UNIVERSIDAD TÉCNICA PRTICULAR DE LOJA

La Universidad Católica de Loja

ÁREA TÉCNICA

TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Diseño e implementación de la interfaz gráfica de usuario para el Gists-UTPL.

TRABAJO DE TITULACIÓN

AUTOR: Navas Castellanos, Andrés Roberto

DIRECTOR: Prisila Marisela, Valdiviezo Díaz, Ing

LOJA - ECUADOR

2015

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN

Ingeniera.

Prisila Marisela Valdiviezo Díaz.

DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de titulación: Diseño e implementación de la interfaz gráfica de usuario para el Gists-UTPL, realizado por Andrés Roberto Navas Castellanos ha sido orientado y revisado durante su ejecución, por se aprueba la presentación del mismo.

Loja, septiembre de 2015

f).....

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

"Yo Navas Castellanos Andrés Roberto declaro ser autor (a) del presente trabajo de titulación: Diseño e implementación de la interfaz gráfica de usuario para el Gists-UTPL, de la Titulación Sistemas Informáticos y de Computación, siendo Prisila Marisela Valdiviezo Díaz director (a) del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

Adicionalmente declaro conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado que se realicen a través, o con el apoyo financiero, académico o institucional (operativo) de la Universidad"

f).....

Autor: Andrés Roberto Navas Castellanos

Cédula: 050283676-0

DEDICATORIA

Dedico este proyecto de fin de carrera a mi madre, Martha mi hermano Felipe y mi padre Marco, quienes me han apoyado incondicionalmente en cada etapa de mi vida y ahora más que nunca con mucha paciencia y cariño.

A mis amigos y compañeros quienes estuvieron siempre dispuestos a apoyarme, soportarme e impulsarme a seguir adelante.

A mis maestros que nunca desistieron al enseñarme y guiarme.

AGRADECIMIENTOS

Quiero extender un agradecimiento especial al PhD. Dongfeng Liu quien me guio y enseñó muchísimo, a mis tutores los ingenieros, Prisila Valdiviezo, Rodrigo Barba y Guido Riofrio por la oportunidad que me brindaron de unirme al proyecto y todo el conocimiento y apoyo que me brindaron, a mi familia que me alentó incondicionalmente a seguir adelante y a mis amigos y compañeros que colaboraron con el presente proyecto.

CONTENIDO

ILUSTRACIONES

RESUMEN

El presente proyecto de titulación se centra en el diseño e implementación de una interfaz gráfica para un juego con contenido serio orientado a la enseñanza de la física a nivel de educación básica o secundaria llamado Gists – UTPL por sus siglas en inglés (3D Game-Based Intelligent Science Tutoring System in UTPL).

El documento inicia con la historia de los videojuegos con contenido serio, incluso antes de que se los conociera de esa forma, el diseño de interfaces para videojuegos, el uso básico de algunas herramientas necesarias para el trabajo, la forma de acoplar los diseños creados al sistema utilizando determinadas herramientas.

El presente proyecto servirá como base para el desarrollo de juegos con contenido serio en la UTPL y la posibilidad de un plan piloto para la implementación de los mismos en las aulas tanto de la universidad como en los colegios de la ciudad en dependencia del juego y de los contenidos que se traten en el mismo.

PALABRAS CLAVE: Gists-UTPL, juegos serios, videojuegos, enseñanza, inteligencia artificial, diseño, interfaz.

ABSTRACT

This project focuses on the design and implementation of a GUI for a game with serious content (Serious Game) oriented teaching school and high school-level physics called Gists - UTPL (3D Game-Based Intelligent Science Tutoring System in UTPL).

The document begins with the history of video games with serious content, even before they knew them that way, the design of interfaces for video games, the basic use of some tools needed for the work, the task of matching designs created into the system using certain tools.

This project will serve as a basis for developing serious games in the UTPL and the possibility of a pilot plan for implementing them in the classroom both the university and high schools of the city depending on the game and the content to be treated in it.

KEYWORDS: Gists-UTPL, serious games, video games, teaching, artificial intelligence, design, GUI.

1. PLANEAMIENTO

En este capítulo se explica el origen del presente del proyecto de fin de carrera, su vinculación con el proyecto Gists –UTPL, una explicación del mismo, el diseño de la solución, los objetivos del proyecto, una pequeña descripción del proyecto y la estructura del documento.

"El Proyecsto Prometeo es una iniciativa del gobierno ecuatoriano, que busca fortalecer la investigación, la docencia y la transferencia de conocimientos en temas especializados, a través de la vinculación de investigadores extranjeros y ecuatorianos residentes en el exterior. Está dirigido a universidades, escuelas politécnicas, institutos públicos de investigación y otras instituciones públicas o cofinanciadas que requieran asistencia en el desarrollo de proyectos de investigación en sectores prioritarios."(Secretaría de Educación Superior de Ciencia Tecnología e Innovación., 2015).

Este proyecto de fin de carrera tomó parte del mega proyecto llamado Gists – UTPL, por sus siglas (3D Game-Based Intelligent Science Tutoring System in UTPL). Es un sistema en tres dimensiones con algoritmos de inteligencia artificial para tutorías a los usuarios de forma automática, donde la aportación de investigación consistió en realizar la interfaz gráfica para el sistema Gists-UTPL.

1.1. Introducción

En la actualidad la importancia de incorporar técnicas y recursos académicos en las instituciones educativas que permitan a los jóvenes incrementar y fortalecer su aprendizaje es cada vez más necesario, mientras las nuevas tecnologías siguen aumentando y convirtiéndose en parte de nuestra vida cotidiana, esto hace necesaria la inclusión de las tecnologías de información de una forma razonable, paulatinamente la sociedad está perdiendo el deseo de obtener conocimientos por todas las comodidades que la tecnología le brinda, por lo que es necesario el uso útil de dichas tecnologías. Esto hace que, sobre todo los jóvenes, empleen su tiempo en otras actividades aparentemente más enriquecedoras como los videojuegos. En la charla en TED de Jane McGonigal titulada "Los juegos online pueden crear un mundo mejor", se expone un dato curioso. "Un Gamer[1] promedio habrá invertido 10000 horas jugando videojuegos en línea para cuando tenga 21 años de edad, mientras que 10080 horas es la cantidad de tiempo que un estudiante invierte desde quinto grado de escuela hasta su graduación del colegio si nunca falta a clase" (McGonigal, 2010). Con esto se demuestra fácilmente la magnitud del tiempo que los jóvenes gastan, si cabe el término, en esta actividad. Pero, qué se lograría si el tiempo de relajación se fusionara con el estudio, y más importante aún, si toda la energía y adrenalina que se aplica en los juegos se re direcciona al estudio. Tanto la cantidad como la calidad de profesionales que dispondría el mundo en el futuro no tendría precedentes, de ahí se originó la idea de los juegos con contenido serio, la fusión de

[1] Gamer: "Toda persona que juegue a un videojuego" (Wikipedia, 2014a)

un juego que brinde la adrenalina y el interés de los videojuegos comerciales, con un bagaje de conocimientos útiles para la vida cotidiana, que los jugadores aprendan mientras se divierten y de esa forma el estudio no sea una actividad tediosa sino enriquecedora en todo sentido.

1.2. Definición del problema

Como parte del proyecto Prometeo, el grupo de Inteligencia Artificial de la UTPL realizó un plan de investigación para desarrollar un sistema para tutoría inteligente basado en un juego con contenido serio en tres dimensiones implementado en una red de área local (LAN) llamado “Gists-UTPL” por sus siglas en inglés (3D Game-Based Intelligent Science Tutoring System in UTPL).

Debido a la necesidad que existe en los estudiantes por reforzar los conocimientos que adquieren en el aula, es que el sistema debe ser atractivo para los usuarios de forma tal que sientan deseos de jugar y no se convierta en una obligación. Además el juego presenta una aplicación práctica y real de los temas estudiados, lo que hace que el alumno se dé cuenta de la importancia de lo que está aprendiendo, para lo cual el presente proyecto de titulación se centra en el diseño y la implementación de una interfaz gráfica para los alumnos que sea intuitiva y agradable.

Una de las facetas del proyecto es el diseño e implementación de una interfaz gráfica de usuario o GUI por sus siglas en inglés (Graphical User Interface), que sea dinámica e interactiva con el usuario utilizando un motor gráfico que soporte interfaces en tres dimensiones, dicha interfaz debe permitir al usuario interactuar dentro del juego, además debe presentar todas las opciones disponibles de forma intuitiva y lúdica para que envolver al usuario en el juego.

1.3. Diseño de la solución

Definiremos brevemente a la interfaz del usuario como todo lo que el usuario puede ver en el juego y con lo que puede interactuar, se dará una explicación más concreta en los capítulos posteriores. Dicho esto la interfaz involucra todos los menús del juego, las animaciones, los efectos visuales y la interacción con el juego en sí, por lo que se diseñará una interfaz completa usando como referencia juegos similares y conceptos de diseño de interfaces con el fin de que el juego sea agradable para el usuario y se divierta usándolo.

1.4. Objetivo general

Diseñar una interfaz gráfica amigable con los usuarios, que sea coherente con la intención y la modalidad del juego Gists – UTPL, basándose en videojuegos comerciales de éxito para hacer al sistema más atractivo a los jugadores.

1.5. Objetivos específicos

- ✓ Diseñar una interfaz original y agradable para los usuarios del juego.
- ✓ Implementar todas las funcionalidades del sistema en la nueva interfaz.
- ✓ Diseñar una transición agradable entre las interfaces de dos y tres dimensiones.
- ✓ Conseguir una navegación natural por los menús y opciones del juego.

1.6. Alcance

El presente proyecto de fin de carrera se limita al diseño e implementación de la interfaz gráfica de usuario para el sistema Gists – UTPL (3D Game-Based Intelligent Science Tutoring System in UTPL) y todo que esto involucre; entiéndase por: menús, pantallas de juego, íconos, pantallas de inicio, y cualquier interacción visual entre el usuario y la aplicación. No incluye las funcionalidades del juego, la lógica de programación o los algoritmos con los que el juego funciona, ya que el presente proyecto de titulación forma parte de un proyecto de investigación (proyecto Prometeo) que se está desarrollando en el Departamento de Ciencias de la Computación y Electrónica, donde se contempla estos componentes. Dependiendo del tiempo que tome el desarrollo, el proyecto se podría extender a la implementación de animaciones de transición y archivos de audio para mejorar la experiencia del usuario, pero queda a criterio del equipo de trabajo del sistema Gists – UTPL y los involucrados en el presente proyecto.

1.7. Descripción del proyecto

El presente proyecto de titulación trata sobre el diseño y la implementación de la interfaz gráfica de usuario para el sistema “Gists – UTPL”.

Gists – UTPL (3D Game-Based Intelligent Science Tutoring System in UTPL), es un sistema de soporte de educación basada en juegos con contenido serio, en otras palabras es un videojuego que aporte conocimiento sobre un tema en particular al jugador de una forma lúdica para que sea interesante para el mismo.

Gists – UTPL se centra en la materia de física en un nivel medio orientado a estudiantes de colegio. El sistema contiene una serie de mini-juegos orientados a diferentes ámbitos de la materia de física, divididos por temas generales. El objetivo es que el jugador practique los conocimientos adquiridos en una forma divertida con el fin de que lo haga más a menudo; por ejemplo para el estudio de proyectiles y movimiento parabólico el jugador controla un tanque de guerra y el profesor le asigna un blanco para disparar, el estudiante debe calcular la fuerza con la que se disparara la bala analizando la distancia a la que se encuentra el blanco y otras variables dependiendo el nivel que se esté evaluando.

En el presente proyecto se diseñará la interfaz para el juego incluyendo los menús, íconos y todo lo que el usuario pueda ver e interactuar durante el tiempo de juego. Posterior a esto se realizará la codificación de dicha interfaz y se la vinculará con el sistema Gists – UTPL.

1.8. Estructura del documento

Capítulo 1 – Planeamiento: en este capítulo se explica el origen del presente proyecto de fin de carrera, su vinculación con el proyecto Prometo, una explicación del mismo, el diseño de la solución, los objetivos del proyecto, una pequeña descripción del proyecto y la estructura del documento.

Capítulo 2 – Estado del arte: se presenta el estado actual de las tecnologías utilizadas para la realización de interfaces gráficas de usuario, así como los ámbitos en que están siendo utilizados los videojuegos con contenido serio y una pequeña reseña histórica sobre los videojuegos.

Capítulo 3 – Desarrollo: se explicará todo el trabajo realizado durante el proyecto dividiendo en fases para mejor entendimiento del lector.

Capítulo 4 – Capacitación sobre desarrollo de Juegos con contenido serio: se detallan los contenidos impartidos a un curso de 20 estudiantes de la carrera sobre los temas tratados en el proyecto.

Conclusiones y Recomendaciones: en este capítulo se mostrarán los resultados obtenidos del presente proyecto de titulación, las conclusiones y recomendaciones que se originaron del mismo, y se detallan las posibles continuaciones del presente proyecto, aplicando lo aprendido y reestructurándolo (de ser necesario) en un sistema similar.

Bibliografía.

Anexos.

2. ESTADO DEL ARTE

En este capítulo se presenta una pequeña reseña histórica sobre los videojuegos, el estado actual de las tecnologías utilizadas para la realización de interfaces gráficas de usuario, así como los ámbitos en que están siendo utilizados los videojuegos con contenido serio.

2.1. Reseña histórica sobre los videojuegos

Si bien es cierto, la idea de adoptar los videojuegos como forma de enseñanza en un aula de clase es bastante novedosa, los videojuegos con un contenido útil no lo son. En 1985 se lanzó al mercado ¿Dónde está Carmen Sandiego en el mundo?[2], para el Apple II, y más tarde fue portado al resto de sistemas. El objetivo del juego es perseguir a Carmen y a sus secuaces por todo el mundo y arrestarlos, para lograrlo se necesita una orden de arresto dirigida al criminal que haya cometido el delito que el jugador está resolviendo. Cada nivel comienza con un crimen distinto y el jugador se dirige a la ciudad donde ocurrió para obtener pistas que le ayuden a descubrir la identidad del criminal y la siguiente ciudad a la que este irá para seguirlo. El jugador tiene un tiempo límite para arrestar al criminal y antes de que se cumpla debe haber conseguido suficientes pistas de su identidad para solicitar la orden de arresto en su contra, cuando encuentra al villano debe arrestarlo y el juego termina cuando toda la banda haya sido arrestada, incluida Carmen Sandiego.

Este juego enseña geografía y algo de historia sobre algunas de las ciudades más importantes del mundo, ya que las pistas que se obtienen son datos como la moneda o los colores de la bandera del país o alguna reseña en particular sobre el lugar al que se dirigió.

El juego tuvo tanto éxito que se hicieron algunas secuelas del mismo como ¿dónde está Carmen Sandiego en EEUU?, en Europa e incluso a través del tiempo y en el espacio; incrementando en cada entrega el conocimiento que el jugador adquiría incluyendo matemáticas y hasta algo de astronomía.

En 1994 Sierra Entertaiment presentó “The lost mind of Dr. Brain[3]”, el juego comienza con un experimento del Dr. Thaddeus “Puzzles” Brain, maestro de los rompecabezas, que intenta compartir su conocimiento con un ratón de laboratorio pero no funciona como él esperaba y su cerebro queda en el cuerpo del ratón mientras su cuerpo queda en algo parecido a un estado vegetal, el objetivo del juego es reparar todo el daño en el cerebro del doctor para que vuelva a su cuerpo. En cada parte del cerebro hay un tipo diferente de juego con tres niveles de dificultad, y aunque cada mini juego es distinto, la mayoría se basa en lógica.

[2] ¿Dónde está Carmen Sandiego en el mundo?: http://www.carmensandiego.com/hmh/site/carmen/
[3] The lost mind of Dr. Brain: http://www.sierragamers.com/aspx/m/649091

Tanto en ¿Dónde está Carmen Sandiego? como en “The lost mind of Dr. Brain”, podemos darnos cuenta que la historia en sí no es compleja, en el primero hay que recopilar pistas para capturar a un criminal y en el segundo hay que resolver acertijos para que el doctor recupere su cuerpo y en ninguno existe una secuencia que nos impida entender la historia en su totalidad si comenzamos el juego a la mitad. Dicho esto se puede alegar que en ese tiempo y para este tipo de juegos lo primordial no era la historia sino la jugabilidad simple que enganche al jugador; pero los juegos fueron evolucionando junto con las computadoras dando paso a una gama de videojuegos más robustos tanto en historia como en forma de juego. Con los juegos de disparo en primera persona como “Call of Duty[4]” o “Halo[5]”, de estrategia como “Age of Empires[6]” o “Total War[7]” y juegos en tercera persona como “Hitman[8]” o “Assassin’s Creed[9]”, podemos apreciar la evolución que ha tenido esta industria, más aun si los comparamos con los primeros videojuegos del mercado que podían ser controlados en su totalidad con cuatro o cinco teclas, sin contar con las historias tan trabajadas de hoy en día que continúan en las secuelas y son una razón más para atraer al jugador, ya que van creando falencias en la historia que el jugador debe rellenar mediante vaya avanzando en el juego, y esta necesidad por entender lo que sucede en el mundo virtual en el que está inmerso hace que le entregue más tiempo al juego.

Pese a todo esto, con la aparición de los dispositivos móviles los juegos simples volvieron a estar en auge, ahora en los mercados de aplicaciones como Google Play o iStore. Los apps como “Angry Birds[10]”, “Plantas vs Zombies[11]” o “Candy Crush[12]” son jugados diariamente por millones de personas, ya sea desde sus dispositivos móviles o desde un computador mediante alguna red social.

Aun así los videojuegos complejos no se han quedado atrás ya que aprovecharon el internet para abrir todo un nuevo mundo de posibilidades con la jugabilidad en línea. “World of Warcraft[13]”, “Second life[14]” o “Minecraft[15]” ocupan puestos bastante altos en las listas de los sitios con más visitas en la web; La segunda wiki más grande del mundo, después de Wikipedia, es la wiki de World of Warcraft, con cerca de 80 000 artículos y más de 5 millones de visitas mensuales. (McGonigal, 2010), así que los videojuegos

[4] Call of Duty: http://www.callofduty.com/
[5] Halo: https://www.halowaypoint.com/
[6] Age of Empires: http://www.ageofempires3.com/
[7] Total War: http://www.totalwar.com/en_us/
[8] Hitman: http://hitman.com/
[9] Assassins Creed: http://assassinscreed.ubi.com/es-es/home/
[10] Angry Birds: https://www.angrybirds.com/
[11] Plantas vs Zombies: http://www.popcap.com/plants-vs-zombies-1
[12] Candy Crush: https://play.google.com/store/apps/details?id=com.king.candycrushsaga&hl=es_419
https://itunes.apple.com/us/app/candy-crush-saga/id553834731?mt=8
[13] World of Warcraft: http://eu.battle.net/wow/es/
[14] Second Life: http://secondlife.com/
[15] Minecraft: https://minecraft.net/

están en una su apogeo, tanto los juegos complejos que requieren un computador, como las simples aplicaciones que se descargan en un celular o se acceden mediante una red social.

2.2. Juegos serios

El instituto de Juegos Serios SGI define a los mismos como: Aplicaciones desarrolladas utilizando tecnologías de juegos de video que sirvan a otros propósitos además del puro entretenimiento"(Arnaba et al., 2013).

Los juegos serios pueden ser aplicados en muchos campos y brindan un sinnúmero de posibilidades para la sociedad, algunos autores han reconocido en sus libros el valor de éstos en la educación; a continuación se citan textualmente sus opiniones:

"Se ha reconocido que los juegos pueden ser un poderoso vehículo para el aprendizaje, y la inteligencia artificial puede ampliar los resultados de aprendizaje con juegos"(Looi, McCalla, Bredeweg, & Breuker, 2005, p. 306).

En (Abt, 1987, p. 17) se menciona que "(…) planificando, jugando y analizando los juegos educativos motivan a los alumnos a estudiar los problemas dramatizados en el juego y la literatura que envuelve los problemas para sintetizar soluciones del problema y evaluar críticamente posibles soluciones que desarrollaron en su investigación"

"En los últimos años, los juegos serios se han convertido en un tema en apogeo en conferencias internacionales, convenciones y simposios. El interés en usar juegos en educación, motivación y cambio de comportamiento ha crecido tremendamente en un corto periodo de tiempo por verdaderos profesionales internacionales, líderes cívicos, defensores de la salud y los derechos humanos, educadores, jugadores e investigadores"(Ritterfeld, Cody, & Vorderer, 2009, p. 3).

2.3. Trabajos relacionados

Un gran ejemplo de juegos con contenido serio son los simuladores, que dan al usuario una experiencia de "juego" muy cercana a la realidad.

En noviembre de 1982 Microsoft presentó al público Flight Simulator 1.0, un simulador de vuelo con variables meteorológicas y hora del día, que fue toda una novedad en su época ya que el jugador podía sentirse como todo un piloto al entender los controles de un avión.

