ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA.

ESCUELA DE INGENIERÍA EN SISTEMAS

DESARROLLO DE UNA HERRAMIENTA WEB DE APOYO A LA EJECUCIÓN DE EXPERIMENTOS EN PRUEBAS DE SOFTWARE.

Trabajo de titulación

Tipo: Proyecto técnico

Previa la obtención del título de:

INGENIERO EN SISTEMAS INFORMÁTICOS

AUTORA: GLORIA JIMENA HERNÁNDEZ GUATO

TUTOR: DR. OMAR GÓMEZ GÓMEZ

Riobamba-Ecuador

2018

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA EN SISTEMAS

El Tribunal del trabajo de titulación certifica que el proyecto técnico: "DESARROLLO DE UNA HERRAMIENTA WEB DE APOYO A LA EJECUCIÓN DE EXPERIMENTOS EN PRUEBAS DE SOFTWARE", de responsabilidad de la estudiante Gloria Jimena Hernández Guato, ha sido minuciosamente revisado por los Miembros del Tribunal del trabajo de titulación, quedando autorizada su presentación.

NOMBRES	**FIRMAS**	**FECHA**
Ing. Washington Luna **DECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**	_____	_____
Ing. Patricio Moreno **DIRECTOR DE LA ESCUELA DE INGENIERÍA EN SISTEMAS**	_____	_____
Dr. Omar Gómez Gómez **DIRECTOR DEL TRABAJO DE TITULACIÓN**	_____	_____
Ing. Raúl Rosero Miranda **MIEMBRO DEL TRIBUNAL DEL TRABAJO DE TITULACIÓN**	_____	_____

AGRADECIMIENTO

Agradezco a Dios y a mi madre por brindarme sus bendiciones y apoyo incondicional, a mi hermana y amigas por estar conmigo cuando las necesité, a mis pequeñas hijas y a mi amado esposo por su paciencia y cariño, a la ESPOCH institución que me abrió sus puertas y a mis queridos maestros quienes permitieron mi formación académica, de manera especial al director de mi trabajo de titulación, que con sus conocimientos fue una guía en el desarrollo del mismo; pues gracias a todos ellos he luchado constantemente para cumplir mis metas, y hacer posible este sueño.

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE GRÁFICOS

ÍNDICE DE ANEXOS

ÍNDICE DE ABREVIATURAS

SW Software

HW Hardware

BD Base de datos

IO Entrada/Salida

SYSTEXP Sistema de experimentación

PHP Personal home page

MVC Modelo Vista Controlador

HTTP Protocolo de transferencia de hipertexto

HTML Lenguaje de marcas de hipertexto

IDE Entorno de desarrollo integrado

PDF Formato de documento portátil

IS Ingeniería de Software

ISO Organización Internacional de Normalización

RESUMEN

El objetivo del presente trabajo fue implementar el sistema web "SystExp" como herramienta de apoyo a la ejecución de experimentos en Ingeniería de Software (IS), en el área de las pruebas de software. En el sistema se implementaron las técnicas de lectura de código por abstracciones sucesivas, pruebas funcionales o caja negra y pruebas estructurales o caja blanca. Para desarrollar el sistema se aplicó la metodología ágil SCRUM, utilizando las tres fases establecidas como son: la fase de planificación, la de desarrollo y la fase de cierre. Con respecto a la primera etapa se definieron los requerimientos funcionales del sistema, los mismos que fueron documentados usando los diagramas de casos de uso. En la segunda etapa se desarrolló un total de 15 Sprints con el fin de cumplir los requerimientos del cliente. Finalmente en la fase de cierre se evaluó la gestión del proyecto mediante un BurnDown Chart. La evaluación de la usabilidad del sistema se realizó mediante dos estudios pre-test y post-test en dos grupos distintos de estudiantes de sexto y noveno semestre de la Escuela de Ingeniería en Sistemas de la ESPOCH. Se evaluó la usabilidad del sistema conformada por los siguientes aspectos: utilidad, facilidad de uso, facilidad de aprendizaje y el nivel de satisfacción. Después de realizar un análisis inferencial se observó una diferencia significativa a favor de la facilidad de uso para aquellos estudiantes que utilizaron el aplicativo desarrollado. En general se observa que el desarrollo de esta herramienta facilita la creación y ejecución de experimentos en el ámbito de la pruebas de software es por ello que se recomienda el uso de la herramienta.

Palabras clave: <TECNOLOGÍA Y CIENCIAS DE LA INGENIERÍA>, <INGENIERÍA DE SOFTWARE>**,** <PRUEBAS DE EXPERIMENTACIÓN> <TÉCNICA DE LECTURA DE CÓDIGO POR ABSTRACCIONES SUCESIVAS> <PRUEBA DE LA CAJA NEGRA> <PRUEBA DE LA CAJA BLANCA> <CASOS DE USO> <HERRAMIENTA DE EXPERIMENTACIÓN>

SUMMARY

The objective of this work is to implement a web system known as "'SystExp", wich is a tool to support the execution of experiments in Software Testing from the Software Engineering (IS) discipline. In this system, reading code technique by successive abstractions, functional testing (or black box testing) and structural testing (or white box testing) techniques were implemented. In order to develop the system, the agile SCRUM methodology was applied, using the three established phases such as the planning phase, the development phase and the closing phase. With respect to the first stage, the functional requirements of the system were defined, which were documented using use case diagrams. In the second stage, 15 Sprints were developed in order to meet the customer's requirements. Finally, in the closing phase, the management of the project was evaluated through a BurnDown Chart. The evaluation of the usability of the developed system was carried out through two pre-test and post-test empirical studies applied in two different groups of sixth and ninth-semester students of the School of Systems Engineering of ESPOCH, respectively. The usability of the system was evaluated conformed by the following aspects: utility, ease of use, ease of learning and the level of satisfaction. After performing an inferential analysis, a significant difference was observed in favor of the ease of use for those students who used the developed web system. In general, it is observed that the development of this tool facilitates the creation and execution of experiments in the field of software testing, that is why the use of the web tool is recommended.

Keywords: <TECHNOLOGY AND ENGINEERING SCIENCES>, <SOFTWARE ENGINEERING>, <SOFTWARE TESTING TOOL>, <CODE READING TECHNIQUE>, <BLACK BOX TESTING>, <WHITE BOX TESTING>, <EXPERIMENTAL SOFTWARE ENGINEERING>

INTRODUCCIÓN

Actualmente, hay una creciente comprensión en la Ingeniería de Software, que los estudios empíricos son necesarios para desarrollar o mejorar procesos, métodos y herramientas para el desarrollo y mantenimiento de software. (Basili, 1996).

La experimentación en ingeniería de software, permite identificar las relaciones causa-efecto, mediante la realización de experimentos controlados donde sólo unas pocas variables cambian. Experimentos controlados en Ingeniería de software involucran a menudo a los estudiantes que resuelven tareas de papel en un entorno de aula. (Potts, 1993).

Como apoyo a la experimentación en ingeniería de software ya existen algunas herramientas que se han desarrollado las cuales se encuentran en el mercado como: SimGrid, Sese y Marmoset.

SimGrid es un conjunto de herramientas que proporciona funcionalidades básicas para la simulación de aplicaciones distribuidas en entornos distribuidos heterogéneos. El objetivo específico del proyecto es facilitar la investigación en el área de sistemas paralelos y distribuidos que sean a gran escala, tales como Grids, sistemas P2P y Cloud (Casanova, Legrand, & Quinson, 2008).

SESE es una herramienta para ejecutar experimentos controlados realistas, para evaluar empíricamente las tecnologías de ingeniería de software. El realismo puede ser incrementado utilizando profesionales además de estudiantes, herramientas reales de desarrollo, en lugar de papel y lápiz, tareas más grandes y un entorno de trabajo típico en lugar de un aula (Arisholm, Sjoberg, Carelius, & Lindsjorn, 2011).

Marmoset, es una herramienta que apoya a los docentes de informática que mantienen fuertes opiniones sobre el enfoque correcto, para la enseñanza de la programación a nivel introductorio (Spacco, et. al. 2005).

Una de las áreas en ingeniería de software, donde se han realizado experimentos son las pruebas de software, las cuales están diseñadas para realizar la detección de defectos de forma estructurada y facilitar el estudio de software.

Las pruebas de software ayudan a proporcionar información objetiva e independiente sobre la calidad del producto software (Zelkowitz & Wallace, 1998).

Algunas técnicas para aplicar pruebas de software son: La lectura de código por abstracciones sucesivas, pruebas funcionales "caja negra" y pruebas estructurales "caja blanca".

Aunque existen algunas herramientas de soporte a la experimentación en ingeniería de software, no se han encontrado herramientas de apoyo a la comparación de técnicas de pruebas de software, es por ello que los experimentos en esta área suelen realizarse de forma manual.

Al realizar experimentos de software de manera manual, existe una demora de tiempo y la falta de control en la recolección de métricas, por lo que no se conoce como influye el uso de un sistema web para apoyar la realización de experimentos en el área de pruebas de software.

FORMULACIÓN DEL PROBLEMA

¿El uso de una herramienta web para ejecutar experimentos en el área de las pruebas de Ingeniería de software (IS) agilita el proceso de la recolección de métricas?

SISTEMATIZACIÓN DEL PROBLEMA

Para la sistematización del problema se propone desarrollar un sistema web para apoyar la realización de experimentos en el área de las pruebas de software, por lo cual es necesario realizar las siguientes preguntas:

- ¿Qué es la experimentación en ingeniería de software?
- ¿Qué herramientas de soporte a la experimentación en ingeniería de software existen?
- ¿Qué son las pruebas de software?
- ¿Qué técnicas de pruebas de software son más comunes?

- ¿Cuáles son las variables y métricas de las técnicas de pruebas de software que se recolectarán?
- ¿Existe alguna familia de experimentos en el ámbito de las pruebas de software?

JUSTIFICACIÓN DEL TRABAJO DE TITULACIÓN

JUSTIFICACIÓN TEÓRICA

La ingeniería de software no es diferente de otras ciencias, este campo ha ido evolucionando para adoptar aproximaciones experimentales (Zelkowitz & Wallace, 1998), y la ingeniería de software está preocupada sobre los modelos apropiados para ser aplicados al desarrollo de sistemas de software (Hochstein, et. al. 2007), mientras más grandes son los sistemas de software se requiere de varios programadores, profesionales, largos periodos de tiempo, esto significa grandes cantidades de recursos económicos, es por ello que a través de la experimentación en Ingeniería de Software, se puede tener mayor conocimiento sobre el proceso de desarrollo de software.

El aula es un entorno donde se puede conducir experimentos en ingeniería de software, por varias razones:

- La mayoría de investigadores están localizados en universidades.
- La formación del curso puede ser integral.
- Las tareas requeridas pueden ser fácilmente integradas.
- Los temas están relacionados en tareas de programación idénticas.

Los resultados obtenidos directamente de estos estudios, son útiles para determinar reglas y diseños experimentales, los cuáles más tarde pueden ser aplicados a temas de la industria.

Para la realización del Sistema Web se ha utilizado el lenguaje de programación PHP, AJAX (Asynchronous JavaScript And XML) que es una técnica de desarrollo web para crear aplicaciones interactivas o RIA, además se utilizó la metodología SCRUM.

JUSTIFICACIÓN METODOLÓGICA

Se conoce la existencia de varias metodologías web, sin embargo al desarrollar una aplicación no siempre se cuenta con una metodología adecuada a pesar de su existencia, es por este motivo que se aplicará la metodología SCRUM.

SCRUM es una metodología ágil que se usa para minimizar los riesgos durante la realización de un proyecto, pero de manera colaborativa, con entregas mensuales o quincenales de resultados, los requisitos más prioritarios en ese momento, se completan lo cual proporciona las siguientes ventajas:

- Gestión regular de las expectativas del cliente y basada en resultados tangibles.
- Resultados anticipados (time to market).
- Flexibilidad y adaptación respecto a las necesidades del cliente, cambios en el mercado, etc.
- Gestión sistemática del Retorno de Inversión (ROI).
- Mitigación sistemática de los riesgos del proyecto.
- Productividad y calidad.
- Alineamiento entre el cliente y el equipo de desarrollo.
- Equipo motivado**.** (Schwaber, 2004)

JUSTIFICACIÓN APLICATIVA

Se ha detectado la necesidad de realizar una aplicación web, en apoyo a la experimentación en ingeniería de software, se va automatizar la aplicación de pruebas de software.

El principal beneficio es disminuir el tiempo e incrementar el uso de una herramienta que apoyará a los experimentadores e investigadores que lleven a cabo experimentos en el área de las pruebas de software.

A continuación se muestra los módulos del sistema web de apoyo a la ejecución de experimentos en el área de IS.

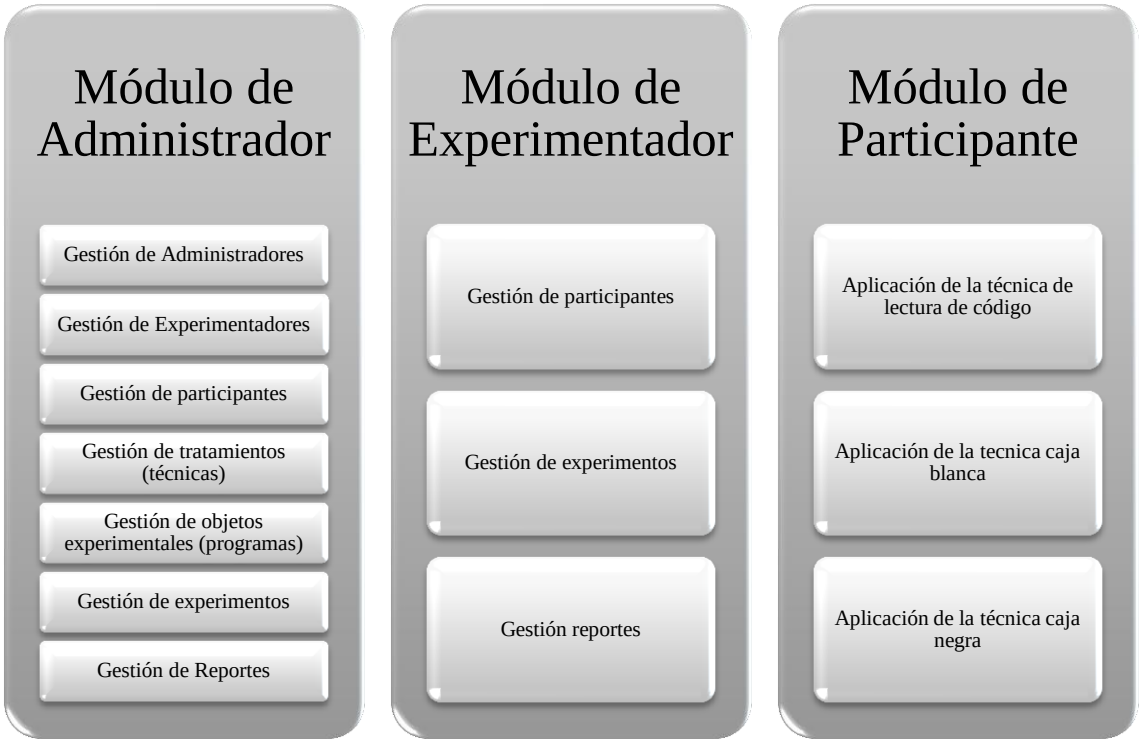

Figura 1: Módulos del sistema.

Realizado por: G. Hernández 2018

El sistema web tiene tres módulos: administrador, experimentador y participante. Un administrador debe ser un usuario experto porque puede gestionar a otros administradores, experimentadores y participantes. Además ingresa los tratamientos (técnicas) y los objetos experimentales (programas) para crear experimentos.

El experimentador tiene solamente el control sobre sus experimentos y los participantes que están dentro de sus prácticas experimentales. En el sistema existen dos tipos de diseño experimental, el primero se denomina diseño factorial el mismo que consiste en la asignación de un tratamiento y un objeto experimental para que sea desarrollado por el participante en una sola sesión. Medidas repetidas es el segundo diseño experimental, consiste en asignar un solo tratamiento y varios objetos experimentales, se realiza en varias sesiones como máximo tres.

