facilitando así interactividad con la escena
Reproducción de animaciones sin especificar fotogramas claves	Sí = 1 No = 0	Permite cargar y reproducir las animaciones sin necesidad de especificar los fotogramas claves de inicio y fin, facilitando la configuración de dichas animaciones
Desarrollado bajo el paradigma de orientación a objetos	Sí = 1 No = 0	Las funcionalidades del motor gráfico fueron desarrolladas bajo el paradigma de orientación a objetos
Posee una herramienta propia para medir el desempeño	Sí = 1 No = 0	Dispone de una herramienta para medir el desempeño de su renderer y mostrar así la cantidad de fotogramas por segundo (FPS)
Subcaracterística	Protección contra errores de usuario	
Alternativas si hay inconvenientes en el renderer WebGL	Sí = 1 No = 0	Dispone de alternativas en caso de que existieran problemas al inicializar el renderer de WebGL
No requiere especificar el nombre de la textura asociada a la malla 3D	Sí = 1 No = 0	Permite cargar mallas 3D sin necesidad de especificar el nombre de su textura asociada, evitando problemas de escritura de la misma
Subcaracterística	Estética de interfaz de usuario	
Provee de algún método o función para redimensionar el renderer	Sí = 1 No = 0	Se puede redimensionar el renderer cuando cambia de tamaño el canvas en el que está trabajando
Render con fondo transparente	Sí = 1 No = 0	El motor gráfico permite renderizar la escena con fondo transparente
Subcaracterística	Accesibilidad	
Optimización del renderer	Sí = 1 No = 0	Incorpora niveles preestablecidos de optimización para el renderer
Incorpora una utilidad de capturas de pantalla desde el motor gráfico	Sí = 1 No = 0	El motor gráfico dispone de alguna herramienta para tomar capturas de pantalla

3.2. Ejecución

3.2.1. Pruebas de rendimiento.

Las pruebas de rendimiento fueron realizadas el 15 de diciembre de 2014, en computadoras de diversos sistemas operativos y características de hardware y usando el navegador Google Chrome para medir la cantidad de GPU utilizada (versión del navegador:39.0.2171.95 m o superior).A continuación se muestra la tabla de dichas pruebas:

Tabla 5.

Pruebas de rendimiento de los motores gráficos

No.	Sistema Operativo	RAM	CPU	GPU (Tarjeta Gráfica)	Promedio de FPS		Cantidad de GPU (Máx. 512 MB)	
					Three	Babylon	Three	Babylon
1	Mac OS X 10.10.1	4 GB	2,3 GHz Intel Core i5	Intel HD Graphics 3000 384 MB	54,11	59,66	3,00	3,40
2	Mac OS X 10.7.5	12 GB	2,5 GHz Intel Core i5	AMD Radeon HD 6750M 512 MB	52,92	52,99	6,00	6,50
3	Windows 7 Home Premium 64 bits	6 GB	1,73 GHz Intel Core i7	Nvidia GeForce GT 330M 1024 MB	53,59	57,63	5,30	5,80
4	Windows 7 Home Premium 64 bits	6 GB	2,00 GHz Intel Core i7	Nvidia GeForce GT 540M 1024 MB	67,97	70,00	8,60	8,50
5	Windows 7 Ultimate 64 bits	6 GB	3,20 GHz Intel Core i5	NvidiaGeForce GT 9500 1024 MB	53,95	58,48	5,30	5,80
6	Windows 7 Ultimate 64 bits	4 GB	3,10 GHz Intel Core i5	Nvidia GeForce GTX 750 Ti 2048 MB	54,35	56,05	4,00	4,00
7	Windows 8 64 bits	4 GB	2,90 GHz Intel Pentium	Intel HD Graphics 32 MB	54,39	47,33	8,00	8,50
8	Windows 8 64 bits	4 GB	1,80 GHz Intel Celeron	Intel HD Graphics 32 MB	54,18	58,95	5,00	4,90
9	Windows 8 Pro 64 bits	4 GB	2,90 GHz Intel Pentium	Intel HD Graphics 256 MB	52,51	59,74	3,50	3,40
10	Windows 8 Pro 64 bits	6 GB	2,40 GHz Intel Core i5	Intel Graphics HD 3000 32 MB	52,87	58,51	3,00	3,40
Promedio Final:					55,09	57,94	5,17	5,42

3.2.2. Adecuación funcional.

En la siguiente tabla se muestran los resultados del modelo de evaluación para la característica de adecuación funcional, aplicado a los motores gráficos.

Tabla 6.

Resultados de la evaluación: adecuación funcional

Subcaracterística	Completitud funcional		
Atributos	**Métrica**	**Three**	**Babylon**
Animación por esqueleto (SkeletalAnimation)	Sí = 1 No = 0	1	1
Animación por deformación de vértices (Morph)	Sí = 1 No = 0	1	0
Funcionalidades extensibles	Sí = 1 No = 0	1	1
Subcaracterística	Corrección funcional		
Atributos	**Métrica**	**Three**	**Babylon**
Búsqueda e interacción de objetos instanciados en escena	Sí = 1 No = 0	1	1
Añadir o remover objetos de la escena en tiempo de ejecución	Sí = 1 No = 0	1	1
Subcaracterística	Pertinencia funcional		
Atributos	**Métrica**	**Three**	**Babylon**
Formato propio para escenas y mallas 3D	Sí = 1 No = 0	1	1
Permite cargar otros formatos de mallas 3D	Sí = 1 No = 0	1	0
Permite configurar animaciones	Sí = 1 No = 0	1	1
Existen exportadores para software de modelado 3D	Sí = 1 No = 0	1	1
Permite el manejo de cámaras de perspectiva y ortogonal	Sí = 1 No = 0	1	1
Contiene materiales predeterminados para las mallas 3D	Sí = 1 No = 0	1	1
Permite utilizar shaders personalizados GLSL	Sí = 1 No = 0	1	1
Permite efectos post-procesamiento	Sí = 1 No = 0	1	1
Incorpora un sistema de partículas	Sí = 1 No = 0	1	1
Incluye diferentes tipos de luces parametrizables	Sí = 1 No = 0	1	1
El render soporta antialiasing	Sí = 1 No = 0	1	1
Puntaje Total		16	14

3.2.3. Eficiencia de desempeño.

En la siguiente tabla se muestran los resultados del modelo de evaluación para la característica de eficiencia de desempeño.

La evaluación de los motores gráficos se la realizó en base a los resultados de las pruebas de rendimiento. El código fuente utilizado para construir las pruebas de rendimiento se adjunta en este proyecto.

Tabla 7.

Resultados de la evaluación: eficiencia de desempeño

Subcaracterística		Comportamiento en el tiempo				
Atributos		**Métrica**	**Three**	**Puntaje**	**Babylon**	**Puntaje**
Cantidad de FPS	FPS	Mayor cantidad de FPS = 1 Menor cantidad de FPS = 0	55,09	0	57,94	1
Subcaracterística		Utilización de recursos				
Atributos		**Métrica**	**Three**	**Puntaje**	**Babylon**	**Puntaje**
Uso de GPU (Tarjeta Gráfica)	MB	Menor cantidad utilizada = 1 Mayor cantidad utilizada = 0	5,17	1	5,42	0
Espacio que ocupa las mallas 3D exportadas	KB	Menor cantidad utilizada = 1 Mayor cantidad utilizada = 0	488	1	1935,36	0
Subcaracterística		Capacidad				
Atributos		**Métrica**	**Three**		**Babylon**	
Busca cargar los contenidos multimedia directamente desde el indexedDB		Sí = 1 / No = 0	0		1	
Puntaje Total:			2		2	

3.2.4. Usabilidad.

A continuación se muestra la tabla con los resultados del modelo de evaluación para la característica de usabilidad de los motores gráficos.

Tabla 8.

Resultados de la evaluación: usabilidad

Subcaracterística	Facilidad para reconocer si resulta apropiado		
Atributos	**Métrica**	**Three**	**Babylon**
Su licencia permite su uso en diversos tipos de proyectos	Sí = 1 / No = 0 Especificar la licencia	1 Licencia MIT	1 Licencia Apache 2.0
En constante soporte y desarrollo	Sí = 1 / No = 0	1	1
Repositorio histórico de versiones	Sí = 1 / No = 0	1	1
Subcaracterística	Facilidad de aprendizaje		
Atributos	**Métrica**	**Three**	**Babylon**
Documentación oficial	Sí = 1 / No = 0	1	1
Wiki Oficial	Sí = 1 / No = 0	1	1
Videotutoriales	Sí = 1 / No = 0	1	1
Ejemplos en línea	Sí = 1 / No = 0	1	1
Libros	Sí = 1 / No = 0	1	0
Subcaracterística	Operabilidad		
Atributos	**Métrica**	**Three**	**Babylon**
Controles de cámara integrados	Sí = 1 / No = 0	1	1
Reproducción de animaciones sin especificar fotogramas claves	Sí = 1 / No = 0	1	0
Desarrollado bajo el paradigma de orientación a objetos	Sí = 1 / No = 0	1	1
Posee una herramienta propia para medir el desempeño	Sí = 1 / No = 0	1	1
Subcaracterística	Protección contra errores de usuario		
Atributos	**Métrica**	**Three**	**Babylon**
Alternativas si hay inconvenientes en el renderer WebGL	Sí = 1 / No = 0	1	0
No requiere especificar el nombre de la textura asociada a una malla 3D	Sí = 1 / No = 0	1	1
Subcaracterística	Estética de interfaz de usuario		
Atributos	**Métrica**	**Three**	**Babylon**
Provee de algún método o función para redimensionar el renderer	Sí = 1 / No = 0	1	1
Render con fondo transparente	Sí = 1 / No = 0	1	1
Subcaracterística	Accesibilidad		
Atributos	**Métrica**	**Three**	**Babylon**
Optimización del renderer	Sí = 1 / No = 0	0	1
Incorpora una utilidad de capturas de pantalla desde el motor gráfico	Sí = 1 / No = 0	0	1
Puntaje Total:		16	15

3.3. Conclusión

A continuación se muestran los resultados finales del estudio comparativo, tabulados con su respectivo puntaje obtenido acorde a la ponderación anteriormente expuesta.

