Universidad de las Ciencias Informáticas

Vertex, Entornos Interactivos 3D, Facultad 5

Módulo destilación para el Laboratorio Virtual de Química

Trabajo de diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autores: Reynier Palomino Echeandia

Leonar Alemañy Socarrás

Tutores: Dr. Ramón Carrasco Velar

Ing. Jandy Miguel Gómez Rodríguez

VERTEX
ENTORNOS INTERACTIVOS 3D

La Habana, 2015

Agradecimientos

A mi mamá Teresa y papá Eugenio que son lo más grande que tengo en la vida, todo lo que soy de los debo a ellos.

A mi hermano Leoder que siempre ha sido mi ejemplo a seguir y estuvo junto a mí en estos cinco años apoyándome en todo.

A mi novia Dayana porque juntos enfrentamos las dificultades de la universidad, fue muy importante para mi tenerla a mi lado en los cuatro años finales de la universidad, depende mucho de ella que hoy yo esté aquí.

A mis abuelos y tíos que siempre estuvieron al tanto de mi transcurso por la universidad.

A Roxana que aunque no se encuentra aquí, en los primeros años de la universidad me brindo el cariño, los regaños y los consejos de una hermana.

Leonar

A mi mamá Martha por ser la mejor madre del mundo.

A mi papá Edilberto por ser mi ejemplo a seguir en todo.

A mi hermano Reynaldo por siempre apoyarme en todo.

A mis tías Eridania y Elvia por estar ahí siempre que las necesito.

Y a todos los colegas que compartieron conmigo momentos buenos y malos durante estos cinco años de sacrificio y malas noches.

Reynier

Declaración de autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

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

Reynier Palomino Echeandia
Autor

Leonar Alemañy Socarrás
Autor

Dr. Ramón Carrasco Velar
Tutor

Ing. Jandy Miguel Gómez Rodríguez
Tutor

Resumen

El trabajo investigativo que se presenta está dirigido al desarrollo de un módulo para el laboratorio virtual de Química que permita a los estudiantes de la enseñanza media ejercitar los conocimientos teóricos y prácticos sobre destilación adquiridos en clases. Debido a las particularidades del contexto de desarrollo se decidió utilizar *Extreme Programming* como metodología de desarrollo de *software* y como herramienta principal el motor de videojuegos *Unity 3D*, que demostró ser una buena alternativa para la creación de este tipo de soluciones.

Para la realización del módulo se realizó un estudio de una serie de conceptos relacionados con los laboratorios virtuales así como los relacionados con los procesos de destilación en Química. Se analizaron además algunas soluciones existentes en el mundo, las cuales presentan algunas limitaciones para su utilización en nuestras escuelas.

Como resultado se obtuvo un módulo flexible que consta de cuatro prácticas de destilación: destilación simple, destilación fraccionada, destilación al vacío y destilación de arrastre por vapor. En cada una de estas el usuario puede realizar actividades como el montaje del equipo de destilación y un cuestionario de preguntas teóricas relacionado con el tema de cada una de estas actividades.

Palabras clave: *Destilación química, Laboratorio virtual de Química, Unity 3D, Módulo*.

Índice general

Índice de figuras

Índice de tablas

Introducción

La evolución de la ciencia y la tecnología ha contribuido en gran medida al desarrollo de la sociedad actual. Con el surgimiento de los ordenadores o computadoras, se crearon nuevos sistemas capaces de realizar tareas que antes eran muy difíciles o complejas de lograr. El avance y desarrollo en la creación de equipos informáticos y el estudio e investigación de las técnicas de gráficos, dieron surgimiento a la realidad virtual. En la actualidad, la misma se encuentra en muchos sistemas que permiten al usuario interactuar en diferentes escenarios virtuales.

En el campo de la educación, estas se convierten en un recurso didáctico que permite incluso reproducir laboratorios tradicionales en entornos virtuales. Estos pueden ser usados por los profesores para motivar y atraer la atención de los estudiantes en el proceso de enseñanza-aprendizaje. El uso de los gráficos tridimensionales y el alto grado de interactividad que se obtiene, permiten al estudiante incorporar el conocimiento de forma visual. Cada vez es mayor el número de centros de enseñanza en los que se utilizan aplicaciones con entornos virtuales entre los que se encuentran los laboratorios virtuales.

Los laboratorios virtuales apoyan la gestión de asignaturas técnicas y contribuyen a desarrollar habilidades y destrezas prácticas a través de interacciones, comparaciones, demostraciones e interpretaciones en la computadora. Es importante destacar que el desarrollo de este tipo de herramientas no persigue sustituir, ni las experiencias prácticas, ni la presencia del profesor en el aula, sino convertirlas en una herramienta de auto preparación que contribuya a adquirir y desarrollar habilidades que después puedan llevar a la práctica en un laboratorio real.

En nuestro país los laboratorios existentes para la docencia son antiguos y se han deteriorado con el paso de los años, además, la mayoría de ellos no cuentan con los recursos indispensables para su trabajo dado que los medios utilizados en los laboratorios se fabrican fuera del país, lo que provoca que se haga muy difícil su adquisición. Este fenómeno es común a todos los niveles de enseñanza, por lo que es imprescindible, no solamente el cuidado de los medios sino también la búsqueda de nuevas soluciones que permitan al estudiante realizar sus prácticas de laboratorio.

La química es una ciencia experimental, y debido al estado actual de las instalaciones existentes, se hace necesario la búsqueda de alternativas en apoyo a la docencia. Una de estas alternativas es el uso de laboratorios virtuales, los cuales constituyen una posible solución para minimizar ciertos accidentes que pueden ocurrir durante una práctica de laboratorio, lo cual permitiría evitar daños tanto a los medios como a las personas que se encuentren en el laboratorio. Entre los procesos básicos a desarrollar en los laboratorios de química están las diferentes técnicas de destilación que requieren tanto del estudio de la teoría como de

la práctica por parte del estudiante para adquirir estos conocimientos.

Una alternativa para solucionar los problemas mencionados anteriormente, lo constituye el proyecto "Laboratorio Virtual de Química" que se desarrolla en el centro VERTEX de la facultad 5. El mismo no cuenta con un módulo "Destilación" que permita ejercitar los conocimientos relacionados con los procesos de destilación química.

Por ello el problema investigativo de este trabajo es: *¿Cómo apoyar el proceso de enseñanza-aprendizaje de la destilación en la química práctica?*

De esta forma el objetivo del mismo es *desarrollar el módulo "Destilación" para el laboratorio virtual de química, que permita contribuir al proceso de enseñanza-aprendizaje de la destilación química, en los distintos niveles educativos del país*.

Como objeto de estudio el trabajo se enmarca en *los laboratorios virtuales de Química*, y como campo de acción *los laboratorios virtuales de Química para el proceso de enseñanza-aprendizaje de las prácticas de destilación*.

Para cumplir el objetivo de nuestra investigación se proponen las siguientes tareas:

1. Estudio del estado del arte en el desarrollo de laboratorios virtuales de química.
2. Análisis de las metodologías de desarrollo de *software*, herramientas de modelado, lenguajes de programación y entornos de desarrollo para seleccionar los más adecuados a utilizar.
3. Diseño y realización del escenario 3D y de los componentes de los equipos.
4. Implementación del evaluador teórico y práctico.
5. Realización de pruebas de caja negra.
6. Validación del *software* con agentes externos.
7. Confección de un Demo.

Para llevar a cabo las tareas investigativas propuestas se utilizarán los siguientes métodos de investigación:

Métodos teóricos

Análisis Histórico-Lógico: Para profundizar en los antecedentes de la utilización de las Tecnologías de la Información y las Comunicaciones (TIC) en los procesos de enseñanza-aprendizaje y sus tendencias actuales.

Analítico-Sintético: Se utiliza en la revisión bibliográfica, el estudio de reportes e informes sobre el estado de la infraestructura tecnológica en nuestro país, sobre la informatización, la tendencia actual al aprendizaje auto gestionado y la introducción de las TIC en el proceso de enseñanza-aprendizaje.

Métodos empíricos

Entrevistas: Se utilizarán para la recopilación de información especializada o dirigida a directivos, profesores y alumnos que interactúan con las TIC en el proceso de enseñanza-aprendizaje presencial o mixta de la asignatura química.

CAPÍTULO 1

Fundamentación Teórica

En el presente capítulo se analizan algunas definiciones sobre los laboratorios virtuales, así como los diferentes tipos que existen, sus ventajas y desventajas. Además se realizará un análisis del estado actual de los laboratorios virtuales de química. Se presenta un análisis sobre la destilación química en el proceso de enseñanza-aprendizaje. Se hace además un estudio sobre las principales metodologías de desarrollo de *software*, las herramientas y las tecnologías que se utilizan en la actualidad; en dependencia de las características de cada una se selecciona la más adecuada para el desarrollo de la propuesta de solución.

1.1. Laboratorios Virtuales

Las TIC constituyen una herramienta cada vez más poderosa e indispensable en las instituciones educacionales. Pueden emplearse de diversas formas: fuente de información, guía para el proceso de aprendizaje de los estudiantes, medio para ejercitar habilidades y motivar el estudio, entre otras muchas aplicaciones. Es por ello que las TIC han incrementado de modo considerable su presencia como medio de enseñanza a disposición de los docentes y educandos. Un ejemplo de esto son los Laboratorios Virtuales (Marqués, 2001).

Según la bibliografía consultada existen diversas definiciones de lo que es un laboratorio virtual.

Vary (Vary, 2000), expresa que un laboratorio virtual es un espacio electrónico de trabajo concebido para la colaboración y la experimentación a distancia con el objetivo de investigar o realizar otras actividades creativas, y elaborar y difundir resultados mediante tecnologías difundidas de información y comunicación.

Se plantea también que un laboratorio virtual es la representación de un lugar dotado de los medios necesarios para realizar investigaciones, experimentos y trabajos de carácter científico o técnico, producido por un sistema informático, que da la sensación de su existencia real (UNED, 2010).

1.1.1. Ventajas y desventajas

Los laboratorios virtuales presentan algunas ventajas y desventajas como son:

Ventajas

a) Superación de problemas espacio-temporales, en personas y en recursos.

- El laboratorio virtual acerca y facilita a un mayor número de alumnos la realización de experiencias, aunque alumno y laboratorio no coincidan en el espacio (González-Castaño et al., 2001).
- Flexibiliza el horario de prácticas y evita la saturación por el solapamiento con otras asignaturas (Turégano y Cózar, 1994).
- Los laboratorios virtuales reducen el coste del montaje y mantenimiento de los laboratorios. El laboratorio virtual es una alternativa barata y eficiente, donde el estudiante simula los fenómenos a estudiar como si los observase en los laboratorios reales (González-Castaño et al., 2001).

b) Aumento de la interactividad en el aprendizaje de conceptos y técnicas.

- Los estudiantes aprenden mediante prueba y error, sin miedo a sufrir o provocar un accidente, sin avergonzarse de realizar varias veces la misma práctica, ya que pueden repetirlas sin límite; sin temor a dañar alguna herramienta o equipo. Además, pueden asistir al laboratorio cuando ellos quieran, y escoger aquellas áreas del laboratorio que resultan más significativas para realizar prácticas de su trabajo (Bermejo y Saboya, 2004).

Además de estas ventajas, los laboratorios virtuales poseen características peculiares que los convierten en herramientas útiles para la docencia, dadas por una parte por las actitudes positivas que muestran los jóvenes hacia las TIC así como la habilidad que estos poseen en el manejo de simuladores e instrumentos informáticos, que los capacita para desenvolverse rápida y fácilmente en este tipo de entornos tecnológicos.

Desventajas

- El laboratorio virtual no puede sustituir la experiencia práctica altamente enriquecedora del laboratorio tradicional (Gil, Blanco y Aulí, 2000).
- En el laboratorio virtual se corre el riesgo de que el alumno se comporte como un mero espectador (Fernandez et al., 2002). Cada simulación en el laboratorio virtual debe venir acompañada de un guion que explique el concepto a estudiar (Marti, 2000).
- El alumno no utiliza elementos reales en el laboratorio virtual, lo que provoca una pérdida parcial de la visión de la realidad (Boix, Fillet y Bergas, 2002).
- No siempre se dispone de la simulación adecuada para el tema que el profesor desea trabajar (Rosado y Herreros, 2003).

