UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 5

INTERFAZ DE USUARIO PARA LOS LABORATORIOS VIRTUALES DE INFORMÁTICA

Trabajo de Diploma para optar por el
Título de Ingeniero en Ciencias Informáticas

Autor
Juan Miguel Rodríguez Sillero

Tutor
Ing. Yaself Machado Tugores

Co-tutor
Ing. Gelson Rafael Saurin Ojeda

Ciudad de La Habana, junio 2011
"Año 53 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor del presente trabajo y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmamos la presente a los __ días del mes de _____del año _____.

_____
Juan Miguel Rodríguez Sillero
Autor

_____
Ing. Yaself Machado Tugores
Tutor

_____
Ing. Gelson Rafael Saurin Ojeda
Co-Tutor

Datos de Contacto

Tutor: Ing. Yaself Machado Tugores.

Edad: 26 años**.**

Ciudadanía: Cubano.

Institución: Universidad de las Ciencias Informáticas.

Título: Ingeniero en Ciencias de Computación.

Categoría Docente: Profesor Instructor.

E-mail: ytugores@uci.cu

Graduado de la Universidad de Ciencias Informáticas.

Co-Tutor: Ing. Gelson Rafael Saurin Ojeda.

Edad: 26 años.

Ciudadanía: Cubano.

Institución: Universidad de las Ciencias Informáticas.

Título: Ingeniero en Ciencias de Computación.

Categoría Docente: Profesor Instructor.

E-mail: grsaurin@uci.cu

Graduado de la Universidad de Ciencias Informáticas.

Resumen.

En los últimos años el uso de las Tecnologías de la Información y las comunicaciones (TICs) ha ido evolucionando e insertándose en distintas ramas de nuestra vida diaria. El uso de las mismas para la educación ha contribuido de manera sorprendente en el desarrollo del aprendizaje de los estudiantes, utilizándose para esto los entornos virtuales y por último introduciéndose en las mismas el concepto de realidad virtual, la cual busca recrear a través de la tecnología sucesos que ocurren en el mundo real. Los Laboratorios Virtuales son una muestra más de cuanto puede hacerse para contribuir con el desarrollo de la enseñanza mediante las TICs. Los mismos pueden ser utilizados como herramientas muy potentes para fomentar los conocimientos adquiridos por los estudiantes en el aula, y además son una alternativa a la falta de recursos didácticos en algunas materias.

Para lograr insertar a los estudiantes en este nuevo mundo virtual es necesario que los sistemas desarrollados muestren una Interfaz Gráfica de Usuario (GUI) que esté acorde con las exigencias de los individuos que los van a utilizar. El presente trabajo de diploma aborda el tema de desarrollo de interfaz gráfica de usuario, lo cual permite una interacción entre los Laboratorios Virtuales y las personas que van a utilizar los mismos.

Para el desarrollo de esta investigación fue necesario abordar en otros temas como: conceptos, estándares, buenas prácticas los tipos de interfaz de usuario existentes en el mundo. También y muy importante fue el estudio de diferentes herramientas multiplataforma relacionadas con el tema, tales como Qt, que es una herramienta para desarrollar interfaz gráfica de usuario, y otras aplicaciones, en consola o para servidores, lenguaje de programación C++. La validación de la misma está basada en pruebas por funcionalidad, y además la misma está incluida en un proyecto desplegado en la hermana República Bolivariana de Venezuela, resultando de muy buena aceptación por los clientes.

Palabras claves.
Interfaz Gráfica de Usuario (GUI), Entornos Virtuales, Laboratorios Virtuales, Interacción.

Introducción

En la actualidad, existe un gran desarrollo científico-técnico donde las Tecnologías de la información y las comunicaciones (TIC) desempeñan un papel trascendental. Dentro de las mismas se destaca la fuerte relación que existe entre la Educación y Tecnología, la cual tiene un proceso de retroalimentación en las que ambas partes se benefician y desarrollan. La utilización los escenarios virtuales para el estudio de distintas ramas de la educación puede ser de gran apoyo para el aprendizaje del estudiante, así como una opción viable ante la falta de recursos que impiden llevar a cabo de una forma tradicional los ejercicios que se simulan.

Nuestro país no se encuentra exento de esta situación. Por lo que surge la idea de iniciar el Proyecto Laboratorios Virtuales (PROLAVI), perteneciente al Centro de Informática Industrial (CEDIN), que se encuentra en la Universidad de las Ciencias Informáticas. El objetivo de dicho proyecto es desarrollar prácticas de laboratorios virtuales para suplir la necesidad de recursos materiales y medios didácticos interactivos en los institutos educacionales.

En estos momentos el grupo de desarrollo se encuentra trabajando en la elaboración de tres laboratorios virtuales: Ensamblaje de un computador, Diseño e Instalación de una red de área local (LAN), y Configuración y Administración de una red de área local (LAN). Los cuales deben contar con varios requisitos dentro de los que se encuentran, brindar al cliente una interfaz amigable, funcional e interactiva que permita a los usuarios interactuar virtualmente de forma fácil con las aplicaciones y que además esté acorde con las especificaciones de los requerimientos de cada uno de los laboratorios virtuales teniendo en cuenta el criterio del cliente, por lo que cada vez se hace más imperativo la mejor interacción hombre-computadora a través de una adecuada interfaz (Interfaz de usuario), que brinde tanto comodidad, como eficiencia. Condiciones con las que deben cumplir los Laboratorios Virtuales, lo que representa un gran reto para el equipo de desarrollo.

Para que los usuarios finales logren la interacción con las prácticas virtuales de laboratorio, necesitan ¨algo¨ que los comunique con el sistema, que constituya a la vez un límite y un espacio común entre ambas partes, lo que se denomina interfaz. Una interfaz inteligente y bien definida es diseñada específicamente según los principios de cada usuario final. Pueden existir interfaces con propósitos comunes, apariencias diferentes y resultados exitosos para el cliente. Diseñada específicamente para la gente que la usará pues la

mayoría de los programas y sistemas operativos ofrecen varias formas de interacción al usuario, a diario interactuamos con cientos de ellas. Muchas son tan conocidas y aceptadas, que se han convertido en un hábito común. Sin embargo numerosas de las interfaces, por nuevas, desconocidas o mal diseñadas, son visibles.

El resultado del mal diseño de una interfaz de usuario, hace que surja la interrogante ¿Cómo encontrar lo que buscamos o cómo hacer lo que queremos con el sistema?, como respuesta a la interrogante se crea un problema de usabilidad entre el usuario y el programa. Por lo que se ha comprobado que el mejor sistema o la herramienta perfecta, es inútil si no podemos interactuar con ella.

En el proyecto PROLAVI los tres laboratorios virtuales que se están desarrollando, no cuentan aún con una interfaz de usuario configurable y funcional que permita la interacción de los usuarios con los Laboratorios Virtuales, que sea capaz de ajustarse a cada una de las generalidades y particularidades de los mismos. Teniendo en cuenta lo planteado surge la necesidad de crear un módulo de interfaz de usuario que posibilite la adaptabilidad con cada una de las aplicaciones a desarrollar.

Dada la situación problémica anteriormente expuesta se define el siguiente **problema científico:** ¿Cómo lograr la interacción de los usuarios con Laboratorios Virtuales, desarrollados por el proyecto PROLAVI?

El **objeto de la investigación** se centra en el desarrollo de interfaz de usuario, enfocándose en el **campo de acción** desarrollo de interfaz gráfica para laboratorios virtuales.

El **objetivo** consiste en desarrollar una interfaz gráfica de usuario configurable y funcional que permita la interacción Hombre-Computadora para los laboratorios virtuales del proyecto PROLAVI.

Como idea a defender de la investigación se tiene: que con el desarrollo de esta interfaz de usuario se logrará la interacción que se persigue con la realización de los laboratorios virtuales, y se contribuirá con agilizar el tiempo de entrega de cada uno de los productos solicitados por el cliente.

Tareas de investigación.

- Análisis del estado del arte referente a los distintos patrones y estándares de interfaz de usuario obteniéndose las bases de la investigación propuesta.
- Selección de las herramientas para el desarrollo de interfaz de usuarios.
- Diseño una arquitectura de clases que permita desarrollar una interfaz de usuario que se adapte a los laboratorios virtuales.
- Análisis los requisitos que deben cumplir las interfaces acorde con la descripción de los casos de usos y el prototipo de interfaz.
- Implementación de una interfaz gráfica genérica para los laboratorios virtuales.
- Realización de pruebas en distintas plataformas para comprobar compatibilidad.

Para todo el proceso de investigación y elaboración de este trabajo se tomará en cuenta la utilización de varios métodos científicos de investigación como:

Histórico – Lógico: método teórico mediante el cual se constatará como ha sido la trayectoria histórica real, la evolución y desarrollo de los aspectos principales de las interfaces de usuario.

Analítico – Sintético: este método teórico será utilizado en la investigación para buscar la esencia de lo que existe en el mundo acerca de la interfaz de usuario para aplicaciones laboratorios virtuales 3D, los rasgos que los caracterizan y los distinguen.

Entrevista: Método empírico que permite obtener información proveniente de los clientes acerca de interfaz de usuario, y los requisitos no funcionales que los mismos desean que tengan los Laboratorios Virtuales.

Consultas de las fuentes de información: Es un método empírico que va a permitir la búsqueda de información existente en el mundo actualmente sobre el tema de las interfaces de usuario.

Entrevistas con los clientes: Método que empírico que va a permitir conocer los requerimientos visuales y expectativas del cliente para el trabajo con los Laboratorios Virtuales y finalmente permitirá conocer la aceptación del producto final.

Consulta de especialistas: Consulta a especialistas en las materias a tratar en los laboratorios virtuales, para conocer los ejercicios a desarrollar y las habilidades a lograr en los usuarios.

La tesis de grado estará estructurada de la siguiente manera:

Capítulo 1. Fundamentación teórica.

Capítulo 2. Propuesta de solución.

Capítulo 3. Diseño del sistema.

Capítulo 4. Implementación y validación.

1

Capítulo 1. Fundamentación teórica.

Introducción.

En el presente capítulo se hace alusión a toda la teoría que sustenta el presente trabajo de diploma, donde se exponen un conjunto de conceptos y características fundamentales acerca los principales términos utilizados para el desarrollo de la investigación dentro de los que se encuentran: Interfaz de usuario, laboratorios virtuales, lenguaje de programación, así como las herramientas, bibliotecas utilizadas y metodología de desarrollo de la interfaz de usuario para los laboratorios virtuales.

1.1. ¿Qué son los laboratorios virtuales?

“El laboratorio virtual es un tipo de colaboración centrada en el logro de determinados objetivos creativos o de ayuda a la toma de decisiones. Por lo tanto, un laboratorio virtual puede dedicarse prácticamente a todas las esferas de la actividad intelectual humana.

Es un “centro sin paredes” cuyos usuarios pueden investigar sin tener en cuenta su situación geográfica interactuando con los colegas, teniendo acceso a los instrumentos, compartiendo los datos y los recursos informáticos, recurriendo a la información de las bibliotecas electrónicas. Ese entorno se apoya en unos programas informáticos que permiten trabajar en colaboración y simultáneamente a diversas personas desde distintos sitios. *[1]*

Teniendo en cuenta los conceptos anteriormente citados por otros autores, se puede definir que un Laboratorio Virtual es un grupo de herramientas y materiales que pueden ser capaces de proveer experiencias en el aprendizaje de los estudiantes, y ofreciendo a los mismos habilidades diferentes. Constituyen también una alternativa informática para suplementar la carencia de algunos recursos didácticos de educación.

1.1.1 Tipos de Laboratorios Virtuales.

Para el estudio y del desarrollo de los laboratorios se han dividido en tres grupos fundamentales teniendo en cuenta sus características particulares: [2]

- *Laboratorios virtuales software:* Están desarrollados como un programa independiente y para ser ejecutados en los ordenadores, su servicio no requiere de un servidor Web.

- *Laboratorios virtuales Web:* Este tipo de laboratorios se basa en un software que depende de los recursos de un servidor determinado.

- *Laboratorios remotos:* Estos laboratorios requieren de equipos servidores específicos que les den acceso a las máquinas a operar de forma remota, y no pueden ofrecer su funcionalidad ejecutándose de forma local.

1.2. Interfaz.

Después de un profundo análisis de las fuentes consultadas se seleccionaron varias definiciones acerca del término “interfaz” a las cuales se hace referencia a continuación.

El Diccionario de la Real Academia de la Lengua Española define interfaz como una palabra derivada del término inglés “interfaces” (superficie de contacto) y la define de la siguiente manera: Conexión física y funcional entre dos aparatos o sistemas independientes.

En un sentido más amplio se puede definir interfaz como el conjunto de comandos y/o métodos que permiten la intercomunicación del programa con cualquier otro programa o entre partes (módulos) del propio programa o elemento externo. De hecho, los periféricos son controlados por interfaces. [3]

Parte de un programa que permite el flujo de información entre el usuario y la aplicación, o entre la aplicación y otros programas o periféricos. Esa parte de un programa está construida por un conjunto de comandos y métodos que permiten estas intercomunicaciones. La pantalla de una computadora es una interfaz entre el usuario y el disco duro de la misma. [4]

1.3. Interfaces para sistemas informáticos.

Para llegar a establecer una buena interacción entre el usuario y el sistema informático entre ambos debe existir una herramienta que permita esa comunicación. [5] La cual

constituye un traductor entre el lenguaje de 0 y 1 de la computadora y el lenguaje común del humano. Esta herramienta se conoce como interfaz de usuario.

Elaborar una interfaz de usuario, que esté bien diseñada, exige una gran dedicación; ya que generalmente la interfaz de cualquier programa es una de las partes primordiales a tener en cuenta. Un usuario que va a utilizar un sistema o programa y se encuentra con una interfaz pobremente elaborada, por muy eficaz que sea dicha aplicación, no le da el valor que tiene. El buen diseño de las interfaces de usuario tiene sin dudas, un impacto revelador en el éxito del software construido. [6]

Si la UI está bien diseñada, el usuario encontrará la respuesta que espera a su acción de una manera fácil y sin necesidad de estar acudiendo a los manuales o la ayuda que brinda el sistema. La mayoría de los sistemas informáticos son utilizados por distintos tipos de usuarios en los que cada uno tiene diferentes puntos de vista y niveles de conocimientos por lo que para algunos puede ser muy sencillo para otros resulta algo complejo. Por lo que no existe una interfaz válida para todos los usuarios y tareas, entonces es muy importante tener en cuenta para crear una aplicación los requerimientos de los usuarios finales.

En el caso de los laboratorios virtuales se va a utilizar una interfaz que simule exactamente como son los instrumentos que va a utilizar en el mundo real, así como la utilización de los distintos componentes, esto le va a permitir al usuario no estar alejado del objetivo para lo cual se crean estos sistemas. La interfaz de un laboratorio virtual debe ser capaz de imitar al máximo tanto en apariencia como en funcionalidad.

1.3.1. Interfaz de usuario (UI).

La Interfaz de Usuario de un programa es un conjunto de elementos de hardware y software de una computadora que presentan información al usuario y le permiten interactuar con la información. También se puede considerar como partes de las UI la documentación (manuales, ayuda, referencia, tutoriales) que acompaña al hardware y al software. [7]

La interfaz de usuario es el medio con que el usuario puede comunicarse con una máquina, un equipo o una computadora, y comprende todos los puntos de contacto entre el usuario y el dispositivo, normalmente suelen ser fáciles de entender y fáciles de accionar. [8]

La interfaz de usuario forma parte de una conexión e interacción entre hardware, software y usuario, es decir como la plataforma o medio de comunicación entre usuario o programa. [9]

Otras definiciones:

"Las interfaces gráficas de usuario son aquellas que incluyen cosas como menús, ventanas, teclado, ratón y algunos otros sonidos que la computadora hace, en general, todos aquellos canales por los cuales se permite la comunicación entre el hombre y la computadora." [10]

"…tipo de entorno que permite al usuario elegir comandos, iniciar programas, ver listas de archivos y otras opciones utilizando las representaciones visuales (iconos) y las listas de elementos del menú…"

"…es aquella parte de un programa que comunica al usuario con el programa mediante representaciones gráficas…" [11]

Finalmente si se analizan las definiciones y conceptos anteriormente tratados sobre el término *interfaz de usuario* se puede determinar que una interfaz de usuario es la parte de la aplicación o sistema informático que da la primera impresión a la vista del usuario y con la que interactúa directamente en espera de una respuesta del programa. La misma actúa como un sistema comunicador entre el programa y el humano como si fuese un traductor entre ambos, mediante menús, iconos y botones el usuario dirige el funcionamiento de dicho sistema. En todo momento el usuario debe de estar al tanto de lo que está ocurriendo con el programa, por tanto la interfaz le debe de brindar a la persona toda esta información, en tiempo real.

1.4. Tipos de interfaz de usuario:

Dentro de las UI se pueden distinguir básicamente dos tipos*: interfaz de hardware* e *interfaz de software.* El primer tipo es el encargado de definir los dispositivos externos usados insertar, dar tratamiento y finalmente entregar los datos ejemplo: el teclado. El segundo está destinado a entregarle al usuario información sobre los procesos y las herramientas de control, mediante lo que el usuario está viendo en la pantalla del ordenador.

Al iniciar cualquier sistema el usuario lo primero que ve e interactúa es con la interfaz. En general las aplicaciones desarrolladas para uno u otro sistema operativo siguen y utilizan los mismos componentes visuales que estos sistemas proporcionan por lo tanto el usuario se siente en un ambiente muy familiar que ya conoce.

Los programas o sistemas tienen varios tipos de interfaz de usuario, entre los que podemos citar:

1.4.1. Interfaz de línea de comandos.

Es una interfaz simple la que requiere solamente que el usuario introduzca la instrucción a través del teclado. Los usuarios más experimentados afirman que es mucho más simple, ágil y la información que da sobre lo que está ocurriendo con el sistema es mucho más detallada que la que brinda cualquier interfaz gráfica. (Ver anexo 1)

1.4.2. Interfaz controlada por menús.

Este tipo de interfaz de usuario muestra un conjunto de menús para realizar las distintas órdenes y operaciones que van a ser ejecutados posteriormente por los usuarios, lo que tiene como ventaja que a diferencia de la interfaz basada en líneas de comandos, el usuario no tiene necesidad de memorizar todas las ordenes. (Ver anexo 2)

1.4.3. Interfaz gráfica del usuario (GUI - Graphical User Interfaces).

En este tipo de interfaz, los usuarios son capaces de controlar y realizar las operaciones que desea a través de gráficos, botones e iconos presentados en la pantalla los cuales que representan las características del programa, solamente haciendo clic sobre ellos. Esta interfaz está basada en el hecho de que las personas reconocen con más facilidad y rapidez las representaciones gráficas, que las palabras o frases que puede leer dentro de una aplicación. Las GUIs cuentan con un número de características que las diferencian de los demás tipos de interfaces. Las mismas permiten el la transferencia de información hacia otros programas, Le brinda una respuesta visual a las órdenes dadas por los usuarios. Además permite que se manipulen directamente los objetos en la pantalla.

Es imposible afirmar que un tipo interfaz es mejor que otra ya que todas cuentan con ventajas y desventajas. Las ventajas de una interfaz basadas en menú son evidentes de cara al usuario que se acerca por primera vez a una aplicación, pero según el usuario pasa el tiempo trabajando con el programa y va adquiriendo destreza con el uso del mismo comienzan a aparecer los inconvenientes y poco a poco se vuelve algo tedioso tener que acudir a cada uno de los menús para realizar alguna acción. En general se puede decir que una interfaz basada en comandos es mucho más difícil de aprender al principio pero luego se vuelve más ágil su utilización. Teniendo en cuenta lo mencionado anteriormente muchos de los desarrolladores de interfaces de usuario hacen una mezcla de estos tipos de interfaces, dándoles así a los usuarios la posibilidad de trabajar alternativamente con cualquiera de las variantes. (Ver Anexo 3)

1.4.4. Interfaces adaptativas.

También existen las conocidas como ***"Interfaces adaptativas"*** como su nombre lo indica son aquellas interfaces de usuario que son capaces de adaptarse a diferencias o cambios que existen o pueden tener lugar en la población de usuarios de un sistema informático. [12]

Cuando se menciona la palabra "*diferencias*" se está refiriendo a las que existen entre distintos usuarios en un momento dado teniendo en cuenta el idioma que hablan, su posición geográfica etc., mientras que cuando se habla de los "*cambios*" se refiere a los que se producen para un mismo usuario con el paso del tiempo (ej. usuario que pasa de ser "novel" a "experto"), entre otras razones por las cuales es necesario desarrollar interfaces adaptativas.