Hasta la fecha Microsoft ha presentado 13 versiones de su simulador de vuelo, siendo la última "Microsoft Flight[16]" presentada al público el 29 de febrero de 2012.

Ahora este pequeño simulador cuenta con un sinnúmero de aviones entre los cuales escoger incluyendo modelos comerciales como el Boeing 747-400, aviones históricos como el de los hermanos Wright y algunos modelos futuristas. Posee un control de tráfico aéreo gracias a la implementación de inteligencia artificial, los usuarios pueden descargar el clima en tiempo real de estaciones meteorológicas lo que le permite al simulador estar sincronizado con el mundo real, sin mencionar las excelentes gráficas que posee brindando al jugador una experiencia única y casi real.

El tutorial de juego enseña al usuario poco a poco la funcionalidad de cada indicador, perilla y botón del tablero de todos los aviones disponibles lo que genera un aprendizaje real. Las posibilidades que brinda un simulador son muchas y de diferente tipo, un niño que quiere ser piloto puede empezar su aprendizaje desde su hogar y desde una edad muy temprana, la calidad de simulación es tan buena que se descubrió que Osama Bin Laden y su gente usaron el simulador de Microsoft de la época para planificar el atentado contra las torres gemelas.

Otro juego que simula la realidad es TRADOC[17], (Training and Doctrine Command), creado por el ejército de los Estados Unidos de América para "asegurar que los soldados y líderes estén familiarizados con los valores, las normas y las culturas del lugar donde serán residentes"(Maureen, 2011). Actualmente está centrado en Iraq y Afganistán, y el juego presenta una serie de niveles donde el jugador debe aprender el idioma, y aplicarlo en posibles misiones respetando las costumbres locales.

"Este juego ganó el primer lugar en las categoría gubernamental en el "Serious Games Challenge[18]" y se desarrolló en asociación con la Universidad de Texas en Dallas, con su programa de arte y tecnología."(Maureen, 2011).

Los soldados cuentan con un avatar en 3D con el que se mueven por una aldea virtual donde deben desenvolverse solos, esto ayuda a que entiendan las consecuencias de sus acciones, de su postura, lenguaje corporal y su temperamento y los prepare de una mejor manera para evitar conflictos cuando tengan que vivirlo en carne propia.

[16] Microsoft Flight: http://www.microsoft.com/games/flight/
[17] TRADOC: http://www.tradocnews.org/
[18] Serious Games Challenge: http://sgschallenge.com/wordpress/

2.4. Plataformas utilizadas en el desarrollo de juegos con contenido serio

2.4.1. Unity[19]

Unity es una famosa empresa de videojuegos comerciales que también desarrolla juegos con contenido serio y sus productos tienen gran éxito en el mercado por la alta calidad que permite crear su IDE.

"Unity es un ecosistema de desarrollo de juegos: un poderoso motor de renderizado totalmente integrado con un conjunto completo de herramientas intuitivas y flujos de trabajo rápido para crear contenido 3D interactivo; publicación multiplataforma sencilla; miles de activos de calidad, listos para usar en la Tienda de Activos y una Comunidad donde se intercambian conocimientos." (Unity Technologies, 2015a)

Unity[20] tiene un módulo especializado para juegos con contenido serio que ofrece herramientas para facilitar el trabajo del equipo de desarrollo como un entorno fácil y rápido para la construcción de escenarios, opciones para producción de audio y video en alta resolución, varios puntos de iluminación efectos especiales para post producción, filtros de audio y transformación de renders con textura para brindar mayor realismo y calidad al producto final.

También cuenta con un asistente para transformar al juego de cliente único a juego en red, ayudando en la creación del mundo virtual y las configuraciones necesarias para la red.

Algunos de los juegos creados con esta plataforma son:

- ✓ 3D Plannign system for Olympic Games: es un modelo interactivo de los juegos olímpicos del invierno de 2014 diseñado para que los participantes conozcan las instalaciones y puedan familiarizarse con ellas y planificar su organización en el evento.
- ✓ US Navy virtual training: permite a sus usuarios practicar el ensamblaje y mantenimiento de equipos militares para ahorrar tiempo y dinero en los entrenamientos tradicionales.
- ✓ 3D Bike configurator: es un app para móviles y navegadores diseñada para ciclistas que deseen realizar mejoras a sus bicicletas, esto facilita la decisión ya que muestra de una forma gráfica el resultado final.
- ✓ Dinasaurs unearthed augmented reality experience: permite ver e interactuar con dinosaurios y brinda conocimiento anatómico e histórico sobre cada uno lo que ayuda a entender al espécimen.

[19] Unity: http://unity3d.com/es
[20] Unity: http://unity3d.com/industries/sim

Como estos, existen muchos otros juegos que guardan un contenido útil para un público específico, son simples pero muy atractivos visualmente hablando además de la interactividad que generan con el usuario; esto los convierte en éxitos en el mercado. Esta información fue obtenida de la página oficial de Unity. (Unity Technologies, 2015b).

2.4.2. Delta3D[21]

Otra herramienta importante en este ámbito es Delta3D. “Es un motor de simulación y juego de código abierto adecuado para la educación, capacitación, visualización y entretenimiento. Ofrece características características adaptadas específicamente a las comunidades de modelado y simulación y comunidades DoD como la Arquitectura de Alto Nivel (HLA), después de la Acción (AAR), el apoyo de terreno a gran escala, y el Sistema de Gestión de Aprendizaje SCORM (LMS) de integración.”(Delta3D, 2004)

A continuación se presentan algunos proyectos desarrollados en Delta3D:

- ✓ Vessel Damage Control Trainer: presenta un ambiente en 3 dimensiones que recrea el interior de un barco de la marina y que presenta daños, el jugador debe reparar el barco con herramientas que encontraría en el mundo real lo que ayuda a su entrenamiento de una forma interactiva y de bajo costo.
- ✓ AVI Develops Tactical Target Analysis and Prediction System: recrea un escenario que requiera una intervención militar de forma tridimensional y facilita al jugador la planificación de una estrategia adecuada según la situación.

Esta información fue recuperada de la página lista de proyectos oficial de Delta3D. (Delta3D, 2009).

[21] Delta3D: http://delta3d.org/

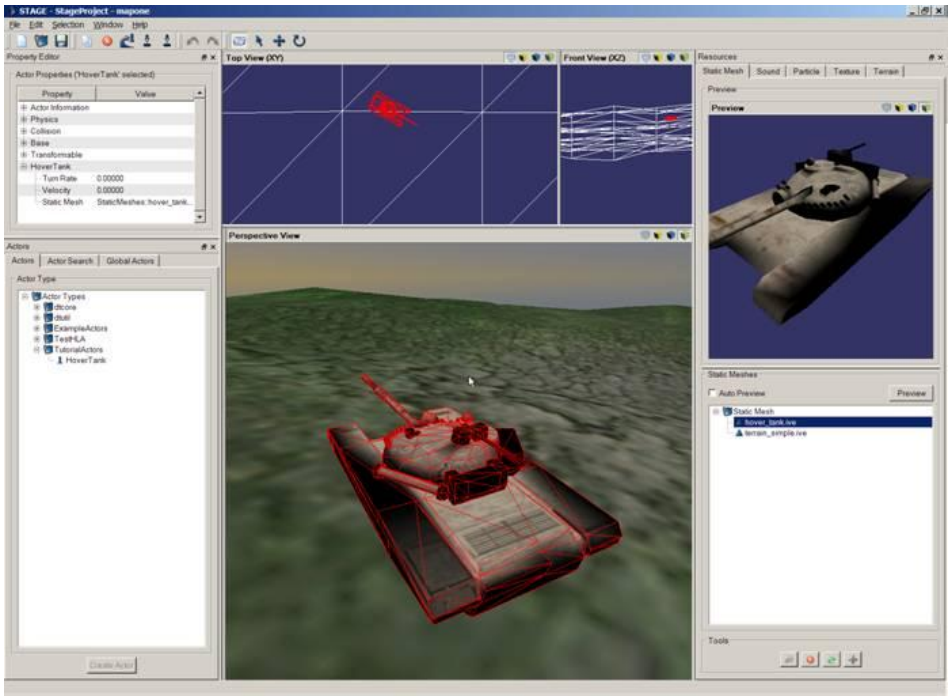


Imagen 1, Interfaz de Delta3D, recuperada de (Coonan, 2006)

Delta3D tiene algunas librerías [22]para trabajar, cada una con un propósito específico. A continuación se listan algunas:

dtCore: Contiene todas las funcionalidades básicas

- ✓ Asignación de dispositivos de entrada (Teclado, mouse, joystick, rastreadores)
- ✓ Modelos de movimiento (Volar, OVNI, caminar, orbitar y primera persona)
- ✓ Renders de entorno (Nubes, neblina, cielo, hora del día)
- ✓ Efectos particulares del Sistema (Humo, explosions, personalizados)
- ✓ Renders de terreno (Terreno procedual infinito, terrenos con alturas)
- ✓ Cargar archivos
  - .3dc, .3ds, .ac, .dw, .flt, .geo, .ive, .logo, .lwo, .lws, .md2, .obj, .osg, .tgz, .x, .zip
  - .bmp, .dds, .gif, .jpg, .pic, .png, .pnm, .rgb, .tga, .tiff, .txp
  - .wav
- ✓ Controles de cámara (Campo de vista, trípode)
- ✓ Soporte para múltiples cámaras
- ✓ Soporte para múltiples ventanas
- ✓ Física (Cuerpos rígidos, detección de colisiones, auto-delimitador de formas)

[22] Las librerías y otros componentes se los puede encontrar en: http://delta3d.org/index.php?topic=downloads

- ✓ Luces
- ✓ Trazo de nodos
- ✓ Completo soporte de OpenGL
- ✓ GLSL Fragmento de sombras

dtABD: Componentes básicos para aplicaciones de alto nivel. Útil para el desarrollo de algunas aplicaciones.

- ✓ Plantillas de aplicaciones.
- ✓ Interfaces de clima (visibilidad, covertura de nubes)
- ✓ Widget para la integración con otras herramientas.

dtUtil: Uso básico de objetos a través de Delta3D.

- ✓ Carga de archivos y utilidades XML.
- ✓ Texturas, ruido, matrices y utilidades de strings.
- ✓ Manejo de librería.
- ✓ Tipo de enumeraciones seguras.

dtGUI: Interfaz directa con Crazy Eddie's GUI.

- ✓ Interfaz de usuario diseñable y renderizable.
- ✓ Interfaces despegables y extensibles.
- ✓ Interfaces de usuario por defecto.

dtGUI es la librería más importante para el presente proyecto ya que nos permite trabajar con la interfaz gráfica de usuario y combinarla con el resto del juego.

2.4.3. CEGUI[23]

"CEGUI son las siglas de Crazy Eddie's Gui System; es una librería libre que proporciona ventanas y widgets para gráficos API / motores donde dicha funcionalidad no está disponible de forma nativa, o es demasiado sencilla. La librería está orientada a objetos, escrito en C++, y está dirigida a los desarrolladores de juegos que debería gastar su tiempo creando grandes juegos, la no construcción de subsistemas GUI!

[23] CEGUI: http://cegui.org.uk/

CEGUI es usada como la alternativa para interfaces gráficas de usuario en proyectos de Ogre3D, es soportado por Microsoft® DirectX® 8.1 and 9, OpenGL, y el motor de Irrlicht de forma nativa."(Turner, 2014)

2.4.4. CEED[24]

CEED es una herramienta gráfica de CEGUI, tiene una interfaz clara y sencilla en caso de que se desee realizar cambios a nivel de código tiene una venta de edición disponible, además de un editos de archivos *.imageset y *.layout.

Es una herramienta muy poderosa especializada en diseño y desarrollo de interfaces gráficas de usuario y es muy fácil de usar, además permite exportar archivos compatibles con el entorno de Delta3D. A continuación se muestra una captura del programa recuperada de la página web de Crazy Eddie.

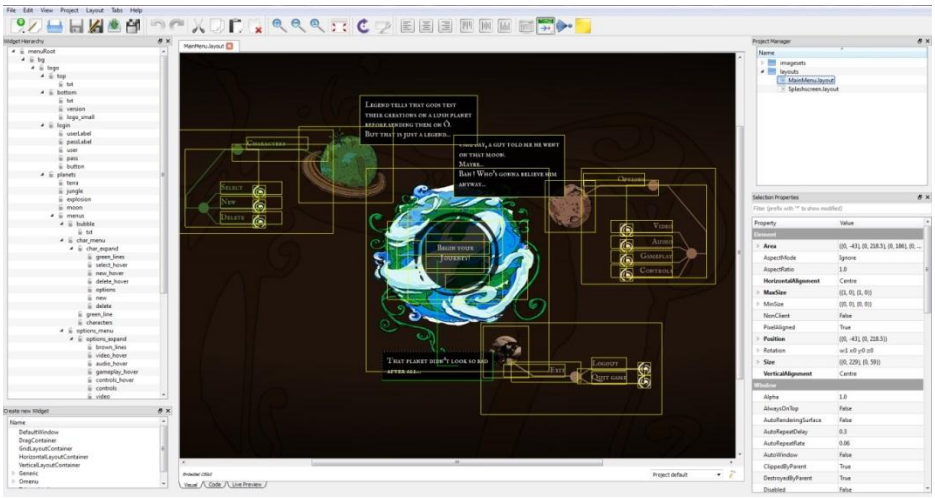


Imagen 2, Inferfaz de CEED, recuperada de (Turner & Team, 2014)

Aunque CEED es una herramienta muy completa, la versión más estable suele presentar problemas de compatibilidad con algunos proyectos, para contrarrestar esto se puede utilizar una herramienta con el mismo propósito pero más antigua que no posee todas las funcionalidades de CEED, pero no presenta ningún problema al momento de exportar los archivos. Dicha herramienta, también diseñada por Crazy Eddie, se llama CELayoutEditor.

[24] CEED: http://cegui.org.uk/download/ceed-snapshot11

2.4.5. CELayoutEditor[25]

Es un editor de diseño de interfaces creado por el equipo de Crazy Eddie; Funciona perfectamente con CEGUI. La ventaja que el programa brinda al usuario es que no necesita codificar nada ya que todas las funcionalidades están disponibles en un entorno visual en el que el usuario simplemente debe arrastrar los componentes hasta la pantalla y la codificación se realiza por detrás de forma automática.

2.5. Herramientas de diseño

Aunque todo diseño inicia con bocetos en papel y lápiz, existen softwares que facilitan dicho trabajo al momento de plasmarlo en un archivo digital.

2.5.1. Adobe Illustrator[26]

Es una herramienta creada por Adobe para diseñadores, es extremadamente versátil y relativamente fácil de aprender; Se pueden realizar diseños de gran calidad ya que posee herramientas extremadamente útiles que facilitan mucho el trabajo de diseño, además al ser tan comercial existe un sin número de tutoriales gratuitos disponibles en línea, puede exportar e importar archivos en diferentes formatos lo que la hacen muy compatible, el único inconveniente es que requiere licencia. La versión utilizada en el presente proyecto fue Adobe Illustrator CS6.

2.5.2. Adobe Photoshop[27]

Al igual que Illustrator, Photoshop es una herramienta de la familia de Adobe, usada generalmente para la edición y retoque de fotografías e imágenes, y en el presente proyecto se trabajó con la versión CS6. Aunque Illustrator es muy completo, hay ciertas tareas que son más sencillas en Photoshop por lo que suelen utilizarse a la par para la creación de diseños.

Photoshop también es muy útil para el manejo y creación de texturas, cosa que Illustrator no maneja muy bien y los archivos pueden intercambiarse entre ambos programas de forma natural. Al ser de adobe tiene básicamente las mismas ventajas y desventajas que Illustrator; tiene herramientas extremadamente útiles, infinidad de tutoriales pero requiere de una licencia pagada.

[25] CELayoutEditor: http://cegui.org.uk/wiki/CELayoutEditor_Downloads_0.7.1
[26] Adobe Illustrator: http://www.adobe.com/la/products/illustrator.html
[27] Adobe Photoshop: http://www.adobe.com/products/photoshop.html

2.5.3. Sodipodi[28]

De acuerdo con la Wikipedia “es un editor de gráficos vectoriales libre distribuido bajo la licencia GNU General Public License. Está disponible para GNU/Linux y Microsoft Windows y actualmente su desarrollo se encuentra descontinuado, siendo la última versión la 0.34, publicada en febrero de 2004”(Wikipedia, 2013).

Como se puede ver, actualmente se encuentra muy descontinuado y no brinda muchas posibilidades de exportación para sus archivos. Tampoco posee un sitio oficial en internet de donde se lo pueda descargar o consultar inquietudes sobre el producto.

2.5.4. Inkscape[29]

También es un editor de gráficos de código abierto y multiplataforma, fue creado por algunos de los desarrolladores de Sodipodi, por lo que se podría decir que es una versión mejorada del mismo. Actualmente se encuentra en la versión 0.91 liberada en enero del 2015, esto es una gran ventaja en comparación a su predecesor descontinuado.

Si bien no es tan potente como los paquetes de adobe, posee algunas de sus funcionalidades más populares y como una mejora propia permite esculpir objetos en 3 dimensiones.

[28] Sodipodi: http://sourceforge.net/projects/sodipodi/
[29] Inkscape: https://inkscape.org/es/

2.6. Análisis comparativo de las herramientas

- ***Motores de juego***

Herramienta	Pros	Contras
Unity	• Código abierto (Condicional) • Herramientas para diseño de alta calidad.	• Conocimiento nulo de la herramienta.
Delta3D	• Código abierto. • Compatibilidad con Visual Studio. • Experiencia del profesor Prometeo en el uso del programa.	• Restricción en cuanto a imágenes de alta calidad.

- ***Herramientas de diseño***

Herramienta	Pros	Contras
Sodipodi	• Intuitivo y sencillo.	• Software descontinuado. • Formatos limitados.
Inkscape	• Modelado 3D.	• Formatos limitados.
Adobe (Illustrator/Photoshop)	• Gran cantidad de tutoriales. • Diseños en alta definición. • Compatibilidad con muchos formátos.	• Licencia pagada.

- ***Implementación de diseños***

Herramienta	Pros	Contras
CELayout Editor	• Muy estable. • Simple.	• Incompatible con imágenes de alta calidad.
CEED	• Intuitivo y sencillo. • Compatible con imágenes de alta calidad.	• Versión inestable.

2.7. Herramientas seleccionadas

Para la etapa de diseño se utilizará el paquete de adobe (CS6) debido a la compatibilidad que poseen sus programas entre sí, lo que facilitaría la posible adición posterior de animaciones de transición y audios en el juego.

Para el desarrollo de las interfaces previa implementación se utilizará CEED debido a la mejor compatibilidad que posee con archivos de alta calidad, y en caso de tener algún problema al momento de exportar alguno de los diseños a la plataforma de implementación se recurrirá a CELayoutEditor (0.7.1).

Para la implementación y vinculación de las interfaces con el sistema Gists – UTPL se utilizará Delta3D debido a que el sistema está desarrollado sobre ésta plataforma y eso facilitará la vinculación de las funcionalidades.

2.8. Interfaces gráficas de usuario

Según (TechTerms.com, 2015) las interfaces gráficas de usuario o “GUI por sus siglas en inglés (Graphical User Interface), se refieren a la interfaz gráfica de una computadora que permite a los usuarios hacer clic y arrastrar los objetos con un ratón en lugar de introducir texto en una línea de comandos. Dos de los sistemas operativos más populares, Windows y Mac OS, están basados en GUI. La interfaz gráfica de usuario se introdujo por primera vez al público por parte de Apple con el Macintosh en 1984. Sin embargo, la idea fue en realidad tomada de una interfaz de usuario anterior desarrollado por Xerox.”

En los juegos de video la interfaz gráfica es el enganche inicial con el usuario, ya que ésta debe ser amigable e interesante para que el jugador se sienta ansioso por jugar. Mientras mayor sea la relación entre la interfaz y el tema del juego mejor será el resultado, ya que el usuario se siente más inmerso en la experiencia y se pierde la barrera de la realidad.

La interfaz tiene algunas partes importantes pero tal vez la más importante sea el HUD, del inglés heads - up display. “El HUD es la forma más eficaz de comunicarse con el jugador. El HUD se refiere a cualquier elemento visual que comunica información al jugador”(Rogers, 2010).

La importancia del HUD recae en el hecho de que el usuario pasa la mayor cantidad de tiempo de juego en contacto con el mismo, ya que transmite la información necesaria para el jugador durante el juego. Según Scott Rogers en su libro Level Up! Guide to great Video Game Design (Rogers, 2010), existen siete elementos básicos en el HUD de un video juego promedio, las siguientes se enumeran a continuación.