1.1.2. Tipos de laboratorios virtuales

Según la bibliografía consultada, los laboratorios virtuales tienen diferentes clasificaciones atendiendo al contenido y a la forma de utilización. Según este último enfoque se clasifican en:

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

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

1.1.3. Laboratorios virtuales de Química

Los laboratorios virtuales de química son herramientas informáticas que simulan un laboratorio de química desde un entorno virtual de aprendizaje. Estos ofrecen más flexibilidad que un laboratorio real en la enseñanza, pues convierten el trabajo de laboratorio en una opción de aprendizaje donde el alumno puede equivocarse y rectificar con una inversión que no sería posible en un laboratorio real. Asimismo, la computadora permite cambiar la imagen negativa que el alumno suele tener de la Química, y la recibe de una manera más interesante al explorar el ambiente virtual. Por otra parte, los laboratorios virtuales de Química son una alternativa complementaria con múltiples ventajas, entre las que destacan (Cabero, 2007):

- Trabajar en un ambiente de enseñanza e investigación protegido y seguro.
- Realizar con los estudiantes un trabajo tanto individual como grupal y colaborativo.
- Ofrecer a los estudiantes prácticas a las que por su costo no tendrían acceso.
- Poder reproducir los experimentos un número indefinido de veces.
- Extender el concepto de laboratorio al aula de clases a través del uso de una computadora.

En el mundo existen actualmente diversos laboratorios virtuales de química, como por ejemplo:

Crocodile Chemistry: Es un laboratorio virtual de Química en el que se pueden simular experimentos, representar resultados en gráficos y observar reacciones en 3D *(Figura 1.1)*. Cuenta con una librería de objetos que permite seleccionar una serie de recipientes como: matraces, probetas y pipetas entre otros. Permite además modificar parámetros de los diferentes componentes, como por ejemplo el tamaño de las

partículas, la concentración o la tasa de flujo de un gas etc. También se pueden realizar experimentos con ácidos y bases, con metales, mezclas y reacciones, con la tabla periódica, con compuestos no metálicos y experimentos de electroquímica y químico-físicos (Clips, 2009).

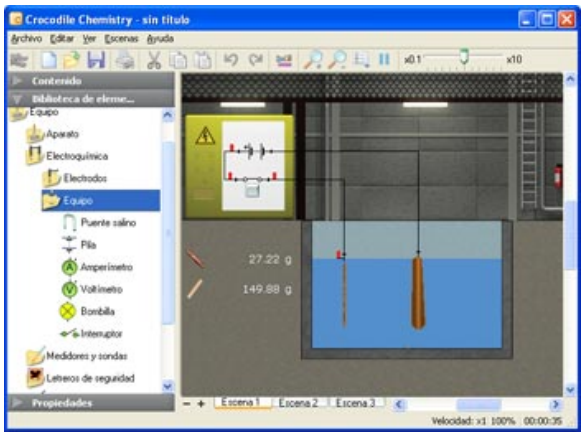

Figura 1.1. Crocodile Chemistry

ChemLab: El programa *Model ChemLab v2.0 (Figura 1.2)* para *Windows*® y *Mac*® *OS (Model Science)* es la simulación interactiva de un laboratorio de Química. En él se usan el equipamiento y los procedimientos comunes de laboratorio para simular los pasos involucrados en la realización de los experimentos. La simulación de cada práctica de laboratorio se halla en un módulo separado y constituye una extensión del programa ChemLab, por lo que es posible la realización de muchas y diferentes prácticas, usando una interfase común de laboratorio (Vidal Castaño y González Medina, 2002).

El ChemLab consta de dos ventanas, una de texto y otra de laboratorio. La ventana de texto sirve para la documentación textual en el ChemLab y está dividida a su vez en 3 ventanas:

- Introducción
- Procedimiento
- Observaciones.

La primera sólo sirve para leer la introducción a la práctica, la segunda permite leer el procedimiento que debe seguir el usuario para realizar el experimento en la ventana de laboratorio, y la tercera está diseñada para que el usuario anote las observaciones que se le indican en la ventana del procedimiento.

La ventana de laboratorio permite visualizar la simulación animada del laboratorio. En ella se agregan los utensilios o equipos a utilizar. Estos objetos y las sustancias que se emplean pueden agregarse y utilizarse usando los comandos del menú principal, la barra de herramientas *(Chem toolbar)* o el menú contextual del botón derecho del ratón (ibíd.).

En el ChemLab el usuario dispone del equipamiento de laboratorio siguiente (ibíd.):

- Vasos de precipitado, frascos erlenmeyer, balones, tubos de ensayo, probetas graduadas, gotero, buretas, pipetas, vidrios reloj, kitasato con embudo buchner, quemador de gas bunsen, plancha de calentamiento con agitador magnético, varas de agitación, plato (cápsula) de evaporación, calorímetro, conductímetro y potenciómetro, entre otros.
- Balanzas: técnica, electrónica y de alta sensibilidad (analítica).
- Equipo de destilación, incluyendo el balón de destilación con manta de calentamiento, cabezal de destilación y el condensador.

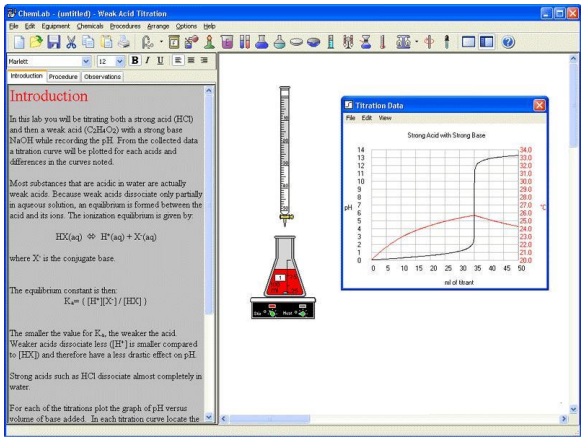

Figura 1.2. ChemLab

VLabQ: Es un simulador interactivo de prácticas de laboratorio de Química *(Figura 1.3)*. Utiliza equipos y procedimientos estándares para simular los procesos que intervienen en un experimento o práctica. Cada simulación o práctica se guarda en un archivo que contiene todos los reactivos y condiciones que se usarán durante el experimento. Se puede guardar en cualquier momento todo el contenido del laboratorio, tanto el equipo como su contenido y las condiciones para continuar con la práctica posteriormente. Una vez cargada una práctica, el simulador muestra diferentes textos que sirven como guía para realizarla (SOFT, 2000).

Contiene instrumentos laboratorio, tales como: vasos de precipitados, matraces erlenmeyer, embudo-buchner, matraz de balón, reactor, buretas, probetas, pipetas y tubo de ensayo entre otros. Brinda además los equipo de medición como pHmetros, termómetros, conductímetros y balanzas. También consta de equipos térmicos como mechero, parrilla y baño de hielo además de otros instrumentos como agitador de vidrio, vidrio de reloj, cápsula de porcelana y calorímetro (ibíd.).

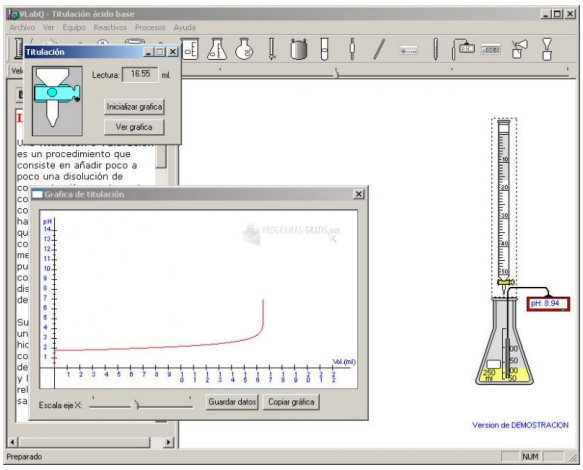

Figura 1.3. VLabQ

QuimiLab: Es un laboratorio virtual diseñado para acompañar al curso de Química del nivel secundario y universitario *(Figura 1.4)*. Este tiene una interfaz basada en arrastrar-y-soltar los elementos y ventanas móviles que simulan productos químicos, vidriería e instrumentos. Brinda una serie de equipamiento agrupados por tipo (LABS, 2010):

- Vidriería: vasos de precipitados, matraces, tubos de ensayo, buretas, agua de grifo y un calorímetro.
- Instrumentación: balanza, plato caliente, medidores de temperatura, presión, de pH y espectrofotómetro.
- Productos químicos: está disponible una base de datos de compuestos químicos. Cada compuesto presenta sus propiedades para todas las fases incluyendo densidades, puntos de ebullición y de fusión y capacidades calóricas.
- Reacciones: la base de datos de reacciones químicas contiene proporciones estequiométricas de los reaccionantes junto con las condiciones experimentales requeridas, datos cinéticos y constantes de equilibrio.

Figura 1.4. QuimiLab

Después de analizar las potencialidades y las características de cada uno estos laboratorios virtuales de química se detectaron una serie de limitaciones como:

- Todos estos laboratorios son privados.
- No cuentan con un evaluador teórico que permita ejercitar la teoría.
- Crocodile Chemistry, VLabQ y QuimiLab no cuentan con prácticas de destilación.
- Tienen un elevado costo en el mercado.
- Requieren de ordenadores con prestaciones mínimas que son superiores a los ordenadores con que contamos en las escuelas.

Además de estas limitantes, todos estos laboratorios están desarrollados en 2D lo que reduce el realismo a las prácticas. Por estas razones se hace necesario la creación de un módulo de destilación para el laboratorio virtual de Química para apoyar al proceso de enseñanza-aprendizaje de la Química en nuestro país.

1.2. La destilación en Química

La Química es una disciplina que forma parte del diseño curricular de las enseñanzas secundaria y preuniversitaria, así como de un gran número de carreras universitarias. Uno de los contenidos más importantes es el correspondiente a la separación de sustancias y en particular la destilación.

La destilación es un proceso que consiste en calentar un líquido hasta que sus componentes más volátiles pasan a la fase de vapor y, a continuación, se enfría el vapor para recuperar dichos componentes en forma líquida por medio de la condensación. El objetivo principal de la destilación es separar una mezcla de varios componentes aprovechando sus distintas volatilidades, o bien separar los materiales volátiles de los no volátiles (Ramírez, 2010).

1.2.1. Tipos de destilación e importancia en los procesos industriales. Ejemplos

Destilación simple a presión atmosférica

Es aquella que se realiza a presión ambiental. Se utiliza fundamentalmente cuando la temperatura del punto de ebullición se encuentra por debajo de la temperatura de descomposición química del producto (ibíd.). Para esta destilación se utiliza el equipo de destilación representado en la *(Figura 1.5)*. Consta de un matraz de destilación, provisto de un termómetro. El matraz descansa sobre una placa calefactora o un mechero. El matraz de destilación va unido a un refrigerante con camisa de refrigeración por la que circula agua a contracorriente. Finalmente, el extremo inferior del refrigerante se une a una alargadera o codillo que conduce el destilado al matraz colector (Avila, 2013).

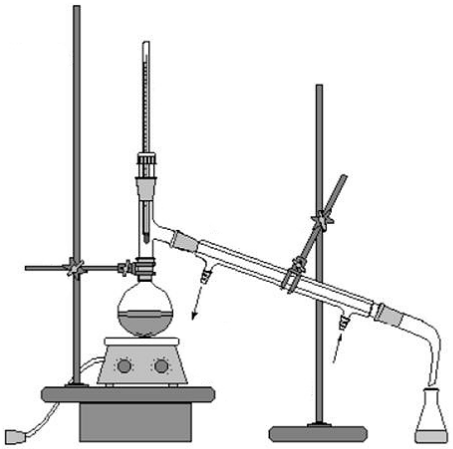

Figura 1.5. Equipo para la destilación simple

Destilación fraccionada

Se utiliza cuando la mezcla de productos líquidos que se pretende destilar contiene sustancias de diferentes puntos de ebullición con una diferencia entre ellos menor a 80 ^{o}C. Al calentar una mezcla de líquidos de diferentes presiones de vapor, el vapor se enriquece en el componente más volátil y esta propiedad se aprovecha para separar los diferentes compuestos líquidos mediante este tipo de destilación. La principal diferencia con el equipo de destilación simple es que, en este caso, entre el matraz esférico o de destilación y la pieza acodada se acopla una columna de fraccionamiento *(Figura 1.6)* (Ramírez, 2010).

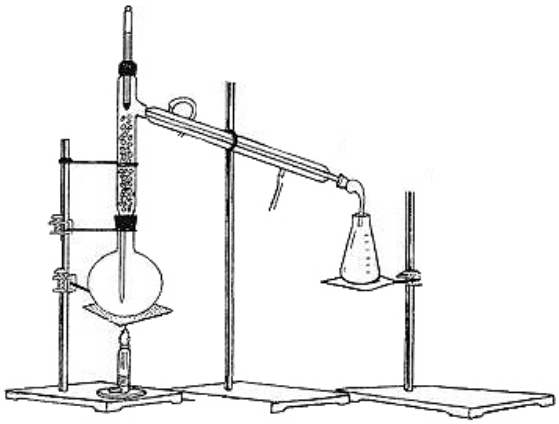

Figura 1.6. Equipo para la destilación fraccionada

Destilación por arrastre de vapor

La destilación por arrastre de vapor *(Figura 1.7)* posibilita la purificación o el aislamiento de compuestos de punto de ebullición elevado mediante una destilación a baja temperatura (siempre inferior a 100 ^{o}C). Es una técnica de destilación muy útil para sustancias de punto de ebullición muy superior a 100 ^{o}C y que se descomponen antes o al alcanzar la temperatura de su punto de ebullición (Ramírez, 2010).

