UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 5

Implementación del diseño instruccional de los escenarios de aprendizaje para los Laboratorios Virtuales

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

AUTORA: Ladimairy Capote González

TUTOR: Lic. Luis Gabriel Viciedo Caraballoso

ASESOR: MsC. Manuel Villanueva Betancourt

La Habana, Junio del 2011.

Año 53 de la Revolución.

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____	_____
Firma del Autor	Firma del Tutor
Ladimairy Capote González.	**Lic. Luis Gabriel Viciedo Caraballoso.**

DATOS DE CONTACTO

Nombre y apellidos: Luis Gabriel Viciedo Caraballoso.

Especialidad de graduación: Licenciatura en Educación, especialidad Física.

Categoría docente: Profesor auxiliar.

Categoría Científica: Licenciado.

Años de experiencia: 11 años.

Correo electrónico: viciedo@uci.cu.

AGRADECIMIENTOS

Quiero hacer llegar mis agradecimientos a todas las personas que me apoyaron en la realización del presente trabajo de diploma. A mi tutor Viciedo por guiarme y ayudarme en todo el transcurso de este año, a mi asesor Manuel Villanueva por su ayuda en el marco teórico. Agradezco a todos mis amigos de la universidad por todo su apoyo en los 5 años de carrera, especialmente a mi amiga Cecilia por su apoyo incondicional.

A mis abuelos por encaminarme en la vida. A mi papá que me impulsó a llegar hasta donde estoy hoy. A mi mamá por su preocupación y cariño. A mis hermanas por el entusiasmo con que apoyan mis metas. A mi novio Omar por su gran amor, su cariño, su comprensión y sobre todo por su ayuda para sobreponerme ante todo. A mis suegros por su preocupación. A toda mi familia.

A todos gracias de corazón.

DEDICATORIA

A mis Abuelos Dorita y Humberto por ser los mejores abuelitos del mundo.

A mis padres por su amor.

A mis hermanas por confiar en mí.

Al mejor hombre del universo por amarme tanto.

A una personita especial que viene dentro de mí.

RESUMEN

El creciente avance de las tecnologías de las comunicaciones y la realidad virtual han propiciado el desarrollo de sistemas de simulación utilizados para el aprendizaje y el entrenamiento en diferentes esferas de la medicina, la industria militar y en la educación. Ejemplo de ellas son los laboratorios virtuales muy utilizados en la actualidad en la educación por sus grandes capacidades de enseñanza. Desde sus inicios el desarrollo de estos laboratorios ha ido cambiando constantemente de acuerdo a su objetivo principal, que cada vez los estudiantes se apropien más del conocimiento brindado. Precisamente es ese el objetivo de este trabajo, presentar una propuesta de desarrollo de estos laboratorios con un diseño semejante al de un videojuego y por consiguiente aprovechar las grandes posibilidades que estos brindan. Proponiendo además el desarrollo de estos mediante el uso de lenguajes de extensión, y las grandes ventajas en calidad y tiempo de desarrollo que posibilitaría el uso de estos lenguajes con un motor gráfico compatible.

Para cumplir con los objetivos de la investigación se analizaron las ventajas de los lenguajes de extensión que más se adaptaran a la solución del problema, así como la selección de los motores gráficos más convenientes. Se realizó el diseño instruccional que debe regir la propuesta de un laboratorio virtual.

Como resultado de este proceso se realizó un demo con algunas funcionalidades de un laboratorio virtual implementado con un motor gráfico utilizando un lenguaje de extensión.

Palabras clave: Diagrama instruccional, laboratorio virtual, lenguaje de extensión, motor gráfico.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Existe un gran número de estudios de psicología cognitiva que demuestran que las personas adquieren mejor el conocimiento con actividades prácticas y reflexionando sobre las consecuencias de sus acciones que mirando o escuchando a alguien que les cuenta lo que deben aprender. Además, la experimentación obliga a los alumnos a implicarse en el aprendizaje convirtiéndose en una parte esencial del aprendizaje de la mayoría de las ramas científicas y técnicas.

La creciente aparición de aplicaciones de las nuevas tecnologías a la enseñanza tales como avances de los entornos, multimedia y la aplicación cada vez más amplia del internet en la educación debido a la enorme cantidad de recursos educativos, obliga a los docentes a hacer uso de las Tecnologías de la Información y las Comunicaciones (TIC) para integrarlas en el trabajo diario. Dentro de estas aplicaciones informáticas resultan de gran interés en las ciencias el uso de los laboratorios virtuales.

Es precisamente de la mano de las TIC desde donde se pueden aportar algunas soluciones para ampliar el acceso a la experimentación en forma de laboratorios virtuales. Así se conseguirán simultáneamente dos objetivos didácticos: (1) Realizar prácticas relacionadas con la asignatura ampliando la disponibilidad de los laboratorios y (2) formar a nuestros alumnos en el uso de las TIC.

Se han realizado disimiles laboratorios virtuales entre ellos se puede mencionar los laboratorios virtuales de química como ejemplo de ello está el VlabQ, este permite usar equipos y materiales presentes en un laboratorio de química para simular procesos tales como conservación de la materia, destilación simple y reversibilidad de las reacciones. Otros ejemplos de laboratorios virtuales de química son: ChemLab, LiveChem. También existen laboratorios de física como Phet, e Interactive Physic este último permite crear simulaciones como estudios del péndulo, cinemática y dinámica.

La Universidad de las Ciencias Informáticas en aras de convertirse en una compañía productora de software de prestigio internacional y hacer de la informática una rama productiva para Cuba, ha venido vinculando la formación educativa con el desempeño productivo. El Centro de Informática Industrial perteneciente a la Facultad 5, tiene varias líneas de trabajo dentro de las cuales se encuentra la Línea de Investigación Científica, específicamente el proyecto Laboratorios Virtuales (PROLAVI). Este proyecto se encarga de la realización de laboratorios virtuales como escenarios de aprendizaje. En la actualidad el desarrollo de estos laboratorios virtuales es muy avanzado y hay mucho auge en esta rama de la realidad virtual; pues esto es lo que pretende PROLAVI, incluir al país en esta ola de aprendizaje tecnológico.

Estos laboratorios virtuales tienen varios escenarios de aprendizaje que presentan una guía de pasos con los cuales el estudiante tiene que cumplir para la realización de sus actividades, estos pasos son llamados diseños instruccionales.

Por tanto, se puede enunciar como **situación problémica** la siguiente: El desarrollo de esta serie de laboratorios virtuales en el proyecto PROLAVI se ve un tanto frenada por el coste en tiempo de realización. Además de que no se diferencia el trabajo de los desarrolladores de los componentes de los escenarios de aprendizaje con el de los programadores de la lógica de aprendizaje de los laboratorios virtuales.

Basada en la problemática planteada se resume el **problema de investigación** en la siguiente interrogante**:**

¿Cómo programar el diseño instruccional de los escenarios de aprendizaje de un laboratorio virtual, que contribuya a disminuir los tiempos de implementación en el proyecto Laboratorios Virtuales?

Lo cual precisa el siguiente **Objeto de estudio:** Integración de componentes informáticos.

Y para darle solución al problema de investigación planteado se propone como **Objetivo General:** Desarrollar el diseño instruccional de un laboratorio virtual con el uso de lenguajes de extensión y un motor gráfico compatible.

Por todo lo anterior se determina como **Campo de acción:** Implementación de diseños instruccionales de los escenarios de aprendizaje en los laboratorios virtuales.

Como **Idea a defender,** se propone**:** La innovación tecnológica en la implementación del diseño instruccional de los escenarios de aprendizaje a realizar contribuirá a disminuir el tiempo de desarrollo de las aplicaciones informáticas en el proyecto PROLAVI.

Y para darle cumplimiento al objetivo se proponen las siguientes **Tareas de investigación:**

- ✓ Elaboración del marco teórico de la investigación a partir del estado del arte sobre la utilización de programación con lenguajes de extensión en el área de los videojuegos.
- ✓ Selección de los engines gráficos opensource basados en el lenguaje de extensión que permitan desarrollar la solución informática a la problemática planteada.
- ✓ Elaboración de las pautas para la realización del diseño instruccional de los escenarios de aprendizaje a implementar.
- ✓ Implementación de la lógica de los escenarios de aprendizaje con el lenguaje de extensión y motor gráfico seleccionado.
- ✓ Valoración de los resultados obtenidos.

Resultados esperados:

Con la realización del trabajo de diploma se espera obtener un diseño instruccional para la integración de los diferentes componentes en los laboratorios virtuales, que acelere el proceso de desarrollo de los mismos.

Métodos de investigación utilizados

Métodos teóricos:

Analítico-Sintético: Para la consulta de la documentación existente acerca de los ambientes de aprendizaje, lenguajes de extensión y motores gráficos, y el estudio de los conceptos empleados en el tema en cuestión.

Análisis Histórico-Lógico: Para tener información sobre las tendencias actuales de los laboratorios virtuales.

Métodos empíricos:

Observación: Para conocer y analizar las características específicas que posee un laboratorio virtual, permitiendo proponer el diseño instruccional que en la investigación se detalla.

Estructura Capitular:

- ✓ En el **Capítulo 1** llamado Fundamentación teórica se selecciona la teoría fundamental asociada al tema a defender como son la descripción de los lenguajes de extensión, motores gráficos.
- ✓ En el **Capítulo 2** nombrado Construcción de la solución, se realiza la selección de la metodología, las herramientas y los lenguajes a utilizar. Además, se definen los requisitos funcionales y no funcionales; actores y casos de uso del sistema así como la descripción de dichos casos de usos.
- ✓ En el **Capítulo 3**. Solución propuesta donde se describe la estructura de trabajo propuesta para satisfacer la solución final, además de la propuesta de diseño utilizada en el desarrollo de la aplicación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se abordarán los elementos teóricos que permitirán dar solución al problema científico planteado. Se explicarán los tipos de ambientes de aprendizaje así como la descripción de los laboratorios virtuales con sus ventajas y desventajas. Se dará una descripción de lo que es un lenguaje de extensión y su utilización en los videojuegos. Además, se explicará lo que es un motor gráfico y las ventajas del uso de estos en los laboratorios virtuales.

1.1. Los ambientes de aprendizaje

En el Informe Horizont del año 2011, aparecen analizadas las seis tecnologías informáticas emergentes que deben impactar la enseñanza universitaria en los próximos cinco años. Entre ellas aparece el aprendizaje basado en juegos. En este sentido, se explica el impacto del juego en el desarrollo cognitivo del adulto joven y la posibilidad de resolución de problemas, la colaboración entre pares, además de la exploración de nuevos roles y la experimentación en ambientes simulados.

La sensación de trabajar hacia objetivos claros y sin ambigüedades, la posibilidad de soluciones a situaciones problemáticas simuladas, asumiendo un rol determinado, la interacción social en la red académica, entre otras, son características provenientes de los juegos, que pueden ser transferidas al proceso de aprendizaje del estudiante universitario.

En el transcurso de la carrera de Ingeniería en Ciencias Informáticas, los futuros ingenieros deben transitar por diferentes roles como son el programador de soluciones informáticas, el administrador de sistemas informáticos, entre otros, donde diferentes ambientes de aprendizajes tridimensionales, pueden jugar un papel fundamental de competencias, al visualizar tridimensionalmente situaciones problemáticas o de cierta singularidad, que deben ser resueltas interactuando y modificando variables espaciales y temporales, proceso que puede ser evaluado posteriormente por profesores y tutores.

Un ambiente de aprendizaje interactivo es un espacio electrónico, donde se simulan en 2D o 3D situaciones problemáticas provenientes de un diseño de aprendizaje, donde deben cumplirse objetivos instructivos y educativos de un programa de estudio, en que estudiantes y tutores colaboran en escenarios simulados y cumplen tareas experimentales o de entrenamiento, asumiendo un rol propio de su profesión y donde son evaluados por ello.

Debemos diferenciar los ambientes de aprendizaje interactivos, de los entornos virtuales de aprendizaje.

Se entiende que un entorno virtual de aprendizaje (EVA) puede ser desde un campus virtual sin interacción presencial hasta una clase convencional que utiliza herramientas telemáticas en el proceso de enseñanza-aprendizaje, siempre que los recursos sean también accesibles fuera del horario regular y la clase asignada. Esta característica es la que hace de los EVA un instrumento de innovación dentro de las instituciones convencionales de enseñanza (Marchena, 2008).