Tabla 9.

Resultados finales del estudio comparativo de los motores gráficos

Características	Puntaje	Ponderación	Three.js		Babylon.js	
			Puntaje	%	Puntaje	%
Adecuación funcional	16	30%	16	30%	14	26%
Eficiencia de desempeño	4	40%	2	20%	2	20%
Usabilidad	18	30%	16	27%	15	25%
Total	38	100%	34	77%	31	71%

Con los resultados expuestos en la tabla 9, se determinó que el motor gráfico que cumple con la mayoría de las características del modelo de evaluación basado en la norma ISO/IEC 25010, es ‘Three.js’, que cumple con el 77% de dichas características.

Por consiguientese escoge ‘Three.js’ para el desarrollo del juego educativo.

CAPÍTULO 4: DESARROLLO DEL JUEGO EDUCATIVO

En este capítulo se presenta la documentación requerida en la guía de desarrollo de juegos propuesta por Jeannie Novak.

4.1. Concepto del juego

Premisa

La idea principal del juego es la de contribuir con el proceso constructivista de la atención y memoria en niños preescolares, y por ende también se busca que sea una herramienta de apoyo para los educadores.La temática consiste en 4 mini juegos cuyos personajes y objetos sean fáciles de identificar, memorizar, y que despierten el interés del jugador.

Motivación

La principal motivación es terminar con éxito el mini juego seleccionado dentro del tiempo establecido. Para conseguirlo, se muestran animaciones de los animales u objetos cuando el jugador realiza correctamente las acciones para completarlo.

Propuesta única de venta

Las características que hacen único y especial al juego son:

- Estilo gráfico de caricaturas, orientado a la audiencia de niños preescolares.
- Reglas y modo de juego simples.
- No requiere de complementos, ni descargas adicionales para ser jugado. Para ello se requiere de un navegador con soporte de HTML5 y WebGL.

Audiencia objetivo

La audiencia objetivo del juego son niños preescolares de entre 4 y 6 años de edad.

Género

El juego a desarrollar se ubica dentro del género educativo.

Clasificación

Acorde a la audiencia objetivo y el nivel de madurez de la misma, el contenido del juego se clasifica apto para mayores de 4 años.

Plataforma objetivo y requerimientos de hardware

Los navegadores con soporte de HTML5 y WebGL son la plataforma tecnológica a la que se orienta el juego. Estos navegadores deben ejecutarse sobre sistemas operativos de escritorio de 32 o 64 bits como: Windows 7, Windows 8, Mac OS X o GNU Linux.

Los requerimientos de hardware son:

- Procesador de doble núcleo de 2.9 GHz o superior.
- Tarjeta gráfica de 32 MB de video dedicado o superior.
- 1 GB de Memoria RAM o superior.

4.2. Propuesta del juego

Atracción

A continuación se listan los elementos característicos que harán atractivo al juego educativo:

- Escenarios y personajes en 3D con animaciones y sonidos.
- Diseño simple y minimalista de los personajes del juego, con estilo de caricatura.
- Reglas de juego simples para cada mini juego, con instrucciones del mismo antes de iniciar.
- Menú simple y de interfaz amigable para el usuario, con íconos intuitivos.
- Diferentes niveles de dificultad para cada mini juego.
- Registro de mejores puntuaciones.

Modo de juego

La condición de victoria para todos los mini juegos consiste en resolverlo antes de que se acabe el tiempo establecido. Para todos los mini juegos existen 3 niveles de dificultad: fácil, intermedio y difícil.

A continuación se describe el modo de juego (gameplay) para cada uno de los 4 mini juegos:

- Encontrar las parejas de animales:

  Este mini juego consiste en encontrar todas las parejas de animales que se esconden tras las cartas. La cantidad de cartas depende de la dificultad. Para revelar un animal se debe hacer click en la carta, y solo se pueden revelar 2 a la vez.

  Si las 2 cartas escogidas son diferentes, se esconden y el jugador no podrá hacer click sobre ninguna otra hasta que se terminen de ocultar, esto es con el propósito de evitar confusiones al revelar muchas cartas.

  Si las 2 cartas escogidas revelan al mismo animal, se muestra la animación y sonido correspondiente del mismo.

- Recolectar las frutas indicadas:

  El objetivo de este mini juego es el de identificar y recolectar la fruta indicada en el menor tiempo posible en cada ronda.La cantidad de rondas varía según la dificultad.

  El tiempo se reduce al iniciar cada ronda. Al completar una ronda el tiempo se incrementa en una cantidad específica que dependedel nivel de dificultad.

- Memorizar y encontrar los animales:

  Este mini juego consiste en memorizar los animales que aparecen en el escenario, y recordar cuáles estaban presentes para seleccionarlos al abrirse el telón. La cantidad de animales depende del nivel de dificultad.

Para escoger el animal, se debe hacer click sobre el mismo, y al hacerlo se iluminará dicho personaje. Si el animal escogido es incorrecto sonará una melodía que indica que el jugador se equivocó. Si el animal escogido es correcto se mostrará la animación y el sonido del mismo.

- Memorizar los objetos y mochilas:

  El objetivo de este mini juego es el de memorizar en qué mochila se guarda cada objeto, y luego recordar las asociaciones.

  La cantidad de éstos depende del nivel de dificultad.

  Para asociar se debe arrastrar el objeto con el mouse y soltarlo dentro de la mochila.

  Si la asociación está incorrecta sonará una melodía indicando que el jugador se equivocó. Por el contrario, si la asociación es correcta, se mostrará un visto y una melodía indicando que el jugador lo hizo bien.

Características en línea

Al ser una aplicación que utiliza WebGL, la aceleración gráfica y procesamiento del juego se realizará a nivel del cliente, solicitando al servidor los recursos multimedia para descargarlos al navegador cuando los necesite. El registro de las mejores puntuaciones se efectuará almacenando los datos en el navegador.

Tecnología

Las plataformas tecnológicas a utilizarse son HTML5 y WebGL. El juego está diseñado para ejecutarse en navegadores que soporten dichas tecnologías y que se ejecuten en sistemas operativos de escritorio, como se especificó previamente el documento de concepto del juego.

Características de Arte y Audio

- Atributos del estilo gráfico:

  El estilo gráfico del juego consiste en definir a los personajes con formas simples y minimalistas, de colores sólidos y texturas simplificadas. De igual forma los objetos también son simples y fáciles de identificar. Todo esto se detalla más adelante en la guía de estilo artístico.

- Música y sonido:

  La música de fondo del juego debe ser agradable y simple, orientada a motivar a los jugadores. Los efectos de sonido consisten en pequeñas melodías para las acciones correctas o erróneas dentro del juego. También se incluyen los sonidos característicos de los animales que aparecerán en el juego.

Detalles de producción

- Equipo de desarrollo:

  El presente proyecto contará con la participación única del autor del mismo, cubriendo todo el proceso de desarrollo.

- Cronograma:

  El cronograma se muestra en la sección 4.7 del presente capítulo: “Cronograma para el proceso de desarrollo”.

Descripción de personajes

Los personajes del juego son animales que se pueden encontrar en la granja. Todos los personajes tienen el mismo estilo gráfico de caricatura que se detallará más adelante en la guía del estilo artístico.

Las animaciones de los personajes corresponden a los movimientos característicos de estos animales cuando emiten su sonido.

Análisis de riesgos

A continuación se detallan los posibles riesgos que pueden presentarse en el desarrollo del juego educativo:

- Tamaño de los recursos multimedia:

  Los recursos multimedia como: mallas tridimensionales, imágenes, sonidos y música pueden llegar a ocupar mucho espacio de almacenamiento, lo que aumentaría el tiempo de descarga de los mismos en el navegador.

  Para solucionar este problema se deben optimizar dichos recursos con el uso de herramientas adecuadas.

- Problemas de almacenamiento para registrar las mejores puntuaciones:

  La plataforma tecnológica en la que se desarrollará el juego puede presentar limitaciones para almacenar información, y puede que no se tenga un fácil acceso al servidor para configurar un método de registro para las mejores puntuaciones.

  Para solucionar esto, se propone utilizar métodos de almacenamiento a nivel del navegador, como puede ser Web Storage o similares.

Presupuesto de desarrollo

Al ser desarrollado por el autor de la investigación, en el presupuesto se omite el costo del puesto de programador.

Tabla 10.

Presupuesto estimado para desarrollar el juego educativo

Herramienta	Cantidad	Costo	Costo total
Computador Sony Vaio VPCF-126FM	1	1200	1200
NetBeans IDE	1	0	0
Motor Gráfico WebGL Open Source	1	0	0
Navegador Web con soporte WebGL	2	0	0
Material Bibliográfico (eBooks)	4	20	80
Total			1280

Arte conceptual

A continuación se muestra el arte conceptual del juego a desarrollar, indicando los aspectos importantes de cómo se crearán los personajes y objetos del juego.

- Ejemplo de un animal del juego:

  En la siguiente figura se muestra un animal de granja, como una muestra de los que aparecerán en los mini juegos. El detalle de cada animal se indica en la guía del estilo artístico.

