UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Guía de aprendizaje para el desarrollo de escenarios virtuales tridimensionales interactivos en la Web.

Autores: Jesús Lázaro Suárez Pérez.

Osvel Vargas Torres.

Tutores: Ing. Jorge Antonio Díaz.

Ing. Gilberto Cao Tarrero.

La Habana, 2011

DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores del presente trabajo de diploma y reconocemos a la Facultad 4 de la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo, con carácter exclusivo.

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

Jesús Lázaro Suárez Pérez

_____

[Firma autor]

Osvel Vargas Torres

_____

[Firma autor]

Ing. Jorge Antonio Díaz.

_____

[Firma tutor]

Ing. Gilberto Cao Tarrero.

_____

[Firma tutor]

DATOS DE CONTACTO

Los autores:

Nombre: Jesús Lázaro

Apellidos: Suárez Pérez.

Correo: jlsuarez@uci.cu

Nombre: Osvel

Apellidos: Vargas Torres.

Correo: ovargas@uci.cu

Los tutores:

Nombre: Jorge Antonio

Apellidos: Díaz Gutiérrez.

Correo: jagutierrez@uci.cu

Nombre: Gilberto

Apellidos: Cao Tarrero.

Correo: gcao@uci.cu

Agradecimientos y Dedicatoria

Osvel

A mi mamá y mi papá que siempre han confiado en mis decisiones y me han apoyado toda la vida. A mi mamá que la quiero con la vida, y a mi papá que siempre ha formado valores de un verdadero hombre y por eso lo admiro tanto y es mi ídolo.

A mi hermana Isvel por confiar en mí e inculcarme siempre ser competente ante todo en la vida.

A mi hermana Arelys que conocí en la universidad y la quiero mucho.

A mis abuelos paternos que los extraño mucho y sé que están orgullosos de mí.

A mi abuelo Luis por su apoyo en los momentos más difíciles. A mi abuela Julia que quiero mucho y creo que heredé muchas de sus virtudes. A mi suegra Mireya que la quiero mucho.

A mi novia Idelaine que me ha apoyado mucho en casi diez años y quisiera que fuera para toda la vida.

A mis grandes amigos y hermanos de la universidad, Adonis y Jesús. A Adonis por ser ese hermano que siempre quise tener, del que he aprendido y me ha apoyado en los momentos más difíciles, por ser un verdadero profesional que admiro. A Jesús, mi compañero en eventos científicos y en la tesis, gracias por luchar juntos y hacer realidad este sueño en el que muchos no creían. A mis sobrinitos Roylan, Liannis, Liliet y Liuver que los quiero mucho, les dedico mi tesis.

A toda mi familia que es bastante grande, les agradezco su apoyo y su confianza.

A mis profesores desde la primaria y que sería casi imposible mencionarlos a todos. Gracias por luchar conmigo por este momento inolvidable.

A William que más que un profesor es un gran amigo que me guió para cumplir este sueño.

A mis compañeros del barrio y de la escuela, de todos siempre aprendí algo nuevo.

A mi prima Yanet y a Carlos Miguel por su apoyo en los primeros años de la carrera.

A mis amigos del pre Leonardo, Pablo, Roberto, Argel, Yordi, Alexey, Roberdani.

Al tribunal por ser críticos, objetivos y por apoyarnos cuando incluso nadie lo hacía.

A Yorgelys por ser nuestra primera guía que nos enseñó los primeros pasos en la universidad y aunque no es mi tutora siempre fue un sueño mío y de Adonis que lo fuera.

A Jorge “El primavera” por su ideas tan geniales, por apoyarnos y aportar a las ideas que le proponíamos.

A Minerva y a Tomás por el apoyo que siempre me dieron y por esa amistad que cada día es más fuerte y que los quiero mucho.

Jesús

A mi compañero de tesis, que bastante pasamos luchando "*Alone in the Dark*" contra ruidos y mareas.

A mi madre, que siempre me apoyó y que tanto esfuerzo ha hecho para que yo terminara la carrera, que ha entendido mis acciones cuando inclusive ni yo mismo las entiendo. Hoy sin duda alguna, me gradúo gracias a ti.

A mi hermana, la "tía" más linda del mundo, que muchas veces lidió con las cosas que yo no, Negri, esto es para ti también.

A mi padre, que siempre ha confiado en mi talento y que nos ayudó en esta tarea, parte de lo persistente que hoy soy, se lo debo a él.

A Isel, y a mi Cristinita, que hacen palpitar mi corazón con cada beso o cada sonrisa.

A mis tíos y tías, todos jugaron un papel en esta obra, esto también es de ustedes.

A Jorge, el "*debugger*" de cada idea que teníamos, sin tu participación esta no hubiera sido nuestra tesis.

Al tribunal por su actitud crítica y por ser casi tutores de este trabajo.

A Yorgelys, que desde primer año nos ha ayudado cuando más lo hemos necesitado.

A mis hermanos en la carrera, Adonis, Osvel y Ronald, que ya son sin duda alguna, parte de mi familia; tantos momentos difíciles hemos compartido que a veces olvidamos decirnos gracias, pero aquí estamos, trazándonos metas más allá de lo imposible, GRACIAS.

A todos los que colaboraron con ruidos al sistema, críticas o consejos, gracias, a todos mis profes de programación, en especial a Tomás y Armando, por incentivar en mí el arte de programar, a Harold, por enseñarme el maravilloso mundo de *Actionscript*, creo que si alguien me motivó a alcanzar cada uno de los resultados que me hacen ingeniero, fueron ustedes.

Agradecemos al equipo del proyecto Calidad que dedicó su esfuerzo a la validación de esta tesis, especialmente al profesor Yudislandry, Celia y los futuros ingenieros Yamila y Gerardo.

RESUMEN

La Universidad de las Ciencias Informáticas (UCI) desde la puesta en práctica de un nuevo modelo de formación-investigación-producción centrado en el aprendizaje, logra un aprovechamiento de las ganancias que representan las aplicaciones informáticas, contribuyendo al desarrollo económico del país. Debido a esto y a la oportunidad de mercado que representan los Escenarios Virtuales Tridimensionales Interactivos en la Web (EVTIW), surge para la UCI la necesidad de incorporarse al desarrollo de aplicaciones similares. Sin embargo, no se cuenta con una documentación ni ejemplos prácticos que posibiliten este trabajo.

La presente investigación consiste en una guía de aprendizaje para el desarrollo de escenarios virtuales haciendo uso del *framework ojEssential*. La guía obtenida como resultado se encuentra basada en la implementación del caso de estudio Paseo Virtual del Docente 5 (PVD5). Para dicho proceso se analizaron las tecnologías empleadas en el desarrollo de escenarios virtuales en la Web, posibilitando una adecuada selección de las herramientas y metodologías utilizadas en el diseño e implementación del caso de estudio. La guía abarca los conceptos básicos del mundo tridimensional, cómo usar *ojEssential* en un desarrollo de este tipo y su empleo en la creación del ejemplo PVD5.

INDICE

INTRODUCCIÓN

El siglo XXI se enfrenta a la creciente implantación de un nuevo modelo de vida basado en la comunicación hombre-máquina, debido a la presencia constante de los medios digitales y al uso masivo de las Tecnologías de la Informática y las Comunicaciones (TIC) en todos los ámbitos. Esto implica la transformación global de los servicios y de los medios de comunicación, propiciando el surgimiento de nuevos modelos de negocio como el gobierno electrónico (*e-government*) y el comercio electrónico (*e-commerce*).

El desarrollo de estos modelos mediante el uso de las TIC, la necesidad de elevar los niveles de usabilidad y funcionalidad de las aplicaciones para estos fines, las mejoras en el *hardware* y los cambios de arquitectura informática, han propiciado el desarrollo de sistemas más avanzados y complejos como las *Rich Internet Applications* (RIA) (1). Este término surgió en el año 2001 para agrupar aplicaciones soportadas por navegadores web y poseedoras de las ventajas que presentan las aplicaciones tradicionales de escritorio.

El uso de aplicaciones web que permitan una interacción rápida y amigable con los clientes, es un tema de interés constante de las empresas en busca de elevar los mercados, las ventas y el número de clientes. El aumento de la demanda de aplicaciones de este tipo y de las expectativas de los clientes, han obligado a los desarrolladores a buscar nuevas soluciones. Una de las alternativas es la inclusión de Escenarios Virtuales Tridimensionales Interactivos en la Web (EVTIW), también conocidos como RIA que incluyen escenarios tridimensionales interactivos, parte fundamental de la definición de Web 3.0 (2), (3). Aplicaciones donde se lleva a cabo una simulación de un escenario con un determinado grado de interactividad desde la Web, siendo estos más demandados cada día en los sitios de Internet dirigidos al entretenimiento, entrenamiento, simulación, educación, defensa, medicina, arquitectura y a la comercialización.

Desde hace más de una década se han desarrollado EVTIW orientados al aprendizaje, como los simuladores de vuelo, los orientados a la enseñanza conocidos como *Massively Multilearner Online Learning Environment* (MMOLE) o los dirigidos al ámbito de la medicina (4). En la actualidad las empresas creadoras de productos dedicados al entretenimiento ven en esta tecnología una nueva posibilidad para el

desarrollo de videojuegos, siendo muchos de estos conocidos como videojuegos masivos en línea o *Massively Multiplayer Online Games* (MMO) (5). Algunos ejemplos son presentados a continuación.

***CryoPolis*:** creado por la compañía *Cryo* y lanzado públicamente en 1999, se trata de una comunidad para demostrar el potencial del lenguaje de programación libre SCOL, combinando de manera efectiva los principales elementos de comunicación 3D, 2D, redes y multimedia. En el año 2002, este escenario alcanzó su máximo potencial, brindando al usuario varias posibilidades, desde el simple *chat*, la proyección de *trailers*, hasta jugar tenis (6).

Second Life* o Segunda Vida:** *software* libre multiplataforma desarrollado por la compañía *Linden Lab,** accesible gratuitamente en Internet. Desde su inicio el 23 de junio del 2003, los usuarios conocidos como residentes, pueden acceder a Segunda Vida (SL por sus siglas en inglés), mediante el uso de programas de interfaces llamados *Viewers* (visores) permitiéndoles interactuar entre ellos mediante un avatar. Los residentes pueden explorar el mundo virtual, establecer relaciones sociales, participar en actividades individuales o grupales, además de crear y comerciar propiedad virtual entre ellos. La programación de este mundo virtual es abierta y libre. El código de SL permite a los usuarios modificar aspectos de su apariencia (7).

Con el transcurso de los años han evolucionado las tecnologías empleadas para el desarrollo de EVTIW, propiciándoles un mayor nivel de funcionalidad. Prueba de esto son los ejemplos abordados anteriormente, los cuales demuestran el nivel de realismo que se logra alcanzar en este tipo de aplicaciones, llegando a utilizarse en todos los aspectos de la vida práctica, desde la preparación de profesionales, hasta el entretenimiento de los usuarios motivados por la semejanza con la realidad, alcanzada en estos escenarios.

Cuba, como resultado del correcto aprovechamiento de las ventajas del alto nivel de preparación del capital humano disponible, y en el camino por alcanzar la invulnerabilidad económica, ha encontrado en la Informática una poderosa base para el desarrollo del país y una importante fuente de ingresos. Por ello, durante los últimos años se han ejecutado variadas estrategias, con el fin de fortalecer la industria del *software*. Objetivo que ha favorecido la reorganización, automatización y diversificación de sus mercados, con el propósito de elevar la informatización de la sociedad, los niveles de exportación de *software* y los servicios informáticos en general.

Una de las estrategias ejecutada es la creación de la Universidad de las Ciencias Informáticas (UCI). En la misma se implementa un nuevo modelo de formación-investigación-producción centrado en el aprendizaje, posibilitando la formación profesional de sus estudiantes desde el desempeño de los roles correspondientes a la industria del *software*. De esta forma se logra introducir un nuevo concepto de universidad-productiva, ya que en la misma los estudiantes y profesores se encuentran vinculados a centros de desarrollo y sus resultados contribuyen al avance económico, político y social del país.

La UCI, está estructurada en una red de veintitrés centros colaborativos más un centro de desarrollo de las Fuerzas Armadas Revolucionarias (FAR) que se encuentra insertado en la misma, dedicados al desarrollo de productos y servicios en diferentes áreas temáticas, además de llevar a cabo una integración de los procesos docentes, productivos e investigativos de alto nivel. El Centro de Tecnologías para la Formación (FORTES) de la Facultad 4 constituye uno de estos, el cual desarrolla proyectos como Alfaomega y las “Colecciones Multisaber y el Navegante” que intentan adentrarse en un desarrollo similar pero desde la Web, a través de la implementación de plataformas para la gestión del aprendizaje y la creación de multimedias educativas respectivamente. En encuestas realizadas a los líderes de estos proyectos se pudo verificar que existe poco dominio de la tecnología y los implicados en el desarrollo no poseen conocimiento de este campo, lo que dificulta el cumplimiento de las metas trazadas en cada caso.

Para el desarrollo de *software* en la UCI se llevan a cabo diversas estrategias con el objetivo de lograr la capacitación del personal, entre las cuales se encuentran los cursos de capacitación y la recopilación de documentación en dependencia del tipo de desarrollo a seguir. En el caso de la implementación de EVTIW no se cuenta con la información necesaria, de manera centralizada, para que desde los puntos de vista teóricos y prácticos posibilite una solución al proceso deseado.

A lo largo de los años en el mundo se han creado mecanismos que facilitan este propósito, desarrollándose recursos para el aprendizaje donde los usuarios cuentan con materiales y medios sobre una temática determinada (8). En el amplio contexto de los recursos para la enseñanza, **las guías de aprendizaje** constituyen uno de los principales elementos pues permiten incrementar la capacidad de consultar, identificar e interpretar la información necesaria y pertinente para conducir al usuario a investigar, descubrir y construir soluciones de acuerdo con el contenido encontrado en dicha guía (9).

Contar con una guía de aprendizaje es importante para los desarrolladores de EVTIW en la UCI, ya que contribuye de manera significativa a disminuir el tiempo de desarrollo y facilita la capacitación del personal dedicado a estas tareas. Este nuevo elemento ayuda a potenciar la apertura de mercados de aplicaciones informáticas para la UCI, contribuyendo la misma al mejoramiento de la industria nacional de *software* y al fortalecimiento de la economía cubana, aprovechando la gran demanda que poseen estas aplicaciones.

Debido a la necesidad de dar solución a la situación planteada, se identifica como **problema de investigación**: inexistencia de una guía de aprendizaje para el proceso de creación de EVTIW basada en un caso de estudio.

Para resolver el problema de investigación se identifica como **objeto de estudio** las Aplicaciones Enriquecidas de Internet (RIA por sus siglas en inglés), precisando como **campo de acción** los EVTIW.

En el marco del trabajo, sus autores se trazaron como **objetivo general**: desarrollar una guía de aprendizaje para el proceso de creación de EVTIW, desglosándose el presente objetivo general en los siguientes **objetivos específicos**:

- Estudiar las tecnologías empleadas para el desarrollo de EVTIW.
- Implementar un EVTIW como caso de estudio.
- Elaborar una guía de aprendizaje para el proceso de creación de EVTIW.