Para lograr el desarrollo exitoso de una interfaz adaptativa se deben de tener en cuenta tres preguntas fundamentales:

¿Qué?

El desarrollador debe ser capaz de determinar primero cuales o que aspectos de la interfaz van a ser adaptables en caso de la ocurrencia de algún cambio. Por ejemplo la presentación de las entradas al sistema.

¿Cuándo?

Es importante determinar el momento en que se van realizar dichas adaptaciones. Lo cual puede ser durante el proceso de instalación (ej. Escoger el idioma) o durante las sesiones, es decir, a través de la petición del usuario en caso de que solicite algún cambio y por ultimo entre las sesiones donde se tiene en cuenta la información recogida en las sesiones anteriores.

¿Cómo?

Para desarrollar una interfaz adaptativa hay que tener en cuenta los diferentes métodos y técnicas a utilizar para poder que estas adaptaciones sean lo más efectivas, y que a la larga no causen un problema mayor, entre las que podemos utilizar encontramos: *La selección* la cual consiste en que el usuario es el encargado de seleccionar algunas opciones predeterminadas; *la habilitación* es otro método que consiste en marcar y desmarcar algunos componentes; *la reconfiguración* es aquella que consiste en modificar la interfaz

usando componentes predefinidos y *la edición* que no es más que adaptar la aplicación sin ningún tipo de restricción para esto se puede utilizar un lenguaje de programación.

1.5. Diseño de una interfaz de usuario.

El diseño de interfaz de usuario que va a ser seleccionado para una aplicación se debe de ajustar principalmente a los usuarios que van a interactuar con dicho sistema, y a las tareas u órdenes que pueden ser ejecutados por los mismos. El diseño de la interfaz de usuario debe permitirle además que los que van a interactuar de alguna manera con la aplicación no tengan que acudir a los manuales para realizar cualquier tarea, sino que muestre una curva de aprendizaje lo más cercana al valor nulo posible, o sea que el usuario sea capaz de aprender a utilizar el programa con solo mirar la interfaz que se le propone.

1.5.1. Etapas del desarrollo de la UI

Actualmente, el proceso del desarrollo de una interfaz se concibe como un ciclo que consta de 4 etapas. [7]

Diseño: En esta etapa se analizan los requerimientos del software que se va a desarrollar, se analizan las tareas a realizar y se general las posibles metáforas.

- Implementación: Es aquí donde se generan los prototipos y se desarrolla la aplicación.
- Medición: Se realiza la prueba piloto y los test con los usuarios.
- Evaluación: Análisis de los datos, elaboración del informe de resultados y recomendaciones. Se hacen comparaciones con estándares internos y/o externos, con otras versiones del producto y con otros productos semejantes, luego se generan las diferencias y se crean nuevas metas.

1.5.2. Principios para el diseño de una interfaz de usuario:

Según "Hammer 83" Una interfaz debe cumplir los siguientes principios:

- *Naturalidad.* El nuevo sistema o programa que se va a desarrollar debe ser lo más similar al antiguo. Se dice que una interfaz es natural cuando provoca al usuario sentimientos de "sentirse como en su casa".

- *Ser fácil de aprender y usar.* Todo sistema debe proporcionar al usuario una ayuda potente, en el cual se le explique al mismo la forma de trabajar con el programa, y

todas las vías posibles para resolver cualquier problema. Se debe de incorporar: Administración de perfiles de usuarios, para que según el grado del usuario el programa ejecutará las distintas acciones. Los mejores sistemas de ayuda son los llamados “sensibles al contexto”. Es capaz de determinar la circunstancia que origina la petición de la ayuda y dar una respuesta concreta.

Otros principios de diseños muy utilizados son:[13]

Anticipación: Una interfaz de usuario debe de algún modo intentar anticiparse a las necesidades más generales del usuario, lo que evita demoras en aspectos en los que se puede agilizar el tiempo de uso de una aplicación o de encontrar alguna opción sencilla.

Autonomía: La computadora, y la UI deben estar a disposición del usuario y brindarle un ambiente flexible para que pueda aprender a usar rápidamente la aplicación pero el entorno de trabajo debe tener ciertas cotas, o sea explorable pero no azaroso. Es importante tener en todo momento a los usuarios alertas e informados, esta información además debe brindarse en ubicaciones fáciles de visualizar y mostrando el estado del sistema en todo momento.

Percepción del color: Aunque cuando se desarrolla una Interfaz de Usuario se utilicen convenciones de colores hay que tener en cuenta también mecanismos secundarios para proveer la información a usuarios con trastornos en la diferenciación de los colores.

Consistencia: La interfaz de usuario debe mantener una uniformidad en cuanto al vocabulario, estilo, colores etc.

La ley de Fitt: Tanto en las UI para aplicaciones Web, como para aplicaciones de Escritorio. El tiempo en que un usuario se demora en alcanzar el objetivo está en función de la distancia que tiene que recorrer para alcanzarlo y el tamaño de dicho objetivo. Por lo que las opciones importantes en una UI deben establecerse utilizando objetos grandes

Interfaces Explorables: Siempre que la aplicación o el programa lo permitan hay que brindarle al usuario la posibilidad de salir del programa dejando un rastro del avance que ha ido alcanzando y que pueda abrir en otro momento para seguir con el desarrollo de su trabajo. Se debe tener además la posibilidad de una orden “Deshacer”, la que le brinda al mismo un apoyo a explorar el sistema sin temores.

Uso de metáforas: Las metáforas son figuras que mentalmente son muy fáciles de recordar. Una UI puede contener objetos que le recuerden al usuario de una manera visual, con sonido u otra característica fácil de percibir por el usuario algún elemento conceptual.

Reducción de la Latencia: Siempre que sea posible es necesario el uso de tramas que permiten colocar la latencia a un segundo plano invisible al usuario, esto es posible gracias a la multitarea.

Protección del trabajo: Se debe asegurar que el usuario nunca pierda el trabajo que está realizando, en caso de cualquier problema, error del sistema en su manipulación o de energía.

Legibilidad: La información mostrada por el sistema debe ser fácil de ubicar y leer, para esto el texto debe de tener un alto contraste utilizar combinaciones tales como texto negro sobre fondo blanco o amarillo suave. Evitar la presentación excesiva de objetos y elementos dentro de la UI.

1.5.3. Buenas prácticas para el diseño y desarrollo de interfaz de usuario.

A la hora de desarrollar una interfaz de usuario, hay que tener en cuenta ciertas normas y reglas a cumplir para que el resultado final sea una aplicación bastante cómoda y que agilice el trabajo del usuario. Aunque no existen reguladores para el desarrollo de las mismas si hay algunas reglas provenientes del sentido común de las personas desde cuestiones bastantes obvias hasta detalles que apenas se pueden ver a simple vista.

Los estándares para el desarrollo de las UI se pueden encontrar divididos en tres grandes grupos. Uno de ellos, el primero se basa en la *interacción* con el usuario y lo que hace es tratar de encontrar una armonía comunicacional con el usuario o grupo de usuarios que va a utilizar la aplicación.

El segundo grupo "*presentación*" está orientado a optimizar la forma en que el sistema le muestra los datos al usuario, intentando darle a conocer al usuario en todo momento en que lugar se encuentra parado.

Y un tercer grupo que se le llama la *estructura* del sistema, el cual intenta optimizar estructuralmente toda la interfaz del usuario contribuyendo a que la interfaz sea navegable por el usuario.

Aunque se ha intentado estandarizar la interfaz, enmarcadas siempre en el usuario final, ya que este es realmente el que va a interactuar con las aplicaciones.

Interacción.

Cuando se habla de *interacción* se está refiriendo a la comunicación del usuario con la computadora mediante el uso de varios controles, elegir un control no es solo cuestión de seguir al pie de la letra una receta, las siguientes normas o buenas prácticas son necesarias para lograr una correcta interacción en una aplicación que se esté desarrollando.

Botones de comando: Los botones constituyen dentro de los cuadros de diálogos la principal manera que tiene el usuario para tomar alguna que otra medida. El límite de botones a presentar en un diálogo es de 6 como máximo, el texto que muestran, tienen que ser lo más claro y conciso posibles, solo se debe utilizar una palabra para el etiquetado del botón. La primera letra debe de comenzar con letra inicial mayúscula. Algunas etiquetas se han convertido en estándares para las aplicaciones como por ejemplo: *Aceptar, Cancelar, Ayuda. (Ver Anexo 4).*

Botones de opciones (radio buttons): Los botones de opciones son los llamados "*radio buttons*" los cuales son utilizados para cuando se valla a escoger una opción entre varias que brinda la aplicación. No se deben usar más de 6 opciones. Los radios pueden ordenarse de las siguientes maneras:

1. *Por frecuencia,* Las opciones con que más frecuentes los usuarios utilizan, en la parte superior.

2. *Por tareas*, seguir el orden usual en que se realizan.
3. *Por lógica*, si existe un orden lógico entre las opciones, ejemplo fechas.
4. *Por orden alfabético.*

Casillas de verificación (check box): Si lo que se desea es escoger alguna opción de ***si o no,*** en lugar de usar botones de acción mejor utilizar un casilla de verificación, el cual se activa o se desactiva. En caso de que el programador anticipe que el usuario pueda o no seleccionar todas las opciones o deseleccionarlas es más recomendable utilizar una lista de chequeo y no incluir dentro del grupo de casillas de verificación estas opciones.

Cajas de texto: Las cajas de texto constituyen la principal vía mediante el cual el usuario ingresa información mediante el teclado a una aplicación, el tamaño de las cajas de textos debe de indicarle al usuario la cantidad aproximada de caracteres que puede teclear dentro de ella. La alineación del texto debe ser hacia la izquierda. Cada cuadro de texto debe tener asignada una etiqueta que muestre el contenido que el usuario debe teclear, en las que no se debe utilizar abreviaturas.

Presentación.

En una interfaz de usuario la presentación está referida a la manera y la cantidad en que se le muestran los datos en las ventanas al usuario, es decir lo que el usuario tiene que ver. Una visualización adecuada de la información puede generar una diferencia muy grande en la utilidad que este le brinda a la interfaz.

Formularios: Los nombres de los formularios deben de ser únicos de una manera lo más descriptiva posible del contenido que tienen, no deben de tener títulos adicionales. Es necesario organizar las ventanas y cuadros de diálogos de manera tal que siga el flujo de lo que el usuario realiza. Cada ventana o cada cuadro de diálogo deben presentar una única tarea o sub-tarea en el flujo de trabajo del usuario.

Encontrar un lugar de residencia: En una aplicación el usuario siempre debe de tener un punto de partida que sirva como base para todas las operaciones que va a realizar dentro de la misma. La base principal de una aplicación no necesariamente es la primera ventana que se muestra en una aplicación El desarrollador de la interfaz también puede usar la variante de crear simplemente una pantalla en blanco con una barra de menús.

Organización de la información dentro de una ventana: Toda la información que se va a mostrar dentro de una ventana o un cuadro de diálogo debe organizarse de manera tal que fluya bien la tarea que el usuario hacer. Colocándose la información más común o critica en la parte superior izquierda de la ventana, manteniendo un flujo de arriba hacia abajo o izquierda a derecha.

Fuentes: Las fuentes más apropiadas para mostrar dentro de una aplicación con UI es ***Sans Serif*** para el texto y las etiquetas porque son mucho más fáciles de leer en la pantalla. ***Time New Roman y Arial*** son dos tipos de letra ***serif***. El uso de letra cursiva o subrayada puede crear dificultades en la lectura de los textos, por algunos usuarios. Siempre se debe de evitar el uso de fuentes de color, es mucho más fácil leer las letras en negro con el fondo blanco o gris, usar la **“negrita”** para resaltar.

Se debe evitar también el cambio del tamaño de la fuente. Como tamaño estándar de la fuente es necesario utilizar por lo menos 8 puntos. Por todos los medios se debe de tratar de utilizar en una ventana un solo tipo de fuente.

Colores: En la actualidad las UI brindan la posibilidad de utilizar varios colores, pero una mala decisión de colores puede en vez de mejorar la usabilidad, distraer la atención del usuario.

El color dentro de una aplicación debe de ser utilizado para llamar la atención al usuario, pero moderadamente o pierde su eficacia. No es recomendable el uso del color solo para fines estéticos. El desarrollador de interfaz de usuario debe de tener en cuenta también el significado cultural de los colores para algunos usuarios. (Ver Anexo 5).

Evitar las combinaciones de rojo y azul, el azul en el texto. En caso de que se deje al usuario configurar sus propios colores hay que asegurar que este encuentre una manera fácil de restablecer los valores iniciales.

Diseño o elección de gráficos: En un programa con UI una imagen puede transmitir más ideas que mil palabras, siempre y cuando esté bien diseñado orientado a la idea que se quiere mostrar al usuario. Además el uso de gráficos puede sustituir la traducción de algunas órdenes a otros idiomas. Sin embargo, hay algunas pautas a tener en cuenta a la hora de seleccionar los gráficos, debe asegurarse de que los gráficos son entendibles por los usuarios; no utilizar gestos ofensivos; utilizar representaciones reconocidas internacionalmente.

Una vez encontrado el diseño más apropiado se debe utilizar de forma consistente. Se debe considerar también el cambio en las imágenes según el estado en que se encuentra el programa, por ejemplo si se tiene una imagen con una computadora apagada y en algún momento esta se inicia, debe de darse la sensación de que está activada.

Estructura.

La estructura de una UI se refiere a las ventanas, cuadros de diálogos y menús que se utilizan para crear su interfaz.

Cuadros de diálogos: Los cuadros de diálogos se utilizan para que los usuarios completen alguna información específica dentro de los sistemas. Se le debe dar la oportunidad a los

usuarios de abrir los cuadros de diálogos de forma *¨modal¨* lo que les permite continuar con su trabajo ya que este tipo de cuadro de diálogo se mantiene abierto y activo.

Etiquetas (Tabs): Las etiquetas son frecuentemente utilizadas en casi todas los nuevos prototipos de interfaces de usuarios, usados para mostrar información independiente. Suelen ser muy útiles aunque también tienen sus inconvenientes, se utilizan principalmente para categorizar alguna información muy concreta. No es recomendable el uso de las etiquetas cuando el desarrollador tiene que mostrar un grupo engorroso de información porque al final terminan quedando desordenadas.

Menús: Los menús juegan un papel crítico dentro de las UI, constituyen un método de navegación dentro de la aplicación, que transmite el modelo mental del usuario. Los nombres tienen que ser bien escogidos para asegurar que tengan sentido de lo que se quiere transmitir al usuario.

Los elementos dentro de los menús representan opciones individuales, los cuales pueden presentar textos, gráficos o combinar, es necesario siempre mostrarle al usuario alguna indicación visual acerca de los elementos del menú, en caso de existir algún elemento del menú que el usuario no tenga que usar, lo que se aconseja es desactivar el uso del mismo. Si todos los elementos del menú están desactivados entonces se debe desactivar el título del menú.

Barra de menú: La barra de menú se encuentra en la parte superior de la aplicación, apunta hacia las funcionalidades principales de la aplicación. Los títulos se deben elegir y organizar cuidadosamente de forma que coincida con el flujo de trabajo de los usuarios. Los elementos frecuentes se deben evitar agrupar en una sola categoría. Para nombrar los títulos se usa una sola palabra. El título de un menú no ejecuta ninguna acción. Si hay un gran número de elementos es necesario ubicarlos en forma de cascadas, aunque estas no deben exceder de dos niveles.

Para algunos elementos del menú que son de mucho uso se recomienda el uso de "*atajos de teclado*", para evitarle al usuario tener que acudir siempre a las opciones del menú.

Barras de herramientas (Tool bar): Se utilizan como un menú contextual o como una manera de representar de manera gráfica algunas opciones que a veces no se puede expresar con palabras lo que se quiere transmitir al usuario. Las barras de herramientas se deben utilizar de manera consistente.

Al usuario tiene que tener la oportunidad de mover alguna de estas barras en diferentes lugares dentro de la pantalla de la aplicación, permitir además cambiar o esconder las barras de herramientas.

Hay que prestar mucha atención a la cantidad de botones que se va a mostrar en una barra de herramientas, demasiados botones puede crear alguna tensión dentro de la vista del usuario y puede que no vean alguno de los botones que contiene dicha barra, si el programador tiene la posibilidad debe incluir herramientas de señalización (Tool Tip) en las acciones de la barra de herramientas.

1.6. Bibliotecas gráficas para el desarrollo de la UI.

Para el desarrollo de la una UI existen en la informática numerosas bibliotecas que posibilitan al desarrollador elaborar una coherente interfaz de usuario, y vincularlas con sistemas que utilicen visualización de componentes en 3D, entre las que se encuentra:

1.6.1. CEGUI

Es una biblioteca escrita en C++. Diseñada en particular para satisfacer las necesidades de los juegos. La principal fuerza de la CEGU consiste en que es altamente configurable. Dicho sistema no carga directamente archivos, para exhibir directamente el texto, lo que hace es interconectar estos con el código definido por el usuario, aunque dentro del núcleo de CEGUI existe un número de módulos que permiten usar ciertos componentes y bibliotecas. Esto le permite al usuario que utilice esta biblioteca pueda utilizarla en cualquier ambiente de funcionamiento. La representación es lograda por módulo back-end, y además incorpora módulos para Diret3D, OpenGL, Motor de OGRE 3D y Motor de Irrlicht. La CEGUI trae consigo un sistema bastante razonable de widget, y es compatible además con Lua sripting el cual puede ser utilizado para definir ciertas relaciones de varias ventanas, widgets y la manipulación de acontecimientos dentro de la misma ventana.

1.6.2. GTK+

Consiste en un grupo de bibliotecas multiplataforma, que se utilizan específicamente para el desarrollo de interfaces de usuario, son mayormente utilizadas para los entornos GNOME, ROX y XFCE, aunque también se han desarrollado para ser usadas en otras plataformas como Windows, MacOS y otros. [14] Utiliza como lenguaje de programación base el C, también se pueden desarrollar programas en C++, Java, C#, Ruby, Perl o Python.

Está escrita sobre licencia LGPL, siendo totalmente libre y de código abierto. Presenta iconos predefinidos, los *widget* que contiene soportan *rendering* personalizados y además que puede permitir la definición de aceleradores.

GTK+ se basa en varias bibliotecas del equipo de GTK+ y de GNOME: [15]

Glib. Biblioteca de bajo nivel estructura básica de GTK+ y GNOME. Proporciona manejo de estructura de datos para C, portabilidad, interfaces para funcionalidades de tiempo de ejecución como ciclos, hilos, carga dinámica o un sistema de objetos.

- **GTK.** Biblioteca la cual realmente contiene los objetos y funciones para crear la interfaz de usuario. Maneja *widgets* como ventanas, botones, menús, etiquetas, deslizadores, pestañas, etc.

- **GDK.** Biblioteca que actúa como intermediario entre gráficos de bajo nivel y gráficos de alto nivel.

- **ATK.** Biblioteca para crear interfaces con características de una gran accesibilidad muy importante para personas discapacitadas o minusválidas. Pueden usarse utilerías como lupas de aumento, lectores de pantalla, o entradas de datos alternativas al clásico teclado o ratón.

- **Pango.** Biblioteca para el diseño y renderizado de texto, hace hincapié especialmente en la internacionalización. Es el núcleo para manejar las fuentes y el texto de GTK+.

- **Cairo.** Biblioteca de renderizado avanzado de controles de aplicación.