La destilación por arrastre de vapor es una técnica de destilación que permite la separación de sustancias insolubles en H_2O y ligeramente volátiles de otros productos no volátiles. A la mezcla que contiene el producto que se pretende separar, se le adiciona un exceso de agua, y el conjunto se somete a destilación. En el matraz de destilación se recuperan los compuestos no volátiles y/o solubles en agua caliente, y en el matraz colector se obtienen los compuestos volátiles e insolubles en agua. Finalmente, el aislamiento de los compuestos orgánicos recogidos en el matraz colector se realiza mediante una extracción (ibíd.).

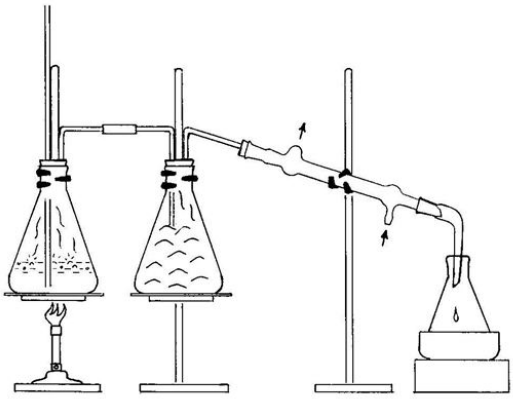

Figura 1.7. Equipo para la destilación por arrastre de vapor

Destilación al vacío

Es una forma de destilación (sencilla o fraccionada) que se efectúa a presión reducida. El montaje es muy parecido a los otros procesos de destilación, con la salvedad de que el conjunto se conecta a una bomba de vacío o trompa de agua *(Figura 1.8)*, lo cual permite destilar líquidos a temperaturas inferiores a su punto de ebullición normal. Muchas sustancias no pueden purificarse por destilación a presión atmosférica porque se descomponen antes de alcanzar sus puntos de ebullición normales. Otras sustancias tienen puntos de ebullición tan altos que su destilación es difícil o no resulta conveniente. En estos casos se emplea la destilación a presión reducida. Un líquido comienza a hervir a la temperatura en que su tensión de vapor se hace igual a la presión exterior, por tanto, disminuyendo esta se logrará que el líquido destile a una temperatura inferior a su punto de ebullición normal (ibíd.).

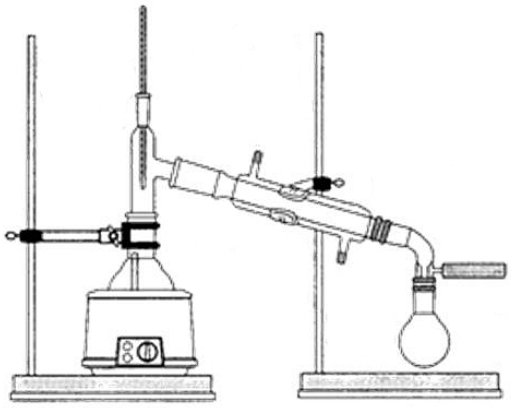

Figura 1.8. Equipo para la destilación al vacío

La destilación constituye una de las principales técnicas de laboratorio para purificar líquidos volátiles. Esta se utiliza en la refinación del petróleo, en procesos de obtención de productos petroquímicos de todo tipo y en otras aplicaciones industriales como por ejemplo, en las industrias farmacéutica, química, alimentaria entre otras (GEA, 2009).

1.2.2. Las prácticas de destilación en la enseñanza de la química

La realización de las prácticas de destilación en la enseñanza de la química posibilita a los estudiantes interactuar con este proceso de una manera más directa, permitiéndole disfrutar de una enseñanza más activa, participativa e individualizada, donde se impulse el método científico y el espíritu crítico (Olivera, 2012).

No obstante, las dificultades reales existentes en el escenario nacional, planteadas en epígrafes anteriores, para la realización de cuatro de los ejercicios académicos fundamentales de destilación (simple, fraccionada, al vacío y arrastre con vapor), sugieren que es posible desde el campo de la informática, contribuir a la enseñanza de los mismos, si se le facilita al estudiante la posibilidad de interactuar en la computadora con el instrumental y aparatura química necesarios para la ejecución de cada una de las tareas.

De este modo se favorece que el alumno: desarrolle habilidades, aprenda técnicas elementales y se familiarice con el manejo de instrumentos y aparatos. La realización de estas prácticas permite poner en crisis el pensamiento espontáneo del alumno, al aumentar la motivación y la comprensión respecto de los conceptos y procedimientos científicos (Olivera, 2012).

1.3. Metodologías de desarrollo

Una metodología es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a realizar un *software*. Para la misma se definen tareas que constituyen las actividades elementales en que se dividen los procesos y permiten ser ejecutadas a través de los procedimientos. En este sentido se definen técnicas que se pueden utilizar una o varias veces y herramientas de *software* que automatizan la aplicación. En el proceso de desarrollo de *software* las metodologías se separan en dos grupos, las tradicionales o pesadas donde se encuentra *Rational Unified Process (RUP)* y las metodologías ágiles en el cual se pueden mencionar Extreme Programming (XP) y SCRUM.

Estas se diferencian fundamentalmente en que las tradicionales intentan lograr el objetivo común por medio de orden y documentación, y las ágiles tratan de mejorar la calidad del *software* a través de una comunicación directa e inmediata entre las personas que intervienen en el proceso (Patón, 2012).

Debido a que el proyecto no se considera de gran envergadura, el equipo de desarrollo es pequeño y no se requiere la generación de gran cantidad de artefactos; se decidió seleccionar una metodología ágil y no una metodología tradicional. Cada una de las metodologías ágiles tiene características propias y hace énfasis en algunos aspectos más específicos. A continuación se hace un estudio de las principales metodologías ágiles.

1.3.1. SCRUM

La metodología SCRUM define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años. Es una metodología que no se fundamenta en el seguimiento de un plan, sino en la adaptación continua de las circunstancias de la evolución del proyecto. Su modo de desarrollo es adaptable antes de predictivo, orientado a las personas más que a los procesos y emplea un modelo de desarrollo incremental basado en iteraciones y revisiones, cada iteración es denominada *sprint* (Palacio y Ruata, 2011).

Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos, el ciclo de vida está dividido en tres fases: planificación del *sprint*, seguimiento del *sprint* y revisión del *sprint*. El resultado de cada *sprint* es un incremento ejecutable que se muestra al cliente. La segunda característica importante se refiere a las reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración (Letelier, 2006).

1.3.2. Crystal Methodoligies

Se trata de un conjunto de metodologías para el desarrollo de *software* caracterizadas por estar centradas en las personas que componen el equipo y la reducción al máximo del número de artefactos producidos. Han sido desarrolladas por *Alistair Cockburn*. El desarrollo de *software* se considera un juego cooperativo de invención y comunicación, limitado por los recursos a utilizar. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener políticas de trabajo en equipo definidas. Estas políticas dependerán del tamaño del equipo, estableciéndose una clasificación por colores, por ejemplo *Crystal Clear* (3 a 8 miembros) y *Crystal Orange* (25 a 50 miembros) (ibíd.).

1.3.3. XP

La programación extrema es una metodología de desarrollo basada en una serie de valores y de buenas prácticas de manera que persigue el objetivo de aumentar la productividad a la hora de desarrollar programas. Una de las características principales de este método de programación, es que sus ingredientes son conocidos desde el principio de la informática. Los autores de XP han seleccionado aquellos que han considerado mejores y han profundizado en sus relaciones y en cómo se refuerzan los unos con los otros. El resultado de esta selección ha sido esta metodología única y compacta. Por esto, aunque no está basada en principios nuevos, sí que el resultado es una nueva manera de ver el desarrollo de *software* (Joskowicz, 2008).

A continuación se presentan algunas características de XP que se adaptan a las necesidades de un proyecto, así como a las condiciones de trabajo (ibíd.):

- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
- Programación por parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera es revisado y discutido mientras se escribe.
- Frecuente interacción del equipo de programación con el cliente o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores sean detectados.

- Simplicidad en el código: la programación extrema apuesta que es más sencillo hacer un código simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar funciones complicadas y quizás nunca utilizarlas.

No existe una metodología universal para hacer frente con éxito a cualquier proyecto de desarrollo de *software*. Toda metodología debe ser adaptada al contexto del proyecto (recursos técnicos y humanos, tiempo de desarrollo, tipo de sistema, entre otros).

Después de un estudio de las principales tendencias en metodologías ágiles de desarrollo se decidió escoger la metodología XP para la realización del módulo de destilación para el laboratorio virtual de Química, ya que muchas de sus características son aplicables al contexto de realización del proyecto. Principalmente porque es ideal para equipos pequeños (en este caso, dos personas, trabajando en una sola máquina), la simplicidad en el código y el cliente forma parte del equipo de desarrollo. Se diseñan e implementan las pruebas antes de programar la funcionalidad y cada integrante tiene autoridad para cambiar cualquier parte del código fuente. Es una metodología que propone integraciones continuas y está diseñada para adaptarse a los cambios de requisitos surgidos una vez empezado el proyecto.

1.4. Herramientas de diseño

Blender

Blender 3D es una poderosa herramienta para modelado 3D digital, pero que tiene un modo de funcionamiento y una interfaz que son distintas a las de otros programas 3D. Es un paquete de modelado y animación en 3D que se actualiza constantemente y cuenta con un motor de juegos que lo hace diferente de los demás paquetes de modelado y animación en 3D. La principal ventaja de esta herramienta es que se trata de *software* libre y además gratuito, condicionando esto, que se pueda descargar e instalar sin ninguna restricción y costo (Moya, 2012).

Partiendo de lo anteriormente planteado se puede decir que *Blender* es un programa de modelado y animación con las siguientes capacidades: animación 3D, creación de videojuegos 3D en tiempo real, generación de ambientes y paisajes 3D, generación de efectos especiales 3D, sombreado 3D, texturizado interactivo 3D, modelado 3D y simulación física.

Autodesk 3dsMax

Potente herramienta de modelado, animación, renderización y composición en 3D que multiplica rápidamente la productividad de los artistas y diseñadores. Es uno de los programas de animación 3D más utilizado, especialmente para la creación de videojuegos, anuncios de televisión, en arquitectura o en películas, además dispone de una arquitectura de *plugins* y una sólida capacidad de edición que permite una mejor interactividad.

El premiado *software Autodesk 3ds Max* es la herramienta preferida por los profesionales líderes en desarrollo de juegos, televisión, cine y composición digital que quieren una solución 3D completa con resultados inmediatos. Además proporciona múltiples opciones de renderización integradas, incluida renderización en red ilimitada con tecnología mental *ray*.

Entre sus características más significativas podemos mencionar las siguientes (Autodesk, 2012):

- Cuenta con más de 100 herramientas de modelado, escultura y manipulación de objetos.
- Existen infinidad de *plugins* que facilitan la creación de proyectos.
- Compatible con multitud de formatos.
- Permite la integración con *Adobe Photoshop*. Se pueden importar los archivos PSD como texturas.

Tanto *Autodesk 3ds Studio Max* como *Blender 3D* son herramientas de diseño muy potentes con las cuales se pueden obtener resultados muy satisfactorios. Partiendo de que la interfaz de *Blender 3D* no es muy intuitiva y que el equipo de desarrollo no tiene ningún conocimiento de la misma, se decide utilizar la herramienta *Autodesk 3ds Studio Max*, tomando en consideración que el equipo de desarrollo posee un conocimiento previo de la misma.

1.5. Herramientas de desarrollo

Ogre3D

Ogre3D (*Object-Oriented Graphic Engine*) es un motor de renderizado de propósito general distribuido bajo licencia LGPL. *Ogre3D* no es un motor de juego. Esto implica que será el desarrollador quien tenga que encargarse de aspectos como la gestión de eventos de entrada (teclado, ratón), físicas, red, interfaces entre otros. En el caso del desarrollo de interfaces existen maneras de crearlas con *Ogre* a través del uso de *overlays*; sin embargo, esta aproximación no es lo suficientemente flexible como para crear interfaces avanzadas.

Las características principales de *Ogre* son (Tirado Granados, 2010):

- Mutiplataforma: permite el desarrollo para sistemas *Windows*, *GNU/Linux* y *Mac OS X*.
- Diseño a alto nivel: *Ogre3D* encapsula llamadas a las librerías gráficas *DirectX* y *OpenGL*. Además, hace uso de patrones de diseño: *observer* para informar de eventos y cambios de estado, *singleton* para evitar que exista más de una instancia de cualquier *manager*, *visitor* para realizar operaciones sobre un objeto y evitar modificarlo (por ejemplo, en los nodos del grafo de escena), *facade* para unificar el acceso a operaciones y *factory* para creación de objetos concretos de interfaces abstractas.
- Grafo de escena: una de las características más importantes del grafo de escena de *Ogre* es que desacopla el propio grafo del contenido de la escena, definiendo una arquitectura de *plugins*.