1. Barra de salud/vidas restantes.
    - ✓ Indica el estado del personaje, sus vidas restantes y/o su salud en un momento dado.
    - ✓ Puede ser presentada por colores (verde, amarillo y rojo), un porcentaje, una jeringuilla que se va vaciando mientras el personaje pierde vida; En fin, como se mencionó anteriormente, el objetivo es que la experiencia del jugador sea más vivida por lo que lo ideal es que el indicador sea de alguna forma algo que concuerde con la historia.
    - ✓ La barra de vida no necesariamente tiene que limitarse a un 100%, se puede agregar vida extra si el personaje consigue una armadura, entonces primero se agotaría el porcentaje de la armadura y luego se afectaría la salud del personaje. Todo depende del juego.
2. Mira.

- ✓ La mira ayuda al jugador a orientar sus disparos y saber hacia dónde está viendo su personaje.
- ✓ Puede ser un punto, una cruz, un láser, una x, un círculo, etc. No hay ninguna limitante más que la imaginación del equipo de diseño.

3. Munición.
   - ✓ Indica al jugador cuantas balas le quedan en el arma que está usando, puede también mostrar los cartuchos restantes y cualquier otra información de este tipo que pueda ser útil dependiendo del arma.
4. Inventario.
   - ✓ El inventario muestra todo lo que el personaje lleva consigo; Armas, mapas, llaves, objetos encontrados, protecciones, dinero, etc.
   - ✓ El inventario suele tener un espacio limitado para aumentar la dificultad del juego ya que esto no permite llevar todo lo que uno desee, sino que obliga al jugador a sacrificar algún ítem para llevar otro.
5. Puntaje/experiencia.
   - ✓ Indica el puntaje que el jugador a acumulado hasta un punto determinado, o la experiencia que ha ido adquiriendo en su viaje a través del videojuego.
6. Radar/ mapa.
   - ✓ Ayuda al jugador a moverse por el mundo virtual. Algunos videojuegos tienen un “mundo” bastante amplio, o complejo por lo que el jugador suele requerir de un mapa para no perderse en el recorrido.
7. Aviso de interacción.
   - ✓ El aviso de interacción informa al jugador que puede interactuar con un objeto cercano. Un aviso común es un pequeño dibujo de la tecla o el botón con el que realiza la acción sobre el objeto.
   - ✓ El aviso de interacción no necesariamente debe ser tan simple como un botón, puede ser un submenú con algunas acciones como en la saga de “Hitman”, donde el jugador puede realizar una serie de acciones con algunos de los objetos con los que se encuentra en el juego, esto puede dificultar un poco el juego, pero eso no necesariamente es algo malo.

Si bien es cierto estos son los elementos básicos de un HUD, para nuestro propósito en particular tal vez no sea necesario involucrarlos a todos, ya que el objetivo de Gists – UTPL no solo es divertir al jugador sino también enseñarle algo, por otro lado es posible que se requieran otros elementos para

ayudar este propósito, por ejemplo un chat para realizar preguntas al profesor. Esto se determinará en el transcurso del proyecto.

Los íconos son quizás los pequeños detalles que hacen que la interfaz tenga una apariencia más involucrada con la historia, por ejemplo en un juego de asesinos los íconos pueden simular armas o disparos, a diferencia de un juego de carreras donde pueden parecer partes de un auto. Pero, qué es un ícono? “Un ícono puede ser cualquier imagen usada para representar a una persona, un lugar, una cosa o una idea.”(McCloud, 1994).

Los íconos deben ser simples para que cualquier persona que los vea le dé un mismo significado. Un ícono debe ser abstracto pero no demasiado general puesto que la idea básica se puede perder.

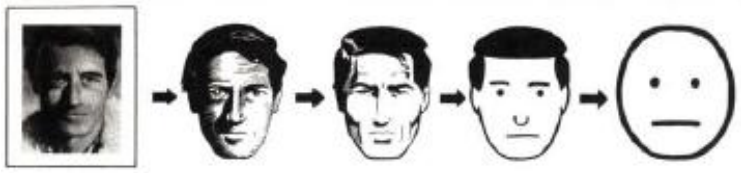


Imagen 3. Abstracción de una imagen recuperada de (McCloud, 1994)

Como se puede ver en la imagen obtenida del libro “Entendiendo los comics” de Soctt McCloud, las imágenes pueden abstraerse hasta su punto más simple siempre y cuando no pierdan el significado de lo que representan.

Otra parte importante del diseño es el lugar donde se ubican los íconos o los menús que los contengan. Se deben ubicar de tal forma que no se conviertan en una molestia para el jugador reduciendo demasiado su campo de visión o llamando demasiado su atención haciendo que se distraiga del juego. El jugador debe saber que están ahí y debe poder acceder a ellos con facilidad sin mayor esfuerzo; Mientras más tiempo pase en el juego más natural se hará el acceso a los menús por lo que lo más conveniente puede ser ubicar dichos accesos en los lugares que suelen ser más comunes como el ícono para salir en la esquina superior derecha por ejemplo.

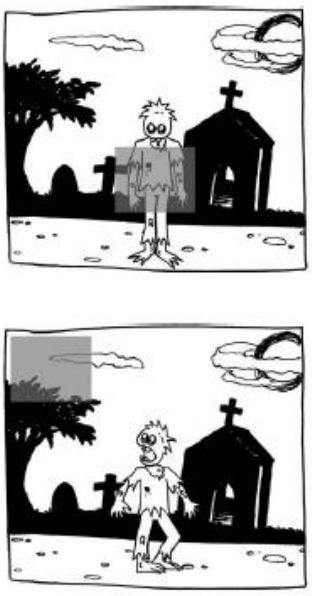


Imagen 4, Ubicación de menús 1, recuperada de (Rogers, 2010)

Imagen 5, Ubicación de menús 2, recuperada de (Rogers, 2010)

Es importante recordar que aunque la pantalla de juego sea la más importante, ya que el jugador pasa casi todo el tiempo en ella, no es la única interfaz que existe. Cada menú y pantalla del juego requieren de una igual preparación, estudio y diseño para realizar una interfaz completa que cumpla con su objetivo a cabalidad que, como ya se dijo, es el de involucrar al usuario en la historia para que se sienta en ella.

2.9. Campos de aplicación de los juegos serios

En la actualidad existen algunas empresas que trabajan en favor de la creación y aplicación de los juegos serios en varios campos, principalmente educativos. A continuación se exponen algunos ejemplos, algunos de los cuales se hace mucha referencia en el presente documento:

- ***Salud***

"Pulse" o pulso en español es uno de los más notables lanzado al mercado en el año 2007 desarrollado por la "Universidad de Texas A&M Corpus-Christi"[30], es un juego en 3 dimensiones que enseña técnicas para realizar diagnósticos médicos en estados de emergencia. Este juego involucra al jugador, tanto civil como militar, en situaciones extremas basadas en casos reales para probar su capacidad de resolver dichas crisis sin necesidad de poner vidas en riesgo, lo que sirve como una excelente práctica para pulir sus habilidades clínicas.

"Dental Implant Training Simulation[31]" o simulador de entrenamiento para implantes dentales fue diseñado para que los estudiantes de odontología puedan realizar prácticas en un ambiente seguro pero realista.

- ***Enseñanza***

El juego matemático "ST Math"[32] o "ST (JiJi) Math"[33] es un juego para nivel primario y secundario que enseña matemáticas mientras el jugador ayuda a un pingüino a cruzar de un lado al otro de la pantalla, disponible también en versión móvil. Se originó de la idea que no es necesario el uso de palabras para la enseñanza, destruyendo así las barreras idiomáticas y haciendo de la obtención de conocimiento una labor completamente intuitiva.

Los niveles básicos del juego, pensados para niveles primarios, forman las bases del pensamiento lógico – matemático en el jugador, explicando las operaciones básicas de una forma visual e interactiva para que el usuario entienda el porqué de cada respuesta; mientras que los niveles más avanzados engloban todo el conocimiento que se imparte en el nivel secundario, llegando incluso a explicar la integración y derivación de factores, sin perder su modalidad de juego, por lo que los usuarios sin importar su edad o lengua, pueden sacarle provecho.

[30] Texas A&M University Corpus Christi: http://www.tamucc.edu/
[31] Dental Implant Training Simulation: http://www.breakawaygames.com/serious-games/solutions/healthcare/
[32] ST Math: http://www.mindresearch.org/stmath
[33] ST (JiJi) Math: https://play.google.com/store/apps/details?id=air.net.mindresearch.stmath&hl=es
https://itunes.apple.com/us/app/st-math-school-version/id550846820?mt=8

El campo de los videojuegos serios, como puede verse, está involucrado en muchos ámbitos tratando de mejorar el aprendizaje de las personas en un tema determinado y de una forma didáctica y entretenida; aquí se muestran algunos ejemplos de juegos serios que existen actualmente y sus respectivas ramas, obtenidos de (BreakAway Ltd, 2012):

- ✓ Seguridad Nacional
  - o "Incident Comander[34]" o Comandante de incidentes, donde el jugador tiene que organizar a algunas entidades públicas como policía, bomberos, ambulancias, etc. Durante un incidente en su ciudad, ya sea un desastre natural, un ataque terrorista, una situación de rehenes, etc.
- ✓ Cambio Social
  - o "A Force More Powerful[35]" o una fuerza más poderosa, hace que el jugador deba dirigir una resistencia no violenta contra los dictadores basándose en los principios y las experiencias de algunos de los activistas no violentos más importantes de la historia como Gandhi, Dr. Martin Luther King, entre otros.

Estos son solo algunos de los juegos serios encontrados en la actualidad, los cuales se analizará para el desarrollo del sistema Gists – UTPL, y en el presente proyecto.

En el ámbito de interfaces también existen algunos proyectos que serán de gran utilidad en la presente investigación, como el proyecto de fin de carrera del Ing. Iván Uzquiano, "Acceso manual e interfaz gráfica para el juego AI-LIVE" (Uzquiano, Fernández, & Ortiz, 2010).

Además existen varios artículos dedicados al diseño y la importancia de las interfaces en juegos de video. Algunos son específicos como el artículo de Mar Canet "Innovación en interfaces para videojuegos desde el Game Art"(Canet, 2012), mientras que otros son un poco más generales como la explicación de Diseño de interfaz del Departamento de Ingeniería del Software e Inteligencia Artificial de la Universidad Complutense de Madrid.(Peinado, 2007).

Tal vez uno de los estudios que sean más útiles para el presente proyecto sea el de "Interfaz para videojuegos" realizado por los tecnólogos informáticos Andrés Pastorini y Alvaro Martínez, cuyo objetivo es "Conocer ciertos principios involucrados en el área de diseño gráfico y visión perceptual, y brindar una introducción en el diseño y programación de interfaces gráficas para videojuegos."(Pastorini & Martínez, 2014).

[34] Incident Comander: http://www.breakawaygames.com/serious-games/solutions/homeland/
[35] A Force More Powerful: http://www.breakawaygames.com/serious-games/solutions/social/

3. DESARROLLO

3.1. Diseño

Para el diseño de la interfaz gráfica del juego se analizó los mini juegos que involucra el sistema para encontrar un factor común que los relacione y trabajar las ventanas con dicho factor. Uno de los mini juegos que contienen el Gists - UTPL es un tanque de guerra que explica el lanzamiento de proyectil, si solo se hubiera tomado en cuenta este juego se hubiera diseñado una interfaz con un ambiente militar, siendo por ejemplo el ícono del chat un radio militar, pero debido a la variedad de ámbitos involucrados en los juegos se decidió que el diseño debía ser algo más neutral y ya que el objetivo del sistema es ejercitar lo aprendido en clase, se diseñaron interfaces relacionadas a la educación.

La ventana inicial es un cuaderno con un diseño similar a los de la UTPL que está sobre un escritorio de madera con el nombre del juego “GISTS” en la parte inferior derecha de la pasta, la cruz de la universidad como símbolo de la misma en el centro y el nombre de la logo de la universidad junto con su escudo en la parte superior derecha.

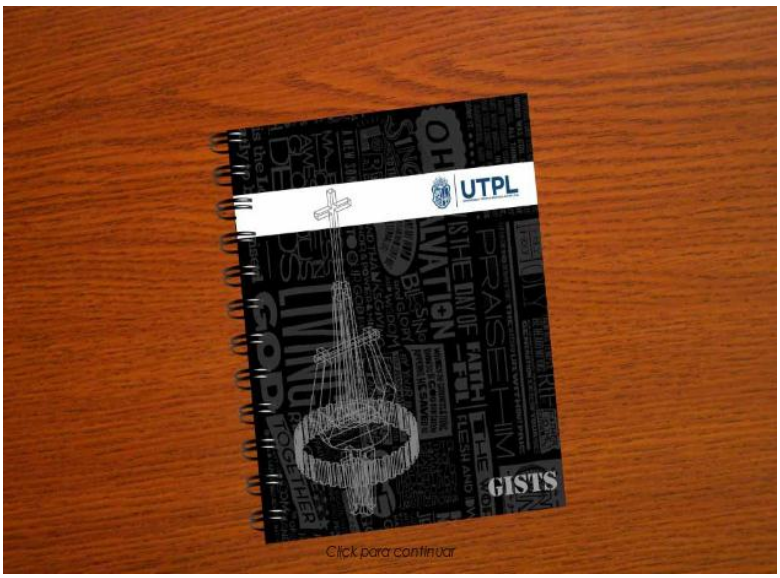


Imagen 6, Pantalla inicial del Gists-UTPL

En la siguiente ventana el jugador debe ingresar al sistema con un usuario y contraseña registrado en el sistema y la dirección de su computador para poder ingresar a la red; en esta ventana el cuaderno está abierto y simula los datos personales del estudiante.

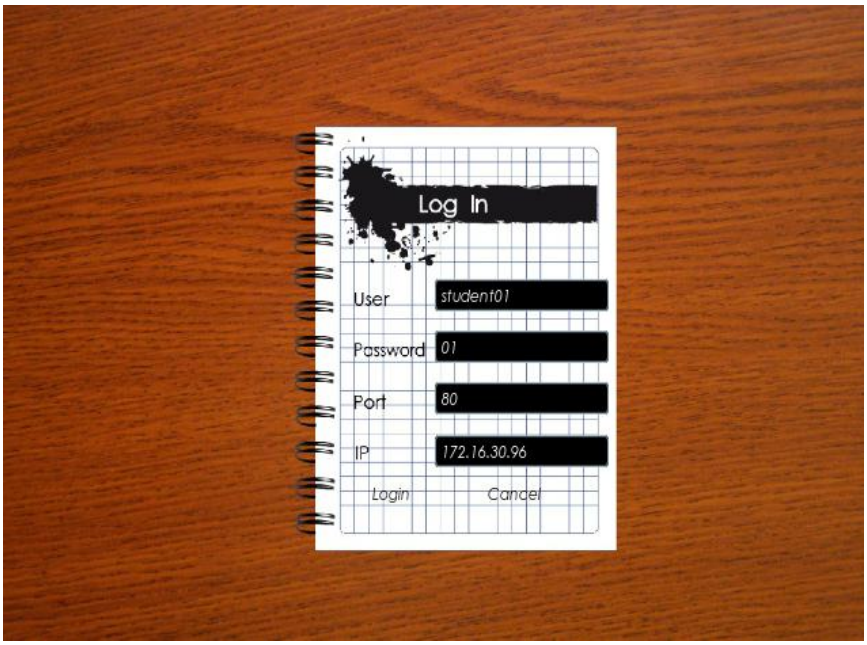


Imagen 7, Pantalla de Logeo del GISTS UTPL

Una vez logueado el usuario puede:

- ✓ Comenzar a jugar.
- ✓ Escoger el idioma del juego.
- ✓ Leer las instrucciones del juego.
- ✓ Cambiar algunas opciones del juego.
- ✓ Volver a la ventana anterior.

En la ventana de idioma puede elegir entre inglés, español o chino mandarín, aunque actualmente dichas opciones no son funcionales debido a que el profesor Prometeo únicamente implementó el sistema en inglés así que se deben realizar las respectivas traducciones. El botón para regresar al menú anterior simula las etiquetas con las que algunos estudiantes separan sus cuadernos. Por defecto el juego está en inglés ya que es un idioma más universal.

Finalmente en la ventana principal del juego se despliega un entorno en tres dimensiones donde se cargan los mini juegos escogidos ya sea por el estudiante o los propuestos por el profesor encargado. Una barra superior que muestra los íconos de los diferentes temas comprendidos en el sistema como

cinemática, electromagnetismo, óptica, termología y física cuántica. También se muestran todas las opciones básicas del juego como comenzar, chatear, guardar la partida, cargar una partida o volver al menú principal.

Una vez el jugador elije una de las áreas en el menú superior situando el cursor sobre la misma se despliega un submenú en la parte izquierda de la pantalla con los subtemas de dicha área sobre las cuales el jugador también puede acceder al poner el cursor sobre cualquiera de ellas y estas a su vez descubren un catálogo en la parte inferior derecha de la pantalla con todos los mini juegos que tengan referencia a dicho subtema.

Para acceder a cada uno de los mini juegos el jugador debe seleccionarlos con el cursor y arrastrarlos hasta el entorno en 3D donde se carga la animación y se muestra una ventana emergente que explica el ejercicio que debe realizar para cumplir el objetivo.

Imagen 8, Pantalla de Juego, fuente Proyecto Prometeo Gists - UTPL

Como ya se explicó anteriormente todos estos menús forman parte del “HUD” y se localizaron en la pantalla tal como se recomienda en “Level Up! Guide to great Video Game Design” (Rogers, 2010), están ubicados en los alrededores de la pantalla para dejar libre el lugar de juego y no molestar al estudiante.

- ***Análisis de colores para la interfaz***

Los colores utilizados en los diseños también fueron analizados, según estudios los colores crema o pasteles fomentan el aprendizaje en los estudiantes, por lo algunas instituciones educativas como la UTPL utilizan dichos colores en sus edificios y aulas. La misma teoría se aplicó para el color de las interfaces del sistema tanto en los menús como en los íconos y el fondo del entorno 3D.

Un gran limitante para los colores del HUD fue el motor gráfico con el que trabaja el sistema, ya que no posee una paleta muy amplia y el color del piso donde corren las animaciones no se puede editar lo que obligó a que los colores sean bajo esa gama.

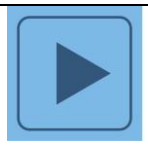
Pese a esto se tomó en cuenta la teoría de colores del Feng shui[36] que se basa en los cinco elementos básicos y los colores que representan a cada uno de ellos para combinarlos de mejor manera y obtener un ambiente equilibrado.(Pérez Pérez & Pérez Domingo, 2014) Los elementos y sus colores son:

- ✓ Fuego: Rojo, anaranjado, rosado; colores cálidos.
- ✓ Tierra: Marrón, beige, amarillo.
- ✓ Metal: Dorado, plateado, blanco.
- ✓ Agua: Negro, azul, turquesa.
- ✓ Madera: Verde.

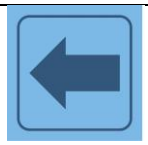
- ***Diseño de Íconos***

En el diseño de los íconos se trató de sintetizar la funcionalidad de cada botón con un diseño general para darle armonía al HUD sin dejar de lado el significado individual de cada uno de forma tal que el usuario final sepa lo que significan sin necesidad de un título.

En las funciones clásicas se dibujaron íconos similares a la mayoría de los programas comerciales, a continuación dichos íconos en su versión normal y con sus colores invertidos, utilizados en el sistema para dar la impresión de que se aplasta un botón.

Función	Ícono normal	Ícono aplastado
Jugar		
Chat		
Guardar		

[36] Feng Shui: http://www.fengshuinatural.com

Cargar		
Atrás		

- ***Incorporación de Diseños al sistema***

Para insertar los diseños en el sistema se deben crear algunos archivos por cada imagen que se detallan a continuación.