Se entiende por EVA al espacio físico donde las nuevas tecnologías, han rebasado el entorno escolar tradicional favoreciendo al conocimiento y a la apropiación de contenidos, experiencias y procesos pedagógicos-comunicacionales. Están conformadas por el espacio, el estudiante, el asesor, los contenidos educativos, la evaluación y los medios de información y comunicación (Avila,1999).

Estos EVA siguen el mismo objetivo principal que sigue la pedagogía, el cual es enseñar. Siguiendo sus mismas funciones pedagógicas: actividades de aprendizaje, situaciones de enseñanza, materiales de aprendizaje, apoyo, autorización y evaluación. Además, incluyen características más específicas y la diferencia clave es el uso de herramientas de telecomunicación en el proceso enseñanza-aprendizaje.

Muchas aplicaciones multimedia, elaboradas con disímiles tecnologías informáticas, pueden considerarse como ambientes de aprendizajes interactivos. En este caso, se consideran solo aquellas aplicaciones informáticas que asumen estrategias y experiencias provenientes de la industria de los videojuegos y de la Realidad Virtual, dada la potencialidad de su aplicación en el área del aprendizaje en contextos educativos. Como ejemplos estas aplicaciones, podemos mencionar: simuladores de vuelos y de conducción, entrenadores médicos, programas de visualización científica, instrumentación virtual, los juegos serios y laboratorios virtuales, donde la principal forma de interacción ocurre entre escenarios con objetos simulados en 2D o 3D.

La industria del videojuego con más de treinta años de experiencia ha aportado tecnologías suficientes para ser empleadas con éxito en el aprendizaje y en desarrollo de competencias en la rama ingenieril, pero solo en los últimos años han sido tomadas en cuenta por los educadores y los especialistas en Tecnología Educativa.

1.2. El laboratorio virtual como tipo de ambiente de aprendizaje interactivo

Un laboratorio virtual es un sistema computacional que pretende simular el ambiente de un laboratorio tradicional, brindando las mismas funcionalidades que estos y permitiendo además explotar, mediante la virtualidad, nuevos elementos que suelen ser riesgosos en su desempeño en un laboratorio real. En estos los experimentos se realizan paso por paso igual que en los laboratorios tradicionales, se visualizan instrumentos y fenómenos mediante objetos dinámicos y se

obtienen resultados ya sean numéricos o gráficos. Aportan los elementos necesarios para la realización de actividades prácticas desde cualquier lugar accesible en intranet, internet u otro ambiente computacional donde los estudiantes realizan las prácticas de una forma lo más similar a un laboratorio físico Algunas de las ventajas que brindan estos laboratorios virtuales se ven reflejadas en la educación. En especial constituye una buena opción para las prácticas de enseñanza en el país debido a la falta de laboratorios científicos tradicionales.

Los laboratorios virtuales son escenarios de aprendizaje electrónicos concebidos para la colaboración y la experimentación dentro de un proceso de enseñanza-aprendizaje, con el objetivo de desarrollar habilidades, investigaciones o realizar otras actividades creativas. Igualmente permiten elaborar, consultar y difundir resultados mediante tecnologías de información y comunicación. Una de sus variantes es realizar prácticas y experimentos de laboratorio de manera simulada en la computadora. Es decir, se manipulan los mismos elementos que en una experimentación real y pueden obtenerse los mismos resultados.

Por lo tanto, una de las características que mejor define el laboratorio virtual es la interacción, ya que el usuario hace realmente un experimento, que sólo se progresa si el software brinda los datos y la información que se necesitan para hacer las transformaciones que se desean.

Los laboratorios virtuales se han clasificado en tres tipos:

Laboratorios virtuales software: Son laboratorios virtuales desarrollados como un programa de software independiente destinado a ejecutarse en la máquina del usuario, y cuyo servicio no requiere de un servidor Web. Es el caso de programas con instalación propia, que pueden estar destinados a plataformas Unix y MS Windows, e incluso necesitar que otros componentes de software estén instalados previamente, pero que no necesitan los recursos de un servidor determinado (como bases de datos o módulos de software de servidor) para funcionar. También determinados laboratorios virtuales pensados inicialmente como aplicaciones Java accesibles a través de un servidor Web se pueden considerar de este tipo si funcionan localmente y no necesitan recursos de un servidor en concreto.

Laboratorios virtuales *Web*: Este tipo de laboratorios se basa en un software que depende de los recursos de un servidor determinado. Esos recursos pueden ser: determinadas bases de datos, software que requieren ejecutarse en su servidor y la exigencia de determinado hardware para su ejecución. No son programas que un usuario pueda descargar en su equipo para ejecutar localmente de forma independiente.

Laboratorios remotos: Se trata de laboratorios remotos que permiten operar remotamente cierto equipamiento, bien sea didáctico como maquetas especificas, o industrial, además de poder

ofrecer capacidades de laboratorio virtual. En general, estos laboratorios requieren de servidores específicos que les den acceso a las máquinas para operar de forma remota, y no pueden ofrecer su funcionalidad ejecutándose de forma local. Otro motivo que hace dependientes estos laboratorios de sus servidores es la habitual gestión de usuarios en el servidor.

1.2.1. Ventajas de los laboratorios Virtuales

- ✓ Acerca y facilita a un mayor número de alumnos a la realización de experiencias, aunque los alumnos y los laboratorios no coincidan. El estudiante accede a los equipos del laboratorio a través de un navegador, pudiendo experimentar sin riesgo alguno, y además, se flexibiliza el horario de prácticas y evita la saturación por el solapamiento con otras asignaturas.

- ✓ Reducen el coste del montaje y mantenimiento de los laboratorios físicos, siendo una alternativa barata y eficiente, donde el estudiante simula los fenómenos a estudiar como si los observase en el laboratorio real.

- ✓ Es una herramienta de auto aprendizaje, donde el alumno altera las variables de entrada, configura nuevos experimentos, aprende el manejo de instrumentos y personaliza el experimento. La simulación en el laboratorio virtual, permite obtener una visión más intuitiva de aquellos fenómenos que en su realización manual no aportan suficiente claridad gráfica. El uso de laboratorios virtuales da lugar a cambios fundamentales en el proceso habitual de enseñanza, en el que se suele comenzar por el modelo matemático. La simulación interactiva de forma aislada posee poco valor didáctico, esta debe ser embebida dentro de un conjunto de elementos multimedia que guíen al alumno eficazmente en el proceso de aprendizaje. Se trata de utilizar la capacidad de procesamiento y cálculo del ordenador, incrementando la diversidad didáctica, como complemento eficaz de las metodologías más convencionales.

- ✓ Los estudiantes aprenden mediante pruebas prácticas, sin miedo a sufrir o provocar un accidente, sin avergonzarse de realizar varias veces la misma prueba, ya que pueden repetirlas sin límite; sin temor a dañar alguna herramienta o equipo. Pueden asistir al laboratorio cuando ellos quieran, y elegir las áreas del laboratorio más significativas para realizar prácticas sobre su trabajo.

- ✓ En Internet encontramos multitud de simulaciones de procesos físicos (en forma de applets de Java y/o Flash). Con estos objetos dinámicos, el docente puede preparar actividades de aprendizaje que los alumnos han de ejecutar, contestando al mismo tiempo las cuestiones que se les plantean.

1.2.2. Desventajas de los laboratorios virtuales

- ✓ El laboratorio virtual no puede sustituir la experiencia práctica altamente enriquecedora del laboratorio tradicional. Debe ser una herramienta complementaria para formar a la persona y obtener un mayor rendimiento.
- ✓ En el laboratorio virtual se corre el riesgo de que el alumno se comporte como un mero espectador. Es importante que las actividades en el laboratorio virtual, vengan acompañadas de un guión que explique el concepto a estudiar, así como las ecuaciones del modelo utilizado. Es necesario que el estudiante realice una actividad ordenada y progresiva, conducente a alcanzar objetivos básicos concretos.
- ✓ El alumno no utiliza elementos reales en el laboratorio virtual, lo que provoca una pérdida parcial de la visión de la realidad. Además, no siempre se dispone de la simulación adecuada para el tema que el profesor desea trabajar. En Internet existe demasiada información, a veces inútil. Para que sea útil en el proceso de enseñanza aprendizaje, se debe de seleccionar los contenidos relevantes para nuestros alumnos. Son pocas las experiencias realizadas con laboratorios en los centros educativos, donde aún impera el uso de recursos tradicionales, tanto en la exposición de conocimientos en el aula como en el laboratorio.

1.2.3. Escenarios virtuales de aprendizaje

Es la interfaz visual donde se realiza el aprendizaje de los contenidos y objetivos previstos, ocurren las interacciones intuidas entre los objetos simulados y los educandos y ha sido objeto de un diseño instructivo por parte del profesor y los especialistas informáticos. Incluye simulaciones 3D, niveles de ayuda en forma de textos, imágenes, sonido y video, encaminado a la estrategia de aprendizaje prevista.

Fase:

Las fases presentes en el escenario virtual de aprendizaje, corresponden a los hitos o referencias previstos y que fueron definidos por el diseño instructivo y que pautan el cumplimiento gradual de los objetivos del laboratorio virtual.

Habilidad:

Significa el dominio de un sistema complejo de acciones y operaciones necesarias para la regulación conveniente de la actividad, donde los individuos se apropian activamente de los fundamentos científicos, pedagógicos y psicológicos que permiten desarrollar con éxito una

actividad práctica, que esta cumpla sus objetivos y no constituya una simple acumulación de acciones anárquicamente ubicadas.

Las habilidades son sistemas cuya dinámica de momentos o pasos hace eficiente el comportamiento humano, el cual puede estar compuesto de diferente “materia prima”, dependiendo de los objetivos y condiciones en que tiene lugar dicho comportamiento. El curso del comportamiento tiene lugar en gran medida a través de acciones y mensajes, luego entonces la eficacia o habilidad tiene lugar gracias a ambos.

1.3. La adaptabilidad de los ambientes de aprendizaje interactivos a las necesidades educativas

Las aplicaciones informáticas con fines educativos, portan una necesidad primordial: deben adaptarse a las necesidades educativas de las instituciones y de los individuos que las necesitan. Ello implica que el grado de personalización que ofrece la aplicación informática debe ser adecuado para responder a esas necesidades y esto se logra informáticamente mediante una arquitectura flexible y con el uso de lenguajes de extensión en la aplicación. Las aplicaciones multimedia por lo general, son genéricas, por lo que queda limitado la posibilidad de personalización del aprendizaje, siendo en ocasiones reemplazado esta necesidad por excesiva información adicional, recargando innecesariamente la aplicación.

Estas estrategias de adaptabilidad han tenido su mayor éxito en el área de los videojuegos, por lo que es una necesidad imperiosa asumir estas experiencias en el desarrollo de aplicaciones informáticas con fines educativos.

1.4. Los lenguajes de extensión

Un lenguaje de extensión o interpretado es un lenguaje de programación, que a diferencia de los lenguajes compilados, no necesita ser procesado mediante un compilador. Esto es posible ya que estos lenguajes presentan un programa encargado de traducir su código, este programa recibe el nombre de intérprete. La principal ventaja es la de ser independiente de la máquina y del sistema operativo ya que no contiene instrucciones propias de un procesador sino que contiene llamadas a funciones que el intérprete deberá reconocer. Basta que exista un intérprete de un lenguaje para dicho sistema y todos los programas escritos en ese lenguaje podrán ejecutarse en ese sistema.

Además, permite modificar en tiempo de ejecución el código que se está ejecutando así como añadirle nuevo, algo que resulta práctico cuando se quiere hacer pequeñas modificaciones en una aplicación y no se desee tener que recompilarla toda cada.

1.4.1. Ventajas de los lenguajes de extensión

- ✓ Independencia de plataforma.
- ✓ Tipos dinámicos.
- ✓ Facilidad en la depuración (es más fácil obtener información del código fuente en lenguajes interpretados).
- ✓ Pequeño tamaño del programa (puesto que los lenguajes interpretados tienen flexibilidad para elegir el código de instrucción).
- ✓ Gestión de memoria automática.

1.4.2. Ejemplos de lenguajes de extensión

PHP

Php es un acrónimo recursivo que significa “PHP Hypertext Pre-processor”, (inicialmente se llamó Personal Home Page). Surgió en 1995, desarrollado por PHP Group. Es uno de los lenguajes de programación más populares, la gran fluidez y rapidez de sus scripts y su prometedor futuro. PHP usa una mezcla entre interpretación y compilación para intentar ofrecer a los programadores la mejor mezcla entre rendimiento y flexibilidad. PHP compila para tu código una serie de instrucciones. Estas instrucciones son entonces ejecutadas una por una hasta que el script termina. Esto es diferente a la manera convencional de compilación de lenguajes como C++ donde el código es compilado a código ejecutable que es después ejecutado. Php es recompilado cada vez que se solicita un script.