- Aceleración *Hardware*: *Ogre* permite definir el comportamiento de la parte programable de la GPU mediante la definición de Shaders, estando al mismo nivel de otros motores como *Unreal* o *CryEngine*.
- Materiales: se definen mediante un sistema de *scripts* y permiten asignar o cambiar los materiales de los elementos de la escena sin modificar el código fuente.
- Animación: tres tipos(*skeletal*, *morph* y *pose*). La animación y la geometría asociada a los modelos se almacena en un único formato binario optimizado. El proceso más empleado se basa en la exportación desde la aplicación de modelado y animación 3D a un formato XML *(Ogre XML)* para convertirlo posteriormente al formato binario optimizado mediante la herramienta de línea de órdenes *OgreXML-Converter*.

Unity3D

Unity 3D es un motor de videojuegos 3D para PC (*Personal Computer* o Computadora Personal) y *Mac* que viene empaquetado como una herramienta para crear juegos, aplicaciones interactivas, visualizaciones y animaciones en 3D y en tiempo real. *Unity* puede implementar contenido para múltiples plataformas como PC, *Mac*, *Nintendo Wii* y *iPhone*. El motor también puede publicar juegos basados en *web* usando el *plugin Unity Web Player* (Technologies, 2011).

El contenido del juego es construido desde el editor del programa usando lenguajes de guion (*scripts*). Esto significa que los desarrolladores no necesitan ser unos expertos en C++ para crear videojuegos con *Unity*, ya que las mecánicas de juego son compiladas usando una versión de *JavaScript*, C# o *Boo* (U. Technologies, 2010).

Unity 3D integra *MonoDevelop* como entorno de desarrollo ya que es libre y gratuito y este fue diseñado primordialmente para C# y otros lenguajes *.NET*. Además este es el entorno de desarrollo incluido por defecto en Unity 3D.

Características de *MonoDevelop* que potencian su uso (Mono, 2013):

- Ambiente sumamente amigable y simple.
- La ayuda es muy completa e incluye ejemplos de casi todo.
- Posee autocompletado de sintaxis.
- Posee un navegador incorporado.
- Es multiplataforma.

Estos motores gráficos descritos anteriormente son utilizados en nuestro centro para el desarrollo de laboratorios virtuales. Ambos tienen características que facilitan en trabajo a los desarrolladores, pero como el laboratorio virtual de Química se desarrolla sobre *Unity 3D*, el módulo se realizará sobre este motor y de esta manera una vez terminado podrá ser integrado con el laboratorio sin presentar problemas de compatibilidad.

1.6. Lenguajes de programación

El motor de juego *Unity3D* brinda a los programadores la posibilidad de contar con tres lenguajes de programación para el desarrollo de aplicaciones. Estos lenguajes son (Unity, 2011):

- **C# (*C Sharp*)**: Está basado en la plataforma *.NET* la cual puede integrarse con *MonoDevelop*. Este lenguaje es una evolución de los lenguajes C y C++. Utiliza muchas de las características de C++ en las áreas de instrucciones, expresiones y operadores.
- ***Javascript***: Es un lenguaje personalizado inspirado en la sintaxis *ECMAScript*.
- ***Boo***: Es un lenguaje de programación orientado a objetos. Este tiene una sintaxis inspirada en *Python*.

Los lenguajes de programación mencionados anteriormente brindan al programador potencialidades similares al trabajar con *Unity3D*, es decir no importa cual se elija, cualquiera servirá para exportar aplicación a las plataformas seleccionadas (ibíd.).

Para desarrollar la aplicación se seleccionó el lenguaje de programación C#, ya que el equipo de desarrollo cuenta con experiencia desarrollando en este lenguaje y tiene un mayor dominio de este que del resto.

1.7. Conclusiones parciales

En el presente capítulo se realizó un estudio de los principales conceptos relacionados con los laboratorios virtuales de Química y los procesos de destilación química. Por otra parte se analizaron algunas de las soluciones existentes, las cuales presentan algunas limitaciones para su utilización. Además se seleccionaron las herramientas, metodología y lenguaje de programación a utilizar en el desarrollo de la solución.

CAPÍTULO 2

Desarrollo de la solución

En el presente capítulo se describe la solución propuesta al problema de investigación. Se exponen los requisitos funcionales que debe cumplir la aplicación, además se realiza una descripción acerca de la estructura y funcionamiento del prototipo elaborado.

2.1. Propuesta de solución

Para el cumplimiento del objetivo del trabajo se propone la creación de un módulo que permita la realización de actividades de destilación. El mismo estará conformado por cuatro actividades de destilación:

- Destilación Simple
- Destilación Fraccionada
- Destilación al Vacío
- Destilación de Arrastre por Vapor

Cada una de las actividades mencionados anteriormente debe permitir al usuario la realización de un ejercicio de tipo práctico y uno de tipo teórico *(Figura 2.1)*. El ejercicio teórico consiste en un cuestionario conformado por preguntas de selección y preguntas de verdadero o falso relacionadas con el tema de la actividad en la que se esté trabajando, mientras que el ejercicio práctico consiste en la selección y ensamblaje de los elementos que componen el equipo de destilación de la actividad que se esté desarrollando.

En el ejercicio práctico, al terminar de manera satisfactoria el ensamblaje del equipo de destilación, se le brinda la posibilidad al usuario de ver una simulación del proceso de destilación correspondiente a la actividad que esté desarrollando. Además, al terminar tanto el ejercicio teórico como el práctico, el usuario recibe una calificación en base a cinco puntos de acuerdo al desempeño del mismo durante cada uno de los ejercicios. Es importante mencionar que el usuario tendrá la libertad de realizar cada uno de los ejercicios sin tener en cuenta un orden específico.

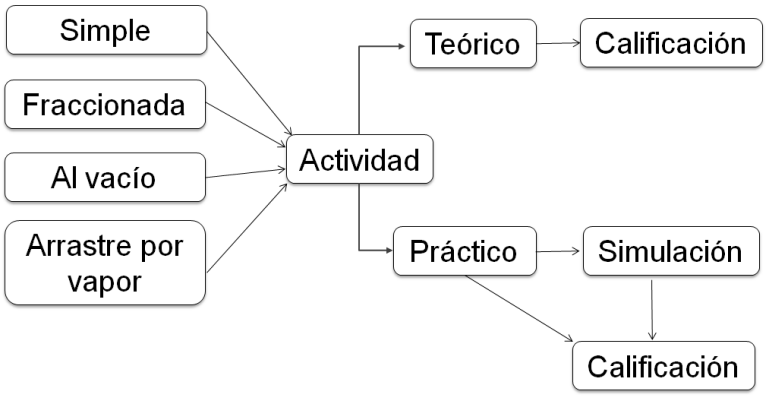

Figura 2.1. Propuesta de solución

2.2. Fase de exploración

En esta fase los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Asimismo, el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizan en el proyecto. Además, se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología (Letelier, 2006).

2.2.1. Historias de Usuario

La gestión de los requisitos funcionales del sistema se hace mediante la definición de las historias de usuarios, que permiten describir y detallar de manera sencilla y sin necesidad del uso de terminología técnica.

Tabla 2.1. Historia de usuario # 1

Historia de usuario	
Número: 1	**Nombre:** Seleccionar actividad
Usuario: Programador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 1.0	**Iteración asignada:** 1
Programador responsable: Reynier Palomino Echeandia	
Descripción: El sistema debe mostrar un menú con las actividades que el usuario puede seleccionar. Una vez seleccionada una actividad, se debe mostrar un nuevo menú donde usuario pueda seleccionar el tipo de ejercicio que desea realizar. Los ejercicios pueden ser de tipo teórico o práctico.	
Observaciones:	

Tabla 2.2. Historia de usuario # 2

Historia de usuario	
Número: 2	**Nombre:** Cargar ejercicio seleccionado
Usuario: Programador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.5	**Iteración asignada:** 1
Programador responsable: Leonar Alemañy Socarrás	
Descripción: El sistema debe permitir cargar el escenario con todos los elementos necesarios para la realización de un ejercicio seleccionado por el usuario a través del menú de ejercicios.	
Observaciones:	

Tabla 2.3. Historia de usuario # 3

Historia de usuario	
Número: 3	**Nombre:** Retornar al menú principal
Usuario: Programador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 1.0	**Iteración asignada:** 1
Programador responsable: Reynier Palomino Echeandia	
Descripción: El sistema debe permitir al usuario regresar al menú inicial desde el ejercicio en el que se encuentre trabajando.	
Observaciones:	

Tabla 2.4. Historia de usuario # 4

Historia de usuario	
Número: 4	**Nombre:** Navegar por la escena
Usuario: Programador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.5	**Iteración asignada:** 1
Programador responsable: Leonar Alemañy Socarrás	
Descripción: El sistema debe permitir que el usuario pueda desplazarse por el escenario del ejercicio práctico que esté desarrollando.	
Observaciones: El usuario puede colisionar con los objetos de la escena, brindando una mayor sensación de realismo.	

Tabla 2.5. Historia de usuario # 5

Historia de usuario	
Número: 5	**Nombre:** Cargar fichero de preguntas teóricas

Continúa en la próxima página

Tabla 2.5. Continuación de la página anterior

Usuario: Programador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 2.0	**Iteración asignada:** 2
Programador responsable: Reynier Palomino Echeandia	
Descripción: El sistema debe ser capaz de acceder a las preguntas teóricas que se encuentran en un fichero *XML* para luego almacenarlas en una estructura de datos.	
Observaciones: Este fichero contiene las preguntas teóricas del ejercicio teórico.	

Tabla 2.6. Historia de usuario # 6

Historia de usuario	
Número: 6	**Nombre:** Aplicar cuestionario teórico
Usuario: Programador	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.8	**Iteración asignada:** 2
Programador responsable: Reynier Palomino Echeandia	
Descripción: El sistema debe mostrar al usuario una interfaz con un cuestionario conformado por preguntas de selección y preguntas de verdadero o falso. Las preguntas son las obtenidas del archivo *preguntas.xml*.	
Observaciones:	

Tabla 2.7. Historia de usuario # 7

Historia de usuario	
Número: 7	**Nombre:** Emitir una nota sobre la actividad teórica
Usuario: Programador	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.2	**Iteración asignada:** 2
Programador responsable: Reynier Palomino Echeandia	
Descripción: El sistema debe ser capaz de verificar si las respuestas del usuario son correctas, en cuyo caso, debe sumar un punto a la calificación y en caso contrario se le resta un punto a la misma. Una vez obtenida la calificación final debe mostrar un panel con los resultados.	
Observaciones: La calificación que se obtiene es en base a cinco puntos.	

Tabla 2.8. Historia de usuario # 8

Historia de usuario	
Número: 8	**Nombre:** Seleccionar instrumental en la escena
Usuario: Programador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto

Continúa en la próxima página

Tabla 2.8. Continuación de la página anterior

Puntos estimados: 2	**Iteración asignada:** 3
Programador responsable: Reynier Palomino Echeandia	
Descripción: El sistema debe permitir al usuario mover los instrumentos hacia la mesa de ensamblaje o dejarlo sobre la mesa en que se encuentran.	
Observaciones: Si el usuario mueve un instrumento que se utiliza en el proceso de destilación en el que se encuentra trabajando, hacia la mesa de ensamblaje, se adiciona un punto a la calificación de la selección. En caso contrario se le resta un punto a la misma.	

Tabla 2.9. Historia de usuario # 9

Historia de usuario	
Número: 9	**Nombre:** Posicionar instrumental
Usuario: Programador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 1.0	**Iteración asignada:** 3
Programador responsable: Leonar Alemañy Socarrás	
Descripción: El sistema debe permitir al usuario suspender el instrumento seleccionado sobre la mesa de trabajo y moverlo hacia posición de ensamblaje, o dejarlo sobre la mesa en que se encontraba.	
Observaciones:	

Tabla 2.10. Historia de usuario # 10

Historia de usuario	
Número: 10	**Nombre:** Ensamblar los elementos seleccionados
Usuario: Programador	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alto
Puntos estimados: 3.0	**Iteración asignada:** 4
Programador responsable: Leonar Alemañy Socarrás	
Descripción: El sistema debe permitirle al usuario colocar de manera permanente el instrumento suspendido si la posición es la correcta. En caso contrario, se debe mostrar un mensaje alertando al usuario que la posición en la que lo desea colocar no es la correcta.	
Observaciones: Si el usuario coloca el instrumento en la posición correcta que ocupa en el equipo de destilación, el elemento se queda fijo en esa posición y no se permite seleccionarlo nuevamente. Además, si fue colocado en la posición correcta, se adiciona un punto a la calificación del ensamblaje y si se intenta colocar en una posición incorrecta, se resta un punto a dicha calificación.	