Primero se debe crear un archivo *.imageset en el cual se especifica el nombre del archivo, la dirección donde se localiza la imagen. Los archivos imageset pueden dividir una imagen en partes, esto es muy útil en un sistema grande que tenga demasiadas imágenes; en estos casos se crea una sola imagen que tenga todos los diseños que se vayan a utilizar en forma de textura, el problema de este método es que cada imagen pierde calidad por lo que se decidió crear archivos individuales. En ambos casos se debe especificar la posición en coordenadas X e Y donde comienza la imagen, el alto y el ancho de la misma. A continuación se muestra un ejemplo sencillo de un archivo imageset.

```
<?xml version="1.0" encoding="utf-8" ?>
	<Imageset Name="CoverBackgroundImage"
Imagefile="imagesets/CoverBackgroundImage.jpg">
		<Image Name="CoverBackgroundImage" XPos="0" YPos="0" Width="1020"
Height="720" />
	</Imageset>
```

Otro archivo necesario para la implementación de una interfaz en el *.layout, en el cual se definen las propiedades de las imágenes por ejemplo para definir un botón, una caja de texto, etc. Por ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<GUILayout >
        <Window Type="TaharezLook/Menubar" Name="RootVanilla/Vanilla/Console" >
                <Property Name="Font" Value="DejaVuSans-10" />
                <Property Name="Text" Value="Console" />
                <Property Name="UnifiedMaxSize" Value="{{0.8,0},{0.8,0}}" />
                <Property Name="UnifiedAreaRect"
Value="{{0.190278,0},{0.192903,0},{0.690278,0},{0.642904,0}}" />
                <Property Name="AutoRenderingSurface" Value="True" />
                <Window Type="TaharezLook/Button" Name="RootVanilla/Vanilla/Console/Submit" >
                        <Property Name="ID" Value="1" />
                        <Property Name="Font" Value="DejaVuSans-10" />
                        <Property Name="Text" Value="Submit" />
                        <Property Name="UnifiedAreaRect" Value="{{0,-7},{0,-13},{0.25,-7},{0,23}}" />
                        <Property Name="VerticalAlignment" Value="Bottom" />
                        <Property Name="HorizontalAlignment" Value="Right" />
                        <Property Name="WantsMultiClickEvents" Value="False" />
                </Window>
                …
```

Para cambiar un tipo de letra en el sistema se debe descargar el archivo fuente y crear un archivo *.font con el nombre del archivo que luego se llamará desde el sistema la dirección del tipo de fuente, el tamaño y las dimensiones que se deseen para el tamaño. A continuación un ejemplo:

```
<?xml version="1.0" ?>
    <Font Name="CenturyGothicItalic-24" Filename="CEGUI/fonts/CenturyGothicItalic.ttf"
Type="FreeType" Size="24" NativeHorzRes="1920" NativeVertRes="1200" AutoScaled="true"/>
```

Una vez creados todos estos archivos deben ser llamados desde un archivo *.scheme donde se especifique la dirección donde están guardados y el nombre con el que realizará el llamado desde el sistema de la siguiente forma:

```
<?xml version="1.0" ?>
<GUIScheme Name="TaharezLook">
    <Imageset Name="CoverBackgroundImage"
Filename="imagesets/CoverBackgroundImage.imageset"/>
        <Font Name="CenturyGothicItalic-24" Filename="CEGUI/fonts/Century-Gothic-Italic-24.font"/>
…
```

5.2. Especificación de Casos de Uso y diagramas de secuencia

Diagrama de casos de uso

Para el diseño de la interfaz se trabajó con el siguiente diagrama de casos de uso, en el cual se puede apreciar que la única acción que diferencia a los usuarios es que el profesor puede plantear un ejercicio para que el alumno resuelva jugando mientras el profesor puede ver en tiempo real las interacciones del alumno con el juego.

Desde el punto de vista de la interfaz esto no afecta ya que la ventana sería la misma simplemente las animaciones que el alumno realiza se ven reflejadas en la venta del profesor por lo que en conjunto con el Profesor Prometeo se decidió que la interfaz sea la misma tanto para cliente como para servidor (alumno, profesor respectivamente).

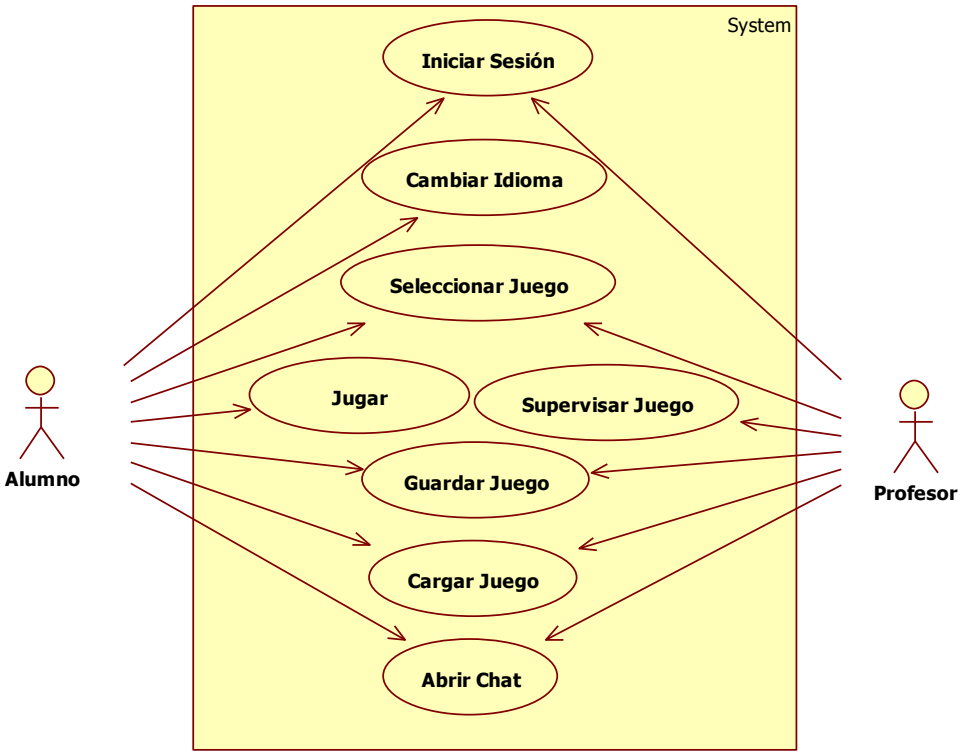


Diagrama 1, Diagrama de casos de uso para la interfaz del Gists-UTPL

Especificación de los casos de uso y Diagramas de secuencia

Nombre: Iniciar Sesión
1.1. Descripción
El usuario se loguea [37]en el sistema.
2. Flujo de Eventos
2.1. Flujo Básico
2.1.1. El caso de uso comienza cuando el usuario da clic en la opción "Click para continuar", y el sistema muestra una ventana con los datos que el usuario debe llenar.
2.1.2. El usuario ingresa los datos solicitados (nombre de usuario, contraseña, puerto y dirección IP), y da clic en "Login".
2.1.3. El sistema cargará la siguiente ventana.
2.2. Flujos Alternativos
2.2.1. Uno de los datos ingresados por el usuario es incorrecto, el sistema mostrará un error.
2.2.2. El usuario da clic en "Cancel" y el sistema vuelve a cargar la ventana inicial.
3. Precondiciones
Ninguna
4. Poscondiciones
Ninguna

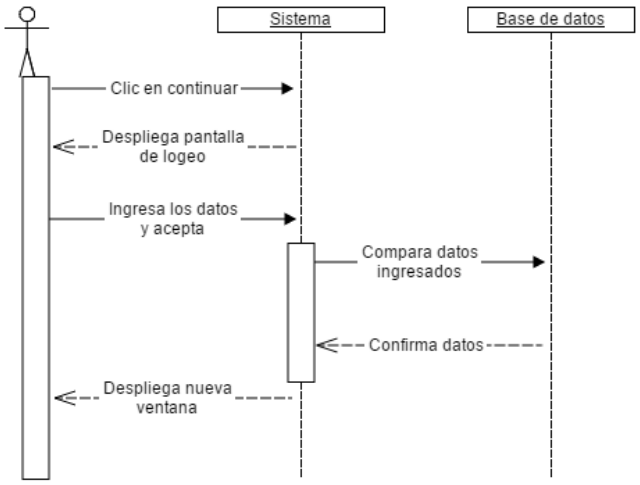


Diagrama 2, Diagrama de secuencia de Iniciar Sesión

[37] Loguear: "Proceso mediante el cual se controla el acceso individual a un sistema informático mediante la identificación del usuario utilizando credenciales provistas por el usuario."(Wikipedia, 2014b)

Nombre: Cambiar Idioma

1.1. Descripción

El usuario cambia el idioma del sistema.

2. Flujo de Eventos

2.1. Flujo Básico

2.1.1. El caso de uso comienza cuando el usuario da clic en la opción "Languages", y el sistema muestra una ventana con los idiomas disponibles.

2.1.2. El usuario selecciona el idioma que desee y el sistema vuelve al menú anterior con actualizando las instrucciones al idioma seleccionado.

2.2. Flujos Alternativos

2.2.1. El usuario da clic en el botón de volver sin elegir ningún idioma, y el sistema despliega el menú anterior sin realizar ningún cambio.

3. Precondiciones

3.1. El usuario debe estar logeado.

4. Poscondiciones

Ninguna

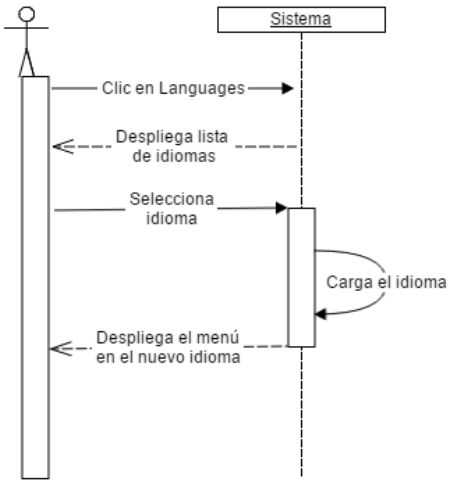


Diagrama 3, Diagrama de secuencia de Cambiar Idioma

Nombre: Seleccionar Juego
1.1. Descripción
El usuario selecciona el juego que desea según el tema que desea estudiar.
2. Flujo de Eventos
2.1. Flujo Básico
2.1.1. El caso de uso comienza cuando el usuario da clic en "Start", y el sistema despliega la pantalla de juego con todos sus menús.
2.1.2. El usuario debe pasar el cursor sobre el área de estudio que desee del menú superior; de izquierda a derecha son las siguientes: Mecánica, magnetismo, óptica, termología y física cuántica. El sistema mostrará un submenú en la parte izquierda de la pantalla.
2.1.3. El usuario debe pasar con el cursor sobre los íconos del submenú y debe escoger uno de los temas. En cada tema el sistema desplegará en la parte inferior derecha una lista de ejercicios relacionados con dicho tema.
2.1.4. El usuario selecciona un ejercicio y lo arrastra al centro de la pantalla donde el sistema cargará el juego correspondiente y desplegará un menú con las instrucciones del mismo.
2.2. Flujos Alternativos
3. Precondiciones
3.1. El usuario debe estar logueado.
4. Poscondiciones
Ninguna

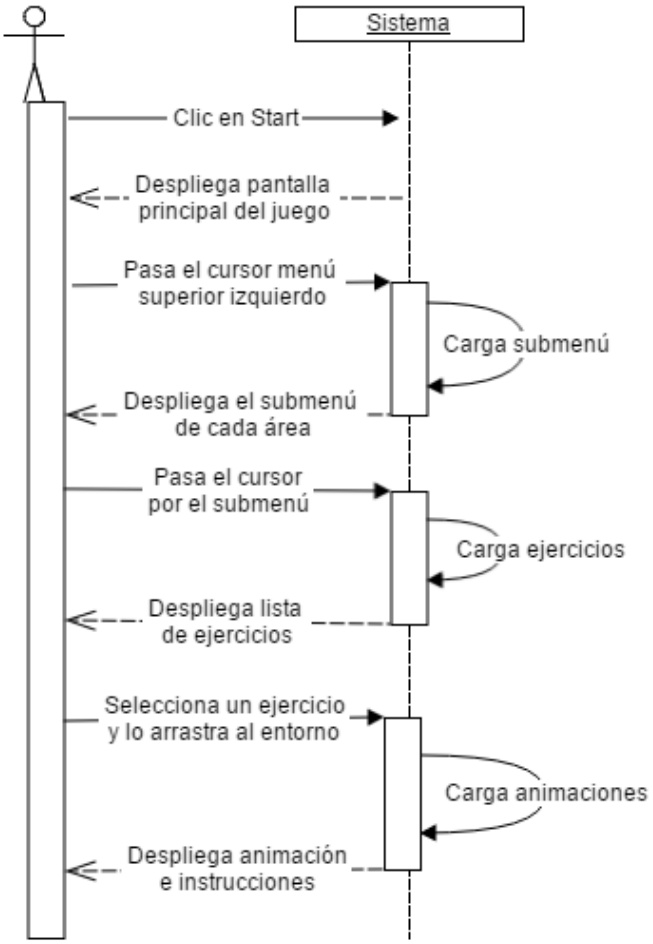


Diagrama 4, Diagrama de secuencia de Seleccionar Juego

Nombre: Jugar

1.1. Descripción

Una vez seleccionado el juego el alumno ejecutará las instrucciones de cada juego.

2. Flujo de Eventos

2.1. Flujo Básico

2.1.1. El caso de uso comienza cuando el alumno o el profesor seleccionaron un juego y el sistema despliega la animación del mismo al alumno.

2.1.2. El alumno lee las instrucciones del ejercicio y resuelve lo que el sistema solicite, ingresa el resultado y da clic en "Play" y el sistema ejecuta el juego con su respuesta.

2.1.3. El alumno ve los resultados de sus cálculos (correctos o incorrectos) por medio de las animaciones.

2.2. Flujos Alternativos

2.2.1. En lugar de dar clic en "Play" el alumno selecciona otro juego, el sistema carga el ejercicio.

2.2.2. En lugar de dar clic en "Play" el alumno carga una partida guardada, el sistema carga la partida guardada.

2.2.3. En lugar de dar clic en "Play" el alumno da clic en el botón "Atrás", el sistema muestra el menú general.

3. Precondiciones

3.1. Haber iniciado sesión como estudiantes.

4. Poscondiciones

Ninguna

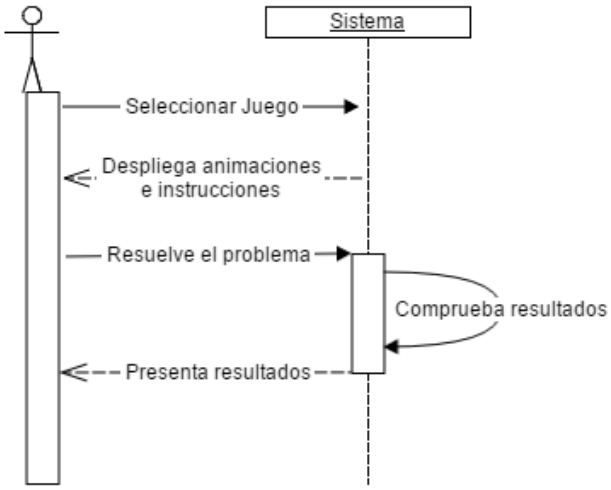


Diagrama 5, Diagrama de secuencia de Jugar

Nombre. Supervisar Juego
1.1. Descripción
Una vez seleccionado el juego el profesor supervisa la resolución del ejercicio.
2. Flujo de Eventos
2.1. Flujo Básico
2.1.1. El caso de uso comienza cuando el profesor seleccionó un juego para que el alumno resuelva y el sistema despliega las animaciones del juego.
2.1.2. El sistema muestra los resultados del alumno y ejecuta la animación según los mismos.
2.2. Flujos Alternativos
2.2.1. En cualquier punto el profesor carga otro ejercicio para el alumno.
3. Precondiciones
Ninguna
4. Poscondiciones
Ninguna

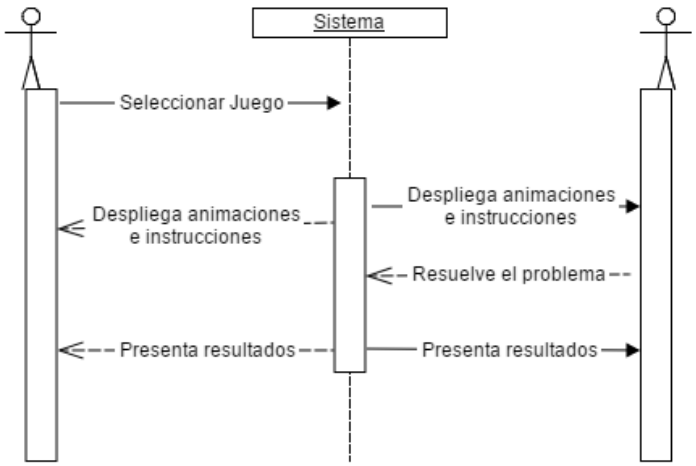


Diagrama 6, Diagrama de secuencia de Supervisar Juego

Nombre: Guardar Juego

1.1. Descripción

El usuario guarda su avance ya sea en un ejercicio o en general.

2. Flujo de Eventos

2.1. Flujo Básico

2.1.1. El caso de uso inicia cuando el usuario da clic en el botón de "Guardar", y el sistema despliega una ventana tipo Windows para que defina la ubicación donde se guardará el avance.

2.1.2. El usuario escoge la ruta y el nombre del archivo y da clic en "Guardar", el sistema crea el archivo y despliega nuevamente la ventana en la que encontraba.

2.2. Flujos Alternativos

2.2.1. El usuario cancela la grabación del archivo en lugar de confirmar el proceso y el sistema despliega la ventana en la que se encontraba.

2.2.2. El nombre de archivo con el que se desea guardar ya existe, el sistema solicitará que el usuario cambie el nombre o que confirme la sobre escritura del mismo.

3. Precondiciones

3.1. El usuario debe estar logueado.

4. Poscondiciones

4.1. Se creará un archivo en una ruta determinada con información del juego.

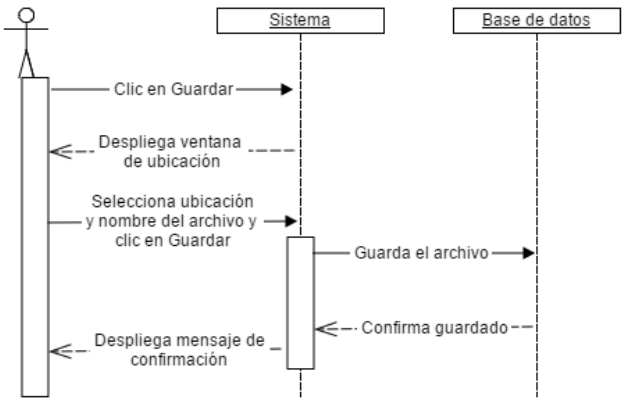


Diagrama 7, Diagrama de secuencia de Guardar Juego

Nombre: Cargar Juego
1.1. Descripción
Un usuario carga un juego guardado anteriormente para continuar con el mismo.
2. Flujo de Eventos
2.1. Flujo Básico
2.1.1. El caso de uso comienza cuando el usuario da clic en el botón de "cargar" y el sistema despliega una ventana de Windows para que el usuario busque la ruta del archivo guardado.
2.1.2. El usuario selecciona el archivo que desea cargar y da clic en "Aceptar", el sistema carga la partida guardada y despliega la ventana del juego con todas las variables como se quedaron en el juego.
2.2. Flujos Alternativos
2.2.1. El usuario cancela la carga del archivo en lugar de confirmar el proceso y el sistema despliega la ventana en la que se encontraba.
3. Precondiciones
3.1. El usuario debe estar logueado.
3.2. Debe existir por lo menos una partida guardada con anterioridad.
4. Poscondiciones
Ninguna

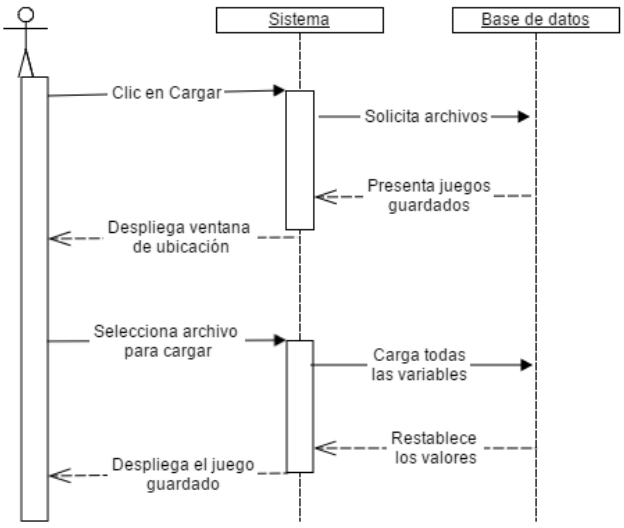


Diagrama 8, Diagrama de secuencia de Cargar Juego

Nombre: Abrir Chat

1.1. Descripción

El usuario abre una sala de chat con otro usuario en línea para intercambiar mensajes en tiempo real.

2. Flujo de Eventos

2.1. Flujo Básico

2.1.1. El caso de uso comienza cuando el usuario da clic en el botón de "Chat", el sistema despliega un submenú con las siguientes opciones: Windows, Motion Models, Assambly Tools, Chat Windows.

2.1.2. El usuario selecciona la opción "Chat Windows" y el sistema muestra una ventana de chat donde el estudiante puede comunicarse con el profesor y viceversa en tiempo real.

2.1.3. El usuario que inicio la conversación escribe el mensaje que desea enviar y da clic en "Submit" o presiona la tecla enter, el sistema envía el mensaje y abre la ventana del chat del destinatario automáticamente.

2.2. Flujos Alternativos

2.2.1. El usuario da clic en cancelar antes de enviar el mensaje y el sistema cierra la ventana de chat.

3. Precondiciones

3.1. Tanto el profesor como el estudiante deben estar logeados, por lo menos deben haber dos usuarios conectados.

4. Poscondiciones

Ninguna

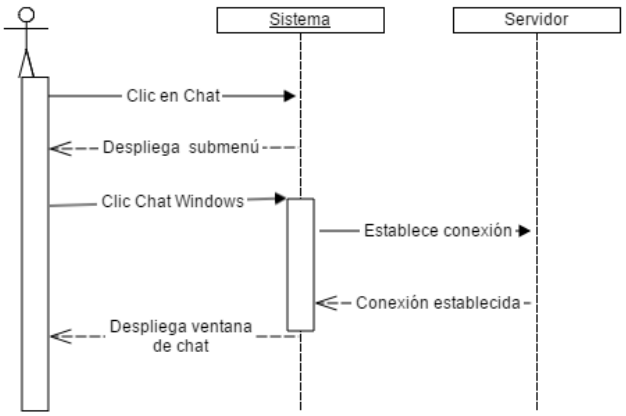


Diagrama 9, Diagrama de secuencia de Abrir Chat

4.1. Codificación

Una vez que todos estos archivos estén listos se puede pasar a la etapa de implementación en el código. Primero se explicará cómo cambiar un fondo de pantalla por uno de los diseños que creamos. Todo el código se realizó en C++.

Se define una variable para el fondo en este caso llamada "_background" y la definimos como una ventana principal que tendrá como fondo una imagen estática y luego indicamos el nombre del archivo imageset; una buena práctica es nombrar el imageset igual que la imagen para evitar confusiones. Luego se definen las propiedades de la variable, en este caso se define que es una imagen, que se utilizará de fondo para la ventana y se especifica la imagen, se desactiva el fondo de la ventana clásico y el marco del fondo y finalmente se define la posición en la pantalla donde se colocará la imagen y las dimensiones de la misma.