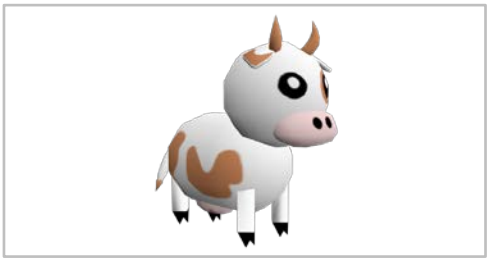


Figura 11. Ejemplo de un animal del juego

- Ejemplo de un objeto del juego:

  El estilo gráfico de los objetos es simple y con colores contrastantes. Los objetos del juego se detallan en la guía del estilo artístico. A continuación se muestra una figura donde se indica el estilo de uno:

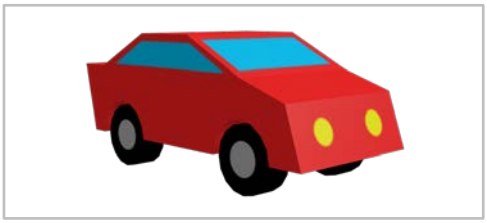


Figura 12. Ejemplo de un objeto del juego

4.3. Guía del estilo artístico

Diseño de los personajes del juego.

En esta sección se detalla el estilo gráfico de los personajes y el diseño de los mismos. Las características comunes a todos son:

- Ojos de canica:

  Los ojos son negros con un brillo blanco en la parte superior.

- Cuerpos de figuras geométricas:

  Los animales están conformados por figuras geométricas básicas, y en su versión final en un espacio tridimensional, estas figuras tienen pocos vértices para mantener la simplicidad y optimizar espacio de almacenamiento y cálculos del renderer.

- Uso de colores contrastantes:

  Las texturas de los animales son simples y tienen colores sólidos que contrastan entre sí y sin gradientes.

A continuación se muestran los diseños y modelos de los personajes:

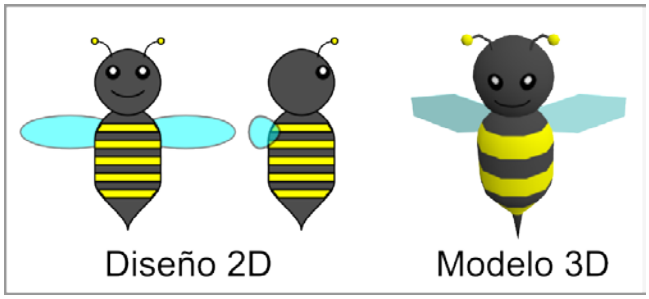


Figura 13. Diseño de la abeja

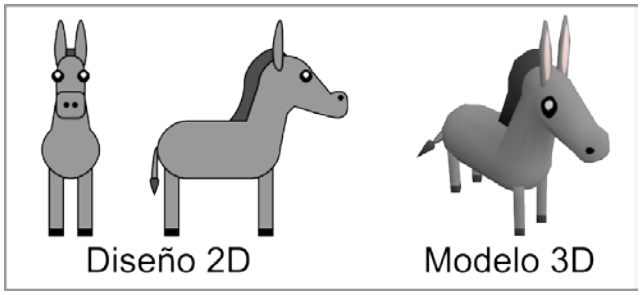


Figura 14. Diseño del burro

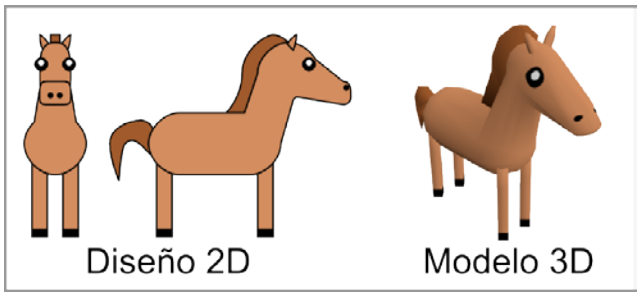


Figura 15. Diseño del caballo

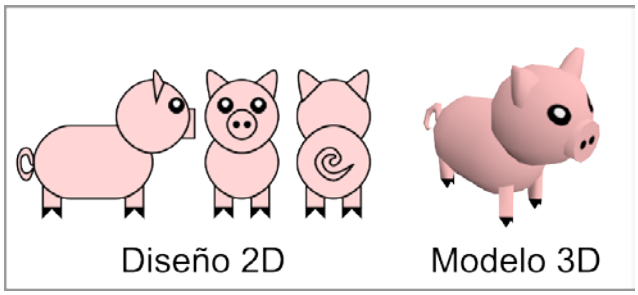


Figura 16. Diseño del cerdo

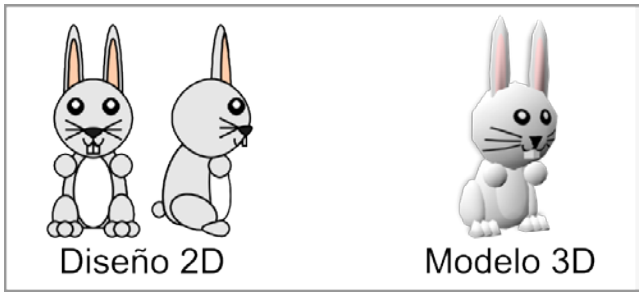


Figura 17. Diseño del conejo

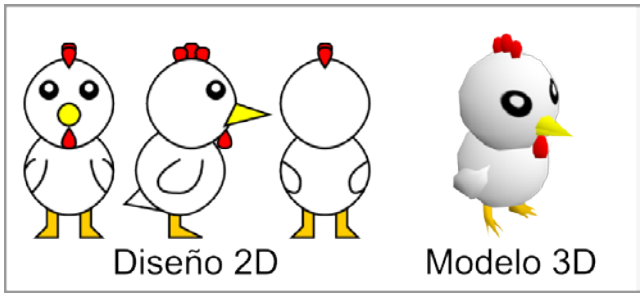


Figura 18. Diseño del polluelo

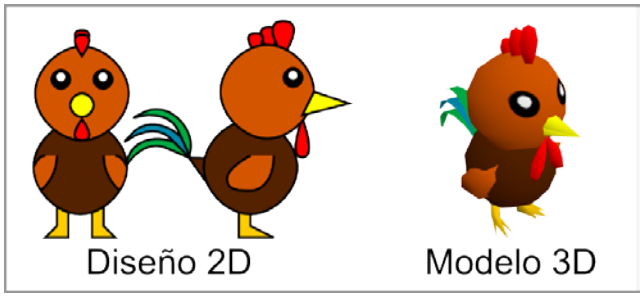


Figura 19. Diseño del gallo

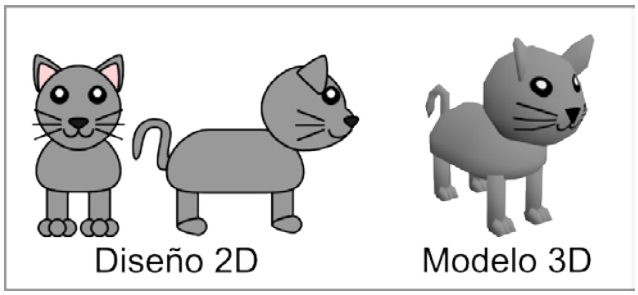


Figura 20. Diseño del gato

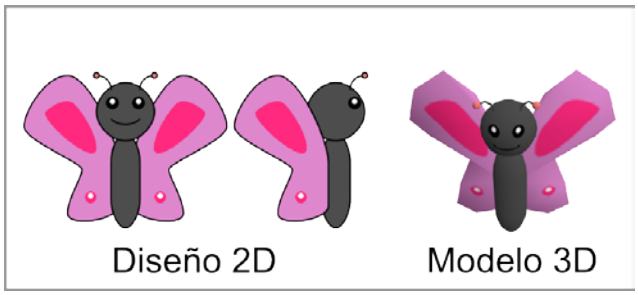


Figura 21. Diseño de la mariposa

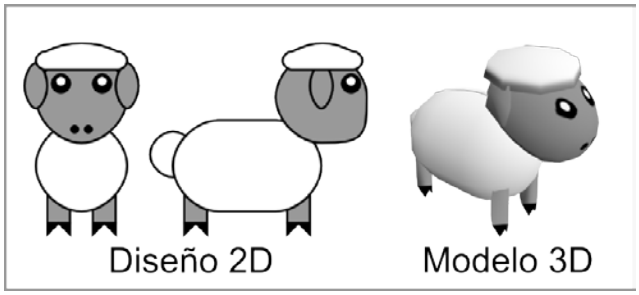


Figura 22. Diseño de la oveja

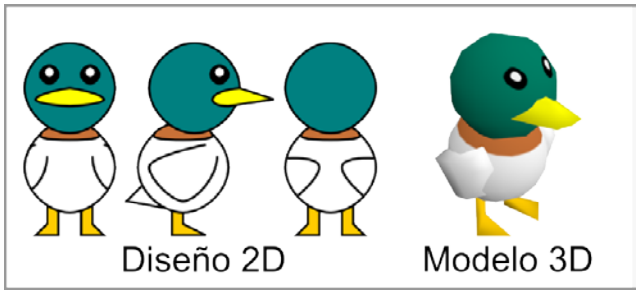


Figura 23. Diseño del pato

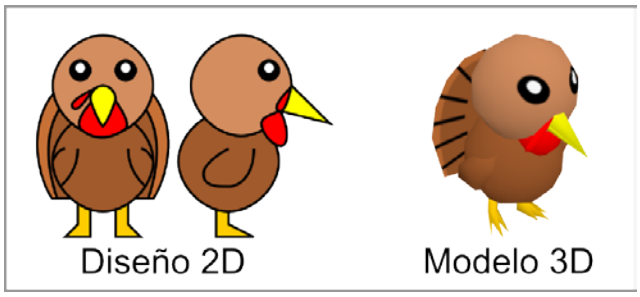


Figura 24. Diseño del pavo

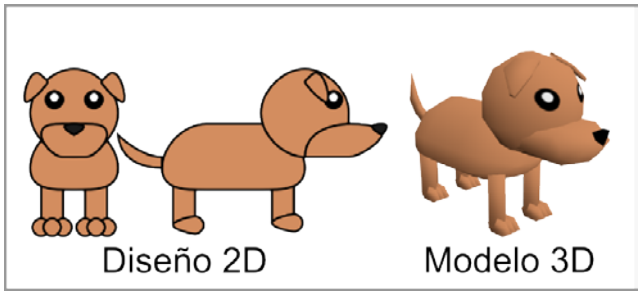


Figura 25. Diseño del perro

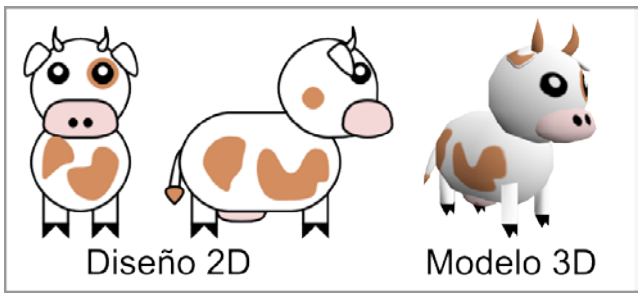


Figura 26. Diseño de la vaca

Diseño de los objetos del juego.

En esta sección se detalla el estilo gráfico de los objetos del juego, indicando su diseño en 2D y su acabado en un modelo tridimensional.

Dentro de los objetos que aparecerán en el juego están las frutas, cuyos diseños se muestran en la siguiente figura:

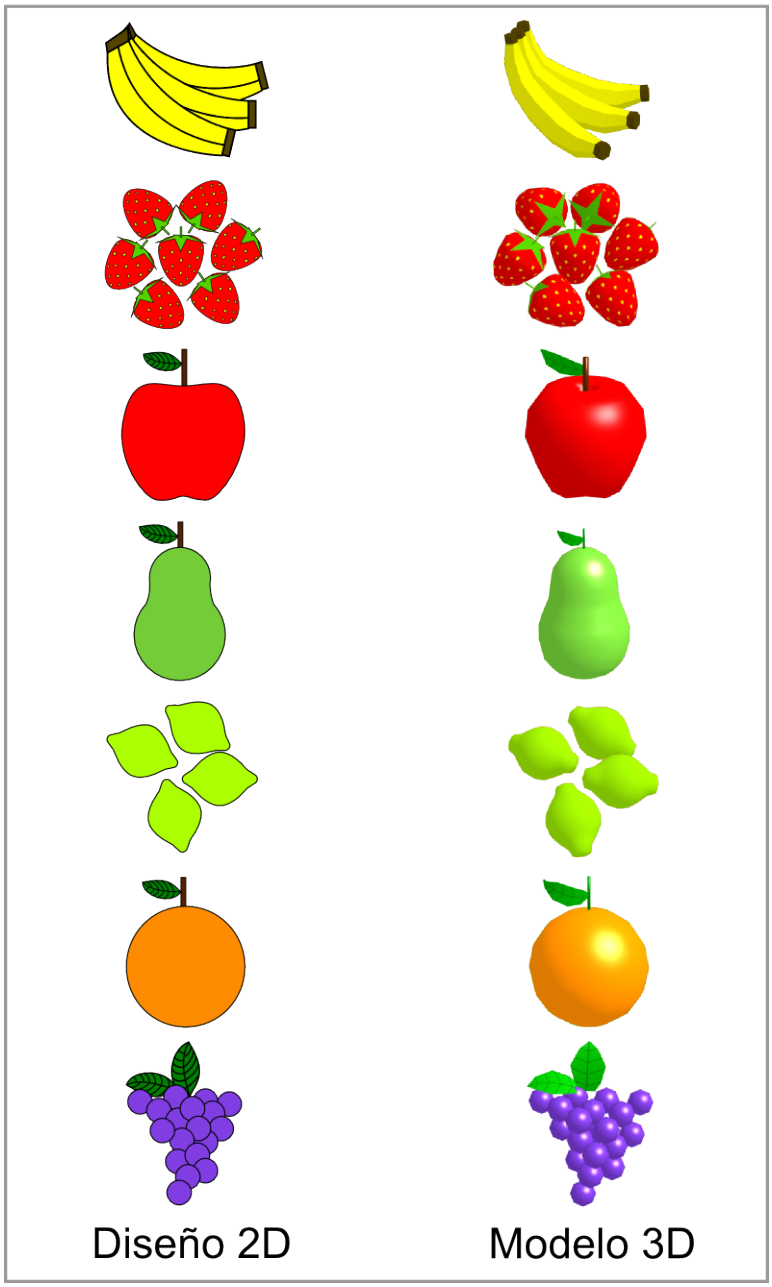


Figura 27. Diseños de las frutas

A continuación se muestran los diseños del resto de los objetos del juego. Cabe aclarar que la textura de la mochila es blanca para que se pueda colorear mediante el motor gráfico en lugar de crear varias texturas de diferentes colores, y así se pueda ahorrar espacio de almacenamiento.

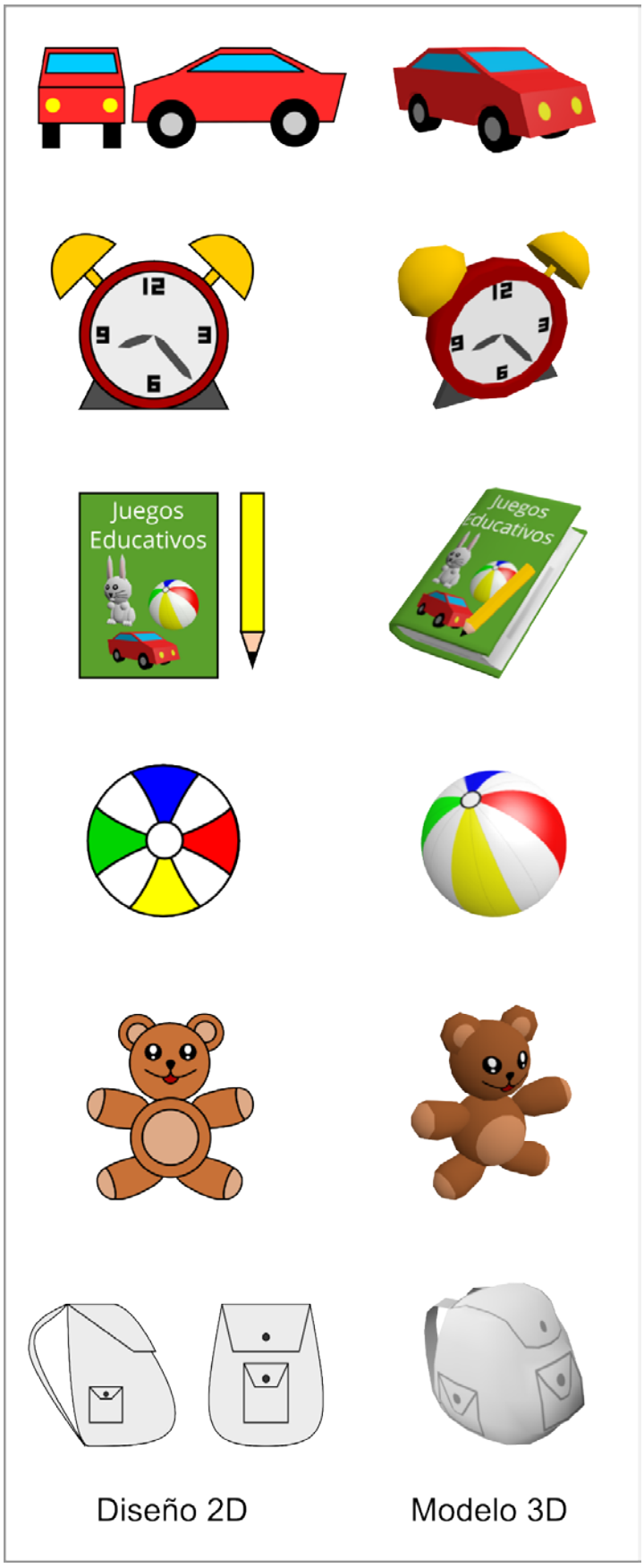


Figura 28. Diseños de los objetos del juego

Diseño de los escenarios del juego.

Para el mini juego de “encontrar las parejas de animales” el escenario es la matriz de cartas. Las cartas tienen el siguiente diseño como textura:

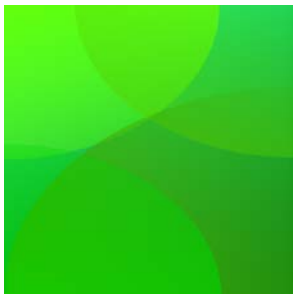


Figura 29. Diseño de la carta

Para el mini juego de “recolectar las frutas indicadas” el escenario lo conforman la mesa y la canasta, cuyos diseños de muestran a continuación:

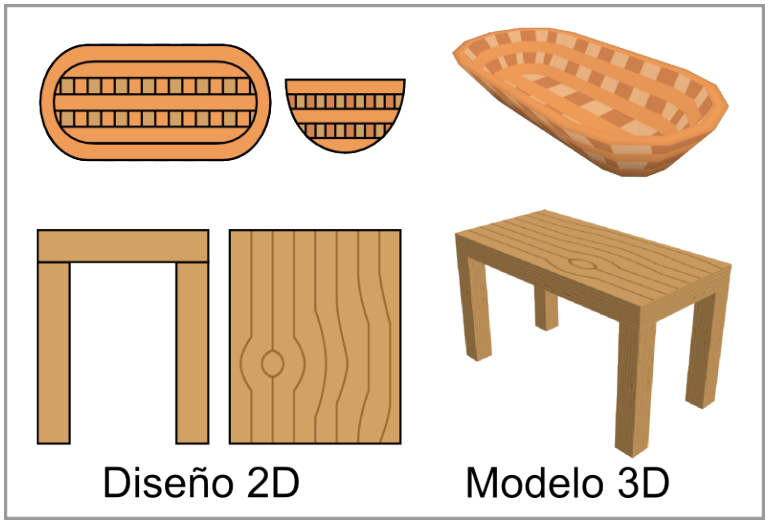


Figura 30. Diseños de la canasta y la mesa

Para el mini juego de “memorizar los animales y encontrarlos al abrirse el telón”, el escenario se compone de una tarima con cortinas rojas, las mismas que tienen animaciones para abrirse y cerrarse.