Una vez definidas las variables del experimento, los participantes podrán ingresar al sistema y deberán aplicar alguna de las técnicas asignadas de manera aleatoria, las variables y métricas recolectadas se guardan en una base de datos para posteriormente ser analizadas por el experimentador.

En el módulo de participantes se localizan los tratamientos. Actualmente en el sistema se encuentra implementadas tres técnicas como son: lectura de código por abstracciones sucesivas, caja blanca o de cristal y caja negra.

OBJETIVOS

Objetivo general

Implementar un sistema web en apoyo a la experimentación en ingeniería de software para la realización de experimentos en el área de pruebas de software.

Objetivos específicos

1. Investigar sobre la experimentación en ingeniería de software para el desarrollo del sistema.
2. Determinar las variables y métricas de las técnicas de pruebas de software que se recolectarán a través de los formularios automatizados.
3. Investigar las tres técnicas de pruebas de software: lectura de código por abstracciones sucesivas, caja negra y caja blanca que se incorporará en el sistema web.
4. Evaluar la usabilidad del sistema web.

CAPÍTULO I

1. MARCO TEÓRICO DE REFERENCIA

1.1. La experimentación en ingeniería de software

La organización del conocimiento ha dejado tratar niveles más altos de abstracción del problema y el espacio de solución, dicho conocimiento está basado en la retroalimentación y el aprendizaje aplicando experimentos y analizando los resultados. La experimentación ha sido utilizada en muchos campos, p. ej., física, medicina, fabricación.

Un ejemplo muy claro es dentro del área de la medicina, donde los objetivos del investigador es comprender el funcionamiento del cuerpo humano, mediante la experimentación se conoce los efectos de los fármacos y así se proporciona el conocimiento necesario.

Sin embargo los primeros investigadores aplicaron varios procesos a hierbas y entregaron el conocimiento a menudo en secreto que pasó de generación a generación. La medicina como campo no progreso hasta que "la aplicación de medicaciones y procedimientos formó una base para evolucionar el conocimiento de la relación entre los efectos y la solución" (Reggia, 1982).

La experimentación permite el control en casos de estudio, porque se construyen los modelos con capacidades predictivas y así entender la relación entre el proceso y el producto (Basili, 1996).

La ingeniería de software no difiere mucho de otras ciencias por lo que también se puede aplicar dentro de un laboratorio, esto implica un componente experimental para comprobar o rectificar teorías, al explorar nuevos ámbitos. La función del investigador es entender la naturaleza de los procesos y productos, y la relación entre ellos. El

experimento es basado en técnicas para proporcionar una visión del trabajo real y responde al cómo, cuándo y los límites.

Actualmente en la ingeniería de software es imprescindible conocer para el desarrollo de nuevas soluciones tecnológicas, el cómo hacer, cuánto tiempo tardará y cuánto va a costar el sistema, esto es un problema cuando no contamos con investigaciones previas, es por ello que la experimentación en ingeniería de software permite conocer de forma más rigurosa el proceso de desarrollo software. Un ejemplo podría ser el estudio de mejoras a los métodos que son utilizados en el desarrollo de software o la prueba de alguna herramienta nueva.

Los estudios empíricos en ingeniería de software permiten ampliar o mejorar procesos, métodos y herramientas para el desarrollo y mantenimiento de software (Basili, 1996).

El método clásico para identificar las relaciones causa-efecto, es conducir experimentos controlados donde sólo unas pocas variables cambian. "Experimentos controlados en ingeniería de software involucran a menudo a los estudiantes que resuelven tareas de papel en un entorno de aula". (Basili, Selby, & Hutchens, 1986)

Los experimentos en ingeniería de software serían más reales si se ejecutarán tareas reales, con sistemas reales y con profesionales que utilicen herramientas de apoyo, en dicho ambiente.

1.2. Proceso de experimentación

La realización de experimentos realistas requiere una buena gestión de los procesos, es por ello que algunos investigadores en IS han definido estos procesos que constan de varias actividades. En IS puede dividirse en cuatro actividades principales que son: definición, diseño o planificación, ejecución y análisis.

En la **Figura 2-1** se representan las actividades principales del proceso de experimentación en IS, donde cada actividad se realiza de manera secuencial (Gómez, Ucán, & Gómez, 2013).

Figura2-1: Actividades del proceso de experimentación en IS, adaptado de (Gómez, Ucán, & Gómez, 2013)

Realizado por: G. Hernández 2018

A continuación se describe cada actividad de experimentación en IS.

1.1.1. Definición

Esta actividad establece de manera general los aspectos principales del experimento a realizar así como en definir las hipótesis de trabajo. En esta fase es posible usar el método GQM (Goal-QuestionMetric) que facilita definir: el objeto de estudio del experimento, el propósito, el enfoque de calidad, la perspectiva así como el contexto donde se realiza el experimento. (Basili, Caldiera, & Rombach, 1994).

1.1.2. Diseño o Planificación

Cuando se ha definido la hipótesis la siguiente fase de este proceso consiste en diseñar cómo se llevará a cabo el experimento. En esta actividad se especifica cómo se asignan los tratamientos a los sujetos, para asegurar que los sujetos funcionen correctamente y que los datos son apropiados, el investigador deberá planificar como monitorear el progreso de cada tema.

Dentro de las actividades se preparan los instrumentos o materiales para realizar o ejecutar el experimento. Respecto a cómo asignar los tratamientos a los sujetos, para los diferentes tipos de diseños experimentales existe en la literatura un claro ejemplo mostrado por Juristo y Moreno (2001), que satisface los propósitos particulares.

1.1.3. Ejecución

Una vez definido el experimento y seleccionado algún diseño, es momento de llevar a cabo el experimento. En esta fase se debe tener listo los materiales a utilizar, los equipos, así como el espacio donde se llevará a cabo la sesión del experimento. Antes de comenzar los sujetos reciben una serie de instrucciones, conjuntamente con los materiales para que trabajen durante la sesión del experimento. (Basili, Caldiera, & Rombach, 1994)

1.1.4. Análisis

Una vez ya ejecutado el experimento, el investigador recolecta una serie de métricas generadas por los sujetos al terminar las tareas, sus resultados son recogidos y almacenados en un lugar seguro. Cuando todos los temas hayan terminado, el investigador puede iniciar el análisis (Gómez, Ucán, & Gómez, 2013).

En esta actividad del proceso, las métricas son analizadas y contrastadas con las hipótesis previamente definidas. En esta actividad se emplean diferentes técnicas estadísticas para analizar e interpretar las métricas obtenidas. Usualmente en el diseño de experimentos se emplea el análisis de la varianza (Fisher, 1918) (en inglés, ANOVA).

Básicamente el ANOVA consiste en examinar la variabilidad de las métricas obtenidas con respecto a los tratamientos aplicados, así como en examinar la variabilidad de otros tipos de variables como pueden ser variables de bloqueo o covariables. (Gómez, Ucán, & Gómez, 2013). El ANOVA suministra una prueba estadística acerca de si las medias o promedios de varios grupos son o no equivalentes. Una vez concluidos los análisis se puede proceder con la presentación y empaquetado.

Este consiste en documentar y presentar los hallazgos encontrados en el experimento. No obstante esta actividad puede considerarse opcional y prescindir.

1.3. Herramientas de apoyo a la experimentación en ingeniería de software.

La experimentación en ingeniería de software ha permitido desarrollar técnicas y métodos para desarrollar software de calidad, pero es necesario el uso de herramientas, que permiten replicar varias veces los experimentos y obtener conclusiones que ayuden a los investigadores. Actualmente ya existen algunas herramientas de apoyo a la experimentación en ingeniería de software entre las que se destacan: Singrid, SESE y Marmoset.

1.3.1. SINGRID

Es posible llevar a cabo un gran número de experimentos consecutivos y obtener resultados teóricos o analíticos para comparar el rendimiento de algoritmos dirigidos a sistemas.

El simulador SimGrid se inició en 1999 y permite el estudio de algoritmos de programación para plataformas heterogéneas. SimGrid v1, facilitó el prototipo de programación heurística y probarlos en una variedad de aplicaciones (expresados como gráficos de tareas) y plataformas. En el 2003, SimGrid v2, amplió las capacidades de su predecesor en dos maneras principales. En primer lugar, el realismo del motor de simulación se mejoró mediante la transición desde un modelo de agujero de gusano a un modelo analítico. En segundo lugar, una se agregó la API para estudiar la programación no centralizada y otro tipo de procesos secuenciales concurrentes (Casanova, Legrand, & Quinson, 2008)

1.3.2. SESE

Simulador de entorno para soporte de un experimento" (SESE), se construyó sobre un producto comercial estándar de KompetanseWeb, que es utilizado por varias organizaciones noruegas. SESE fue y sigue siendo desarrollado a través de un estrecho contacto entre Simula Research Laboratory (SRL) y KompetanseWeb.

SESE es una herramienta para ejecutar experimentos controlados realistas, para evaluar empíricamente las tecnologías de ingeniería de software. El realismo puede, por

ejemplo, ser incrementado utilizando profesionales además de estudiantes, herramientas reales de desarrollo en lugar de papel y lápiz, tareas más grandes y un entorno de trabajo típico en lugar de un aula. La logística de ejecutar experimentos realistas es mucho más compleja que para los experimentos sencillos de estudiantes que emplean lápiz y papel (Arisholm, et. al. 2011).

Un experimento realizado con SESE incluyó 130 profesionales en desarrollo de Java, nueve de asesoría de diferentes compañías y 60 alumnos por tema. El experimento tuvo lugar durante un periodo de dos meses y estuvo organizado en 12 sesiones separadas por un día, con diferentes participantes. Durante el día de la sesión por cada tema, tuvieron que solucionar seis tareas de programación en su ordenador que utiliza la herramienta de desarrollo Java. Mientras este experimento era de escala más grande y en muchas maneras más realistas que la mayoría de experimentos de ingeniería del software, aparecieron nuevos retos:

- El material experimental como cuestionarios, descripciones de tarea, código y herramientas, tuvo que ser distribuido a cada tema de modo oportuno.
- El diseño experimental es decir todo el material no podía ser distribuido inmediatamente, porque cada desarrollador profesional estaba sentado en su oficina habitual y debían participar del experimento, el problema se centró en controlar el progreso individual que era crucial.
- Los resultados como respuestas a cuestionarios y soluciones de programa, tuvieron que ser almacenado para futuros análisis.

El estudio implicaba correr los experimentos y es lo que motivó la necesidad de una herramienta que podría automatizar algunos de los procesos, por lo tanto, se desarrolló el sistema web SESE.

SESE permitió definir experimentos, incluyendo todo el detalle de cuestionarios, descripciones de tareas y código necesario, para asignar los temas a una sesión de experimento, con el objetivo de controlar cada sesión del experimento y recoger los resultados de cada tema para futuros análisis.

Otro de los objetivos que buscaban los investigadores con SESE es hacer experimentos de ingeniería de software reales y así posiblemente generalizar los resultados a una práctica industrial a escala mundial.

1.3.3. Marmoset

Sistema automatizado de envío y pruebas de software (Marmoset) es una herramienta que permite a los docentes de informática involucrar a los estudiantes novatos al desarrollo de software. Los estudiantes envían versiones de sus proyectos a un servidor central, que automáticamente los prueba y registra los resultados.

Marmoset recolecta copias de código mientras los estudiantes trabajan en los proyectos y presenta un esquema de datos que permite una gran variedad de consultas. (Spacco, et.al, 2005)

1.4. Pruebas de software

Las pruebas de software están diseñadas para realizar la detección de defectos de forma estructurada y facilitar el estudio de software. En las pruebas de software no se quita ninguna información después de que los sujetos la reciben. Existen varias técnicas de pruebas de software entre las que se destacan: la lectura de código por abstracciones sucesivas, caja negra y caja blanca. (Kamsties & Lott, 1995)

1.4.1. Lectura del código por abstracciones sucesivas.

En la lectura de código por abstracciones sucesivas, no se utiliza el computador. Se inicia cuando los sujetos ven el código fuente impreso, pero no ven las especificaciones. “Los sujetos leen el código fuente y escriben sus propias especificaciones del código, esto es basado en la técnica de lectura por abstracciones sucesivas” (Linger, Mills, & Witt, 1979)

Los sujetos identifican subprogramas principales (líneas consecutivas de código), escriben una especificación para el sub-programa lo más apropiado, agrupan las

abstracciones y sus especificaciones, este proceso se repite hasta que hayan abstraído todo el código fuente.

Después de escribir sus propias abstracciones, los sujetos desarrollan una especificación aproximada y finalmente reciben la especificación real a cambio y comienzan con la detección de fallas.

En la detección de fallas, los sujetos comparan la especificación real con la suya para observar las inconsistencias, entre el comportamiento especificado y el esperado del programa. Los sujetos aíslan los fallos de inconsistencia, para ello no se especifica ninguna técnica especial en el aislamiento de fallos. Finalmente, los sujetos hacen una lista de inconsistencias identificadas y fallas aisladas (Kamsties & Lott, 1995).

1.4.2. Pruebas funcionales o caja negra.

Se la denomina prueba de comportamiento, se basa en la especificación del programa o componente a ser probado para elaborar los casos de prueba. El componente se ve como una "caja negra" el comportamiento puede ser determinado estudiando sus entradas y las salidas obtenidas a partir de ellas (Beizer, 1990).

El estudio de todas las posibles entradas y salidas de un programa es impracticable por lo que se selecciona un conjunto de ellas sobre las que se realizan las pruebas, para seleccionar el conjunto de entradas y salidas sobre las que se va a trabajar, es imprescindible tener en cuenta que en todo programa existe un conjunto de entradas que causan un comportamiento erróneo en el sistema, y como consecuencia producen una serie de salidas que revelan la presencia de defectos. (Gómez, et.al, 2014)

Entonces, dado que la prueba exhaustiva es imposible, el objetivo final es encontrar una serie de datos de entrada cuya probabilidad de pertenecer al conjunto de entradas que causan dicho comportamiento erróneo sea lo más alto posible (Beizer, 1990).

Los casos de prueba de caja negra pueden especificarse con distintos criterios, algunos de ellos son:

- **Particiones de clase de equivalencia.**

Una clase de equivalencia representa un conjunto de estados válidos o no válidos para condiciones de entrada. Usualmente una condición de entrada es un valor numérico determinado, un rango de valores, un conjunto de valores relacionados o una condición lógica.

- **Análisis de Valores Límite (AVL)**

Es una técnica de diseño de casos de prueba que complementa a la partición equivalente. Los errores tienden a darse más en los límites del campo de entrada que en el centro, es por ello que se debe elegir los valores límites con (AVL) y esto conlleva a la elección de casos de prueba en los extremos de la clase con el objetivo de no seleccionar cualquier elemento de una clase de equivalencia. El AVL obtiene casos de prueba también para el campo de salida, en lugar de centrarse solamente en las condiciones de entrada.

1.4.3. Pruebas estructurales o caja blanca.

Es también conocida como técnica de caja transparente o de cristal (Beizer, 1990). Esta técnica se centra en cómo diseñar los casos de prueba atendiendo al comportamiento interno y la estructura del programa. En el programa se examina la lógica interna sin considerar los aspectos de rendimiento. La técnica tiene como objetivo diseñar casos de prueba para que se ejecuten, al menos una vez, todas las sentencias del programa, y todas las condiciones tanto en su vertiente verdadera como falsa (Beizer, 1990).

En algunos programas puede ser impracticable realizar una prueba exhaustiva de todos los caminos del mismo, es por ello que se han definido distintos criterios de cobertura lógica, que permiten decidir qué sentencias o caminos se deben examinar con los casos de prueba.