```
_background = _gui
>CreateWidget(_mainWindow, "TaharezLook/StaticImage", "CoverBackgroundImage");
_background
>setProperty("Image", "set:CoverBackgroundImage image:CoverBackgroundImage");
_background
>setProperty("BackgroundEnabled", "false");
_background
>setProperty("FrameEnabled", "false");
_background
>setPosition(CEGUI::UVector2(cegui_reldim(0.0f), cegui_reldim(0.0f)));
_background
>setSize(CEGUI::UVector2(cegui_reldim(1.0f), cegui_reldim(1.0f)));
```

A continuación se explica cómo dar a un texto la funcionalidad de un botón y cambiar el tipo de letra de dicho texto. Primero se define una variable dándole atributos de botón, se define el texto del botón, el tamaño, la posición y el tipo de fuente tal y como se lo definió en el archivo *.scheme.

```
_enter=static_cast<CEGUI::PushButton*>(_gui
>CreateWidget(_background,"TaharezLook/Button","Enter_English"));
_enter
>setText("Click para continuar");
_enter
>setSize(CEGUI::UVector2(cegui_reldim(0.6f), cegui_reldim(0.2f)));
_enter
>setPosition(CEGUI::UVector2(cegui_reldim(0.2f), cegui_reldim(0.85f)));
_enter
>setFont("CenturyGothicItalic
24");
```

De igual manera para crear un editbox el proceso es casi el mismo; se define una variable en la que se especifica las funcionalidades de un editbox, y luego se define el tamaño la posición, el texto que irá por defecto y el tipo de letra.

```
_nameBox=static_cast<CEGUI::Editbox*>(_gui
>CreateWidget(_background,"TaharezLook/Editbox","surname"));
_nameBox
>setSize(CEGUI::UVector2(cegui_reldim(0.2f), cegui_reldim(0.05f)));
_nameBox
>setPosition(CEGUI::UVector2(cegui_reldim(0.42f), cegui_reldim(0.46f)));
_nameBox
>setText("student01");
_nameBox
>setFont("CenturyGothicItalic
24");
```

Cabe a recalcar que las propiedades no se restringen a las utilizadas aquí, y no implica que se requiera de todas para que el sistema funcione, por ejemplo si no se define un tipo de letra el programa utilizará el que esté designado por defecto.

En caso de diseñar un botón incluido en el fondo como se realizó en la ventana de idioma al simular la etiqueta del cuaderno, el procedimiento es el mismo, se crea la variable para el botón, se define su posición y su tamaño de forma que abarque todo el área del botón diseñado y en lugar de texto se llena con algunos espacios en blanco. A continuación el ejemplo del botón de “atrás“ de la ventana de idioma.

```
mExit=static_cast<CEGUI::PushButton*>(gui.CreateWidget(mBackground,"TaharezLook/Button","mexit")
);
mExit->setSize(CEGUI::UVector2(cegui_reldim(0.2f),cegui_reldim(0.1f)));
mExit->setPosition(CEGUI::UVector2(cegui_reldim(0.62),cegui_reldim(0.24f)));
mExit->setText("    ");
```

Para crear un menú se define la variable que lo albergará y se declara como Menubar, se especifica el grado de trasparencia del menú siendo 0.0 la máxima transparencia y 1.0 transparencia nula y al igual que en los botones se establece el tamaño y la posición del menú en la pantalla.

```
mWindowMenuBar =
static_cast<CEGUI::Menubar*>(gui.CreateWidget(mUpBackground,"TaharezLook/Menubar","RunningWindow
Menubar"));
mWindowMenuBar->setAlpha(1.0f);
mWindowMenuBar->setSize(CEGUI::UVector2(cegui_reldim(1.00f),cegui_reldim(1.0f)));
mWindowMenuBar->setPosition(CEGUI::UVector2(cegui_reldim(0.00f),cegui_reldim(0.00f)));
```

Una vez creado el menú se definen las opciones del mismo, a continuación se muestra cómo definir un ícono que actuará como opción en el menú, para esto definimos la variable que lo contendrá como una instancia de la variable del menú que se creó previamente, indicamos el nombre del archivo imageset del ícono que se utilizará y se define el tamaño y la posición, finalmente creamos un evento para mostrar el submenú cuando el cursor esté sobre el ícono.

```
mMechanics=static_cast<CEGUI::Window*>(gui.CreateWidget(mWindowMenuBar
"TaharezLook/StaticImage"
"mMechanics"));
mMechanics->setProperty("Image"
"set:mechanics image:mechanics");
mMechanics->setSize(CEGUI::UVector2(cegui_reldim(0.1f)
cegui_reldim(1.0f)));
mMechanics->setPosition(CEGUI::UVector2(cegui_reldim(0.0f)
cegui_reldim(0.0f)));
CEGUI::WindowManager::getSingleton().getWindow("mMechanics")-
>subscribeEvent(CEGUI::Window::EventMouseEntersArea.c_str()
dtGUI::GUI::Subscriber(&RunningWindow::onShowMechanicsMenuBar
this));
```

Aunque el objetivo es que los íconos sean lo más explícitos posibles es recomendable mostrar también una etiqueta de título que indique la funcionalidad de los botones. Para hacerlo se define un evento que se efectúe cuando el usuario arrastre el cursor sobre dicho botón, donde se especifica la posición de la pantalla donde aparecerá el título, el texto, y el cambio de estado de visible a invisible.

```
bool RunningWindow::handlemPlay(const CEGUI::EventArgs& e)
{
_upTextBackground-
>setPosition(CEGUI::UVector2(cegui_reldim(0.53f),cegui_reldim(0.1f)));
       _upText->setText("Play");
       _upTextBackground->show();
       return true;
}
```

Para los íconos del HUD se diseñó el mismo botón con los colores invertidos para dar la apariencia de que el botón se hunde al oprimirlo; para implementar esto, ambas imágenes deben tener un archivo *.imageset individual donde el único cambio será el nombre de cada imagen, sin alterar el tamaño ni la posición de la misma para que el cambio entre imágenes se vea natural, en la codificación se define una variable dándole las propiedades de botón y se le asigna la imagen del botón sin pulsar, se define la propiedad de “PushedImage” y a ésta se le asigna la otra imagen.