Tabla 2.11. Historia de usuario # 11

Historia de usuario	
Número: 11	**Nombre:** Emitir nota sobre la actividad práctica
Usuario: Programador	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.5	**Iteración asignada:** 4
Programador responsable: Leonar Alemañy Socarrás	
Descripción: El sistema debe ser capaz de mostrar en un panel los valores de las variables que representan las calificaciones de la selección y el ensamblaje, además de una nota en base a cinco puntos, que promedie las calificaciones de selección y ensamblaje.	
Observaciones:	

2.2.2. Lista de reserva del producto

Los requerimientos no funcionales son las características que hacen al producto atractivo, usable, rápido y confiable. Aunque no son parte de la razón fundamental del producto, son necesarios para hacer que el *software* funcione de la manera deseada. También se deben tener en cuenta a la hora de diseñar e implementar el sistema, ya que tienen repercusión directa en la calidad de la herramienta.

- **Usabilidad:** El módulo implementado podrá ser utilizado por cualquier desarrollador que trabaje sobre el motor de videojuegos *Unity 3D*.
- **Soporte:** El sistema es compatible en su versión inicial con *Windows XP* o versiones superiores, pero puede ser migrado a *Mac OS* y *GNU\Linux*.
- **Hardware:** Es necesario para el funcionamiento del sistema como mínimo, un microprocesador *Intel Pentium IV* a 2,6 GHz, 2 GB RAM, 32 bits de profundidad de color y 128 MB de memoria de video integrado o 64 MB de video dedicado, compatible con *Open GL* o *DirectX 3D*.
- **Portabilidad:** El módulo puede ser trasladado de un proyecto a otro incluso a máquinas diferentes sin que se altere su funcionamiento.

2.3. Fase de Planificación

En esta fase el cliente establece la prioridad de cada historia de usuario. Correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Además, se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días.

El equipo de desarrollo mantiene un registro de la velocidad de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración (Letelier, 2006).

2.3.1. Estimación del esfuerzo por historia de usuario

Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos (Letelier, 2006).

Tabla 2.12. Estimación de esfuerzo por historia de usuario

Iteración	Historias de usuario		Puntos estimados (semanas)
1	1	Seleccionar actividad	1.0
	2	Cargar ejercicio seleccionado	0.5
	3	Retornar al menú principal	1.0
	4	Navegar por la escena	0.5
2	5	Cargar fichero de preguntas teóricas	2.0
	6	Aplicar cuestionario teórico	0.8
	7	Emitir una nota sobre la actividad teórica	0.2
3	8	Seleccionar instrumental en la escena	2.0
	9	Posicionar instrumental	1.0
4	10	Ensamblar los elementos seleccionados	2.0
	11	Emitir nota sobre la actividad práctica	0.5
Total			**11.5**

2.3.2. Duración de las iteraciones y plan de entrega

Las iteraciones deben tener un tiempo de duración de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no siempre es posible ya que es el cliente quien decide qué historias se implementarán en cada iteración (ibíd.).

Tabla 2.13. Plan de duración de las iteraciones

Iteración	Historias de usuario		Duración (semanas)
1	1	Seleccionar actividad	3.0
	2	Cargar ejercicio seleccionado	
	3	Retornar al menú principal	
	4	Navegar por la escena	
2	5	Cargar fichero de preguntas teóricas	3.0
	6	Aplicar cuestionario teórico	
	7	Emitir una nota sobre la actividad teórica	
3	8	Seleccionar instrumental en la escena	3.0
	9	Posicionar instrumental	

Continúa en la próxima página

Tabla 2.13. Continuación de la página anterior

4	10	Ensamblar los elementos seleccionados	2.5
	11	Emitir nota sobre la actividad práctica	
Total			**11.5**

Plan de entrega

A partir del plan de iteraciones anterior *(Tabla 2.13)* se realiza el plan de entregas, que tiene como objetivo definir las entregas que se deben realizar y el orden de las mismas.

Tabla 2.14. Plan de entrega

Entrega	**Iteración 1 (2da semana de marzo)**	**Iteración 2 (1ra semana de abril)**	**Iteración 3 (4ta semana de abril)**	**Iteración 4 (3ra semana de mayo)**
Entrega 1	versión 1	versión 2	versión 3	versión 4(Final)
Entrega 2	-	versión 1	versión 2	versión 3(Final)
Entrega 3	-	-	versión 1	versión 2(Final)
Entrega 4	-	-	-	versión 1(Final)

2.3.3. Desarrollo de iteraciones

- **Iteración 1:** Esta iteración tiene como objetivo la implementación de las historias de usuario 1, 2, 3 y 4 que representan la base de la estructura del negocio. Las mismas son críticas para el proyecto, pues las demás funcionalidades dependen de estas.
- **Iteración 2:** Esta iteración tiene como objetivo la implementación de las historias de usuario 5, 6 y 7 correspondientes al evaluador teórico.
- **Iteración 3:** Esta iteración tiene como objetivo la implementación de las historias de usuario 8 y 9 correspondientes a la primera parte de la actividad práctica.
- **Iteración 4:** Esta iteración tiene como objetivo la implementación de las historias de usuario 10, 11 y 12 correspondientes a la segunda parte de la actividad práctica.

Tareas de desarrollo

Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores (ibíd.).

Tabla 2.15. Tarea de desarrollo # 1

Tarea	
Número de tarea: 1	**Número de historia de usuario:** 1
Nombre de la tarea: Seleccionar actividad	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.0
Fecha de inicio: 2 de febrero de 2015	**Fecha de fin:** 7 de febrero de 2015
Programador responsable: Reynier Palomino Echeandia	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite que el usuario navegando por diferentes menús puedan seleccionar una actividad.	

Tabla 2.16. Tarea de desarrollo # 2

Tarea	
Número de tarea: 2	**Número de historia de usuario:** 2
Nombre de la tarea: Cargar actividad práctica seleccionada	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 9 de febrero de 2015	**Fecha de fin:** 10 de febrero de 2015
Programador responsable: Leonar Alemañy Sacarrás	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite que una vez seleccionada una actividad se cargue a través de la función LoadLevel("level1") el escenario 3D en el que se desarrolla dicha actividad.	

Tabla 2.17. Tarea de desarrollo # 3

Tarea	
Número de tarea: 3	**Número de historia de usuario:** 2
Nombre de la tarea: Cargar actividad teórica seleccionada	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 11 de febrero de 2015	**Fecha de fin:** 12 de febrero de 2015
Programador responsable: Leonar Alemañy Sacarrás	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite que una vez seleccionada una actividad se cargue el cuestionario de preguntas.	

Tabla 2.18. Tarea de desarrollo # 4

Tarea	
Número de tarea: 4	**Número de historia de usuario:** 3
Nombre de la tarea: Retornar al menú principal	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.0
Fecha de inicio: 13 de febrero de 2015	**Fecha de fin:** 19 de febrero de 2015

Continúa en la próxima página

Tabla 2.18. Continuación de la página anterior

Programador responsable: Reynier Palomino Echeandia
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite que una vez en el escenario 3D en el que se desarrolla la actividad poder volver a la ventana de selección de las actividades.

Tabla 2.19. Tarea de desarrollo # 5

Tarea	
Número de tarea: 5	**Número de historia de usuario:** 4
Nombre de la tarea: Navegar por la escena	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 23 de febrero de 2015	**Fecha de fin:** 25 de febrero de 2015
Programador responsable: Leonar Alemañy Sacarrás	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite que el usuario pueda desplazarse por el escenario 3D donde se desarrollan las actividades.	

Tabla 2.20. Tarea de desarrollo # 6

Tarea	
Número de tarea: 6	**Número de historia de usuario:** 5
Nombre de la tarea: Cargar fichero de preguntas teóricas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.0
Fecha de inicio: 26 de febrero de 2015	**Fecha de fin:** 12 de marzo de 2015
Programador responsable: Reynier Palomino Echeandia	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite obtener las preguntas que están contenidas en un fichero xml.	

Tabla 2.21. Tarea de desarrollo # 7

Tarea	
Número de tarea: 7	**Número de historia de usuario:** 6
Nombre de la tarea: Aplicar cuestionario teórico	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.8
Fecha de inicio: 13 de marzo de 2015	**Fecha de fin:** 18 de marzo de 2015
Programador responsable: Reynier Palomino Echeandia	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite crear y aplicar al usuario un cuestionario con las preguntas obtenidas del xml.	

Tabla 2.22. Tarea de desarrollo # 8

Tarea	
Número de tarea: 8	**Número de historia de usuario:** 7
Nombre de la tarea: Emitir una nota parcial sobre la actividad teórica	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.2
Fecha de inicio: 19 de marzo de 2015	**Fecha de fin:** 21 de marzo de 2015
Programador responsable: Reynier Palomino Echeandia	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite mostrar una gráfica con los resutados obtenidos por el usuario en el cuestionario.	

Tabla 2.23. Tarea de desarrollo # 9

Tarea	
Número de tarea: 9	**Número de historia de usuario:** 8
Nombre de la tarea: Seleccionar instrumental en la escena	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.0
Fecha de inicio: 23 de marzo de 2015	**Fecha de fin:** 5 de abril de 2015
Programador responsable: Reynier Palomino Echeandia	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite que el objeto seleccionado por el usuario se coloque en el área de trabajo.	

Tabla 2.24. Tarea de desarrollo # 10

Tarea	
Número de tarea: 10	**Número de historia de usuario:** 9
Nombre de la tarea: Posicionar instrumental	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.0
Fecha de inicio: 13 de abril de 2015	**Fecha de fin:** 18 de abril de 2015
Programador responsable: Leonar Alemañy Socarrás	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite suspender en el aire el objeto seleccionado por el usuario para que posteriomente pueda ser movido.	

Tabla 2.25. Tarea de desarrollo # 11

Tarea	
Número de tarea: 11	**Número de historia de usuario:** 10
Nombre de la tarea: Ensamblar los elementos seleccionados	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.0
Fecha de inicio: 20 de abril de 2015	**Fecha de fin:** 3 de mayo de 2015
Programador responsable: Leonar Alemañy Socarrás	

Continúa en la próxima página

Tabla 2.25. Continuación de la página anterior

Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite colocar los objetos en las posiciones para lograr el ensamblaje de los equipos de destilación.

Tabla 2.26. Tarea de desarrollo # 12

Tarea	
Número de tarea: 12	**Número de historia de usuario:** 11
Nombre de la tarea: Emitir nota parcial sobre la actividad práctica	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 4 de mayo de 2015	**Fecha de fin:** 6 de mayo de 2015
Programador responsable: Leonar Alemañy Socarrás	
Descripción: Con el desarrollo de esta tarea se realiza la funcionalidad que permite mostrar una gráfica con los resutados obtenidos por el usuario en la práctica de laboratorio.	

2.4. Fase de diseño

2.4.1. Arquitectura

La arquitectura de *software* puede ser vista como la estructura del sistema en función de la definición de los componentes y sus interacciones (Astudillo, 2014).

El módulo se desarrolla sobre el motor gráfico de *Unity* 3D y posee una arquitectura en 3 capas que se describe a continuación:

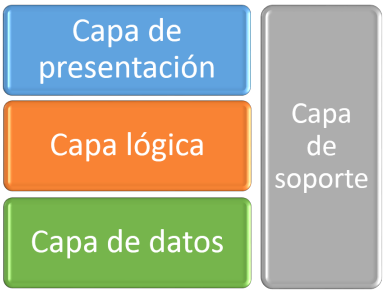

Figura 2.2. Arquitectura

- **Capa de Presentación:** Esta contiene todos los elementos visuales que le permite al usuario interactuar con el sistema y se relaciona directamente con la capa lógica y la capa de soporte. En esta capa se utilizan las siguientes clases.

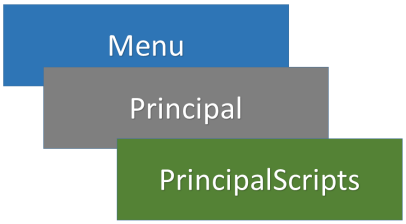

Figura 2.3. Clases de la capa de presentación

- **Capa de Lógica:** Es la encargada de manejar la lógica entre las entidades clases y objetos, y a través de la interacción con las demás capas permite dar respuesta a las acciones que ejecuta el usuario en la capa de presentación. En esta capa se utilizan las siguientes clases.