Estos criterios son:

- **Cobertura de sentencias.** Se escriben casos de prueba suficientes para que cada sentencia en el programa se ejecute, al menos, una vez.
- **Cobertura de decisión**. Se escriben casos de prueba suficientes para que cada decisión en el programa se ejecute una vez con resultado verdadero y otra con el falso.
- **Cobertura de condiciones.** Se escriben casos de prueba suficientes para que cada condición en una decisión tenga una vez resultado verdadero y otra falso.
- **Cobertura de condición múltiple.** Se escriben casos de prueba suficientes para que todas las combinaciones posibles de resultados de cada condición se invoquen al menos una vez.
- **Cobertura de caminos.** Se escriben casos de prueba suficientes para que se ejecuten todos los caminos de un programa. Entendiendo camino como una secuencia de sentencias encadenadas desde la entrada del programa hasta su salida. (Gómez, et.al, 2014)

1.5. Experimento de pruebas de software

El aplicativo propuesto en este trabajo de titulación se basa en un experimento que examina tres técnicas de software como son: lectura de código por abstracciones sucesivas, caja blanca y caja negra, con el uso del aplicativo el experimentador puede detectar la mayoría de los defectos es decir, máxima efectividad y condiciones en las que la técnica ayuda al desarrollador a detectar los defectos más rápidamente es decir, la eficiencia. Tales condiciones pueden incluirse cuando los desarrolladores tienen poca experiencia con el estándar de lenguaje de programación o cuando la mayoría de las fallas son causadas por mal manejo de casos de prueba.

Un fallo es el comportamiento de un programa que se desvía visiblemente de la especificación. Una falla es cualquier problema en el programa. Código fuente que puede manifestarse en un error durante la ejecución. Se usa el defecto como un término general que abarca las fallas de las fallas. (IEEE, 1983).

Según (Humphrey, 1995) un experimento se convierta en un ejercicio estándar que los desarrolladores utilizan para evaluar y afinar sus habilidades de detección de defectos.

En **la técnica de lectura de código por abstracciones sucesivas** los participantes pueden visualizar el código fuente y no pueden ver la especificación. Ellos registran las abstracciones en el sistema. En el segundo paso los participantes pueden descargarse la especificación y registran las inconsistencias encontradas a partir de las abstracciones realizadas por ellos.

En la **técnica caja negra,** los participantes se pueden descargar la especificación del sistema, pero no ven el código fuente. Identifican clases de equivalencia de acuerdo a los datos de entrada y construyen los casos de prueba utilizando las clases de equivalencia, prestando especial atención a los valores límite y si son válidos o no.

Los participantes ejecutan sus casos de prueba en el equipo. Se les instruye para no generar pruebas adicionales durante el paso 2, pero no se puede prevenir ni medir esto hasta que se haya concluido es decir cuando los participantes imprimen sus resultados y cierran la sesión del computador. "Para la detección de fallas, los sujetos usan la especificación real para observar los fracasos que se revelaron en su ejecución" (Weyuker, 1982).

Después de registrar las fallas, los participantes reciben el código fuente para intercambiar y comenzar a aislar los defectos. Los participantes usan el código fuente para aislar las fallas que causaron los defectos observados. No se especifica ninguna técnica especial para la actividad de aislamiento de fallas. Finalmente, los participantes registran las fallas observadas y fallas aisladas

Caja blanca, al iniciar la ejecución de la prueba, los participantes usan una versión establecida del programa para ejecutar sus casos de prueba y ver los informes de valores de cobertura alcanzados.

Los participantes desarrollan pruebas adicionales de casos hasta que alcanzan el 100% de cobertura, o creen que no pueden lograr una mejor cobertura. Después de ejecutar los casos de prueba, los participantes registran en el sistema la salida, se descargan la especificación y comienza la detección de fallas. Los sujetos utilizan la especificación para observar fallas en su salida (Juristo & Vegas, 2003).

Los participantes aíslan las fallas que causaron las fallas observadas, no se especifica ninguna técnica para el aislamiento de fallas. Por último, los participantes registran las fallas observadas y fallas aisladas.

1.6. Familia de Experimentos en pruebas de software

Referente al experimento usado para desarrollar el aplicativo propuesto, este se enmarca en una familia de experimentos ya realizados, en esta sección se aborda un resumen de la familia de experimentos.

El primer experimeno de esta familia se reportó en Basili & Selby (1987) los autores realizaron un experimento cuyo objetivo era examinar la efectividad, eficiencia y el costo de tres técnicas de prueba de software: caja negra por partición de clase de equivalencia y análisis de valores límite, caja blanca por cobertura de código y lectura de código por abstracciones sucesivas.

Después de unos años los autores Kamsties y Lott (1995) realizaron replicaciones de este experimento. Unos años más tarde, los autores Roper, Wood, y Miller (1997) realizaron otra replicación. Los autores Juristo y Vegas (2003) también llevaron a cabo varias réplicas. La eficiencia de las técnicas de prueba de software fue replicada de manera controlada por Farooq y Quadri (2013) en un experimento con meta-análisis. Recientemente una réplica fue realizada por Gómez, et. al. (2017).

Estas réplicas no toman como referencia el mismo experimento por ejemplo, la segunda y tercera replicación se dio a conocer en (Basili & Selby, 1987) y toma como referencia su primer experimento. Los experimentos reportados en (Kamsties & Lott, 1995) es la replicación de (Basili & Selby, 1987).

En el caso del experimento reportado en (Roper, Wood, & Miller, 1997), los autores utilizaron la replicación del paquete del experimento de (Kamsties & Lott, 1995). Otro experimento de los autores se basa en los paquetes con respecto al experimento (Farooq & Quadri, 2013) y está relacionado con el paquete de replicación de (Kamsties & Lott, 1995). En el caso de experimento reportado en (Gómez, Aguilar, & Ucán, 2014), los autores adaptaron el paquete de replicación de (Juristo & Vegas, 2003) .

CAPÍTULO II

2. MARCO METODOLÓGICO

2.1. Tipo de investigación

La investigación realizada es del tipo aplicativa, porque se busca la generación de conocimiento a partir de la aplicación directa al problema, al no existir una herramienta web que apoye la ejecución de experimentos en pruebas de software.

Se tomó fundamentalmente hallazgos tecnológicos previos, es por ello que se realizó el enlace entre la teoría y el sistema web. El presente proyecto se convierte en una fuente de información secundaria porque no se ha generado un conocimiento nuevo, sin embargo se ha investigado la información existente y se ha aplicado en el desarrollo del mismo.

Además es una investigación descriptiva por que la validación de la usabilidad fue realizada mediante la aplicación de dos cuestionarios, sin influir en las decisiones de los participantes. Para la ejecución del experimento se diseñaron dos escenarios uno manual y otro automatizado. Los participantes debían aplicar la técnica de lectura de código por abstracciones sucesivas al finalizar contestaron la encuesta correspondiente al escenario.

2.2. Métodos y técnicas para el desarrollo del sistema

El sistema web no se aplicó en una empresa, sino en un contexto académico porque será una herramienta que apoye a la ejecución de experimentos en IS. Para la realización del proyecto se tuvo la necesidad de una gestión regular de las expectativas del cliente y la presentación de resultados tangibles es por ello se utilizó la metodología ágil SCRUM.

2.3. Metodología SCRUM

En el desarrollo del proyecto se ha utilizado SCRUM porque es una metodología ágil y flexible, minimiza los riesgos en la ejecución de proyectos. Además existe un monitoreo continuo en cada etapa del proyecto con el conocimiento del cliente. Las tareas asignadas bajo esta metodología son de desarrollo iterativo incremental es decir el desarrollo del proyecto se ha segmentado en períodos de tiempo concretos.

SCRUM consta de tres fases principales: Fase de planificación, fase de desarrollo y fase de finalización o cierre. Antes de aplicar la metodología se ha realizado un análisis preliminar en el que se desarrollaron las siguientes actividades: Análisis de procesos, el estudio de factibilidad, el análisis y gestión de riesgos y cálculo de estimaciones. Los requerimientos del sistema no se documentaron mediante las historias de usuario sino que se utilizó los casos de uso y se desarrollaron los escenarios por cada uno de ellos.

2.4. Estructura de la solución

Con la finalidad de resolver el problema encontrado se ha propuesto la siguiente estructura de solución, la cual ha sido desglosada en cuatro componentes, los puntos principales se describen en el presente documento, con el fin de detallar las técnicas y los métodos utilizados para la elaboración y validación del mismo, en la siguiente **Figura 3-2** se puede apreciar la estructura utilizada de manera general.

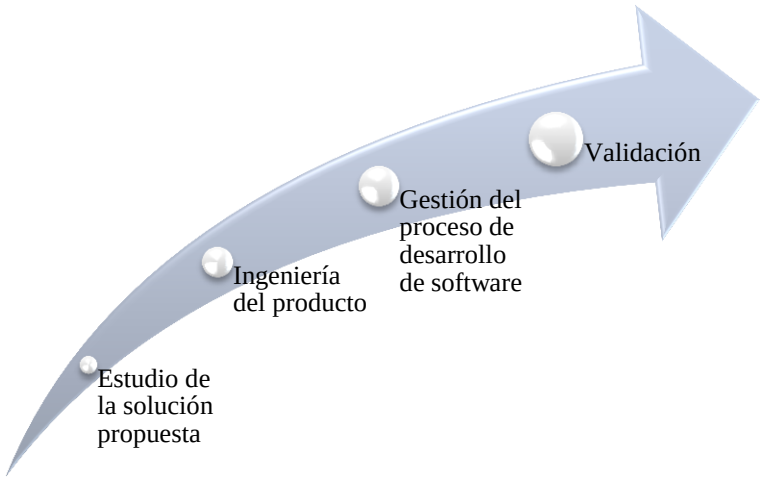

Figura 3-2: Estructura de la solución

Realizado por: G. Hernández 2018

En este apartado se describen los métodos y técnicas empleadas, así como su orden de aplicación para llegar a la solución final, en este caso una aplicación web. Es así que para **el estudio de la solución propuesta** se realizó el análisis de procesos y el estudio de factibilidad.

Una vez demostrado que el sistema es factible y realizada la investigación de los procesos se pasa al siguiente componente que es la **Ingeniería del producto** la misma que consta de:

- Ingeniería de requerimientos
- Arquitectura de software
- Construcción
- Verificación

Al terminar con el segundo componente se prosiguió a la **gestión del proceso de desarrollo software**, donde se especifica la gestión del proyecto y la gestión de riesgos. Finalmente se procedió a la **validación** de la propuesta. Para ello los estudiantes de sexto y noveno semestre de la carrera de ingeniería en sistemas (IS) de la facultad de informática y electrónica (FIE) de la Escuela Superior Politécnica de Chimborazo (ESPOCH).

A continuación se desarrolla los principales componentes de la solución propuesta.

2.5. Estudio de la solución propuesta

El objetivo principal del **estudio de la solución propuesta** es obtener un conocimiento del dominio del problema, para poder comunicarse de manera eficaz con los clientes y los usuarios, es donde el equipo trata de comprender las reglas del negocio, entender las necesidades y con ello proponer una solución adecuada, todo lo referente a la teoría del campo de las pruebas de software en IS se encuentra en el capítulo 1 del presente documento. A continuación se describe el estudio de factibilidad y el análisis de procesos realizado.

2.5.1. Estudio de factibilidad

El estudio de factibilidad es un instrumento que sirve para orientar la toma de decisiones en la evaluación de un proyecto. Se formula en base a la información existente con el fin de medir las posibilidades de éxito o fracaso del proyecto.

En el estudio de factibilidad se determinó los recursos económicos, técnicos y operativos para definir si el proyecto es realizable y de esta manera plantear al usuario un costo aproximado del sistema web "Systexp". Para ello se realizó el estudio de la factibilidad técnica, factibilidad operativa y la factibilidad económica.

2.5.1.1 Factibilidad técnica

La factibilidad técnica estudia la posibilidad tecnológica, es decir la existencia de los equipos para llevar a cabo los procesos, la infraestructura que es la existencia de instalaciones para los equipos, la parte legal que comprende la existencia de regulaciones.

Para el desarrollo del sistema se utilizó el paquete XAMPP porque es independiente de la plataforma, de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl; en el ordenador de desarrollo se instaló el paquete XAMPP. Al terminar el desarrollo del sistema, se migró la aplicación y la base de datos al dominio: http://testingtool.sdjournal.net

2.5.1.2. Factibilidad Operativa

La factibilidad operativa consiste en la creación de métodos y procedimientos que permitan al personal involucrado en el sistema, identificar las funciones y se comprometan con las mismas.

El manejo del sistema estará a cargo del administrador y los experimentadores. El administrador otorgará los permisos pertinentes para las diferentes opciones del sistema

y visualizará los reportes acerca de los experimentos realizados. Un experimentador puede ejecutar experimentos, administrar participantes y ver reportes pero solamente los que él haya creado.

Los participantes son usuarios que aplicarán las técnicas de acuerdo a los tratamientos generados por el sistema de apoyo a la ejecución de experimentos en IS. En el desarrollo del sistema se contó con el siguiente personal técnico.

El personal técnico comprende aquellas personas que dominan un área en particular, es por ello que para el desarrollo del sistema se contó con el personal técnico descrito en la siguiente **Tabla 1-2** se puede apreciar las funciones que cumplieron cada uno de los miembros de acuerdo con la metodología SCRUM.

Tabla 1-2: Personal técnico

Nombre	**Función**
Gloria Hernández	Desarrollador
Ing. Raúl Rosero	SCRUM Manager
Dr. Omar Gómez	Tester Owner

Realizado por: G. Hernández, 2018.

2.5.1.3. Factibilidad Económica

En la factibilidad económica se realizó una evaluación de los recursos económicos para el desarrollo, instalación y puesta en marcha del sistema. Para la instalación del sistema no se requirió de licencias por que XAMPP es un paquete libre. Todos los gastos económicos fueron autofinanciados.

Finalmente en el estudio se determinó que el proyecto es factible, ya que existen los recursos tanto humanos, materiales y económicos que facilitan la implementación del sistema. Actualmente el sistema se encuentra funcionando en el siguiente dominio: http://testingtool.sdjournal.net.

2.6. Análisis de procesos

Para comprender los procesos que intervienen en el proyecto se ha realizado un análisis y establecimiento de los pasos que se realizan al ejecutar un experimento, se ha tenido reuniones con el Dr. Omar Gómez quien ha sido un eje fundamental para el desarrollo del proyecto, debido a sus conocimientos de experimentación en Ingeniería de Software.

2.7. Ingeniería del producto

La ingeniería del producto trata del diseño y desarrollo del producto software en este caso una aplicación web, a continuación se describe lo realizado para desarrollar la aplicación

2.7.1. Ingeniería de requerimientos

La ingeniería de requerimientos permite desarrollar una especificación completa, consistente y no confusa, la misma que servirá como base para los acuerdos comunes entre las partes involucradas y en dónde se describen las funciones que realizará el sistema.

Una parte fundamental de la especificación de requerimientos es documentarlos, porque permite continuar con la definición del sistema software a desarrollar, tomando como punto de partida los requisitos generales, los objetivos de la herramienta web a implantar, es por eso que para documentar los requerimientos del sistema se utilizó los diagramas de casos de uso.

En la metodología SCRUM para documentar los requerimientos del sistema se utilizan las historias de usuario sin embargo se realizó una adaptación y se utilizó los diagramas de casos de uso.

Cada uno de los requerimientos del sistema se ha documentado a través de los diagramas de casos de uso, porque son una descripción de los pasos o las actividades que debe realizar el aplicativo. Para especificar cada caso de uso de la herramienta de apoyo a la ejecución de experimentos en IS, se hizo una plantilla que consta de 4 partes

importantes como son: la descripción del caso de uso, los escenarios, la descripción detallada y finalmente la información táctica.

En la descripción del caso de uso resaltan los siguientes puntos: el nombre que es una descripción corta del caso de uso, el propósito u objetivo del caso de uso desde la perspectiva del usuario, el líder o las personas del equipo de desarrollo responsables, la pre-condición o estado del sistema antes de que comience el caso de uso, la post-condición que se refiere al estado del sistema después de que se ha completado el escenario del caso de uso y finalmente los actores primarios o el rol principal que estimula para que se ejecute el caso de uso.

Los escenarios se refieren a los diferentes tipos de caminos que un caso de uso puede tener, es así que en el escenario principal se contiene las secuencias de pasos o actividades que por lo general realizará el usuario del caso de uso. En el escenario secundario o alterno, la acción a realizar puede contener varias opciones para obtener algún resultado. Y el escenario con excepciones, es donde se espera que aquellas excepciones puedan ocurrir tras realizar las actividades del caso de uso.