Una ventaja importante de interpretar el código es que toda la memoria usada por el código es manejada por PHP, y el lenguaje automáticamente vacía esta memoria cuando el script finaliza. Esto significa que no hay que preocuparse de las conexiones a la base de datos, porque PHP se encarga de ello (Valdés, 2007).

Ventajas:

- ✓ Muy fácil de aprender.
- ✓ Se caracteriza por ser un lenguaje muy rápido.
- ✓ Soporta en cierta medida la orientación a objeto. Clases y herencia.
- ✓ Es un lenguaje multiplataforma: Linux, Windows, entre otros.

- ✓ Capacidad de conexión con la mayoría de los manejadores de base de datos: MySQL, PostgreSQL, Oracle, entre otras.
- ✓ Capacidad de expandir su potencial utilizando módulos.
- ✓ Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.
- ✓ Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- ✓ Incluye gran cantidad de funciones.
- ✓ No requiere definición de tipos de variables ni manejo detallado del bajo nivel.

Desventajas:

- ✓ Se necesita instalar un servidor web.
- ✓ Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- ✓ La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- ✓ La programación orientada a objetos es aún muy deficiente para aplicaciones grandes.
- ✓ Dificulta la organización por capas de la aplicación.

JavaScript

JavaScript es un lenguaje de secuencia de comandos desarrollado por Netscape Communications que está diseñado para el desarrollo de aplicaciones cliente y servidor de Internet. Netscape Navigator está diseñado para interpretar el código JavaScript contenido en las páginas Web. Es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado (Valdés, 2007).

Características:

Java Script es un lenguaje de secuencias de comandos basado en objetos e interpretado. Aunque tiene menos capacidades que los lenguajes orientados a objetos de altas prestaciones como C++ y Java, Java Script es suficiente para los propósitos para los que está creado. Java Script no es una versión reducida de cualquier otro lenguaje (solo está relacionado, distante e indirectamente, con Java, por ejemplo), ni es una simplificación de ningún lenguaje. Java Script es un lenguaje limitado. Por ejemplo, no es posible escribir aplicaciones independientes en Java Script y la capacidad de

lectura y escritura de archivos es mínima. Las secuencias de comandos de Java Script sólo pueden ejecutarse con un intérprete, que bien puede estar en un servidor Web o en un explorador de Web.

Java Script es un lenguaje que no necesita declarar los tipos de datos. Esto significa que no es necesario declarar explícitamente los tipos de datos de las variables. De hecho, no es posible declarar explícitamente los tipos de datos en Java Script. Más aún, en muchos casos Java Script realiza conversiones, automáticamente, cuando son necesarias. Por ejemplo, si intenta agregar un número a un elemento que contiene texto (una cadena), el número se convierte en texto.

Ventajas:

- ✓ Lenguaje de scripting seguro y fiable.
- ✓ Los script tienen capacidades limitadas, por razones de seguridad.
- ✓ El código JavaScript se ejecuta en el cliente.

Desventajas:

- ✓ Código visible por cualquier usuario.
- ✓ El código debe descargarse completamente.
- ✓ Puede poner en riesgo la seguridad del sitio, con el actual problema llamado XSS (significa en inglés Cross Site Scripting renombrado a XSS por su similitud con las hojas de estilo CSS).

Perl

Fue creado a mediados de los ochenta por Larry Wall. Es un lenguaje visualizado para la manipulación de cadenas de caracteres, archivos y procesos. Es una combinación de las características de los lenguajes más usados por los programadores de sistemas, como son los Shell del sistema operativo UNIX e incluso características de Pascal, aunque su potencia se basa en la similitud con las mejores características del lenguaje estructurado C. Actualmente existen dos versiones altamente populares de Perl, la 4.3 y la 5.0 (Byron, 2008).

El lenguaje Perl no es pre compilado, pero aún así es más rápido que la mayoría de lenguajes interpretados. Esto se debe a que los programas en Perl son analizados, interpretados y compilados por el interprete Perl antes de su ejecución.

Características:

- ✓ Es fácil de usar, aunque es difícil de aprender.
- ✓ Es rápido de crear, ya que no posee funciones que, aunque sean bastante interesantes, hagan disminuir la velocidad de desarrollo de una aplicación del lenguaje.
- ✓ Se puede utilizar en varios entornos, como puede ser Windows 95, OS/2, Linux, entre muchos otros, sin realizar cambios de código, siendo únicamente necesario la introducción del intérprete Perl correspondiente a cada sistema operativo.
- ✓ Tiene características que soportan una variedad de paradigmas de programación, como la estructural, funcional y la orientada a objetos. Al mismo tiempo, Perl no obliga a seguir ningún paradigma en particular, ni obliga al programador a elegir alguna de ellas.
- ✓ Tiene incorporado un poderoso sistema de procesamiento de texto y una enorme colección de módulos disponibles.
- ✓ Ofrece una ayuda en línea desde la consola de comandos. Por ejemplo para obtener ayuda sobre la función *print*, se escribe en una ventana MSDOS*: perldoc –f print.*
- ✓ Se ejecuta desde la línea de comandos de una ventana del sistema operativo.

ASP

Es una tecnología del lado de servidor desarrollada por Microsoft para el desarrollo de sitios web dinámicos. ASP significa en inglés (Active Server Pages), fue liberado por Microsoft en 1996. Las páginas web desarrolladas bajo este lenguaje es necesario tener instalado Internet Information Server (IIS).

ASP no necesita ser compilado para ejecutarse. Existen varios lenguajes que se pueden utilizar para crear páginas ASP. El más utilizado es VBScript, nativo de Microsoft. ASP se puede hacer también en Perl and Jscript (no JavaScript). El código ASP puede ser insertado junto con el código HTML. Los archivos cuentan con la extensión (asp).

Ventajas:

- ✓ Usa Visual Basic Script, siendo fácil para los usuarios.
- ✓ Comunicación óptima con SQL Server.
- ✓ Soporta el lenguaje Jscript (JavaScript de Microsoft).

Desventajas:

- ✓ Código desorganizado.
- ✓ Se necesita escribir mucho código para realizar funciones sencillas.
- ✓ Tecnología propietaria.
- ✓ Hospedaje de sitios web costosos.

Lenguaje Python

Es un lenguaje de programación creado en el año 1990 por Guido van Rossum, es el sucesor del lenguaje de programación ABC. Python es comparado habitualmente con Perl. Los usuarios lo consideran como un lenguaje más limpio para programar. Permite la creación de todo tipo de programas incluyendo los sitios web.

Su código no necesita ser compilado, por lo que se llama que el código es interpretado. Es un lenguaje de programación multiparadigma, lo cual fuerza a que los programadores adopten por un estilo de programación particular:

- ✓ Programación orientada a objetos.
- ✓ Programación estructurada.
- ✓ Programación funcional.
- ✓ Programación orientada a aspectos.

Ventajas:

- ✓ Libre y de fuente abierta.
- ✓ Lenguaje de propósito general.
- ✓ Gran cantidad de funciones y librerías.
- ✓ Sencillo y rápido de programar.
- ✓ Multiplataforma.
- ✓ Licencia de código abierto.
- ✓ Orientado a Objetos.
- ✓ Portable.

Desventajas:

- ✓ Lentitud por ser un lenguaje interpretado.

Lenguaje Ruby

Es un lenguaje interpretado de muy alto nivel y orientado a objetos. Su sintaxis está inspirada en Python, Perl. Es distribuido bajo licencia de software libre (Opensource). Ruby es un lenguaje dinámico para una programación orientada a objetos rápida y sencilla (Valdés, 2007).

Características:

- ✓ Existe diferencia entre mayúsculas y minúsculas.
- ✓ Múltiples expresiones por líneas, separadas por punto y coma ";".
- ✓ Dispone de manejo de excepciones.
- ✓ Ruby puede cargar librerías de extensiones dinámicamente si el (Sistema Operativo) lo permite.
- ✓ Portátil.

Ventajas:

- ✓ Permite desarrollar soluciones a bajo Costo.
- ✓ Software libre.
- ✓ Multiplataforma.

El lenguaje de programación Lua

Lua es un lenguaje de programación imperativo y estructurado, potente, ligero y muy rápido, está considerado como uno de los lenguajes de más reputación por su rendimiento. A reivindicar ser "tan rápido como Lua" es una aspiración de otros lenguajes de scripting. Varios indicadores muestran Lua como el lenguaje más rápido de interpretar.

Lua es fácil de aprender, la sintaxis es simple y clara. Uno puede leer y entender pequeños Lua scripts sin saber nada del lenguaje. Por lo tanto, es posible utilizar Lua como el idioma de los archivos de configuración que puede ser editado por personas que no saben nada acerca de programación. Lua es un lenguaje de tipos dinámicos, lo que básicamente significa que usted tendrá una gran libertad y flexibilidad al programar. La máquina virtual (un programa que toma el código Lua y lo ejecuta) es uno de los más rápidos este tipo de programas para lenguajes de script.

Lua rápidamente puede compilar y ejecutar secuencias de comandos de largo con miles de líneas de código y fácilmente manejar grandes cantidades de datos.

La máquina virtual de Lua es de un poco más de 160 KB, y agregando el compilador sólo otros 200 KB.

Lua está escrito en la llanura ANSI C, por lo que es muy portable. Funciona en casi todas las máquinas y sistema operativo (Emerich, 2009).

Características.

Lua es un lenguaje de extensión, suficientemente compacto para usarse en diferentes plataformas. En Lua las variables no tienen tipo, sólo los datos y pueden ser lógicos, enteros, números con punto flotante o cadenas. Estructuras de datos como matrices, conjuntos, tablas hash, listas y registros pueden ser representadas utilizando la única estructura de datos de Lua: la tabla.

La semántica de Lua puede ser extendida y modificada redefiniendo funciones de las estructuras de datos utilizando metatablas. Lua ofrece soporte para funciones de orden superior, colector de basura. Combinando todo lo anterior, es posible utilizar Lua en programación orientada a objetos.

Lua es de código abierto de software libre, distribuido bajo una licencia muy permisiva (el conocido MIT licence). Puede ser utilizado para cualquier propósito, incluyendo fines comerciales, sin ningún costo (Rio, 1999).

1.4.3. Los lenguajes interpretados en los videojuegos

Los lenguajes de extensión comenzaron a usarse en los videojuegos como una necesidad de los programadores, de extender las diferentes funcionalidades sin tener que compilar, recompilar y linkar código. Tanto así que surgieron muchos lenguajes interpretados que están básicamente dirigidos al mundo de los videojuegos. Con el avance que ha tenido la industria de los videojuegos en sus técnicas y métodos de desarrollo, impulsadas por la creciente demanda de los usuarios de obtener juegos cada vez más sofisticados y que incluyan más su participación, el uso de los lenguajes script ha tenido un papel fundamental.

La participación de los usuarios en los videojuegos con el uso de scripts se refiere a funcionalidades que presentan algunos juegos de cambiar su interfaz, adaptándola particularmente al gusto del usuario, es el caso de juegos como World of Warcraft, uno de los juegos más jugados mundialmente, el cual presenta una carpeta Addons donde se puede incluir o cambiar los scripts Lua existentes allí y modificar toda la interfaz del juego, incluso los sonidos o agregar la música de fondo que se desee. Como este son varios los juegos que presentan estas funcionalidades solo posibles con el uso de lenguajes interpretados.

- ✓ World of Warcraft, donde el usuario tiene la posibilidad de personalizar la interfaz creando añadidos que permiten informarle todo lo que cambie en su correspondiente carpeta Interface, en la que el WoW.exe tiene el intérprete de Lua y ejecuta en su Interface el Addon creado en Lua. (Emerich, 2009)

- ✓ El RTS a gran escala Supreme Commander, el cual es modificable por el usuario en casi todos sus aspectos.

- ✓ El juego de RPG Tibia, Modificable casi totalmente (poderes, mapas, etc.) usando Lua y XML.

1.4.4. Uso de los lenguajes interpretados en los laboratorios virtuales

Tras haber visto las diversas aplicaciones que se le da a los lenguajes de extensión en el área de los videojuegos, de acuerdo a las ventajas que brindan los mismos, se puede afirmar que su uso en el desarrollo de los laboratorios virtuales potenciaría muchos beneficios, ya que ganarían las funcionalidades que brindan los lenguajes interpretados a los videojuegos, tales como son las configuraciones y la personalización de sus ambientes adecuándose a un usuario específico o a las necesidades de los clientes. También sería más factible utilizar uno de estos lenguajes interpretados, por su sencillez a la hora de programar y de entender sus códigos, en la integración de los laboratorios virtuales que es algo muy engorroso de realizar hoy día.