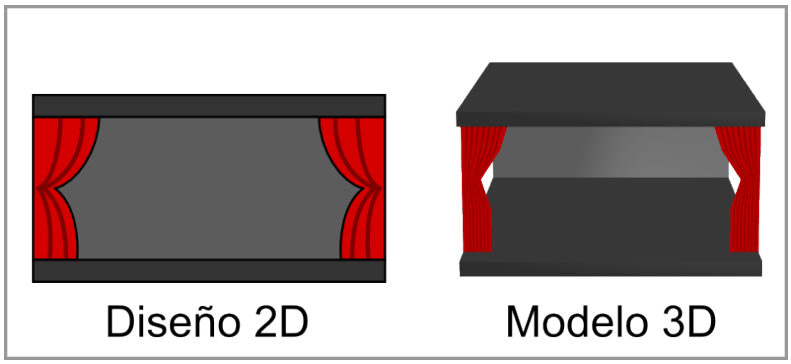


Figura 31. Diseño de la tarima con cortinas

Diseño dela interfaz gráfica del juego (GUI).

Estos íconos tienen un diseño minimalista con forma circular y están adornados con un borde blanco para resaltarlos.

Todos los íconos del juego tienen una silueta blanca simbolizando su propósito (por ejemplo: el engrane para la configuración), y un fondo con gradiente para dar la sensación de que el mismo tiene volumen.

En la siguiente figura se muestran todos los íconos que conforman la interfaz gráfica de los menús del juego:

Figura 32. Diseño de los íconos de la interfaz gráfica

Los íconos de cada mini juego son abstracciones simplificadas de los mimos. El propósito de utilizar estos íconos, en lugar de capturas de pantalla de cada mini juego, es el de reducir la carga visual del menú principal. A continuación se muestran los diseños de los mismos:

Figura 33. Diseño de los íconos de los mini juegos

A continuación se muestra el ícono de navegador del juego:

Figura 34. Diseño del ícono del juego educativo

El visto también conforma parte de la interfaz gráfica. Solo aparece en los mini juegos que necesiten indicar que el jugador realizó una acción correcta. En la siguiente figura se muestra su diseño:

Figura 35. Diseño del visto

4.4. Diseño del juego

Interfaz del juego

Se compone por la interfaz física y la interfaz visual. La interfaz física para el juego a desarrollar concierne al hardware necesario para que el jugador pueda interactuar. Como la plataforma objetivo del juego es HTML 5 y WebGL en navegadores de escritorio, los periféricos con los que se logrará dicha interacción son mouse y teclado.

La interfaz visual del juego la conformarán los menús del mismo y los elementos de los mini juegos con los que el jugador podrá realizar acciones para completarlos, tales como arrastrar un objeto o dar click sobre un personaje. Al pasar el mouse sobre un elemento con el que se pueda interactuar, ya sea un botón o una malla tridimensional, cambiará el cursor para intuir al jugador qué acción puede realizar.

Los diseños de los elementos que conforman la interfaz visual se detallan en la guía del estilo artístico anteriormente expuesta.

Mundo del juego

En esta sección se describen los personajes, objetos y animaciones que aparecerán en el juego a desarrollar.

Los animales son los únicos personajes del juego, las animaciones que contienen representan los movimientos característicos de los mismos acompañados por el sonido que emiten. Los animales que aparecerán en el juego son:abeja, burro, caballo, cerdo, conejo, polluelo, gallo, gato, mariposa, oveja, pato, pavo, perro y vaca.

Entre los objetos del juego se encuentran las frutas. Cuando el jugador da click sobre una, se muestra una animación en la que se eleva de la mesa, y dependiendo de la fruta, esta animación también tiene rotación. Al soltar la fruta, se muestra una animación en la que regresa a su posición original. Las frutas que aparecen en el juego son: banana, frutilla, limón, manzana, naranja, pera y uva.

Las cortinas que aparecen en el mini juego de “memorizar y encontrar los animales” tienen animaciones para abrirse y cerrarse, y

además conforman parte del escenario de dicho mini juego. Con estas animaciones se crearán cinemáticos (escenas con animaciones automáticas) antes de que aparezca el botón empezar, y al momento de iniciar el juego.

De igual forma, existen cinemáticos para el mini juego de “memorizar los objetos y mochilas”. Se mostrarán las mochilas abriéndose antes de que aparezca el botón empezar, y al iniciar se mostrarán los objetos guardándose en las mochilas a las que fueron asociados, para luego cambiar aleatoriamente su posición.

De entre los objetos del juego, los únicos que tienen animaciones son las cortinas y las mochilas. Los demás objetos del juego son estáticos y son: canasta, mesa, escenario, carro, despertador, libro y oso de peluche.

El modo de juego será el mismo que se describió previamente en el documento de propuesta del juego.

Habilidades de personajes:

Los personajes del juego no tienen habilidades debido a que no son personajes controlables. Las acciones que tienen se limitan a mostrar sus animaciones y reproducir sus sonidos cuando el jugador ha realizado una acción correcta en el mini juego en el que esté participando.

Motor de juego:

El juego a desarrollar, utilizará el motor gráfico que resultó escogido del estudio comparativo previamente realizado en el capítulo 3, el cual es Three.js r69.

El presente proyecto será desarrollado con dicho motor gráfico, y utilizará las etiquetas de audio de HTML 5 para reproducir los sonidos y la música de fondo.Para el registro de las mejores puntuaciones, se utilizará el almacenamiento del navegador mediante Local Storage.

Finalmente, para complementar las animaciones de los menús y botones que aparecerán en el juego, se utilizará la librería JQuery.

Cronograma

El cronograma se muestra en la sección 4.7 del presente capítulo: “Cronograma para el proceso de desarrollo”.

4.5. Diseño técnico

En esta sección se incluyen los diagramas generados de la metodología OOHDM, que contribuirán para la implementación del juego educativo.

Requerimientos funcionales del juego

La lógica de cada mini juego describe los requerimientos funcionales del mismo, y se especifican en la sección “modo de juego” dela“Propuesta del juego” anteriormente expuesto.

En el siguiente diagrama de casos de uso se muestran los requerimientosfuncionales que deben estar presentes en el juego educativo a desarrollar.

Figura 36. Diagrama de casos de uso del juego

Diseño conceptual

El lenguaje de programación para desarrollar los mini juegos es JavaScript. Este lenguaje utiliza prototipos para crear aplicaciones bajo el paradigma orientado a objetos, similar a lo que serían las clases en otros leguajes de programación.

Apoyado en la metodología OOHDM, se utilizará el diagrama de clases específico de cada mini juego para guiar la creación de dichos prototipos.

A continuación se muestran los diagramas de los mini juegos:

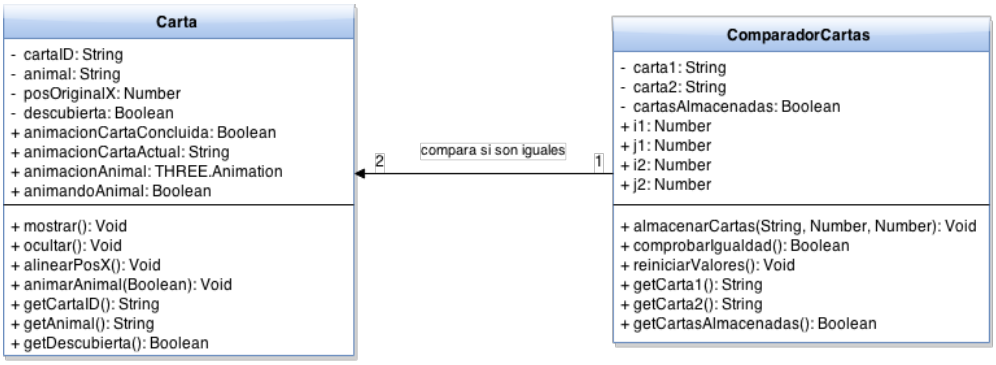


Figura 37. Diagrama clases del mini juego “Encontrar las parejas de animales”