En la definición de los requerimientos se tuvo varias reuniones con el Dr. Omar Gómez para establecer las necesidades del sistema. Con el fin de modelar los requisitos del sistema se ha empleado los diagramas de casos de uso.

A continuación se detallan los casos de uso especificados para el sistema desarrollado:

2.7.1.1. Diagrama de casos de uso de un administrador

En el siguiente **Gráfico 1-2** se puede apreciar los casos de uso de un administrador, debe autentificarse, el sistema maneja internamente el tipo de usuario, es por ello que se ha realizado una sola entrada al sistema. Es indispensable recalcar que un administrador es una persona experta porque tiene el control de todo el sistema, puede gestionar a más administradores, gestionar experimentadores e incluso a los participantes. Es el único rol donde se puede gestionar las técnicas y los objetos experimentales para que sean usados al preparar la práctica. Un administrador puede crear un experimento y asignarlo

a un experimentador. Puede ver todos los reportes generados de los experimentos realizados en el sistema.

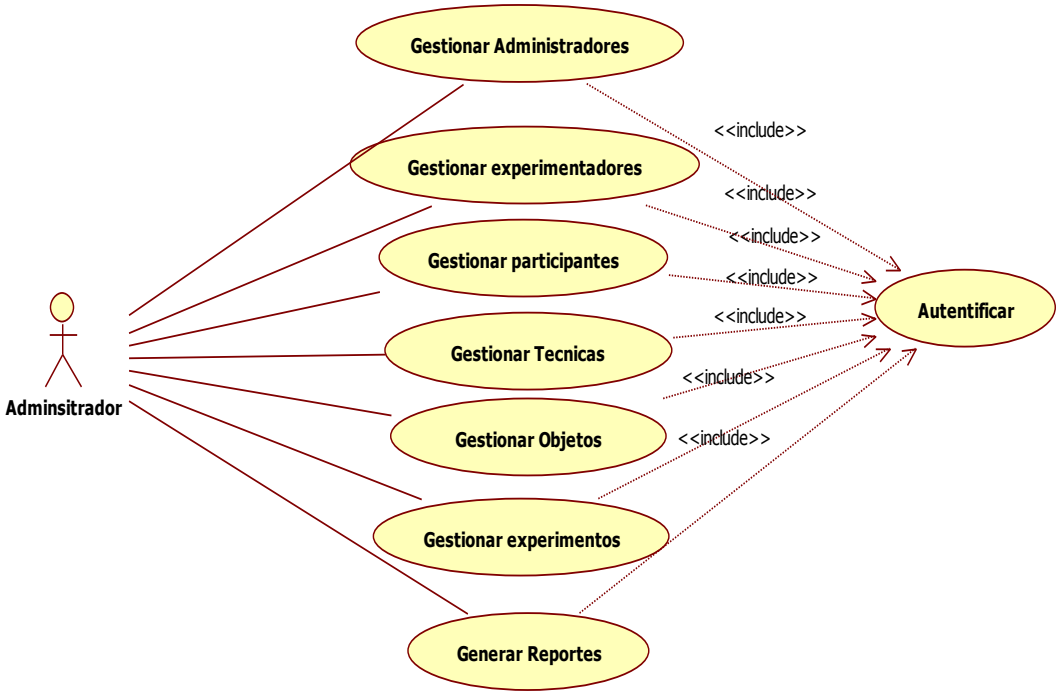

Gráfico 1-2: Diagrama de Casos de uso de un administrador
Realizado por: G. Hernández, 2018

El caso de uso para gestionar administradores fue documentado en la **Tabla 2-2** donde se puede apreciar el nombre del caso de uso, el propósito, la precondición y la secuencia de los escenarios. En la secuencia principal se observa como un administrador registra a un nuevo administrador, debe ingresar los datos correspondientes del nuevo administrador y presionar en el botón guardar. En el escenario secundario se puede apreciar los pasos que debe seguir un administrador para editar los datos de un administrador. Finalmente en el escenario con excepciones se presenta los pasos que debe seguir el administrador para eliminar a un usuario con el rol administrador.

Tabla 2-2: Caso de uso 01 para gestionar administradores

<table>
<tr><td colspan="3">CU-01</td></tr>
<tr><td colspan="3">Nombre: Gestionar administradores</td></tr>
<tr><td colspan="3">Propósito: Agregar, editar o eliminar un administrador que es una persona experta que tiene el control de todo el sistema.</td></tr>
<tr><td colspan="3">Pre-condición: El Administrador debe estar registrado y autenticado previamente en el sistema para usar el mismo.</td></tr>
<tr><td colspan="3">Post-condición: El administrador es capaz de gestionar a los administradores del sistema. La gestión implica agregar, editar, listar y eliminar administradores</td></tr>
<tr><td colspan="3">Actores primarios: Administrador</td></tr>
<tr><td colspan="3">Escenarios</td></tr>
<tr><td colspan="3">Descripción detallada</td></tr>
<tr><td colspan="3">Secuencia del escenario principal</td></tr>
<tr><td rowspan="4">El administrador registra un administrador</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador registra el nombre, apellido, correo y contraseña de un administrador</td></tr>
<tr><td>2</td><td>El Sistema registra el tipo de usuario, nombre de usuario y el estado activo.</td></tr>
<tr><td>3</td><td>Guardar los datos.</td></tr>
<tr><td colspan="3">Secuencia del escenario secundario</td></tr>
<tr><td rowspan="5">El administrador edita un administrador</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador selecciona un administrador de la lista y hace click en el icono en forma de lápiz.</td></tr>
<tr><td>2</td><td>El sistema muestra los datos del administrador.</td></tr>
<tr><td>3</td><td>El administrador edita los datos</td></tr>
<tr><td>4</td><td>Guardar los datos</td></tr>
<tr><td colspan="3">Secuencia del escenario con excepciones</td></tr>
<tr><td rowspan="5">El administrador elimina un administrador</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador selecciona un administrador de la lista y hace click en el icono en forma de una “X”.</td></tr>
<tr><td>2</td><td>El sistema le muestra los datos del administrador a eliminar.</td></tr>
<tr><td>3</td><td>El administrador selecciona eliminar.</td></tr>
<tr><td>4</td><td>El sistema elimina los datos</td></tr>
<tr><td colspan="3">Información Táctica</td></tr>
<tr><td colspan="2">Prioridad: Prioridad Alta (1)</td><td>Interfaz de usuario: Se requiere una interfaz sencilla.</td></tr>
</table>

Realizado por: G. Hernández, 2018.

El siguiente caso de uso fue desarrollado para gestionar experimentadores en la **Tabla 3-2** se puede observar el nombre del caso de uso, el propósito, la precondición y la secuencia de los escenarios. En la secuencia principal se observan los pasos que debe realizar un administrador para registrar a un nuevo experimentador, debe ingresar los datos correspondientes del nuevo experimentador y presionar en el botón guardar. En el escenario secundario se puede apreciar como un administrador puede editar los datos de

un experimentador. En el último escenario con excepciones se observan los pasos que debe seguir el administrador para eliminar a un experimentador.

Tabla 3-2: Caso de uso 02 para gestionar experimentadores

<table>
<tr><td colspan="3">CU-02</td></tr>
<tr><td colspan="3">Nombre: Gestionar experimentadores</td></tr>
<tr><td colspan="3">Propósito: Agregar, editar o eliminar un experimentador.</td></tr>
<tr><td colspan="3">Pre-condición: El Administrador debe estar registrado y autenticado previamente en el sistema para usar el mismo.</td></tr>
<tr><td colspan="3">Post-condición: El administrador es capaz de gestionar a un experimentador del sistema. La gestión implica agregar, editar, listar y eliminar experimentadores</td></tr>
<tr><td colspan="3">Actores primarios: Administrador</td></tr>
<tr><td colspan="3">Escenarios</td></tr>
<tr><td colspan="3">Descripción detallada</td></tr>
<tr><td colspan="3">Secuencia del escenario principal</td></tr>
<tr><td rowspan="4">El administrador registra un experimentador</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador registra el nombre, apellido, correo y contraseña de un experimentador</td></tr>
<tr><td>2</td><td>El Sistema registra el tipo de usuario, nombre de usuario y el estado activo.</td></tr>
<tr><td>3</td><td>Guardar los datos.</td></tr>
<tr><td colspan="3">Secuencia del escenario secundario</td></tr>
<tr><td rowspan="5">El administrador edita un experimentador</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador selecciona un experimentador de la lista y hace click en el icono en forma de lápiz.</td></tr>
<tr><td>2</td><td>El sistema muestra los datos del experimentador.</td></tr>
<tr><td>3</td><td>El administrador edita los datos del experimentador</td></tr>
<tr><td>4</td><td>Guardar los datos</td></tr>
<tr><td colspan="3">Secuencia del escenario con excepciones</td></tr>
<tr><td rowspan="5">El administrador elimina un experimentador</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador selecciona un experimentador de la lista y hace click en el icono en forma de una “X”.</td></tr>
<tr><td>2</td><td>El sistema muestra los datos a eliminar del experimentador</td></tr>
<tr><td>3</td><td>El administrador selecciona eliminar.</td></tr>
<tr><td>4</td><td>El sistema elimina los datos</td></tr>
<tr><td colspan="3">Información Táctica</td></tr>
<tr><td colspan="2">Prioridad: Prioridad Alta (1)</td><td>Interfaz de usuario: Se requiere una interfaz sencilla.</td></tr>
</table>

Realizado por: G. Hernández, 2018.

Para la gestión de participantes se realizó el siguiente caso de uso, en la **Tabla 4-2** se puede evidenciar el nombre, el propósito y la precondición del caso de uso. Este caso de uso tiene tres secuencias. En la secuencia principal se observan los pasos que debe realizar un administrador para registrar a un nuevo participante, debe ingresar los datos correspondientes y presionar en el botón guardar. En el escenario secundario se puede apreciar como un administrador puede editar los datos de un participante y al presionar en el botón guardar los datos son almacenados. En el último escenario se observan los pasos que debe seguir el administrador para eliminar a un participante

Tabla 4-2: Caso de uso 03 para gestionar participantes

<table>
<tr><td colspan="3">CU-03</td></tr>
<tr><td colspan="3">Nombre: Gestionar participantes</td></tr>
<tr><td colspan="3">Propósito: Agregar, editar o eliminar un participante.</td></tr>
<tr><td colspan="3">Pre-condición: El Administrador debe estar registrado y autenticado previamente en el sistema para usar el mismo.</td></tr>
<tr><td colspan="3">Post-condición: El administrador es capaz de gestionar a un participante del sistema. La gestión implica agregar, editar, listar y eliminar participantes.</td></tr>
<tr><td colspan="3">Actores primarios: Administrador</td></tr>
<tr><td colspan="3">Escenarios</td></tr>
<tr><td colspan="3">Descripción detallada</td></tr>
<tr><td colspan="3">Secuencia del escenario principal</td></tr>
<tr><td rowspan="4">El administrador registra un participante.</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador registra el nombre, apellido, correo, contraseña y nombre del experimento de un participante</td></tr>
<tr><td>2</td><td>El Sistema registra el tipo de usuario, nombre de usuario y el estado activo.</td></tr>
<tr><td>3</td><td>Guardar los datos.</td></tr>
<tr><td colspan="3">Secuencia del escenario secundario</td></tr>
<tr><td rowspan="5">El administrador edita un participante</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador selecciona un participante de la lista y hace click en el icono en forma de lápiz.</td></tr>
<tr><td>2</td><td>El sistema le muestra los datos del participante.</td></tr>
<tr><td>3</td><td>El administrador edita los datos del participante</td></tr>
<tr><td>4</td><td>Guardar los datos</td></tr>
<tr><td colspan="3">Secuencia del escenario con excepciones</td></tr>
<tr><td rowspan="5">El administrador elimina un participante</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador selecciona un particpante de la lista y hace click en el icono en forma de una “X”.</td></tr>
<tr><td>2</td><td>El sistema muestra los datos a eliminar del participante</td></tr>
<tr><td>3</td><td>El administrador selecciona eliminar.</td></tr>
<tr><td>4</td><td>El sistema elimina los datos</td></tr>
<tr><td colspan="3">Información Táctica</td></tr>
<tr><td colspan="2">Prioridad: Prioridad Alta (1)</td><td>Interfaz de usuario: Se requiere una interfaz sencilla.</td></tr>
</table>

Realizado por: G. Hernández, 2018.

Con el fin de gestionar las técnicas o tratamientos, se documentó el caso de uso 04 que se encuentra plasmado en la **Tabla 5-2.** Un administrador debe ingresar la técnica con sus respectivas tareas, por cada tarea existe una serie de instrucciones que son almacenadas en el sistema. En el caso de uso se puede apreciar tres secuencias, siendo la primera el ingreso de una técnica, en la segunda secuencia el administrador puede ingresar las tareas y debe seleccionar los objetos experimentales que desea mostrar por cada tarea.

Por ejemplo, en la primera tarea de la técnica de abstracciones sucesivas se muestra el código fuente, no se debe mostrar al participante el código ejecutable o la especificación. En el segundo paso se muestra al participante la especificación para que él pueda ingresar las inconsistencias. Cabe recalcar que las tres técnicas definidas en el sistema no pueden ser borradas solamente pueden ser editadas. La secuencia principal para registrar una técnica se lo realizó con el propósito de que a futuro nuevas técnicas puedan ser implementadas en el sistema.

Tabla 5-2: Caso de uso 04 para gestionar técnicas

<table>
<tr><td colspan="3">CU-04</td></tr>
<tr><td colspan="3">Nombre: Gestionar Técnicas (Treatments)</td></tr>
<tr><td colspan="3">Propósito: Agregar o editar técnicas de pruebas de software así como sus respectivas tareas.</td></tr>
<tr><td colspan="3">Pre-condición: El Administrador debe estar registrado y autenticado previamente en el sistema para usar el mismo.</td></tr>
<tr><td colspan="3">Post-condición: El administrador es capaz de gestionar las técnicas de pruebas a aplicar. La gestión implica agregar, editar y listar las técnicas.</td></tr>
<tr><td colspan="3">Actores primarios: Administrador</td></tr>
<tr><td colspan="3">Escenarios</td></tr>
<tr><td colspan="3">Descripción detallada</td></tr>
<tr><td colspan="3">Secuencia del escenario principal</td></tr>
<tr><td rowspan="4">El administrador registra una técnica.</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador registra el nombre de la técnica</td></tr>
<tr><td>2</td><td>El administrador registra la descripción de la técnica</td></tr>
<tr><td>3</td><td>Guardar los datos.</td></tr>
</table>

(Continua)

Secuencia del escenario secundario		
El administrador agrega una tarea a una técnica	**Paso**	**Descripción**
	1	El administrador selecciona una técnica de la lista, haciendo click en el icono en forma de lupa.
	2	El sistema muestra los datos de aquella técnica.
	3	El administrador hace click en el botón (Add) para añadir una nueva tarea.
	4	El administrador selecciona los objetos experimentales que se mostrará en cada tarea. Dichos objetos son la especificación del objeto, el programa ejecutable y el código fuente.
	5	Guardar los datos
Secuencia del escenario con excepciones		
	Paso	**Descripción**
El administrador edita una técnica	**1**	El administrador selecciona una técnica de la lista.
	2	El administrador edita la descripción de la técnica
	3	El administrador edita las tareas de la técnica presionando en el icono que tiene un lápiz.
	4	El administrador edita las instrucciones de la tarea. Cada instrucción se puede separar mediante un interlineado.
	5	Guarda los datos
Información Táctica		
Prioridad: Prioridad Alta (1)		**Interfaz de usuario**: Se requiere una interfaz sencilla.

Realizado por: G. Hernández, 2018.