1.5. Inclusión del diseño instruccional en el desarrollo de PROLAVI

Se define el proceso del diseño del aprendizaje o instruccional como:

"*... un proceso interactivo que consta de cinco fases básicas, incluido el análisis, diseño, desarrollo, implementación y evaluación*" (Branch, 1997).

En el diseño instruccional, la tarea de los diseñadores es planificar la instrucción para que el estudiante pueda utilizar una o más de las estrategias cognoscitivas para aprender el material, para activamente procesar el contenido. A medida que el diseñador planifica la instrucción, se deben tomar medidas acerca de que estrategia o estrategias son más apropiadas para el contenido y para los estudiantes particulares (Dorrego, 1997).

A continuación se presentan las cinco fases básicas:

Fase de análisis:

- ✓ Evaluar las necesidades e identificar los objetivos instructivos del programa con los que está relacionado.

- ✓ Analizar objetivos, tareas, competencias y contextos.
- ✓ Analizar las características del estudiante.
- ✓ Analizar el tipo de aprendizaje que declara el programa de estudio y los contextos en que se desempeñarían los estudiantes.

Fase de Diseño:

- ✓ Generar, agrupar y determinar la prioridad de los objetivos que deben cumplirse.
- ✓ Determinar cómo el estudiante puede valorar su desempeño.
- ✓ Seleccionar los medios de aprendizaje a emplear.

Fase de desarrollo:

- ✓ Adquirir los materiales.
- ✓ Crear diagramas de flujo y storyboards.
- ✓ Reevalúe y modifique materiales que necesita para el aprendizaje.
- ✓ Produzca materiales de aprendizaje.

Fase de implementación y evaluación:

- ✓ Asigne las tareas y controle el proceso de aprendizaje.
- ✓ Planifique y dirija la evaluación del proceso (Atsusi Hirumi, 2010).

Se considera de vital importancia la inclusión de un diseño instruccional en los laboratorios virtuales, ya que debe establecerse un equilibrio entre los aspectos del aprendizaje que es evaluado en el resultado final y los aspectos de producción del producto, que no siempre se logra.

En la primera versión de Laboratorios Virtuales, resultaron insuficientes los guiones didácticos y técnicos y fue necesario reajustar posteriormente como valorar el desempeño de los estudiantes, ya que fue importante no solo contabilizar los errores del estudiante en cada escenario de aprendizaje, sino, las posibles secuencias en que se podían efectuar las tareas experimentales en el escenario.

1.6. Los motores gráficos

El término motor gráfico es ampliamente utilizado en todo el mundo principalmente por programadores de videojuegos. El trabajo de un motor gráfico es el de realizar todas las tareas de

bajo nivel tales como comunicarse con el adaptador gráfico, administrar la escena, transformar la geometría del mundo 3D ó 2D y lidiar con diversos procesos matemáticos que afectan su comportamiento. Todo este trabajo es necesario, pero es algo con lo cual el programador inexperto no desea lidiar al desarrollar una aplicación y es entonces cuando se recurre a un motor gráfico en el cual ya se encuentra implementadas las funcionalidades requeridas. El motor del juego es el elemento más importante de todo proyecto ya que está compuesto de todas las instrucciones necesarias para la representación del videojuego, incluyendo imagen, inteligencia artificial, sonido, animación y detector de colisiones.

El motor gráfico, engine o core es la parte de un programa que controla, gestiona y actualiza los gráficos en tiempo real.

Entre los engines más utilizados están los motores de Quake 3 y Unreal tournament a partir de los actuales se han desarrollado el Medal of Honor o Clive Barker´s Undying entre otros. El motor gráfico es un aspecto que se tiene en cuenta desde el primer momento del desarrollo de la aplicación que lo requiera. Un motor gráfico se puede utilizar ya desarrollado, tanto comercial como libre, o crear uno nuevo, lo primero es lo más fácil, pero si se busca que sea de calidad los engines comerciales son bastante caros.

1.6.1. Comparación de los factores considerados en los motores gráficos estudiados

Tabla 1: Comparación de los motores gráficos

Engines	Multiplataforma	2D/3D	Licencias	Costos	Lenguajes soportados	Editor de Escenas
Ogre3D	Si	Si			C++	No
Shiva3D	Si	Si	• Personal Learning (N/A) • Basic (por máquina) • Advanced (por máquina)	• Sin costos • 169 Euros • 1499 Euros	Lua, C++	Si
Cafu3D	Si	Si	GPL	Sin costos	C++, Lua	Si
Löve2D	Si	No	ZLIB	Sin costos	C++, Lua	No
GeexLAB	Si	Si	• Freeware • Pro	• Sin costos • 20 Euros	C++, Lua, Python	Si

Conclusiones parciales del capítulo

Con la conclusión de este capítulo se dieron a conocer un conjunto de elementos que marcan el punto de partida desde el cual se comenzará el desarrollo del diseño instruccional de los laboratorios virtuales, teniendo como meta que se cumpla con los objetivos propuestos. Se realizó un estudio de los entornos virtuales de aprendizaje enfocándose principalmente en los laboratorios virtuales, siendo estos unos de los más usados actualmente. Se hizo una descripción de los lenguajes interpretados y motores gráficos más utilizados así como la comparación de estos últimos. Se dio a conocer la necesidad de incluir el diseño instruccional en los laboratorios virtuales puesto que facilitaría el aprendizaje del estudiante.

CAPÍTULO 2: CONSTRUCCIÓN DE LA SOLUCIÓN

En este capítulo se dará a conocer las herramientas así como lenguajes de programación y modelado a utilizar. Se tratarán los conceptos más importantes mediante un modelo de dominio. Se describen las funcionalidades que debe realizar el sistema mediante los requisitos funcionales así como las características del mismo expresadas mediante los requisitos no funcionales. También se definen los actores del sistema así como los casos de uso del sistema y la descripción de los mismos.

2.1. Metodología y Herramientas a utilizar

2.1.1. Metodología RUP

El Proceso Unificado Racional (RUP por sus siglas en ingles, Rational Unified Process) es un proceso para el desarrollo de software. Se divide en 4 fases y 9 flujos de trabajo, dentro de las cuales se realizan varias iteraciones según el proyecto y en las que se hace mayor o menor esfuerzo en las distintas actividades. Dentro de cada una de ellas el equipo de trabajo pasa por todos los flujos que son transversales a las fases, inclusive en varias iteraciones. El resultado final de un ciclo es una versión del sistema. Es una metodología pesada que requiere un equipo de desarrollo amplio, no necesita que el cliente forme parte del mismo. Facilita la superación de los miembros que componen el grupo de trabajo.

Las principales bibliografías que abordan esta metodología coinciden en que RUP presenta las siguientes características principales:

Iterativo e incremental: Donde cada fase se desarrolla en iteraciones, de forma tal que se pueda dividir en pequeños proyectos mejorando su comprensión y desarrollo. RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente unos más que otros.

Dirigido por casos de uso: Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso.

Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos

del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente.

2.1.2. Herramientas

Visual Paradigm se usará como herramienta de modelado pues es una potente herramienta CASE multiplataforma, que utiliza como lenguaje de modelado el UML. Soporta el ciclo de vida completo del desarrollo de software, ayuda a una rápida construcción de aplicaciones de calidad y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Presenta algunas ventajas entre las que se encuentran:

- ✓ Entorno de creación de diagramas para UML 2.0 y 2.1.
- ✓ Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Capacidades de ingeniería directa e inversa.
- ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- ✓ Disponibilidad de múltiples versiones, para cada necesidad.
- ✓ Disponibilidad en múltiples plataformas.
- ✓ Producto de calidad.
- ✓ Soporta aplicaciones Web.
- ✓ Varios idiomas.
- ✓ Fácil de instalar y actualizar.
- ✓ Compatibilidad entre ediciones.

Para el desarrollo del diseño instruccional se utilizó la herramienta **CompendiumLD**, que permite ajustar el trabajo de los diseñadores del aprendizaje y el equipo de producción y junto al resto de los artefactos de producción convencionales, contribuye a ganar claridad en los resultados finales de los productos generados.

CompendiumLD, permite asumir las complejidades inherentes al diseño instruccional y permite sortear con rapidez, las posibles indefiniciones de los guiones y asegura, hacia el cliente, una idea más clara de las tareas que cumple el estudiante, cuáles materiales deben generarse de acuerdo a los objetivos declarados y en determinar los tiempos permisibles en que deben ejecutar las tareas los estudiantes, de acuerdo al nivel de las competencias que poseen y valorar las mismas.

Shiva3D es un motor3D con un editor gráfico diseñado para crear fácilmente aplicaciones y juegos de vídeo para la web, consolas y dispositivos móviles en tiempo real por el hecho de poseer un editor de tipo WYSIWYG.

WYSIWYG es el acrónimo de *What You See Is What You Get* (en español, "lo que ves es lo que obtienes"). Se aplica a los procesadores de texto y otros editores de texto con formato que permiten escribir un documento viendo directamente el resultado final, frecuentemente el resultado impreso.

Shiva3D puede producir juegos en 3D y simulaciones gráficas para Windows, Mac, Linux, iPhone, IPAD, Android, Palm OS, Wii y WebOS, independiente o integrados en los navegadores web (Internet Explorer, Firefox, Safari, Mozilla, Netscape, etc.).

El motor del juego utiliza OpenGL, DirectX y también se puede ejecutar en modo de software. ShiVa3D también es compatible con los estándares de la industria plug-ins como NVIDIA PhysX, Mod y ARToolKit.

Características más notables:

- ✓ Diseñado para crear cualquier género de juego en una fracción del tiempo.
- ✓ Motor de juego multiplataforma diseñado para el desarrollo de juegos de próxima generación.
- ✓ Ofrece todas funciones de programación, de renderizado, animación y efectos especiales necesarios para crear cualquier juego con facilidad.
- ✓ Motor gráfico con optimización de iluminación dinámica y sombras.
- ✓ Herramienta unificada de edición, compila código fuente generado por el editor en una aplicación ejecutable para todas disímiles plataformas.
- ✓ Ofrece un integrado estándar de la industria plug-ins para ampliar las capacidades del motor.
- ✓ Provee ODE motor de física.
- ✓ Alto nivel de desarrollo utilizando lenguaje Lua.
- ✓ Los juegos pueden ser codificados por completo en C, C++, Cocoa y Objective-C.
- ✓ Presenta un editor de tipo WYSIWYG.

- ✓ Licencia única para todas sus plataformas.

2.1.3. Lenguajes

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. Prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real.

Está compuesto por diversos elementos gráficos que se combinan para conformar diagramas; es un lenguaje por lo que cuenta con reglas para combinar tales elementos. UML no es una guía para realizar el análisis y diseño orientado a objetos, es decir, no es un proceso. Es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos (Larman, 2004).

Lua se escogió como lenguaje de extensión a utilizar de entre los estudiados, por ser el que más se adapta a las exigencias requeridas para esta propuesta. Es el lenguaje de extensión por excelencia en el mundo de los videojuegos por sus notables características positivas, podemos destacar entre algunas su posibilidad de embeberse con otros lenguajes de alto nivel como C++, lo potente, ligero y rápido que es, con respecto a los demás lenguajes de extensión abordados y la facilidad y claridad de su sintaxis.