Para dar cumplimiento a los objetivos específicos trazados se formularon las siguientes **tareas de investigación**:

- Realización de un estudio sobre el estado del arte en el desarrollo de RIA.
- Investigación de los aspectos a tener en cuenta para elaborar una guía de aprendizaje.
- Definición de las funcionalidades a incorporar en el escenario.
- Definición de la arquitectura base de la aplicación.
- Modelación del escenario tridimensional.

- Creación de mapas texturas.
- Elaboración de mapas de iluminación.
- Desarrollo de mapas de colisiones.
- Implementación de las funcionalidades definidas.
- Diseño de la interfaz de la aplicación.
- Empaquetamiento del escenario virtual para su despliegue en la Web.
- Validación de la aplicación.
- Selección de las técnicas de aprendizaje a utilizar en la guía.
- Definición de la estructura de la guía.
- Elaboración de la guía de aprendizaje.
- Validación de la guía de aprendizaje.

Estructura capitular:

Capítulo 1: Fundamentación teórica, se realiza un estudio sobre el estado actual en que se encuentran las metodologías de desarrollo de *software*, tecnologías, lenguajes de programación, *frameworks*, librerías, y herramientas a utilizar. Además se analiza cómo surgieron las guías de aprendizaje y la importancia que tiene su uso para acelerar el proceso enseñanza-aprendizaje en una temática determinada, así como los principales elementos para su elaboración.

Capítulo 2: Descripción del caso de estudio PVD5, se definen los aspectos funcionales y no funcionales de la aplicación, las historias de usuarios a implementar, la planificación y la arquitectura del sistema.

Capítulo 3: Implementación de un EVTIW, se describe cómo se llevó a cabo el proceso de desarrollo del caso de estudio PVD5 siguiendo cada aspecto definido en el capítulo 2.

Capítulo 4: Guía de aprendizaje para el desarrollo de escenarios virtuales tridimensionales interactivos en la Web, se brinda una descripción de la guía obtenida como resultado de la investigación, así como de los elementos utilizados en la confección de la misma.

Capítulo 1: Fundamentación teórica.

1.1. Aplicaciones enriquecidas de Internet (RIA)

Las RIA surgen como una combinación de las ventajas que brindan las aplicaciones web y las aplicaciones de escritorio. Utilizan un navegador web para ejecutarse y por medio de complementos o una máquina virtual, logran brindar a los usuarios nuevas funcionalidades (1). Son una nueva generación de aplicaciones cuyo objetivo principal es mejorar la experiencia de navegación de los usuarios. Estas combinan las grandes ventajas que brindan los estándares, tecnologías y prácticas para llevar a cabo su principal objetivo: mejorar la calidad y proporcionar una mayor interactividad, velocidad de respuesta y flexibilidad al usuario.

Los desarrolladores se han esforzado en mejorar la arquitectura de las mismas para lograr una mayor eficiencia y reducir la brecha entre las aplicaciones web y las de escritorio, existiendo varias herramientas para la creación de entornos RIA. Entre las más conocidas se pueden mencionar *Adobe Flash*, *Adobe Flex* y *Microsoft Silverlight* (10).

El desarrollo de RIA permite brindarle al usuario aplicaciones más intuitivas, efectivas, interactivas, dinámicas, rápidas y multiplataforma, además facilitan las descargas progresivas para recuperar contenidos y datos, inclusive adoptan ampliamente los estándares de Internet (1).

Para el desarrollo de la presente investigación sus autores consideraron llevar a cabo la construcción de una RIA, ya que las mismas proveen mecanismos para procesar las peticiones realizadas desde las estaciones clientes, garantizando de esta forma un equilibrio en la carga de trabajo entre los servidores y clientes. Un caso específico de las RIA, son los EVTIW que posibilitan la representación de un mundo virtual en tres dimensiones. En este trabajo se lleva a cabo el desarrollo de un EVTIW logrando realizar una simulación de un entorno determinado alcanzando una gran semejanza con la realidad.

1.2. Metodologías de desarrollo de software

Para asegurar el éxito durante el desarrollo de *software* no es suficiente contar con nociones de modelado y herramientas, hace falta un elemento importante: la **metodología de desarrollo.**

Entre las metodologías de desarrollo de *software* más usadas en el mundo se encuentran *Rational Unified Process* (RUP), SCRUM y *Extreme Programming* (XP).

RUP se caracteriza por ser una de las más tradicionales, **dirigida por casos de uso** donde estos definen lo que el usuario desea a partir de la captura de requisitos y la modelación del negocio. **Centrada en la arquitectura**, característica que brinda una visión completa del sistema. Además de ser **iterativa e incremental** donde el desarrollo se lleva a cabo por fases a través de iteraciones (11). Aunque la metodología RUP puede adaptarse a cualquier proyecto, no es recomendable aplicarla en proyectos pequeños cuyo objetivo fundamental sea obtener el producto que el cliente necesita en el menor tiempo posible. Su uso en este tipo de proyectos trae consigo la generación de artefactos que muchas veces no son relevantes. (Ver anexo 1)

SCRUM es una metodología de desarrollo que hace énfasis en la gestión de proyecto, permitiendo un control de los procesos que se realizan. No define prácticas de ingeniería de *software* en detalle, sin embargo en muchas ocasiones se combina con otras metodologías de desarrollo para suplir sus carencias. Aunque es una metodología ágil, no se beneficia de algunas de las técnicas de desarrollo que este grupo define, por ejemplo no se especifica la programación en parejas ni las pruebas unitarias (12). (Ver anexo 1)

eXtreme Programming (**XP**) se encuentra dentro de las metodologías ágiles, planteando la **constante retroalimentación con el cliente** teniéndolo como parte del equipo y propiciando la interacción del mismo con el producto en desarrollo. Presenta **cortas iteraciones**, donde en cada una se obtiene un producto listo para entregar y que tiene valor para el cliente. Es **muy flexible a cambios**, previendo futuras modificaciones y evitando momentos que paralizan el desarrollo del producto (13). Además utiliza prácticas proporcionadas por las metodologías ágiles, tales como la **refactorización** proporcionando un cambio gradual del código fuente hacia un diseño más adaptable. Facilita además la **programación en parejas** permitiendo a dos programadores compartir conocimientos técnicos y posibilita una i**ntegración continua**, asegurando que exista siempre un sistema disponible en el que se ejecuten todas las pruebas. (Ver anexo 2)

Debido a las características de esta investigación, en cuanto al reducido equipo de desarrollo, el corto tiempo para la implementación de la solución y los cambios que pueden producirse a través del proceso

de optimización, se considera que la metodología XP es la más factible para guiar este proceso de desarrollo de software. Esta consideración también se debe a que RUP puede desviar el curso de la investigación hacia la generación de artefactos que son irrelevantes, y SCRUM no define algunas prácticas de ingeniería definidas por las metodologías agiles, como la programación en parejas, entre otras que aceleran el proceso de implementación.

1.3. Bases tecnológicas

En los últimos años, la socialización de la informática y las mejoras en las comunicaciones han dado lugar al desarrollo de aplicaciones y herramientas que permiten evolucionar y aprovechar al máximo las actividades que realiza el hombre cada día, facilitando así, por ejemplo, un claro aumento de la productividad y una sustancial mejora en las aplicaciones desarrolladas, brindando mayor satisfacción a los usuarios (14). Actualmente para la creación de cualquier solución informática se hace uso de distintas tecnologías en dependencia de las necesidades existentes. Esto persigue como objetivo llevar a cabo una implementación de *software* que responda específicamente a un tipo de aplicación a través del Entorno Integrado de Desarrollo (IDE por sus siglas en inglés).

1.3.1. Entorno de desarrollo integrado

El incremento de las soluciones informáticas a través de los años, propicia en los desarrolladores la necesidad de contar con aplicaciones que permitan una rápida implementación, dependiendo del tipo de *software* que se quiera lograr. Con el objetivo de satisfacer esta necesidad surgen los IDE, programas que mediante diferentes herramientas de programación como son un compilador y un depurador, entre otras, brindan un ambiente amigable para el desarrollo de cualquier aplicación informática. Para la construcción de la solución se investigaron los IDE *Adobe Flex Builder* 3, *Microsoft Silverlight* 4 y *Adobe Flash Builder* 4. (Ver anexo 5)

***Adobe Flex Builder* 3** es una aplicación comercializada por *Adobe*, la cual permite la creación de RIA utilizando un lenguaje de etiquetas conocido como MXML para la definición de la interfaz de usuario, y *Actionscript* 3.0 para el comportamiento de dichas aplicaciones. Permite una visualización interactiva de los datos, además emplea la refactorización de código y las ventajas de *Flash Player* 9 (15).

***Microsoft Silverlight* 4**, es un *software* comercializado por la compañía *Microsoft*, utiliza los lenguajes de programación *XAML*, C# y *Visual Basic*. *Silverlight* hace uso de potentes herramientas como un *framework* para la administración de extensibilidad, otras para la visualización de datos y para la personalización de componentes, permite además la interacción con otras tecnologías y aplicaciones como *Microsoft Office Excel* con el objetivo de crear reportes (16).

***Adobe Flash Builder* 4**, es una herramienta privativa y compatible con las plataformas de *Windows* y *Mac OS*. Entre sus principales características se encuentra el uso de su herramienta Monitor de Red, para inspeccionar todo el flujo de datos entre las diferentes partes de las aplicaciones desarrolladas. Posibilita además la utilización de elementos como el nuevo explorador de datos y servicios, una interfaz para la programación de aplicaciones enfocadas a interactuar con Redes Sociales y la máquina virtual *Flash Player* 10 (17).

Como resultado de una investigación acerca de los IDE, los autores de este trabajo seleccionaron el IDE *Adobe Flash Builder* 4 para el desarrollo del EVTIW, por las numerosas ventajas que posee como su compatibilidad íntegra con *Adobe Flash Player* 10 y el desarrollo centrado en los datos. Además provee un ambiente integrado muy práctico, donde convergen el *framework*, la librería y el lenguaje de programación, elementos seleccionados en esta investigación (Ver anexo 6). Además posee un compilador y un depurador, que constituyen un factor importante para la realización de las pruebas unitarias.

1.3.2. Bibliotecas de código

Las bibliotecas de código o librerías, como también suelen llamarse, brindan una colección de subprogramas para el desarrollo de *software*. Contienen código y datos auxiliares, que brindan servicios a programas independientes, lo que permite la distribución y modificación del código además de los datos de forma modular (18).

En la actualidad existen varias bibliotecas creadas por los desarrolladores de *software* para facilitar la implementación de EVTIW, entre las más usadas se encuentran *Sandy3D*, *Papervision3D* y *Away3D*, las cuales permiten una representación tridimensional bastante semejante a la realidad. Además se pueden crear efectos avanzados de sombreado, sistema de materiales para cambiar fácilmente la apariencia de

los objetos. Estas son compatibles con el *Flash Player* en sus versiones 9 o 10, y permiten manipular modelos tridimensionales en varios formatos 3D (DAE, 3DS, ASE, MD2) (19). Estas librerías presentan notables diferencias principalmente en cuanto a la simplicidad del código, el rendimiento, y la generación de cuadros por segundo (Ver anexo 3)

Sandy3D no incluye ningún mecanismo para obtener información interna sobre el motor 3D, es decir, la medida de cuadros por segundo y número total de polígonos en la escena (20). Estos elementos son de gran utilidad durante el desarrollo de EVTIW para las pruebas de rendimiento. Por lo que resulta inconveniente el uso de esta librería en la construcción de aplicaciones donde se desee alto grado de rendimiento. *Sandy3D*, por estar desarrollado en el lenguaje de programación *Actionscript* 2.0, no permite aprovechar las grandes ventajas que brinda la versión 3.0 de este, dificultando la reutilización y simplicidad del código, así como la rapidez del desarrollo del *software*, elementos primordiales para la realización de este trabajo.

Papervision3D, desarrollada con el lenguaje de programación *Actionscript* 3.0 que permite obtener información interna sobre el motor de 3D. Sin embargo, la comunidad de desarrollo que presenta, ha disminuido en los últimos años. Una parte de los programadores que la impulsaron inicialmente se han movido hacia *Unity* (plataforma completa para desarrollo de juegos). Entre sus usuarios, también ha habido una cierta transición hacia *Away3D* en los últimos tiempos (21).

Away3D provee una de las formas más consistentes para la representación 3D. Su licencia de código abierto permite a cualquier persona aportar correcciones de errores o mejora de sus funciones. Tiene una comunidad activa para la colaboración, brindando técnicas libres y además cuenta con un sitio oficial donde regularmente se publican tutoriales, demos y otros servicios que facilitan el aprendizaje de esta potente librería. Define las clases como secciones de código con formas y funciones específicas, agrupadas en paquetes que responden a los elementos de un escenario (22). Esto facilita el trabajo de los desarrolladores, pues se cuenta con una estructura sólida, donde cada paquete de clases brinda información sobre las funciones que este permite.

Debido a la necesidad del rápido y eficiente desarrollo de un EVTIW que posea un código fuente legible, se considera que las librerías *Sandy3d* y *Papervision3d* no cumplen con los requisitos necesarios para el desarrollo de la presente investigación. Conociéndose además, que se cuenta con un equipo de desarrollo

pequeño y un tiempo reducido. En tales circunstancias, *Away3D* resulta la librería de código abierto por excelencia, que soluciona los aspectos negativos analizados en las bibliotecas de código anteriores. *Away3D* es la librería seleccionada por los autores de la investigación.

1.3.3. Framework seleccionado para el desarrollo

En general, un *framework* es una estructura conceptual y tecnológica de soporte, compuesta de componentes personalizables e intercambiables para el desarrollo de aplicaciones. Se puede considerar como una aplicación genérica incompleta y configurable a la que se le pueden añadir elementos para construir una aplicación concreta. Puede incluir soporte de programas, librerías y un lenguaje interpretado entre otros elementos para desarrollar y unir los diferentes componentes de un proyecto (23). Además de forma general persiguen como objetivos principales, acelerar el proceso de desarrollo, reutilizar el código ya existente y promover buenas prácticas como el uso de patrones de diseño.

***Flex* 4.0** es el *framework* analizado pues facilita el flujo de trabajo entre diseñadores y desarrolladores de aplicaciones RIA, y además presenta como principales objetivos:

Diseño en mente: la arquitectura de personalización se simplificó.

Productividad del desarrollador: se mejoró el desempeño del compilador y se perfeccionó el enlazamiento de los datos relacionados a los componentes.

Evolución del *framework*: se añadieron nuevos componentes y se modificó el *Kit* de Desarrollo de *Software* (SDK por sus siglas en inglés) para aprovechar las características del *Flash Player* 10, además es puesta en práctica la nueva arquitectura de componentes *Spark* que termina con los mayores problemas que cada componente exhibe, haciendo las tareas más sencillas ya que simplifica el flujo de trabajo entre desarrolladores y diseñadores separando la parte lógica de estos elementos (24).

Los cambios en *Flex* 4.0 representan un gran paso de avance para el desarrollo de aplicaciones en la plataforma *Flash* ya que aportan funcionalidades tales como servicios web, objetos remotos, gráficas, efectos de animación y otras interacciones simples (25).