La GTK+ ha sido utilizada para crear numerosas aplicaciones incluso muy conocidas en el mundo de la informática, tales como: El navegador de internet Firefox, Chromium, la herramienta de diseño 3D K3D, el editor de imágenes GIMP, cliente de correo Evolution, el Pidgin que es un cliente de mensajería instantánea entre otros.

1.6.3. Qt.

Es un framework muy útil para desarrollar aplicaciones de escritorio, entre sus características principales presenta; es multiplataforma, aunque está escrito en C++ es posible utilizarlo con otros lenguajes mediante el uso de blindings, entre los que se

encuentran el C#, Python, PHP, Ruby entre otros. Proporciona la mayoría de las entidades gráficas que verá empleadas en una aplicación KDE: menús, botones, regletas, etc. Las aplicaciones basadas en Qt tienen una buena respuesta y un uso aceptable de la memoria. [16] En un principio de su creación solo se usaba para crear UI; en estos momentos existen bibliotecas para el tratamiento de Bases de datos, XML, OpenGL, multimedia etc.

Ingresa al C++ algunas características nuevas como los ciclos *foreach, sentencias forever e introspección.* Su origen tuvo lugar en 1991 y solo estaba disponible para plataformas Windows y X11. En 1994 se forma la compañía Trolltech, al principio con el nombre de Quasar Technologies. Ya en el 2001 se le agrego el soporte para Mac OS X, en el 2005 fue liberado bajo licencia GPL y posteriormente en el año 2008 Nokia adquiere el proyecto Trolltech. En estos momentos la compañía Canonical está utilizando estas bibliotecas para el desarrollo de su nieva interfaz de usuario *Unity 2D.*

Qt se ha utilizado en el desarrollo de numerosas aplicaciones muy conocidas en todo el mundo tales como: Google Earth, Adobe Photoshop Album, VirtualBox, VLC media player y muchas otras más.

1.7. Lenguaje de programación.

Un lenguaje de programación es un dialecto artificial trazado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden utilizarse para crear sistemas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Constituyen además una técnica de comunicación mediante el cual el programador le entrega las instrucciones al computador. Pueden clasificarse teniendo en cuenta distintos criterios: lenguajes interpretados, lenguajes compilados, lenguajes de script entre otros, estas características dividen a los lenguajes de programación en dos grupos, declarativos e imperativos. Para desarrollar interfaz de usuario UI, existen en el mundo un conjunto de lenguajes de programación tanto para plataformas GNU/Linux, como para Windows, entre los más frecuentes de encontrar están: C++, Java, C# y Python. Para elegir alguno de estos lenguajes es importante tener en cuenta las características de la aplicación que se quiere desarrollar, teniendo en cuenta las ventajas y a facilidades que brinda cada lenguaje particularmente.

1.7.1. C++.

El C++ tiene su origen en 1980 y su nombre fue propuesto por Rick Mascitti es un lenguaje de programación resultado de la extensión del C, agregándole la característica de que permite la manipulación de objetos, luego se le agregaron facilidades de programación genérica, una de las posibilidades que brinda el C++ como lenguaje es que permite la sobrecarga de operadores y crear nuevos tipos. Otras de las ventajas que presenta son:

- Muy potente y robusto, por lo que se utiliza mucho para creación de sistemas complejos, desde programas “Hello World” hasta Sistemas Operativos.
- Puede compilar y ejecutar código de C.
- Tiene una gran cantidad de documentación en internet códigos de ejemplo.
- Es un lenguaje muy portable por lo que se puede utilizar en varias plataformas de desarrollo.
- Es muy rápido en ejecución.

También presenta algunas desventajas señaladas a continuación:

- El uso de DLLs muy complejo.
- Manejo de punteros y memoria, es una ventaja porque permite un mejor control de la memoria, pero la inexperiencia de los desarrolladores o la pérdida de costumbre puede conllevar a un desastre.
- No es recomendable para el desarrollo de la Web.

1.7.2. Python.

Python es un lenguaje de programación de alto nivel el cual provee un código muy limpio y fácil de leer. Es un lenguaje de programación que soporta programación imperativa, orientada a objeto y en menos grado programación funcional. Es multiplataforma y su licencia es de código abierto entre las características que lo distingue se pueden mencionar: [17]

Resolución dinámica de nombres; es decir, lo que enlaza un método y un nombre de variable durante la ejecución del programa (también llamado ligadura dinámica de métodos).

- Facilidad de extensión. Se pueden escribir nuevos módulos fácilmente en C o C++. Python puede incluirse en aplicaciones que necesitan una interfaz programable.
- Los tipos de datos en Python permiten expresar operaciones más complejas en sentencias cortas.
- El agrupamiento de sentencias se realiza por identación en lugar de llaves de principio y fin
- No se necesita de la declaración de variables.
- Python es extensible por lo que puede ser utilizado su lenguaje en extensiones de C.

1.8. Entorno de desarrollo Integrado (IDE).

Es un programa o sistema informático que está integrado por un grupo de herramientas de programación, un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Puede ser específicamente para un solo lenguaje de programación o varios al mismo tiempo. Proveen un marco de trabajo muy amigable y fácil de configurar para numerosos lenguajes de programación, puede funcionar también como un sistema en tiempo de ejecución lo que permite utilizar el lenguaje de manera interactiva, independiente del trabajo orientado a archivos de texto.

Existen varios IDE los cuales pueden ser usados para el desarrollo de los Laboratorios Virtuales entre ellos se encuentran: Eclipse, C++ Builder 6, Qt Creator, y CodeBlock.

1.8.1. Qt Creator.

Qt Creator es un IDE creado por Trolltech para el desarrollo de aplicaciones con las bibliotecas Qt, requiriendo su versión 4.x. Los sistemas operativos que soporta en forma oficial son:

- GNU/Linux 2.6.x, para versiones de 32 y 64 bits con Qt 4.x instalado. Además hay una versión para Linux con gcc 3.3.
- Mac OS X 10.4 o superior, requiriendo Qt 4.x
- Windows XP y Vista, requiriendo el compilador MinGW y Qt 4.4.3 para MinGW.

1.8.2. Microsoft Visual Studio .

El Microsoft Visual Studio (VS) es un IDE, que solo existe para Windows, fue creado por Microsoft Corporation y soporta varios lenguajes de programación, tales como: ASP.NET,

Visual J#, Visual C++, Visual C#. En la actualidad se le están agregando extensiones para muchos otros más. Es un IDE que permite a los desarrolladores crear sitios web, aplicaciones para escritorio y para dispositivos móviles. Desde el lanzamiento de la versión 6.0 ha alcanzado un gran desarrollo, brindándole en cada versión más facilidades al programador. El VS 2010 es la versión más reciente la cual ya presenta herramientas para la creación de aplicaciones para Windows 7, entre las características que más se destacan es la capacidad de utilizar más de un monitor, y compila las características de todas las ediciones comunes, Professional, Team Studio, Test, conocida como Visual Studio Ultimate.

El VS le permite al desarrollador:

- Programar sus aplicaciones disfrutando de un entorno altamente productivo, que además cuenta diseñadores gráficos.
- Desarrollar y depurar aplicaciones multicapa de servidor desde un mismo entorno unificado de desarrollo.

Es un IDE que le brinda una interfaz amigable al desarrollador pero está bajo licencia privativa, por lo tanto se ve restringido su uso.

1.9. Metodología de desarrollo.

Todo desarrollo de software es riesgoso y difícil de controlar, pero si no se lleva una metodología de por medio, lo que obtenemos es clientes insatisfechos con el resultado y desarrolladores aún más insatisfechos. [18] Las metodologías surgen para solucionar el problema que existe entre los encargados de definir y analizar los problemas y los responsables en darle solución. Su principal función es establecer un equilibrio entre los integrantes de un proyecto, de manera que se encuentre una solución más eficaz que cumpla con todos los requerimientos y necesidades de los clientes, con un buen tiempo de entrega, teniendo en cuenta los recursos con los que dispone el equipo de desarrollo.

1.9.1. RUP (Rational Unified Process).

El proceso unificado conocido como RUP, es una metodología que permite el desarrollo de software a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad. [19] Modela visualmente el software empleando el estándar UML. Verifica la calidad de los productos del software asegurando que cumple los estándares. El ciclo de vida de RUP se caracteriza por ser: [20]

Dirigido por Casos de Uso: Los casos es una manera muy interactiva de mostrarle al usuario que va a utilizar la aplicación lo que desean, estos son captados en el momento que se modela el negocio con los clientes (usuarios), y son representados mediante los requerimientos del software, además son los encargados de guiar el proceso de desarrollo ya que como resultado de cada uno de los flujos se obtienen distintos modelos que demuestra el desarrollo del software como tal.

Centrado en la arquitectura: Esta es otra de las características de RUP, que es descrita mediante diferentes vistas, e incluye todo lo referente a aspectos estáticos y dinámicos que son más significativos en el sistema, cada producto tiene una función y una forma de realizarla, ninguna es suficiente por si sola.

Iterativo e incremental: RUP plantea que el trabajo puede ser dividido en mini-proyectos que consisten en varias partes más pequeñas consideradas como iteraciones que resultan en un incremento. Las iteraciones son una secuencia lógica de actividades que responden a un plan y a criterios de evaluación, estas traen como resultado una versión del software.

1.9.2. DSDM (Dynamic System Development Method).

Fue creada en enero de 1994, su objetivo fue crear una metodología RAD (Rapid Application Development) unificada. Plantea la necesidad de que el equipo de trabajo y los usuarios trabajen de manera conjunta para de esta manera evitar que se desarrollen sistemas que una vez terminados; no cumplan con los requerimientos, no funcionen correctamente o caigan en desuso. Es un proceso iterativo e incremental.

Los principios fundamentales de DSDM son: [21]

La participación directa del usuario en el desarrollo del software.

- El equipo de desarrollo puede tomar decisiones dentro del ciclo de desarrollo.
- El producto se entrega frecuentemente a medida que se van obteniendo resultados.
- Se ajusta muy bien a los objetivos del software.
- Los cambios realizados son reversibles.
- Es capaz de levantar requerimientos globales.
- Durante todo el ciclo de vida presenta pruebas integradas.

1.9.3. Programación Extrema (Extreme Programing, XP).

XP11 es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. [22]

Los fundamentos de XP son provenientes del sentido común de las personas pero llevadas al extremo, de ahí es donde sale su denominación. Esta metodología define los siguientes roles: Programador, Cliente, Encargado de pruebas, Encargado de seguimiento, entrenador, consultor y gestor.

El éxito final de un proyecto sucede cuando el cliente realiza una selección de lo que va a implementar, basado en la habilidad que tiene el equipo. El ciclo de vida de Xp consiste en:

1. El usuario define el valor de negocio a implementar.

2. El desarrollador se encarga d estimar el esfuerzo necesario, para realizar la tarea.

3. El usuario selecciona lo que finalmente va a construir.

4. El desarrollador construye ese valor de negocio.

5. Se vuelve al paso 1.

1.10. Lenguaje de Modelado.

A lo largo de los años, el desarrollo de los proyectos de software causan bastantes confusiones y malas interpretaciones en los requerimientos de los clientes y usuarios, en parte debido a la abundancia de notaciones, metodologías y conceptos que hace que los desarrolladores de sistemas no se pongan de acuerdo en que es lo que realmente están elaborando.

1.10.1. UML (Unified Modeling Language).

Lenguaje de modelado UML es un lenguaje gráfico mediante el cual se puede visualizar, especificar y documentar un software, ofrece un plano de descripción del sistema mediante el uso de modelo estándar. Incluye además otros aspectos conceptuales como el proceso de negocio y funciones que presenta el sistema, así como otros aspectos como son lenguajes de programación, alguna de sus expresiones, bases de datos, y componentes que pueden reutilizarse. [14]

Algunas de las propiedades de UML como lenguaje de modelado estándar son: [23]

- Concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividades actuales y futuras.
- Reemplaza a decenas de notaciones empleadas con otros lenguajes.
- Modela estructuras complejas.
- Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.
- Comportamiento del sistema: casos de uso, diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas.

1.11. Herramientas CASE.

Se puede definir a una herramienta CASE (en español, ingeniería de software asistida por computadora, en inglés, Computer Arded Software Engineering) como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante los pasos del ciclo de vida de desarrollo de un software. [24]
Las herramientas CASE son de mucha utilidad dentro del ciclo de vida de un proyecto de software, mediante las cuales se puede realizar todo el diseño del proyecto, cálculo de los costos, generación de códigos automáticos teniendo en cuenta el diseño creado por los ingenieros, documentación completa del desarrollo del sistema y la obtención de errores durante su desarrollo. Contribuyen además a la mejora de la planificación, aumento de la calidad del producto final, también permite que el software pueda ser reutilizado en futuros productos y en la estandarización de la documentación.

1.11.1. Visual Paradigm.

El *Visual Paradigm* está considerada como una herramienta muy eficiente y fácil de utilizar, es además multiplataforma, utiliza como lenguaje de modelado el UML. Y proporciona interoperabilidad con otras aplicaciones. Es muy sencilla también a la hora de instalarse y actualizarse, les ofrece a los usuarios la capacidad de desarrollar la ingeniería inversa y directa, está sustentada en varios idiomas.

Conclusiones generales del capítulo.

En este capítulo se realizó un estudio de los diferentes conceptos de interfaz de usuario, y los tipos de interfaz que existen, tomando de las mismas las ventajas de cada una. También se hace un recorrido por diferentes “buenas prácticas” que son de mucha ayuda y van a ser tomadas en cuenta para el desarrollo de los Laboratorios Virtuales, tales como tamaño y color de la fuente, distribución de la información dentro de un formulario y el uso de algunos *Widget* que van a ser utilizados durante el desarrollo de la interfaz. Se hace un análisis por el proceso de desarrollo de las interfaces de usuario, mediante el cual se obtiene una serie de tareas a realizar.

Finalmente se trataron aspectos fundamentales de las herramientas libres, utilizadas en el mundo para le creación de interfaces de usuario, haciendo un análisis de sus características, ventajas y desventajas, se identificó también el IDE de desarrollo a utilizar y el lenguaje de programación en el cual se va a desarrollar dicha interfaz. Todo esto va encaminado a obtener elementos necesarios que dan solución al objetivo de la investigación.

2

Capítulo 2. Características del sistema.

Introducción.

En este capítulo se dará una breve panorámica de la solución propuesta para el problema que encierra este trabajo de diploma, por lo que se abordarán temas sin los cuales no hubiese sido posible el desarrollo de la interfaz gráfica de usuario.

2.1. Solución propuesta.

La investigación tiene como propósito principal lograr el desarrollo de una interfaz de usuario, que sea adaptable a los Laboratorios Virtuales. Una interfaz de usuario es la parte del sistema con que el usuario interactúa y la que le brinda el acceso y el control a todas las funcionalidades e información que brinda la aplicación informática, por lo que el diseño tiene que estar orientado al tipo de usuario que va a interactuar con la misma, en este caso, por lo general son estudiantes, y su propósito es ver cómo se observan y se manipulan los componentes mostrados en la pantalla, no están interesados en código, es decir en el cómo sino en el qué hace.

Al analizar los distintos tipos de interfaz de usuario que existen, así como las buenas prácticas para el desarrollo de las mismas, para el desarrollo de la interfaz de usuario de los laboratorios virtuales, se utilizó la interfaz gráfica, debido a que en la misma está plasmada el uso de metáforas, que son de gran utilidad para transmitir con pocas palabras al usuario, lo que este debe de hacer en cada momento que trabaja con la aplicación. Además se hace uso de los menús y de las barras de herramientas, mediante los cuales se le indica al usuario las órdenes a desarrollar en el laboratorio virtual.

Las buenas prácticas tratadas en el capítulo anterior se tuvieron en cuenta para la manera de posicionar la información, menús y los botones dentro de la aplicación, estos muestran opciones individuales, y en caso de que alguno de estos elementos en un momento determinado del uso de la aplicación no se pueda utilizar, los mismos se muestran desactivados, y cada uno de los nombres de las órdenes fueron seleccionados para que el usuario encuentre de manera fácil, lo que necesita hacer.

Un aspecto relevante tenido en cuenta en el diseño de la interfaz de usuario fue la opinión del cliente, ya que para este es que se desarrollaron los laboratorios virtuales, y la opinión del usuario que va a utilizar el sistema es lo más importante para su éxito. Esto se tuvo en cuenta en aspectos como, los colores, la orientación y el lugar donde se colocaron los paneles, la tipografía utilizada y muy importante, las terminologías tratadas para cada uno de los elementos mostrados en la interfaz.

La propuesta de interfaz de usuario se muestra a continuación:

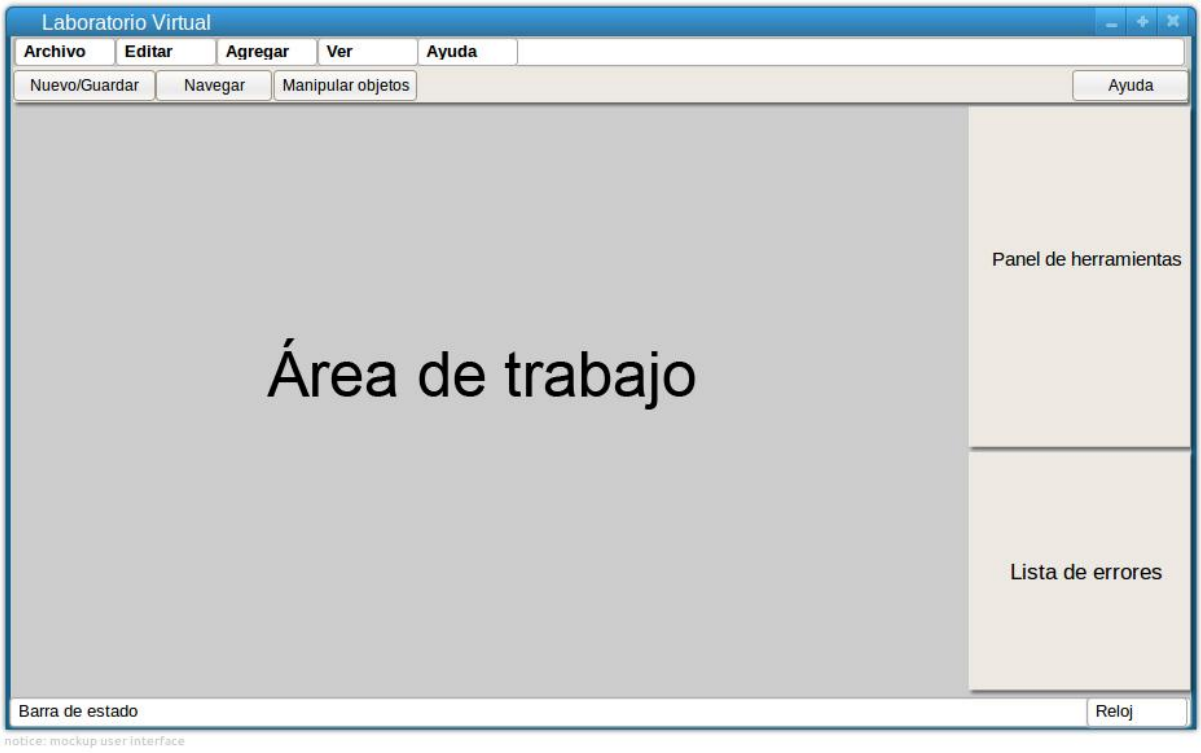


FIGURA 1. *ESTRUCTURA DE LA INTERFAZ GRÁFICA DE USUARIO.*

2.2. Funcionalidades.

La interfaz gráfica de usuario de los laboratorios virtuales cuenta con un conjunto de requisitos y especificidades funcionales o no que garantizan que se facilite la interacción con el usuario. Dentro de los requerimientos se encuentran los funcionales de ellos hay un grupo que son comunes para los tres laboratorios virtuales que se han desarrollado, como son por ejemplo, cargar las escenas de los laboratorios, que por lo general son funcionalidades que ocurren invisible a los usuarios, pues una vez que carga la aplicación y

el mismo determina el tipo de práctica (fase) que va a desarrollar, internamente la aplicación carga los modelos necesarios. El resultado final de todo lo realizado por el usuario dentro de la aplicación será registrado en un reporte.