2.2. Diseño instruccional del laboratorio virtual

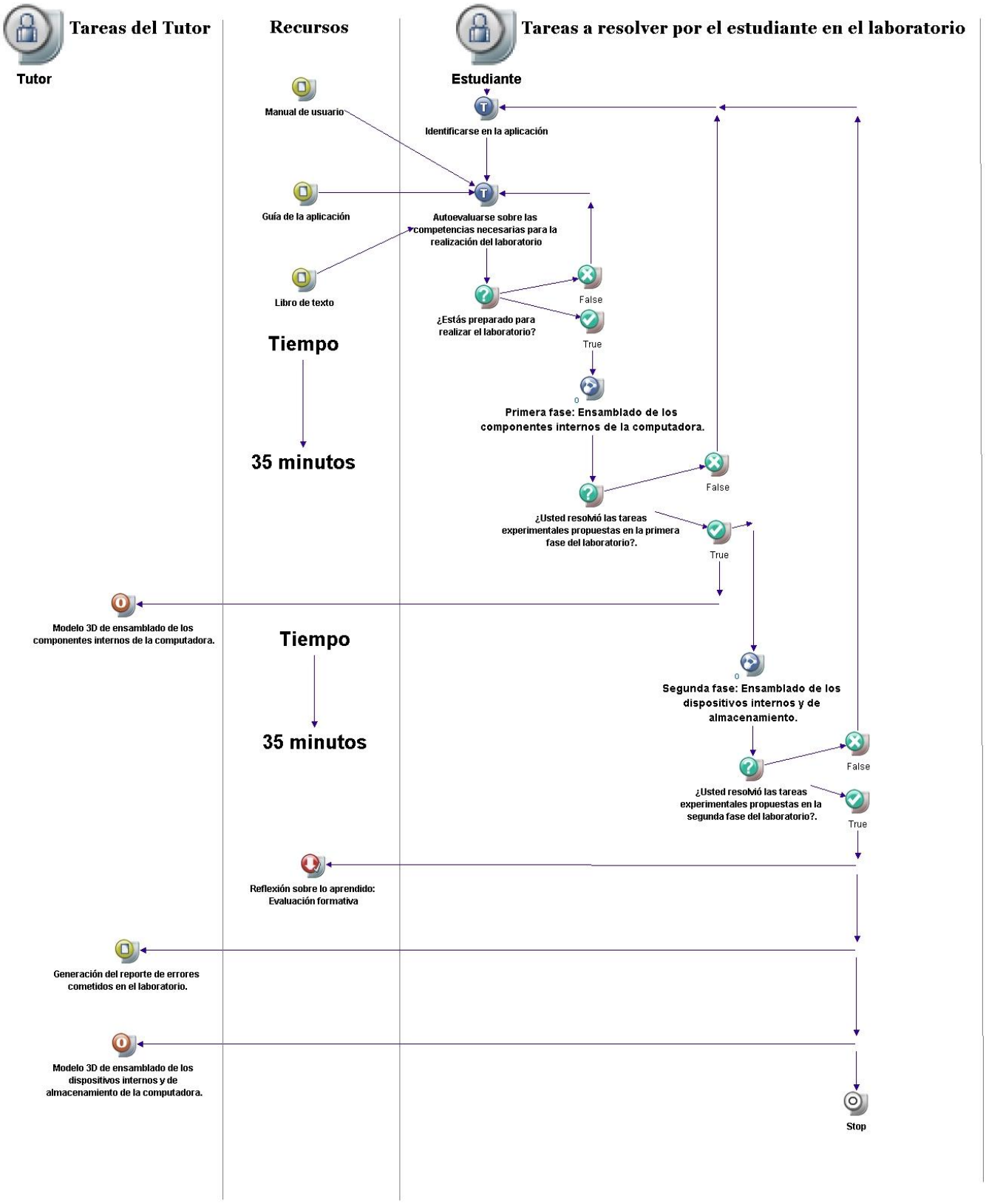

Figura 1: Diseño instruccional

2.3. Modelo de dominio

Una vez analizado el negocio se comprueba que los procesos existentes no se encuentran bien definidos, se identificaron los principales conceptos presentes y se decide realizar un Modelo de Dominio, los cuales se presentarán relacionados en un marco conceptual. Además, este modelo permitirá de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio de los procesos actuales del negocio. Esto ayudará a los usuarios, clientes, desarrolladores e interesados, a utilizar un vocabulario común para poder entender el contexto en que se sitúa el sistema. Se identificaron los siguientes conceptos:

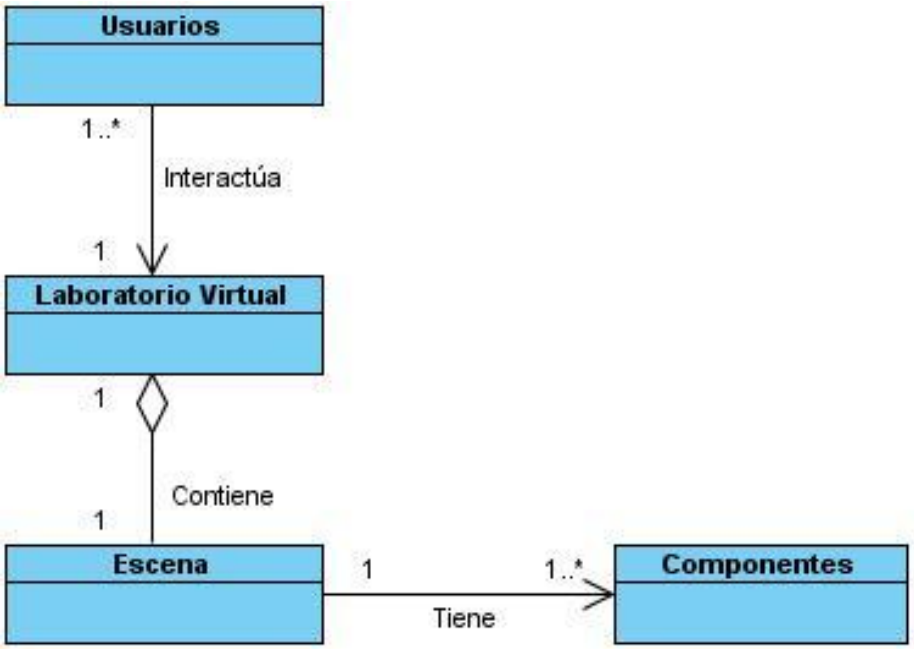

Figura 2: Modelo de dominio

2.4. Captura de requisitos

El flujo de trabajo Requerimientos define las capacidades o condiciones que debe cumplir el sistema como requisitos funcionales y las restricciones impuestas o propiedades del sistema como requisitos no funcionales. A continuación se describe las funcionalidades y características del sistema.

2.4.1. Requisitos Funcionales

- ✓ **RF 1. Autenticar usuario:** El sistema debe permitir autenticar el usuario que desee entrar a la aplicación.
- ✓ **RF 2. Responder Cuestionario:** El sistema debe permitirle a los estudiantes que respondan los cuestionarios que se les presente.

- ✓ **RF 3. Seleccionar Componente:** El sistema debe permitir seleccionar el componente que el estudiante desee para realizar sus actividades.

2.4.2. Requisitos No Funcionales

Apariencia e interfaz externa

Se debe proporcionar un ambiente agradable y fácil para el usuario.

Usabilidad

El acceso a la aplicación debe realizarse de forma fácil y rápida. Para utilizar el sistema es necesario poseer conocimientos elementales de computación.

Rendimiento

La capacidad de respuesta de la aplicación debe ser la más rápida posible.

Soporte

La aplicación podrá ser ejecutada en varias plataformas.

Software

La aplicación permite ejecutarse en cualquier versión de los sistemas operativos MS Windows, así como en MAC OS.

Hardware

Requerimientos mínimos:

- ✓ Intel Pentium IV 2Ghz o AMD ATLON XP 2600+.
- ✓ 512 megabytes (MB) de RAM.
- ✓ Tarjeta gráfica con 64MB mínimo de RAM.
- ✓ Resolución 1440x900.

Requerimientos recomendados:

- ✓ Intel Core Dúo 1.8Ghz o AMD Athlon 64 X2 3600+.
- ✓ 1024 megabytes (MB) de RAM.
- ✓ Tarjeta gráfica con 256MB mínimo de RAM (NVidia Geforce 6600 o ATI X600).

- ✓ Resolución 1680x1050.
- ✓ Dispositivos de audio.

Disponibilidad

La aplicación debe estar disponible a tiempo completo y debe recuperarse rápidamente ante cualquier tipo de fallo.

Restricciones de diseño e implementación

La aplicación será implementada usando el motor gráfico Shiva3D. Como lenguaje de programación se utilizará Lua. Se utilizará un estándar de codificación para la implementación.

Portabilidad

La aplicación permite ejecutarse en los sistemas operativos Windows y Linux.

2.5. Modelo de Casos de Uso del Sistema

En esta sección se reconocen los actores del sistema a desarrollar, y se definen los casos de uso del sistema. Además, se seleccionan los casos de uso correspondientes al primer ciclo de desarrollo para hacerles sus especificaciones textuales en formato expandido.

2.5.1. Actores del Sistema

Los actores del sistema son agentes externos, roles que los usuarios o dispositivos juegan cuando interactúan con el software. En este caso se tiene un actor del sistema el cual es el estudiante.

Tabla 2: Actor del sistema

Actor	Descripción
Estudiante	Es quien interactúa con el sistema para realizar las actividades del laboratorio virtual.

2.5.2. Diagrama de Casos de Uso del Sistema

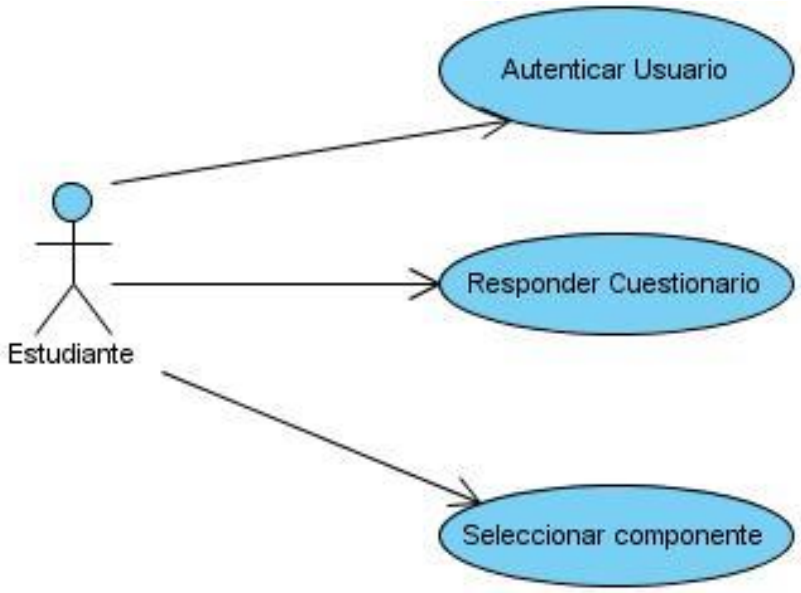

Figura 3: Diagrama de casos de uso

2.5.3. Descripción de Casos de Uso del Sistema

Cada caso de uso tiene una descripción de las funcionalidades que ejecutará el sistema propuesto como respuesta a las acciones del usuario. Las tablas presentadas a continuación argumentan los flujos operacionales de cada caso de uso.

Tabla 3: Descripción Caso de Uso Autenticar Usuario

Caso de Uso:	Autenticar Usuario
Actores:	Estudiante
Resumen:	El caso de uso comienza cuando el actor ejecuta el sistema y este le brinda la posibilidad de autenticarse.
Referencias	RF 1
Prioridad	Crítica
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra un cuadro de diálogo para que el usuario se autentique.

1.1 El usuario se autentica. 2. Oprime el botón aceptar.	2.1. Cierra el cuadro de diálogo y procede a autenticar al usuario.
Prototipo de Interfaz	
Pos condiciones	

Tabla 4: Descripción Caso de Uso Responder Cuestionario

Caso de Uso:	Responder Cuestionario
Actores:	Estudiante
Resumen:	El caso de uso se inicia cuando el sistema le presenta al actor el cuestionario que debe resolver. El actor procede a realizar dicho cuestionario.
Precondiciones:	Debe haberse ejecutado el caso de uso Autenticar Usuario.
Referencias	RF 1, RF 2
Prioridad	Crítica
Flujo Normal de Eventos	
Acción del Actor	**el Respuesta del Sistema**
	1. Muestra el cuestionario.
1.1 Realiza el cuestionario.	
Prototipo de Interfaz	
Pos condiciones	

Tabla 5: Descripción Caso de Uso Seleccionar Componente

Caso de Uso:	Seleccionar Componente.
Actores:	Estudiante
Resumen:	El caso de uso se inicia cuando el sistema muestra los componentes a seleccionar. El actor procede a seleccionar el componente que desee.
Precondiciones:	Deben haberse ejecutado los casos de uso Autenticar Usuario y Responder

	Cuestionario.
Referencias	RF 1, RF2, RF3
Prioridad	Crítica
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra los componentes que debe escoger el usuario.
1.1 Selecciona el componente que desee. **2.** Da clip al componente.	**2.**1 Procede a colocar el componente escogido encima de la mesa.
Prototipo de Interfaz	
Pos condiciones	

2.6. Conclusiones Parciales del Capítulo

Con la realización de este capítulo se logra desarrollar un análisis crítico de los procesos involucrados en el Campo de acción. La utilización de la metodología RUP facilitará el seguimiento y monitoreo de un proceso de desarrollo que agilice el trabajo de los implicados en el mismo, de ahí que la obtención de un producto con la calidad requerida, sea un hecho seguro. Se selecciona Visual Paradigm como herramienta para el modelado, principalmente porque utiliza a UML como lenguaje de modelado, permite modelar todo el ciclo de desarrollo del software y es multiplataforma. La utilización de las herramientas y lenguajes de programación escogido permitirá el desarrollo claro y fluido de un sistema construido sobre bases sólidas y un entorno de desarrollo bien definido. Se define la propuesta de solución del sistema. Se presenta un listado de RF y RNF, donde se recogen las principales necesidades del sistema a desarrollar. Estas necesidades fueron traducidas a un conjunto de CU, las cuales representan las principales funcionalidades del sistema.