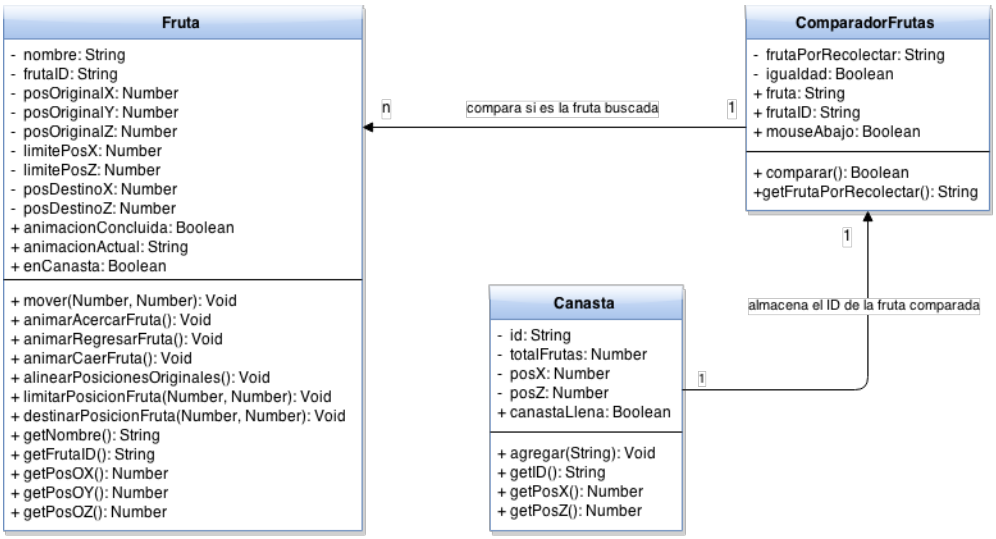


Figura 38. Diagrama clases del mini juego “Recolectar las frutas indicadas”

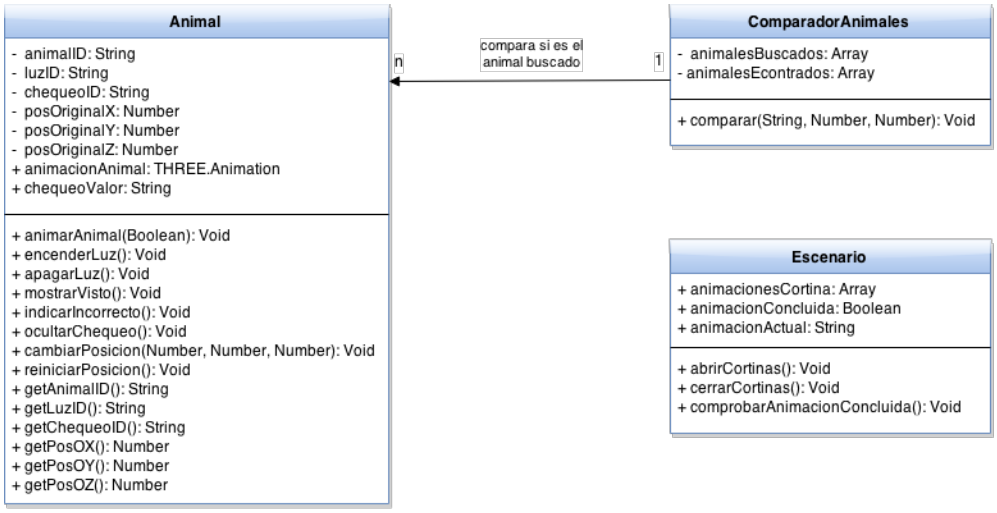


Figura 39. Diagrama clases del mini juego "Memorizar y encontrar los animales"

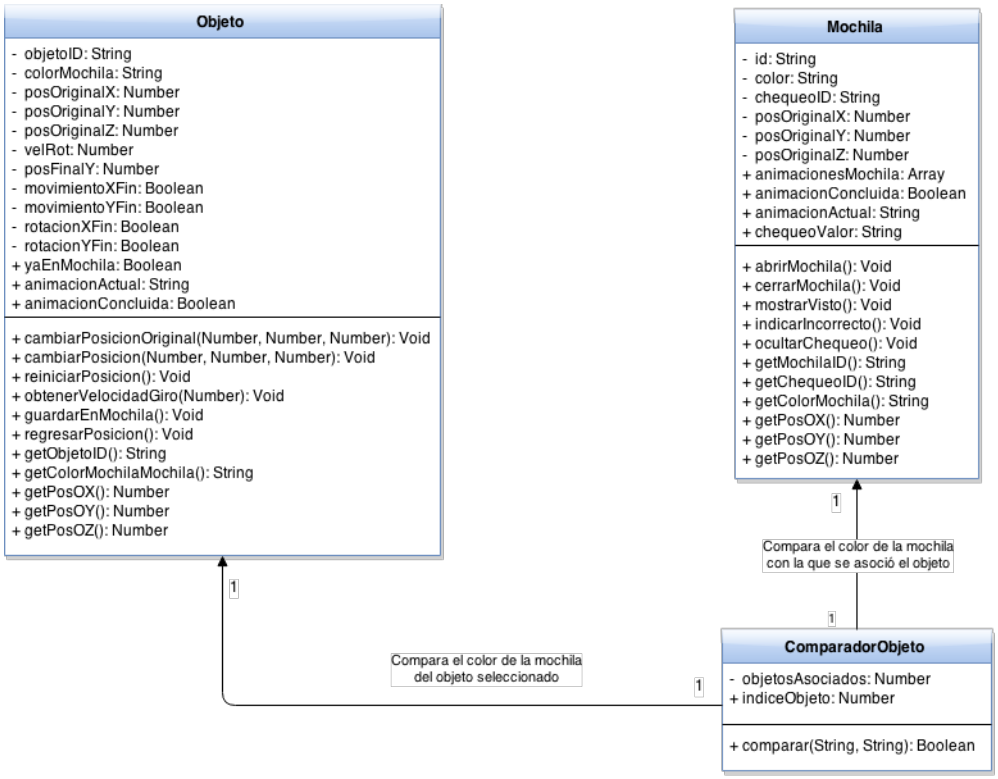


Figura 40. Diagrama clases del mini juego "Memorizar los objetos y mochilas

Diseño navegacional

Para optimizar el tiempo de descarga de los recursos multimedia, el juego estará diseñado para cargar dentro de un contenedor el mini juego escogido, de esta forma se cargarán la música y los sonidos en la página principal, y el mini juego hará llamadas a las etiquetas de audio cada vez que lo necesite. Los menús del juego también se incluirán en la página principal, para que cuando el jugador pause o finalice el mini juego se desplieguen sobre el mismo. En la siguiente figura se muestra el diagrama navegacional:

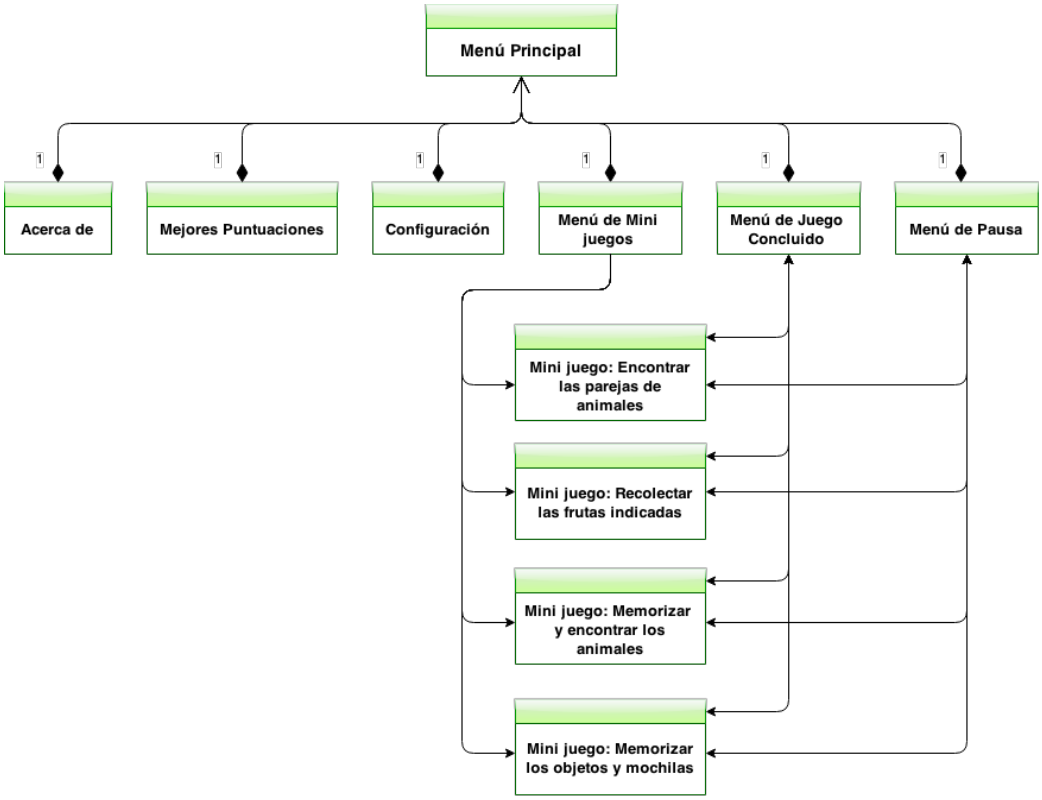


Figura 41. Diagrama navegacional del juego educativo

Diseño de interfaz abstracta

Está basado en el diseño navegacional, y su propósito también es el de orientar la implementación del juego con el fin de optimizar el tiempo de carga de los recursos multimedia. En la siguiente figura se muestran las interfaces que aparecerán en el juego y las interacciones entre sí.