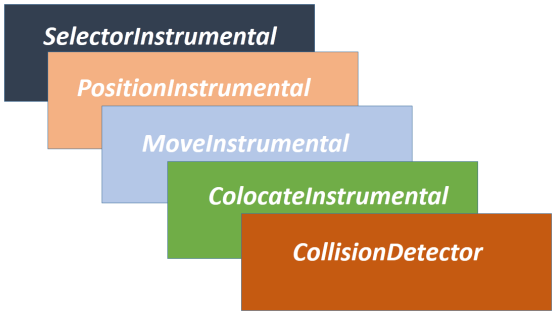

Figura 2.4. Clases de la capa lógica

- **Capa de Datos:** Esta capa en conjunto con la capa de lógica y la capa de soporte le permite al sistema obtener los datos que se encuentran guardados en un archivo *XML*. En esta capa se utilizan las siguientes clases.

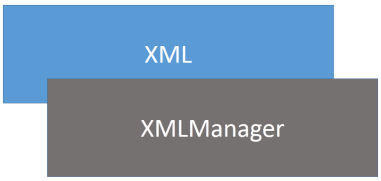

Figura 2.5. Clases de la capa lógica

- **Capa de Soporte:** Contiene las bibliotecas para la física, los gráficos, el sonido, y el almacenamiento

de los datos brindados por el motor de videojuegos *Unity 3D*.

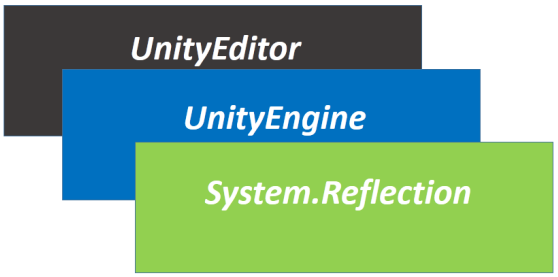

Figura 2.6. Bibliotecas de soporte

Estas capas mencionadas anteriormente son las encargadas de formar la arquitectura del sistema y se encuentran muy relacionadas entre si, puesto que existe dependencia de funcionalidad entre ellas. La capa de presentación se encarga de servir de vía de comunicación entre el usuario y las demás capas y utiliza variables y métodos implementados en la capa de soporte. La capa lógica como su nombre lo indica se encarga de procesar toda la lógica del negocio, permitiendo que el sistema realice las funcionalidades que fueron concebidas y para esto también utiliza variables y métodos predefinidos en la capa de soporte.

2.4.2. Tarjetas Clase-Responsabilidad-Colaboración

En la metodología XP no es necesaria la descripción del sistema a través de diagramas de clase utilizando notación UML, sino que se guía por técnicas como las tarjetas Clase-Responsabilidad-Colaboración (CRC).

Las tarjetas CRC son fichas en las que se escriben brevemente las responsabilidades de la clase y una lista de los objetos con los que colabora para llevar a cabo esas responsabilidades.

Tabla 2.27. Tarjeta CRC # 1

Tarjeta CRC	
Clase: SeleccionarActividad	
Responsabilidad	**Colaboración**
• Seleccionar el tipo de proceso deseado • Seleccionar la actividad de destilación deseada • Seleccionar el tipo de ejercicio deseado • Volver al menú que le antecede • Cargar la escena perteneciente al ejercicio seleccionado	Principal PrincipalScript EjercicioScript

Tabla 2.28. Tarjeta CRC # 2

Tarjeta CRC	
Clase: Navegar por la escena	
Responsabilidad	**Colaboración**
• Permite al usuario moverse hacia delante, atrás y hacia los lados en la escena • Permite al usuario girar hacia los lados en la escena • Detectar las colisiones con otros objetos de la escena	MouseLook CharacterMotor CharacterControler

Tabla 2.29. Tarjeta CRC # 3

Tarjeta CRC	
Clase: Interactuar con los modelos en la escena	
Responsabilidad	**Colaboración**
• Seleccionar los objetos de la escena • Mostrar panel para mover un objeto • Mostrar panel para posicionar un objeto • Posicionar los objetos • Mover los objetos por la escena • Detectar las colisiones con otros objetos de la escena • Mostrar panel para colocar los objetos • Colocar objetos	SelectorInstrumental PositionInstrumental MoveObjects CollisionDetector ColocateInstrumental

Tabla 2.30. Tarjeta CRC # 4

Tarjeta CRC	
Clase: Retornar al menu principal	
Responsabilidad	**Colaboración**
• Permite al usuario que se encuentra dentro de cualquier actividad regresar al menú principal	SalirActividad

Tabla 2.31. Tarjeta CRC # 5

Tarjeta CRC	
Clase: Realizar ejercicio teórico	
Responsabilidad	**Colaboración**
• Cargar fichero que contiene las preguntas teóricas • Aplicar cuestionario de preguntas teóricas • Emitir nota parcial sobre el cuestionario de preguntas teóricas	CargarPreguntas MostrarResultado RepetirCuestionario

Tabla 2.32. Tarjeta CRC # 6

Tarjeta CRC	
Clase: Mostrar nota sobre ejercicio práctico	
Responsabilidad	**Colaboración**
• Permite al usuario terminar el ejercicio práctico. • Muestra al usuario los resultados obtenidos en el ejercicio	MostrarResultadoPractico PanelAdvertencia RepetirActividadPractica

2.5. Fase de Codificación

2.5.1. Patrones del diseño

Los Patrones son soluciones comunes a problemas de diseño de *software* orientado a objetos y que además poseen ciertas características de efectividad para resolver ese problema. Son reusables ya que pueden ser aplicados en otros diseños o problemas (Rojas, 2007).

Patrones de Asignación de Responsabilidades (GRASP)

Los Patrones de Asignación de Responsabilidades (GRASP) son un sistema orientado a objetos, los cuales envían mensajes a otros objetos para llevar a cabo las operaciones requeridas. Los diagramas de interacción describen gráficamente estas operaciones, a partir de los objetos en interacción y que se responsabilizan de una actividad determinada. Estos patrones no compiten con los patrones de diseño; sino que representan una guía para ayudar a encontrar los patrones de diseño concretamente (Saavedra, 2008).

Para el desarrollo del módulo se utilizaron los patrones:

- **Experto:** Para la asignación de responsabilidades en la aplicación se utilizó el patrón **Experto** *(Expert)* el cual indica que la clase que cuenta con la información necesaria para cumplir la responsabilidad es la responsable de manejar la información. De este modo se obtendrá un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento). Sus ventajas se centran en que mantiene el encapsulamiento, los objetos utilizan su propia información para llevar a cabo sus tareas, se distribuye el comportamiento entre las clases que contienen la información requerida y son más fáciles de entender y mantener. Este se aplica en la entidad "SelectorInstrumental".

- **Creador:** Dadas las propiedades de todo sistema orientado a objetos se utilizó el patrón **Creador** *(Creator)*. Este patrón ayuda a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases. Una de las consecuencias de usar este patrón es la visibilidad entre la clase creada y la clase creadora. Una ventaja es el bajo acoplamiento, lo cual supone facilidad de mantenimiento y reutilización. Este se aplica en las entidades *"CollisionDetector"*, *"MoveObjects"* y *"MoveInstrumental"*.

- **Alta cohesión:** Este se utiliza debido a que los conceptos de cohesión y acoplamiento están íntimamente relacionados. Un mayor grado de cohesión implica un menor grado de acoplamiento. Este patrón plantea que la información que almacena una clase debe de ser coherente y debe estar (en la medida de lo posible) relacionada con la clase. Este se aplica en la entidad *"ColocateInstrumental"*.

- **Bajo acoplamiento:** Para fomentar la reutilización de código se tuvo en cuenta el patrón **Bajo Acoplamiento** *(LowCoupling)* que indica una menor dependencia entre clases. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases. Este se aplica en las entidades "MostrarResultado" y "MostrarResultadoPractico".

2.5.2. Estándares de codificación

Para el desarrollo del módulo se utilizaron convenciones propuestas por el equipo de desarrollo para la escritura el código fuente. Ejemplo: nombre de las variables, los métodos y clases y estilo de indentación. Todo esto respetando siempre el estilo de codificación del lenguaje C#.

El estándar de codificación propuesto se enfoca a la legibilidad del código ya que esto repercute directamente en la comprensión del *software* por parte del programador. Además, el mantenimiento del sistema es más fácil y puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores o mejorar el rendimiento.

Clases

Las clases comienzan con letra mayúscula y en caso de estar conformada por palabras compuestas, la definición debe ser continua y cada palabra debe iniciar con mayúscula siguiendo el estilo determinado.

Ejemplo:

public class MostrarResultado {}

Declaración de variables

Los nombres de las variables comienzan con minúscula y en caso de estar conformada por palabras compuestas, la definición debe ser continua y las demás palabras deben iniciar con mayúscula, como se muestra a continuación.
Ejemplo:

string seleccion;
int notaFinal;

Métodos

En el caso de los métodos, el nombre debe comenzar con mayúscula y en caso de estar conformada por palabras compuestas, la definición debe ser continua y cada palabra debe iniciar con mayúscula.
Ejemplo:

void Mostrar ()
void RepetirCuestionario ()

Estilo de indentación

El estilo de indentación utilizado es propio para lenguajes de programación que usan llaves para indentar o delimitar bloques lógicos de código y es también un punto clave para hacer el código más legible. Está presente en los ciclos y estructuras de control.
Ejemplo:

void Mostrar ()
{
instruccion1;
instruccion2;
}

2.6. Conclusiones Parciales

Al culminar el presente capítulo, se obtuvieron los artefactos generados por la metodología empleada, los cuales permitieron una mejor comprensión de las funcionalidades, elementos y guía de trabajo a seguir por los desarrolladores durante todo el ciclo de desarrollo del *software*. Se definió la arquitectura sobre la cual se construye la aplicación además de los patrones de diseño que se aplican al diseño del *software*.

CAPÍTULO 3

Resultados

En el presente capítulo se exponen los resultados obtenidos en el proceso de pruebas que se llevaron a cabo. Estas se realizaron con el objetivo de que los desarrolladores pudieran probar las funcionalidades descritas mediante las historias de usuario y validar la calidad del sistema, el rendimiento y las características esperadas por el cliente. Además se presentan los resultados que se obtuvieron en la validación del sistema.

3.1. Pruebas

Uno de los pilares de XP es el proceso de pruebas. XP anima a probar constantemente, tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones (Beck, 2000).
Las pruebas deben tener un conjunto de características para que se obtengan buenos resultados (Letelier, 2006):

- Crear la prueba abstrayéndose del futuro código, de esta forma se asegura la independencia de esta respecto al código que se evalúa.
- Observar la refactorización. La prueba permite verificar que un cambio en la estructura de un código no tiene por qué cambiar su funcionamiento.
- Realizar pruebas a las funcionalidades generales que debe cumplir el programa especificado en las Historias de Usuario.

Existen dos tipos de pruebas ampliamente utilizadas en la metodología XP, las pruebas unitarias y las pruebas de aceptación. Las unitarias son desarrolladas por los programadores y se encargan de verificar el código automáticamente. Las de aceptación verifican que el final de cada iteración de las historias de usuario cumple con la funcionalidad asignada y satisfagan las necesidades del cliente.