CAPÍTULO 3: SOLUCIÓN PROPUESTA

En este capítulo se dará una explicación de la estructura utilizada para el desarrollo de la solución. Se analizará las formas de exportación que brinda el motor gráfico y como sería la exportación seleccionada. Se realizará una descripción del sistema propuesto.

3.1. Estructura de trabajo utilizada

Como casi todos los motores gráficos existentes, Shiva3D posee una estructura propia de trabajo. En Shiva3D se define una estructura completa para el desarrollo de cualquier aplicación 3D de una forma muy sencilla y rápida. A continuación se muestra la arquitectura general que utiliza el engine en su trabajo:

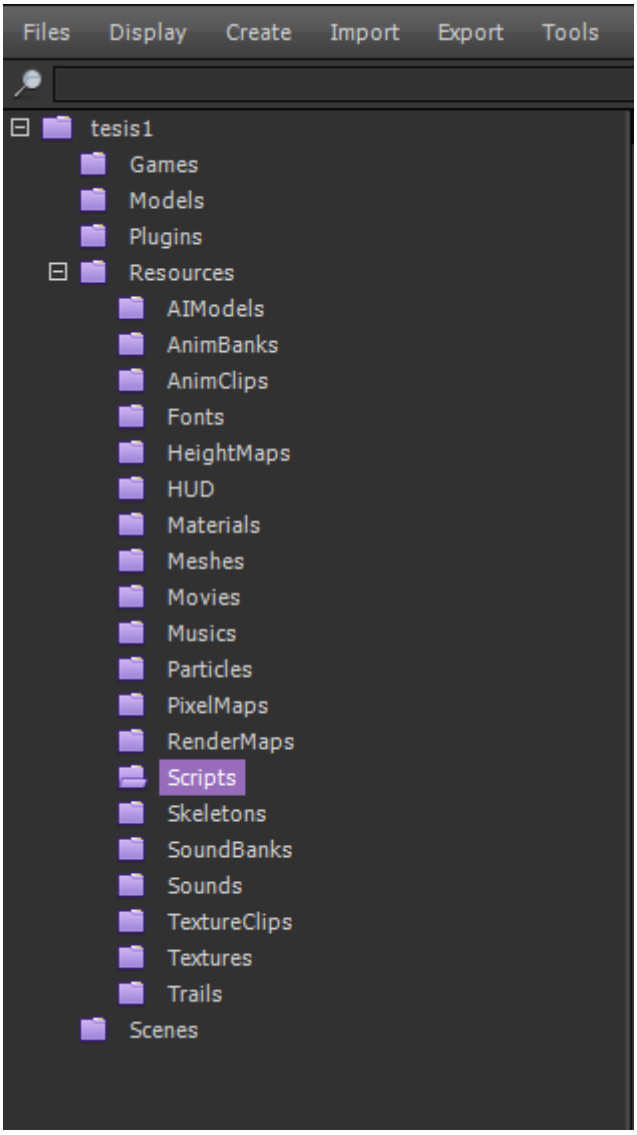

Figura 4: Estructura de trabajo de Shiva3D.

Como se pudo apreciar, Shiva3D divide el trabajo en subcarpetas donde cada una se especializa en controlar una función con un editor específico para cada una de ellas, de ahí la rapidez, facilidad, pero sobre todo la alta calidad de la edición por la gran cantidad de opciones y funcionalidades que se brindan en el trabajo, por el hecho de tener cada editor especializado en una función determinada. Por ejemplo para incluir un material en su escena el proceso sería el siguiente.

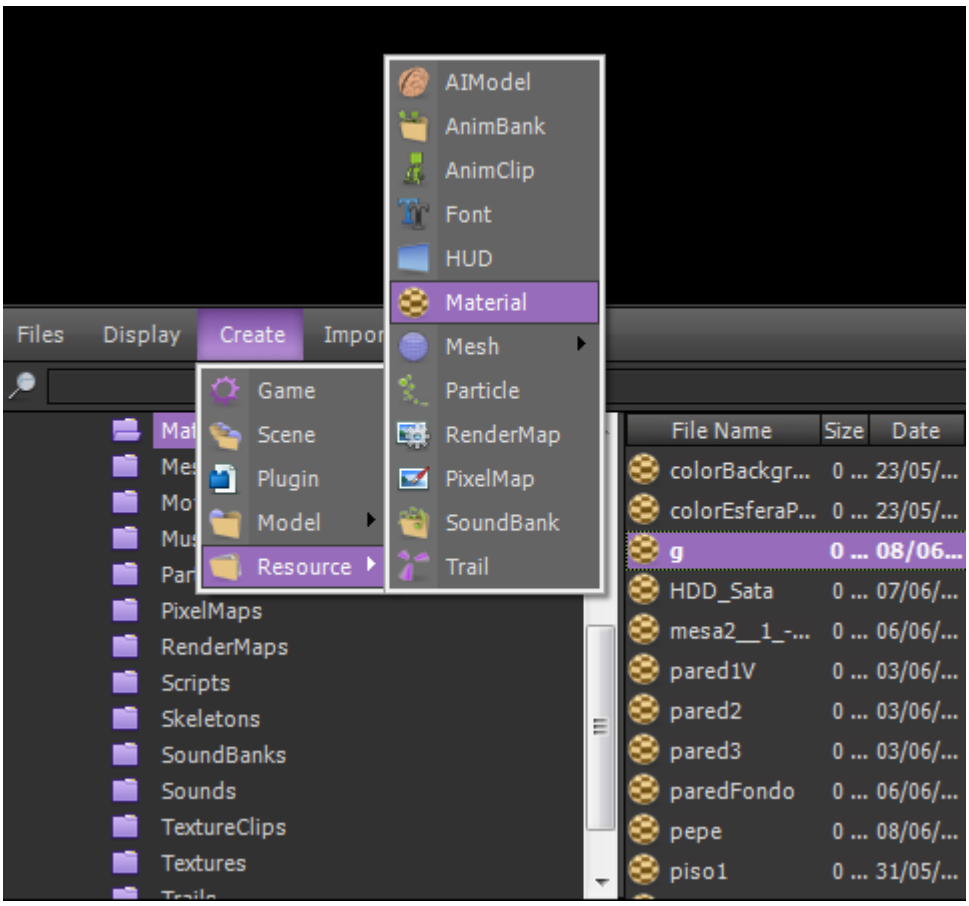

Figura 5: Forma de creación de un material en Shiva3D.

Como se ve en la figura de una manera muy sencilla se crea un material y este pasa a formar parte dentro de la carpeta donde se contienen todos los materiales utilizados en el proyecto. Una vez allí se utiliza el editor encargado del manejo del material como puede ser entre otras, cargar el material desde una dirección, aplicarle iluminación y niebla. A continuación se muestra el editor específico de los materiales.

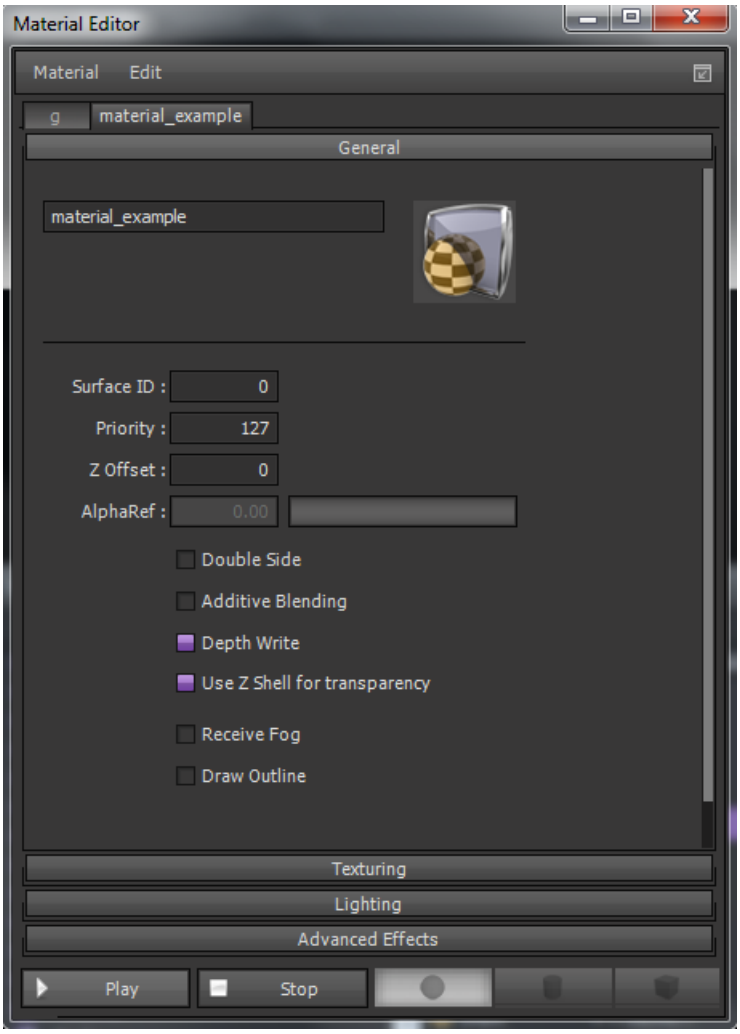

Figura 6: Editor característico de los materiales.

De igual manera que la carpeta *Materials* con su editor característico para el manejo específico de los materiales, son tratadas las demás carpetas y subcarpetas que conforman la arquitectura o estructura que utiliza Shiva3D, encargadas de sus funciones características, entre ellas se encuentran entre otras:

- ✓ *Movies*: Para los videos a utilizar en la aplicación.
- ✓ *Musics*: Para los diferentes tipos de música utilizados.
- ✓ *Sounds*: Para los demás sonidos.
- ✓ *Textures*: Para las texturas de los objetos en la escena.
- ✓ *Scripts*: Donde se encuentran los scripts Lua que definen las diferentes funcionalidades del sistema.
- ✓ *AIModels*: En esta carpeta se realiza la asignación de inteligencia artificial a los diferentes objetos presentes en la escena y las cámaras.

3.1.1. Editor de trabajo de tipo WYSIWYG

Anteriormente en la selección de la herramienta Shiva3D para implementar la solución, se pudo conocer que el motor gráfico está basado en la edición de tipo WYSIWYG. El hecho de ir viendo la solución final que va a tener la aplicación e ir construyendo el sistema según se van logrando resultados hace el trabajo mucho más sencillo y al trabajar con el lenguaje de extensión Lua que posee una sintaxis clara y sencilla con respecto a lenguajes avanzados como C++ utilizado en motores gráficos como OGRE, hacen que el desarrollo de aplicaciones como el sistema propuesto, pueda ser llevado a cabo por estudiantes de menos nivel en programación como estudiantes del 2do y 3er año de la carrera.

3.1.2. Funcionalidades básicas predefinidas

Una de las facilidades que posee el motor gráfico es el hecho de contener funcionalidades previas para realizar actividades que son propias de este tipo de aplicaciones y ya no son necesarias implementaciones solo bastaría su uso o modificación, muchas de estas funcionalidades fueron usadas en el desarrollo de la aplicación. Esto constituye otra de las ventajas de esta propuesta el hecho de ahorrar tiempo de desarrollo mediante el uso de muchas funcionalidades que ya provee el propio *engine*.

3.2. Posibilidades en la exportación

Uno de los problemas más frecuentes que presentan los softwares es el hecho de ser incompatibles con algunos sistemas operativos que difieren de los utilizados en su desarrollo. Entre las posibilidades que brindan algunos motores gráficos, se encuentra la presencia de una aplicación encargada de exportar la solución implementada. En el caso del motor gráfico shiva3D, se cuenta con la herramienta *Shiva 3D Authoring Tools* que ofrece posibilidades de exportar a varios sistemas operativos, dispositivos Wii, dispositivos móviles y la Web. Es muy importante tomar en cuenta estas funcionalidades brindadas por el engine propuesto, ya que permite de una manera muy sencilla y sin necesidad de re-implementar, modificar su proyecto de acuerdo a los requerimientos de hardware y software, e incluso ofrecer exportado a Web, donde solo se necesitaría del uso de un navegador que bien puede venir con su sistema operativo.