El siguiente caso de uso se documentó con el propósito de agregar, eliminar un objeto experimental o programa. Al conversar con el cliente se pudo apreciar que un objeto experimental consta de una especificación, del código ejecutable y de uno o varios archivos con el código fuente. La especificación del sistema es una descripción del comportamiento del sistema y está escrito en lenguaje natural. En la **Tabla 6-2** se puede evidenciar las tres secuencias para la gestión de objetos experimentales. La secuencia principal permite el registro de un objeto. La segunda secuencia permite al administrador agregar a un objeto experimental más archivos con el código fuente. En la secuencia con excepciones se evidencia como un objeto puede ser eliminado, está acción no se ejecutará si el objeto es parte de un experimento.

Tabla 6-2: Caso de uso 05 para gestionar objetos experimentales

<table>
<tr><td colspan="3">CU-05</td></tr>
<tr><td colspan="3">Nombre: Gestionar objetos (Programs)</td></tr>
<tr><td colspan="3">Propósito: Agregar, eliminar un objeto experimental.</td></tr>
<tr><td colspan="3">Pre-condición: El Administrador debe estar registrado y autenticado previamente en el sistema para usar el mismo.</td></tr>
<tr><td colspan="3">Post-condición: El administrador es capaz de gestionar a un objeto del sistema. La gestión implica agregar, consultar, listar y eliminar objetos.</td></tr>
<tr><td colspan="3">Actores primarios: Administrador</td></tr>
<tr><td colspan="3">Escenarios</td></tr>
<tr><td colspan="3">Descripción detallada</td></tr>
<tr><td colspan="3">Secuencia del escenario principal</td></tr>
<tr><td rowspan="3">El administrador registra un programa</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador registra la descripción, y los archivos con la especificación, programa ejecutable y el código fuente.</td></tr>
<tr><td>2</td><td>Guardar los datos.</td></tr>
<tr><td colspan="3">Secuencia del escenario secundario</td></tr>
<tr><td rowspan="5">El administrador agrega a un objeto más archivos con el código fuente.</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador selecciona un objeto de la lista y hace click en el icono en forma de “#”</td></tr>
<tr><td>2</td><td>El sistema le muestra el objeto</td></tr>
<tr><td>3</td><td>El administrador sube el archivo con el código fuente.</td></tr>
<tr><td>4</td><td>Guardar los datos</td></tr>
<tr><td colspan="3">Secuencia del escenario con excepciones</td></tr>
<tr><td rowspan="5">El administrador elimina un objeto</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El administrador selecciona un objeto de la lista y hace click en el icono en forma de una “X”.</td></tr>
<tr><td>2</td><td>El sistema muestra los datos a eliminar del objeto.</td></tr>
<tr><td>3</td><td>El administrador selecciona eliminar.</td></tr>
<tr><td>4</td><td>El sistema elimina los datos del objeto que no esté implicado en un experimento.</td></tr>
<tr><td colspan="3">Información Táctica</td></tr>
<tr><td colspan="2">Prioridad: Prioridad Alta (1)</td><td>Interfaz de usuario: Se requiere una interfaz sencilla.</td></tr>
</table>

Realizado por: G. Hernández, 2018.

En la siguiente **Tabla 7-2** se observa el caso de uso para gestionar experimentos, es necesario que las técnicas y los objetos experimentales se encuentren previamente registrados en el sistema. La secuencia principal de un experimento es ingresar la descripción, el tipo de diseño, las técnicas, los objetos experimentales y la fecha de ejecución.

La descripción debe ser clara y evitar caracteres especiales porque dicha descripción permitirá al participante registrarse en el sistema. El diseño experimental debe ser elegido cuidadosamente porque en el diseño factorial se permite una sola sesión, con una sola técnica y un objeto experimental. Mientras que el diseño experimental de medidas repetidas permite realizar varias sesiones con una sola técnica e implica a varios objetos que serán mostrados por cada sesión.

En la secuencia secundaria se observa como un administrador puede asignar un experimento a otro experimentador. Los experimentos se muestran solamente a la persona quien lo creó. Esta secuencia permite que el administrador del sistema cree un experimento y él pueda asignarlo a un experimentador entonces se convierte él en el nuevo propietario. También es posible editar la fecha del experimento. No se permite más cambios en el experimento porque entonces se convierte en otro experimento.

La secuencia con excepciones permite eliminar un experimento, siempre y cuando este no haya sido ejecutado por algún participante.

La secuencia para correr el experimento se da cuando los participantes se han inscrito en el sistema, en ese momento ellos se encuentran desactivados y no podrán ingresar en el sistema. Es necesario que el experimentador se cerciore que todos los participantes estén inscritos. Entonces el experimentador debe seleccionar el experimento y presionar en el botón (Randomization), en ese instante la cuenta de los participantes es activada y reciben la asignación de la técnica y del objeto.

Hay que tener mucho cuidado cuando falta alguno de los participantes, el experimentador debe inscribirlo en el sistema y puede volver a correr el experimento. No se podrán realizar nuevas asignaciones si alguno de los participantes ingresa y comienza la práctica. Si esto ocurriese se recomienda que genere un nuevo experimento con los participantes faltantes.

Tabla 7-2: Caso de uso 06 para gestionar experimentos

CU-06		
Nombre: Gestionar experimentos		
Propósito: Agregar, editar o eliminar un experimento		
Pre-condición: El Administrador debe estar registrado y autenticado previamente en el sistema para usar el mismo.		
Post-condición: El administrador es capaz de gestionar a un experimento del sistema. La gestión implica agregar, editar, listar y eliminar experimentos.		
Actores primarios: Administrador		
Escenarios		
Descripción detallada		
Secuencia del escenario principal		
El administrador registra un experimento	**Paso**	**Descripción**
	1	El administrador registra la descripción, el tipo, selecciona dos o tres técnicas, dos o tres objetos y la fecha de la sesión.
	2	El sistema registra el dueño del experimento.
	2	Guardar los datos.
Secuencia del escenario secundario		
El administrador asigna un experimento a un experimentador	**Paso**	**Descripción**
	1	El administrador selecciona un experimento de la lista y selecciona en el icono en forma de lápiz
	2	El sistema le muestra el experimento y la lista de los administradores y experimentadores que existen.
	3	El administrador selecciona el nuevo dueño.
	4	Guardar los datos
Secuencia del escenario con excepciones		
El administrador elimina un experimento	**Paso**	**Descripción**
	1	El administrador selecciona un experimento de la lista y hace click en el icono en forma de una "X".
	2	El sistema muestra los datos a eliminar del experimento
	3	El administrador selecciona eliminar.
	4	El sistema elimina los datos del experimento que no tenga una asignación realizada.
Secuencia del escenario correr el experimento		
El administrador ejecuta el experimento.	**1**	El administrador selecciona el experimento y presiona en el botón (Run randomization).
	2	El sistema verifica que existen participantes registrados en el experimento
	3	El sistema mezcla a los participantes
	4	El sistema hace la combinación de la técnica con los programas.
	5	El sistema registra los tratamientos.
Información Táctica		
Prioridad: Prioridad Alta (1)	**Interfaz de usuario**: Se requiere una interfaz sencilla.	

Realizado por: G. Hernández, 2018.

Como parte de los requerimientos, el cliente solicitó que se generen tres reportes que son: reporte del tiempo, un reporte general y el reporte de asignaciones. En el primer reporte se visualiza el tiempo en minutos capturado por el sistema, es decir el sistema calcula cuanto tiempo se demoró un participante en realizar cada tarea de acuerdo con la asignación. En el segundo reporte se visualiza los datos ingresados por los participantes, y en el reporte de participantes se puede observar la lista de ellos con sus asignaciones, también este reporte puede ser usado para conocer que participantes están registrados en el experimento sin que este haya sido corrido aun. En la **Tabla 8-2** se muestra el caso de uso de un administrador para generar reportes.

Tabla 8-2: Caso de uso 07 para generar reportes

CU-07		
Nombre: Generar reportes		
Propósito: Generar reportes de los experimentos		
Pre-condición: El Administrador debe estar registrado y autenticado previamente en el sistema para usar el mismo.		
Post-condición: El administrador es capaz de obtener reportes.		
Actores primarios: Administrador		
Escenarios		
Descripción detallada		
Secuencia del reporte uno		
El administrador hace un reporte del tiempo por cada experimento.	**Paso**	**Descripción**
	1	El administrador hace click en "Report time in exp."
	2	El sistema le muestra la lista de los experimentos.
	3	El administrador presiona en el botón Ok
	4	El sistema le muestra los datos en un archivo pdf
Secuencia del reporte dos		
El administrador hace un reporte general por cada experimento.	**Paso**	**Descripción**
	1	El administrador hace click en "General report."
	2	El sistema le muestra la lista de los experimentos.
	3	El administrador presiona en el botón Ok
	4	El sistema le muestra los datos en un archivo pdf
Secuencia del reporte tres		
El administrador hace un reporte de los participantes por cada experimento.	**Paso**	**Descripción**
	1	El administrador hace click en "List of participants."
	2	El sistema le muestra la lista de los experimentos.
	3	El administrador presiona en el botón Ok.
	4	El sistema le muestra los datos en un archivo pdf
Información Táctica		
Prioridad: Prioridad Alta (1)	**Interfaz de usuario**: Se requiere una interfaz sencilla.	

Realizado por: G. Hernández, 2018.

2.7.1.2. Diagrama de casos de uso de un experimentador

En el siguiente diagrama se puede apreciar los casos de uso de un experimentador, el mismo que debe autentificarse para que el sistema le muestre la pantalla inicial correspondiente a su rol. Un experimentador puede gestionar a los participantes, experimentos y generar reportes. Tiene el control sobre los experimentos que ha creado.

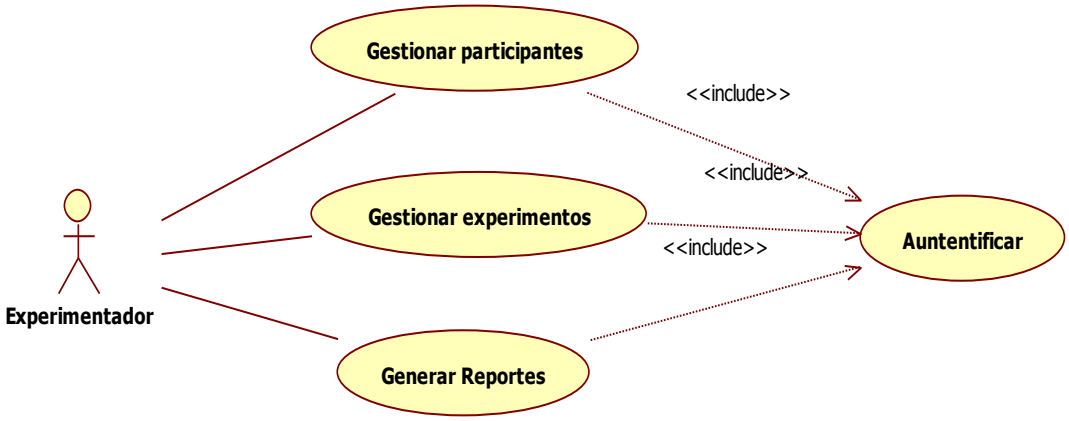

Gráfico 2-2: Diagrama de Casos de uso de un experimentador

Realizado por: G. Hernández, 2018.

Un experimentador puede realizar varias acciones tales como: gestionar participantes, gestionar experimentos, y generar reportes. En la **Tabla 9-2** se puede constatar el caso de uso para gestionar un participante. Un experimentador puede registrar, editar o consultar un participante. En este rol solo se puede visualizar los participantes que son parte de algún experimento ejecutado por el experimentador. El experimentador no podrá borrar los datos de un participante porque este pudo haber desarrollado alguna práctica, entonces la función de eliminar está delegada para el administrador del sistema.

Tabla 9-2: Caso de uso 08 para gestionar participantes con el rol de experimentador

<table>
<tr><td colspan="3">CU-08</td></tr>
<tr><td colspan="3">Nombre: Gestionar participantes</td></tr>
<tr><td colspan="3">Propósito: Agregar, editar un participante.</td></tr>
<tr><td colspan="3">Pre-condición: El experimentador debe estar registrado y autenticado previamente en el sistema para usar el mismo.</td></tr>
<tr><td colspan="3">Post-condición: El experimentador es capaz de gestionar a un participante del sistema. La gestión implica agregar, editar, listar participantes.</td></tr>
<tr><td colspan="3">Actores primarios: Experimentador</td></tr>
<tr><td colspan="3">Escenarios</td></tr>
<tr><td colspan="3">Descripción detallada</td></tr>
<tr><td colspan="3">Secuencia del escenario principal</td></tr>
<tr><td rowspan="4">El experimentador registra un participante.</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El experimentador registra el nombre, apellido, correo, contraseña y nombre del experimento de un participante</td></tr>
<tr><td>2</td><td>El Sistema registra el tipo de usuario, nombre de usuario y el estado inactivo hasta que se ejecute las asignaciones.</td></tr>
<tr><td>3</td><td>Guardar los datos.</td></tr>
<tr><td colspan="3">Secuencia del escenario secundario</td></tr>
<tr><td rowspan="5">El experimentador edita un participante</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El experimentador selecciona un participante de la lista y hace click en el icono en forma de lápiz.</td></tr>
<tr><td>2</td><td>El sistema le muestra los datos del participante.</td></tr>
<tr><td>3</td><td>El experimentador edita los datos del participante</td></tr>
<tr><td>4</td><td>Guardar los datos</td></tr>
<tr><td colspan="3">Información Táctica</td></tr>
<tr><td colspan="2">Prioridad: Prioridad Alta (1)</td><td>Interfaz de usuario: Se requiere una interfaz sencilla.</td></tr>
</table>

Realizado por: G. Hernández, 2018.

En el caso de uso 09 de la **Tabla 10-2** se visualiza tres funciones que puede realizar un experimentador. Cabe recalcar que el experimentador tiene acceso a los experimentos creados por él. Un experimentador no tiene el control sobre las técnicas o sobre los objetos, él simplemente usa los instrumentos para desarrollar el experimento. Debe elegir la descripción, el diseño experimental, las técnicas, los objetos experimentales y la fecha de ejecución. La descripción del experimento debe ser lo más concisa y precisa sin caracteres especiales, el diseño experimental puede ser factorial o de medidas repetidas. El experimentador solo puede ejecutar sus experimentos y eliminar alguno de ellos. No podrá eliminar un experimento que haya sido ejecutado por algún participante.

Tabla 10-2: Caso de uso 09 para gestionar experimentos con el rol de experimentador

<table>
<tr><td colspan="3">CU-09</td></tr>
<tr><td colspan="3">Nombre: Gestionar experimentos</td></tr>
<tr><td colspan="3">Propósito: Agregar, editar o eliminar un experimento</td></tr>
<tr><td colspan="3">Pre-condición: El Experimentador debe estar registrado y autenticado previamente en el sistema para usar el mismo.</td></tr>
<tr><td colspan="3">Post-condición: El experimentador es capaz de gestionar sus experimentos. La gestión implica agregar, editar, listar y eliminar sus experimentos.</td></tr>
<tr><td colspan="3">Actores primarios: Experimentador</td></tr>
<tr><td colspan="3">Escenarios</td></tr>
<tr><td colspan="3">Descripción detallada</td></tr>
<tr><td colspan="3">Secuencia del escenario principal</td></tr>
<tr><td rowspan="3">El experimentador registra un experimento</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El experimentador registra la descripción, el tipo, selecciona dos o tres técnicas, dos o tres objetos y la fecha de la sesión.</td></tr>
<tr><td>2</td><td>El sistema registra el dueño del experimento.</td></tr>
<tr><td>2</td><td>Guardar los datos.</td></tr>
<tr><td colspan="3">Secuencia del escenario con excepciones</td></tr>
<tr><td rowspan="5">El experimentador elimina un experimento</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El experimentador selecciona un experimento de la lista y hace click en el icono en forma de una “X”.</td></tr>
<tr><td>2</td><td>El sistema muestra los datos a eliminar del experimento</td></tr>
<tr><td>3</td><td>El experimentador selecciona eliminar.</td></tr>
<tr><td>4</td><td>El sistema elimina los datos del experimento que no tenga una asignación realizada.</td></tr>
<tr><td colspan="3">Secuencia del escenario correr el experimento</td></tr>
<tr><td rowspan="6">El experimentador ejecuta el experimento.</td><td>1</td><td>El experimentador selecciona el experimento y presiona en el botón (Run randomization).</td></tr>
<tr><td>2</td><td>El sistema verifica que existen participantes registrados en el experimento</td></tr>
<tr><td>3</td><td>El sistema mezcla a los participantes</td></tr>
<tr><td>4</td><td>El sistema hace la combinación de la técnica con los programas.</td></tr>
<tr><td>5</td><td>El sistema registra los tratamientos.</td></tr>
<tr><td colspan="3">Información Táctica</td></tr>
<tr><td colspan="2">Prioridad: Prioridad Alta (1)</td><td>Interfaz de usuario: Se requiere una interfaz sencilla.</td></tr>
</table>

Realizado por: G. Hernández, 2018.