Flex 4 es un *framework* desarrollado y compuesto, en su totalidad por archivos de *Actionscript* 3.0 y componentes de *Flash* (ficheros SWC del inglés *Shockwave Component*), los cuales abarcan varias funcionalidades. La selección de su uso se debe fundamentalmente a que se encuentra incluido en el IDE *Flash Builder* 4 y provee las funcionalidades necesarias para la integración de *Flash Player* 10 con la librería *Away3d* 3.5.

1.3.4. Lenguaje de programación

Los lenguajes de programación son usados para escribir instrucciones que pueden ser traducidas a lenguaje de máquina y luego ejecutados por una computadora (26). Entre los más destacados se encuentran *Javascript*, *PHP*, *Delphi*, *Visual Basic*, *C++*, *C#*, *Pascal*, *Java* y *Actionscript*, permitiendo la construcción de un software específico (27). Estos lenguajes surgen con el objetivo de facilitar el trabajo del hombre en la creación de *software* ante un problema determinado como la automatización de procesos o la simulación de entornos virtuales.

El lenguaje *Actionscript* se ha convertido en la herramienta por excelencia para los desarrolladores de aplicaciones *Flash*. Desde su versión 2.0 hasta la versión 3.0 han revolucionado la forma de representación visual de EVTIW, haciendo posible la interacción de los usuarios en un mundo totalmente virtual. Aunque *Actionscript* 2.0 es fácil de usar no resulta conveniente su utilización en la presente investigación, ya que posee deficiencias en su rendimiento para procesar el código con *Flash Player*, y no es totalmente compatible con el estándar internacional para el lenguaje de creación de *scripts* ECMAscript (ECMA-262), aprobada por la ISO/IEC 16262 (28), (29).

***Actionscript* 3.0** es un lenguaje de programación orientado a objetos para el desarrollo de aplicaciones que incorporan elementos multimedia. Se encuentra soportado por tecnologías como *Adobe Flash* y *Adobe Flex* (30). Permite la ejecución del código con una rapidez diez veces mayor que las versiones anteriores del lenguaje. Se encuentra enfocado al desarrollo de aplicaciones complejas con grandes volúmenes de datos y una base de código orientada a objetos y reutilizables. Además incorpora soporte para elementos desarrollados con el lenguaje PHP y presenta un conjunto de construcciones del lenguaje para la manipulación de ficheros XML. Permite ejecutar un mayor volumen de código respecto a otras versiones con la misma cantidad de recursos del CPU (31).

Es posible programar utilizando este lenguaje a través del IDE *Adobe Flash Builder* 4, herramienta clave para el desarrollo de este trabajo. Además resulta provechoso su uso teniendo en cuenta que la librería de código abierto *Away3D* está desarrollada con *Actionscript* 3.0, logrando una integración total con las tecnologías seleccionadas. (Ver anexo 6)

1.3.5. Herramientas de modelado 3D

En el constante trabajo sobre las tres dimensiones, surge la necesidad de minimizar el tiempo de realización de los modelos tridimensionales del objeto que se quiera tratar y aumentar la calidad de los mismos. Como solución a esto surgen las herramientas de modelado, aplicaciones que facilitan la confección de modelos 3D, haciendo uso de distintas técnicas de dibujo y de numerosas herramientas. En el curso de esta investigación se estudiaron tres de las más utilizadas en el mundo, *Blender*, *Autocad* y *Google Sketchup*.

***Blender* 2.5** es una aplicación libre, multiplataforma, que permite realizar un diseño tridimensional con un alto grado de detalle (32). Brinda la posibilidad de exportar los modelos 3D hacia los formatos 3ds, dae, fbx, obj, y x3d; incluye además un motor de juegos 3D y hace uso de la cinemática inversa como técnica de animación (33), (34). (Ver anexo 4)

***Autocad* 18.0**, por su parte es una aplicación desarrollada por la compañía *Autodesk*, dedicada al diseño arquitectónico de forma general utilizando para esto el trabajo en dos o tres dimensiones (35). Entre sus principales características se destacan el administrador de conjuntos de planos, el uso del dibujo paramétrico y la exportación hacia dwf, pdf, dxf, fbx, wmf, entre otros formatos (36).

***Google Sketchup* 8** es un *software* desarrollado por la empresa *Google*, el cual permite una rápida conceptualización de entornos arquitectónicos, gracias a las herramientas incorporadas como son unir, intersecar, sustraer, recortar, y dividir sólidos. Hace uso de la herramienta empujar/tirar para el modelado tridimensional a partir de formas en dos dimensiones (37).

Para la realización del modelo tridimensional se escogió la herramienta *Blender* 2.5, ya que posibilita el diseño de modelos 3D utilizando la técnica de mapeado de texturas así como la correcta exportación de las coordenadas de superficie (UV) del modelo asociadas a la textura. Además cuenta con las

herramientas necesarias para el empleo de la técnica *bake* posibilitando la incorporación de sombras a los modelos, con soporte para numerosos formatos de exportación e importación logrando de esta forma una compatibilidad extrema con otras aplicaciones.

En el caso de *Google Sketchup* y *Autocad*, son aplicaciones privativas a diferencia de *Blender*, dedicadas al modelado de edificaciones, sin embargo *Google Sketchup* presenta problemas de exportación en las coordenadas UV asociadas a las texturas en el modelo, repercutiendo esto de manera negativa en la selección de las mismas para el modelado tridimensional.

1.3.6. Visual Paradigm for UML 6.4 Enterprise Edition

Visual Paradigm for UML 6.4 *Enterprise Edition* es una herramienta Computer Aided Software Engineering (CASE) que soporta los principales estándares de la industria tales como Lenguaje de Modelado Unificado (UML por sus siglas en inglés) y *Systems Modeling Language* (SysML), entre otros. Brinda un conjunto de herramientas a los equipos de desarrollo de *software* necesario para la captura de requisitos, la planificación de controles, el modelado de clases, modelado de paquetes y modelado de datos (38). Adicionalmente, *Visual Paradigm* permite generar código *Actionscript* a partir de los diagramas de clases, ofreciendo una gran flexibilidad al proceso de desarrollo del producto y la arquitectura de la aplicación. Su uso es de vital importancia, ya que permite ganar en tiempo y eficiencia.

1.3.7. Servidor de aplicaciones

Con el desarrollo de Internet y los servicios web, se han creado varias posibilidades para el acceso a la información prácticamente desde cualquier sitio. Esto representa un desafío para los desarrolladores de *software*, ya que los avances en las TIC demandan cada vez aplicaciones más rápidas y robustas. Los servidores de aplicaciones se crearon con el fin de gestionar las peticiones del usuario y enviarles información a través de un protocolo de comunicación (39).

1.3.7.1. Servidor web Apache

Apache es un servidor *Hypertext Transfer Protocol* (HTTP) de código abierto que por su facilidad de configuración, robustez y estabilidad se encuentra desplegado en millones de servidores. Es multiplataforma, haciéndolo prácticamente universal, tecnología gratuita de código abierto, patentado a

través de la licencia *Apache*, además es extensible. Constituye el servidor HTTP más popular y usado, lo que permite que sea fácil de adquirir y brinda una amplia fuente de ayuda (40). Muestra, entre otras características, mensajes de errores altamente configurables, posee bases de datos de autenticación y negociado de contenido, facilita la utilización de *Perl*, PHP, *Java* y otros lenguajes *script* (41). Además es muy fácil de configurar para la creación y gestión de *logs*, permitiendo un mayor control sobre lo que sucede en el servidor (42).

Como resultado de la investigación se tendrá un EVTIW desplegado en un servidor de aplicaciones. Debido a las grandes ventajas que presenta *Apache*, principalmente su robustez y estabilidad, es seleccionado en este trabajo, pues se prevé el acceso a este escenario por usuarios conectados simultáneamente. Su uso provee la infraestructura necesaria para las aplicaciones web posibilitando que los programadores se dediquen en exclusiva a implementar la lógica del dominio, ya que algunos servicios de uso común, como son las transacciones, la seguridad y la persistencia, son proporcionados por *Apache*.

1.4. Guías de aprendizaje

Desde tiempos remotos el hombre siempre se ha esforzado para adaptarse al medio ambiente que lo rodea por muy dinámico que fueran los cambios, por lo cual tuvo que ampliar siempre su conocimiento hacia nuevas ramas del saber. En el constante quehacer que esto implica, surge la necesidad de facilitar y agilizar el proceso de enseñanza-aprendizaje de cualquier tema que se esté tratando. Hoy en día la sociedad cuenta con una herramienta que brinda una solución a esta problemática: las guías de aprendizaje.

Estas tienen como objetivo dirigir a los estudiantes hacia una enseñanza efectiva, ya que brindan una explicación del tema a tratar en un lenguaje lo más entendible y claro posible. Además mientras mejor esté diseñada en cuanto a los temas que aborda, se logra un mejor entendimiento y avance sobre los mismos. De esta forma, ayudan a identificar de forma rápida y precisa el contenido de estudio, a través del uso de técnicas de aprendizaje, y evacuando las principales dudas que puedan surgir en el camino del nuevo saber (43).

1.4.1. Técnicas de aprendizaje

Las técnicas de aprendizaje son un grupo de estrategias y procedimientos de carácter cognitivo integradas directamente al proceso de estudio; tales como la planificación de las actividades, el subrayado, el resumen, la elaboración de esquemas, así como otras estrategias que tienen un carácter más complementario, como pueden ser la toma de apuntes o la realización de trabajos escolares. Son distintas perspectivas aplicadas al aprendizaje, las cuales deben ser críticas para alcanzar el éxito en el conocimiento, considerándoseles esenciales a lo largo de la vida (44).

Hay una variedad de técnicas de estudio, que pueden enfocarse en el proceso de organizar y procesar nueva información, retener información, o superar exámenes. Estas técnicas ayudan a la retención de listas de información y toma de notas efectivas.

El uso de estas técnicas es de vital importancia para el entendimiento rápido y preciso de la guía de aprendizaje que se obtendrá como resultado de la presente investigación, abriendo al estudiante un camino para transitar por las sendas del conocimiento de la programación tridimensional. Además ofrecerá teorías y ejemplos prácticos que facilitan el desarrollo de EVTIW de forma ágil y con el mayor grado de detalle posible.

1.4.2. Guías de aprendizaje en la Informática

Desde el descubrimiento por el hombre de una ciencia como la Informática, ha divisado la posibilidad de informatizar las actividades que realiza en la vida diaria. Para lograr esto y acelerar el proceso de desarrollo de las aplicaciones informáticas se ha requerido siempre de un personal altamente calificado, debido a que mientras mejor preparado se encuentre el recurso humano para el desarrollo, son mayores las posibilidades de obtener un *software* en un mínimo tiempo de realización y con una excelente calidad. En los esfuerzos por lograr una mayor y mejor superación profesional, el personal informático existente hoy en día hace uso de diferentes recursos como son los libros digitales, las guías de aprendizajes, los tutoriales, los *blogs* y los video tutoriales, medios enfocados a brindar un conocimiento de la mejor forma posible según sus propias limitaciones acerca de la temática que se quiera abordar.

Todos estos medios, para proporcionarle al usuario un conocimiento lo más claro y entendible posible, hacen uso de diversos recursos como son las imágenes, las capturas de pantalla, el análisis por fragmentos del código fuente y los videos. Con la constante variabilidad que sufren las tecnologías de la información, se requiere de medios de enseñanza más interactivos y prácticos para obtener los conocimientos, con el objetivo de facilitar aún más el proceso de aprendizaje de sus usuarios. Ejemplo de esto son las guías de aprendizaje, enfocadas por lo general en los aspectos prácticos de un contenido determinado y ofrecen elementos básicos de la teoría correspondiente. Además utilizan numerosos recursos como ejemplos anexos, archivos del código fuente y animaciones para visualizar de manera gráfica cada parte del conocimiento que se desea transmitir. La utilización de los diferentes recursos que se logra en las guías de aprendizaje ha demostrado en la última década la posibilidad de un aprendizaje efectivo y en un mínimo de tiempo, garantizando así la motivación de los usuarios en comparación a otros medios de enseñanza.

Capítulo 2. Descripción del caso de estudio PVD5.

2.1. Descripción del sistema

La descripción inicial de cualquier solución informática es una idea que generalmente se analiza antes de una reunión formal con un equipo de desarrollo de *software*. Es una visión que permite a los clientes imaginar y percibir la utilidad del uso del producto que desea, pues son ellos mismos quienes realizan la propuesta a partir de sus necesidades. De ahí que sea muy común escuchar, en los primeros encuentros con los clientes, peticiones informales de requisitos e ideas de la interfaz de usuario, las cuales aportan mucho sobre la concepción que tienen del *software* que desean.

En la investigación se aprovechan las ventajas que brinda la metodología de desarrollo XP la cual propone para las primeras reuniones con los clientes, realizar un levantamiento de los aspectos relacionados a sus necesidades (47). En estos encuentros se pueden incluir prácticas muy efectivas como:

- **Entrevistas** frecuentes para aclarar dudas que puedan ir apareciendo de manera continua.
- **Tormenta de ideas** entre el equipo de desarrollo y el cliente de forma que vayan surgiendo nuevas vías de implementar el producto.
- **Observación** de los procesos más importantes a informatizar.
- **Juegos de rol** donde el cliente simule ser un usuario que interactúa con el sistema.
- **Prototipos** que faciliten la representación visual de las necesidades más relevantes del cliente (48).

En el presente trabajo el cliente forma parte del equipo de desarrollo. Esto es un aspecto positivo que permite la descripción de sus necesidades, facilitando la definición del alcance y el tiempo de entrega del producto, dos de los aspectos principales en el desarrollo de un proyecto guiado por la metodología XP.

La definición del alcance del producto permite delimitar el trabajo a realizar para cumplir con los objetivos y desarrollar los entregables del proyecto (49). A continuación se especifica el alcance del producto a desarrollar:

Esta solución consiste en un EVTIW, el cual complementa la guía de aprendizaje para el desarrollo de aplicaciones similares. De esta forma el escenario debe permitir:

- Desplazar la cámara mediante el teclado hacia adelante, atrás, izquierda o derecha a través de todo el escenario.
- Girar la cámara mediante el teclado hacia la izquierda o la derecha.
- Encender y apagar las luces de determinadas localizaciones en el escenario haciendo *clic* en los encendedores a una determinada distancia.
- Colisionar con las paredes y objetos del escenario.
- Girar la cámara mediante el *mouse* hacia todas las direcciones.

Para el control de la cámara se deberán tener en cuenta algunas funciones del teclado y del *mouse*, encontrándose disponible para ello las teclas A, S, D, W, *Ctrl*, *Shift*, ←, ↑, →, ↓. Al iniciarse la aplicación se podrá girar hacia la derecha o a la izquierda mediante las teclas A, D, ←, → y si se acciona la tecla *Ctrl*, se alternarán las funciones asociadas a estas teclas, entre desplazar la cámara de forma lateral o girar la cámara. Además accionando la tecla *Shift* permitirá desplazarse con una mayor rapidez en el escenario.