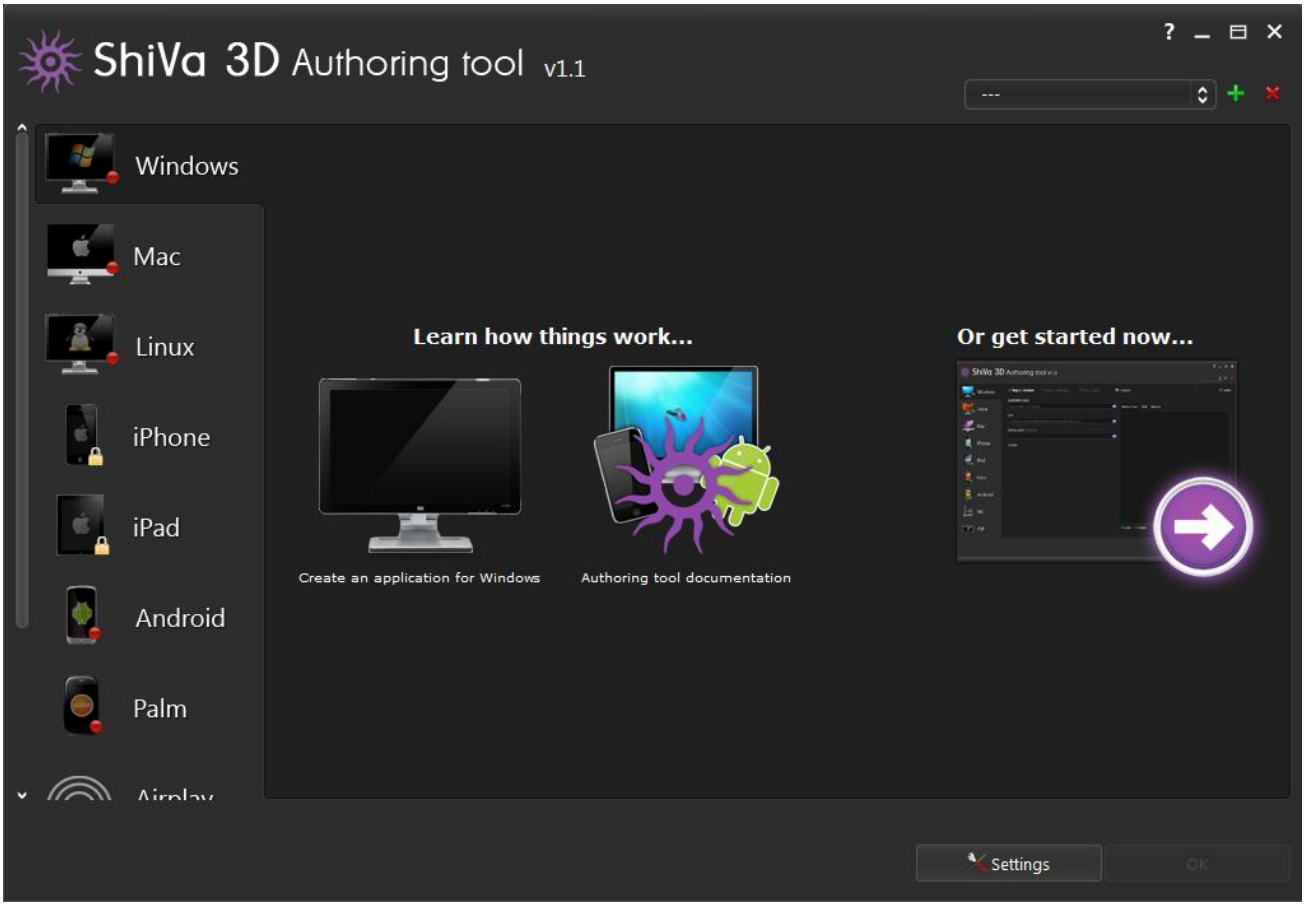

Figura 7: Shiva3D Authoring Tools

Esta aplicación carga la solución en el formato exportado por el motor gráfico y se encarga de convertirla al sistema operativo o dispositivo seleccionado. Shiva 3D permite, como se muestra en la figura #, exportar sus soluciones de proyecto, que en su caso denomina como *games,* como paquetes ejecutables (**.stk*), es el que usa por defecto, en archivos ejecutables (**.ste*), en ficheros separados y en paquetes ejecutables (*.stk con código en C++).

El proceso de exportación con la herramienta Shiva Authoring Tools en el caso del demo propuesto en este trabajo sería el siguiente:

Para exportar la solución a una aplicación ejecutable de Windows en este caso, consiste de tres pasos, (1) *Content* donde se selecciona el proyecto a exportar, con algunas opciones como la de añadir un icono al ejecutable. (2) *Authoring* donde se seleccionan las opciones de formato donde se puede exportar la aplicación como un ejecutable directo o incluso construirlo como un instalador, resolución de pantalla, nivel de definición, sombras, entre otras. (3) *Build* y el tercer paso donde se escoge directorio para guardar la construcción y se construye la aplicación ejecutable de Windows. A continuación imágenes de esta secuencia:

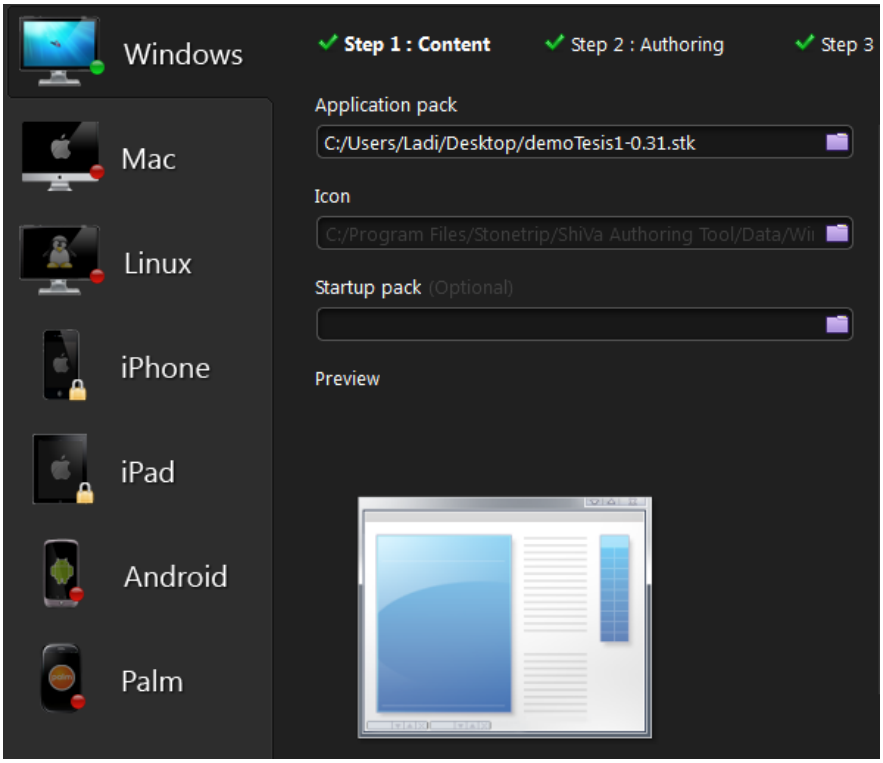

Figura 8: Exportación a sistema operativo Windows, fase 1.

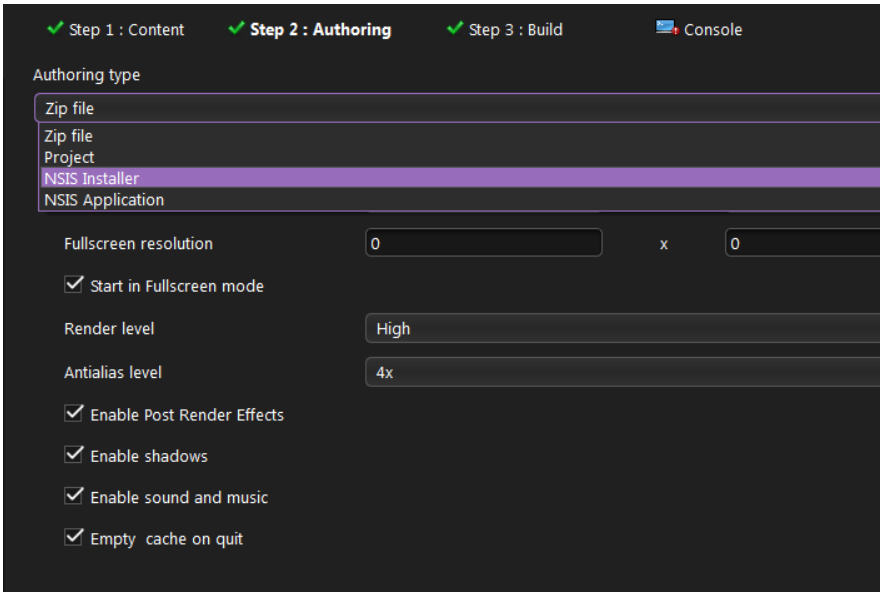

Figura 9: Exportación a sistema operativo Windows, fase 2.

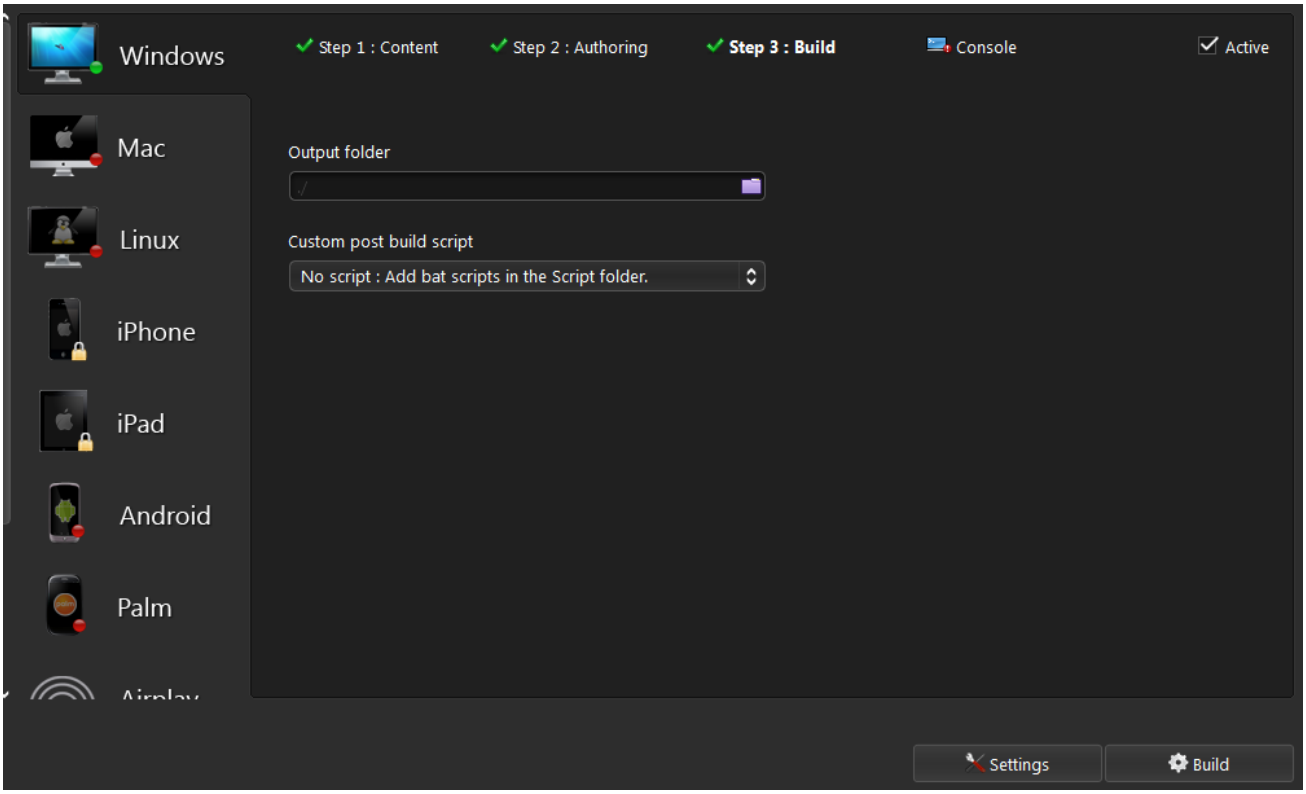

Figura 10: Exportación a sistema operativo Windows, fase 3.