En los tres laboratorios virtuales también los menús y los paneles se crean a partir de la fase o ejercicio seleccionado por el usuario, otra de las principales funcionalidades comunes con que cuentan los laboratorios virtuales es el manejo de la escena por parte del usuario, donde el mismo puede navegar por la dicha escena mediante el teclado o en algunas ocasiones utilizar elementos de la barra de herramientas (Tool Bar). Donde puede por ejemplo: Acercar y/o alejar la cámara, en caso de encontrarse perdido restablecer el lugar inicial donde se encontraba dentro de la navegación 3D. Lo primero que hace el usuario para usar los laboratorios virtuales es autenticarse en el sistema, y entonces luego este le permite realizar la práctica. En cada uno de los laboratorios virtuales, se manejan objetos en 3 dimisiones, los cuales son partes de la escena que se va creando a medida que el usuario desarrolla la práctica de laboratorio, la aplicación debe controlar además los errores que se cometen en la ejecución del ejercicio, y por último generar un reporte final.

Las funcionalidades de los laboratorios virtuales es necesario tratarlas de manera diferenciada, ya que no todos los laboratorios virtuales persiguen el mismo objetivo práctico. A continuación se dará un breve resumen de cada laboratorio virtual por separado, en qué consisten y cuáles son sus principales funcionalidades.

2.2.1. Laboratorio Virtual Arquitectura de Computadoras.

Este laboratorio virtual, como su nombre lo indica, consiste en que el estudiante debe ser capaz de armar una computadora, la misma cuenta con 2 fases: en la primera se ensamblan los elementos internos de una computadora, tales como: memoria RAM, micro-procesador, tarjeta madre, entre otros; la segunda fase tiene el mismo principio de funcionamiento pero con las partes externas de la computadora, ejemplo: mouse, teclado, los cables etc.

La principal funcionalidad de este laboratorio virtual es el manejo de objetos de la escena, mediante los cuales el usuario va conformando una computadora, con los elementos que contiene el panel de herramientas, independientemente de la fase en que se encuentre desarrollando el laboratorio virtual, y en común para todas se muestra un conjunto de herramientas que pueden ser utilizadas en el ensamblaje de la computadora, por ejemplo: destornilladores, pinzas y otros. El reporte que se genera contiene los dispositivos que se

conectaron en la computadora, los errores cometidos en cada dispositivo conectado y el tiempo en que se conectó finalmente la tarjeta madre dentro del case.

2.2.2. Laboratorio Virtual Diseño e Instalación de una Red Local (LAN).

Este laboratorio virtual está dividido en 3 fases, en cada una de ellas se persiguen diferentes objetivos; la primera fase, consiste en confeccionar un cable de red, para el cual se brindan un grupo de herramientas y finalmente se hace una prueba para comprobar el estado del cable; la fase número 2, consiste en diseñar una red local en dos dimensiones, la principal funcionalidad de esta fase es permitirle al usuario confeccionar una red, colocando los dispositivos y conectándolos entre ellos; la última fase de este laboratorio tiene como principal funcionalidad diseñar una red en un local (Edificio), donde el usuario coloca los distintos dispositivos de red y los conecta entre ellos, desplazándose por todo el edificio. La interfaz le muestra al usuario un panel lateral donde se encuentran todos los equipos que puede colocar en el diseño de la red, tanto para la fase 2, como para la fase 3, y en caso de la fase 1 se muestran también los tipos de cables que puede ensamblar el usuario, además en común se muestra para cada una de ellas, las herramientas que se pueden utilizar en las mismas, ejemplo: Pinzas y otros.

Al finalizar el usuario puede exportar el estado de la red diseñada. Las principales funcionalidades desarrolladas y mostradas por la interfaz de usuario para este laboratorio virtual son; mostrar los objetos en la escena, así como manipular los mismos, mediante los cuales se confecciona el cable y se diseña la red local.

2.2.3. Laboratorio Virtual Administración y Configuración de una Red Local. (LAN).

Este laboratorio virtual está dividido en varias actividades, el mismo comienza cargando la red que se diseña en el Laboratorio Virtual de Diseño e Instalación de la red, por tanto la interfaz de usuario debe de mostrar un cuadro de dialogo donde se pueda realizar esta funcionalidad, de cargar un archivo. Se le brinda al mismo la posibilidad de Administrar y configurar los diferentes terminales de red (Computadoras), todo esto mediante un diálogo que le muestra las distintas opciones de lo que puede configurar en uno u otro sistema Operativo (GNU/Linux MS Windows). Para el caso específico de MS Windows, la interfaz muestra también un conjunto de Asistentes que simulan las distintas operaciones de

configuración en este sistema operativo. Para esto el usuario debe de manipular los distintos objetos gráficos mostrados en la escena.

2.3. Principios de diseño, buenas prácticas y herramientas utilizadas.

De los tipos de interfaz de usuario que se investigaron; como por ejemplo la interfaz de usuario por línea de comando, la basada en menús, y la interfaz gráfica de usuario; para los laboratorios virtuales se escogió la interfaz gráfica de usuario, debido a que este sistema que se está desarrollando va a ser utilizado como medio de enseñanza para estudiantes, y la interfaz gráfica de usuario contiene numerosas ventajas, como por ejemplo, el uso de metáforas e imágenes que muestran sin palabras las funcionalidades de la aplicación.

El diseño de la UI le brinda en todo momento la información al usuario de lo que está haciendo, los errores que ha cometido durante la ejecución del ejercicio, y el tiempo que está demorando en ejercer la práctica de laboratorio. Todo esto lo hace mediante el uso de relojes y barras de estado en la parte inferior de la aplicación, y en la parte superior la barra de título mostrándole el nombre de la ventana o diálogo.

Tomando el criterio de las buenas prácticas para el desarrollo de una correcta interfaz de usuario, se le da uso también a la barra de menús, donde son colocados cada uno de los menús que van a ser mostrados por la aplicación, y los elementos de los mismos fueron organizados teniendo en cuenta el flujo lógico del usuario dentro de la aplicación. También se hace uso de la barra de herramientas en la cual se muestran las principales funcionalidades, recogidas en los menús, y representados con iconos y gráficas, para un mejor entendimiento del usuario, todos los elementos incluidos en la barra de herramientas siguen el mismo patrón de colores y son mostrados con imágenes muy sugerentes que dan una idea de lo que el usuario quiere hacer.

Las etiquetas son utilizadas para mostrar información al usuario de manera uniforme y el texto que contiene siempre comienza con letra inicial mayúscula. Por otra parte los cuadros de diálogos utilizados contienen la misma estructura de los botones, así como un título que le indica al usuario la operación que va a realizar dentro del mismo.

Los colores, la tipografía y el tamaño de la fuente fueron seleccionados teniendo en cuentas las pautas tipográficas y cromáticas que ya habían sido definidas para el entorno al que se

iban integrar los laboratorios, utilizándose como el color principal el negro sobre el fondo gris, aportando un nivel de seriedad a la aplicación.

Los menús, así como la mayoría de las principales funcionalidades contienen atajos de teclado, para en caso de que el usuario este más familiarizado con la aplicación pueda utilizarlos y así no se vuelve tedioso el uso de los mismos. La aplicación puede cerrarse incluso al no haber concluido la práctica en tal caso se le muestra al usuario un mensaje previo de aviso para que genere el reporte de lo que ha hecho.

Entre los principales principios de diseño se tuvo en cuenta también, que el sistema tenga con un punto de partida, o sea un lugar de residencia, que no es exactamente la primera pantalla que muestra la aplicación, sino la ventana principal donde se encuentra el área de trabajo, desde ahí parten todas las demás funcionalidades de los laboratorios virtuales.

El tamaño de los gráficos (metáforas) e iconos mostrados en la misma son fáciles de visualizar por el usuario, estos son usados de manera consistente en todos los laboratorios virtuales y en todos los lugares donde se pueden encontrar los mismos, dándole uniformidad al sistema y de esta manera contribuir a que se agilice el proceso de aprendizaje de los mismos y una vez familiarizado con cualquiera de los laboratorios, puede utilizar fácilmente los demás, cumpliendo así con otro de los principios de diseño de una interfaz de usuario que consiste en desarrollar un sistema que el usuario sea capaz de interactuar con él, sin necesidad de acudir a los manuales.

Contiene además un panel lateral donde se encuentran todos los elementos que pueden aparecer dentro de la escena y además otro panel que contienen los errores que se van cometiendo dentro de la ejecución del ejercicio. Ambos (paneles y barras de herramientas) pueden moverse y cerrarse a gusto del usuario, permitiéndole a este sentirse dueño de la aplicación, y además que tenga más espacio de trabajo en caso de que lo necesite.

Es una interfaz que se adapta a la práctica de laboratorio que quiere hacer el usuario en el momento de utilizar la aplicación, los elementos mostrados en los menús y en el panel de herramientas, son cargados desde un directorio, y en caso de que el usuario no quiera utilizarlos, los puede eliminar.

Otro principio muy importante tenido en cuenta dentro del desarrollo de la interfaz de usuario fue la ley de Fitt, dándole mayor importancia dentro del área de trabajo, a la escena que se muestra, en la aplicación.

2.3.1. Biblioteca gráfica.

Luego de un estudio teórico realizado, se optó por desarrollar la interfaz de usuario utilizando el Qt, es totalmente orientado a objeto y su código está implementado directamente en C++, la calidad en tiempo de ejecución es muy alta, y no necesita de otras herramientas ni otra biblioteca. [16] Posee un uso adecuado de la memoria y es multiplataforma, no es necesario aprender a utilizar otra API, por lo que se puede desarrollar cualquier aplicación y ser portable para varias plataformas. Contiene además un conjunto de widgets, y además permite crear otros controles personalizados, presenta una muy buena documentación y ayuda online en internet, así como foros etc.

2.3.2. IDE

Después de haber estudiado varios IDEs. Se escogió el Qt Creator para el desarrollo de los Laboratorios Virtuales ya que presenta numerosas características que hacen mucho más ágil el trabajo con las bibliotecas de Qt por ejemplo:

- Avanzado editor de código C++.
- Soporta los lenguajes C#, .NET, Python, Ada, Pascal, Perl, PHP y Ruby
- Trae un GUI integrado y un diseñador de formularios.
- Herramientas para la administración de proyectos.
- Ayuda sensible al contexto.
- Depurador visual
- Resaltado y autocompletado de código

Otras características que presenta son: [14]

- Qt4 Proyecto de Generación Wizard: Este asistente permite al usuario generar un proyecto para una aplicación de consola, una aplicación GUI, o una librería C++.
- Qt Ayuda de Integración: Se puede acceder a toda la documentación de Qt fácilmente haciendo clic en el botón de Ayuda.
- Qt Integración de Diseño: formas de interfaz de usuario puede ser diseñado en Qt Creator, simplemente con hacer doble clic en la interfaz de usuario de un archivo dentro del explorador de proyectos para poner en marcha la integración.

- Localizador: Una herramienta de navegación muy potente que permite al usuario localizar los archivos y las clases utilizando pulsaciones mínima.

2.4. Modelo de dominio.

El negocio se representa mediante un Modelo de Dominio, ya que es fácil de interpretar el funcionamiento del sistema y la simplicidad del entorno donde está enmarcado. Se utiliza el modelo de dominio ya que es un subconjunto del modelo del negocio y se utiliza cuando las fronteras de negocio no están bien definidas. El modelo de dominio es una representación visual de las clases conceptuales del mundo real. Se debe concebir como un diccionario visual de abstracciones que será utilizado en fases posteriores. La función del modelo de dominio es ayudar a comprender el problema que se plantea.

2.4.1. Diagrama modelo de dominio.

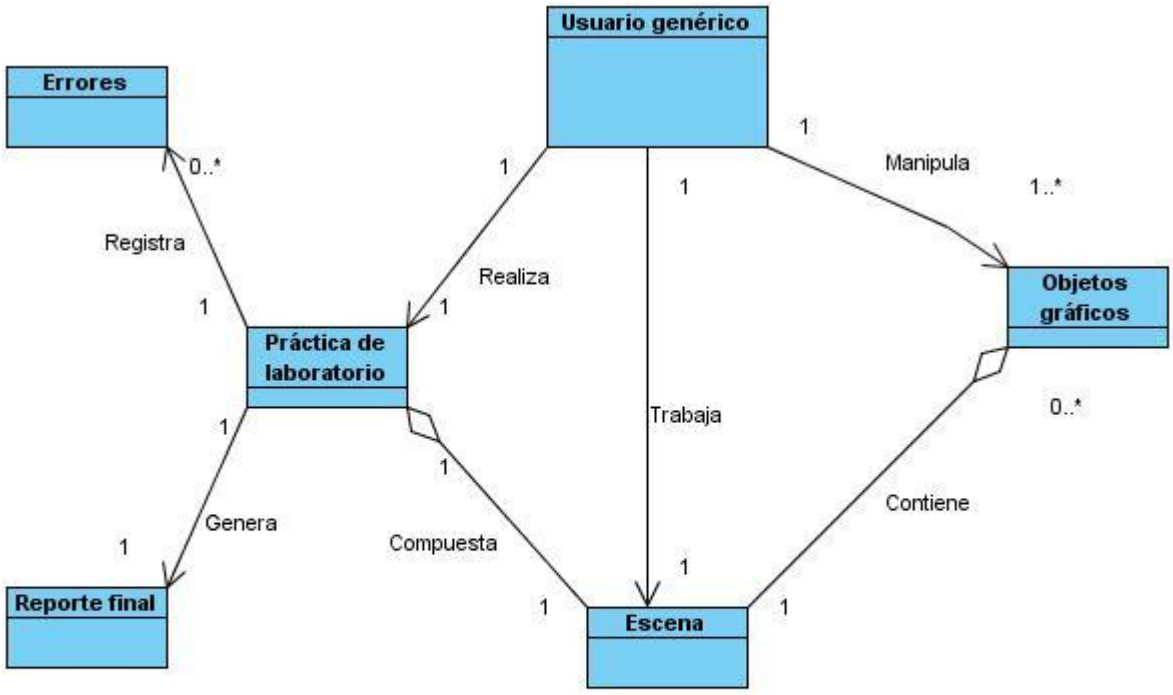


FIGURA 2. *DIAGRAMA MODELO DE DOMINIO.*

2.4.2. Descripción de las clases que componen el modelo de dominio.

Usuario genérico: Es aquel estudiante o profesor que está capacitado para realizar la práctica de laboratorio, en caso del primero, es el usuario que va a ser evaluado, y en el caso del segundo, es el encargado de revisar el reporte final.

Práctica de laboratorio: Constituye el ambiente de trabajo con la que va a interactuar el usuario, y el medio mediante el cual se desarrollará la solución propuesta.

Escena: Es el lugar donde se van a ubicar los objetos gráficos para interactuar con ellos, y realizar la práctica de laboratorio.
Objetos gráficos: Podrán ser utilizados por los estudiantes para visualizar los componentes de la práctica de laboratorio.

Errores: En cada práctica de laboratorio el sistema almacena y muestra los errores cometidos por los usuarios.

Reporte final: Guarda el resumen de toda la práctica realizada por el usuario.

El usuario realiza una serie de prácticas de laboratorios, interactuando con los elementos gráficos de una ensena. Durante la práctica se van controlando los errores cometidos y finalmente se emite un reporte con los datos y las actividades realizadas durante la práctica del laboratorio.

2.5. Levantamiento de requisitos del sistema.

Este flujo de trabajo es de vital importancia para el desarrollo de una aplicación informática, es donde se deja claro, lo que se quiere hacer, y lo que el cliente desea ver en el sistema que se va a desarrollar. El mismo tiene como misión convertir el problema expresado, en términos de dominio del negocio a soluciones descritas en lenguaje de dominio de la tecnología de información. Consiste en un conjunto de actividades de Ingeniería de software que se ocupa de recoger las necesidades de los clientes y los usuarios, sobre un sistema dado y traducirlas a especificaciones técnicas del sistema. Los requisitos se clasifican en requisitos funcionales y no funcionales, y constituyen el contrato que se debe de cumplir. La interfaz de usuario de los laboratorios virtuales debe de brindar la posibilidad de que el

sistema cumpla con las expectativas, y los requisitos funcionales que serán expuestos a continuación.

2.5.1. Requisitos funcionales generales.

RF1. Autenticar.

RF2. Mostrar objetos en la escena.

RF3. Mover objetos gráficos.

RF4. Cargar archivos.

RF5. Salvar archivo.

RF6. Mostrar errores cometidos por el usuario.

RF7. Mostrar tiempo de duración de la práctica.

RF8. Realizar Zoom.

RF9. Generar reporte final.

2.5.2. Requisitos no funcionales.

Los requisitos no funcionales son aquellos que especifican los criterios que pueden usarse para juzgar la operación de un sistema, no su comportamiento especifico, por lo tanto, se puede definir requisito no funcional aquel requisito que no describe una funcionalidad dentro del sistema, sino cualidades con los que debe cumplir el mismo. Los requisitos no funcionales, se clasifican en varios grupos.

1. ***Hardware.***
    a. Procesador Pentium 4 a 2.7 GHz o superior.
    b. Memoria RAM de 1GB de capacidad.
    c. Espacio en disco duro de 512 MB.

2. ***Usabilidad.***

a. El sistema deberá ser multiplataforma, es decir deberá correr sobre los sistemas operativos Windows y Linux.

b. El sistema se encontrará incluido dentro del portal de aplicaciones educativas.

3. Requisitos no funcionales de Restricciones en el Diseño e Implementación.

a. La aplicación se desarrollará utilizando Ogre como motor gráfico.
b. Se utilizará QT Creator como IDE de desarrollo.
c. En el trabajo con gráficos tridimensionales se utilizará la herramienta Blender.

4. Requisitos no funcionales de Interfaz.

En esta sección se definen las interfaces que deben ser soportadas por la aplicación. La aplicación contará con dos tipos de interfaces de usuario, las cuales se detallan a continuación:

a. Interfaz principal: la misma estará formada por todos los elementos gráficos con los que contará la práctica de laboratorio virtual. Es la que permitirá la interacción entre el usuario y la computadora.

b. La interfaz alfanumérica: servirá como intérprete de mandato, sólo presentará los textos dentro de la aplicación.

2.6. Definición del actor del sistema.

Actor del sistema	Justificación
Usuario genérico	Persona con privilegios para interactuar con la aplicación. Haciendo uso de todas las funcionalidades con que cuenta la misma con fines educativos. Este usuario genérico puede ser el estudiante, el profesor o cualquier persona con el objetivo de interactuar con dicha aplicación.

TABLA 1. *ACTOR DEL SISTEMA.*

2.7. Casos de usos del sistema.

Los casos representan la manera en que los usuarios interactúan con el sistema. Son artefactos narrativos que describen el comportamiento de una aplicación teniendo en cuenta el punto de vista del actor. Para tratar los casos de uso del sistema de laboratorios virtuales de manera general para los tres laboratorios virtuales.