En el caso de la utilización del *mouse*, la aplicación permitirá girar la cámara hacia todas las direcciones, procediendo para esto de la siguiente manera: se deberá hacer *clic* y arrastrar el *mouse* hacia la dirección en la que se desee girar.

Respecto a la interacción con las luces, se debe acercar la cámara al encendedor de la luz y posteriormente darle *clic* para cambiar el estado de las luces.

De la misma manera la guía de aprendizaje obtenida como resultado, comprende los principales pasos a desarrollar para construir un EVTIW, utilizando para ello imágenes, fragmentos de código, capturas de pantalla, notas, vínculos de descarga y advertencias.

2.2. Personal vinculado al sistema

Como resultado de la presente investigación se obtuvo el *framework ojEssential*, una guía de aprendizaje y el caso de estudio PVD5 para el desarrollo de EVTIW. Estos resultados están dirigidos a un personal con características particulares, las cuales se detallan a continuación.

Para la utilización del EVTIW por parte del personal es necesario un conocimiento básico de informática, puntualmente en el manejo del *mouse* y teclado.

En el caso del *framework* y la guía de aprendizaje pueden ser utilizados por cualquier usuario o desarrollador que posea conocimientos básicos del lenguaje de programación *Actionscript* 3.0.

2.3. Aspectos funcionales del producto

Los aspectos funcionales son capacidades o condiciones que el producto debe cumplir, motivo por el que constituyen un elemento de primer orden para el desarrollo exitoso de *software*.

AF1: Desplazar la cámara mediante el teclado.

AF1.1: Desplazar la cámara hacia la derecha.

AF1.2: Desplazar la cámara hacia la izquierda.

AF1.3: Desplazar la cámara hacia el frente.

AF1.4: Desplazar la cámara hacia atrás.

AF2: Girar la cámara con el teclado:

AF2.1: Girar la cámara hacia la derecha.

AF2.2: Girar la cámara hacia la izquierda.

AF3: Girar la cámara con el *mouse* hacia todas las direcciones.

AF4: Interactuar en el escenario.

AF4.1: Encender la luz haciendo *clic* en los encendedores a una determinada distancia.

AF4.2: Apagar la luz haciendo *clic* en los encendedores a una determinada distancia.

AF5: Colisionar con las paredes y objetos del escenario.

2.4. Aspectos no funcionales de la aplicación

Los aspectos no funcionales son propiedades o cualidades que el producto debe tener y que lo hacen más atractivo, usable, rápido y confiable. Estos elementos son importantes para que clientes y usuarios puedan valorar las características del producto.

Aspectos de *software*

- Máquina virtual *Flash Player* 10.

Diseño e implementación

- Diseñar el producto de forma tal que permita ser extensible.
- Permitir la visualización del EVTIW en los navegadores *Mozilla Firefox* 3.5, *Google Chrome*, *Opera* e *Internet Explorer*.

Apariencia o interfaz externa

- Brindar a los usuarios una interfaz que le proporcione información sobre el lugar en que se encuentra.

Portabilidad

- Tamaño mínimo del paquete a desplegar.

2.5. Exploración

La exploración es la fase en la que se define el alcance general del proyecto. El cliente establece lo que necesita mediante la redacción de sencillas Historias de Usuario (HU). Además los programadores estiman los tiempos de desarrollo en base a esta información, por lo que debe quedar claro que las estimaciones realizadas en esta fase son primarias (ya que estarán basadas en datos de muy alto nivel), y podrían variar cuando se analicen con más detalle en cada iteración. Esta etapa dura típicamente un par de semanas, y el resultado es una visión general del sistema, y un plazo total estimado. Además constituye el punto de partida que propone XP para la construcción de un producto (50).

Una vez que los clientes entregan su propuesta al equipo de desarrollo, comienza el análisis en grupo, las tormentas de ideas, la confección de diagramas representando diferentes situaciones y la conceptualización del *software*. Un aspecto importante en el esclarecimiento de las dudas que puedan surgir sobre los procesos a automatizar es que el cliente forma parte del equipo de desarrollo.

2.6. Historias de usuarios

Las Historias de usuario (HU) en XP tienen gran similitud con los casos de uso en RUP, con la diferencia de que deben ser descritas en no más de tres líneas e idealmente es el cliente quien las redacta y prioriza. Posteriormente se le suma un tiempo estimado de desarrollo que lo define el propio equipo de desarrollo. En esencia no son más que las ideas del cliente, organizadas y agrupadas de acuerdo a su funcionalidad. También se tiene en cuenta un orden tal que permita al mismo priorizar sus necesidades y al equipo de trabajo definir las que resultan críticas o claves en el desarrollo de la solución (51).

Las HU también conducen el proceso de creación de los *tests* de aceptación (empleados para verificar que las HU han sido implementadas correctamente) (Ver epígrafe 3.2.2). Existen diferencias entre estas y la tradicional especificación de requisitos, fundamentalmente en el nivel de detalle. Las HU solamente proporcionaran los detalles sobre la estimación del riesgo y del tiempo empleado para su implementación (51).

En la claridad de su descripción radica el éxito del proyecto. La falta de comunicación y la comprensión errónea de las necesidades del cliente son la principal causa de fracaso en proyectos pequeños. Es por ello que las HU juegan un papel tan importante en este proceso y se les dedica un gran esfuerzo para su realización efectiva. A continuación aparecen las HU de la solución:

Historia de Usuario	
No.: 1	**Nombre:** Desplazar la cámara mediante el teclado.
Usuario: Todos	
Prioridad en el Negocio: Alta.	**Nivel de Complejidad:** Alta.
Estimación: 2 semanas laborales	**Iteración Asignada:** 1
Descripción: El usuario puede desplazarse por todo el escenario y controlar la cámara, utilizando para esto las teclas A, S, D, W, *Ctrl*, *Shift*, ←, ↑, →, ↓. Si acciona la tecla *Ctrl*, se alternan las funciones asociadas a las teclas ←, →, A o D, entre desplazar la cámara de forma lateral o girar la cámara. Además accionando la tecla *Shift* permite desplazarse con una mayor rapidez en el escenario.	
Información adicional (Observaciones): Responde al aspecto funcional AF1, "Desplazar la cámara mediante el teclado".	

Historia de Usuario	
No.: 2	**Nombre:** Girar la cámara con el teclado.
Usuario: Todos	
Prioridad en el Negocio: Media.	**Nivel de Complejidad:** Baja.
Estimación: 1 semana laboral	**Iteración Asignada:** 1
Descripción: El usuario puede girar la cámara hacia la izquierda y hacia la derecha, utilizando para esto las teclas ←, →, A o D. Si se acciona la tecla *Ctrl*, se alternan las	

funciones asociadas a las teclas ←, →, A o D, entre desplazar la cámara de forma lateral o girar la cámara, permitiendo una mejor visión en el escenario hacia estas direcciones.
Información adicional (Observaciones): Responde al requisito AF2, “Girar la cámara con el teclado”.

Historia de Usuario	
No.: 3	**Nombre:** Girar la cámara con el *mouse* hacia todas las direcciones.
Usuario: Todos	
Prioridad en el Negocio: Alta.	**Nivel de Complejidad:** Media.
Estimación: 1 semana laboral	**Iteración Asignada:** 2
Descripción: El usuario puede girar la cámara hacia todas las direcciones, haciendo uso del *mouse*, para ello se debe hacer *clic* y arrastrar el mismo hacia la dirección en la que se quiera girar, posibilitando observar un punto específico del escenario.	
Información adicional (Observaciones): Responde al aspecto funcional AF3, “Girar la cámara con el *mouse* hacia todas las direcciones”.	

Historia de Usuario	
No.: 4	**Nombre:** Colisionar con las paredes y objetos del escenario.

Usuario: Todos	
Prioridad en el Negocio: Media.	**Nivel de Complejidad:** Media.
Estimación: 3 semanas laborales	**Iteración Asignada:** 4
Descripción: El usuario en su desplazamiento a través del escenario puede colisionar con los límites del mismo, como son paredes y objetos.	
Información adicional (Observaciones): Da cumplimiento al aspecto funcional AF5: “Colisionar con las paredes y objetos del escenario”.	

Historia de Usuario	
No.: 5	**Nombre:** Interactuar en el escenario.
Usuario: Todos	
Prioridad en el Negocio: Media.	**Nivel de Complejidad:** Media.
Estimación: 3 semanas laborales	**Iteración Asignada:** 3
Descripción: El usuario es capaz de encender y apagar luces haciendo *clic* en los encendedores a una determinada distancia.	
Información adicional (Observaciones): Da cumplimiento al aspecto funcional AF4: “Interactuar en el escenario”.	

Para el tiempo de duración (en semanas) de las HU se realizaron estimaciones de las mismas, teniendo en cuenta que son laborales cinco días a la semana.

Ejemplo de los días laborales en un mes:

Mes de ejemplo						
Domingo	**Lunes**	**Martes**	**Miércoles**	**Jueves**	**Viernes**	**Sábado**
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

	Días feriados
	Próximo mes
	Fin de semana
	Días laborales

2.7. Planificación

La planificación es una fase corta, en la que el cliente y el grupo de desarrolladores acuerdan el orden en que deben implementarse las HU y asociadas a éstas, las fechas de entrega. Típicamente esta fase consiste en una o varias reuniones de planificación, posibilitando estas la confección del plan de entregas que se detalla en el epígrafe 2.10 (50). Una vez terminado esto, se procede a organizar las HU en las iteraciones correspondientes, en dependencia de la prioridad especificada por el cliente y del tiempo de desarrollo de cada una.

2.8. Iteraciones

En las iteraciones son desarrolladas las funcionalidades del producto, generando al final de cada una un entregable funcional que implementa las HU asignadas a la iteración y que tiene un valor para el cliente. El mismo queda totalmente satisfecho al finalizar la última iteración, ya que se obtiene el producto acordado inicialmente. Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance (50), (13).

Para organizar las iteraciones se recomienda no extenderlas más de un mes laboral. Los plazos de entrega y retroalimentación largos atentan contra el cumplimiento de las expectativas del cliente. No hay mejor forma de evitar esto que probando el sistema en el entorno de despliegue del mismo. Las pruebas de los usuarios finales y las correcciones a las que se somete la aplicación en las revisiones, son la mejor forma de verificar que el *software* avanza adecuadamente.

Historias de usuario organizadas por iteraciones:

Iteraciones	HU a implementar	Tiempo de trabajo
1	Desplazar la cámara mediante el teclado	2 semanas laborales
1	Girar la cámara con el teclado	1 semana laboral
2	Girar la cámara con el *mouse* hacia todas las direcciones	1 semana laboral
3	Colisionar con las paredes y objetos del escenario	3 semanas laborales
4	Interactuar en el escenario	3 semanas laborales

2.9. Plan de entrega

El Plan de entrega establece cuáles HU serán agrupadas para conformar una entrega, y el orden de las mismas. Este elemento es el resultado de una reunión entre el cliente y los desarrolladores, haciendo el equipo de desarrollo un compromiso final con el cliente sobre la fecha de entrega del producto. Este es un elemento de vital importancia para el negocio entre ambas partes, ya que la entrega tardía o temprana de la solución, repercute notablemente en los recursos y moral de todos los involucrados. Tanta seriedad requiere este proceso, que luego de algunas iteraciones es recomendable realizar nuevamente una reunión con los miembros del proyecto, para evaluar el plan de entregas y ajustarlo si es necesario (50).

El plan de entregas se realiza en base a las estimaciones de tiempos realizadas por los desarrolladores, constituyendo esto uno de los temas más complicados del desarrollo de un proyecto de *software* y es por ello que resulta muy importante tener precisadas las necesidades del cliente, el estilo de trabajo del

equipo de desarrollo y el tiempo disponible por el cliente para tener en sus manos la solución. En el más extremo de los casos, la sinceridad debe predominar en lugar de la incertidumbre y es muy importante saber decir si se puede terminar el proyecto a tiempo o no. Es más conveniente ser claros con los clientes que defraudarlos luego de haber ocupado su tiempo y sus recursos (48).

Entregable	**Final 1ra iteración** **3ra semana diciembre**	**Final 2da iteración** **2da semana enero**	**Final 3ra iteración** **4ta semana marzo**	**Final 4ta iteración** **2da semana abril**
PVD5	Versión 1.0	Versión 1.5	Versión 2.0	Versión 2.5

2.10. Diseño

La metodología XP hace especial énfasis en los diseños simples y claros, ya que estos se implementan más rápido que los complejos y brindan la posibilidad de ganar en tiempo y eficiencia. Se sugiere nunca adelantar la implementación de funcionalidades que no correspondan a la iteración en la que se trabaja.

2.10.1. Descripción de la arquitectura

Para la construcción del EVTIW se utilizó una arquitectura cliente-servidor, como se muestra en la figura 1. En el servidor se almacena la aplicación en su totalidad, para su posterior acceso desde una estación cliente, siendo posible su ejecución a través de la máquina virtual *Flash Player* 10.

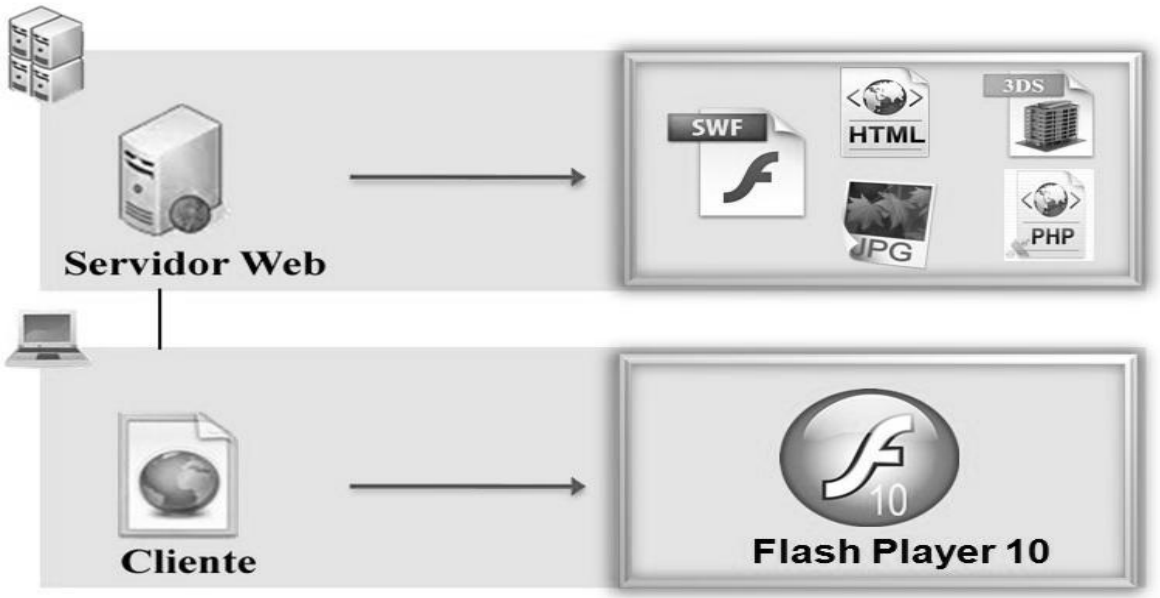


Figura 1. Arquitectura cliente-servidor

Dicha máquina virtual se encarga de manejar los cambios que se producen en la pantalla, produciendo las modificaciones requeridas solamente sobre aquellos elementos que necesitan ser actualizados, evitando de esta forma recargar completamente las páginas con el fin de optimizar la aplicación.