```
(...)
mPlayOff->setProperty("NormalImage","set:mPlayOff image:mPlayOff");
mPlayOff->setProperty("HoverImage","set:mPlayOff image:mPlayOff");
mPlayOff->setProperty("PushedImage","set:mPlayOn image:mPlayOn");
(...)
```

4. CAPACITACIÓN SOBRE DESARROLLO DE JUEGOS CON CONTENIDO SERIO

Como parte del proyecto el profesor Prometeo responsable, realizó una capacitación de 20 horas impartida a 20 estudiantes de la titulación de Sistemas Informáticos y de Computación de la UTPL, en la cual la función del tesista fue de asistente de cátedra. El curso llamado “Planeación y diseño de juegos formativos – Serious Games” de dividió en dos partes; Primero, la lógica del juego en C++ y segundo, el diseño de interfaces en dos dimensiones para videojuegos con contenido serio, este último estuvo a mí cargo durante el curso; a continuación se describen los temas impartidos.

- ✓ Importancia de la interfaz en un juego
- ✓ Diseño
- ✓ Menús y HUD “Heads Up Display”
- ✓ Tipos de archivos
- ✓ CELayout Editor
- ✓ Implementación en C++

A continuación se describen brevemente cada uno de los contenidos impartidos:

La interfaz gráfica de un juego es una de las partes más importantes del mismo ya que es lo que llama la atención del jugador inicialmente, es el primer impacto y lo que hará que el jugador se interese por el juego antes de conocer su historia y jugabilidad que son los componentes que finalmente atrapan al jugador, si la interfaz es aburrida o demasiado compleja no envolverá al jugador y el riesgo de que no llame la atención del usuario es mayor.

Se puede dividir a la interfaz gráfica en componentes de dos y tres dimensiones. La parte plana o de dos dimensiones involucra a todos los **menús** del juego, los fondos de pantalla, el **HUD**, etc; mientras que la parte de tres dimensiones se centra en el diseño de los personajes, animación de los mismos, creación de escenarios, objetos móviles e inmóviles con los que el jugador puede interactuar, entre otros. Debido a que el presente proyecto se enmarca en dos dimensiones fue de este tema que se trató el curso, aunque primero se explicó la diferencia entre ambas y se indicó algunos programas con los que los estudiantes pueden realizar la parte en tres dimensiones como 3D Studio Max, Maya, entre otros.

Una buena interfaz gráfica guarda coherencia con el objetivo del juego tratando de eliminar la barrera entre el jugador y su personaje, logrando que no se vea al juego como una realidad alterna sino como una oportunidad de explotar todo su potencial en un ambiente fantástico. Esto es muy difícil de lograr ya que hay partes de la interfaz o del HUD que no existen en el mundo real como por ejemplo el hecho de tener más de una vida o poder cambiar la dificultad del juego, lo que recuerda al jugador la diferencia entre el

mundo virtual y la realidad, aunque hasta cierto punto esta separación es necesaria para mantener la salud mental del usuario.

En cuanto al **diseño** la capacitación fue un poco más compleja, debido a que este sería el trabajo de un artista más que de un ingeniero, en las empresas dedicadas al desarrollo de videojuegos tanto comerciales como con contenido serio, se emplean equipos de diseñadores gráficos, dibujantes y desarrolladores de personajes para la creación de los mismos; arquitectos, historiadores y artistas plásticos para la creación de escenarios, etc. Tomando esto en cuenta solo se dieron pautas básicas de diseño como contraste, repetición, alineado y proximidad, estos son conceptos básicos recopilados de varias fuentes pero principalmente de la colección The Non-Designer's Books de Robin Williams (Williams, 2004).

El contraste trata de evitar el uso de elementos similares dentro de una misma página como color, letra, tamaño, forma, entre otros, debido a que pierden importancia y afectan visualmente a la experiencia haciendo que el diseño se vea muy monótono.

La uniformidad o repetición de los elementos en el diseño general también es una parte importante, toda las pantallas de la interfaz deben guardar una coherencia entre ellas y con el juego en sí; todos los elementos deben ser colocados por una razón y en lo posible se debe mantener la idea general a lo largo del juego ya que es más digerible para el usuario, una vez que entienda la primera interfaz las consiguientes serán aún más intuitivas, y genera una transición más natural.

Por la misma razón debe existir una alineación entre los elementos incluso si el diseño es "desordenado" a simple vista cada elemento debe ubicarse con una razón que lo justifique y en lo posible que se explique por si misma para que el usuario la entienda; si bien el hecho de entender el porqué de la interfaz no es primordial para el jugador, su experiencia es mejor si lo hace ya que se siente identificado con la misma y eso provoca que cree empatía con el juego.

Finalmente la proximidad entre los elementos del mismo tipo facilita la ubicación del usuario dentro de la interfaz ya que brinda una organización que no es perceptible (a menos que ese sea el objetivo), a menos que esté mal planteada ya que no es común que un usuario final felicite un diseño bien hecho porque lo utiliza con naturalidad y no aprecia el trabajo que se realizó, pero si no es el adecuado será molesto cada vez que necesite utilizar alguna función.

El diseño más complejo de realizar es el que da como resultado mayor simplicidad. Se deben analizar muchos puntos de vista para llegar a un resultado favorable para los usuarios finales a quienes esté dirigido el diseño.

Otro punto de la capacitación fue la explicación y creación de los diferentes **tipos de archivos** necesarios para vincular las interfaces creadas dentro del sistema, para lo cual se creó una interfaz simple y se crearon los archivos *.imageset, *.layout, *.font y *.scheme. Como ya se explicó anteriormente cada imagen que vaya a ser incluida en el sistema debe tener un archivo tipo *.imageset llamado igual que la imagen que contenga la información de la misma en XML para que el sistema pueda utilizarla. De igual forma con los archivos *.layout. Los archivos *.font por otro lado solo deben ser cerados si se desea implementar un tipo de letra diferente al que viene configurado por defecto, y finalmente en el archivo *.scheme se deben listar todos los archivos creados, tiene la funcionalidad de un directorio para el aplicativo donde se lista la dirección y el nombre de todos los archivos que se deban cargar al momento de ejecutarlo.

Si bien es cierto estos archivos pueden ser creados en línea de código, existen algunas herramientas personalizadas para realizar este proceso y en el curso se demostró la creación de los archivos de ambas formas, desde "Sublime Text[38]" para la creación de los documentos en línea de código y "**CELayout Editor**[39] como herramienta visual, el resultado final es el mismo, la diferencia está en la preferencia de trabajo del desarrollador.

Por último en la **implementación** se explicó la sintaxis para incorporar las interfaces que se diseñaron y se codificaron con la información necesaria, dentro del sistema final.

Antes del inicio del curso se preparó las máquinas del laboratorio con todas las herramientas y configuraciones necesarias con el fin de no desperdiciar tiempo en dicho proceso con los alumnos. Refiérase al anexo 1 para una explicación detallada de dicho proceso.

Finalmente se evaluó a los alumnos para reforzar los conocimientos adquiridos en el curso tanto en la parte de codificación dictada por el Profesor Prometeo como la parte de interfaz, los parámetros de evaluación fueron propuestos por el Profesor Prometeo como tutor de la materia.

[38] Sublime Text: http://www.sublimetext.com/
[39] CELayout Editor: http://cegui.org.uk/wiki/CELayout_Editor_-_Getting_Started_/_Building

CONCLUSIONES

- ✓ Los juegos con contenido serio pueden ser explotados en infinidad de campos educativos y gracias al proyecto Gists-UTPL la Universidad tiene un punto de partida sólido y funcional en el cual basarse para los trabajos futuros, ya sean sobre el mismo sistema o reutilizando código para implementarlo utilizando otra de las herramientas más comerciales descritas en el estado del arte o herramientas nuevas que pueda surgir.
- ✓ El motor gráfico que posee Delta3D no es lo suficientemente potente para crear un juego que atraiga la atención de los Gamers o del publico adolecente en general por lo que se sugiere un estudio más profundo de las herramientas existentes en donde se realicen prototipos simples del mismo juego con las diferentes herramientas para poder decidir con un criterio más concreto cuál es la mejor para el trabajo.
- ✓ Si bien es cierto la interfaz gráfica de usuario es una parte fundamental para un video juego con contenido serio, la temática del juego en sí es igual o incluso más importante, por lo que se le debe dedicar el debido análisis para garantizar la efectividad del juego.
- ✓ El equipo de trabajo involucrado en la creación de un juego con contenido serio involucra profesionales de diferentes ámbitos trabajando en equipo para lograr un sistema final de buena calidad.
- ✓ Durante la investigación se encontraron diversas aplicaciones de este tipo alrededor del mundo demostrando el auge que estos temas están teniendo en la actualidad, tanto en la educación formal como colegios y universidades, informal como organismos independientes que presentan proyectos abiertos en forma de juegos, e incluso en el campo laboral de ciertas ramas como la medicina y la economía.

RECOMENDACIONES Y TRABAJOS FUTUROS

- ✓ Realizar una prueba piloto con el Gists-UTPL para validar el juego y medir la aceptación de los usuarios y la implicación en su aprendizaje.

- ✓ Análisis de la posibilidad de migrar el sistema a otra plataforma como Unity, o un lenguaje más visual como Java, para darle continuidad al proyecto.

- ✓ Reutilización del código fuente del Gists-UTPL para la implementación de más juegos con contenido serio en otros ámbitos educativos.

BIBLIOGRAFÍA

Abt, C. (1987). *Serious games* (p. 176). Retrieved from http://books.google.com/books?hl=en&lr=&id=axUs9HA-hF8C&oi=fnd&pg=PR13&dq=Serious+Games&ots=dYY19kw5yT&sig=rkAjtnDKpQP_JPOueiWL4MXcR9g

Arnaba, S., Brownb, K., Clarkea, S., Dunwella, I., Limc, T., Suttiec, N., … de Freitas, S. (2013). *The development approach of a pedagogically-driven serious game to support Relationship and Sex Education (RSE) within a classroom setting* (p. 16). Retrieved from http://www.sciencedirect.com/science/article/pii/S0360131513001644?np=y

BreakAway Ltd. (2012). Serious Games by BreakAway. Retrieved July 2, 2014, from http://www.breakawaygames.com/serious-games/overview/

Canet, M. (2012). Innovación en interfaces para videojuegos desde el Game Art, 135–149. Retrieved from http://www.injuve.es/sites/default/files/2012/46/publicaciones/Revista98_10.pdf

Coonan. (2006). GM Tutorial Parts 1 and 2 - delta3d. Retrieved June 28, 2014, from http://delta3d.org/article.php?story=20060620123144266

Delta3D. (2004). Delta3D. Retrieved February 21, 2014, from http://delta3d.org/index.php?topic=about

Delta3D. (2009). Projects - delta3d. Retrieved March 5, 2015, from http://delta3d.org/index.php?topic=projects

Looi, C., McCalla, G., Bredeweg, B., & Breuker, J. (2005). *Artificial Intelligence in Education* (p. 1012). Retrieved from http://books.google.com.ec/books?id=bR1pKYNcHA4C&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

Maureen, R. (2011). TRADOC-sponsored simulation wins Serious Games Challenge. Retrieved February 10, 2014, from http://www.tradocnews.org/

McCloud, S. (1994). Understanding Comics (The Invisible Art). Retrieved from https://iteachu.uaf.edu/files/2013/10/McCloud-Understanding-Comics.pdf

McGonigal, J. (2010). *Gaming can make a better world*. Retrieved from http://www.ted.com/talks/jane_mcgonigal_gaming_can_make_a_better_world.html

Pastorini, A., & Martínez, A. (2014). Interfaz para video juegos. Retrieved from http://www.fing.edu.uy/tecnoinf/mvd/cursos/vj2d/material/vj2d-clase04-Interfaz.pdf

Peinado, F. (2007). Diseño – Interfaz. Retrieved from http://web.fdi.ucm.es/profesor/fpeinado/courses/gamedesign/gamedesign_interfaces_es.pdf

Pérez Pérez, S., & Pérez Domingo, N. (2014). Los colores en feng shui. Retrieved November 23, 2014, from http://www.fengshuinatural.com/colores.html

Ritterfeld, U., Cody, M., & Vorderer, P. (2009). *Serious Games Mechanisms and Effects* (p. 552). Retrieved from

http://books.google.com.ec/books?hl=en&lr=&id=eGORAgAAQBAJ&oi=fnd&pg=PP1&dq=serious+games&ots=1tapm9RgVy&sig=zrPiyxZwHAIZXZLV-5jyz5HXEho#v=onepage&q=serious games&f=false

Rogers, S. (2010). *Level Up! Guide to great Video Game Design* (p. 492). Retrieved from https://books.google.com.ec/books?id=09VFAwAAQBAJ&pg=PA438&lpg=PA438&dq=Level+Up!+Guide+to+Great+Video+Game+Design&source=bl&ots=6KFan6ed4p&sig=lTkxIGn-rTUvLu8mNyVmgHbnUYE&hl=es-419&sa=X&ved=0CFEQ6AEwBmoVChMI6fqxtqjoxwIVgtWACh0Qbgi-#v=onepage&q=Level Up! Guide to Great Video Game Design&f=false

Secretaría de Educación Superior de Ciencia Tecnología e Innovación. (2015). ¿Qué es Prometeo? / What is Prometeo? | Prometeo. Retrieved February 27, 2015, from http://prometeo.educacionsuperior.gob.ec/que-es-prometeo/

TechTerms.com. (2015). GUI (Graphical User Interface) Definition. Retrieved May 29, 2014, from http://www.techterms.com/definition/gui

Turner, P. (2014). Cegui Wiki - Crazy Eddie's Gui System for Games (Open Source). Retrieved June 27, 2014, from http://cegui.org.uk/wiki/FAQ#What_is_CEGUI.3F

Turner, P., & Team, T. C. D. (2014). CEGUI. Retrieved June 27, 2014, from http://cegui.org.uk/features

Unity Technologies. (2015a). Unity - Game engine, tools and multiplatform. Retrieved February 27, 2015, from https://unity3d.com/es/unity

Unity Technologies. (2015b). Unity - SIM. Retrieved February 27, 2015, from http://unity3d.com/industries/sim

Uzquiano, I., Fernández, S., & Ortiz, J. (2010). Acceso manual e interfaz gráfica para el juego AI-LIVE, 105. Retrieved from http://e-archivo.uc3m.es/handle/10016/10531

Wikipedia. (2013). Sodipodi. Retrieved March 5, 2015, from https://es.wikipedia.org/wiki/Sodipodi

Wikipedia. (2014a). Gamer. Retrieved February 25, 2015, from https://es.wikipedia.org/wiki/Jugador_de_videojuegos

Wikipedia. (2014b). Login. Retrieved February 20, 2015, from http://www.alegsa.com.ar/Dic/login.php

Williams, R. (2004). *The Non-Designer's Design Book* (Second, p. 194). Retrieved from http://www.indusvalley.edu.pk/library/e books/The non-designers design book.pdf

ANEXOS

Anexo 1: Instalación y configuración de herramientas para el curso "Planeación y diseño de juegos formativos – Serious Games"

Como parte del soporte que brindé durante el curso, realicé las configuraciones necesarias para trabajar con el motor de simulación utilizado en el desarrollo del sistema, (Delta3D) en las computadoras del laboratorio asignado para el curso, mismas que se detallan a continuación para futuros trabajos:

1) Descargar los componentes de Delta3D del link presentado a continuación, http://delta3d.org/index.php?topic=downloads
2) Crear una carpeta en la raíz del disco C:\ y copiar todos los componentes descargados dentro de la misma; descomprimir la información de ser necesario creando una estructura similar a la siguiente:

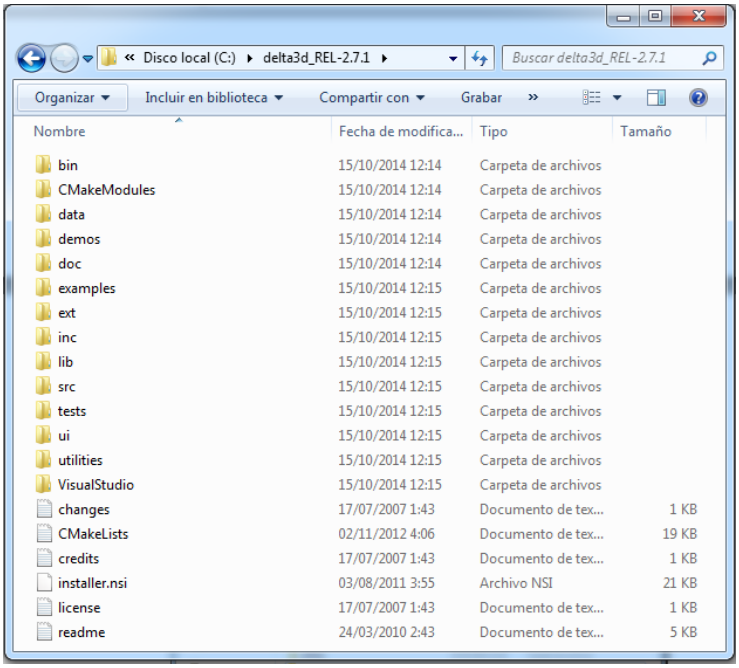


Imagen 9, Captura de pantalla de los archivos descargados desde delta3d

3) A continuación se deben crear las siguientes variables de entorno; para esto entrar en Propiedades del sistema, Configuración avanzada del sistema, Opciones avanzadas, Variables de entorno:
    a. DELTA_ROOT=C:\delta3d_REL-2.7.1 (Este es el directorio donde se copiaron los archivos descargados)
    b. DELTA_INC = %DELTA_ROOT%\inc;%DELTA_ROOT%\ext\inc
    c. DELTA_LIB = %DELTA_ROOT%\lib;%DELTA_ROOT%\ext\lib

d. DELTA_DATA = %DELTA_ROOT%\data

e. PATH = %DELTA_ROOT%\bin;%DELTA_ROOT%\ext\bin

4) Finalmente se importa desde visual basic los archivos de la carpeta creada anteriormente.

a. Click en la pestalla de herramientas y seleccionar opciones.

b. Pestaña proyectos y soluciones, opción Directorios VC++ y agregar librerías y archivos de inclusión:

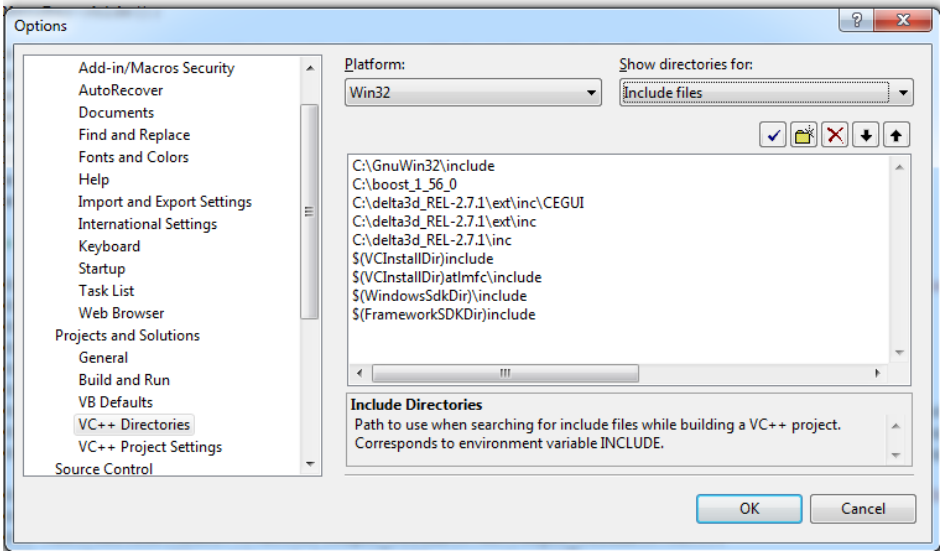


Imagen 10, Captura de pantalla de la configuración de Microsoft Visual Studio

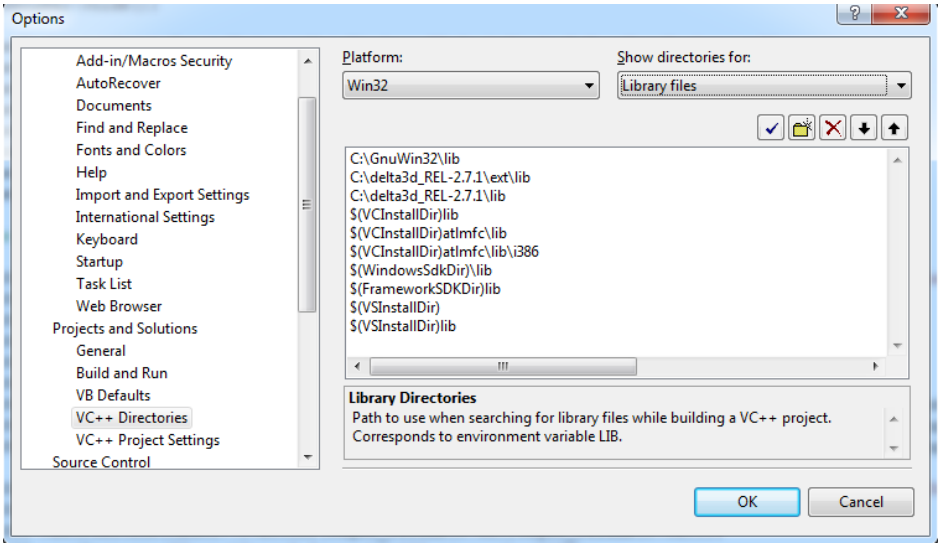


Imagen 11, Captura de pantalla de la configuración de Microsoft Visual Studio

Se realizaron estas configuraciones en veinte computadoras para la realización del curso, además se descargaron los programas necesarios para la implementación de las interfaces descritos en el presente documento. Por falta de tiempo no se incluyó en la capacitación el diseño de interfaces en los programas de adobe, simplemente se dieron pautas generales de cómo diseñar las interfaces.

Anexo 2: Evidencia del curso impartido.

Imagen 12, Curso diseño de juegos con un contenido serio

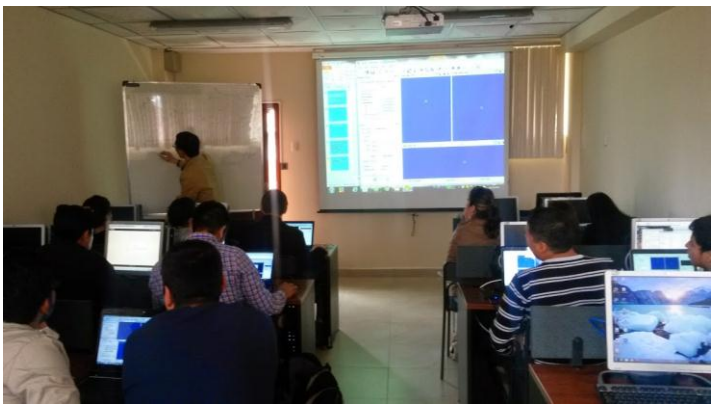


Imagen 13, Curso diseño de juegos con un contenido serio

Imagen 14, Curso diseño de juegos con un contenido serio

Anexo 3: Capturas de la interfaz

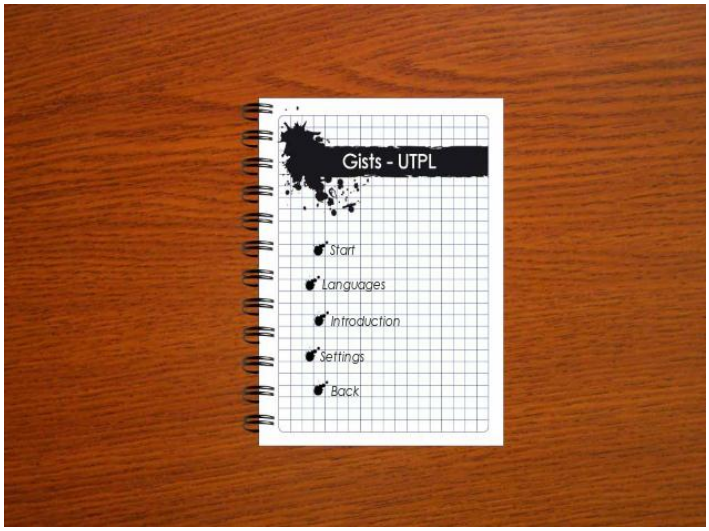


Imagen 15, Menú Principal

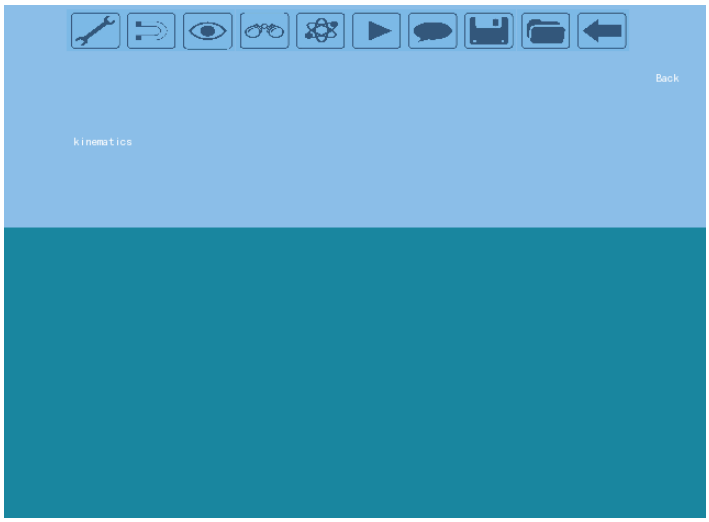


Imagen 16, HUD

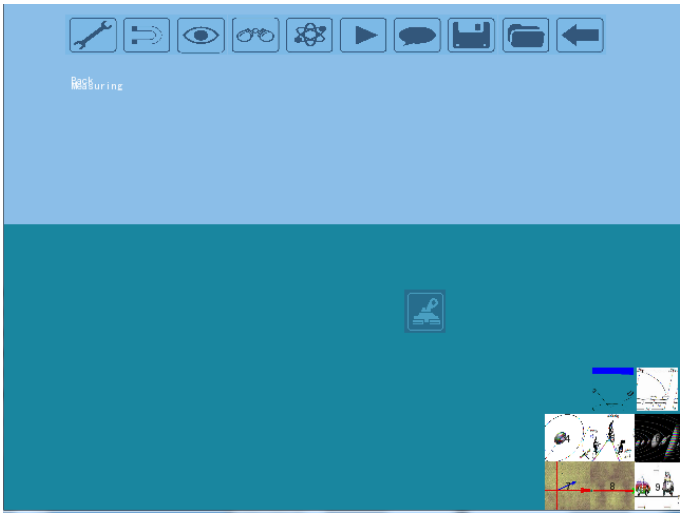


Imagen 17, HUD 3

Anexo 4: Artículo resultado del proyecto de titulación

Diseño de la interfaz gráfica de usuario para el Gists-UTPL

Design of graphical user interface for Gists-UTPL

Resumen — **En el presente documento trata sobre el diseño e implementación de una interfaz gráfica para un juego con contenido serio orientado a la enseñanza de la física a nivel de educación básica o secundaria llamado Gists – UTPL por sus siglas en inglés (3D Game-Based Intelligent Science Tutoring System in UTPL). El documento inicia con la historia de los videojuegos con contenido serio, incluso antes de que se los conociera de esa forma, el diseño de interfaces para videojuegos, el uso básico de algunas herramientas necesarias para el trabajo, la forma de acoplar los diseños creados al sistema utilizando determinadas herramientas.**

Palabras Clave – Gists-UTPL, juegos serios, videojuegos, enseñanza, inteligencia artificial, diseño, interfaz.

Abstract — **This paper discusses on the design and implementation of a GUI for a game with serious content, oriented to teach physics for primary and secondary education, called Gists-UTPL for its acronym (3D Game-Based Intelligent Science Tutoring System in UTPL). The document begins with the history of video games with serious content, even before they knew them that way, the design of interfaces for video games, the basic use of some tools needed for the job, the task of matching designs created to system using certain tools.**

Keywords - Gists-UTPL, serious games, video games, teaching, artificial intelligence, design, GUI.

I. INTRODUCCIÓN

En la actualidad la importancia de incorporar técnicas y recursos académicos en las instituciones educativas que permitan a los jóvenes incrementar y fortalecer su aprendizaje es cada vez más necesario, mientras las nuevas tecnologías siguen aumentando y convirtiéndose en parte de nuestra vida cotidiana, esto hace necesaria la inclusión de las tecnologías de información de una forma razonable, paulatinamente la sociedad está perdiendo el deseo de obtener conocimientos por todas las comodidades que la tecnología le brinda, por lo que es necesario el uso útil de dichas tecnologías. Esto hace que, sobre todo los jóvenes, empleen su tiempo en otras actividades aparentemente más enriquecedoras como los videojuegos. En la charla en TED de Jane McGonigal titulada "Los juegos online pueden crear un mundo mejor", se expone un dato curioso. "Un Gamer promedio habrá invertido 10000 horas jugando videojuegos en línea para cuando tenga 21 años de edad, mientras que 10080 horas es la cantidad de tiempo que un estudiante invierte desde quinto grado de escuela hasta su graduación del colegio si nunca falta a clase"(McGonigal, 2010). Con esto se demuestra fácilmente la magnitud del tiempo que los jóvenes gastan, si cabe el término, en esta actividad. Pero, qué se lograría si el tiempo de relajación se fusionara con el estudio, y más importante aún, si toda la energía y adrenalina que se aplica en los juegos se re direcciona al estudio. Tanto la cantidad como la calidad de profesionales que dispondría el mundo en el futuro no tendría precedentes, de ahí se originó la idea de los juegos con contenido serio, la fusión de un juego que brinde la adrenalina y el interés de los videojuegos comerciales, con un bagaje de conocimientos útiles para la vida cotidiana, que los jugadores aprendan mientras se divierten y de esa forma el estudio no sea una actividad tediosa sino enriquecedora en todo sentido

II. MARCO TEÓRICO

A. Reseña histórica sobre los videojuegos

Si bien es cierto, la idea de adoptar los videojuegos como forma de enseñanza en un aula de clase es bastante novedosa, los videojuegos con un

contenido útil no lo son. En 1985 se lanzó al mercado ¿Dónde está Carmen Sandiego en el mundo?[40] , para el Apple II, y más tarde fue portado al resto de sistemas. El objetivo del juego es perseguir a Carmen y a sus secuaces por todo el mundo y arrestarlos, para lograrlo se necesita una orden de arresto dirigida al criminal que haya cometido el delito que el jugador está resolviendo. Cada nivel comienza con un crimen distinto y el jugador se dirige a la ciudad donde ocurrió para obtener pistas que le ayuden a descubrir la identidad del criminal y la siguiente ciudad a la que este irá para seguirlo. El jugador tiene un tiempo límite para arrestar al criminal y antes de que se cumpla debe haber conseguido suficientes pistas de su identidad para solicitar la orden de arresto en su contra, cuando encuentra al villano debe arrestarlo y el juego termina cuando toda la banda haya sido arrestada, incluida Carmen Sandiego.

Este juego enseña geografía y algo de historia sobre algunas de las ciudades más importantes del mundo, ya que las pistas que se obtienen son datos como la moneda o los colores de la bandera del país o alguna reseña en particular sobre el lugar al que se dirigió.

El juego tuvo tanto éxito que se hicieron algunas secuelas del mismo como ¿dónde está Carmen Sandiego en EEUU?, en Europa e incluso a través del tiempo y en el espacio; incrementando en cada entrega el conocimiento que el jugador adquiría incluyendo matemáticas y hasta algo de astronomía.

En 1994 Sierra Entertaiment presentó “The lost mind of Dr. Brain”[41], el juego comienza con un experimento del Dr. Thaddeus “Puzzles” Brain, maestro de los rompecabezas, que intenta compartir su conocimiento con un ratón de laboratorio pero no funciona como él esperaba y su cerebro queda en el cuerpo del ratón mientras su cuerpo queda en algo parecido a un estado vegetal, el objetivo del juego es reparar todo el daño en el cerebro del doctor para que vuelva a su cuerpo. En cada parte del cerebro hay un tipo diferente de juego con tres niveles de dificultad, y aunque cada mini juego es distinto, la mayoría se basa en lógica.