En el siguiente caso de uso se muestran los pasos que debe realizar un experimentador para generar reportes. Un experimentador puede ver tres tipos de reportes: el reporte de tiempo, el reporte general y el reporte de participantes.

Los experimentos que sean propios del experimentador se muestran a través de una lista, a continuación debe seleccionar uno de ellos y presionar en el botón (Generate report), el sistema le mostrará los datos correspondientes, como por ejemplo: En el reporte de tiempo se muestran los minutos que tardó un participante en terminar una asignación.

El siguiente reporte permite observar los datos ingresados por los participantes. Y en el último reporte se visualiza la lista de participantes inscritos en el experimento, cuando el experimentador corre el experimento el sistema automáticamente activa a los participantes y les da una técnica con un objeto experimental, entonces este reporte muestra la lista de participantes con las asignaciones. En la **Tabla 11-2** se puede apreciar el caso de uso para generar reportes.

Tabla 11-2: Caso de uso 10 para generar reportes con el rol de experimentador

CU-10		
Nombre: Generar reportes		
Propósito: Generar reportes de los experimentos		
Pre-condición: El Experimentador debe estar registrado y autenticado previamente en el sistema para usar el mismo.		
Post-condición: El experimentador es capaz de obtener reportes.		
Actores primarios: Experimentador		
Escenarios		
Descripción detallada		
Secuencia del escenario principal		
El experimentador hace un reporte del tiempo por cada experimento.	**Paso**	**Descripción**
	1	El experimentador hace click en “Report time in exp.”
	2	El sistema le muestra la lista de los experimentos.
	3	El experimentador presiona en el botón Ok
	4	El sistema le muestra los datos en un archivo pdf
Secuencia del escenario secundario		
El experimentador hace un reporte general por cada experimento.	**Paso**	**Descripción**
	1	El experimentador hace click en “General report.”
	2	El sistema le muestra la lista de los experimentos.
	3	El experimentador presiona en el botón Ok

(Continua)

<table>
<tr><td></td><td>4</td><td>El sistema le muestra los datos en un archivo pdf</td></tr>
<tr><td colspan="3">Secuencia del escenario con excepciones</td></tr>
<tr><td rowspan="5">El experimentador hace un reporte de los participantes por cada experimento.</td><td>Paso</td><td>Descripción</td></tr>
<tr><td>1</td><td>El experimentador hace click en “List of participants.”</td></tr>
<tr><td>2</td><td>El sistema le muestra la lista de los experimentos.</td></tr>
<tr><td>3</td><td>El experimentador presiona en el botón Ok.</td></tr>
<tr><td>4</td><td>El sistema le muestra los datos en un archivo pdf</td></tr>
<tr><td colspan="3">Información Táctica</td></tr>
<tr><td colspan="2">Prioridad: Prioridad Alta (1)</td><td>Interfaz de usuario: Se requiere una interfaz sencilla.</td></tr>
</table>

Realizado por: G. Hernández, 2018.

2.7.1.3. Diagrama de casos de uso de un participante

En el Gráfico 3-2 se puede observar los casos de uso de un participante, el mismo que puede registrarse en un experimento. Dicho experimento debe estar activo y dentro del rango de tiempo especificado por el experimentador. Una vez que los participantes se han registrado, se ejecuta la asignación aleatoria de los tratamientos, el participante puede acceder al sistema para desarrollar el tratamiento. El participante no podrá ingresar al sistema hasta que se hayan hecho las asignaciones aleatorias, después el podrá autentificarse y el sistema le mostrará solamente la asignación que debe desarrollar.

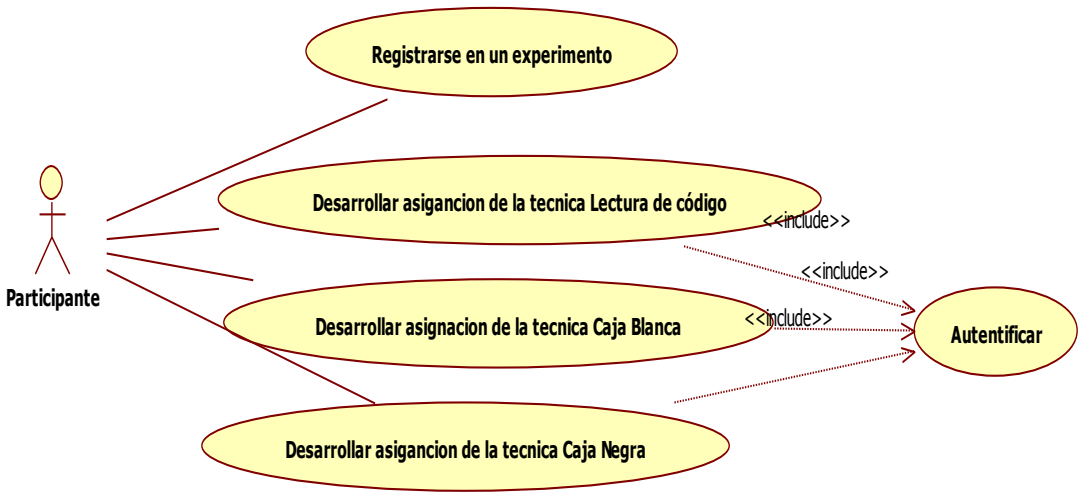

Gráfico 3-2: Diagrama de Casos de uso de un participante

Realizado por: G. Hernández, 2018.

Para que un participante pueda registrarse en un experimento, este debe estar creado, porque el sistema en el formulario de registro le pedirá al participante que ingrese el nombre de la práctica y los datos del usuario. El participante no podrá ingresar al sistema hasta que el experimentador corra el experimento y se hayan realizado las asignaciones. En el caso que no se pueda registrar el participante debe comunicarse con el experimentador para que él le registre en la práctica. En la Tabla 12-2 se puede apreciar el caso de uso de lo descrito anteriormente.

Tabla 12-2: Caso de uso 11 para que un participante pueda registrarse en un experimento

CU-11		
Nombre: Registrarse en un experimento		
Propósito: Registrarse en un experimento para acceder a una práctica.		
Pre-condición: El experimento debe estar creado		
Post-condición: Se ejecutara la aleatorización y asignación de los tratamientos a los participantes		
Actores primarios: Participante		
Escenarios		
Descripción detallada		
Secuencia del escenario principal		
El participante se registra	**Paso**	**Descripción**
	1	El participante accede a la página principal del sistema
	2	El participante presiona en el botón "Registrarse"
	3	El participante ingresa su nombre, apellido, correo, contraseña y nombre de la práctica.
	4	El sistema registra el nombre de usuario, tipo de usuario y le da un estado inactivo hasta que se ejecuten las asignaciones.
	5	Guardar los datos
Secuencia del escenario secundario		
El participante no se puede registrar	**Paso**	**Descripción**
	1	El experimentador accede al sistema
	2	El experimentador registra el nombre, apellido, correo, contraseña y selecciona el experimento.
	3	El sistema registra el nombre de usuario, tipo de usuario y le da un estado inactivo hasta que se ejecuten las asignaciones.
	4	Guardar los datos
Información Táctica		
Prioridad: Prioridad Alta (1)	**Interfaz de usuario**: Se requiere una interfaz sencilla.	

Realizado por: G. Hernández, 2018.

En el sistema se implementaron tres técnicas como son: la técnica de lectura de código por abstracciones sucesivas, caja blanca y caja negra. En la **Tabla 13-2** se define el caso de uso de la técnica de lectura de código por abstracciones sucesivas. Un participante puede acceder a una sola técnica, depende del tipo de diseño experimental para que el experimento se realice en una o varias sesiones. Esta técnica se desarrolla en dos tareas, en la primera tarea el sistema le da acceso al participante al código fuente y a las

instrucciones de la tarea. El participante puede ingresar las abstracciones a partir del código fuente, cuando está seguro de que ha realizado todas las abstracciones presiona en el botón Next task, en la siguiente tarea el sistema le da acceso al participante a la especificación y a las abstracciones para que pueda registrar las inconsistencias.

Tabla 13-2: Caso de uso 12 para desarrollar la técnica lectura de código

CU-12		
Nombre: Desarrollar asignación de la técnica Lectura de código		
Propósito: Obtener datos a partir del uso de la técnica		
Pre-condición: El participante debe estar registrado, el experimentador debe ejecutar las asignaciones y finalmente el participante se autentica en el sistema para usar el mismo.		
Post-condición: El participante aporta con datos al aplicar la técnica.		
Actores primarios: Participante		
Escenarios		
Descripción detallada		
Secuencia del escenario principal		
El participante accede a la asignación	**Paso**	**Descripción**
	1	El sistema le muestra la o las practicas del participante
	2	El participante selecciona una práctica de la lista, solo podrá ejecutar aquella que esté disponible en el rango de tiempo.
	3	El participante en el paso uno registra las abstracciones a partir de la especificación del objeto experimental, dicha abstracción consta de la línea de código y una descripción .
	4	El participante presiona el botón “Siguiente tarea”
	5	El sistema verifica que exista por lo menos una abstracción y registra el tiempo empleado en la tarea uno
	6	En el segundo paso no podrá regresar al anterior y registra las inconsistencias que contienen la abstracción, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.
	7	El participante presiona el botón “Terminar práctica”
	8	El sistema verifica que por lo menos exista una inconsistencia y registra el tiempo empleado en la tarea dos.
	9	El sistema actualiza el estado de la práctica como finalizada.
Secuencia del escenario secundario		
El participante edita los datos de la técnica de Lectura de código.	**Paso**	**Descripción**
	3	En el paso uno registra las abstracciones a partir de la especificación del objeto experimental
	3.1	El participante selecciona una abstracción de la lista y presiona en el botón con un lápiz
	3.2	El sistema muestra los datos
	3.3	El participante edita los datos y presión a el botón Editar.
	4	El participante presiona el botón “Siguiente tarea”
	5	El sistema verifica que por lo menos existe una abstracción y registra el tiempo empleado en la tarea uno

(Continua)

	6	En el segundo paso no podrá regresar al anterior y registra las inconsistencias que contienen la abstracción, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.
	6.1	El participante selecciona una inconsistencia de la lista y presiona en el botón con un lápiz
	6.2	El sistema muestra los datos
	6.3	El participante edita los datos y presión el botón "Editar".
	7	El participante presiona el botón "Terminar práctica"
	8	El sistema registra el tiempo empleado en la tarea dos.
	9	El sistema verifica que existe por lo menos una inconsistencia.
	10	El sistema actualiza el estado de la práctica como finalizada, guarda los datos
Secuencia del escenario con excepciones		
El participante elimina los datos de la técnica de Lectura de código.	**Paso**	**Descripción**
	3	En el paso uno registra las abstracciones a partir de la especificación del objeto experimental
	3.1	El participante selecciona una abstracción de la lista y presiona en el botón con una "X"
	3.2	El sistema elimina esa abstracción
	4	El participante presiona el botón "Siguiente tarea", no podrá pasar a la siguiente tarea sino tiene por lo menos una abstracción.
	5	El sistema registra el tiempo empleado en la tarea uno
	6	En el segundo paso no podrá regresar al anterior y registra las inconsistencias que contienen la abstracción, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.
	6.1	El participante selecciona una inconsistencia de la lista y presiona el botón con una "X"
	6.2	El sistema elimina esa inconsistencia.
	6.3	El participante edita los datos de la inconsistencia.
	7	El participante presiona el botón "Terminar práctica"
	8	El sistema registra el tiempo empleado en la tarea dos.
	9	El sistema verifica que existe por lo menos una inconsistencia.
	10	El sistema actualiza el estado de la práctica como finalizada, guarda los datos
Información Táctica		
Prioridad: Prioridad Alta (1)		**Interfaz de usuario**: Se requiere una interfaz sencilla.

Realizado por: G. Hernández, 2018.

La técnica de caja blanca está documentada en la **Tabla 14-2** en la cual se puede apreciar los pasos que debe seguir el participante para desarrollar dicha técnica. Esta técnica tiene dos tareas, en la primera tarea el sistema muestra el código ejecutable y los participantes deben generar los casos de prueba, en la segunda tarea ellos se descargan la especificación y tienen acceso a sus casos de prueba con ello pueden registrar los defectos encontrados. Un caso de prueba está compuesto por el objetivo, los datos de prueba y la salida esperada.

Tabla 14-2: Caso de uso 13 para desarrollar la técnica caja blanca

CU-13		
Nombre: Desarrollar asignación de la técnica Caja Blanca		
Propósito: Obtener datos a partir del uso de la técnica		
Pre-condición: El participante debe estar registrado, el experimentador debe ejecutar las asignaciones y finalmente el participante se autentica en el sistema para usar el mismo.		
Post-condición: El participante aporta con datos al aplicar la técnica.		
Actores primarios: Participante		
Escenarios		
Descripción detallada		
Secuencia del escenario principal		
El participante accede a la asignación	**Paso**	**Descripción**
	1	El sistema le muestra la o las prácticas del participante
	2	El participante selecciona una práctica de la lista, solo podrá ejecutar aquella que esté disponible en el rango de tiempo.
	3	El participante en el paso uno registra los casos de prueba que constan del objetivo, datos y la salida esperada.
	4	El participante presiona el botón "Siguiente tarea"
	5	El sistema verifica que exista por lo menos un caso de prueba y registra el tiempo empleado en la tarea uno
	6	En el segundo paso no podrá regresar al anterior y el participante registra las fallas encontradas que contienen el caso de prueba, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.
	7	El participante presiona el botón "Terminar práctica"
	8	El sistema verifica que por lo menos exista una falla y registra el tiempo empleado en la tarea dos.
	9	El sistema actualiza el estado de la practica como finalizada
Secuencia del escenario secundario		
El participante edita los datos de la técnica de Caja Blanca.	**Paso**	**Descripción**
	3	En el paso uno registra las abstracciones a partir de la especificación del objeto experimental
	3.1	El participante selecciona un caso de prueba de la lista y presiona en el botón con un lápiz.
	3.2	El sistema muestra los datos del caso de prueba
	4	El participante presiona el botón "Siguiente tarea"
	5	El sistema registra el tiempo empleado en la tarea uno
	6	En el segundo paso no podrá regresar al anterior y registra las fallas encontradas que contienen el caso de prueba, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.

(Continua)

	6.1	El participante selecciona una falla de la lista y presiona en el botón con un lápiz
	6.2	El sistema muestra los datos
	6.3	El participante edita los datos de la falla.
	7	El participante presiona el botón "Terminar práctica"
	8	El sistema registra el tiempo empleado en la tarea dos.
	9	El sistema verifica que existe por lo menos una falla.
	10	El sistema actualiza el estado de la práctica como finalizada.
Secuencia del escenario con excepciones		
El participante elimina los datos de la técnica de Caja Blanca.	**Paso**	**Descripción**
	3	En el paso uno registra las abstracciones a partir de la especificación del objeto experimental
	3.1	El participante selecciona un caso de prueba de la lista y presiona en el botón con una "X"
	3.2	El sistema elimina ese caso de prueba
	4	El participante presiona el botón "Siguiente tarea"
	5	El sistema registra el tiempo empleado en la tarea uno
	6	En el segundo paso no podrá regresar al anterior y registra las fallas encontradas que contienen el caso de prueba, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.
	6.1	El participante selecciona una falla de la lista y presiona en el botón con una "X"
	6.2	El sistema elimina los datos de la falla.
	6.3	El participante edita los datos de la falla.
	7	El participante presiona el botón "Terminar práctica"
	8	El sistema verifica que existe por lo menos una falla.
	9	El sistema registra el tiempo empleado en la tarea dos.
	10	El sistema actualiza el estado de la práctica como finalizada.
Información Táctica		
Prioridad: Prioridad Alta (1)		**Interfaz de usuario**: Se requiere una interfaz sencilla.