Debido a que el IDE seleccionado permite la creación de aplicaciones haciendo uso de varios ficheros SWF que se comunican entre sí, se logra evitar la centralización de la aplicación en determinados lugares posibilitando así la flexibilidad de la misma.

2.10.2. Diagrama de paquetes

En el desarrollo del presente EVTIW se empleó una estructura por paquetes, con el fin de agrupar funcionalidades del mismo carácter, motivo que posibilita una mayor reutilización de los diferentes elementos utilizados en este proceso.

La figura 2 constituye el diagrama de paquetes utilizado.

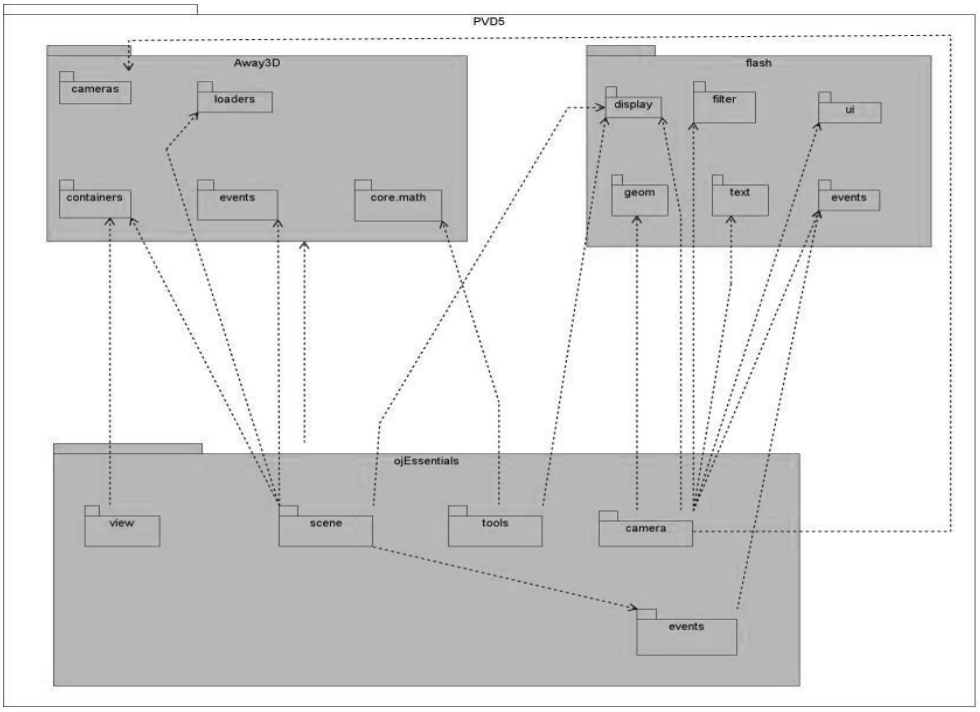


Figura 2. Diagrama de paquetes empleado en la arquitectura de la solución.

Paquete *Away3D*

Es la librería utilizada, la cual proporciona distintos paquetes para su empleo en el desarrollo de *software*. Ejemplo de esto es el paquete *camera* que contiene clases para representar los distintos tipos de cámaras utilizadas en los escenarios. También está el paquete *containers* el cual posee la clase *View3D* y *Scene3D*, ambas empleadas para la personalización de la vista y en la incorporación de elementos visuales al escenario respectivamente.

Paquete *Flash*

Agrupa distintas funcionalidades proporcionadas por el *Framework Flex* 4.0, como son el uso de eventos desde el paquete *events* y la utilización de los elementos del teclado desde el paquete *ui.*

Framework ojEssential

El *framework ojEssential* presenta cinco paquetes de clases, ***view***, ***camera***, ***tools***, ***scene***, y ***events***, respondiendo cada uno acorde con el elemento del escenario que le proporcionan funcionalidades. Cada clase agrupa responsabilidades específicas pertenecientes a los diferentes elementos que conforman un EVTIW.

El uso de *General Responsibility Asignment Software Patterns* (GRASP) como son el Patrón Experto y el Patrón Bajo Acoplamiento proporciona a ***ojEssential*** una mejor organización en cuanto a responsabilidades por clase, mejorando su rendimiento general al depender cada una de estas del menor número de elementos posibles.

2.10.3. Diagrama de clases del sistema

Los diagramas de clase brindan una vista general del funcionamiento de la aplicación, mostrando la interacción entre las clases utilizadas para el desarrollo de la misma. La imagen mostrada a continuación constituye el diagrama de clases de la arquitectura utilizada en el desarrollo del EVTIW.

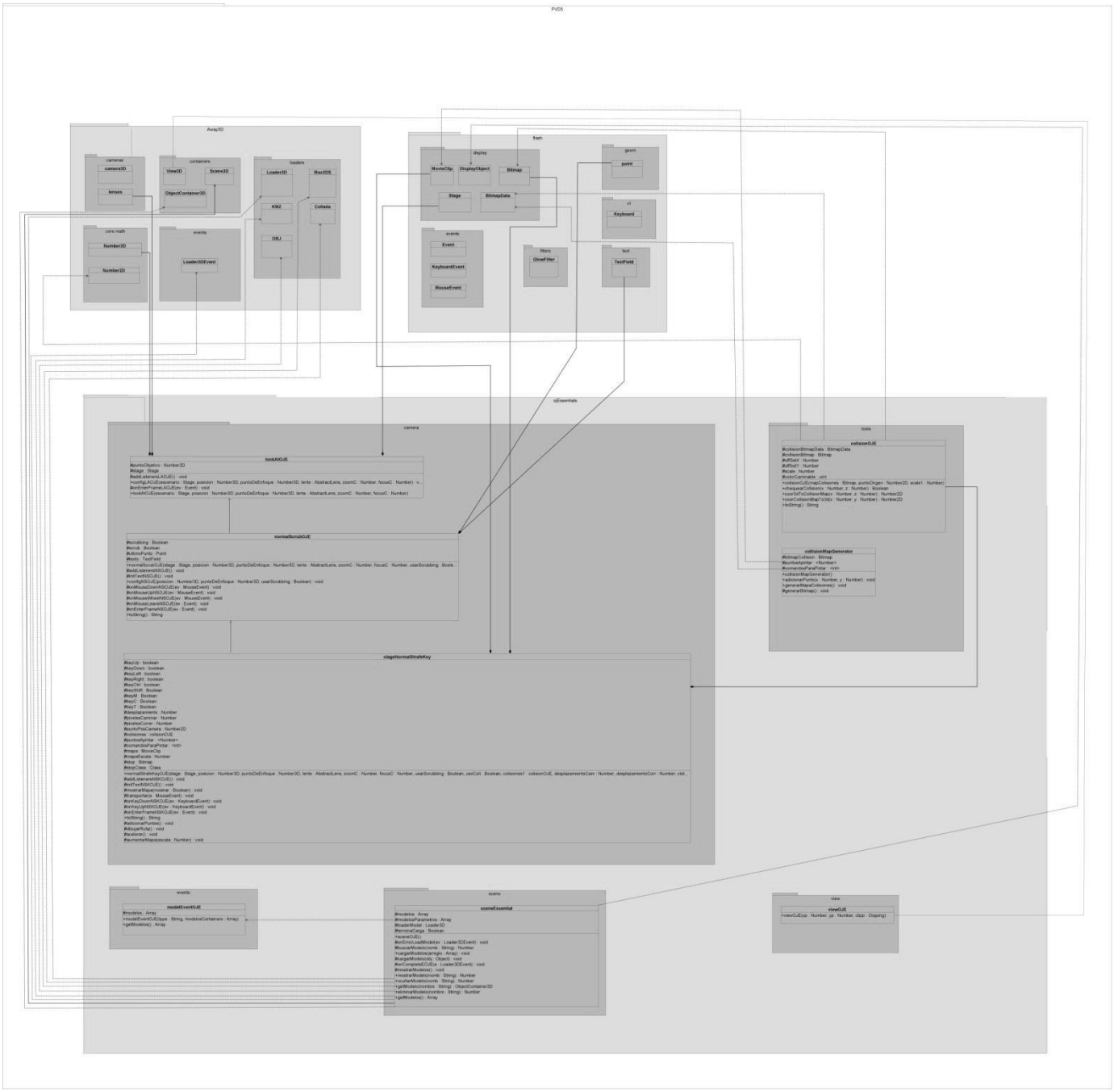


Figura 3. Diagrama de clases empleado en la arquitectura de la solución.

Capítulo 3. Implementación y prueba del caso de estudio.

3.1. Implementación

XP define un desarrollo por iteraciones, realizándose en cada una, la implementación de las historias de usuario establecidas en la etapa de diseño. Este desarrollo se encuentra guiado por pruebas, que se deben realizar cuando se termina la implementación de cada historia de usuario.

Uno de los aspectos más importantes a tener en cuenta para la implementación de las HU es el uso de estándares de codificación, de forma tal que se mantenga el código consistente y facilite su comprensión y escalabilidad. La creación de *tests* que prueben el funcionamiento del código implementado ayuda a desarrollar dicho código, conocer qué es exactamente lo que debe hacer, y así paulatinamente se consigue un desarrollo que cumple todos los requisitos especificados. Este es uno de los elementos tenidos en cuenta en la investigación, pues algunos fragmentos del código fuente de la solución, son plasmados en la guía de aprendizaje para el desarrollo de aplicaciones similares a la que se describe en el presente trabajo.

XP opta por la programación en pareja ya que permite un código más eficiente y con una gran calidad, mediante el uso de repositorios de código dónde se publican, cada pocas horas, los códigos implementados y corregidos por desarrolladores del equipo. Además propone un modelo de desarrollo colectivo en el que los programadores están implicados en todas las tareas; cualquiera puede modificar una clase o una función de otro programador si es necesario, actualizarla en el repositorio de código, facilitando que al final de cada iteración, se lleve a cabo la optimización del código en caso de que sea posible (52).

De la aplicación de estas buenas prácticas, definidas por la metodología XP, depende el cumplimiento de los requerimientos especificados por el cliente en las HU descritas en el epígrafe 2.7.

Para comenzar la implementación del producto, XP propone tener como base una arquitectura lo más flexible posible para evitar grandes cambios en las próximas iteraciones y en las modificaciones que propone el cliente. Teniendo en cuenta que la presente solución está enfocada a los detalles prácticos de forma que estos puedan ser recopilados en una guía de aprendizaje, es necesario tener bien definida la

arquitectura del *software*. Una vez trazada la misma, se procede a la primera iteración. (Ver epígrafe 2.10.1)

3.1.1. 1ra. Iteración

El principal objetivo de la primera iteración es desarrollar la **HU Nro. 1**: Desplazar la cámara mediante el teclado y la **HU Nro. 2**: Girar la cámara con el teclado.

Esta iteración arroja como resultado la versión 1.0 del producto, permitiendo que el usuario pueda desplazarse a través de todo el escenario y controlar la cámara, utilizando para esto las teclas A, S, D, W, *Ctrl*, *Shift*, ←, ↑, →, ↓. Si se acciona la tecla *Ctrl*, se alternarán las funciones asociadas a las teclas ←, →, A o D, entre desplazar la cámara de forma lateral o girar la cámara. Si acciona la tecla *Shift* permitirá desplazarse con una mayor rapidez en el escenario. Para su implementación se trazaron las tareas que se describen a continuación:

Tarea Nro. 1: Definición de la arquitectura de paquetes y de clases.

Tarea Nro. 2: Implementación de las clases principales de la aplicación.

Tarea Nro. 3: Definición de estándares de código y patrones de diseño.

Tarea Nro. 4: Implementación de la clase cámara con las funcionalidades de teclado.

Tarea	
Número de la tarea: 1	**Número HU:** 1 y 2
Nombre de la tarea: Definición de la arquitectura de paquetes y de clases.	
Tipo de tarea: Configuración.	**Estimación:** 5 días.
Fecha inicio: 1 de Febrero 2011.	**Fecha fin:** 5 de Febrero 2011

Responsable: Jesús Lázaro Suárez Pérez, Osvel Vargas Torres.

Descripción: Se crea la estructura de la aplicación basada en paquetes, lo que permite una mejor organización de la misma. Además se define la interacción entre las clases implicadas, así como las funcionalidades por clases para el desarrollo de la solución.

Tarea

Número de la tarea: 2 **Número HU:** 1 y 2

Nombre de la tarea: Implementación de las clases principales de la aplicación.

Tipo de tarea: Desarrollo. **Estimación:** 10 días.

Fecha inicio: 10 de Febrero 2011. **Fecha fin:** 20 de Marzo 2011

Responsable: Osvel Vargas Torres.

Descripción: Implementación de la clase principal de la aplicación utilizando la programación orientada a objetos, haciendo uso de algunas funciones de las clases definidas en el *framework ojEssential* y en la librería *Away3D*.

Paquetes: *ojEssential*

Fichero: *PVD5.mxml*

Tarea

Número de la tarea: 3 **Número HU:** 1 y 2

Nombre de la tarea: Definición de estándares de código y patrones de diseño.

Tipo de tarea: Configuración. **Estimación:** 4 días.

Fecha inicio: 5 de Febrero 2011. **Fecha fin:** 10 de Febrero 2011

Responsable: Jesús Lázaro Suárez Pérez, Osvel Vargas Torres.

Descripción: Definición del estándar de código a utilizar teniendo como base los utilizados internacionalmente en proyectos de *Actionscript* 3.0. Hacer un estudio de la documentación existente en la red para la definición del mismo.

Tarea

Número de la tarea: 4 **Número HU:** 1 y 2

Nombre de la tarea: Implementación de la clase cámara con las funcionalidades de teclado.

Tipo de tarea: Desarrollo. **Estimación:** 8 días.

Fecha inicio: 12 de Febrero 2011. **Fecha fin:** 20 de Febrero 2011

Responsable: Jesús Lázaro Suárez Pérez.

Descripción: Implementación de la clase cámara, mediante el uso de la librería *Away3D* y el *framework Flex* 4.0 para el desarrollo del giro y el desplazamiento lateral de la cámara en el escenario a través del teclado.

Paquetes: *ojEssential.camera*

Clases: *normalStrafeKeyOJE*

3.1.2. 2da. Iteración

El principal objetivo de la segunda iteración es desarrollar la **HU Nro. 3**: Girar la cámara con el *mouse* hacia todas las direcciones. Esta iteración aporta como resultado la versión 1.5 del producto, permitiendo que el usuario gire la cámara hacia todas las direcciones, haciendo uso del *mouse*, para ello se debe hacer *clic* y arrastrar el mismo hacia la dirección en la que se desea girar, posibilitando observar un punto específico del escenario. Para su implementación se trazaron las tareas que se describen a continuación:

Tarea Nro. 5: Desarrollo de las funciones requeridas para dar cumplimiento a la funcionalidad girar la cámara con el *mouse* en la clase *normalScrubOJE*.