2.7.1. Diagrama de casos de uso del sistema

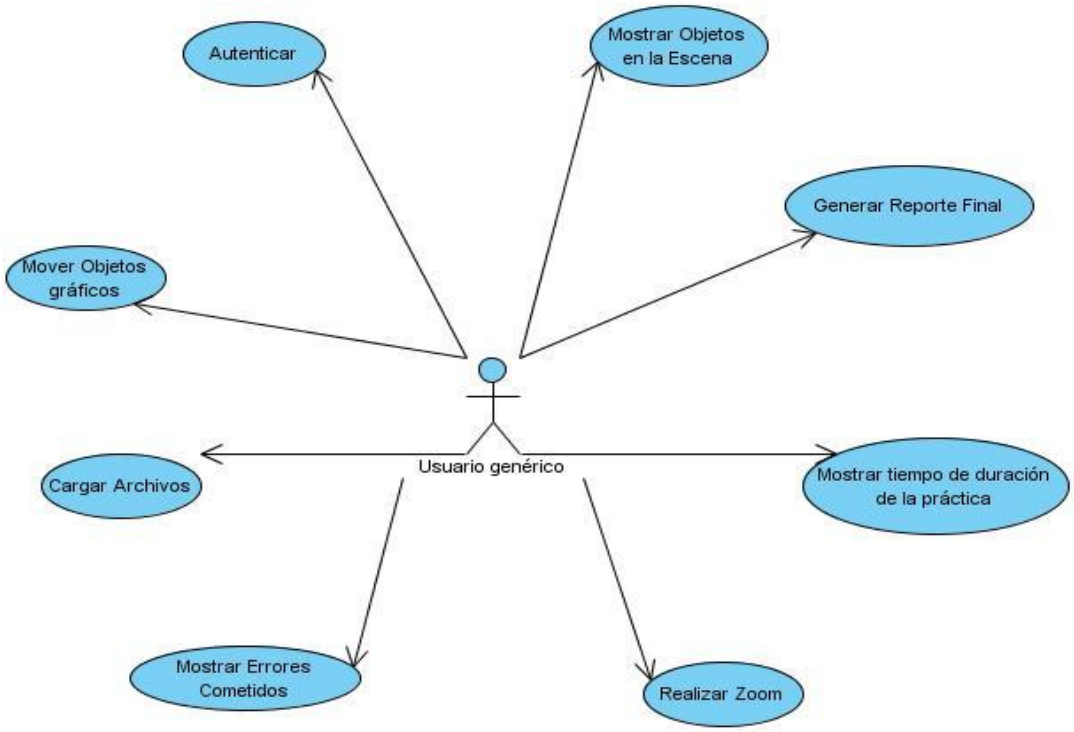


FIGURA 3. *DIAGRAMA CASOS DE USO DEL SISTEMA.*

2.7.2. Descripción de los casos de uso del sistema.

Caso de Uso	Autenticar.
Objetivo	Permitir autenticarse en la aplicación.
Actores	Estudiante y usuario (genérico).
Resumen	El caso de uso se inicia cuando el usuario ejecuta el laboratorio Virtual. La aplicación deberá mostrar una ventana donde el usuario inserte su nombre, apellido(s), número de cédula o pasaporte. Finaliza cuando comienza la realización de la práctica de laboratorio.
Complejidad	Alta
Requisitos de Referencia	RF 1
Prioridad	Crítico
Postcondiciones	El estudiante y usuario (genérico) pueden acceder a la realización de la práctica del laboratorio virtual.

<table>
<tr><td colspan="4">Flujo de eventos</td></tr>
<tr><td colspan="4">Flujo básico: Autenticar Usuario</td></tr>
<tr><td colspan="4">Sección 1: Autenticar Usuario.</td></tr>
<tr><td></td><td colspan="2">Actor</td><td>Sistema</td></tr>
<tr><td>➢</td><td colspan="2">Ejecutar el Laboratorio Virtual.</td><td>1.1. La interfaz muestra un diálogo donde el usuario ingresa los datos que lo identifica, y escoge la fase que va a realizar la práctica.</td></tr>
<tr><td>1.2</td><td colspan="2">Entrar los datos y comprobarlos.</td><td>1.3 Si los datos son correctos se la interfaz muestra el Laboratorio seleccionado.</td></tr>
<tr><td colspan="4">Flujos alternos</td></tr>
<tr><td colspan="4">Sección 1: Autenticar Usuario.</td></tr>
<tr><td colspan="4">Nº Evento Los datos son incorrectos.</td></tr>
<tr><td></td><td colspan="2">Actor</td><td>Sistema</td></tr>
<tr><td>3.</td><td colspan="2">Inserta mal algunos de los datos pedidos. Nombre y número de identificación.</td><td>3.1. Emite un mensaje especificando el error cometido.</td></tr>
<tr><td>Relaciones</td><td>CU Incluidos</td><td colspan="2">-</td></tr>
<tr><td></td><td>CU Extendidos</td><td colspan="2">-</td></tr>
<tr><td>Requisitos no funcionales</td><td colspan="3">-</td></tr>
<tr><td>Asuntos pendientes</td><td colspan="3">-</td></tr>
</table>

TABLA 2. *DESCRIPCIÓN DEL CASO DE USO AUTENTICAR.*

Caso de uso.	Mostrar objetos en la escena.
Objetivo	Permitir mostrar los objetos de la escena.
Actores	Estudiante y usuario (genérico).

Resumen	El caso de uso se inicia cuando el usuario ha comenzado la realización de la práctica. La aplicación deberá mostrar los objetos que contiene la escena después de cargada. Termina cuando ha completado la práctica del Laboratorio.	
Complejidad	Alta	
Requisitos de Referencia	RF 2	
Prioridad	Secundario	
Precondiciones	Comprobar que los objetos sean mostrados según la fase en la que se encuentre el usuario.	
Postcondiciones	Los objetos gráficos serán mostrados según la fase a la que corresponden.	
Flujo de eventos		
Flujo básico: Mostrar objetos de la escena.		
Sección:"Mostrar objetos de la escena."		
	Actor	**Sistema**
1	Accederá a la realización un laboratorio específico en la fase seleccionada.	1.1. Mostrará solamente los objetos que corresponden a la fase seleccionada Objetos- Laboratorio Virtual Arquitectura de Computadoras. Fase 1-Escenario 1: Dispositivos internos de la computadora: Objetos- Laboratorio Virtual Diseño e Instalación. Fase 1-Escenario 1: Herramientas para confeccionar Cables de Red. Objetos- Laboratorio Virtual Diseño e Instalación. Fase 2-Escenario 1: Dispositivos para diseñar una red

		LAN en dos dimensiones. Objetos- Laboratorio Virtual Diseño e Instalación. Fase 3-Escenario 1: Dispositivos para diseñar una red LAN en tres dimensiones. Objetos- Laboratorio Virtual Configuración Y Administración - Escenario 1: Escenario cargado que generó el Laboratorio Virtual Diseño e Instalación.
Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales	-	
Asuntos pendientes	-	

TABLA 3. *DESCRIPCIÓN DEL CASO DE USO MOSTRAR OBJETOS EN LA ESCENA.*

Caso de uso.	Mover Objetos gráficos.
Objetivo	La aplicación permitirá al usuario mover los objetos gráficos dentro de la escena.
Actores	Usuario genérico
Resumen	La aplicación le permitirá al usuario mover los objetos gráficos dentro de la escena.
Complejidad	Baja
Requisitos de Referencia	RF 3
Prioridad	Secundario
Precondiciones	El usuario debe haber seleccionado algún objeto.
Postcondiciones	El usuario mueve los objetos gráficos dentro de la escena en la que se encuentre.

<table>
<tr><td colspan="5">Flujo de eventos</td></tr>
<tr><td colspan="5">Flujo básico: Mover objetos gráficos.</td></tr>
<tr><td colspan="5">Sección: ”Seleccionar objetos gráficos”</td></tr>
<tr><td></td><td colspan="2">Actor</td><td colspan="2">Sistema</td></tr>
<tr><td>1</td><td colspan="2">Accede a la aplicación.</td><td colspan="2"></td></tr>
<tr><td>2</td><td colspan="2">Selecciona el objeto gráfico a mover.</td><td colspan="2">2.1. El sistema le permite mover el objeto gráfico terminando así el caso de uso.</td></tr>
<tr><td colspan="2">Relaciones</td><td>CU Incluidos</td><td colspan="2">-</td></tr>
<tr><td colspan="2"></td><td>CU Extendidos</td><td colspan="2"></td></tr>
<tr><td colspan="2">Requisitos no funcionales</td><td colspan="3">-</td></tr>
<tr><td colspan="2">Asuntos pendientes</td><td colspan="3">-</td></tr>
</table>

TABLA 4. *DESCRIPCIÓN DEL CASO DE USO MOVER OBJETOS GRÁFICOS.*

Caso de uso.	Cargar archivos.
Actor.	Usuario genérico
Resumen.	En el caso del laboratorio virtual de Configuración y administración de redes, la aplicación deberá cargar un archivo previamente salvado por la misma o salvado en la fase 3 del laboratorio virtual “Diseño e Instalación de una red LAN”, si los datos del usuario que trata de hacerlo coinciden con los del archivo a cargar.
Referencia.	RF 5

TABLA 5. *DESCRIPCIÓN DEL CASO DE USO CARGAR ARCHIVOS.*

Caso de uso.	Mostrar errores cometidos
Objetivo	Permitir mostrar al estudiante o usuario (genérico) las estadísticas

	de errores y los errores más comunes según el escenario y la fase de la práctica realizada.
Actores	Estudiante y usuario (genérico).
Resumen	El caso de uso se inicia cuando el estudiante o usuario (genérico) durante la práctica del Laboratorio comete algún error, la aplicación muestra la cantidad de errores cometidos por el usuario según el escenario y fase en que se encuentre el mismo. Se mostrará como una pantalla dentro de la aplicación. Luego selecciona la opción reporte y finaliza cuando se almacena automáticamente los datos en la aplicación.
Complejidad	Baja
Requisitos de Referencia	RF 6
Prioridad	Secundario
Precondiciones	El usuario (genérico) debe haber realizado la práctica.
Postcondiciones	El usuario (genérico) pudo acceder a los errores cometidos en la práctica.

Flujo de eventos		
Flujo básico: Monitorizar las estadísticas de los errores.		
Sección:"Mostrar los errores cometidos por el usuario."		
	Actor	**Sistema**
1	Realiza la práctica de laboratorio.	1.1 Registra y muestra los errores cometidos durante el desarrollo de la misma a medida que va cometiendo imperfecciones en la práctica de laboratorio.
2	Selecciona la opción *reporte-usuario* una vez terminada la práctica.	2.1. Muestra una pantalla con los siguientes campos: ➢ Cantidad de errores cometidos: número ➢ Errores cometidos: Tipos de errores. ➢ Tiempo de duración de la práctica: horas-minutos-

		segundos.
2	Observa sus errores cometidos.	2.1. Automáticamente guardará sus evaluaciones.
Flujos alternos		
Nº Evento Errores cometidos		
	Actor	**Sistema**
1.1	Si no ha cometido errores	1.1. Emitirá un mensaje: "Usted ha realizado satisfactoriamente su práctica."
Flujos alternos		
Nº Evento Monitorizar las estadísticas de los errores.		
	Actor	**Sistema**
1.	Si no ha cometido errores.	1.1. Emitirá un mensaje: "Usted ha realizado satisfactoriamente su práctica."

Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales	-	
Asuntos pendientes	-	

TABLA 6. *DESCRIPCIÓN DEL CASO DE USO MOSTRAR ERRORES COMETIDOS.*

Caso de uso.	Mostrar tiempo de duración de la práctica
Objetivo	Permitir mostrar al usuario (genérico) el tiempo de duración de la práctica.
Actores	Estudiante y usuario (genérico).
Resumen	El caso de uso se inicia cuando el usuario (genérico) selecciona la opción comenzar la práctica. La aplicación deberá mostrar el tiempo que demoró el usuario desde que accedió a la práctica hasta su culminación. El formato en que se mostrará el tiempo es cantidad de horas-minutos-segundos transcurridos, desde que el

<table>
<tr><td colspan="2"></td><td colspan="3">usuario accede al primer escenario y fase. Finaliza cuando termina dicha práctica.</td></tr>
<tr><td colspan="2">Complejidad</td><td colspan="3">Baja</td></tr>
<tr><td colspan="2">Requisitos de Referencia</td><td colspan="3">RF 7</td></tr>
<tr><td colspan="2">Prioridad</td><td colspan="3">Auxiliar</td></tr>
<tr><td colspan="2">Precondiciones</td><td colspan="3">El usuario (genérico) debe haber comenzado la práctica.</td></tr>
<tr><td colspan="2">Postcondiciones</td><td colspan="3">El usuario (genérico) una vez concluida la práctica.</td></tr>
<tr><td colspan="5">Flujo de eventos</td></tr>
<tr><td colspan="5">Flujo básico: Mostrar el tiempo de duración de la práctica.</td></tr>
<tr><td colspan="5">Sección: “Mostrar el tiempo de duración de la práctica.”</td></tr>
<tr><td></td><td colspan="2">Actor</td><td colspan="2">Sistema</td></tr>
<tr><td>1</td><td colspan="2">Realiza la práctica una vez registrado los datos del usuario.</td><td colspan="2">1.1. Muestra en la esquina inferior derecha de la pantalla el tiempo de duración desde que fue iniciada la práctica hasta la culminación de la misma, una vez terminada las fases.</td></tr>
<tr><td></td><td colspan="2"></td><td colspan="2">1.2. Muestra el tiempo de duración de la práctica con el siguiente formato:
cantidad de horas-minutos-segundos transcurridos.</td></tr>
<tr><td colspan="2">Relaciones</td><td>CU Incluidos</td><td colspan="2"></td></tr>
<tr><td colspan="2"></td><td>CU Extendidos</td><td colspan="2"></td></tr>
<tr><td colspan="2">Requisitos no funcionales</td><td colspan="3">-</td></tr>
<tr><td colspan="2">Asuntos pendientes</td><td colspan="3">-</td></tr>
</table>

TABLA 7. *DESCRIPCIÓN DEL CASO DE USO MOSTRAR TIEMPO DE DURACIÓN DE LA PRÁCTICA.*

Caso de uso.	Realizar Zoom
Actor.	Usuario genérico
Resumen.	La aplicación le permitirá al usuario acercar y volver a la distancia original en la que se encuentran los objetos en cada escenario.
Referencia.	RF 8

TABLA 8. *DESCRIPCIÓN DEL CASO DE USO REALIZAR ZOOM.*

Caso de uso.	Generar reporte final.
Actor.	Usuario genérico
Resumen.	La aplicación deberá darle la posibilidad al usuario de generar un reporte de lo que ha realizado en la práctica de laboratorio, el cual contenga el tiempo, los errores y las acciones realizadas dentro de la misma. El reporte será guardado con formato PDF, para luego ser utilizado por el profesor.
Referencia.	RF 9

TABLA 9. *DESCRIPCIÓN DEL CASO DE USO GENERAR REPORTE FINAL.*

Conclusiones generales del capítulo.

En el capítulo se da la solución propuesta del problema planteado para la investigación, además se muestra el prototipo de la interfaz que se va a desarrollar. También se da una descripción del modelo de dominio, para comprender mejor el funcionamiento del sistema. Se realizó la selección de los actores que intervienen en el sistema, y se detectaron los requisitos funcionales y no funcionales de los laboratorios virtuales, luego agrupados en casos de usos y descritos para una mejor comprensión de las funcionalidades con que debe de contar la interfaz desarrollada.

3

Capítulo 3. Diseño del Sistema.

Introducción.

En este capítulo se trataran el proceso de diseño, el cual va a ser representado mediante artefactos tales como los diagramas de clases de diseño, los diagramas de secuencia, que le dan solución a la interfaz de usuario de los laboratorios virtuales.

3.1. Diseño del sistema.

El diseño tiene entre sus propósitos fundamentales adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, sistemas operativos y tecnologías de interfaz de usuarios, entre otros. Es el encargado de crear una entrada apropiada y un punto de partida para las posteriores actividades de implementación. Dentro de este flujo de trabajo se elaboran los diagramas de clases de diseño, los cuales muestran las clases finales que dan la realización y el desarrollo de los casos de uso que fueron anteriormente modelados.

Los diagramas de clases obtenidos a partir de este flujo de trabajo muestran las clases del sistema y las relaciones que existen entre ellas, y muestran lo que el sistema puede realizar y como puede ser construido el mismo. Existen dos tipos de diagramas que utilizan la misma información pero cada uno enfatizando un aspecto específico estos diagramas son los de secuencia y colaboración.

3.2. Diagrama de Clases del diseño.

Los diagramas de clases constituyen un tipo de diagrama estático, utilizados en el proceso de análisis y diseño, muestra de manera descriptiva la estructura del sistema, las clases que lo conforman atributos y las relaciones entre las mismas.

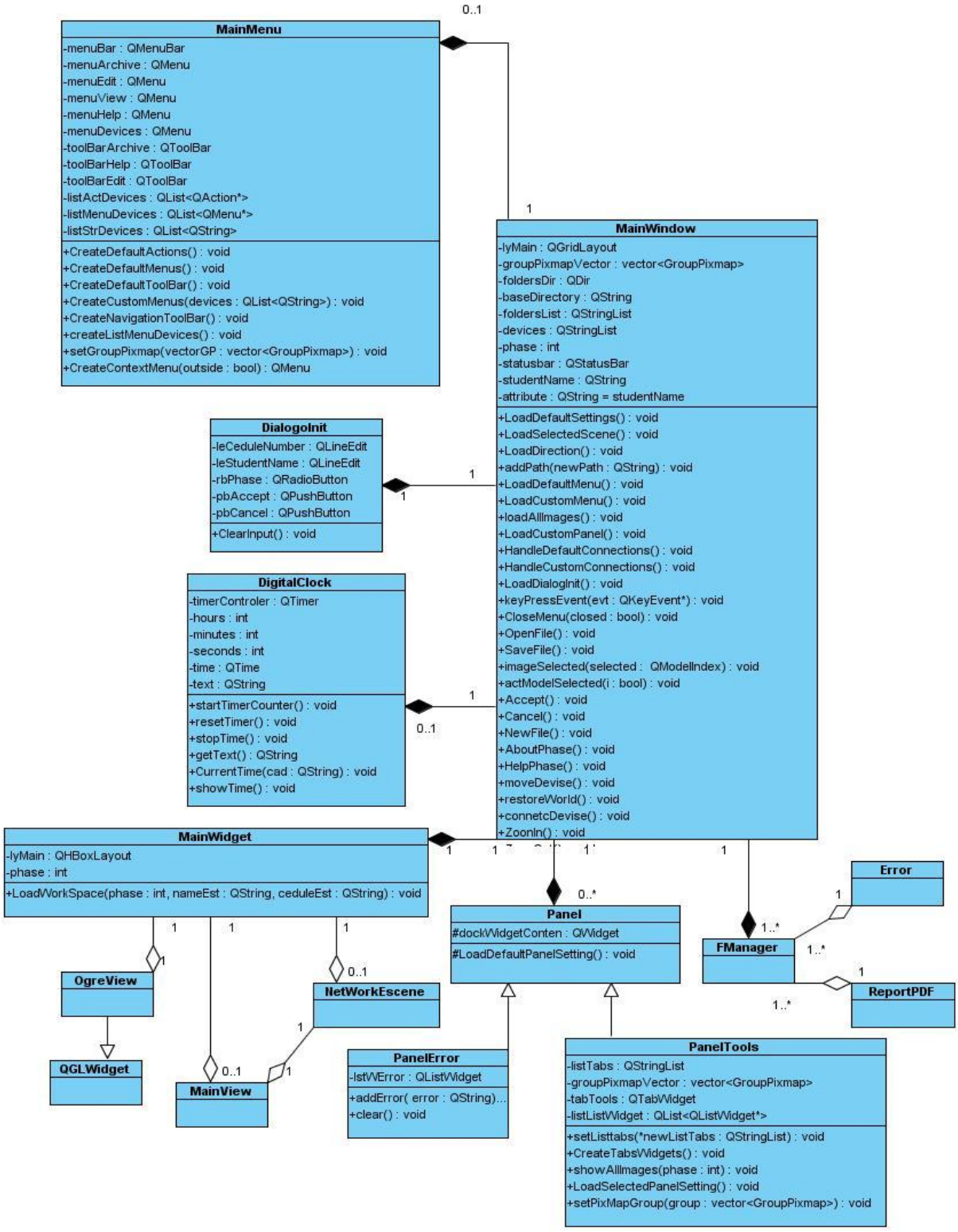


FIGURA 4. *DIAGRAMA DE CLASES DEL DISEÑO.*

3.3. Diagramas de Interacción.

Los diagramas de interacción describen el comportamiento y en que colaboran un cierto número de objetos. Por lo general son orientados a un solo caso de uso. Dentro de los mismos encontramos los diagramas de secuencia los cuales son designados para mostrar gráficamente los eventos originados por los actores que tienen un impacto fundamental dentro del sistema. La elaboración de estos diagramas tiene gran dependencia de la formulación que se le dé a los casos de uso. En los mismos los actores son encargados de generar varios eventos.