Realizado por: G. Hernández, 2018.

La técnica de caja negra posee cuatro tareas, en la primera tarea los participantes deben generar las clases de equivalencia válidas o no válidas, en el segundo paso deben generar los casos de prueba a partir de las clases ingresadas, en el tercer paso los participantes registran la salida observada del caso de prueba y en el último paso los participantes registran los defectos encontrados. Una clase de equivalencia consta de la

descripción y de un estado válido o no. Los casos de prueba están compuestos por una o varias clases de equivalencia, por los datos de prueba y la salida esperada. En la **Tabla 15-2** se observa el caso de uso para desarrollar la técnica de caja negra.

Tabla 15-2: Caso de uso 14 para desarrollar la técnica caja negra

CU-14		
Nombre: Desarrollar asignación de la técnica Caja negra		
Propósito: Obtener datos a partir del uso de la técnica		
Pre-condición: El participante debe estar registrado, el experimentador debe ejecutar las asignaciones y finalmente el participante se autentica en el sistema para usar el mismo.		
Post-condición: El participante aporta con datos al aplicar la técnica.		
Actores primarios: Participante		
Escenarios		
Descripción detallada		
Secuencia del escenario principal		
El participante accede a la asignación	**Paso**	**Descripción**
	1	El sistema le muestra la o las prácticas del participante
	2	El participante selecciona una práctica de la lista, solo podrá ejecutar aquella que esté disponible en el rango de tiempo.
	3	El participante en el paso uno registra las clases de equivalencia que consta de una descripción y si son válidas o no.
	4	El participante presiona el botón "Siguiente tarea"
	5	El sistema verifica que exista por lo menos una clase de equivalencia y registra el tiempo empleado en la tarea uno
	6	En el segundo paso no podrá regresar al anterior y el participante registra los casos de prueba que contienen la clase de equivalencia, los datos de prueba y los datos esperados.
	7	El participante presiona el botón "Siguiente tarea"
	8	El sistema verifica que exista por lo menos un caso de prueba y registra el tiempo empleado en la tarea dos
	9	En el paso tres no podrá regresar al anterior y el participante registra la salida actual de acuerdo al caso de prueba.
	10	El participante presiona el botón "Siguiente tarea"
	11	El sistema verifica que por lo menos exista un caso de prueba y registra el tiempo empleado en la tarea tres.
	12	En el paso cuatro no podrá regresar al anterior y registra las fallas encontradas que contienen el caso de prueba, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.
	13	El participante presiona el botón "Finalizar práctica"
	14	El sistema verifica que por lo menos exista un caso de prueba y registra el tiempo empleado en la tarea cuatro
	15	El sistema actualiza el estado de la práctica como finalizada

(Continua)

Secuencia del escenario secundario		
El participante edita los datos de la técnica de Caja Negra.	**Paso**	**Descripción**
	3	El participante en el paso uno registra las clases de equivalencia que consta de una descripción y si son válidas o no.
	3.1	El participante selecciona una clase de equivalencia de la lista y presiona en el botón con un lápiz.
	3.2	El sistema muestra los datos de la clase
	3.3	El participante edita los datos de la clase de prueba.
	4	El participante presiona el botón "Siguiente tarea"
	5	El sistema verifica que exista por lo menos una clase de equivalencia y registra el tiempo empleado en la tarea uno
	6	En el segundo paso no podrá regresar al anterior y el participante registra los casos de prueba que contienen la clase de equivalencia, los datos de prueba y los datos esperados.
	6.1	El participante selecciona un caso de prueba de la lista y presiona en el botón con un lápiz.
	6.2	El sistema muestra los datos del caso de prueba
	6.3	El participante edita los datos del caso de prueba.
	7	El participante presiona el botón "Siguiente tarea"
	8	El sistema verifica que exista por lo menos un caso de prueba y registra el tiempo empleado en la tarea dos
	9	En el paso tres no podrá regresar al anterior y el participante registra la salida actual de acuerdo al caso de prueba.
	9.1	El participante selecciona la salida actual de un caso de prueba de la lista y presiona en el botón con un lápiz.
	9.2	El sistema muestra la salida actual del caso de prueba
	9.3	El participante edita la salida actual del caso de prueba.
	10	El participante presiona el botón "Siguiente tarea"
	11	El sistema verifica que por lo menos exista un caso de prueba y registra el tiempo empleado en la tarea tres.
	12	En el paso cuatro no podrá regresar al anterior y registra las fallas encontradas que contienen el caso de prueba, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.
	12.1	El participante selecciona una falla de la lista y presiona en el botón con un lápiz.
	12.2	El sistema muestra la falla
	12.3	El participante edita los datos de la falla.
	13	El participante presiona el botón "Finalizar práctica"
	14	El sistema verifica que por lo menos exista un caso de prueba y registra el tiempo empleado en la tarea cuatro
	15	El sistema actualiza el estado de la práctica como finalizada

(Continua)

Secuencia del escenario con excepciones		
El participante elimina los datos de la técnica de Caja Negra.	**Paso**	**Descripción**
	3	El participante en el paso uno registra las clases de equivalencia que consta de una descripción y si son válidas o no.
	3.1	El participante selecciona una clase de equivalencia de la lista y presiona en el botón con una "X"
	3.2	El sistema elimina esa clase de equivalencia
	4	El participante presiona el botón "Siguiente tarea"
	5	El sistema verifica que exista por lo menos una clase de equivalencia y registra el tiempo empleado en la tarea uno
	6	En el segundo paso no podrá regresar al anterior y el participante registra los casos de prueba que contienen la clase de equivalencia, los datos de prueba y los datos esperados.
	6.1	El participante selecciona un caso de prueba de la lista y presiona en el botón con una "X"
	6.2	El sistema elimina ese caso de prueba
	7	El participante presiona el botón "Siguiente tarea"
	8	El sistema verifica que exista por lo menos un caso de prueba y registra el tiempo empleado en la tarea dos
	9	En el paso tres no podrá regresar al anterior y el participante registra la salida actual de acuerdo al caso de prueba.
	9.1	El participante selecciona la salida actual de un caso de prueba de la lista y presiona en el botón con una "X"
	9.2	El sistema elimina la salida actual del caso de prueba
	10	El participante presiona el botón "Siguiente tarea"
	11	El sistema verifica que por lo menos exista un caso de prueba y registra el tiempo empleado en la tarea tres.
	12	En el paso cuatro no podrá regresar al anterior y registra las fallas encontradas que contienen el caso de prueba, el resultado esperado, una breve explicación, si fue encontrado con la técnica y el grado de seguridad.
	12.1	El participante selecciona una falla de la lista y presiona en el botón con una "X"
	12.2	El sistema elimina la falla.
	13	El participante presiona el botón "Finalizar práctica"
	14	El sistema verifica que por lo menos exista un caso de prueba y registra el tiempo empleado en la tarea cuatro
	15	El sistema actualiza el estado de la práctica como finalizada
Información Táctica		
Prioridad: Prioridad Alta (1)	**Interfaz de usuario**: Se requiere una interfaz sencilla.	

Realizado por: G. Hernández, 2018.

En total se realizaron 3 diagramas de casos con los siguientes actores: el administrador, el experimentador y el participante. Por cada uno de los diagramas se documentó los escenarios es así que en el primer diagrama se obtuvo 7 escenarios, en el segundo diagrama se obtuvo 3 escenarios y en el diagrama de casos de uso de los participantes se obtuvo 4 escenarios. Finalmente se obtuvo un total de 14 escenarios. El caso de uso para autentificar a los usuarios se encuentra implícito en el módulo de administradores, experimentadores y participantes.

2.7.2. Arquitectura del software

La arquitectura de software se basa en describir el sistema desde el punto de vista de diferentes interesados, tales como usuarios finales, desarrolladores o directores de proyecto. Para facilitar el entendimiento del sistema se ha utilizado el modelo de 4+1 vistas que describe la arquitectura del software usando cinco vistas concurrentes. (Kruchten, 1994). Tal como se muestra en la **Figura 4-2.**

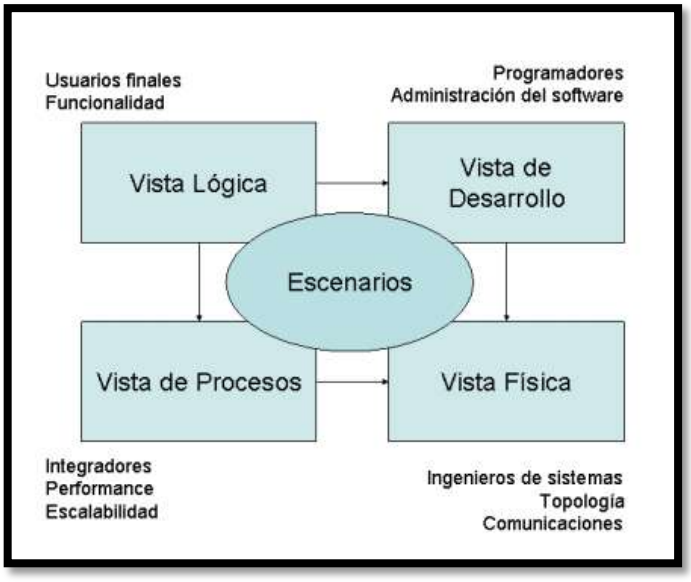

Figura 4-2: Modelo "4+1" vistas.
Fuente: (Kruchten, 1994, pág. 2)

2.7.2.1. Vista lógica

La vista lógico describe el modelo de objetos del diseño cuando se usa un método de diseño orientado a objetos. Para diseñar una aplicación muy orientada a los datos, se puede usar un enfoque alternativo para desarrollar algún otro tipo de vista lógica, tal como diagramas de entidad-relación (Kruchten, 1994).

Con el fin de garantizar la persistencia de la información y datos, para obtener acceso a la información exacta y actualizada, se realizó el diseño de la base de datos mediante un Diagrama Entidad Relación (DER). El **Gráfico 4-2** fue realizado utilizando la aplicación gratuita draw.io de google, el siguiente diseño permite identificar las entidades involucradas en el desarrollo del sistema, además entender las relaciones más relevantes.

Existen 7 entidades que más resaltan y son: Objeto, tratamiento, experimento, usuario, lectura de código, caja blanca y caja negra. Los atributos para cada entidad se encuentran descritos en el diagrama, la identificación de las entidades y atributos se lo realiza utilizando el estándar lowerCamelCasese.

A continuación, se describe las relaciones entre entidades del DER del sistema "SystExp".

- Un experimento está compuesto por un tratamiento, un objeto experimental y una fecha de ejecución.
- Un participante debe inscribirse en el sistema, después se le asigna un experimento.
- Un objeto experimental o también llamado programa es parte de un experimento.
- Un tratamiento o también llamado técnica es parte de un experimento.
- Un usuario puede ser de tipo: administrador, experimentador o participante.

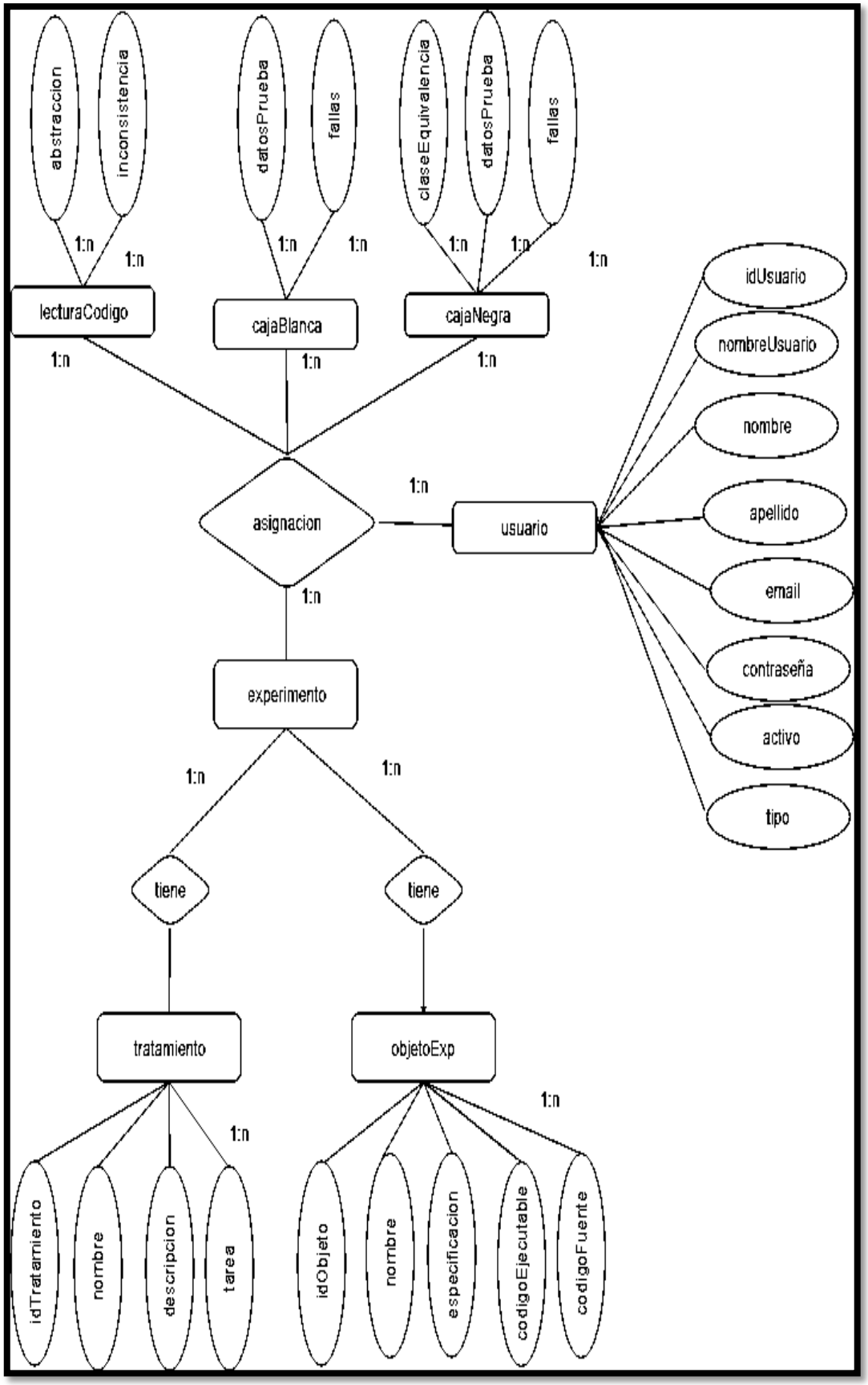

Gráfico 4-2: Diagrama Entidad Relación
Realizado por: G. Hernández, 2018.

2.7.2.2. La vista de procesos

La vista de procesos describe los aspectos de concurrencia y sincronización del diseño. En el transcurso de la investigación realizada se identificó que el proceso de experimentación tiene dos fases muy importantes que son: La fase de preparación del experimento y la fase de ejecución.

Fase de preparación del experimento

En la primera fase el experimentador debe preparar las técnicas y los objetos experimentales o también conocidos como programas. Las técnicas que se involucran en este proceso son: Lectura de código, caja blanca y caja negra. Además el experimentador decide el tipo de diseño experimental. Cuando se tiene un diseño factorial al conjunto de participantes se le asigna de manera aleatoria una técnica y un programa que será desarrollado en una sola sesión.

Sin embargo al tener un diseño de medidas repetidas al conjunto de participantes se les asigna de manera aleatoria una técnica y dos o tres programas dependiendo del número de técnicas. Si el experimentador ha optado por dos técnicas entonces al participante se le asignará una técnica y dos programas que serán desarrollados en dos sesiones. Dichas sesiones no pueden ocurrir el mismo día. El experimentador decide las fechas para las sesiones y la hora.