Tarea Nro. 6: Implementación de la funcionalidad girar la cámara con el *mouse* hacia todas las direcciones en la clase principal de la aplicación utilizando la clase *normalScrubOJE*.

Tarea Nro. 7: Desarrollar un comportamiento de cámara sin utilizar la técnica de *Scrubbing* para incorporar en el *framework ojEssential*.

Tarea	
Número de la tarea: 5	**Número HU:** 3
Nombre de la tarea: Desarrollo de las funciones requeridas para dar cumplimiento a la funcionalidad girar la cámara con el *mouse* en la clase *normalScrubOJE*.	
Tipo de tarea: Desarrollo.	**Estimación:** 3 días.
Fecha inicio: 20 de Febrero 2011.	**Fecha fin:** 23 de Febrero 2011
Responsable: Jesús Lázaro Suárez Pérez.	
Descripción: Desarrollar las funciones *addListenersNSOJE*, *onMouseDownNSOJE*, *onMouseUpNSOJE*, *onMouseWheelNSOJE*, *onMouseLeaveNSOJE*, *onEnterFrameNSOJE*, *toString* para dar cumplimiento a la funcionalidad girar la cámara con el *mouse* en la clase *normalScrubOJE*.	

Paquetes: *ojEssential.camera*

Clases: *normalScrubOJE*

Tarea

Número de la tarea: 6 **Número HU:** 3

Nombre de la tarea: Implementación de la funcionalidad girar la cámara con el *mouse* hacia todas las direcciones en la clase principal de la aplicación utilizando la clase *normalScrubOJE*.

Tipo de tarea: Desarrollo. **Estimación:** 3 días.

Fecha inicio: 23 de Febrero 2011. **Fecha fin:** 26 de Marzo 2011

Responsable: Jesús Lázaro Suárez Pérez.

Descripción: Implementación de la función *createCamera* en la clase principal de la aplicación, con la utilización de la clase *ojEssential.camera.normalScrubOJE*.

Paquetes: *ojEssential.camera*

Fichero: *PVD5.mxml*

Tarea

Número de la tarea: 7 **Número HU:** 3

Nombre de la tarea: Desarrollar un comportamiento de cámara sin utilizar la técnica de *Scrubbing* para incorporar en el *framework ojEssential*.

Tipo de tarea: Desarrollo.	**Estimación:** 8 días.
Fecha inicio: 15 de Febrero 2011.	**Fecha fin:** 25 de Febrero 2011
Responsable: Osvel Vargas Torres.	
Descripción: Desarrollar otro tipo de comportamiento de la cámara como girar la cámara sin el uso de la técnica *"Scrubbing"* mediante la clase *normalScrubOJE.* **Paquetes:** *ojEssential.camera* **Clases:** *normalScrubOJE*	

3.1.3. 3ra. Iteración

El principal objetivo de la tercera iteración es desarrollar la **HU Nro. 4**: Colisionar con las paredes y objetos del escenario.

Esta iteración aportó como resultado la versión 2.0 del producto, permitiendo que en su desplazamiento a través del escenario pueda colisionar con los límites del mismo, como son paredes y objetos. Para su implementación se trazaron las tareas que se describen a continuación:

Tarea Nro. 8: Modelado del escenario tridimensional.

Tarea Nro. 9: Importación del modelo en la clase principal de la aplicación.

Tarea Nro. 10: Texturización del modelo tridimensional.

Tarea Nro. 11: Inicialización de los componentes del modelo en la clase principal.

Tarea Nro. 12: Desarrollo del mapa de colisiones según el modelo.

Tarea Nro. 13: Implementación de colisiones en la clase principal de la aplicación.

Tarea	
Número de la tarea: 8	**Número HU:** 4
Nombre de la tarea: Modelado del escenario tridimensional.	
Tipo de tarea: Modelado.	**Estimación:** 25 días.
Fecha inicio: 25 de Febrero 2011.	**Fecha fin:** 20 de Marzo 2011
Responsable: Jesús Lázaro Suárez Pérez.	
Descripción: Mediante el uso de la herramienta de modelado se construye la representación tridimensional del escenario.	

Tarea	
Número de la tarea: 9	**Número HU:** 4
Nombre de la tarea: Importación del modelo en la clase principal de la aplicación.	
Tipo de tarea: Desarrollo.	**Estimación:** 5 días.
Fecha inicio: 25 de Febrero 2011.	**Fecha fin:** 1 de Marzo 2011
Responsable: Osvel Vargas Torres.	
Descripción: Mediante el uso del *framework ojEssential* se importa el modelo tridimensional para visualizar su contenido en el escenario. **Paquetes**: *PVD5.assets.blender*	

Fichero: *PVD5.mxml*

Tarea

Número de la tarea: 10	**Número HU:** 4
Nombre de la tarea: Texturización del modelo tridimensional.	
Tipo de tarea: Diseño.	**Estimación:** 10 días.
Fecha inicio: 10 de Marzo 2011.	**Fecha fin:** 20 de Marzo 2011
Responsable: Osvel Vargas Torres.	
Descripción: Mediante el uso de *Blender* 2.5 se texturiza el modelo utilizando la técnica de mapeado de texturas y luego se exporta a formato 3ds.	

Tarea

Número de la tarea: 11	**Número HU:** 4
Nombre de la tarea: Inicialización de los componentes del modelo en la clase principal.	
Tipo de tarea: Desarrollo.	**Estimación:** 5 días.
Fecha inicio: 15 de Febrero 2011.	**Fecha fin:** 20 de Febrero 2011
Responsable: Jesús Lázaro Suárez Pérez.	
Descripción: Mediante el uso de *Away3D* y *ojEssential* se define la forma de representación de los	

componentes del modelo.

Paquetes: PVD5

Fichero: *PVD5.mxml*

Función: *initObjects*

Tarea

Número de la tarea: 12 **Número HU:** 4

Nombre de la tarea: Diseño del mapa de colisiones según el modelo.

Tipo de tarea: Diseño. **Estimación:** 5 días.

Fecha inicio: 15 de Marzo 2011. **Fecha fin:** 20 de Marzo 2011

Responsable: Jesús Lázaro Suárez Pérez.

Descripción: Se confecciona una imagen a dos colores, uno para permitir el movimiento del usuario en un área determinada y otro para impedírselo.

Tarea

Número de la tarea: 13 **Número HU:** 4

Nombre de la tarea: Implementación de colisiones en la clase principal de la aplicación.

Tipo de tarea: Desarrollo. **Estimación:** 20 días.

Fecha inicio: 20 de Marzo 2011.	**Fecha fin:** 10 de Abril 2011
Responsable: Osvel Vargas Torres.	
Descripción: Desarrollar las funciones y atributos necesarios en la clase principal para permitir al usuario colisionar con los límites del escenario. **Paquetes**: PVD5 **Fichero**: *PVD5.mxml*	

3.1.4. 4ta. Iteración

El objetivo de la primera iteración es desarrollar la **HU Nro. 5**: Interactuar en el escenario.

Esta iteración aporta como resultado la versión 2.5 del producto, permitiendo que el usuario pueda encender y apagar luces haciendo *clic* en los encendedores a una determinada distancia. Para su implementación se trazaron las tareas que se describen a continuación:

Tarea Nro. 14: Modelación de un encendedor con la herramienta de modelado.

Tarea Nro. 15: Incorporación el modelo del encendedor al modelo del escenario tridimensional.

Tarea Nro. 16: Implementación de la función *cambiarLuces* en la clase principal de la aplicación.

Tarea	
Número de la tarea: 14	**Número HU:** 5
Nombre de la tarea: Modelación de un encendedor con la herramienta de modelado.	
Tipo de tarea: Modelado.	**Estimación:** 3 días.

Fecha inicio: 15 de Marzo 2011. **Fecha fin:** 18 de Marzo 2011

Responsable: Osvel Vargas Torres.

Descripción: Mediante el uso de la herramienta de modelado se construye el modelo del encendedor.

Tarea

Número de la tarea: 15 **Número HU:** 5

Nombre de la tarea: Incorporación del modelo del encendedor al modelo del escenario tridimensional.

Tipo de tarea: Desarrollo. **Estimación:** 2 días.

Fecha inicio: 18 de Marzo 2011. **Fecha fin:** 20 de Marzo 2011

Responsable: Osvel Vargas Torres.

Descripción: Mediante el uso de la herramienta de modelado se incorpora el modelo del encendedor al modelo general.

Paquetes: PVD5

Fichero: *PVD5.mxml*

Tarea

Número de la tarea: 16 **Número HU: 5**

Nombre de la tarea: Implementación de la función *cambiarLuces* en la clase principal de la aplicación.

Tipo de tarea: Desarrollo.	**Estimación:** 15 días.
Fecha inicio: 20 de Marzo 2011.	**Fecha fin:** 5 de Abril 2011
Responsable: Jesús Lázaro Suárez Pérez.	
Descripción: Implementación de la función *cambiarLuces* en la clase principal de la aplicación, posibilitando esto el cambio en la iluminación del escenario. **Paquetes**: PVD5 **Fichero:** *PVD5.mxml* **Función:** *cambiarLuces*	

3.2. Pruebas

La metodología XP propone el desarrollo dirigido por pruebas, *Test Driven Development* (TDD). Lo primero que debe escribirse son los *tests* que se le deben realizar al sistema de forma tal que la producción de código esté dirigida por pruebas. Estas son denominadas pruebas unitarias realizadas por los desarrolladores a todos los módulos del proyecto antes de estos ser liberados. Se establecen antes de escribir el código y son ejecutadas constantemente ante cada modificación del sistema. En este contexto de desarrollo evolutivo y de énfasis en pruebas constantes, es crucial la automatización para apoyar esta actividad (50).

3.2.1. *Test Driven Development* (TDD) o pruebas unitarias

TDD se enfoca en la implementación basada en pruebas. Exige que el código deba ser probado paso a paso y obtener un resultado funcional. Se aplica antes de comenzar a implementar la tarea en desarrollo, asumiendo que la prueba es insatisfactoria desde un inicio. Solo una vez que se haya cumplido la lógica del código a probar de la forma más sencilla posible, se asume como satisfactoria. Luego se realiza un

proceso de refactorización de código, el cual consiste en organizarlo y adaptarlo a patrones de diseño. En esencia, TDD se centra en la lógica del código (53).

Actionscript 3.0 no cuenta con un *framework* de pruebas unitarias, por lo que es necesario utilizar el depurador de *Flash Builder* 4 (48).

Una vez terminada cada tarea definida, se procede a probar las funcionalidades definidas. Luego se realizan las pruebas de “caja gris”, en la cual se valida el comportamiento general de la solución en ambientes adversos como la pérdida de conectividad, pocos recursos de *hardware* y sobrecarga del mismo (54). Una vez hecho esto, el usuario final realiza las llamadas pruebas de aceptación.

3.2.2. Pruebas de aceptación

Son creadas en base a las HU en cada iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una de las HU ha sido correctamente implementada.

Las pruebas de aceptación son consideradas como “pruebas de caja negra”. Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. En caso que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una HU no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es de todo el equipo de desarrollo, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todos estén al tanto de esta información (50). Su objetivo es comprobar, desde la perspectiva del usuario final, el cumplimiento de las especificaciones en cuanto a las funcionalidades del producto.

A continuación, se relacionan las pruebas de aceptación realizadas a la solución propuesta:

Caso de prueba de aceptación	
Código: CP1_HU1	**Historia de Usuario:** 1
Nombre: Desplazamiento de la cámara con el teclado	
Descripción: Prueba para la funcionalidad de desplazamiento de la cámara con el teclado.	

Condiciones de ejecución: El usuario debe haber iniciado la aplicación a través de un navegador y debe tener instalado *Flash Player* 10 en su estación de trabajo.

Entrada/Pasos de ejecución: El usuario presiona una de las teclas A, S, D, W, ←, ↑, →, ↓ para desplazarse en el escenario.

Resultado esperado: Si presiona la tecla “A” el desplazamiento es hacia la izquierda, “S” hacia atrás, “D” hacia la derecha, “W” hacia adelante. Si presiona alguna de las teclas ←, ↑, →, ↓ se desplaza hacia la dirección que estas indican.

Evaluación de la prueba: Satisfactoria.

Caso de prueba de aceptación

Código: CP2_HU2 **Historia de Usuario:** 2

Nombre: Giro de la cámara con el teclado.

Descripción: Prueba para la funcionalidad de giro de la cámara con el teclado.

Condiciones de ejecución: El usuario debe haber iniciado la aplicación a través de un navegador y debe tener instalado *Flash Player* 10 en su estación de trabajo.

Entrada/Pasos de ejecución: El usuario acciona la tecla *Ctrl* para alternar las funciones asociadas a las teclas ←, →, A y D, entre desplazar la cámara de forma lateral o girar la cámara y posteriormente trata de girar la cámara a través de las teclas antes mencionadas.

Resultado esperado: Si presiona la tecla “A” gira hacia la izquierda, “D” hacia la derecha. Si presiona alguna de las teclas ← o → gira hacia la dirección que estas indican.

Evaluación de la prueba: Satisfactoria.

Caso de prueba de aceptación

Código: CP3_HU3 **Historia de Usuario:** 3

Nombre: Giro de la cámara con el *mouse*.

Descripción: Prueba para la funcionalidad de giro de la cámara con el *mouse*.

Condiciones de ejecución: El usuario debe haber iniciado la aplicación a través de un navegador y debe tener instalado *Flash Player* 10 en su estación de trabajo.

Entrada/Pasos de ejecución: El usuario hace *clic* y arrastra el *mouse* hacia la dirección en la que desea girar la cámara.

Resultado esperado: Gira la cámara en la dirección que el usuario desplaza el *mouse*.

Evaluación de la prueba: Satisfactoria.

Caso de prueba de aceptación

Código: CP4_HU4 **Historia de Usuario:** 4

Nombre: Colisionar con paredes y objetos del escenario.

Descripción: Prueba para la funcionalidad colisionar con paredes y objetos del escenario.

Condiciones de ejecución: El usuario debe haber iniciado la aplicación a través de un navegador y

debe tener instalado *Flash Player* 10 en su estación de trabajo.

Entrada/Pasos de ejecución: Se procede a desplazar la cámara con las teclas W, A, S, D, ←, ↑, →, ↓ hasta chocar con una pared u otro objeto en el escenario.

Resultado esperado: Aunque se siga presionando cualquier tecla de las anteriores para desplazarnos contra los límites del escenario, la cámara no se desplaza en esa dirección sufriendo una variación en su movimiento.

Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación

Código: CP5_HU5 **Historia de Usuario:** 5

Nombre: Encendido y apagado de luces en el escenario.

Descripción: Prueba para la funcionalidad de encendido y apagado de luces en el escenario.

Condiciones de ejecución: El usuario debe haber iniciado la aplicación.

Entrada/Pasos de ejecución: Se procede a desplazar la cámara con las teclas W, A, S, D, ←, ↑, →, ↓ para acercar la misma hacia un encendedor de la luz, cuando esté cerca se hace *clic* en el mismo.

Resultado esperado: Se apagan o se encienden las luces.