3.3.1. Diagrama de secuencia Autenticar.

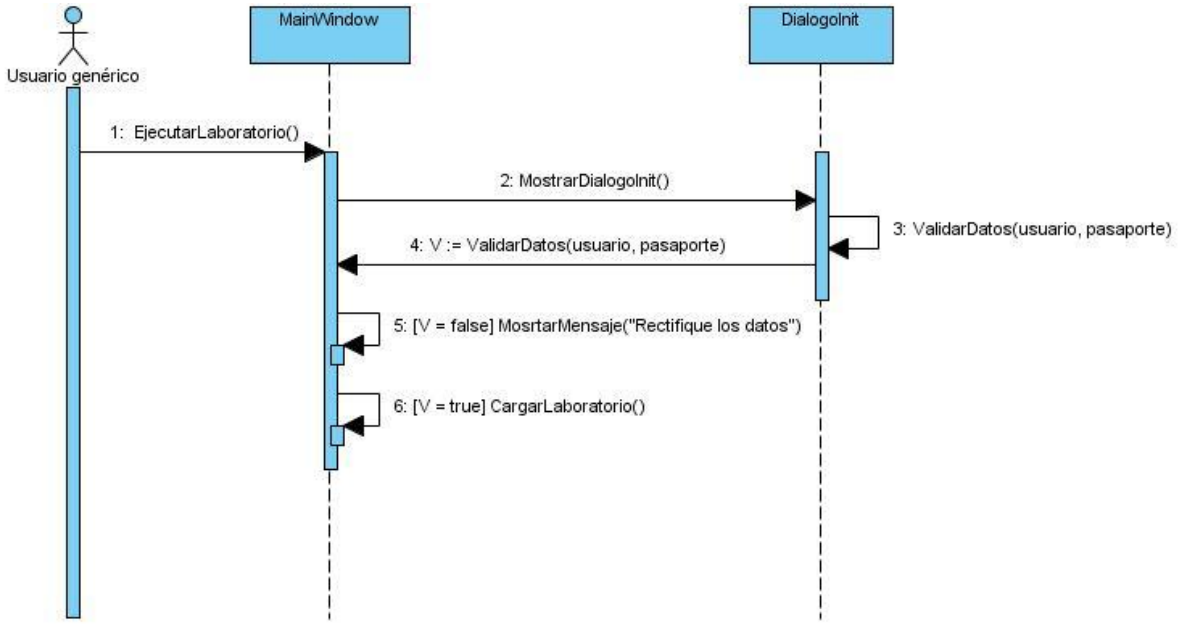


FIGURA 5. *DIAGRAMA DE SECUENCIA AUTENTICAR USUARIO.*

3.3.2. Diagrama de secuencia Acceder a la práctica de laboratorio.

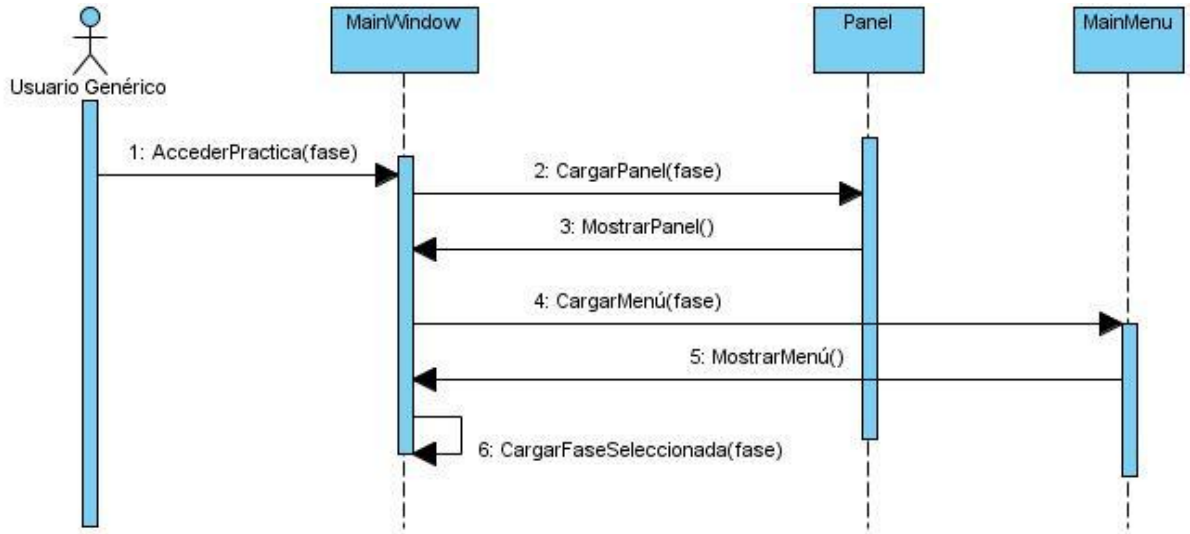


FIGURA 6. *DIAGRAMA DE SECUENCIA ACCEDER AL LABORATORIO VIRTUAL.*

3.3.3. Diagrama de secuencia Mostrar objetos de la escena.

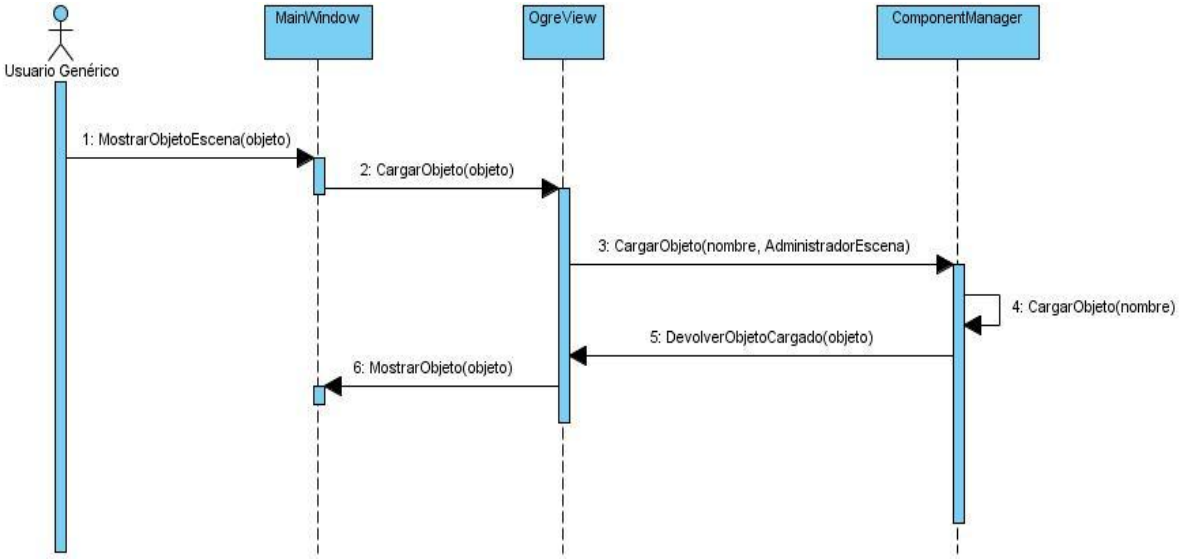


FIGURA 7. *DIAGRAMA DE SECUENCIA MOSTRAR OBJETOS EN LA ESCENA.*

3.3.4. Diagrama de secuencia Cargar archivos.

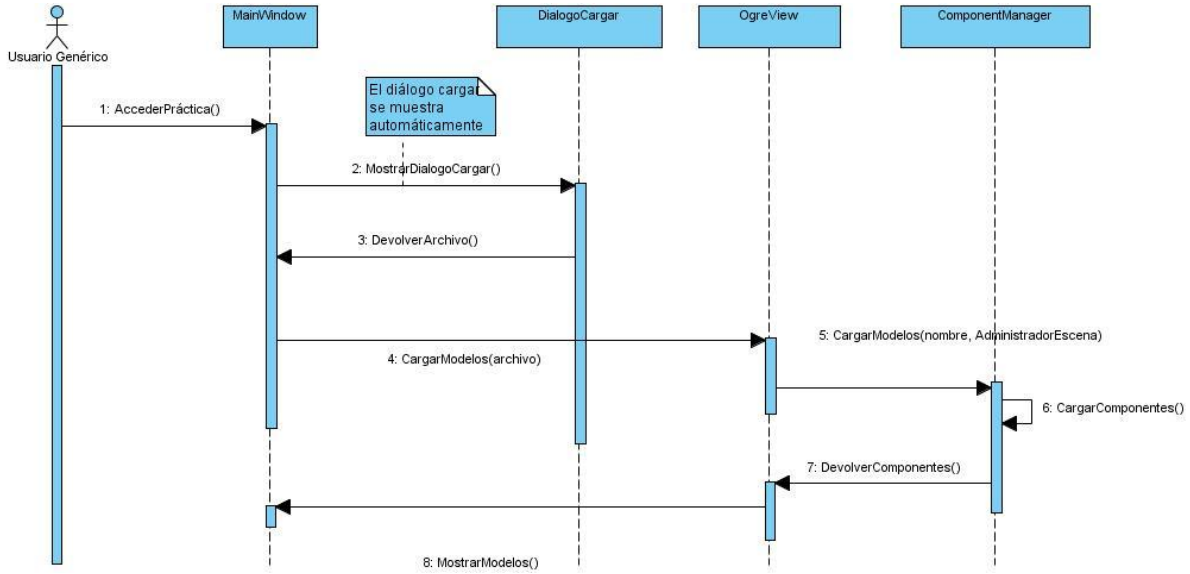


FIGURA 8. *DIAGRAMA DE SECUENCIA CARGAR ARCHIVO.*

3.3.5. Diagrama de secuencia Mostrar errores.

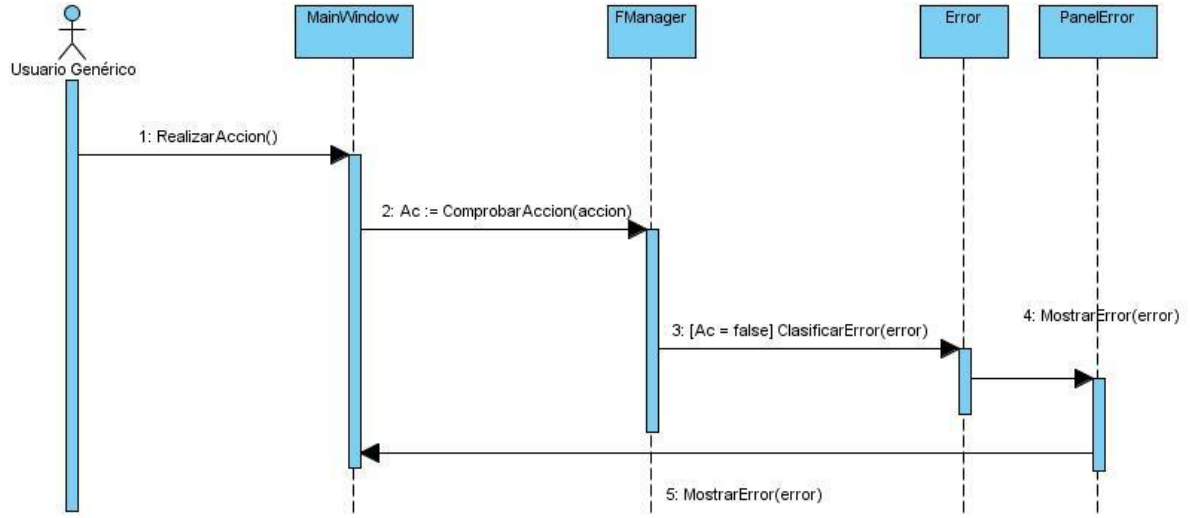


FIGURA 9. *DIAGRAMA DE SECUENCIA MOSTRAR ERRORES.*

3.3.6. Diagrama de secuencia Mostrar tiempo de duración de la práctica.

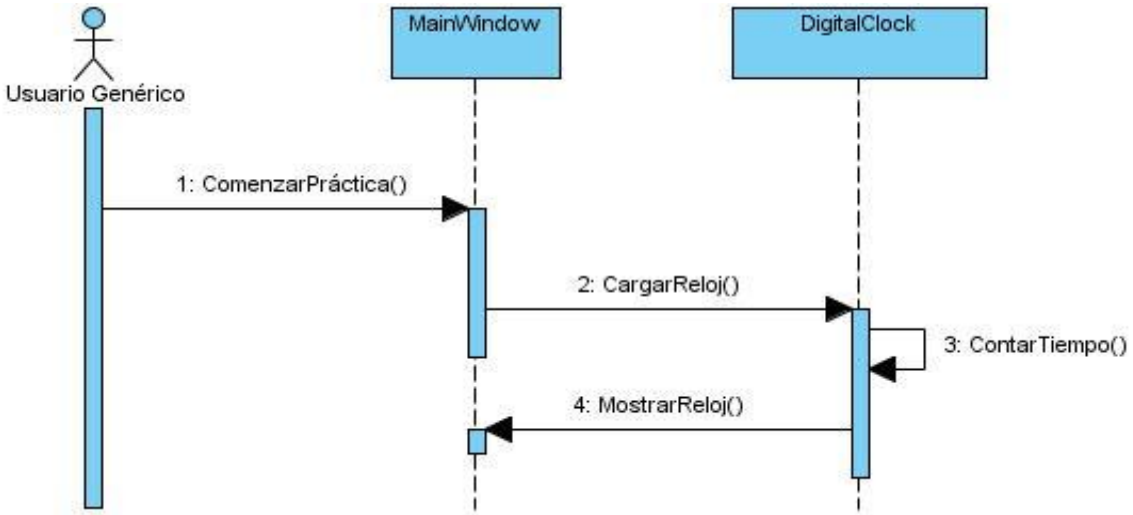


FIGURA 10. *DIAGRAMA DE SECUENCIA MOSTRAR TIEMPO DE DURACIÓN DE LA PRÁCTICA.*

3.3.7. Diagrama de secuencia Generar reporte final.

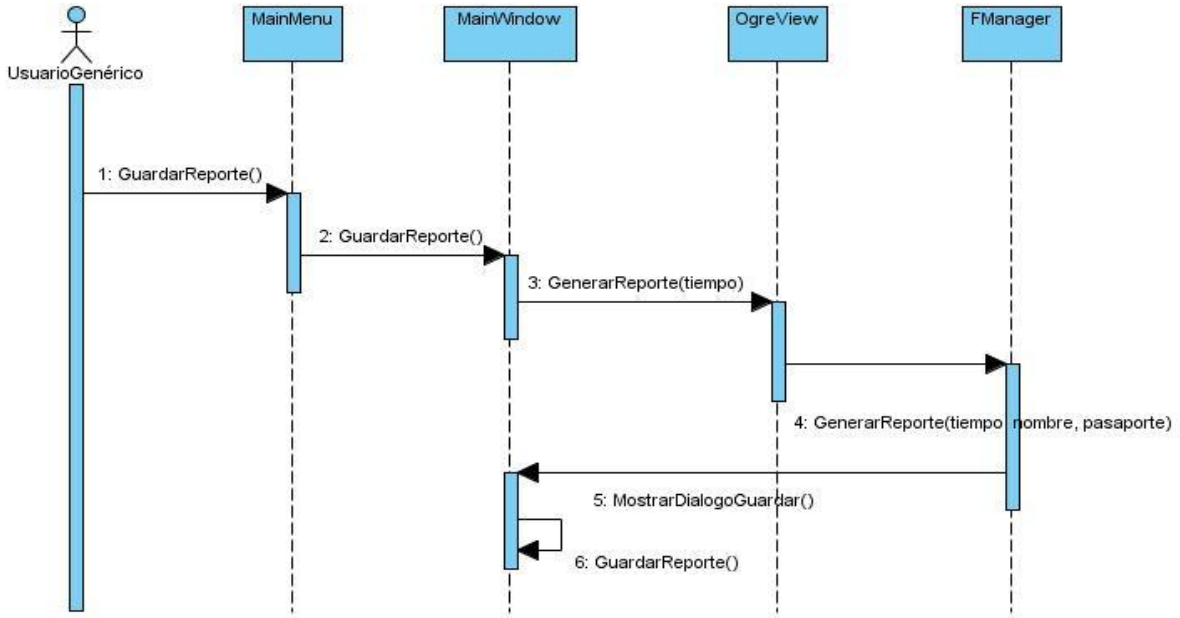


FIGURA 11. *DIAGRAMA DE SECUENCIA GENERAR REPORTE.*

Conclusiones generales del capítulo.

En el presenta capítulo se desarrolló lo referente al diseño del sistema, lo cual es de suma importancia para el desarrollo de todos software. Se comenzó dando una breve descripción de lo que constituye el diseño de un sistema. Luego se concluyó con el diseño final del sistema del cual se obtuvo el diagrama de clases de diseño donde se plasmaron las clases que conforman el sistema, así como las relaciones entre las mismas y posteriormente los diagramas de secuencia de los casos de uso a desarrollar.

4

Capítulo 4. Implementación y validación del Sistema.

Introducción.

El presente capítulo está dedicado al flujo de implementación realizado en el sistema. La implementación se hace con el objetivo de definir la organización del código teniendo en cuenta los subsistemas de implementación organizadas por capas, la implementación de los elementos de diseño en términos de ficheros fuentes, binarios, ejecutables y para poder integrar los diferentes componentes de desarrolladores o equipos y generar un ejecutable entregable o producto final.[25]

Luego de concluir la implementación del sistema el mismo debe cumplir con todos los requisitos funcionales y no funcionales del software.

4.1. Diagrama de componentes.

Los componentes constituyen la parte física de un sistema, las clases necesarias para la interfaz de usuario de los laboratorios virtuales se hacen físicas mediante componentes, a continuación se muestran los diagramas de componentes de dicha interfaz lo cual permite a los desarrolladores y a los clientes conocer la estructura física que tiene el sistema y como se relacionan sus partes.