3.1.1. Pruebas de aceptación

A continuación se presentan las pruebas de aceptación diseñadas a partir de las Historias de Usuario:

Tabla 3.1. Prueba de aceptación # 1

Caso de prueba de aceptación	
Código: HU1_P1	**Historia de usuario:** 1
Nombre: Seleccionar actividad	
Descripción: Prueba para la funcionalidad selecionar una actividad	
Condiciones de ejecución:	
Pasos de ejecución: • El usuario selecciona una actividad haciendo clic en el botón con el nombre correspondiente a la actividad deseada. • El usuario mediante el botón **"Comenzar"** muestra el panel de selección correspondiente al tipo de actividad.	
Resultados esperados: • Se muestra el panel para seleccionar el tipo de actividad que se desea realizar.	
Evaluación: Satisfactoria	

Tabla 3.2. Prueba de aceptación # 2

Caso de prueba de aceptación	
Código: HU2_P2	**Historia de usuario:** 2
Nombre: Cargar actividad teórica seleccionada	
Descripción: Prueba para la funcionalidad de cargar la actividad teórica seleccionada	
Condiciones de ejecución: • El usuario debe haber seleccionado previamente una actividad teórica.	
Pasos de ejecución: • El usuario selecciona como tipo de actividad "Teórica" mediante el botón **"Teórica"**. • El usuario carga un cuestionario teórico con diez preguntas de selección y de verdadero o falso mediante el botón **"Comenzar"**.	
Resultados esperados: • Se muestra un cuestionario de diez preguntas para ser resuelto por el usuario.	
Evaluación: Satisfactoria	

Tabla 3.3. Prueba de aceptación # 3

Caso de prueba de aceptación	
Código: HU2_P3	**Historia de usuario:** 2
Nombre: Cargar actividad práctica seleccionada	
Descripción: Prueba para la funcionalidad de cargar la actividad práctica seleccionada	
Condiciones de ejecución: • El usuario debe haber seleccionado previamente una actividad práctica.	
Pasos de ejecución: • El usuario selecciona como tipo de actividad "Práctica" mediante el botón **"Práctica"**. • El usuario mediante el botón **"Comenzar"** carga el escenario que contiene los elementos a utilizar en el desarrollo de la actividad práctica seleccionada.	
Resultados esperados: • Se carga el escenario correspondiente a la actividad práctica seleccionada en el que el usuario puede desplazarse libremente y realizar la actividad.	
Evaluación: Satisfactoria	

Tabla 3.4. Prueba de aceptación # 4

Caso de prueba de aceptación	
Código: HU3_P4	**Historia de usuario:** 3
Nombre: Retornar al menú principal desde la actividades teóricas	
Descripción: Prueba para la funcionalidad de retornar al menú principal	
Condiciones de ejecución: • El usuario debe haber cargado previamente una actividad de tipo teórica.	
Pasos de ejecución: • El usuario mediante el botón **"Atrás"** regresa al menú principal en el panel correspondiente a la selección del tipo de actividad.	
Resultados esperados: • Se muestra el menú principal en el panel correspondiente a la selección del tipo de actividad.	
Evaluación: Satisfactoria	

Tabla 3.5. Prueba de aceptación # 5

Caso de prueba de aceptación	
Código: HU3_P5	**Historia de usuario:** 3

Continúa en la próxima página

Tabla 3.5. Continuación de la página anterior

Nombre: Retornar al menú principal desde la actividades prácticas
Descripción: Prueba para la funcionalidad de retornar al menú principal
Condiciones de ejecución: • El usuario debe haber cargado previamente una actividad de tipo práctica. • El usuario debe haber terminado la actividad.
Pasos de ejecución: • El usuario mediante el botón **"Salir"** regresa al menú principal en el panel correspondiente a las destilaciones.
Resultados esperados: • Se muestra el menú principal en el panel correspondiente a las destilaciones.
Evaluación: Satisfactoria

Tabla 3.6. Prueba de aceptación # 6

Caso de prueba de aceptación	
Código: HU4_P6	**Historia de usuario:** 4
Nombre: Navegar por la escena	
Descripción: Prueba para la funcionalidad de navegar por el escenario	
Condiciones de ejecución: • El usuario debe haber cargado previamente una actividad de tipo práctica.	
Pasos de ejecución: • El usuario se desplaza por el escenario mediante las teclas **A**, **W**, **D**, **S** y el **botón central del *mause***.	
Resultados esperados: • El usuario se mueve libremente en el escenario.	
Evaluación: Satisfactoria	

Tabla 3.7. Prueba de aceptación # 7

Caso de prueba de aceptación	
Código: HU7_P7	**Historia de usuario:** 7
Nombre: Emitir nota sobre la actividad teórica	
Descripción: Prueba para la funcionalidad de emitir nota sobre la actividad teórica	

Continúa en la próxima página

Tabla 3.7. Continuación de la página anterior

Condiciones de ejecución: • El usuario debe haber cargado previamente una actividad de tipo teórica.
Pasos de ejecución: • El usuario responde el cuestionario y mediante el botón **"Terminar"** muestra un panel con los resultados obtenidos por el usuario en cuestionario.
Resultados esperados: • El usuario obtiene los resultados del cuestionario en un panel de resultados.
Evaluación: Satisfactoria

Tabla 3.8. Prueba de aceptación # 8

Caso de prueba de aceptación	
Código: HU8_P8	**Historia de usuario:** 8
Nombre: Seleccionar instrumental	
Descripción: Prueba para la funcionalidad que permite seleccionar el intrumental a utilizar en la actividad práctica.	
Condiciones de ejecución: • El usuario debe haber cargado previamente una actividad de tipo práctica.	
Pasos de ejecución: • El usuario al hacer clic en los objetos situados sobre la mesa de selección muestra un panel con las opciones de tomar o cancelar. • El usuario mediante el botón **"Tomar"** coloca el objeto en la mesa de trabajo.	
Resultados esperados: • El usuario coloca el instrumental que se utiliza en la actividad sobre la mesa de trabajo.	
Evaluación: Satisfactoria	

Tabla 3.9. Prueba de aceptación # 9

Caso de prueba de aceptación	
Código: HU9_P9	**Historia de usuario:** 9
Nombre: Posicionar instrumental	
Descripción: Prueba para la funcionalidad que permite posicionar el intrumental a utilizar en la actividad práctica.	

Continúa en la próxima página

Tabla 3.9. Continuación de la página anterior

Condiciones de ejecución: • El usuario debe haber seleccionado previamente el instrumental para la actividad.
Pasos de ejecución: • El usuario al hacer clic en los objetos situados sobre la mesa de trabajo muestra un panel con las opciones de tomar o cancelar. • El usuario mediante el botón **"Tomar"** suspende en el aire el objeto para ser movido a su posición de ensamblaje.
Resultados esperados: • El usuario posiciona los elementos para ser ensamblados de manera individual.
Evaluación: Satisfactoria

Tabla 3.10. Prueba de aceptación # 10

Caso de prueba de aceptación	
Código: HU10_P10	**Historia de usuario:** 10
Nombre: Ensamblar elemento	
Descripción: Prueba para la funcionalidad que permite ensamblar los elmento correspondiente a la actividad.	
Condiciones de ejecución: • El usuario debe haber posicionado previamente el elemento a ensamblar.	
Pasos de ejecución: • El usuario al posicionar el elemento en una posible posición correcta muestra un botón **"Colocar"** que posiciona el elemento en su posición final.	
Resultados esperados: • El usuario posiciona el elemento en su posición de ensamblaje final.	
Evaluación: Satisfactoria	

Tabla 3.11. Prueba de aceptación # 11

Caso de prueba de aceptación	
Código: HU11_P11	**Historia de usuario:** 11
Nombre: Ensamblar elemento	
Descripción: Prueba para la funcionalidad que permite mostrar los resultado obtenidos por el usuario en la actividad práctica.	

Continúa en la próxima página

Tabla 3.11. Continuación de la página anterior

Condiciones de ejecución: • El usuario debe haber cargado previamente una actividad de tipo práctica.
Pasos de ejecución: • El usuario mediante el botón **"Terminar"** muestra un panel con los resultados obtenidos en la práctica.
Resultados esperados: • El usuario obtiene los resultados de la actividad práctica en un panel de resultados.
Evaluación: Satisfactoria

Al concluir las pruebas en cada una de las iteraciones se obtuvieron algunas no conformidades las que fueron resueltas en cada iteración. A continuación se muestra una gráfica que relaciona cuantas no conformidades fueron detectadas y resueltas en cada iteración.

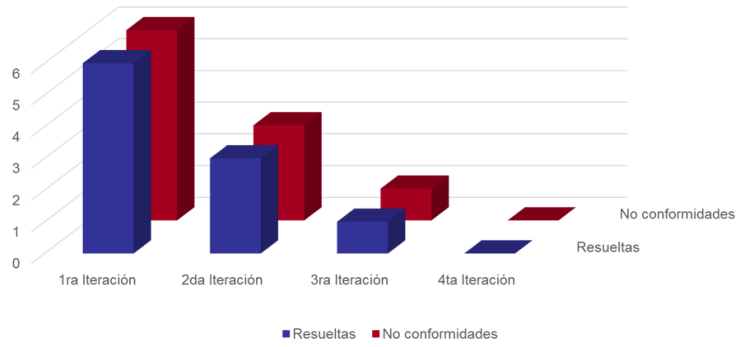

Figura 3.1. Resultados de las pruebas de Aceptación

3.2. Pruebas unitarias

La producción de código está dirigida por las pruebas unitarias. Las pruebas unitarias son establecidas antes de escribir el código y son ejecutadas constantemente ante cada modificación del sistema. Los clientes escriben las pruebas funcionales para cada historia de usuario que deba validarse (Letelier, 2006).

En cada una de las iteraciones se realizaron pruebas de unidad haciendo uso de la técnica de camino básico. A continuación se muestra una de las pruebas realizadas en la primera iteración.

```
public void comenzarActividad()
{

    if (numActividad==0)
    {
        Actividades.GetComponent<TweenPosition>().PlayReverse();
        Destilacion.GetComponent<TweenPosition>().Play();
        panel=2;
    }
    else if (numActividad==1)
    {
        Actividades.GetComponent<TweenPosition>().PlayReverse();
        Extraccion.GetComponent<TweenPosition>().Play();
        panel=3;

    }
    else
    {

        panel=1;
    }

}
```

Figura 3.2. Código fuente del método comenzarActividad()

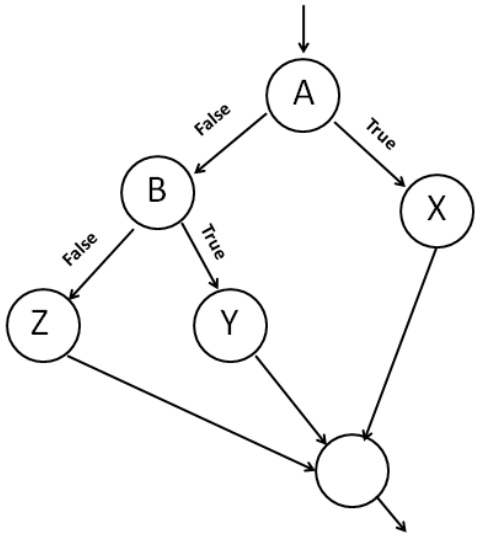

Figura 3.3. Grafo de flujo asociado al método comenzarActividad()

Luego de haber construido el grafo de flujo se realiza el cálculo de la complejidad ciclomática mediante la fórmula:

V(G) = Aristas - Nodos + 2

V(G) = 7 - 6 + 2

V(G) = 3

Una vez calculada la complejidad ciclomática podemos saber la cantidad de posibles caminos que existen y la cantidad de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez. En este caso existen 3 caminos posibles y se deben realizar 3 casos de pruebas.

Los caminos posibles son:

Camino1: A-X

Camino2: A-B-Y

Camino3: A-B-Z

Tabla 3.12. Casos de prueba definidos para los caminos

No. Camino	Caso de prueba	Resultados esperados	Resultado de la prueba
1	"Seleccionar Destilación"	Cambia las posiciones de los paneles Actividad y Destilaciones.	Se cambiaron de manera correcta las posiciones.
2	"Seleccionar Extracción"	Cambia las posiciones de los paneles Actividad y Extracciones.	Se cambiaron de manera correcta las posiciones.
3	"No seleccionó ninguna opción"	El panel Actividades mantiene la posición.	Se mantuvo el panel Actividades en su posición.

Luego de aplicar los distintos casos de pruebas, se pudo comprobar que el flujo de trabajo de la función es correcto ya que los resultados obtenidos son equivalentes a los esperados.

Durante las pruebas unitarias se obtuvieron algunas no conformidades la cuales fueron resultas en cada una de las iteraciones. A continuación se muestra una gráfica *(Figura 3.4)* que relaciona cuantas no conformidades fueron detectadas y resueltas en cada iteración.

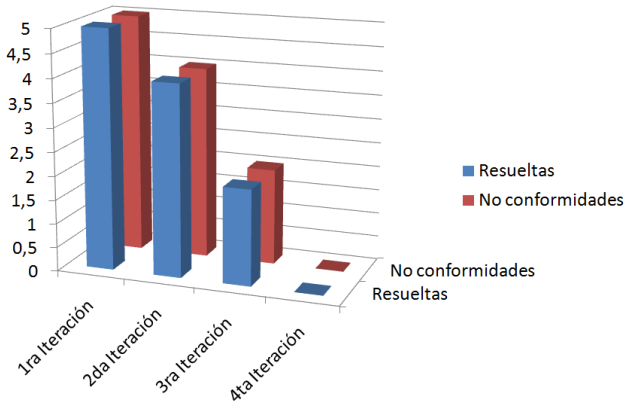

Figura 3.4. Gráfica de no conformidades detectadas y resueltas

3.3. Validación del sistema

Una vez concluido el proceso de desarrollo del *software* y terminadas las pruebas de manera satisfactoria, se realizó una encuesta *(Figura 3.5)* en dos formatos, uno con expresiones afirmativas y el otro con interrogaciones. En ella los encuestados debían responder Bien, Regular o Mal para el primer caso y Si o No[1] para el segundo.

Esta encuesta anónima permitirá establecer mejoras al software que Uds. evaluarán.
Por esa razón les agradecemos mucho su colaboración.