Evaluación de la prueba: Satisfactoria

Capítulo 4. Guía de aprendizaje para el desarrollo de escenarios virtuales tridimensionales interactivos en la Web.

4.1. Introducción

La creación de EVTIW es uno de los temas más demandados especialmente por los desarrolladores de *software* que centran sus esfuerzos en la Web. Para este tipo de desarrollo existen importantes aspectos como el IDE, el *framework*, la biblioteca de código y el lenguaje de programación a utilizar. Cierta preparación en estas temáticas se hace necesaria, para poder llevar a cabo la construcción de una correcta simulación. Debido a esto es frecuente en los desarrolladores la búsqueda de bibliografía que brinde información de manera clara y concisa, especialmente si son inexpertos en el tema. Dentro de los tipos de fuentes bibliográficas disponibles para estas temáticas se encuentran los libros, las revistas, los *blogs*, los manuales y las guías de aprendizaje (56).

Las guías de aprendizaje son un tipo de bibliografía que proporciona información teórica y práctica, concentrándose en esta última de manera rápida, ya que por lo general no son tan extensas como los libros. Como todos los aspectos de la vida, sufrieron su transformación con el surgimiento de Internet, logrando así intensificar sus objetivos a partir del uso de las ventajas que representa el medio digital. Como consecuencia de este aprovechamiento resurgen las guías de aprendizaje pero desde los medios digitales, abordando cualquier temática vinculada o no a la informática y haciendo uso de recursos como imágenes, videos, gráficos y capturas de pantallas (57), (58). Cada uno de estos recursos se modifican de acuerdo con la temática que se trata, en el caso de las guías enfocadas a los desarrolladores de *software* existe otro factor fundamental, el código fuente. Con el aprovechamiento de este elemento y su entrelazada relación con los anteriores mencionados, vuelven a la práctica las guías de aprendizaje (59).

Uno de los resultados consiste en una guía de aprendizaje para desarrollar EVTIW utilizando el *framework ojEssential*, fruto también de este trabajo. A continuación se describe el contenido de dicha guía.

4.2. Estructura de la guía de aprendizaje para el desarrollo de escenarios virtuales tridimensionales interactivos en la Web

Escenarios virtuales tridimensionales en la Web. Conceptos básicos

Las aplicaciones 3D en la Web presentan un gran auge en el mundo de la Informática debido al alto nivel de realismo que se logra representar. De ahí que muchos desarrolladores se incorporen a la creación de este tipo de *software*.

En este capítulo se describen los conceptos fundamentales que deben dominarse para la construcción de aplicaciones 3D en la Web, haciendo énfasis en los elementos que conforman un EVTIW tales como, la vista, la escena, y la cámara. Además se brindan los pasos a seguir para la configuración del IDE *Adobe Flash Builder* 4.

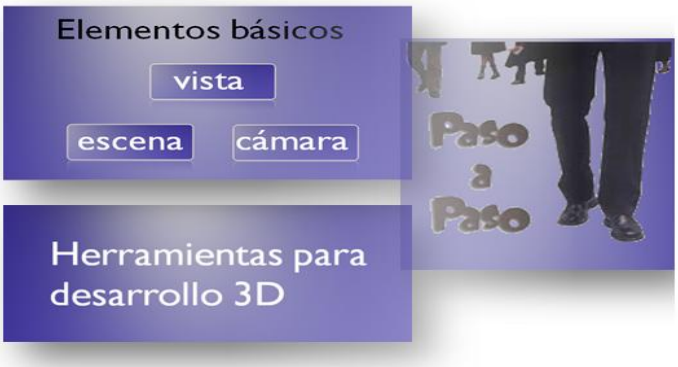


Ilustración 1. Elementos del Capítulo 1.

Capítulo 2

Framework ojEssential

El capítulo 2 es la esencia de la guía de aprendizaje, en él se describe detalladamente el *framework ojEssential* como elemento clave para el desarrollo de EVTIW. *ojEssential* incorpora un conjunto de clases organizadas por paquetes en dependencia de su funcionalidad, que permiten el desarrollo rápido y eficiente de EVTIW tan complejos como el desarrollador desee crear.

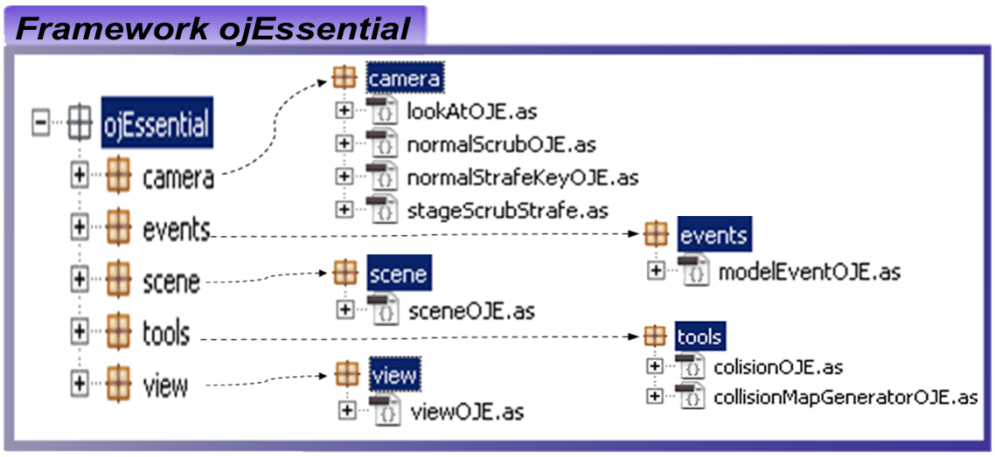


Ilustración 2. Estructura del *framework ojEssential*.

Capítulo 3

Desarrollo de EVTIW empleando *ojEssential*

En este capítulo se describen los pasos a seguir para la implementación de un EVTIW haciendo uso del *framework ojEssential* especificando dos variantes: sin el uso de modelos 3D creados con alguna herramienta CAD y usando estos. Además se brinda el código fuente de los casos de estudio PVD5 y EVTIWBasico.

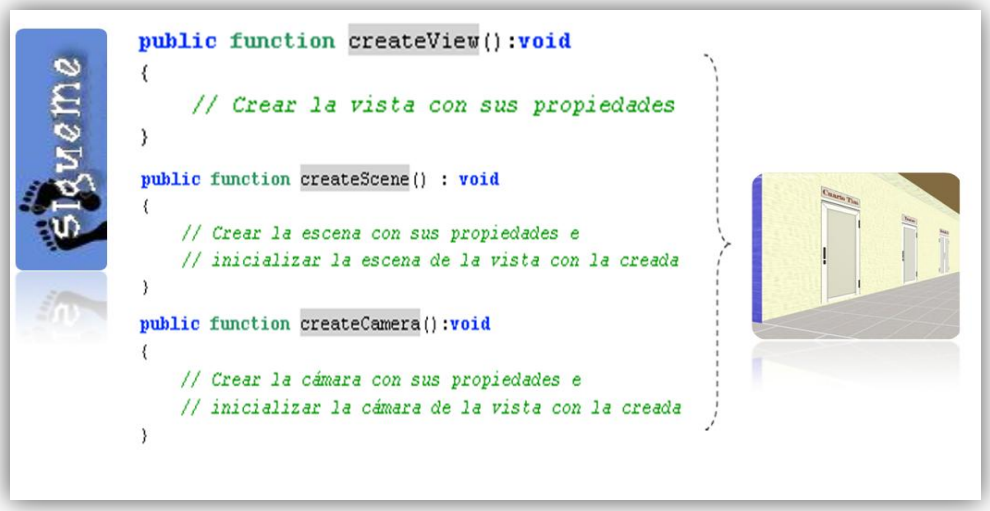
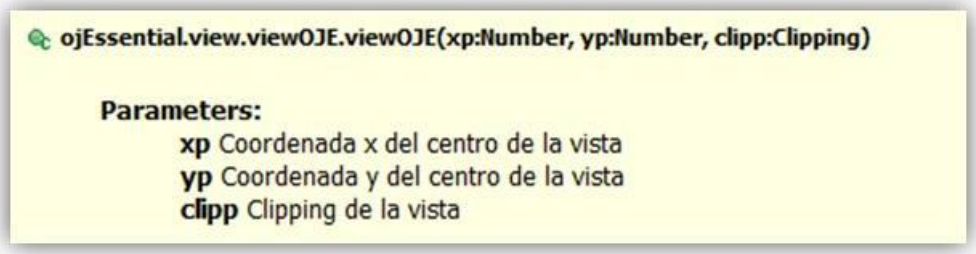


Ilustración 3. Elementos del Capítulo 3.

4.3. Elementos clave presentes en la guía.

Uno de los factores que hace a las guías de aprendizaje más entendibles es el uso de diagramas, ilustraciones y descripciones. En la guía de aprendizaje se utilizaron elementos como imágenes, fragmentos de código, capturas de pantallas, notas, advertencias y vínculos de descarga. Estos elementos resultan de vital importancia para el entendimiento de la temática abordada en cada epígrafe de la guía.

Imágenes



Fragmentos de código

```
public function createCamera():void
{
	pasoApaso=false;

	//Inicializamos la camara a utilizar

	camera=new lookAtOJE(stage,new Number3D(0,0,0),new Number3D(0,0,-
	40),
	new SphericalLens(),4,166);

	camera.name="camera";

	//Actualizamos la propiedad camara de la vista

	view.camera=camera;

	progreso(20);
}
```

Capturas de pantalla

Notas

Clipping *es el proceso de recortar toda la representación 3D de lo que se encuentra en la escena, para definir un área rectangular a través de la cual se observa el escenario.*

Advertencias

Si selecciona un proyecto que ya existe se mostrará el mensaje "A project with the same name already exists in your workspace. Rename or delete the existing project.", indicando que existe un proyecto con el mismo nombre del proyecto que se está importando.

Vínculo de descarga

Away3D *es una librería de código abierto para el desarrollo de escenarios virtuales 3D en la Web. Disponible para su descarga en la sesión Downloads de su sitio oficial* http:/www.away3d.com.

Conclusiones

- El uso de la metodología *eXtreme Programming* (XP) permitió la adaptación de la aplicación ante los distintos cambios que surgieron.
- El desarrollo del *framework ojEssential* permitió simplificar la implementación de la aplicación, logrando una descripción más sencilla de este proceso en la guía de aprendizaje.
- El uso de estándares de código posibilitó brindarle a los usuarios de la guía un análisis más detallado del código fuente del caso de estudio.
- La utilización de algunos recursos como imágenes, notas, capturas de pantalla, fragmentos de código y vínculos de descargas, posibilitaron la visualización de los temas abordados en la guía.
- La realización de la guía de aprendizaje, como principal producto de la presente investigación, permitió concentrar la información necesaria para llevar a cabo el desarrollo de EVTIW, mediante la utilización del *framework ojEssential*.
- Esta investigación fue presentada en eventos como el “IX Fórum de Ciencia y Técnica” y en la IX Jornada Científica Estudiantil obteniendo resultados de destacada en ambos eventos. Además fue premiada en la primera edición del evento “Mi Tesis es Ciencia” de la Facultad 4.

Recomendaciones

- Complementar la guía con videos-tutoriales.
- Adaptar la guía de aprendizaje para su despliegue en la Web.
- Ampliar el *framework ojEssential* para hacer posibles las colisiones tridimensionales.
- Implementar varias clases que puedan utilizarse como plantillas en la construcción de escenarios virtuales.

Referencias bibliográficas

1. **Palacios, Eliseo.** Aplicaciones ricas en Internet (RIA). Un enfoque de refactorizacion. 2008.

2. **Tecnolives.** Glosario de términos en Second Life. [En línea] http://www.tecnolives.com/glosario-de-terminos-en-second-life/.

3. LA WEB 2.0 Y 3.0: DE LOS DATOS AL CONOCIMIENTO. **Fernández Nodarse, Francisco.** 2010.

4. **Kaplan EduNeering's Headquarters.** KAPLAN EDUNEERING. [En línea] http://www.kaplaneduneering.com/kappnotes/index.php/2007/01/definition-massively-multi-learner/.

5. **VGCL.** THE VIDEO GAME CONSOLE LIBRARY. [En línea] http://www.videogameconsolelibrary.com/art-dictionary.htm.

6. **Cryo.** Cryopolis. [En línea] http://www.cryopolis.com/.

7. **Linden Lab.** Secondlife. [En línea] http://secondlife.com/.

8. **Martínez Mercado, María Cristina.** IV Congreso de Imagen y Pedagogía. [En línea] http://www.conimagen.dgme.sep.gob.mx/memorias/documento_cm.doc.

9. **González Arencibia, Mario.** Ministerio de Hacienda. [En línea] http://www.hacienda.go.cr/centro/datos/Libro/Gu%C3%ADa%20de%20aprendizaje%20de%20%C3%A9tica%20inform%C3%A1tica.pdf.

10. **iwexcoder.** Aplicaciones RIA.Programación en AJAX, Flex,OpenLaszlo,Silverlight,JavaFX Script. [En línea] http://www.iwexcoder.com/index.php/aplicaciones-ria/.

11. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. 2000.

12. **Marcelo Hernán, Schenone.** Diseño de una Metodología Ágil de Desarrollo de Software. 2004.

13. Métodologías Ágiles en el Desarrollo de Software. **H. Canós, José, Letelier, Patricio y Carmen Penadés, Maria.** 2002.

14. **Monografias.com S.A.** monografias.com. [En línea] http://www.monografias.com/trabajos15/nvas-tecnologias/nvas-tecnologias.shtml.

15. **Adobe.** Características de Flex Builder 3. [En línea] http://www.adobe.com/es/products/flex/features/flex_builder/.

16. **Microsoft.** Frequently Asked Questions Silverlight. [En línea] http://www.microsoft.com/silverlight/faq/.

17. **Adobe.** What's new in Flash Builder 4. [En línea] http://www.adobe.com/devnet/flex/articles/flashbuilder4_whatsnew.html.

18. **Real Academia Española.** REAL ACADEMIA ESPAÑOLA. [En línea] http://buscon.rae.es/draeI/SrvltGUIBusUsual?LEMA=biblioteca.

19. **Slideshare.** Away3d workshop. [En línea] http://www.slideshare.net/jensa/away3d-workshop-slides.

20. **Cycle-IT.** Blog Oficial de CycleIt, Conclusiones. [En línea] http://nosmoke.cycle-it.com/2010/11/04/3d-conclusiones/.

21. **Cycle-IT.** Blog oficial de Cycle IT, Sandy. [En línea] http://nosmoke.cycle-it.com/2010/11/02/sandy/.

22. **BATEMAN, ROB y OLSSON, RICHARD.** The Essencial Guide To 3D in Flash. 2010.

23. **European Commission's Seventh Framework Programme.** d4 SCIENCE. [En línea] http://www.d4science.eu/glossary.

24. **Adobe Systems Incorporated.** Adobe Open Source. [En línea] http://opensource.adobe.com/wiki/display/flexsdk/Flex+4.