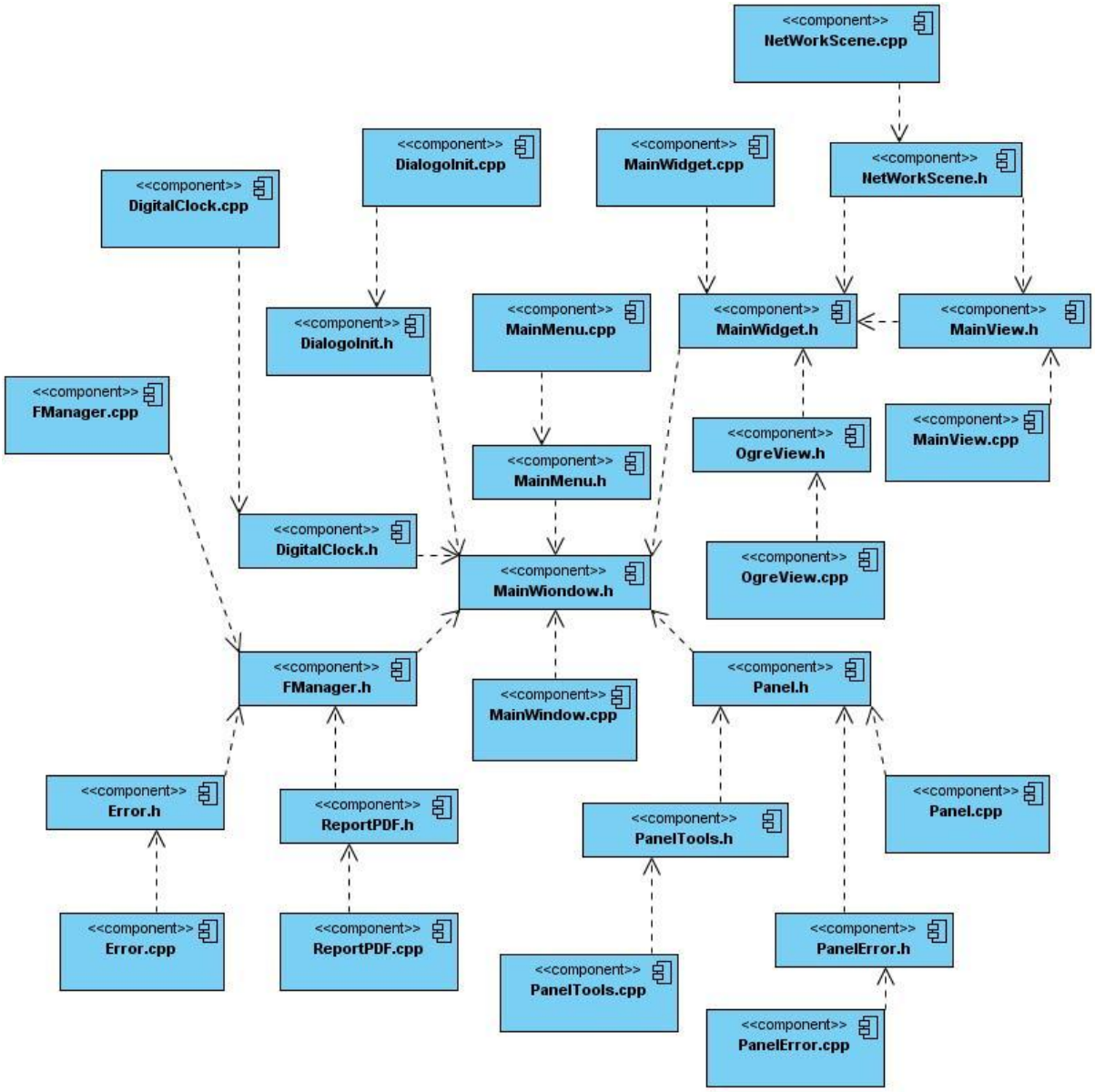


FIGURA 12. *DIAGRAMA DE COMPONENTES DE LA INTERFAZ GRÁFICA DE USUARIO.*

4.2. Validación.

Una de las características del siclo de vida de un software es la revisión periódica de su desarrollo, todo software también debe ser probado mediante su ejecución controlada antes de ser entregado al cliente, por lo tanto las pruebas son otro método para validar un software. Para la validación de la interfaz de usuario para los laboratorios virtuales se utiliza la prueba por casos de uso, las cuales están centradas principalmente a la calidad del producto y el funcionamiento del mismo.

Las pruebas realizadas son descritas teniendo en cuenta la sección, los escenarios la descripción de la funcionalidad y por último la descripción de las variables que intervienen en la funcionalidad que se va a probar.

4.2.1. Validación del caso de uso Autenticar.

Nombre de la Sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Autenticar	EC 1.1: Autenticar	El usuario tendrá la posibilidad de introducir los datos pedidos por la aplicación.
	EC 1.2: El usuario no llena todos o ninguno de los campos pedidos.	El usuario genérico (estudiantes o profesor) deja algunos campos pedidos (nombre, apellidos y número de cédula o número de pasaporte) sin llenar, por el cual la aplicación no le permite acceder a comenzar la práctica mostrándole un mensaje de error.
	EC 1.3: El usuario entra datos no válidos al sistema.	El usuario genérico (estudiantes o profesor, entra datos no válidos al sistema, por ejemplo un número en el campo nombre, letras en el campo número de cédula, o un nombre con menos de 8 caracteres, por lo que la aplicación no le permite acceder a comenzar la práctica mostrándole un mensaje de error.

TABLA 10. *SECCIONES A PROBAR EN EL CASO DE USO AUTENTICAR.*

Descripción de las variables:

Nombre y apellidos: Campo de texto, no admite datos nulos, sólo permite letras.

Número de cédula: No admite valor nulo, este campo solo permite números hasta 8 dígitos.

Número de pasaporte: Este campo no admite valores nulos, permite letras y números solo hasta 8 caracteres.

El usuario genérico debe de especificar su nombre y su número de cédula o pasaporte.

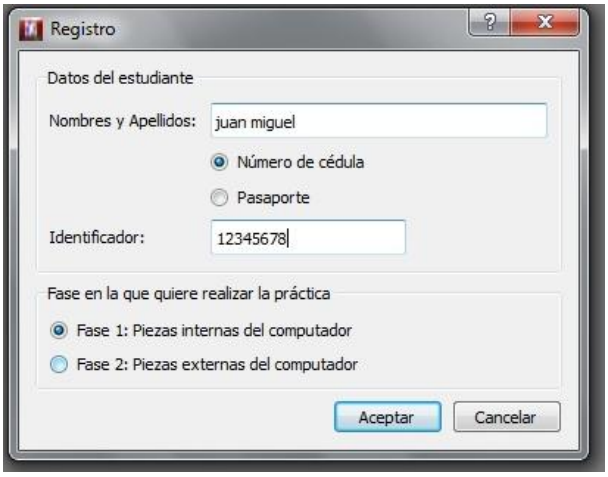


FIGURA 13. *DIÁLOGO AUTENTICAR USUARIO.*

- Error campos vacíos.

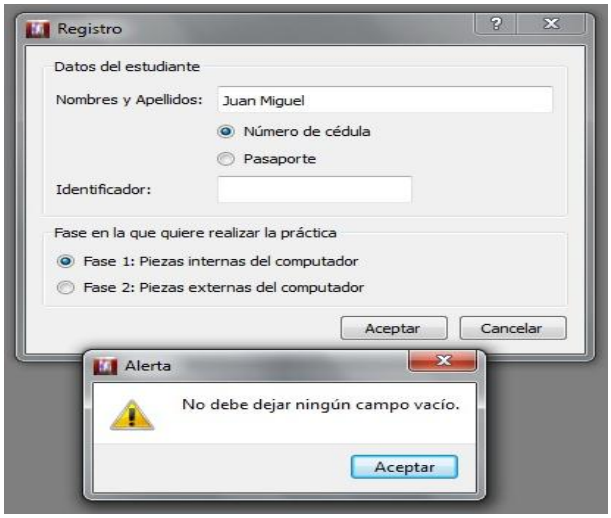


FIGURA 14. *DIALOGO AUTENTICAR USUARIO. MENSAJE DE ERROR CAMPOS VACÍOS.*

- Error carácter no válido.

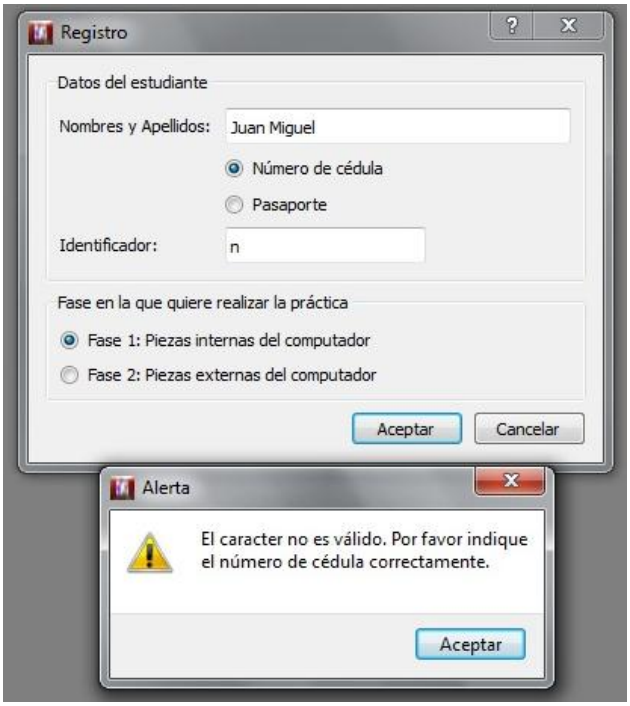


FIGURA 15. *DIALOGO AUTENTICAR USUARIO. MENSAJE DE ERROR NÚMERO DE IDENTIDAD NO VÁLIDA.*

- Error carácter nombre no válido.

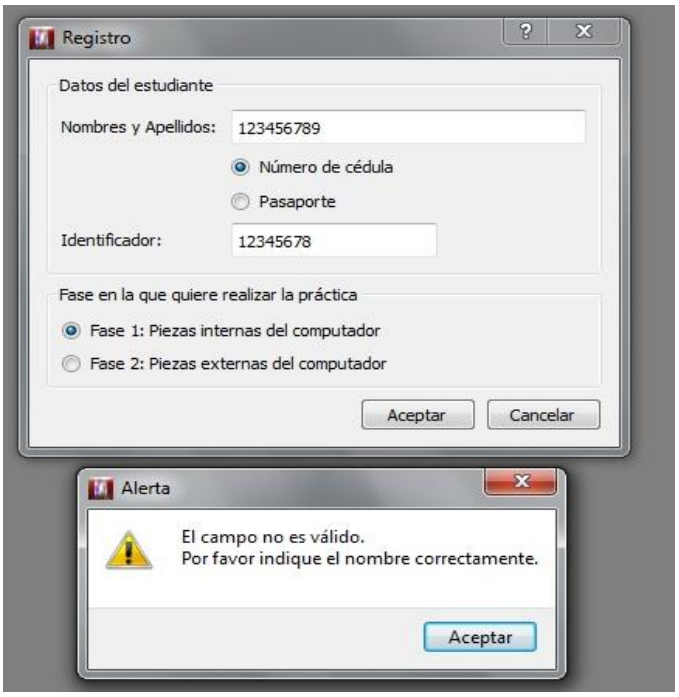


FIGURA 16. *DIALOGO AUTENTICAR USUARIO. MENSAJE DE ERROR NOMBRE NO VÁLIDO.*

4.2.2. Validación del caso de uso Mostrar objetos de la escena.

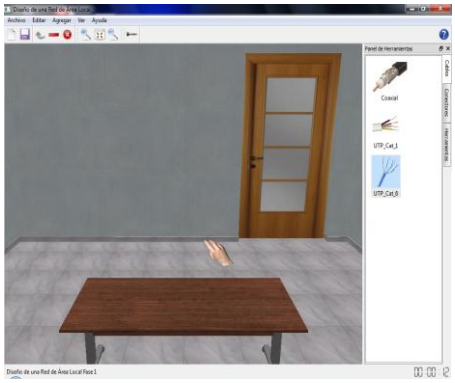
Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Mostrar objetos de la escena.	EC 1.1: Mostrar objetos de la escena.	El usuario tendrá la posibilidad de observar los objetos en la escena.
	EC 1.2:	

TABLA 11. *SECCIONES A PROBAR EN EL CASO DE USO MOSTRAR OBJETOS DE LA ESCENA.*

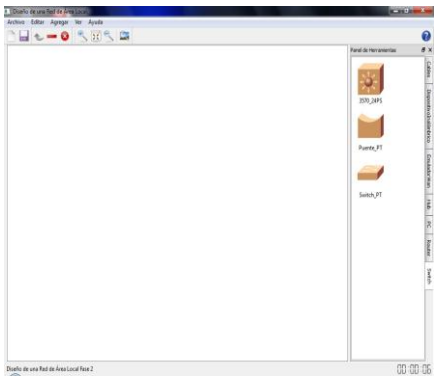
Descripción de las variables:

Objetos: Objetos encontrados en el panel de las herramientas.

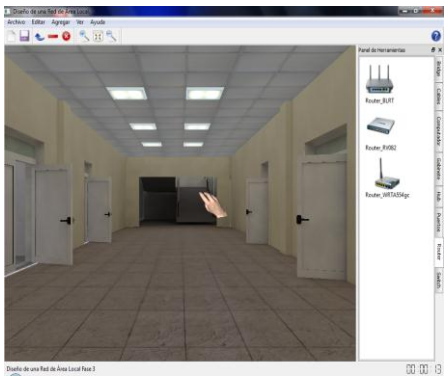
Una vez comenzado la práctica, la aplicación muestra los objetos, ya sea en la escena como en el panel de herramienta de acuerdo a las fases.



Fase 1:
Confección de un cable de red.



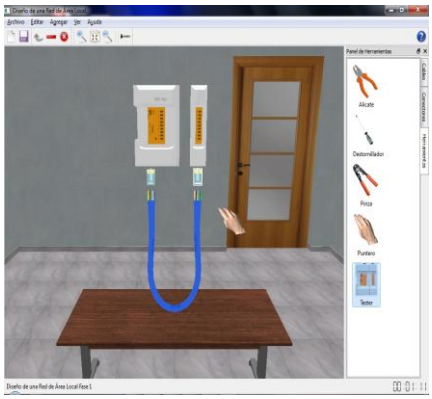
Fase 2:
Diseño de una red LAN en un entorno de dos dimensiones.



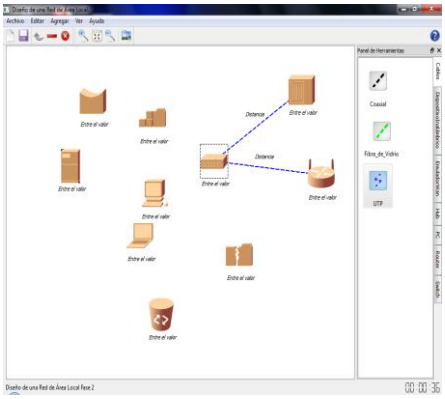
Fase 3:
Diseño de una red LAN en un entorno de tres dimensiones.

FIGURA 17. *ESCENA LIMPIA. DISEÑO E INSTALACIÓN DE UNA RED LAN.*

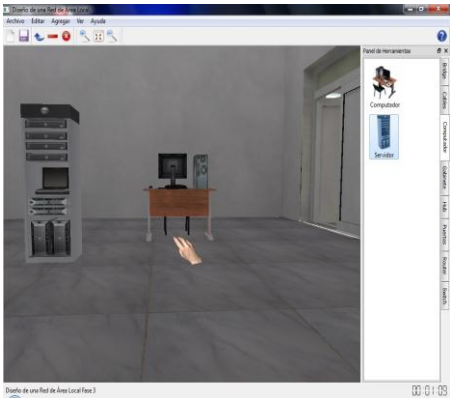
El usuario puede acceder a esta funcionalidad de la siguiente forma: La aplicación muestra en la escena los objetos previamente seleccionados en el Panel de Herramientas.



Fase 1:
Confección de un cable de red.



Fase 2:
Diseño de una red LAN en un entorno de dos dimensiones.



Fase 3:
Diseño de una red LAN en un entorno de tres dimensiones.

FIGURA 18. *OBJETOS CARGADOS EN LA ESCENA. DISEÑO E INSTALACIÓN DE UNA RED LAN.*

En el caso particular del Laboratorio Virtual Administración y Configuración de una Red LAN. La escena es cargada desde un fichero de configuración exportada por la fase 3, del Laboratorio Virtual Diseño e Instalación de una Red LAN.

4.2.3. Validación del caso de uso Mover objetos gráficos.

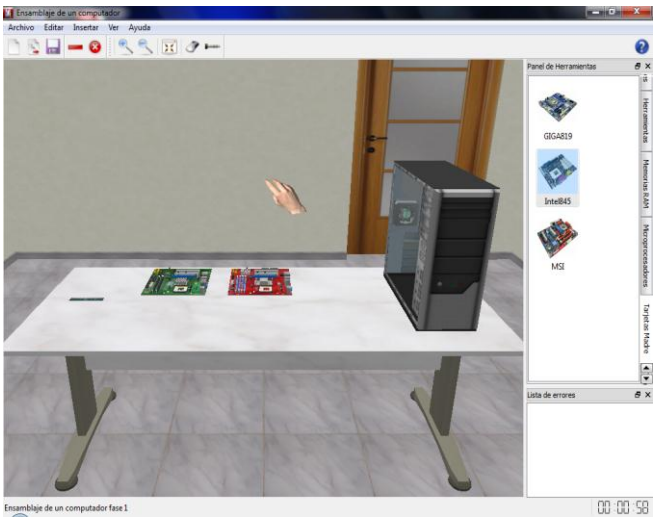
Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Mover objetos gráficos.	EC 1.1: Mover objetos gráficos.	La aplicación permite modificar la posición de los objetos de la escena sin alterar la estructura de la misma.

TABLA 12. *SECCIONES A PROBAR EN EL CASO DE USO MOVER OBJETOS GRÁFICOS.*

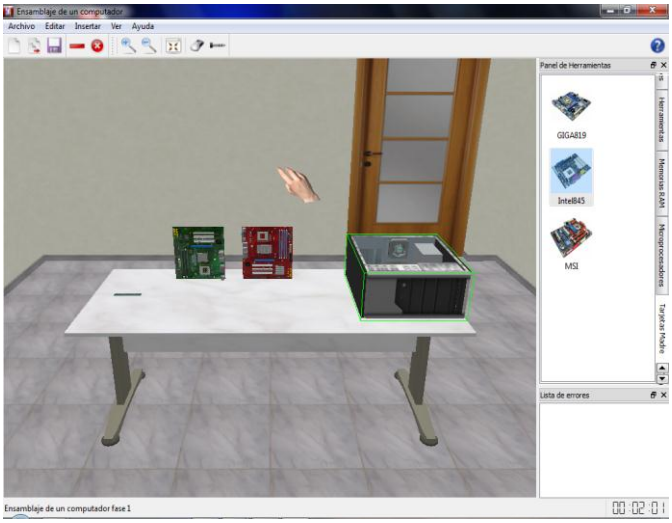
Descripción de las variables:

Objetos: Son aquellos objetos gráficos en dos dimensiones y tres dimensiones, que se encuentran en la escena, no admiten valores válidos

El usuario puede acceder a esta funcionalidad de la siguiente forma: Seleccionar los objetos que se encuentran en la mesa de trabajo y luego modificar la posición del objeto seleccionado.



Posición original de los objetos.



Posición de los objetos luego de ser movidos.

FIGURA 19. *OBJETOS MOVIDOS DURANTE LA FASE 1. ARQUITECTURA DE COMPUTADORAS.*

4.2.4. Validación del caso de uso Cargar archivo.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Cargar archivo	EC 1.1: Cargar archivo	La aplicación permitirá cargar un fichero.

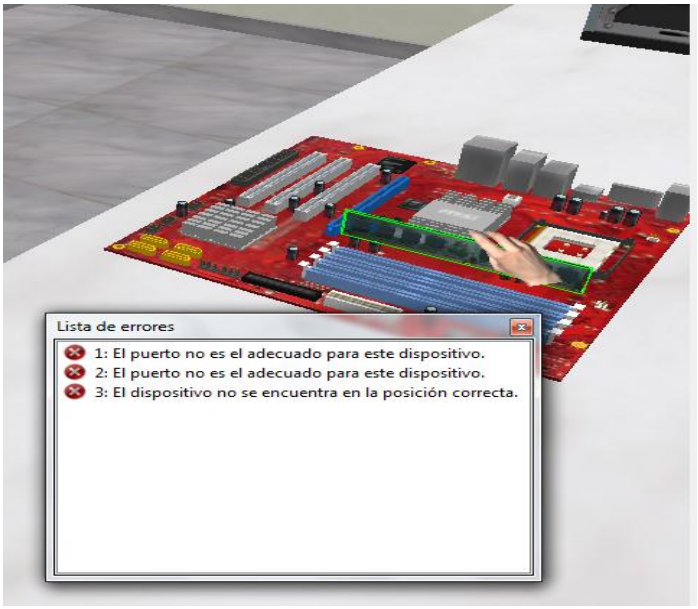
TABLA 13. *SECCIONES A PROBAR EN EL CASO DE USO MOSTRAR OBJETOS DE LA ESCENA.*

La aplicación muestra una ventana, en la cual va a poder buscar la dirección donde está guardado el archivo. El usuario puede acceder a esta funcionalidad de la siguiente forma: Primero localiza el fichero guardado, que va ser la escena de trabajo, luego selecciona y dar clic en la opción ***Abrir.***

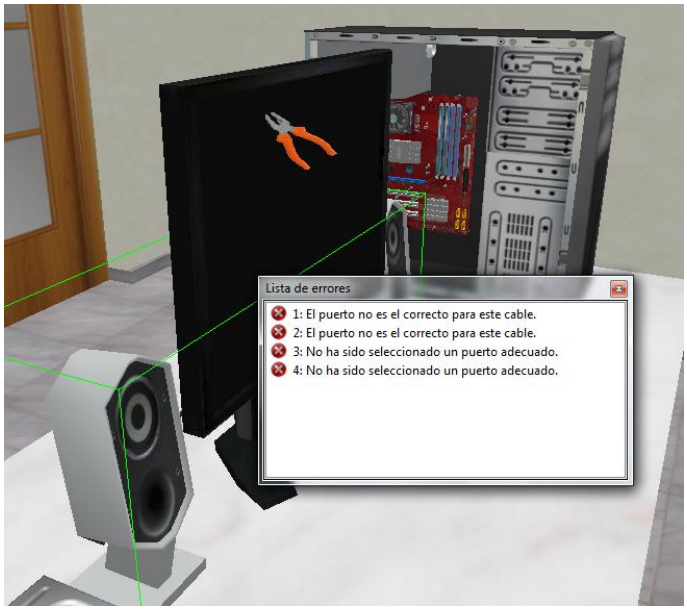
4.2.5. Validación del caso de uso Comprobar errores.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Comprobar errores	EC 1.2: Mostrar los errores cometidos por el usuario.	El usuario tendrá la posibilidad de observar los errores cometidos en la práctica, la aplicación lo reflejará en el reporte de la misma.