3.3. Propuesta de solución desarrollada

El sistema propuesto cuenta con un diseño semejante al utilizado por los videojuegos, donde se limpia la escena de trabajo y se muestran los mínimos componentes posibles. Se evita el uso de un marco siempre visible con menús y barras. Los componentes o herramientas del laboratorio se muestran en un menú flotante en la parte posterior derecha de la escena con el estilo utilizado en muchos juegos de roles para la selección y manipulación de los elementos del juego.

Se adoptó una interactividad teclado-mouse, donde se puede transitar por la escena con el uso de las teclas de desplazamiento del teclado y el mouse para la selección de los componentes a manejar en la escena. Mediante el uso de la tecla *escape* del teclado se cambia de la actividad del teclado al uso del mouse y viceversa para lograr un tiempo de reflexión por parte del estudiante en la actividad que está desempeñando.

La solución está compuesta por una primera parte donde se muestra una forma que corresponde a la acreditación del estudiante, es una especie de *login* para pasar a la segunda fase del laboratorio. Una vez que el estudiante se acredita pasa a responder una serie de preguntas como preparación previa antes de comenzar la práctica de laboratorio y una vez que el estudiante haya respondido todas las preguntas comienza la práctica dentro de la escena.

Figura 11: Diseño del laboratorio virtual propuesto.

Conclusiones parciales del capítulo

Al concluir este capítulo se puedo apreciar de manera general la estructura utilizada en el desarrollo de la aplicación según la propuesta que brinda Shiva3D, así como el proceso de creación de un elemento en la escena. Se mostraron las posibilidades de exportación brindadas por la herramienta Shiva3D Authoring Tools y se ejemplificó la exportación de la solución hacia el sistema operativo Windows. Culmina el capítulo con el diseño propuesto para este tipo de aplicaciones de aprendizaje.

CONCLUSIONES GENERALES

Se desarrolló un diseño instruccional para los laboratorios virtuales mediante la herramienta CompendiumLD posibilitando una guía para los desarrolladores.

Se obtuvo un demo que muestra las grandes ventajas que conlleva realizar laboratorios virtuales con el uso del lenguaje de extensión Lua y el motor gráfico Shiva3D, permitiendo así disminuir el tiempo de implementación.

RECOMENDACIONES

Se proponen las siguientes recomendaciones:

- ✓ Culminar el desarrollo del laboratorio en su totalidad en cuanto al diseño instruccional propuesto, para explotar muchas otras funcionalidades del motor gráfico Shiva3D.
- ✓ Que se evalúen las posibilidades que brinda esta solución, demostrada con el desarrollo de este trabajo, por parte del Proyecto de Laboratorios Virtuales en futuros desarrollos de aplicaciones de este tipo.

REFERENCIAS BIBLIOGRÁFICAS

Atsusi Hirumi, Bob Appelman, Lloyd Rieber, Richard Van Eck. 2010. *Preparing Instructional Designers for Game-Based. Learning: Part III Game Design as a Collaborative Process.* 2010.

Avila, Patricia. 1999. *Ambientes Virtuales de Aprendizaje, una nueva experiencia.* 1999.

Branch, Gustafson &. 1997. *Survey of Istructional Development Model.* 1997.

Byron, Chaves Barquero. 2008. *Características del lenguaje Perl 5.0 y su aplicación como herramienta de desarrollo en la elaboración de un Servidor Web.* San José, Costa Rica. : Universidad de Costa Rica, Escuela de las Ciencias de la Computación e Informática., 2008.

Dorrego, Elena. 1997. *Investigacion sobre los efectos de los eventos instruccionales en las estrategias de aprendizaje a traves de los medios.* 1997.

Emerich, Paul. 2009. *Beginning Lua with World of Warcraft Addons.* 2009.

Larman, Craig. 2004.. *UML y Patrones.* 2004.

Marchena, Daniel. 2008. *Entornos Virtuales de Aprendizaje.* Venezuela : s.n., 2008.

Pressman. 2001. *Ingeniería del Software un enfoque práctico.* 2001.

Rio, UPC. 1999. The programming Language Lua. *The programming Language Lua.* [En línea] UPC Rio, 1999. [Citado el: 12 de 02 de 2011.] http://www.lua.org.

Valdés, Damián Pérez. 2007. *Los diferentes lenguajes de programación para la web.* 2007.

Bibliografía

Cázares, Santiago Inzunsa. 2010. *Entornos Virtuales de Aprendizaje: Un enfoque alternativo para la enseñanza y el aprendizaje.* 2010.

Ledesma, Pedro Pleguezuelos. 2007. *Creación de juegos con LUA.* 2007.

Lozano, M. 2004. *Entornos Virtuales 3D clasicos e inteligentes: Hacia un nuevo marco de simulacion para aplicaciones graficas 3D interactivas.* 2004.

Luis E. Mendoza, María A. Pérez, Gabriela Díaz-Antón, Anna Grimán. 2005. *Mejoras en RUP para la implementación de aulas virtuales.* 2005.

Martelli, Alex. 2008. *Guía de referencia de Python.* España : s.n., 2008.

Moreno, Mauro Callejas Cuervo y Oscar Yovany Baquero. 2005. *Herramientas libres para modelar software.* 2005.

Olalde., Juan Murua. 2006. *Metodologías para el desarrollo de software.* 2006.

Onrubia, Javier. 2005. *Aprender y enseñar en entornos virtuales:actividad conjunta, ayuda pedagógica y construcción del conocimiento.* Barcelona : s.n., 2005.

Río, UPC. The Programming Language Lua. *The Programming Language Lua.* [En línea] http://www.lua.org/.

S. Dormido, J. Sánchez, F. Morilla. 2005. *Laboratorios Virtuales y remotos para la práctica de la automática.* Madrid, España. : s.n., 2005.

Schach, Stephen R. 2007. *Análisis y Diseño Orientado a Objetos con UML y el Proceso Unificado.* 2007.

Stonetrip. 2003-2010. Shiva3D 3D Game Engine with Development Tools. *Shiva3D 3D Game Engine with Development Tools.* [En línea] 2003-2010. http://www.stonetrip.com/.

GLOSARIO DE TÉRMINOS

Android. Sistema operativo basado en Linux diseñado originalmente para dispositivos móviles, tales como teléfonos inteligentes, actualmente se encuentra en desarrollo para usarse en netbooks y PCs.

ANSI. El Instituto Nacional Estadounidense de Estándares (ANSI, por sus siglas en inglés: American National Standards Institute) es una organización sin ánimo de lucro que supervisa el desarrollo de estándares para productos, servicios, procesos y sistemas.

Applets. Componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador web. El applet debe ejecutarse en un contenedor, que lo proporciona un programa anfitrión, mediante un plugin, o en aplicaciones como teléfonos móviles que soportan el modelo de programación por 'applets'.

CSS. Siglas de Cascading Style Sheets (Hojas de Estilo en Cascada) que es un lenguaje que describe la presentación de los documentos estructurados en hojas de estilo para diferentes métodos de interpretación.

DirectX Es una colección de API desarrolladas para facilitar las complejas tareas relacionadas con multimedia, especialmente programación de juegos y vídeo, en la plataforma Microsoft Windows.

HTML. Siglas de Hypertext Markup Language (Lenguaje de Marcado de Hipertexto), es el lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

IPAD. Dispositivo electrónico tipo Tablet PC desarrollado por Apple Inc. Anunciado el 27 de enero de 2010, el 2 de marzo de 2011 apareció la segunda generación. Se sitúa en una categoría entre un "teléfono inteligente" (Smartphone) y una computadora portátil, enfocado más al acceso que a la creación de contenido.

IPhone. Contracción de *inteligent phone* es una familia de teléfonos inteligentes multimedia con conexión a Internet, pantalla táctil capacitiva y una interfaz de software minimalista diseñados por la compañía Apple Inc.

Java. Es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

MIT license. Esta licencia permite reutilizar el software así licenciado tanto para ser software libre como para ser software no libre, permitiendo no liberar los cambios realizados al programa original.

Mod. En el mundo de los videojuegos un **mod** (del inglés *modification*) es una extensión que modifica un juego original proporcionando nuevas posibilidades, ambientaciones, personajes, diálogos, objetos, etc.

MSDOS. Siglas de Microsoft Disk Operating System (Sistema operativo de disco de Microsoft), es un sistema operativo para computadores basados en x86. Fue el miembro más popular de la familia de sistemas operativos DOS de Microsoft, y el principal sistema para computadoras personales compatible con IBM PC en la década de 1980 y mediados de 1990, hasta que fue sustituida gradualmente por sistemas operativos que ofrecían una interfaz gráfica de usuario, en particular por varias generaciones de Microsoft Windows.

Netscape Communications. Netscape Communications Corporation es una empresa de software famosa por ser la creadora del navegador web Netscape Navigator. Fue comprada por AOL en 1999.

NVIDIA PhysX. PhysX es un motor propietario de capa de software intermedia o "middleware" y un kit de desarrollo diseñados para llevar a cabo cálculos físicos muy complejos. Conocido anteriormente como la SDK de NovodeX, fue originalmente diseñado por AGEIA y tras la adquisición de AGEIA, es actualmente desarrollado por Nvidia e integrado en sus chips gráficos más recientes.

Objective-C. Lenguaje de programación orientado a objetos creado como un superconjunto de C pero que implementase un modelo de objetos parecido al de Smalltalk. Originalmente fue creado por Brad Cox y la corporación StepStone en 1980.

ODE. Es una colección de alto rendimiento y código abierto para la simulación de cuerpos rígidos. Presenta una estructura estable, madura y fácil de utilizar en aplicaciones API de C y C++.ODE es útil para la simulación de vehículos, objetos en entornos de realidad virtual y criaturas virtuales. En la actualidad se utiliza en muchos juegos de ordenador, herramientas de creación 3D y herramientas de simulación.

OGRE. Acrónimo del inglés Object-Oriented Graphics Rendering Engine, es un motor de renderizado 3D orientado a escenas, escrito en el lenguaje de programación C++. Sus bibliotecas evitan la dificultad de la utilización de capas inferiores de librerías gráficas como OpenGL y Direct3D, y además, proveen una interfaz basada en objetos del mundo y otras clases de alto nivel. El motor es software libre, licenciado bajo MIT y con una comunidad muy activa.

OpenGL. (Open Graphics Library) es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. La interfaz consiste en más de 250 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples, tales como puntos, líneas y triángulos.

Palm OS. Es un sistema operativo que fue hecho por PalmSource, Inc. para computadores de mano (PDAs) fabricados por varios licenciatarios.

Pascal. Es un lenguaje de programación desarrollado por el profesor suizo Niklaus Wirth entre los años 1968 y 1969 y publicado en 1970. Su objetivo era crear un lenguaje que facilitara el aprendizaje de programación a sus alumnos, utilizando la programación estructurada y estructuración de datos. Sin embargo con el tiempo su utilización excedió el ámbito académico para convertirse en una herramienta para la creación de aplicaciones de todo tipo.

Plug-ins. Un plug-in es un módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

Psicología cognitiva. La psicología cognitiva es una escuela de la psicología que se encarga del estudio de la cognición, es decir, los procesos mentales implicados en el conocimiento. Tiene como objeto de estudio los mecanismos básicos y profundos por los que se elabora el conocimiento, desde la percepción, la memoria y el aprendizaje, hasta la formación de conceptos y razonamiento lógico.

Shell. Programas que proveen una interfaz de usuario para acceder a los servicios del sistema operativo.

Storyboards. Guiones gráficos.

UNIX. Unix (registrado oficialmente como UNIX®) es un sistema operativo portable, multitarea y multiusuario; desarrollado, en principio, en 1969 por un grupo de empleados de los laboratorios Bell de AT&T, entre los que figuran Ken Thompson, Dennis Ritchie y Douglas McIlroy.

WebOS. Es un sistema operativo multitarea para sistemas embebidos basado en Linux, desarrollado por Palm, Inc.

Wii. Es la sexta videoconsola de sobremesa producida por Nintendo. Fue desarrollada en colaboración con IBM y ATI. Es la sucesora directa de la Nintendo GameCube y compite actualmente contra la Playstation 3 de Sony y la Xbox 360 de Microsoft como parte de las videoconsolas de séptima generación.