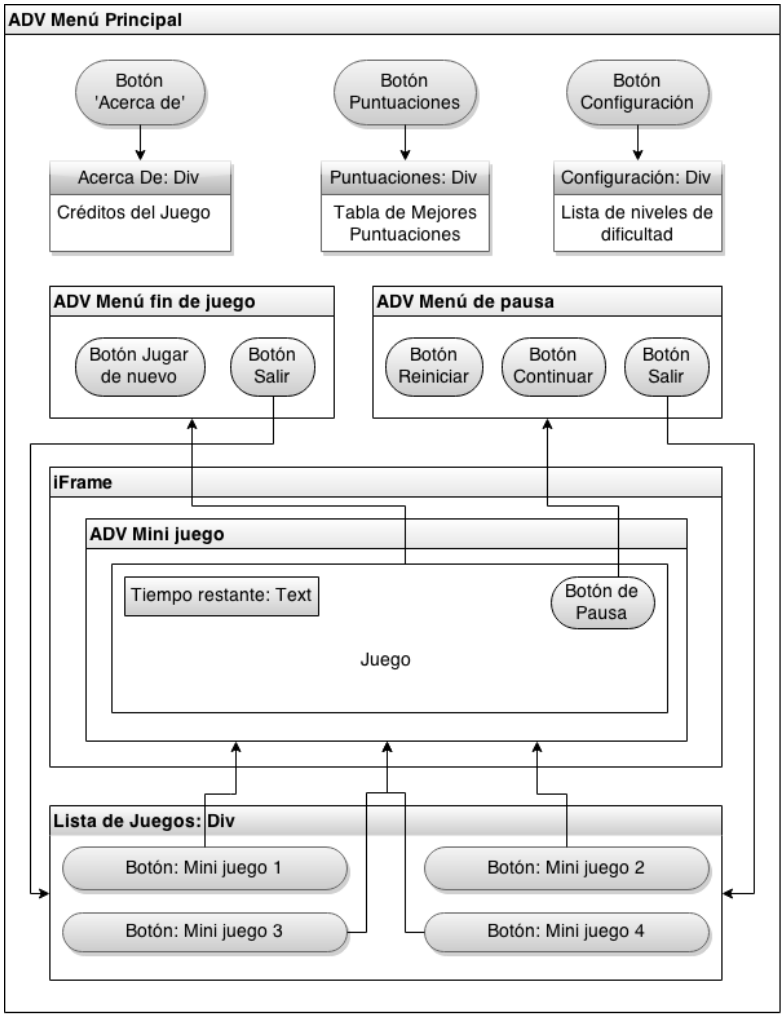


Figura 42. Diseño de interfaz abstracta del juego educativo

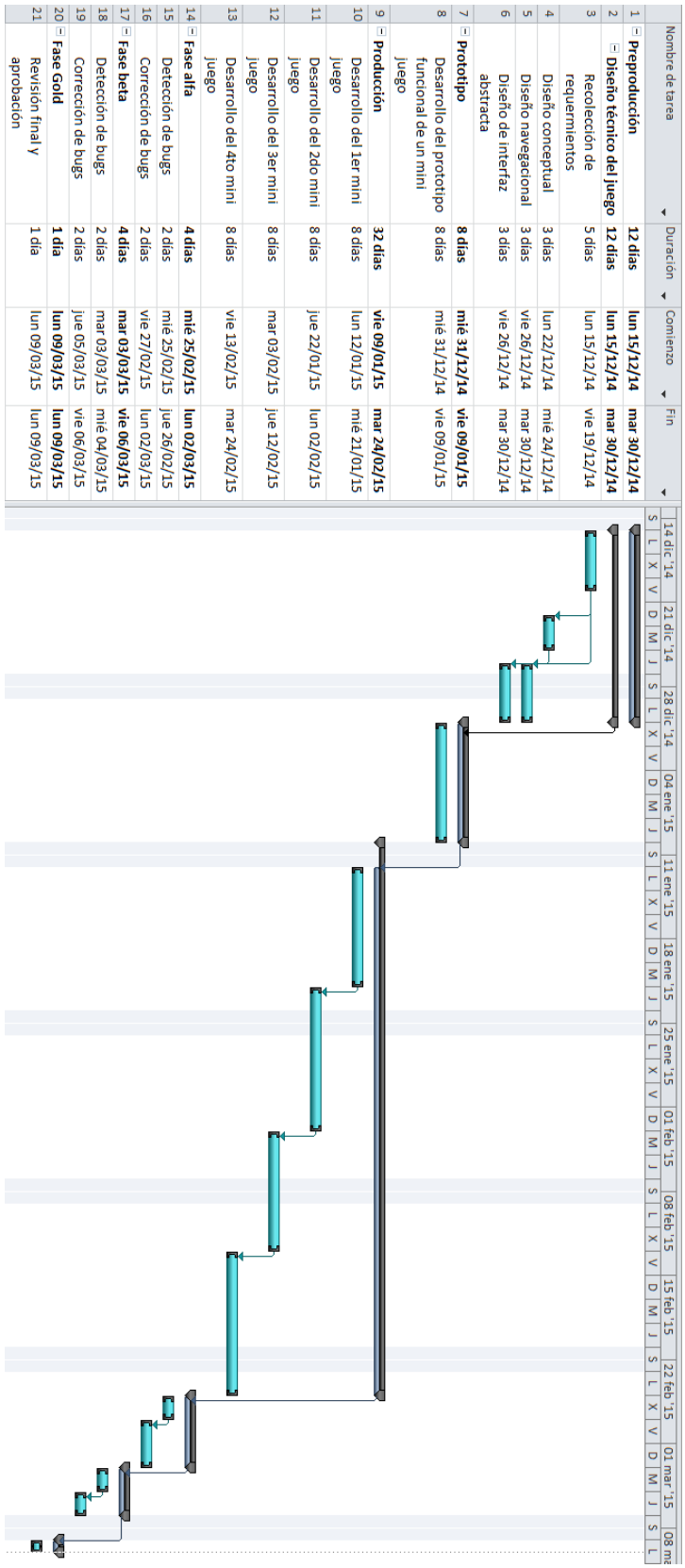
4.6. Prototipo del juego

El mini juego escogido para construir el prototipo es el de “encontrar las parejas de animales”, debido a que contiene a los personajes que aparecerán en el resto del juego y también el estilo artístico del mismo.

El prototipo se adjunta en la carpeta “Código Fuente” del presente proyecto.

4.7. Cronograma para el proceso de desarrollo

En la siguiente figura se muestra el cronograma que cubre las fases de producción del juego acorde a la guía propuesta por J. Novak.



	Nombre de tarea	Duración	Comienzo	Fin
1	**Preproducción**	**12 días**	**lun 15/12/14**	**mar 30/12/14**
2	**Diseño técnico del juego**	**12 días**	**lun 15/12/14**	**mar 30/12/14**
3	Recolección de requermientos	5 días	lun 15/12/14	vie 19/12/14
4	Diseño conceptual	3 días	lun 22/12/14	mié 24/12/14
5	Diseño navegacional	3 días	vie 26/12/14	mar 30/12/14
6	Diseño de interfaz abstracta	3 días	vie 26/12/14	mar 30/12/14
7	**Prototipo**	**8 días**	**mié 31/12/14**	**vie 09/01/15**
8	Desarrollo del prototipo funcional de un mini juego	8 días	mié 31/12/14	vie 09/01/15
9	**Producción**	**32 días**	**vie 09/01/15**	**mar 24/02/15**
10	Desarrollo del 1er mini juego	8 días	lun 12/01/15	mié 21/01/15
11	Desarrollo del 2do mini juego	8 días	jue 22/01/15	lun 02/02/15
12	Desarrollo del 3er mini juego	8 días	mar 03/02/15	jue 12/02/15
13	Desarrollo del 4to mini juego	8 días	vie 13/02/15	mar 24/02/15
14	**Fase alfa**	**4 días**	**mié 25/02/15**	**lun 02/03/15**
15	Detección de bugs	2 días	mié 25/02/15	jue 26/02/15
16	Corrección de bugs	2 días	vie 27/02/15	lun 02/03/15
17	**Fase beta**	**4 días**	**mar 03/03/15**	**vie 06/03/15**
18	Detección de bugs	2 días	mar 03/03/15	mié 04/03/15
19	Corrección de bugs	2 días	jue 05/03/15	vie 06/03/15
20	**Fase Gold**	**1 día**	**lun 09/03/15**	**lun 09/03/15**
21	Revisión final y aprobación	1 día	lun 09/03/15	lun 09/03/15

Figura 43. Cronograma para el proceso de desarrollo

4.8. Fase Alfa

La versión alfa del juego se adjunta en la carpeta “Código Fuente” del presente proyecto.El manual del mismo se adjunta en la carpeta “Manual del Juego”.

A continuación se listan los bugs detectadosen cada mini juego:

Encontrar las parejas de animales:

- Problema al centrar el botón empezar al momento de redimensionar.
- Se reinicia el contador después de ganar.

Recolectar las frutas indicadas:

- Se reproduce el sonido de victoria cuando finaliza la última ronda, incluso sinhaber ganado.
- Se crean más rondas que las que limita el nivel de dificultad.

Memorizar y encontrar los animales:

- Vuelve a aparecer el div 'Empezar' si se redimensiona poco después de haberseocultado.
- No se abren las cortinas si el número de animales que aparecen en escenario es menor que el total que hay.

Memorizar los objetos y mochilas:

- No muestra el tiempo al hacer click en 'empezar', después de haber redimensionado la pantalla.
- Problema al centrar el botón empezar al momento de redimensionar.
- Al cambiar la posición de los objetos en las dificultades fácil y normal,se descuadran de las posiciones de las mochilas.

A continuación se listan los bugs encontrados en el menú principal:

- La lista de puntuaciones al tener un solo elemento, muestra solo 3 resultados y no los completa con 0s.

- Al finalizar cualquier juego el mensaje de "se acabó el tiempo" aparece y desaparece constantemente.
- Al regresar al menú principal antes de haber finalizado un juego, el botón de pausa reaparece.
- Al reiniciar un juego el botón de pausa no desaparece.
- Al finalizar un juego y guardar la puntuación, no se registra la puntuación de otro juego.