TABLA 14. *SECCIONES A PROBAR EN EL CASO DE USO MOSTRAR ERRORES COMETIDOS.*



Fase 1: Dispositivos internos.



Fase 2: Dispositivos externos.

FIGURA 20. *LISTA DE ERRORES COMETIDOS DURANTE LA PRÁCTICA. ARQUITECTURA DE COMPUTADORAS.*

4.2.6. Validación del caso de uso Mostrar tiempo de duración de la práctica.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Mostrar tiempo de duración de la práctica.	EC 1.1: Mostrar tiempo de duración de la práctica.	La aplicación muestra el tiempo de duración de dicha práctica desde el instante en que comienza.
	EC 1.2:	

TABLA 15. *SECCIONES A PROBAR EN EL CASO DE USO MOSTRAR TIEMPO DE DURACIÓN DE LA PRÁCTICA.*

Descripción de variable:

Tiempo: Este campo no permite valores nulos, y debe ser escrito en el siguiente formato: hh:mm:ss comenzando siempre por los valores 00:00:00.

Una vez que comienza la práctica, la aplicación muestra en el lado derecho inferior el tiempo en el siguiente formato (00:00:00), de esta manera el usuario puede acceder a esta funcionalidad.

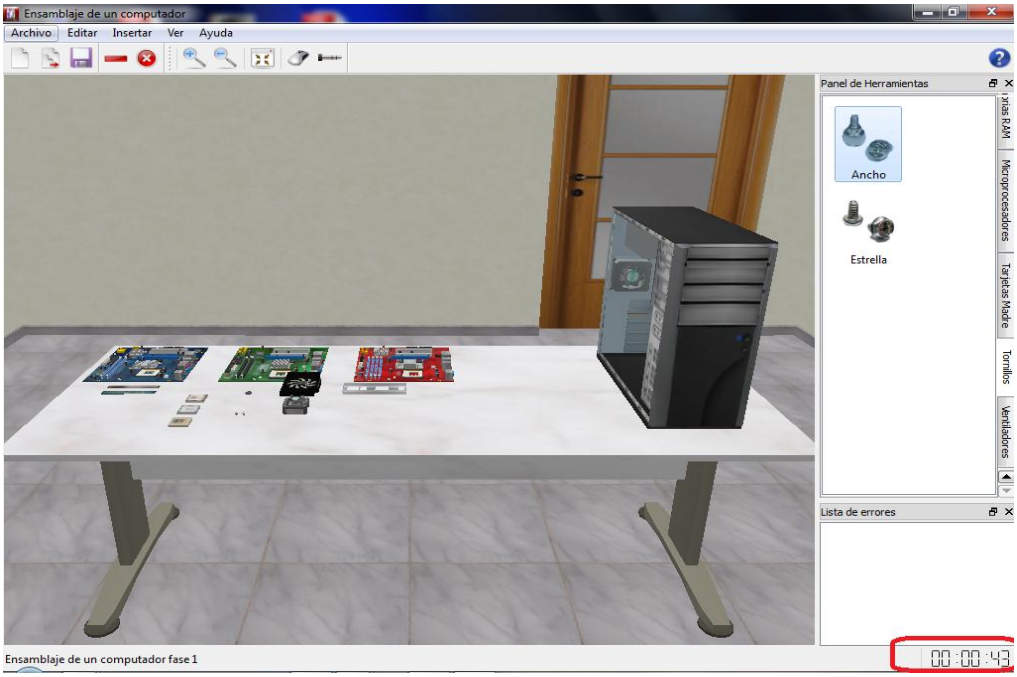


FIGURA 21. *CONTADOR DE TIEMPO PARA LA PRÁCTICA DE LABORATORIO. ARQUITECTURA DE COMPUTADORAS.*

4.2.7. Validación del caso de uso Realizar Zoom.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Acercar.	EC 1.1: Acercar	El usuario tendrá la posibilidad de acercar los objetos de la escena.
SC 2: Alejar.	EC 2.1: Alejar	El usuario tendrá la posibilidad de alejar los objetos de la escena.
SC 3: Restablecer.	EC 3.1: Restablecer	El usuario tendrá la posibilidad de restablecer la cámara en la posición inicial del laboratorio.

TABLA 16. *SECCIONES A PROBAR EN EL CASO DE USO REALIZAR ZOOM.*

Una vez que el usuario comience la práctica de laboratorio podrá alejar la escena. El usuario puede acceder a esta funcionalidad de la siguiente forma: Seleccionar en dependencia de lo que desea realizar la opción *Alejar o Acercar* ya sea desde el menú Ver o desde la barra de tareas.

4.2.8. Validación del caso de uso Guardar reporte final.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Guardar Reporte	EC 1.1: Salvar	El usuario tendrá la posibilidad de guardar el reporte de la práctica de laboratorio.
	EC 1.2:	

TABLA 17. *SECCIONES A PROBAR EN EL CASO DE USO GENERAR REPORTE FINAL.*

Una vez seleccionado la opción salvar, la aplicación muestra una ventana donde el usuario especifica el lugar donde desea guardar el reporte de la práctica y el nombre de la misma. El usuario puede acceder a esta funcionalidad de la siguiente forma: dar clic en la opción Salvar del menú Archivo o la opción Salvar de la barra de herramientas. Luego la aplicación muestra una ventana donde el usuario debe seleccionar la ubicación donde desee guardar el reporte y especificar el nombre del mismo, finalmente dar clic en la opción Guardar.

Conclusiones generales del capítulo.

En el presente capítulo se abordaron las temáticas de implementación, la cual fue mostrada mediante el diagrama de componentes, donde se brinda una información más detallada de la estructura de los componentes que conforman la interfaz de usuario de los laboratorios virtuales. Además se realizó la validación de dicha interfaz, con el método prueba por casos de uso, de esta manera se validó la funcionalidad del software y la respuesta de la interfaz de usuario a los distintos objetivos perseguidos con cada uno de los laboratorios virtuales desarrollados. Todas las pruebas realizadas a la interfaz de usuario arrojaron resultados satisfactorios, por lo tanto se decidió no realizar otra iteración de estas pruebas.

Conclusiones.

Con la realización de este trabajo de diploma se alcanzaron los objetivos del mismo, que fue desarrollar una interfaz de usuario, configurable y funcional que permita la interacción Hombre-Computadora para los laboratorios virtuales del proyecto PROLAVI, los cuales son de gran utilidad para el desarrollo de habilidades en los estudiantes para las distintas disciplinas y materias tratadas en estas laboratorio. La interfaz creada cumple con varios principios para el diseño de interfaz de usuario, y está desarrollado bajo el uso de buenas prácticas para el desarrollo de la misma. El sistema cumple con las políticas de Open Source, y se utilizaron para la elaboración del mismo, herramientas y bibliotecas libres.

La interfaz gráfica de usuario para los laboratorios virtuales le proporciona al usuario realizar las acciones que necesita durante todas las prácticas, además le permite a los mismo recoger los errores y generar un reporte de las tareas realizadas durante el mismo, registrándose además el tiempo que demoró en la realización del ejercicio, de esta manera le da una idea al estudiante o al profesor, de cuanto tiene que trabajar para consolidar sus conocimientos.

Esta interfaz fue integrada a los Laboratorios Virtuales liberados por el proyecto PROLAVI en la hermana República Bolivariana de Venezuela siendo de gran aceptación por el cliente que solicitó el desarrollo de los mismos. Propicio el desarrollo de un sistema es multiplataforma, puede ser ejecutado en Sistema Operativo Windows XP, Windows 7 y en plataformas libres GNU/Linux. Por lo que constituye una solución que tiene en cuenta la tendencia de nuestro país y el mundo fomentando el uso de software libre. Los Laboratorios Virtuales para los cuales fue desarrollada esta interfaz de usuario pueden ser utilizados también en carreras técnicas y universitarias, que traten las temáticas desarrolladas en las distintas prácticas, además pueden ser extensibles a otras materias.

Recomendaciones.

Con el desarrollo de la interfaz gráfica de usuario para los laboratorios virtuales se logró alcanzar el objetivo por el cual fue creada, es necesario seguir perfeccionando en el tema para que se logre una interfaz adaptable a todos los Laboratorios Virtuales que se le pueda presentar en el futuro al equipo de desarrollo, ya que los mismos son extensibles a varias materias dentro de la enseñanza utilizando las TICs.

Se recomienda para la misma, incluirle otras funcionalidades que mejoren la comunicación del usuario con estas herramientas educativas como por ejemplo, darle la oportunidad al usuario de modificar la interfaz en temas como colores, tipos de letras entre otros, que lo hagan sentirse dueño de la aplicación y que además pueda crear un perfil de configuración. Incluir dentro de los paneles en caso de que el usuario necesite conocer una descripción de los elementos contenidos en el mismo. Incluirle nuevas barras de herramientas que sean útiles para la navegación dentro de la escena, y un buscador al panel de que contiene los objetos para facilitar la búsqueda por nombres de los mismos en caso de que el usuario lo necesite.

Desarrollar una ayuda más interactiva, que responda al momento a las necesidades del usuario.

Valorar la inclusión los Laboratorios Virtuales desarrollados como herramientas didácticas para algunas asignaturas dentro de nuestra universidad y hacerlos extensibles por todas las universidades del país.

Glosario de términos.

3D: En computación, las tres dimensiones son el largo, el ancho y la profundidad de una imagen. Técnicamente hablando el único mundo en 3D es el real, la computadora sólo simula gráficos en 3D, pues, en definitiva toda imagen de computadora sólo tiene dos dimensiones, alto y ancho (resolución).

CASE (Computer Aided Software Engineering): Bajo el término de Ingeniería de Software Asistida por Ordenador se incluyen una serie de herramientas, lenguajes y técnicas de programación que permiten la generación de aplicaciones de manera semiautomática. Las herramientas CASE liberan al programador de parte de su trabajo y aumentan la calidad del programa a la vez que disminuyen sus posibles errores.

Framework: En el desarrollo de software, un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

GNU/GPL: Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License o simplemente su acrónimo del inglés GNU GPL, es una licencia creada por la Free Software Foundation a mediados de los 80, y está orientada principalmente a proteger la libre distribución, modificación y uso de software.

GNU/Linux: Es uno de los términos empleados para referirse a la combinación del núcleo o kernel libre similar a Unix denominado Linux, que es usado con herramientas de sistema GNU. Su desarrollo es uno de los ejemplos más prominentes de software libre; todo su código fuente puede ser utilizado, modificado y redistribuido libremente

GUI (Graphical User Interface): Sistema de interacción entre el ordenador y el usuario, caracterizado por la utilización de iconos y elementos gráficos en su concepción. Es un paso más allá de los interfaces basados en caracteres, que sólo incluían líneas de texto para introducir comandos y conocer las respuestas del sistema.

Interfaz de usuario: Engloba la forma en la que el operador interactúa con el ordenador, los mensajes que éste recibe en pantalla, las respuestas del ordenador a la utilización de periféricos de entrada de datos, etc.

Laboratorio Virtual: Sistema informático que pretende simular el ambiente de un laboratorio real y que mediante simulaciones interactivas permite desarrollar las prácticas de laboratorio.

Multiplataforma: Dicho de una aplicación o de un producto informático: Que puede ser utilizado por distintos sistemas o entornos.

TIC: Tecnologías de la Información y de la Comunicación conforman el conjunto de recursos necesarios para manipular la información, particularmente los ordenadores, programas informáticos y redes necesarias para convertirla, almacenarla, administrarla, transmitirla y encontrarla.

Widget: Abreviación de las palabras *window* y *gadget* (ventana y gadget o dispositivo, en inglés). En efecto, un *widget* es una mini-aplicación de ordenador que se presenta como una pequeña ventana o caja.

Citas Bibliográficas.

1. Silva., I. P. (s.f.). Reflexión sobre laboratorios virtuales. *Revista espacializada electrónica.*

2. Á. Salavarría, L. F. (julio de 2006). *Laboratorio Virtual para el autoaprendizaje de la electrónica aplicada.* Recuperado el 12 de Enero de 2011, de Universidad del País Vasco UPV/EHU.: http://www.euitt.upm.es/taee06/papers/SD/p90..

3. *alegsa*. (s.f.). Recuperado el 2011 de Febrero de 12, de http://www.alegsa.com.ar/Dic/interfaz.php

4. *La Interfaz*. (s.f.). Recuperado el Febrero de 19 de 2011, de http://www.lainterfaz.com

5. Núñez, Y. A. (Junio 2009). Adaptación de la Metodología Diseño y Desarrollo de Interfaz de Usuario Web para aplicarla al proyecto Contenidos Educativos Digitales. *Ciudad de la Habana*.

6. *Universia*. (2007). Recuperado el Marzo de 2 de 2011, de http://ocw.universia.net/es/tags/659/interfaz/

7. Mercovich, E. G. (2000). *Ponencia sobre Diseño de Interfaces y Usabilidad: cómo hacer productos más útiles, eficientes y seductores.* Recuperado el 12 de marzo de 2011, de http://www.gaiasur.com.ar/infoteca/siggraph99/diseno-de-interfaces-y-usabilidad.html

8. Sur, G. (2000). *Ponencia sobre Diseño de Interfaces y Usabilidad: cómo hacer productos más útiles, eficientes y seductores*. Recuperado el marzo de 18 de 2011, de http://www.gaiasur.com.ar/infoteca/siggraph99/diseno-de-interfaces-y-usabilidad.html

9. (2000). *La intersección entre factores humanos, diseño gráfico, interacción y comunicación.* Buenos Aires. Argentina.

10. Lewis-Rieman. (1993). *Task-centered user interface design.* Obtenido de http://hcibib.org/tcuid

11. Stephanidis. (2001). *User Interfaces for All.* Obtenido de http://books.google.com/books?id=kGslhSbZqzkC&pg=PA3&ots=5VCbLMr-5H&dq=Interfaces&sig=sjh97iBXzUoXmkTU2hJ0lPaSCQA

12. (30 de Noviembre de 2010). Obtenido de http://interfacesadaptativas.blogspot.com

13. Gómez, L. S. (s.f.). *Diseño de Interfaces de Usuario Principios, Prototipos y Heurísticas para Evaluación.* Obtenido de gomezsebastian@yahoo.com: http://sebastiangomez.sytes.net/papers/DIU.pdf

14. Yamila L. Verdecia Álvarez, L. G. (Junio 2010). Propuesta de Interfaz Gráfica de Usuario para el proyecto VISMEDIC. Ciudad de La Habana.

15. *Toolkit para Widgets.* (s.f.). Recuperado el Marzo de 23 de 2011, de http://eqaula.org/eva/file.php/1011/moddata/forum/301/2184/toolkits_para_Widgets.doc

16. Rocío Gutiérrez González, R. S. (s.f.). *Comparación entre las bibliotecas graficas GTK y Qt.* Recuperado el 10 de Diciembre de 2011, de http://yboon.net/~azamudio/linux/Comparacion%20entre%20las%20bibliotecas%20graficas%20GTk%20y%20Qt.pdf

17. Mark. (2010). Learning Python, Fourth Edition.

18. Sánchez., M. A. (s.f.). *Metodologías De Desarrollo De Software.* Recuperado el Marzo de 5 de 2011, de http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

19. Mtra. María de Lourdes Santiago Zaragoza. Desarrollando aplicaciones informáticas con el Proceso de Desarrollo Unificado RUP). [En Línea] [Citado: 8 de febrero del 2011] http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm

20. Ivar Jacobson, G. B. (s.f.). *El Proceso Unificado de Desarrollo de Software.*

21. Vázquez, L. M. (21 de Febrero de 21 Febrero 2008). Introduction to software engineering. Guatemala.

22. Penadés, P. L. (s.f.). Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). Universidad Politécnica de Valencia.

23. James Rumbaugh, Ivar Jacobson, Grady Booch. El Lenguaje Unificado de Modelado.

24. Kendall, K. y. (s.f.). Análisis y Diseño de Sistemas. S.l.: Prentice-Hall.

25. Hassán Lombera Rodríguez, A. E. (Julio 2008). Edición de sistemas articulados de cuerpos rígidos para las animaciones en los videojuegos. Cuidad de la Habana.

Bibliografía.

1. **Moure, M.J.** Virtual laboratory as a tool to improve the effectiveness of actual laboratories. [En línea]

2. *International Journal of Engineering Education.* 2, 2004, Vol. 20. 188-192.

3. **Interfaz de usuario.** http://www.fismat.umich.mx/~crivera/tesis/node6.html. [En línea] 23 de Mayo de 2000.

4. *GUI LNF Standards – DENR (Interact).* 22 Junio 1998.

5. Rena. Red escolar nacional. [En línea] http://rena.edu.ve.index.html.

6. *GUI LNF Standards - DENR (Presentation).* 22 Julio 1998.

7. *GUI LNF Standards – DENR (Structure).* 22 Julio 2008.

8. eleZeta la odisea del aprendiz. [En línea] http://elezeta.net/.

9. Latium Software. [En línea] 2000. http://www.latiumsoftware.com/es/developers/index.php.

10. *Estándares de Interfaz Gráfica.* **Roddy Del Carmen Ríos Acosta, Karen Marlene Pancorbo Méndez.** UNIVERSIDAD CATÓLICA DEL PERÚ FACULTAD DE CIENCIAS E INGENIERÍA : s.n.

11. Audoux, Pascal. Impressions on MFC vs Qt Programming. [En línea] http://phil.freehackers.org/index.html.

12. **Molpeceres, Alberto.** Procesos de desarrollo RUP, XP, FDD. [En línea] http://www.willydev.net/descargas/articulos/general/cualxpfddrup.PDF.

13. Herramientas Case. [En línea] http://jhoel-jp.tripod.com/paginas/herramientascase.pdf.

14. *Applying UML and Patterns.* **Hall, Craig Larman. Ed Prentice.** 27 Abril 2008.

15. *Definición del modelo del negocio y del dominio utilizando. Dapena.* **s.l., MSc. Martha D. Delgado.** s.l. : Centro de Estudios de Ingeniería de Sistemas.

16. *Etapa de requerimientos, Desarrollo de Software I.* **Valdez, Antonio.** Palmira : Universidad del Valle.

17. *Diseño de Casos de Prueba proyecto PROLAVI.* **Hechavarria, Sailyn Salas.** Cuidad de la Habana : s.n., 2011.

18. *Especificación de casos de uso del proyecto PROLAVI.* **Sailyn Salas Hechavarria, Brenda Batista Fons.** Cuidad de la Habana : s.n., 2011.

19. Pixelco blog Un blog sobre diseño y desarrollo web, Internet y tecnología. [En línea] 2011. http://pixelcoblog.com/qt-creator-completo-entorno-de-desarrollo-multiplataforma/..

20. Zator Systems. [En línea] 2011. http://www.zator.com/Cpp/E1_2.htm.

21. Word Press. [En línea] 2000. http://mredison.wordpress.com/2007/12/02/caractersticas-de-visual-studio-2008/.

22. *GUI LNF Standards – DENR (Structure).* 22 Julio 1998.