B. Juegos Serios

El instituto de Juegos Serios SGI define a los mismos como: Aplicaciones desarrolladas utilizando tecnologías de juegos de video que sirvan a otros propósitos además del puro entretenimiento”(Arnaba et al., 2013).

Los juegos serios pueden ser aplicados en muchos campos y brindan un sinnúmero de posibilidades para la sociedad, algunos autores han reconocido en sus libros el valor de éstos en la educación; a continuación se citan textualmente sus opiniones:

“Se ha reconocido que los juegos pueden ser un poderoso vehículo para el aprendizaje, y la inteligencia artificial puede ampliar los resultados de aprendizaje con juegos”(Looi et al., 2005).

En el libro de Abt Clark Serious Games, se menciona que “(…) planificando, jugando y analizando los juegos educativos motivan a los alumnos a estudiar los problemas dramatizados en el juego y la literatura que envuelve los problemas para sintetizar soluciones del problema y evaluar críticamente posibles soluciones que desarrollaron en su investigación”(Abt, 1987)

C. Interfacez gráficas de usuario

Según TechTerms.com las interfaces gráficas de usuario o “GUI por sus siglas en inglés (Graphical User Interface), se refieren a la interfaz gráfica de una computadora que permite a los usuarios hacer clic y arrastrar los objetos con un ratón en lugar de introducir texto en una línea de comandos. Dos de los sistemas operativos más populares, Windows y Mac OS, están basados en GUI. La interfaz gráfica de usuario se introdujo por primera vez al público por parte de Apple con el Macintosh en 1984. Sin embargo, la idea fue en realidad tomada de una

[40] ¿Dónde está Carmen Sandiego en el mundo?: http://www.carmensandiego.com/hmh/site/carmen/
[41] The lost mind of Dr. Brain: http://www.sierragamers.com/aspx/m/649091

interfaz de usuario anterior desarrollado por Xerox."(TechTerms.com, 2015)

En los juegos de video la interfaz gráfica es el enganche inicial con el usuario, ya que ésta debe ser amigable e interesante para que el jugador se sienta ansioso por jugar. Mientras mayor sea la relación entre la interfaz y el tema del juego mejor será el resultado, ya que el usuario se siente más inmerso en la experiencia y se pierde la barrera de la realidad.

La interfaz tiene algunas partes importantes pero tal vez la más importante sea el HUD, del inglés heads - up display. "El HUD es la forma más eficaz de comunicarse con el jugador. El HUD se refiere a cualquier elemento visual que comunica información al jugador"(Rogers, 2010).

La importancia del HUD recae en el hecho de que el usuario pasa la mayor cantidad de tiempo de juego en contacto con el mismo, ya que transmite la información necesaria para el jugador durante el juego. Según Scott Rogers en su libro Level Up! Guide to great Video Game Design, existen siete elementos básicos en el HUD de un video juego promedio, las siguientes se enumeran a continuación y se describen brevemente.

- Barra de salud/vidas restantes.

Indica el estado del personaje, sus vidas restantes y/o su salud en un momento dado.

- Mira.

La mira ayuda al jugador a orientar sus disparos y saber hacia dónde está viendo su personaje.

Puede ser un punto, una cruz, un láser, una x, un círculo, etc. No hay ninguna limitante más que la imaginación del equipo de diseño.

- Munición.

Indica al jugador cuantas balas le quedan en el arma que está usando, puede también mostrar los cartuchos restantes y cualquier otra información de este tipo que pueda ser útil dependiendo del arma.

- Inventario.

El inventario muestra todo lo que el personaje lleva consigo; Armas, mapas, llaves, objetos encontrados, protecciones, dinero, etc.

El inventario suele tener un espacio limitado para aumentar la dificultad del juego ya que esto no permite llevar todo lo que uno desee, sino que obliga al jugador a sacrificar algún ítem para llevar otro.

- Puntaje/experiencia.

Indica el puntaje que el jugador a acumulado hasta un punto determinado, o la experiencia que ha ido adquiriendo en su viaje a través del videojuego.

- Radar/ mapa.

Ayuda al jugador a moverse por el mundo virtual. Algunos videojuegos tienen un "mundo" bastante amplio, o complejo por lo que el jugador suele requerir de un mapa para no perderse en el recorrido.

- Aviso de interacción.

El aviso de interacción informa al jugador que puede interactuar con un objeto cercano. Un aviso común es un pequeño dibujo de la tecla o el botón con el que realiza la acción sobre el objeto. (Rogers, 2010)

Si bien es cierto estos son los elementos básicos de un HUD, para nuestro propósito en particular tal vez no sea necesario involucrarlos a todos, ya que el objetivo de Gists – UTPL no solo es divertir al jugador sino también enseñarle algo, por otro lado es posible que se requieran otros elementos para ayudar este propósito, por ejemplo un chat para realizar preguntas al profesor. Esto se determinará en el transcurso del proyecto.

Los íconos son quizás los pequeños detalles que hacen que la interfaz tenga una apariencia más involucrada con la historia, por ejemplo en un juego de asesinos los íconos pueden simular armas o disparos, a diferencia de un juego de carreras donde pueden parecer partes de un auto. Pero, qué es un ícono? "Un ícono puede ser cualquier

imagen usada para representar a una persona, un lugar, una cosa o una idea."(McCloud, 1994)

Los íconos deben ser simples para que cualquier persona que los vea le dé un mismo significado. Un ícono debe ser abstracto pero no demasiado general puesto que la idea básica se puede perder.

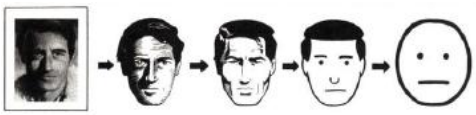


Imagen 18, Abstracción de una imagen, recuperada de "Understanding Comics" (McCloud, 1994)

Como se puede ver en la imagen obtenida del libro "Entendiendo los comics" de Soctt McCloud, las imágenes pueden abstraerse hasta su punto más simple siempre y cuando no pierdan el significado de lo que representan.

Otra parte importante del diseño es el lugar donde se ubican los íconos o los menús que los contengan. Se deben ubicar de tal forma que no se conviertan en una molestia para el jugador reduciendo demasiado su campo de visión o llamando demasiado su atención haciendo que se distraiga del juego. El jugador debe saber que están ahí y debe poder acceder a ellos con facilidad sin mayor esfuerzo; Mientras más tiempo pase en el juego más natural se hará el acceso a los menús por lo que lo más conveniente puede ser ubicar dichos accesos en los lugares que suelen ser más comunes como el ícono para salir en la esquina superior derecha por ejemplo.

Imagen 19, Ubicación de menús, recuperada de "Level Up! Guide to great Video Game Design"(Rogers, 2010)

Es importante recordar que aunque la pantalla de juego sea la más importante, ya que el jugador pasa casi todo el tiempo en ella, no es la única interfaz que existe. Cada menú y pantalla del juego requieren de una igual preparación, estudio y diseño para realizar una interfaz completa que cumpla con su objetivo a cabalidad que, como ya se dijo, es el de involucrar al usuario en la historia para que se sienta en ella.

D. Campos de aplicación de Juegos Serios

En la actualidad existen algunas empresas que trabajan en favor de la creación y aplicación de los juegos serios en varios campos, principalmente educativos. A continuación se exponen algunos ejemplos, algunos de los cuales se hace mucha referencia en el presente documento:

- Salud

"Pulse" o pulso en español es uno de los más notables lanzado al mercado en el año 2007 desarrollado por la "Universidad de Texas A&M Corpus-Christi"[42], es un juego en 3 dimensiones que enseña técnicas para realizar diagnósticos médicos en estados de emergencia. Este juego involucra al jugador, tanto civil como militar, en

[42] Texas A&M University Corpus Christi: http://www.tamucc.edu/

situaciones extremas basadas en casos reales para probar su capacidad de resolver dichas crisis sin necesidad de poner vidas en riesgo, lo que sirve como una excelente práctica para pulir sus habilidades clínicas.

"Dental Implant Training Simulation[43]" o simulador de entrenamiento para implantes dentales fue diseñado para que los estudiantes de odontología puedan realizar prácticas en un ambiente seguro pero realista.

- Enseñanza

El juego matemático "ST Math[44]" o "ST (JiJi) Math[45]" es un juego para nivel primario y secundario que enseña matemáticas mientras el jugador ayuda a un pingüino a cruzar de un lado al otro de la pantalla, disponible también en versión móvil. Se originó de la idea que no es necesario el uso de palabras para la enseñanza, destruyendo así las barreras idiomáticas y haciendo de la obtención de conocimiento una labor completamente intuitiva.

Los niveles básicos del juego, pensados para niveles primarios, forman las bases del pensamiento lógico – matemático en el jugador, explicando las operaciones básicas de una forma visual e interactiva para que el usuario entienda el porqué de cada respuesta; mientras que los niveles más avanzados engloban todo el conocimiento que se imparte en el nivel secundario, llegando incluso a explicar la integración y derivación de factores, sin perder su modalidad de juego, por lo que los usuarios sin importar su edad o lengua, pueden sacarle provecho.

[43] Dental Implant Training Simulation: http://www.breakawaygames.com/serious-games/solutions/healthcare/
[44] ST Math: http://www.stmath.com/
[45] ST (JiJi) Math: https://play.google.com/store/apps/details?id=air.net.mindresearch.stmath&hl=es

III. ANALISIS COMPARATIVO DE LAS HERRAMIENTAS.

- **Motores de juego**

Herramienta	Pros	Contras
Unity	• Código abierto (Condicional) • Herramientas para diseño de alta calidad.	Conocimiento nulo de la herramienta.
Delta3D	• Código abierto. • Compatibilidad con Visual Studio. • Experiencia del profesor Prometeo en el uso del programa.	Restricción en cuanto a imágenes de alta calidad.

- **Herramientas de diseño**

Herramienta	Pros	Contras
Sodipodi	• Intuitivo y sencillo.	• Software descontinuado. • Formatos limitados.
Inkscape	• Modelado 3D.	• Formatos limitados.
Adobe (Illustrator/Photoshop)	• Gran cantidad de tutoriales. • Diseños en alta definición. • Compatibilidad con muchos formátos.	• Licencia pagada.

- **Implementación de diseños**

Herramienta	Pros	Contras
CELayout Editor	• Muy estable. • Simple.	• Incompatible con imágenes

		de alta calidad.
CEED	• Intuitivo y sencillo. • Compatible con imágenes de alta calidad.	• Versión inestable.

A. Selección de herramientas

Para la etapa de diseño se utilizará el paquete de adobe debido a la compatibilidad que poseen sus programas entre sí, lo que facilitaría la posible adición posterior de animaciones de transición y audios en el juego.

Para el desarrollo de las interfaces previa implementación se utilizará CEED debido a la mejor compatibilidad que posee con archivos de alta calidad, y en caso de tener algún problema al momento de exportar alguno de los diseños a la plataforma de implementación se recurrirá a CELayoutEditor.

Para la implementación y vinculación de las interfaces con el sistema Gists – UTPL se utilizará Delta3D debido a que el sistema está desarrollado sobre ésta plataforma y eso facilitará la vinculación de las funcionalidades.

IV. DISEÑO.

A. Diseño de la interfaz

Para el diseño de la interfaz gráfica del juego se analizó los mini juegos que involucra el sistema para encontrar un factor común que los relacione y trabajar las ventanas con dicho factor. Si solo se tomaba en cuenta uno de los juegos, por ejemplo el de lanzamiento de proyectil con el tanque de guerra se hubiera diseñado una interfaz con un ambiente militar, siendo por ejemplo el ícono del chat un radio militar, pero debido a la variedad de ámbitos involucrados en los juegos se decidió que el diseño debía ser algo más neutral y ya que el objetivo del sistema es ejercitar lo aprendido en clase, se diseñaron interfaces relacionadas a la educación.

La ventana inicial es un cuaderno con un diseño similar a los de la UTPL que está sobre un escritorio de madera con el nombre del juego “GISTS” en la parte inferior derecha de la pasta, la cruz de la universidad como símbolo de la misma en el centro y el nombre de la logo de la universidad junto con su escudo en la parte superior derecha.

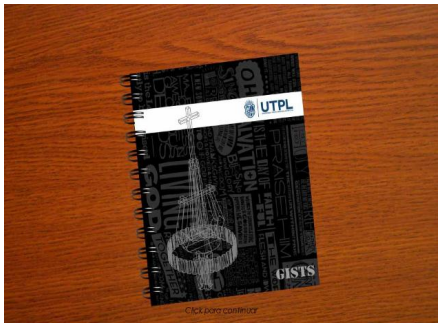


Imagen 20, Pantalla inicial

En la siguiente ventana el jugador debe ingresar al sistema con un usuario y contraseña registrado en el sistema y la dirección de su computador para poder ingresar a la red; en esta ventana el cuaderno está abierto y simula los datos personales del estudiante.

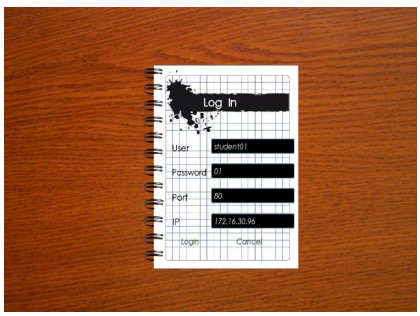


Imagen 21, Pantalla de Logeo

Una vez logueado el usuario puede:

- Comenzar a jugar.
- Escoger el idioma del juego.
- Leer las instrucciones del juego.
- Cambiar algunas opciones del juego.
- Volver a la ventana anterior.

A continuación se muestra una captura del entorno de juego con un ejercicio seleccionado.

Imagen 22, Entorno de juego, fuente proyecto Prometeo

Como ya se explicó anteriormente todos estos menús forman parte del “HUD” y se localizaron en la pantalla tal como se recomienda en “Level Up! Guide to great Video Game Design” (Rogers, 2010), están ubicados en los alrededores de la pantalla para dejar libre el lugar de juego y no molestar al estudiante.

B. Análisis de colores para la interfaz

Los colores utilizados en los diseños también fueron analizados, según estudios los colores crema o pasteles fomentan el aprendizaje en los estudiantes, por lo algunas instituciones educativas como la UTPL utilizan dichos colores en sus edificios y aulas. La misma teoría se aplicó para el color de las interfaces del sistema tanto en los menús como en los íconos y el fondo del entorno 3D.

Un gran limitante para los colores del HUD fue el motor gráfico con el que trabaja el sistema, ya que no posee una paleta muy amplia y el color del piso donde corren las animaciones no se puede editar lo que obligó a que los colores sean bajo esa gama.

Pese a esto se tomó en cuenta la teoría de colores del Feng shui que se basa en los cinco elementos básicos y los colores que representan a cada uno de ellos para combinarlos de mejor manera y obtener un ambiente equilibrado.(Pérez Pérez & Pérez Domingo, 2014) Los elementos y sus colores son:

- Fuego: Rojo, anaranjado, rosado; colores cálidos.
- Tierra: Marrón, beige, amarillo.
- Metal: Dorado, plateado, blanco.
- Agua: Negro, azul, turquesa.
- Madera: Verde.

C. Diseño de íconos

En el diseño de los íconos se trató de sintetizar la funcionalidad de cada botón con un diseño general para darle armonía al HUD sin dejar de lado el significado individual de cada uno de forma tal que el usuario final sepa lo que significan sin necesidad de un título.

En las funciones clásicas se dibujaron íconos similares a la mayoría de los programas comerciales, a continuación dichos íconos en su versión normal y con sus colores invertidos, utilizados en el sistema para dar la impresión de que se aplasta un botón.

V. CONCLUSIONES

- Los juegos con contenido serio pueden ser explotados en infinidad de campos educativos y gracias al Gists-UTPL la universidad tiene un punto de partida sólido y funcional en el cual basarse para los trabajos futuros, ya sean sobre el mismo sistema o reutilizando código para implementarlo utilizando otra de las herramientas más comerciales descritas en el estado del arte o herramientas nuevas que pueda surgir.
- El motor gráfico que posee el Delta3D no es lo suficientemente potente para crear un juego que atraiga la atención de los Gamers o del publico adolecente en general por lo que sugiero un estudio más profundo de las herramientas existentes en donde se realicen prototipos simples del mismo juego con las diferentes herramientas para poder decidir con un criterio más concreto cuál es la mejor para el trabajo.

- Si bien es cierto la interfaz gráfica de usuario es una parte fundamental para un video juego con contenido serio, la temática del juego en sí es igual o incluso más importante, por lo que se le debe dedicar el debido análisis para garantizar la efectividad del juego.
- El equipo de trabajo involucrado en la creación de un juego con contenido serio consta de profesionales de diferentes ámbitos trabajando en equipo para lograr un sistema final de buena calidad

.

VI. REFERENCIAS BIBLIOGRÁFICA

Abt, C. (1987). *Serious games* (p. 176). Retrieved from http://books.google.com/books?hl=en&lr=&id=axUs9HA-hF8C&oi=fnd&pg=PR13&dq=Serious+Games&ots=dYY19kw5yT&sig=rkAjtnDKpQP_JPOueiWL4MXcR9g

Arnaba, S., Brownb, K., Clarkea, S., Dunwella, I., Limc, T., Suttiec, N., … de Freitas, S. (2013). *The development approach of a pedagogically-driven serious game to support Relationship and Sex Education (RSE) within a classroom setting* (p. 16). Retrieved from http://www.sciencedirect.com/science/article/pii/S0360131513001644?np=y

BreakAway Ltd. (2012). Serious Games by BreakAway. Retrieved July 2, 2014, from http://www.breakawaygames.com/serious-games/overview/

Canet, M. (2012). Innovación en interfaces para videojuegos desde el Game Art, 135–149. Retrieved from http://www.injuve.es/sites/default/files/2012/46/publicaciones/Revista98_10.pdf

Coonan. (2006). GM Tutorial Parts 1 and 2 - delta3d. Retrieved June 28, 2014, from http://delta3d.org/article.php?story=20060620123144266

Delta3D. (2004). Delta3D. Retrieved February 21, 2014, from http://delta3d.org/index.php?topic=about

Delta3D. (2009). Projects - delta3d. Retrieved March 5, 2015, from http://delta3d.org/index.php?topic=projects

Looi, C., McCalla, G., Bredeweg, B., & Breuker, J. (2005). *Artificial Intelligence in Education* (p. 1012). Retrieved from http://books.google.com.ec/books?id=bR1pKYNcHA4C&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

Maureen, R. (2011). TRADOC-sponsored simulation wins Serious Games Challenge. Retrieved February 10, 2014, from http://www.tradocnews.org/

McCloud, S. (1994). Understanding Comics (The Invisible Art). Retrieved from https://iteachu.uaf.edu/files/2013/10/McCloud-Understanding-Comics.pdf

McGonigal, J. (2010). *Gaming can make a better world*. Retrieved from http://www.ted.com/talks/jane_mcgonigal_gaming_can_make_a_better_world.html

Pastorini, A., & Martínez, A. (2014). Interfaz para video juegos. Retrieved from http://www.fing.edu.uy/tecnoinf/mvd/cursos/vj2d/material/vj2d-clase04-Interfaz.pdf

Peinado, F. (2007). Diseño – Interfaz. Retrieved from http://web.fdi.ucm.es/profesor/fpeinado/courses/gamedesign/gamedesign_interfaces_es.pdf

Pérez Pérez, S., & Pérez Domingo, N. (2014). Los colores en feng shui. Retrieved November 23, 2014, from http://www.fengshuinatural.com/colores.html

Ritterfeld, U., Cody, M., & Vorderer, P. (2009). *Serious Games Mechanisms and Effects* (p. 552). Retrieved from http://books.google.com.ec/books?hl=en&lr=&id=eGORAgAAQBAJ&oi=fnd&pg=PP1&dq=serious+games&ots=1tapm9RgVy&sig=zrPiyxZwHAIZXZLV-5jyz5HXEho#v=onepage&q=serious games&f=false

Rogers, S. (2010). *Level Up! Guide to great Video Game Design* (p. 492). Retrieved from https://books.google.com.ec/books?id=09VFAwAAQBAJ&pg=PA438&lpg=PA438&dq=Level+Up!+Guide+to+Great+Video+Game+Design&source=bl&ots

=6KFan6ed4p&sig=lTkxIGn-rTUvLu8mNyVmgHbnUYE&hl=es-419&sa=X&ved=0CFEQ6AEwBmoVChMI6fqxtqjoxwIVgtWACh0Qbgi-#v=onepage&q=Level Up! Guide to Great Video Game Design&f=false

Secretaría de Educación Superior de Ciencia Tecnología e Innovación. (2015). ¿Qué es Prometeo? / What is Prometeo? | Prometeo. Retrieved February 27, 2015, from http://prometeo.educacionsuperior.gob.ec/que-es-prometeo/

TechTerms.com. (2015). GUI (Graphical User Interface) Definition. Retrieved May 29, 2014, from http://www.techterms.com/definition/gui

Turner, P. (2014). Cegui Wiki - Crazy Eddie's Gui System for Games (Open Source). Retrieved June 27, 2014, from http://cegui.org.uk/wiki/FAQ#What_is_CEGUI.3F

Turner, P., & Team, T. C. D. (2014). CEGUI. Retrieved June 27, 2014, from http://cegui.org.uk/features

Unity Technologies. (2015a). Unity - Game engine, tools and multiplatform. Retrieved February 27, 2015, from https://unity3d.com/es/unity

Unity Technologies. (2015b). Unity - SIM. Retrieved February 27, 2015, from http://unity3d.com/industries/sim

Uzquiano, I., Fernández, S., & Ortiz, J. (2010). Acceso manual e interfaz gráfica para el juego AI-LIVE, 105. Retrieved from http://e-archivo.uc3m.es/handle/10016/10531

Wikipedia. (2013). Sodipodi. Retrieved March 5, 2015, from https://es.wikipedia.org/wiki/Sodipodi

Wikipedia. (2014a). Gamer. Retrieved February 25, 2015, from https://es.wikipedia.org/wiki/Jugador_de_videojuegos

Wikipedia. (2014b). Login. Retrieved February 20, 2015, from http://www.alegsa.com.ar/Dic/login.php

Williams, R. (2004). *The Non-Designer's Design Book* (Second, p. 194). Retrieved from http://www.indusvalley.edu.pk/library/e books/The non-designers design book.pdf

Anexo 5: Manual del Programador

Lenguaje de programación y base de datos

El lenguaje de programación con el que se realizó el presente sistema es C++, utilizando el IDE Microsoft Visual Studio 2008 y librerías de Delta3D.

La base de datos del sistema en realidad es en conjunto de mapas (archivos *.dtmap) ubicados en el siguiente directorio y realizados en código XML: VirtualSchool-client-20141117\Release\Data\maps.

Cada uno de los archivos guarda información específica que explica al sistema los datos que deben cargarse dependiendo del caso.

Ejecución del sistema

Para ejecutar el Gists-UTPL en su estado actual se requiere de por lo menos dos computadoras, una que actúe como servidor y otra como cliente. Ambas computadoras deben estar conectadas a la misma red, de preferencia una red específica para el funcionamiento del mismo (LAN) que a su vez tenga acceso a internet.

En la máquina que funcione como servidor se debe correr el archivo “VirtualSchool” de la carpeta “VirtualSchool-server-20141117”. En la ventana de logeo se ingresa el usuario y la contraseña del profesor y la dirección IP de la computadora, el puerto está establecido por defecto en 80. Una vez iniciado el servidor se pueden iniciar las computadoras clientes ejecutando el archivo “VirtualSchool” de la carpeta “VirtualSchool-client-20141117” y de igual forma en la ventana de logeo se ingresa el usuario y la contraseña del estudiante y la dirección IP de cada una de las computadoras clientes, el puerto también se encuentra definido por defecto en el puerto 80.

Si el servidor está en línea se creará la conexión con los clientes, de otra forma los clientes podrán utilizar el sistema de forma independiente en modo de práctica.

Métodos dentro del sistema

Clase: Coverwindow