Califica como B, R o M, cada una de las afirmaciones siguientes	Bien	Regular	Mal
1) Ud. como usuario sabe a cada momento sobre sus posibles errores.			
2) Ud. como usuario es correctamente alertado sobre los siguientes pasos a ejecutar.			
3) Las imágenes y objetos mostrados le permiten a Ud. experimentar un alto nivel de realismo.			
4) Con el laboratorio virtual resulta más fácil visualizar e interpretar los conceptos teóricos de la asignatura.			
5) Las preguntas teóricas le ayudan a consolidar los conocimientos en la asignatura.			
6) El laboratorio virtual le resulta útil para comprender y fijar conceptos sobre los procesos de destilación.			
7) Fue fácil realizar cada ejercicio.			
8) Los nombres que aparecen en el laboratorio describen correctamente cada componente.			

Marque Si o No según su criterio	Si	No
9) ¿Fue fácil para Ud. interactuar con el laboratorio virtual?		
10) ¿Considera que este laboratorio puede ser una herramienta de apoyo en el proceso de enseñanza-aprendizaje?		

Figura 3.5. Encuesta

Los parámetros incluidos para evaluar y las preguntas asociadas a cada uno de ellos se muestran en la *Tabla 3.13*.

[1]Las preguntas de Si o No se toman como Bien o Mal respectivamente

Tabla 3.13. Relación entre parámetros y preguntas

Parámetro	Preguntas
Usabilidad	7,8
Interactividad	1,2,9
Calidad visual	3
Contribución	4,5,6,10

La misma se le aplicó a un grupo de 15 personas, entre las que se encontraban, especialistas, estudiantes y profesores de Química. Después de explicar el funcionamiento del sistema y los objetivos perseguidos, se les permitió interactuar con éste, para que de esa manera pudieran evaluar las funcionalidades del módulo desarrollado y emitir los criterios correspondientes.

En la *Tabla 3.14* se presentan los valores correspondientes a cada una de las preguntas.

Tabla 3.14. Resultados de la encuesta

Parámetro	Pregunta	Bien	Regular	Mal	% de Bien
Usabilidad	7	10	3	2	66, 7
	8	15	0	0	100
Interactividad	1	9	4	2	60
	2	8	4	3	53,3
	9	12	0	3	80
Calidad visual	3	13	1	1	86,7
Contribución	4	14	1	0	93,3
	5	12	2	1	80
	6	15	0	0	100
	10	13	2	0	86,7

De la tabla se puede observar como en todos los casos, los encuestados calificaron de bien los diferentes parámetros seleccionados para encuestar. Sin embargo existe una contradicción aparente en las respuestas a las preguntas encaminadas a evaluar el parámetro interactividad, lo cual puede deberse a diferentes causas. Estas pudieran ser de orden subjetivo cuando los encuestados respondieron positivamente a la pregunta 9 en la que se les inquiría de manera directa acerca de la interactividad de la aplicación, en la que las respuestas pudieron estar condicionadas por la habilidad que gran número de jóvenes poseen por la utilización de juegos en 3D. Sin embargo, para posteriores versiones deberá tenerse en cuenta las opiniones adversas reflejadas en los bajos porcientos de aceptación asociados a las preguntas 1 y 2.

La usabilidad presenta también un comportamiento aparentemente divergente en las respuestas al encontrarse una discrepancia de casi un 40 % en las dos preguntas asociadas (7 y 8). No obstante, aunque estimadas como asociadas al mismo parámetro, no están estrictamente vinculadas pero, debe tenerse en cuenta el relativamente bajo nivel de aceptación para futuras versiones de la aplicación.

Un análisis global de los resultados muestra que el índice de aceptación general, calculado del porciento

total de calificaciones de bien entre el número total de respuestas da un 80 % de aceptación general, lo cual, para esta primera versión, constituye un índice aceptable. Por lo tanto se considera que el módulo desarrollado constituye una alternativa de apoyo al proceso de enseñanza-aprendizaje de la destilación en la química práctica.

Además la encuesta realizada y como parte de la validación del sistema se emitió un acta de aceptación *(Anexo 1)* donde queda expuesta la aceptación del producto por parte del cliente, en este caso los tutores.

3.4. Conclusiones Parciales

En el capítulo se realizaron las pruebas de aceptación necesarias a las historias de usuarios antes definidas e implementadas, las cuales se culminaron y obtuvieron resultados satisfactorios. Además, se realizó una encuesta en la que se obtuvieron resultados que apoyan la factibilidad del *software*, y se emitió una un acta de aceptación del producto.

Conclusiones

Para dar cumplimiento al objetivo del trabajo se desarrolló un módulo para el Laboratorio Virtual de Química en 3D que permite la realización de ejercicios prácticos y teóricos para el ensamblaje de cuatro diferentes equipos de destilación (simple, al vacío, fraccionada y arrastre por vapor), que incluye recreación de los fenómenos y un procedimiento para la evaluación integral de cada una de las prácticas. Además, se corroboró que la utilización del motor de videojuegos *Unity 3D* es una buena alternativa para la creación de módulos de software para laboratorios virtuales de química, ya que permite crear un ambiente virtual de elevado realismo de una manera factible y además porque brinda soporte para la implementación y flexibilidad para el desarrollo e integración de nuevas herramientas y módulos.

Recomendaciones

Para futuras versiones se recomienda:

- Añadir niveles de diferente complejidad que le permitan al usuario, realizar actividades con diferentes grados de dificultad.
- Adecuar el sistema de evaluación del módulo para que las calificaciones se correspondan con la complejidad de los ejercicios.
- Adicionar una funcionalidad que permita al usuario recibir una nota final que refleje la nota conjunta del ejercicio teórico y el práctico.

Acrónimos

CRC Clase-Responsabilidad-Colaboración. 33

TIC Tecnologías de la Información y las Comunicaciones. 2–5

XP Extreme Programming. 14–16, 39

Referencias bibliográficas

Astudillo, Hernán (2014). «Arquitectura de Software». En: (vid. pág. 31).

Autodesk (2012). *Productos Autodesk 3ds Max*. URL: http://www.autodesk.es/adsk/servlet/pc/index?siteID=455755&id=14626995. (vid. pág. 17).

Avila, Aylén (2013). «Laboratorio de Química Orgánica». En: (vid. pág. 11).

Beck, Kent (2000). *Extreme programming explained: embrace change*. Addison-Wesley Professional (vid. pág. 39).

Bermejo, Sergio y Aniceto Saboya (2004). «Tutores inteligentes basados en asistentes personales». En: *Disponible Junio* (vid. pág. 5).

Boix, O, S Fillet y J Bergas (2002). «Nuevas posibilidades en laboratorios remotos de enseñanzas técnicas». En: *Congreso Virtual CIVE 2002* (vid. pág. 5).

Cabero, J (2007). «Las TICs en la enseñanza de la química: aportaciones desde la Tecnología Educativa». En: *Química: vida y progreso, Murcia, Asociación de Químicos de Murcia* (vid. pág. 6).

Clips, Crocodile (2009). *Crocodile Chemistry*. URL: http://www.crocodile-clips.com/es/Crocodile_Chemistry/ (vid. pág. 7).

Fernandez, Rodrigo O et al., (2002). «Laboratório virtual aplicado à Educaçao a Distancia». En: *Internet: http://sim. lme. usp. br/˜ nathalia/publication/sbie00. pdf* (vid. pág. 5).

GEA (2009). *Destilación*. URL: http://www.gea-niro.com.mx/lo-que-suministros/destilacion.asp (vid. pág. 13).

Gil, L, E Blanco y JM Aulí (2000). «Software educativo orientado a la experimentación». En: *I Congreso Internacional de Docencia Universitaria e Innovación*, pág. 118 (vid. pág. 5).

González-Castaño, Francisco J et al., (2001). «Internet access to real equipment at computer architecture laboratories using the Java/CORBA paradigm». En: *Computers & Education* 36.2, págs. 151-170 (vid. pág. 5).

Joskowicz, José (2008). «Reglas y prácticas en eXtreme Programming». En: *Universidad de Vigo*, pág. 22 (vid. pág. 15).

LABS, STUDYROOM (2010). *LABORATORIO DE QUIMICA VIRTUAL QUIMILAB PROGRAMAS PARA EDUCACION VIRTUAL*. URL: http://www.studyroomlabs.com/Software/Simulacion_laboratorio_quimica.htm (vid. pág. 9).

Letelier, Patricio (2006). «Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)». En: (vid. págs. 14, 15, 21, 25-27, 39, 45).

Marqués, Pere (2001). «Algunas notas sobre el impacto de las TIC en la universidad». En: *EDUCAR* (vid. pág. 4).

Marti, Albert Gras (2000). «Uso de las TIC en la enseñanza de la Física». En: *II Jornadas Nacionales de Tecnología y Educación* (vid. pág. 5).

Mono (2013). «MonoDevelop». En: (vid. pág. 18).

Moya, Rafael (2012). «Guía para Modelar arquitectura en Blender». En: (vid. pág. 16).

Olivera, Pedro Zaine Cabrera (2012). *IMPORTANCIA DE LAS PRÁCTICAS DE LABORATORIO EN EDUCACIÓN*. URL: http://tecnologiaeducativazaineuvm.blogspot.com/2012/05/importancia-de-las-practicas-de.html (vid. págs. 13, 14).

Palacio, Juan y Claudia Ruata (2011). *Scrum Manager Gestión de Proyectos* (vid. pág. 14).

Patón, Medina (2012). «Metodologías de Desarrollo de Software». En: (vid. pág. 14).

Ramírez, Eduardo (2010). *Destilación. Teoria y tipos de destilación*. URL: http://www.alambiques.com/destilaciones.htm (vid. págs. 10-13).

Rojas, Mc Juan Carlos Olivares (2007). «Patrones de Diseño». En: (vid. pág. 35).

Rosado, Luis y Juan Ramón Herreros (2003). *Internet y multimedia en didáctica e investigación de la física: tratado teórico-práctico para profesores y doctorandos*. Universidad Nacional de Educación a Distancia, UNED (vid. pág. 5).

Saavedra, Jorge (2008). «GRASP». En: (vid. pág. 35).

SOFT, SIBEES (2000). *VlabQ SIMULADOR DE EXPERIMENTOS QUIMICOS*. URL: http://www.sibees.com/prog.php?id=11 (vid. pág. 8).

Technologies (2011). «Motores Gráficos». En: (vid. pág. 18).

Technologies, Unity (2010). «Unity 3D. Navmesh and Pathfinding». En: (vid. pág. 18).

Tirado Granados, Gonzalo (2010). «Introducción a Ogre 3D». En: *Proyecto de innovación Educativa PIE07-084 subvencionado por el Servicio de innovación Educativa y el Servicio de Enseñanza Virtual y Laboratorio Tecnológicos de la Universidad de Málaga.* Pág. 73 (vid. pág. 17).

Turégano, JA y JM Cózar (1994). «Generalización del carácter práctico de los estudios universitarios: el laboratorio de prácticas con ordenadores personales». En: *Actas Congreso UNIMAC'94* (vid. pág. 5).

UNED (2010). «Laboratorio virtual». En: (vid. pág. 4).

Unity (2011). *Motor de videojuegos Unity 3D*. URL: http://developeando.net/unity3d/ (vid. pág. 19).

Vary, Jame (2000). «Informe de la reunión de expertos sobre laboratorios virtuales». En: *Instituto Internacional de Física Teórica y Aplicada (IITAP), Ames, Iowa–UNESCO, París* (vid. pág. 4).

Vidal Castaño, Gonzalo e Hilda González Medina (2002). «EVALUACIÓN PEDAGÓGICA DEL SIMULADOR DEL LABORATORIO QUÍMICO MODEL CHEMLAB». En: *Revista Pedagogía Universitaria* (vid. pág. 7).

Apéndices

Acta de aceptación

ACTA DE ACEPTACIÓN

En cumplimiento del **Proyecto de Tesis de Diploma** y en función de la ejecución del proyecto de investigación-desarrollo: ***Impacto de los Laboratorios Virtuales de Química en la Enseñanza Práctica de la Disciplina*** se hace entrega de los productos que se relacionan a continuación:

- *Módulo de destilación para el Laboratorio Virtual de Química*

La Parte Tutor, luego de haber revisado el producto del trabajo determina que se acepta el software creado porque constituye el eslabón inicial de un proyecto mayor y ha sido realizado con rigor y calidad, cumpliendo con las especificaciones que se demandaron para esta etapa.

Comentarios

Para la ejecución del software y la documentación que lo acompaña, los estudiantes Reynier Palomino Echeandia y Leonar Alemañy Socarrás debieron incursionar en áreas del conocimiento ajenas a la especialidad como lo es la química, e incorporar otros nuevos como es que caso de la programación empleando UNITY 3D.

Entrega	**Recibe**
Nombre y apellidos: Reynier Palomino Echeandia y Leonar Alemañy Socarrás	**Nombre y apellidos:** Jandy Miguel Gómez-González y Ramón Carrasco-Velar
Cargo:	**Cargo:** Tutores
Firma:	**Firma:**

Fecha: 30/04/2015