25. **Universidad Autónoma de Barcelona.** dipósit digital de documents de la UAB. [En línea] http://ddd.uab.cat/pub/trerecpro/2010/hdl_2072_83173/INFORME+FINAL+MTIG11+ROY+A.+JUSTO.pdf.

26. **Answers.com.** Answers.com. [En línea] http://www.answers.com/topic/programming-language.

27. **Lenguajes de Programación 2009.** Lenguajes de Programación. [En línea] http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml.

28. **Ecma International.** Standards@Internet Speed. [En línea] http://www.ecma-international.org.

29. **Ecma International.** Standard ECMA-262 ECMAScript Language Specification . [En línea] http://www.ecma-international.org/publications/standards/Ecma-262.htm.

30. **Adobe.** ActionScript Technology Center. [En línea] http://www.adobe.com/devnet/actionscript.html.

31. **Adobe Corporation.** PROGRAMACIÓN CON ACTIONSCRIPT™ 3.0. 2008 .

32. **Blender Foundation.** Blender Manual. [En línea] http://wiki.blender.org/index.php/Doc:Manual/Introduction.

33. **Blender Foundation.** blender. [En línea] http://wiki.blender.org/index.php/Extensions:2.4/Py/Scripts.

34. **Blender Foundation.** Comunidad de Blender. [En línea] http://wiki.blender.org/index.php/Doc:Manual/Introduction/Community.

35. **Enrique y Javier.** Historia de Autocad. [En línea] http://www.buenastareas.com/categorias/Tecnolog%C3%ADa/14/0.html.

36. **Autodesk.** Autocad Trabajando con Datos en otros Formatos. [En línea] http://docs.autodesk.com/ACD/2011/ENU/filesAUG/WSfacf1429558a55de13a2b791006b29c3f6-7a16.htm.

37. **Google Corporation.** Novedades de Google SketchUp 8. [En línea] http://sketchup.google.com/intl/es/product/newin8.html.

38. **Visual Paradigm International.** UML, BPMN and Database Tool for Software Development. [En línea] http://www.visual-paradigm.com/product/vpuml/.

39. **Fernández Acebal, César.** Escuela de Ingeniería Informática de la Universidad de Oviedo. [En línea] http://aainfo.ccu.uniovi.es/ficheros/apuntes/comercio_electronico/Tema%206%20-%20Servidores%20de%20aplicaciones.pdf.

40. **Morrison, Aileen.** 2do ENCUENTRO NACIONAL DE LINUX. [En línea] http://2001.encuentrolinux.cl/documentacion/ServidorWebApache.pdf.gz.

41. **The Apache Software Foundation.** Apache HTTP SERVER PROJECT. [En línea] http://httpd.apache.org/.

42. **Ciberaula.** Área Temática de Linux. Portal de Linux. [En línea] http://linux.ciberaula.com/articulo/linux_apache_intro/.

43. **DeConceptos.** Concepto de guía. [En línea] http://deconceptos.com/general/guia.

44. **Monografias.com S.A.** Estrategias de aprendizaje. [En línea] http://www.monografias.com/trabajos13/tecnes/tecnes.shtml.

45. **Freeman, Eric y Freeman, Elisabeth.** Head First Desing Patterns. 2004.

46. **Stichting Blender Foundation.** Volumen I de la Documentación de Blender - Guía de Usuario. [En línea] http://es.scribd.com/doc/6134018/Manual-Blender-Opt.

47. **OMG Project Management Institute.** MILESTONE. [En línea] http://www.milestone.com.mx/CursoAdmReqCasosDeUso.htm.

48. **Díaz Gutiérrez, Jorge Antonio.** Desarrollo de un IDE libre y multiplataforma para la creación de componentes visuales de ActionScript para Software Educativo: codeDraw. 2009.

49. **Scribd.** El alcance del proyecto. [En línea] http://es.scribd.com/doc/4736336/El-Alcance-del-Proyecto.

50. **Cisa, Agustín.** Instituto de Ingeniería Eléctrica "Prof. Ing. Agustín Cisa". [En línea] http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf.

51. **Escribano, Gerardo Fernández.** UCLM Universidad de Castilla la Mancha. [En línea] www.info-ab.uclm.es/asignaturas/42551/.../Presentacion-XP.pdf.

52. **Castillo, Oswaldo, Sevilla, Hector y Figueroa, Daniel.** tripod. [En línea] http://programacionextrema.tripod.com/fases.htm.

53. **Blé Jurado, Carlos, y otros.** Dirigido por Tests. [En línea] http://www.dirigidoportests.com/wp-content/uploads/2009/12/disenoAgilConTDD.pdf.

54. **A2Secure.** A2Secure. [En línea] http://www.a2secure.com/auditorias/test-de-intrusion.

55. **Linden Lab, Second Life Residents.** Second Life Wiki. [En línea] http://wiki.secondlife.com/wiki/LSL_Portal.

56. **Gassner, David.** Flash® Builder™ 4 and Flex® 4 Bible. 2011.

57. **SOLIMAN.** Apuntes de Blender de SOLIMAN. [En línea] http://apuntesdeblender.iespana.es/2coloresFondo.htm.

58. **la100rra.** Aprender Dreamweaver CS4 - Primeros pasos: 02 [Video Tutorial]. [En línea] http://la100rra.mx/video-tutorial-aprender-dreamweaver-cs4-primeros-pasos-02/.

59. **J., Ing. Alfonso E. Martínez de Castro.** tutorialphp.net. [En línea] http://tutorialphp.net/cap7_2base_de_datos_mysql_conectar_a_base_de_datos_php.php.

60. **Cycle-IT.** Blog Oficial de CycleIt. [En línea] http://nosmoke.cycle-it.com/2010/11/04/3d-conclusiones/.

61. **papervision3d.org.** Papervision3D API Documentation. [En línea] http://www.papervision3d.org/docs/as3/org/papervision3d/cameras/Camera3D.html.

62. **Google Corporation.** Google Code. [En línea] http://code.google.com/p/sandy/.

63. **Cycle-IT.** Blog Oficial de Cycle IT, AWAY3D. [En línea] http://nosmoke.cycle-it.com/2010/07/19/away/.

64. **Cycle IT.** labs.cycle-it.com. [En línea] http://labs.cycle-it.com/.

65. **Thomas Pfeiffer.** Sandy 3D engine (AS3 & AS2) for Adobe Flash. [En línea] http://www.flashsandy.org/technotes.

66. **Blender Foundation.** Blender History. [En línea] http://www.blender.org/blenderorg/blender-foundation/history/.

67. **Grupo ASDSC.** Historia de Autocad. [En línea] http://arkinetia.com/Recursos/art92.aspx.

68. **Google Corporation.** Qué hace que SketchUp sea una maravilla. [En línea] http://sketchup.google.com/intl/es/product/gsu.html.

69. **Adobe.** Flex´s FAQ. [En línea] www.adobe.com/products/flex/faq/open_source.

70. **Microsoft.** What is Silverlight. [En línea] http://www.microsoft.com/silverlight/what-is-silverlight/.

71. **Adobe.** Requisitos del sistema Flash Builder. [En línea] http://www.adobe.com/es/products/flex/systemreqs/.

Glosario de términos

RIA: *Rich Internet Applications* (Aplicaciones Enriquecidas de Internet) son un nuevo tipo de aplicaciones con más ventajas que las tradicionales aplicaciones web. Esta surge como una combinación de las ventajas que ofrecen las aplicaciones web y las aplicaciones tradicionales.

UML: Lenguaje Unificado de Modelado (UML por sus siglas en inglés) es el lenguaje de modelado de sistemas de *software* más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

REST: *Representational State Transfer,* es un servicio basado en el concepto de transferencia de estado entre sistemas a través de la red. Con REST los paquetes de datos que son transferidos por el servicio web deben ser identificados individualmente por medio de un esquema estándar de direccionamiento.

SOAP: Define cómo se utilizan XML y HTTP para tener acceso a servicios, objetos y servidores independientemente del sistema operativo.

SDK: Conjunto de herramientas de desarrollo que le permite a un programador crear aplicaciones para un sistema concreto, por ejemplo ciertos paquetes de *software* y los *frameworks*.

MXML: *Macromedia eXtensible Markup Language*, es un lenguaje que describe interfaces de usuario, crea modelos de datos y tiene acceso a los recursos del servidor, del tipo RIA.

***Flash Player* 10:** *Adobe Flash Player* es una aplicación en forma de reproductor multimedia creado inicialmente por *Macromedia* y actualmente distribuido por *Adobe Systems*. Permite reproducir archivos SWF que pueden ser creados con la herramienta de autoría *Adobe Flash*, con *Adobe Flex* o con otras herramientas de *Adobe* y de terceros. Estos archivos se reproducen en un entorno determinado. En un sistema operativo tiene el formato de aplicación del sistema, mientras que si el entorno es un navegador, su formato es el de un Plug-in u objeto *ActiveX*.

Anexos

Anexo 1: Metodologías de Desarrollo de *Software*

Tabla comparativa. Metodologías tradicionales y Metodologías ágiles. (11), (12), (13)

Metodologías tradicionales	**Metodologías ágiles**
Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo	Basadas en heurísticas provenientes de prácticas de producción de código
Se centran especialmente en el control del proceso.	Se centra fundamentalmente en el factor humano o el producto *software*.
Cierta resistencia a cambios.	Flexible a cambios.
Proceso muy controlado con numerosas políticas y normas.	Proceso menos controlado, centrado en suplir las necesidades del cliente.
La arquitectura de *software* es esencial y se expresa en modelos.	Menos énfasis en la arquitectura del *software*.
Equipos grandes y posiblemente distribuidos en diferentes sitios.	Equipos pequeños (menos de diez integrantes) y trabajando en el mismo sitio.
El equipo interactúa con el cliente mediante reuniones planificadas.	El cliente es parte del equipo de desarrollo.
Se genera gran cantidad de artefactos.	Se generan solo los artefactos necesarios definidos por el equipo de desarrollo.
Se establecen muchos roles.	Presentan pocos roles.

Anexo 2: Scrum vs XP

Tabla comparativa. Metodología de desarrollo XP y Metodología SCRUM. (13), (12)

Nombre / **Indicador**	**SCRUM**	**XP**
Base	Centrada en las prácticas de organización y gestión de proyecto.	Centrada en las buenas prácticas de programación.
Características	Hace hincapié en la parte de gestión de un proyecto y no define prácticas de ingeniería de *software* en detalle.	Especifica las prácticas de ingeniería de *software* tales como el desarrollo guiado por pruebas, la refactorización, la programación en parejas, la integración continua, el diseño incremental.
Utilización de técnicas de desarrollo de *software* ágil	Sólo se beneficia de algunas de las técnicas de desarrollo de *software* ágil.	Se beneficia de todas las técnicas de desarrollo ágil.
Flexibilidad a cambios	Tienen que esperar hasta concluir la iteración para introducir el cambio.	Es más aceptable para el cambio durante la iteración.
Tiempo de cada Iteración	Treinta días.	Como máximo catorce días.

Anexo 3: Bibliotecas de código

Tabla comparativa. Bibliotecas de código (60), (61)

Nombre / **Indicador**	**Sandy3D**	**Papervision3D**	**Away3D**
Licencia	*Mozilla Public License* (62)	*Apache* 2.0.	*Apache* 2.0 (63)
Lenguaje de Programación	*Actionscript* 2.0	*Actionscript* 3.0	*Actionscript* 3.0
Inicialización de objetos (63)	Los constructores admiten gran número de parámetros.	Los constructores admiten gran número de parámetros.	Constructores con mínimo de parámetros, en muchos casos solo el parámetro *Object*.
Tasa de cuadros por segundo	Ochenta cuadros por segundo	Cien cuadros por segundo	Ciento once cuadros por segundo.
Eficiencia	Baja	Media	Alta (20).
Rendimiento (64), (65)	Capaz de mostrar de dos mil a cinco mil polígonos correctamente.	Capaz de mostrar cuatro mil ochocientos polígonos correctamente.	Capaz de mostrar cuatro mil ochocientos a cinco mil polígonos correctamente.
Frecuencia de actualización	Sólo veinte contribuciones en los últimos diez meses de 2010.	Ha disminuido.	Se mantiene al día, evolucionando con mejoras, correcciones, nuevos ejemplos y documentación

Comunidad de desarrollo	Ha decrecido gradualmente.	Ha disminuido en los últimos años	Se mantiene en constante evolución

Anexo 4: Herramientas de modelado 3D

Tabla comparativa. Herramientas de modelado 3D

Nombre / **Indicador**	***Blender***	***AutoCad***	***Google Sketchup***
Empresa desarrolladora	Fundación *Blender* (66)	*Autodesk* (35)	*Google*
Licencia	GPL (66), (32)	*Software* Propietario	*Software* Propietario
Sistema Operativo	Multiplataforma (32)	*UNIX*, *MS-DOS*, *Windows* y *MAC* (67)	*Windows* y *MAC*
Objetivo de surgimiento	Mejorar herramienta de animación (66)	Desarrollar una herramienta para el diseño.	Herramienta para acelerar el diseño en 3d.
Extensible	Si	Si	Si
Formatos de exportación	3DS,DAE,FBX,OBJ,X 3D (33)	DWF,PDF,DXF,FBX (36)	DAE,KMZ, 3DS,OBJ,XSI,FBX,VRML, DXF, DWG (68)
Facilidad de uso	Media	Media	Alta
Recursos que	Motor de juegos 3D,	Dibujo paramétrico,	Modelador de edificios,

brinda	cinemática inversa (34)	bloques dinámicos, administrador de conjuntos	galería de objetos 3D, nuevas herramientas para el uso de los sólidos (37).

Anexo 5: Entorno de Desarrollo Integrado

Tabla comparativa. Entorno de Desarrollo Integrado

Nombre / **Indicador**	***Flex Builder* 3**	***Silverlight* 4**	***Flash Builder* 4**
Empresa desarrolladora	*Adobe*	*Microsoft*	*Adobe*
Licencia	MPL (69)	*Software* propietario (70)	*Software* propietario
Plataforma	Multiplataforma	*Windows* y *Mac* OS (16)	*Windows* y *Mac* OS (71)
Características	Visualización interactiva de datos, pre-visualización de valores de CSS, refactorización de código (15).	Herramientas para la visualización de datos, personalización de componentes (16).	Nuevo explorador de datos/servicios, Servicio *Flash* para Plataformas Sociales (17).
Lenguaje de Programación	MXML, *Actionscript* 3.0 (15)	XAML, C#, *Visual Basic*, *Ruby* y *Python* (16).	MXML, *Actionscript* 3.0 (17)
Utilización de *framework*	Sí	Sí	Sí

Frecuencia de actualización	Alta	Media	Alta
Comunidad de desarrollo	Sí	Sí	Sí
Integración con otras tecnologías	CSS, *Adobe Creative Suite* 3, *Adobe* AIR, PHP, ASP.NET, *ColdFusion*, *LiveCycle Data Services* (15)	HTML, XML, *Windows Phone* 7, *Microsoft Office* (16)	*Adobe Flash Professional*, *Illustrator*, *Photoshop* o *Fireworks*, *Adobe Flash Catalyst*, PHP, *Adobe ColdFusion*, REST y SOAP (17)

Anexo 6: Integración de las tecnologías.