4.9. Fase Beta

Se realizaron las correcciones pertinentes a los bugs anteriormente encontrados, y se completó la documentación del manual del juego con capturas de pantallas y explicaciones de la interfaz gráfica y reglas de cada mini juego. La versión beta del juego se adjunta en la carpeta “Código Fuente” del presente proyecto. El manual final del mismo se adjunta en la carpeta “Manual del Juego”.

4.10. Fase Gold

Versión final del juego.El mismo fue desarrollado acorde a la documentación de la guía propuesta por J. Novak, y su revisión está completa. El juego educativo se adjunta en la carpeta “Código Fuente” del presente proyecto

CAPÍTULO 5: CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

- Se cumplieron los objetivos planteados, y se desarrollóexitosamente el juego educativo con un diseño completo y detallado, utilizando la guía de desarrollo propuesta por Jeannie Novaken conjunto con la metodología OOHDM.
- Se construyó un modelo de evaluación para los motores gráficos basado en la norma de calidad ISO/IEC 25010, que permitió determinar el motor más adecuado para desarrollar el juego.
- El estudio comparativo determinó que el renderer de Babylon.js muestra más fotogramas por segundo, pero consume más recursos de memoria de video y de espacio de almacenamiento que Three.js, lo que puede producir un impacto en aplicaciones 3D que sean complejas.
- El estudio comparativo realizado, contribuye como referencia para destacar los principales aspectos que deben estar presentes un motor gráfico para plataformas web.

5.2. Recomendaciones

- Se recomienda la utilización de WebGL como plataforma tecnológica para desarrollar aplicaciones web interactivas que requieran gráficos 3D, ya que ayuda a enriquecer las interfaces de las mismas.
- Para mantener la portabilidad de las aplicaciones WebGL, se recomienda utilizar las APIs estándares de HTML 5 y no las implementadas de un navegador en particular.
- Considerando que los videojuegos son un tipo de software distinto a los sistemas informáticos, se recomienda utilizar la guía de desarrollo propuesta por Jeannie Novak para llevar a cabo la implementación.
- Se recomienda que en la asignatura de computación gráfica, se profundicen los conocimientos referentes a modelación 3D, animación y programación de videojuegos.

BIBLIOGRAFÍA

(2012). En A. Anyuru, *Professional WebGL Programming: Developing 3D Graphics for the Web* (págs. 2-3). John Wiley & Sons.

Babylon.js team. (06 de Junio de 2013). *Babylon.js*. Recuperado el 25 de Mayo de 2014, de http://babylonjs.com/

Blender Foundation. (14 de Agosto de 2011). *About*. Recuperado el 25 de Mayo de 2014, de http://www.blender.org/about/

Cabello, R. (24 de Abril de 2010). *Three.js*. Recuperado el 25 de Mayo de 2014, de http://threejs.org/

Cabello, R. (26 de 11 de 2012). Recuperado el 06 de Agosto de 2014, de Features | Three.js: https://github.com/mrdoob/three.js/wiki/Features

Catuhe, D. (06 de Junio de 2013). Recuperado el 29 de Mayo de 2014, de Babylon.js: a complete JavaScript framework for building 3D games with HTML 5 and WebGL: http://blogs.msdn.com/b/eternalcoding/archive/2013/06/27/babylon-js-a-complete-javascript-framework-for-building-3d-games-with-html-5-and-webgl.aspx

Chenard, J., Barreau, A., & Lepère, L. (22 de Julio de 2014). *Babylon.js v1.12 Index*. Recuperado el 06 de Agosto de 2014, de Features: http://www.sokrate.fr/documentation/babylonjs/index.html

Gregory, J. (2009). *Game Engine Architecture.*Florida, E.E.U.U.: CRC Press.

International Organization for Standardization. (3 de Enero de 2011). Recuperado el 30 de Abril de 2014, de ISO/IEC 25010:2011: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=35733

Kyaw, A. S., & Stein, J. (2011). *Irrlicht 1.7 Realtime 3D Engine Beginner's Guide.*Birmingham, Reino Unido: Packt Publishing Ltd.

Morrison, G. S. (2005). *Educación preescolar* (Novena ed.). Madrid: Pearson Educación S.A.

Mullen, T. (2011). *Introducing Character Animation with Blender* (Segunda ed.). Indianápolis, Indiana, EEUU: John Wiley & Sons.

Novak, J. (2012). *Game Development Essentials: An Introduction.* E.E.U.U.: Cengage Learning.

(2014). En T. Parisi, *Programming 3D Applications with HTML5 and WebGL: 3D Animation and Visualization for Web Pages* (págs. 364, 366). O'Reilly Media, Inc.

Rossi, G., Pastor, O., Schwabe, D., & Olsina, L. (2007). *Web Engineering: Modelling and Implementing Web Applications.* Londres: Springer Science & Business Media.

Silicon Graphics International Corp. (3 de Agosto de 2009). Recuperado el 21 de Abril de 2014, de http://www.sgi.com/products/software/opengl/

Sukin, I. (2013). *Game Development with Three.js.* Packt Publishing Ltd.

The GIMP Team. (02 de Mayo de 2012). *About GIMP*. Recuperado el 25 de Mayo de 2014, de http://www.gimp.org/about/

The Khronos Group. (3 de Marzo de 2011). *WebGL - OpenGL ES 2.0 for the Web*. Recuperado el 18 de Abril de 2014, de https://www.khronos.org/webgl/

The Khronos Group. (31 de 08 de 2013). Recuperado el 06 de Agosto de 2014, de Shader: http://www.opengl.org/wiki/Shader

EMPRESA VLBS CIA. LTDA.

Sangolquí, 05 de Junio del 2014

PARA: ING. MAURICIO CAMPAÑA, MsC.
DIRECTOR DE LA CARRERA DE ING. DE SISTEMAS
DE: ING. MARGARITA ZAMBRANO
GERENTE TÉCNICO DE VLBS CIA. LTDA.

ASUNTO: Carta de Auspicio

En mi calidad de Gerente Técnico de la Empresa VLBS, me permito informarle que nuestra empresa está auspiciando el Proyecto de Tesis titulado: "Estudio Comparativo de los Motores Gráficos WebGL 'Three.js' y 'Babylon.js' para el Desarrollo de un Juego Educativo", a ser desarrollado por el Sr. Estudiante: Andrés Enrique Vaca De Lucca. Por la atención que se digne dar a la presente le hacemos llegar nuestros más sinceros agradecimientos.

Atentamente,

Ing. Margarita Zambrano
Gerente Técnico de VLBS

EMPRESA VLBS CIA. LTDA.

EMPRESA VLBS CIA. LTDA.

Sangolquí, 20 de Abril del 2015

PARA: ING. MAURICIO CAMPAÑA, MsC.
DIRECTOR DE LA CARRERA DE ING. DE SISTEMAS
DE: ING. MARGARITA ZAMBRANO
GERENTE TÉCNICO DE VLBS CIA. LTDA.

ASUNTO: Carta de Aceptación

En mi calidad de Gerente Técnico de la Empresa VLBS, me permito informarle que nuestra empresa auspició el Proyecto de Tesis titulado "Estudio Comparativo de los Motores Gráficos WebGL 'Three.js' y 'Babylon.js' para el Desarrollo de un Juego Educativo", que fue desarrollado por el Sr. Egresado: Andrés Enrique Vaca De Lucca, ha sido concluida en su totalidad y cumple con lo establecido en el plan de Tesis. Por la atención que se digne dar a la presente le hago llegar mis más sinceros agradecimientos.

Atentamente,

Ing. Margarita Zambrano
Gerente Técnico de VLBS

CURRICULUM VITAE

DATOS PERSONALES

Nombre	Andrés Enrique Vaca De Lucca
Cédula de Identidad	1719467787
Fecha de Nacimiento	04 de mayo de 1990
Nacionalidad	Ecuatoriana
Lugar de Nacimiento	Guayaquil
Estado Civil	Soltero
Dirección	Sto. Domingo de Conocoto, Oe8 - 164
Teléfono	02 2 601 972
E-mail	andresvacadlucca@hotmail.com

FORMACIÓN ACADÉMICA

Universitarios	Universidad de las Fuerza Armadas “ESPE”. Carrera de Ingeniería en Sistemas e Informática.
Estudios Secundarios	Unidad Educativa Salesiana Cardenal Spellman Bachiller en Ciencias
Estudios Primarios	Liceo Naval de Quito “Comandante César Endara Peñaherrera”
Idioma Extranjero	Suficiencia en el Idioma Inglés Instituto de Lenguas ESPE

SEMINARIOS Y CURSOS

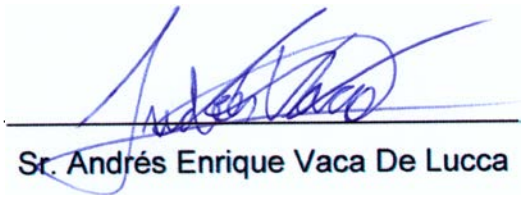
- ESPE- Departamento de ciencias de la energía y mecánica: 3DStudio Max con foto realismo, del 7 al 11 de septiembre del 2009
- Centro de Transferencia Tecnológica: Seminario introducción al software libre, efectuado el 9 de octubre del 2010
- Curso online: Introduction to Artificial Intelligence

EXPERIENCIA LABORAL

- Programador en SOFYA SISTEMS S.A., desempeñando actividades:
  - Análisis de requerimientos
  - Desarrollo de aplicaciones usando lenguajes PHP y Java, en conjunto con base de datos de MySQL

HOJA DE LEGALIZACIÓN DE FIRMAS

ELABORADO POR



Sangolquí, Mayo de 2015