```
void CoverWindow::enable(bool enable)
{
   _isEnabled=enable;
   _isEnabled? _background->show():_background->hide();
}
```

Habilita la imagen de fondo en la ventana.

```
bool CoverWindow::onStart(const CEGUI::EventArgs& e)
{
   _guiCom->sendGameStateChangedMessage(GameState::STATE_COVER_WINDOW,
GameState::STATE_LOGIN);
   return true;
}
```

Llama a la ventana de logeo.

Clase: Loginwindow

```
void LoginWindow::enable(bool enable)
{
   mIsEnabled=enable;
   mIsEnabled? _background->show():_background->hide();
}
```

Habilita la imagen de fondo de la pantalla.

```
bool LoginWindow::_onLogin(const CEGUI::EventArgs& e){
   {
      MessageManager::GetInstance().sendWarningMessage("server connection
failed!");
      _guiCom-
>sendGameStateChangedMessage(GameState::STATE_LOGIN,GameState::STATE_MENU
);
      return false;
   }
   _guiCom-
>sendGameStateChangedMessage(GameState::STATE_LOGIN,GameState::STATE_MENU
);
   _guiCom->getRunningWin()->getChatWin()->setupNetwork();
   return true;
}
```

Verifica la conexión a la red.

```
bool LoginWindow::_onCancel(const CEGUI::EventArgs& e)
{
   _guiCom-
>sendGameStateChangedMessage(GameState::STATE_LOGIN,GameState::STATE_COVE
R_WINDOW);
   return true;
}
```

Regresa a la ventana inicial del sistema.

```
void LoginWindow::_sendLoginMessage(const std::string& surname, const
std::string& passport)
{}
bool LoginWindow::_checkUserDataBase(void)
{
   std::vector<dtCore::BaseActorObject*> proxies;
   _guiCom->GetGameManager()->FindActorsByName("*",proxies);
   return false;
}
```

Verifica los datos del usuario al momento de ingresar al sistema.

Clase: Mainmenuwindow

```
void MainMenuWindow::enable(bool enable)
{
   _isEnabled=enable;
   _isEnabled? _background->show():_background->hide();
}
```

Oculta la imagen de fondo de pantalla.

```
bool MainMenuWindow::OnEnter(const CEGUI::EventArgs& e)
{
   MessageManager::GetInstance().emitEnterEvent();
   return true;
}
```

Reconoce la tecla Enter como un evento en cualquier momento.

```
bool MainMenuWindow::OnIntro(const CEGUI::EventArgs& e)
{
   _guiCom-
>sendGameStateChangedMessage(GameState::STATE_MENU,GameState::STATE_INTRO
TEXT);//GameState::STATE_INTRO
   return true;
}
```

Muestra la ventana de juego con el HUD.

```
bool MainMenuWindow::OnLanguage(const CEGUI::EventArgs& e)
{
   _guiCom-
>sendGameStateChangedMessage(GameState::STATE_MENU,GameState::STATE_LUG_C
HOICE);//GameState::STATE_INTRO
   return true;
}
```

Despliega la ventana de idioma para que el usuario elija el que desea.

```
bool MainMenuWindow::OnExit(const CEGUI::EventArgs& e)
{
   _guiCom->sendGameStateChangedMessage(GameState::STATE_MENU,
GameState::STATE_LOGIN);
   return true;
}
```

Vuelve a la ventana de logeo.

Clase: Gamecomponent

```
void GameComponent::OnAddedToGM()
{
   base::OnAddedToGM();
   _camera=GetGameManager()->GetApplication().GetCamera();
   _camera->SetClearColor(osg::Vec4(0.544f,0.744f,0.908f,0.9f));
   _root=GetGameManager()->GetApplication().GetScene()->GetOSGNode()-
>asGroup();
}
```

Inicializa todos los elementos del juego como el HUD, el mapa y la cámara.

```
void GameComponent::OnRemovedFromGM()
{
   base::OnRemovedFromGM();
}
```

Elimina componentes del GameManager, cuando el juego se cierra o cualquier evento deja de utilizar un objeto debe llamar a este método para que lo termine.

```
void GameComponent::_sentObjectSelectedMessage(const dtCore::UniqueId&
id)
{
   dtCore::RefPtr<dtGame::Message> msg;
   GetGameManager()-
>GetMessageFactory().CreateMessage(ga::MessageType::SUBMIT_ANSWER_TYPE,ms
g);
   ga::SubmitAnswerMessage&
osm=static_cast<ga::SubmitAnswerMessage&>(*msg);
   osm.Set_id(id);
   GetGameManager()->SendMessage(osm);
}
```

Inicializa mensajes para que el sistema pueda enviar notificaciones al usuario.

```
void GameComponent::_adjustCamera(const osg::Vec3& eye,const osg::Vec3&
lookAt,const osg::Vec3& up)
{
   if(_camera!=NULL)
   {
      dtCore::Transform trans;
      trans.Set(eye,lookAt,up);
      _camera->SetTransform(trans);
   }
}
```

Ajusta la cámara a cualquier posición que el usuario desee.

```
void GameComponent::_freshSubmission(const dtCore::UniqueId& id)
{
   if(*_currentState==GameState::STATE_MOTION_2D)
   {
   }
   else if(*_currentState==GameState::STATE_MOTION_1D)
   {

   }
   else if
(*_currentState==GameState::STATE_DISPLACEMENT_VELOCITY_ACCELERATION)
   {

   }
   else if(*_currentState==GameState::STATE_VECTORS)
   {
      _submissionVector(id);
   }
}
```

Define si el juego deseado funciona en una o dos dimensiones, para cargar los movimientos que se ajusten al mismo.

```
void GameComponent::_freshFireEvent()
{
   if(*_currentState==GameState::STATE_MOTION_2D)
   {
      ga::TankActor*
tank=static_cast<ga::TankActor*>(ga::Utils::getSingleActor(GetGameManager
(),*ga::ActorRegistry::TANK_ACTOR_TYPE));
      tank->fire();
   }
   else if(*_currentState==GameState::STATE_MOTION_1D)
   {
      _const_v_actor->activate(true);
      _const_v_actor->startWalkAnim();
      _const_a_actor->activate(true);
      _const_a_actor->startWalkAnim();
   }
   else if
(*_currentState==GameState::STATE_DISPLACEMENT_VELOCITY_ACCELERATION)
   {
      _velocityActor->enable();
      _velocityActor->setState(ga::VelocityActor::FIRED);
   }
   else if(*_currentState==GameState::STATE_VECTORS)
   {
   }
}
```

Dependiendo del juego en que el usuario se encuentre realiza la animación requerida, por ejemplo el disparo del tanque de guerra o el movimiento de las personas.

```
void GameComponent::_displacement_velocity_acceleration()
{
    _adjustCamera(osg::Vec3(0.0f,0.0f,100.0f),osg::Vec3(0.0f,0.0f,0.0f)
,osg::Vec3(0.0f,1.0f,0.0f));
    dtCore::DeltaDrawable* actor;
    actor=ga::Utils::getSingleActor(GetGameManager(),*ga::ActorRegistry::
VELOCITY_ACTOR_TYPE);
    _velocityActor=static_cast<ga::VelocityActor*>(actor);
    _velocityActor->setAcce(2.0f);
    _target=osg::Vec3(30.0f,0.0f,0.0f);
    _target_radius=2.0f;
    dtCore::RefPtr<osg::MatrixTransform>x_axis=ga::Utils::arrow
(osg::Vec3(-28.0f,0.0f,0.0f),_target,osg::Vec4(1.0f,0.0f,0.0f,0.0f)
,10.0f);
    _root->addChild(x_axis.get());
    _root->addChild(ga::Utils::circle_filled(_target,_target_radius,20,
osg::Vec4(0.0f,1.0f,0.0f,0.0f)));
}
```

Regula la velocidad de un elemento ajustando la cámara, y la distancia que recorre en un determinado tiempo para acelerarla o frenarla.

```
void GameComponent::_freshGameScene()
{
    if(*_currentState == GameState::STATE_MENU){}
    else if(*_currentState==GameState::STATE_UNKNOWN){}
    else if(*_currentState == GameState::STATE_INTRO){}
    else if(*_currentState == GameState::STATE_RUNNING){
      _compass=new dtCore::Compass(_camera);
      GetGameManager()->GetApplication().AddDrawable(_compass.get());
      _adjustCamera(osg::Vec3(0.0f,-
10.0f,5.0f),osg::Vec3(0.0f,0.0f,0),osg::Vec3(0.0f,0.0f,1.0f));
      _cable=new vst::CableGeode();
      _root->addChild(_cable->createGround());
      _root->addChild(_cable->getNode());
    }
    else if(*_currentState == GameState::STATE_BREAKING){}
    else if(*_currentState==GameState::STATE_MOTION_2D)
    {
      _motion_2D();
    }
    else if(*_currentState==GameState::STATE_MOTION_1D)
    {
      _motion_1D();
    }
    else if(*_currentState==GameState::STATE_VECTORS)
    {
      _vector();
    }
    Else if
(*_currentState==GameState::STATE_DISPLACEMENT_VELOCITY_ACCELERATION)
    {
      _displacement_velocity_acceleration();
    }
}
```

Refresca el escenario del juego reseteando los valores por defecto tanto en la cámara como en los elementos involucrados en cada juego.

```
void GameComponent::_motion_2D()
{
    _adjustCamera(osg::Vec3(0.0f,-20.0f,3.0f),osg::Vec3(0.0f,9.0f,0.0f),
osg::Vec3(0.0f,0.0f,1.0f));
}
```

Ajusta la cámara para los juegos en dos dimensiones.

```
void GameComponent::_motion_1D()
{
    _adjustCamera(osg::Vec3(60.0f,35.0f,1.0f),osg::Vec3(20.0f,20.0f,0.0f)
,osg::Vec3(0.0f,0.0f,1.0f));
    osg::Vec3 t_pos=osg::Vec3(40.0f,30.0f,-4.0f);
    ga::VirtualHumanActorProxy* proxy;
    GetGameManager()->FindActorByName("const_velocity_agentActor",proxy);
   _const_v_actor=static_cast<ga::VirtualHumanActor*>(proxy->GetActor());
   _const_v_actor->setTargetPos(t_pos);
   _const_v_actor->setVelocity(3.0f);
   _const_v_actor->setAcce(0.0f);
   _const_v_actor->startIdleAnim();
   GetGameManager()->FindActorByName("const_acceleration_agentActor",
proxy);
   _const_a_actor=static_cast<ga::VirtualHumanActor*>(proxy->GetActor());
   _const_a_actor->setTargetPos(t_pos);
   _const_a_actor->setVelocity(0.0f);
   _const_a_actor->setAcce(1.0f);
   _const_a_actor->startIdleAnim();
   osg::Vec3 v_pos=_const_v_actor->getPosition();
   osg::Vec3 a_pos=_const_a_actor->getPosition();
   _root->addChild(ga::Utils::line(v_pos,t_pos,osg::Vec4
(1.0f,0.0f,0.0f,0.0f)));
   _root->addChild(ga::Utils::line(a_pos,t_pos,osg::Vec4
(0.0f,1.0f,0.0f,0.0f)));
   _root->addChild(ga::Utils::circle(t_pos,2.0f,20,osg::Vec4
(0.0f,1.0f,0.0f,0.0f)));
}
```

Ajusta la cámara para juegos con movimiento en una dimensión, inicializa los actores que involucren el juego y las acciones con sus parámetros iniciales.

```
void GameComponent::_submissionVector(const dtCore::UniqueId& id)
{
    dtGame::GameActorProxy* gap=GetGameManager()->FindGameActorById(id);
    if(gap==NULL)return;
    dtCore::DeltaDrawable* drawable=gap->GetActor();
    ga::BombActor* ball=dynamic_cast<ga::BombActor*>(drawable);
    if(ball!=NULL)
    {
      osg::Vec3 pos=ball->getPosition();
      osg::Vec3 origin(0.0f,0.0f,-1.0f);
       _root->addChild(ga::Utils::arrow(origin,pos,osg::Vec4
(0.0f,0.0f,1.0f,0.0f),6.0f));
    }
}
```

Ubica al actror por id y si no lo encuentra lo inicializa con los valores predeterminados.

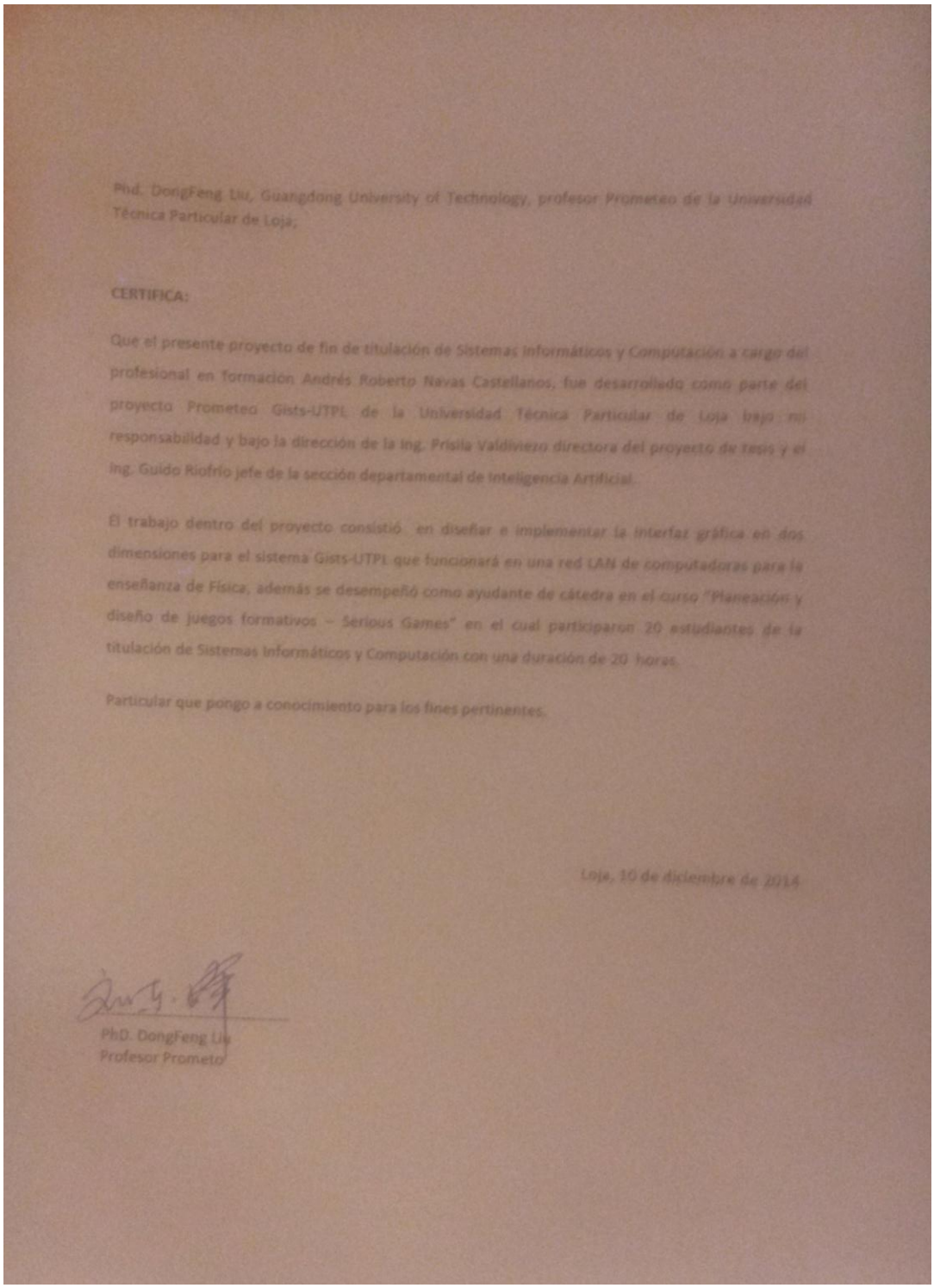
```
void GameComponent::_vector()
{
    _adjustCamera(osg::Vec3(0.0f,0.0f,20.0f),osg::Vec3(0.0f,0.0f,0.0f),
osg::Vec3(0.0f,1.0f,0.0f));
    _arrow=ga::Utils::arrow(osg::Vec3(-3.0f,0.0f,0.0f),osg::Vec3(1.0f,-
1.0f,0.0f),osg::Vec4(1.0f,0.0f,0.0f,0.0f),10.0f);
   dtCore::RefPtr<osg::MatrixTransform> x_axis=ga::Utils::arrow(osg::Vec3
(-8.0f,0.0f,0.0f),osg::
Vec3(8.0f,0.0f,0.0f),osg::Vec4(1.0f,0.0f,0.0f,0.0f),10.0f);
   dtCore:: RefPtr<osg::MatrixTransform> y_axis=ga::Utils::arrow
(osg::Vec3(0.0f,-6.0f,0.0f),osg::Vec3(0.0f,6.0f,0.0f),osg::
Vec4(1.0f,0.0f,0.0f,0.0f),10.0f);
    _root->addChild(x_axis.get());
    _root->addChild(y_axis.get());
    std::vector<ga::BombActor*> actors;
    ga::DataBetweenGAME2BDI::getInstance().getBalls(actors);
    for(size_t i=0;i<actors.size();i++)
    {
      ga::BombActor* ball=actors[i];
      osg::Vec3 pos(dtUtil::RandRange(-6,6),dtUtil::RandRange(-5,5),-
1.0);
      ball->setPosition(pos);
    }
}
```

Inicializa un agente vector con la posición inicial de la cámara y las coordenadas y dimensiones del mismo.

```
void GameComponent::_freshTick(float dt)
{
    if(!_bTick)return;
    if(*_currentState==GameState::STATE_RUNNING)
    {
      _cable->onTick(dt);
    }
    else if(*_currentState==GameState::STATE_MOTION_2D)
    {
      _tickMotion2D(dt);
    }
    else if(*_currentState==GameState::STATE_MOTION_1D)
    {
      _tickMotion1D(dt);
    }
    else if(*_currentState==GameState::STATE_DISPLACEMENT_VELOCITY_ACCELERATION)
    {
      _tickDisplacementVelocityAcceleration(dt);
    }
    else if(*_currentState==GameState::STATE_VECTORS)
    {
      _tickVector(dt);
    }
}
```

Recarga el mapa y las posiciones de los agentes al estado actual.

Anexo 6 :Certificado de la participación dentro del proyecto Prometeo



Phd. DongFeng Liu, Guangdong University of Technology, profesor Prometeo de la Universidad Técnica Particular de Loja.

CERTIFICA:

Que el presente proyecto de fin de titulación de Sistemas Informáticos y Computación a cargo del profesional en formación Andrés Roberto Navas Castellanos, fue desarrollado como parte del proyecto Prometeo Gists-UTPL de la Universidad Técnica Particular de Loja bajo mi responsabilidad y bajo la dirección de la Ing. Prisila Valdivieso directora del proyecto de tesis y el Ing. Guido Riofrío jefe de la sección departamental de Inteligencia Artificial.

El trabajo dentro del proyecto consistió en diseñar e implementar la interfaz gráfica en dos dimensiones para el sistema Gists-UTPL que funcionará en una red LAN de computadoras para la enseñanza de Física, además se desempeñó como ayudante de cátedra en el curso "Planeación y diseño de juegos formativos – Serious Games" en el cual participaron 20 estudiantes de la titulación de Sistemas Informáticos y Computación con una duración de 20 horas.

Particular que pongo a conocimiento para los fines pertinentes.

Loja, 10 de diciembre de 2014

PhD. DongFeng Liu
Profesor Prometeo

Oficio 1, Firmado por el PhD Dongfeng Liu explicando el trabajo realizadfo.