Fase de ejecución

Los participantes deben llenar los formularios conforme a la técnica y al objeto experimental que se les asigno. Cada técnica difiere entre sí, es así que en la técnica de lectura de código, el participante debe generar las clases de equivalencia y después por cada clase debe ingresar el fallo o defecto encontrado, si fue encontrado con la aplicación de la técnica y señalar el grado de seguridad que sea un fallo.

En la técnica de caja blanca, los participantes deben registrar los casos de prueba y después el resultado esperado, una breve descripción de los fallos encontrados, seleccionar si encontró el fallo al aplicar la técnica y posteriormente el grado de seguridad.

Finalmente en la técnica de caja negra, existen cuatro pasos. En primer lugar los participantes deben registrar las clases de equivalencia válidas o no válidas, después deben generar los casos de prueba a partir de las clases de equivalencia. Cada caso de prueba consta de los datos de prueba, la salida esperada y la salida observada. Para finalizar los participantes registran los fallos encontrados, si fue hallado con el uso de la técnica y el grado de seguridad que sea un fallo. Todo lo descrito anteriormente se visualiza en el **gráfico 5-2.**

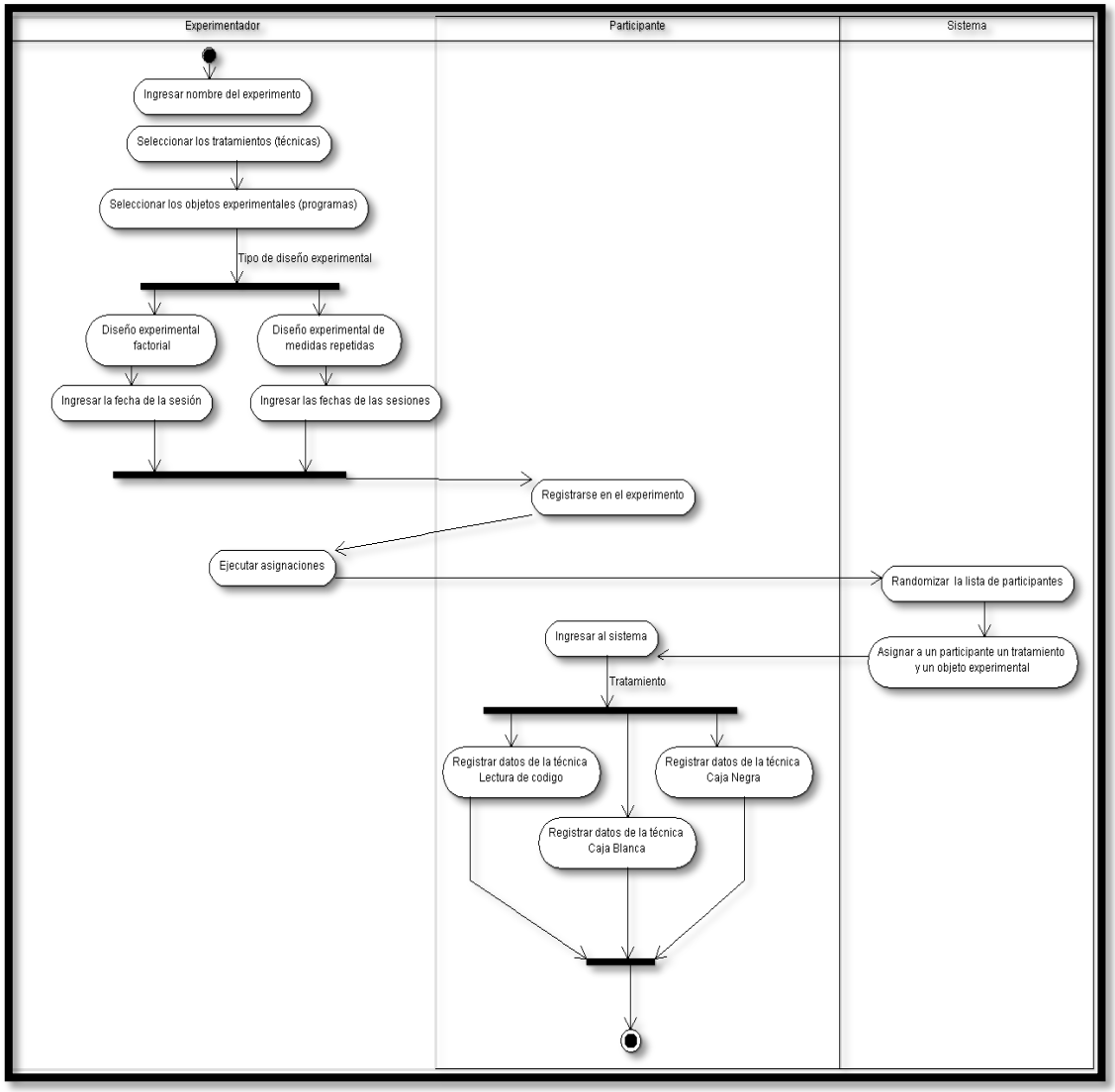

Gráfico 5-2: Procesos de la ejecución de experimentos en IS.
Realizado por: G. Hernández 2018

2.7.2.3. La vista física

La vista física describe el mapeo del software en el hardware y refleja los aspectos de distribución. Para la solución del sistema se optó por la arquitectura de N-capas la cual nos proporciona la escalabilidad del sistema esto permitirá en un futuro mejorar el sistema sin cambiar la arquitectura.

En el **Gráfico 6-2** se muestra los recursos físicos y lógicos necesarios para el funcionamiento del sistema, a continuación se visualiza como se implementó cada uno de los componentes, con la finalidad de proveer los servicios necesarios de manera fácil y segura.

El diagrama de despliegue de la arquitectura del sistema está compuesto por:

- Servidor web, donde se aloja el sistema web.
- Servidor de la base de datos
- PC del cliente.

El sistema se encuentra actualmente en el donimio http://testingtool.sdjournal.net

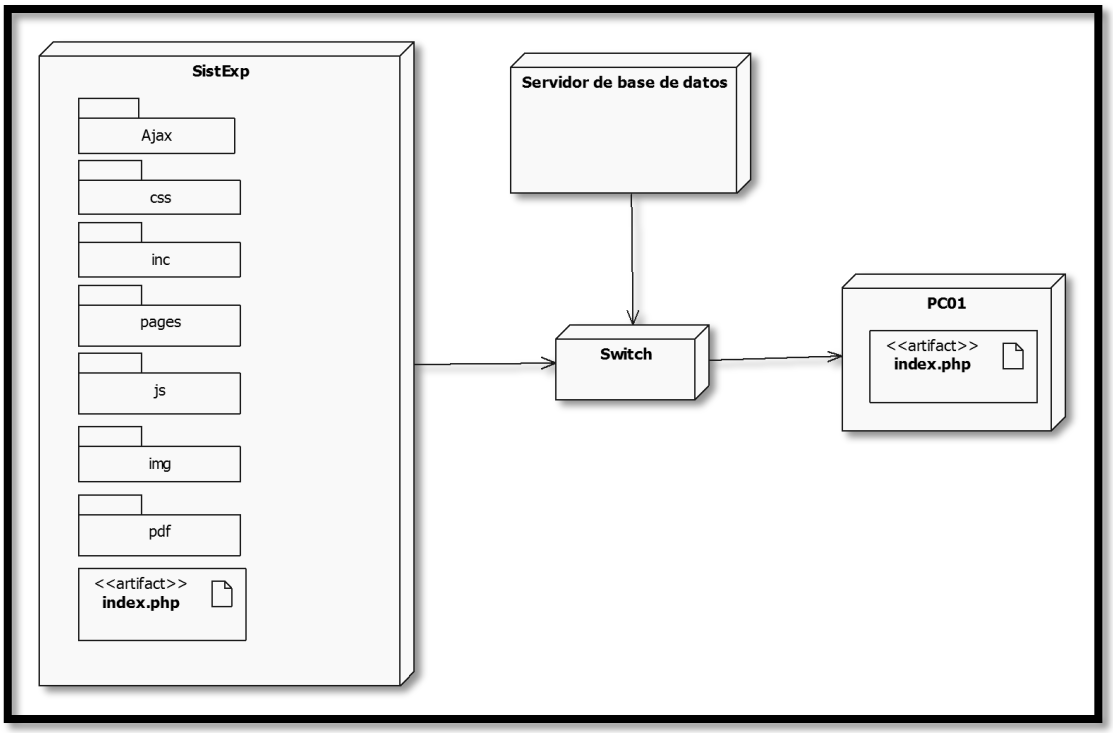

Gráfico 6-2: Diagrama de despliegue del sistema SystExp
Realizado por: G. Hernández, 2018.

Con el equipo de desarrollo se llegó al acuerdo de utilizar una arquitectura en 7 capas. Para la implementación de este requerimiento no funcional se realizó la respectiva historia técnica, tareas de ingeniería y pruebas de aceptación las cuales están desarrolladas a continuación.

2.7.2.4. La vista de desarrollo

La vista de desarrollo describe la organización estática del software en su ambiente de desarrollo (Kruchten, 1994).

Para el desarrollo del sistema se utilizó AJAX es el acrónimo de Asynchronous JavaScript And XML, que es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Se incorporó varias hojas de estilo, que no es más que una colección de reglas que afectan a la apariencia de un documento html, dichas hojas se muestra en el siguiente **Gráfico 7-2.**

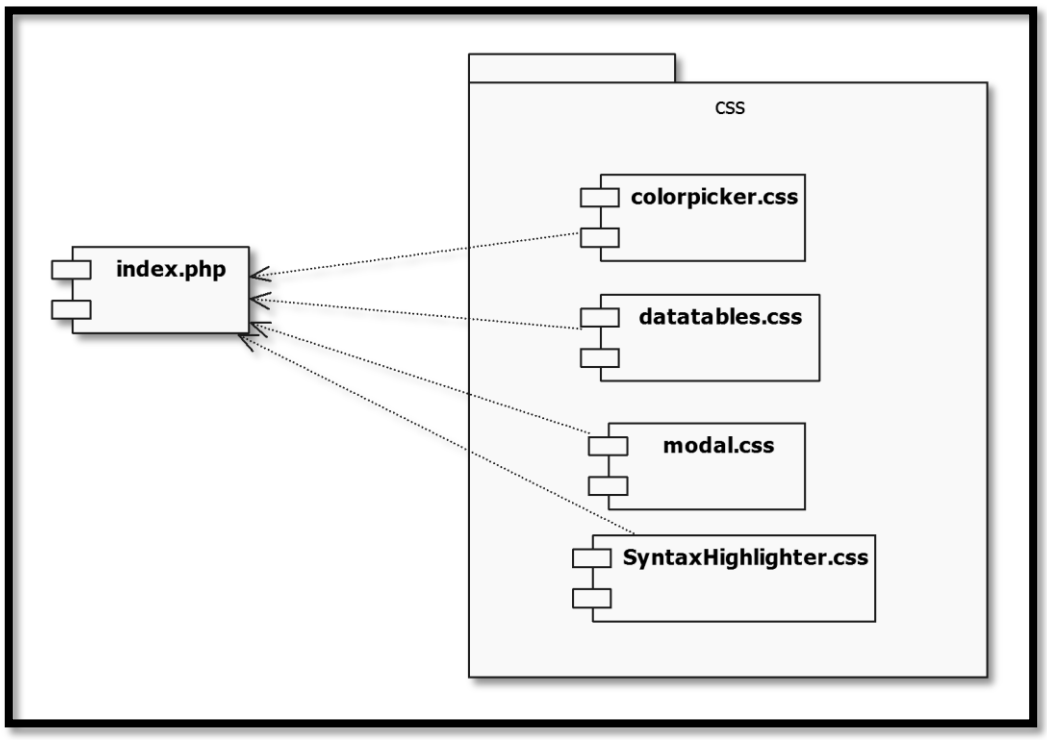

Gráfico 7-2: Diagrama de despliegue del sistema SystExp
Realizado por: G. Hernández, 2018.

En el **Gráfico 7-2** se puede observar los ficheros lógicos de la carpeta css. La hoja colorpicker.css permite cambiar los colores de la aplicación de negro o azul y viceversa, la hoja de estilo datatables.css dá el formato a todas las tablas donde se muestran las listas, modal.css es utilizado como una ventana que permite a los participantes registrarse en el sistema, además se utiliza en las tareas de cada técnica con el objetivo de mostrarle a los participantes las instrucciones del sistema. Finalmente SyntaxHighlighter.css es la que da el formato al código fuente, del programa que es parte de la práctica o sesión de experimentación.

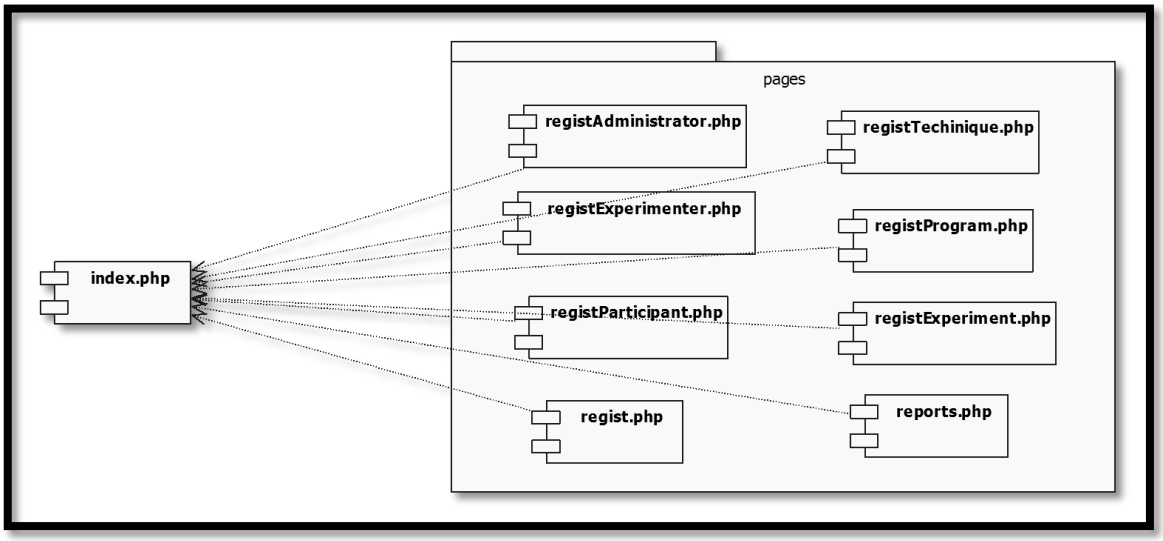

Gráfico 8-2: Diagrama de despliegue del sistema SystExp
Realizado por: G. Hernández, 2018.

En el **Gráfico 8-2** se puede apreciar los diferentes archivos php que se generaron para el manejo del sistema, cada uno de ellos tienen sus propios métodos. Los archivos regsitAdministrator.php, registExperimenter, RegistParticipant.php, registTechinique, registProgram y regsitExperiment tienen los métodos de agregar, editar, consultar y listar. En el archivo regist se localizan todos los métodos para desarrollar alguna de las tres técnicas. Y finalmente en reports.php se encuentran los métodos para realizar alguno de los tres reportes que solicitó el cliente.

2.7.3. Construcción

La construcción de software es desarrollar el sistema utilizando los estándares de software porque tienen una función muy importante en la gestión de calidad del software. Un estándar ofrece reglas, guías o características para que se use repetidamente y así mantener la homogeneidad del sistema.

2.7.3.1. Estándar de programación

LowerCamelCase

Esta técnica de codificación se deriva de la palabra CamelCase que se podría traducir como Mayúsculas/Minúsculas Camello, se realiza una combinación de palabras mayúsculas y minúsculas, en especial en la técnica lowerCamelCase se debe escribir la primera palabra en minúsculas y la primera letra de las siguientes palabras en mayúsculas. De esta manera se garantiza un código entendible, de fácil reutilización y mantenimiento, a continuación se muestra el fichero lógico donde se alojan las páginas web del sistema.

En la **Tabla 16-2** se puede apreciar los ficheros lógicos de la carpeta "pages", es así que se puede evidenciar el uso del estándar de codificación.

Tabla 16-2: Ficheros lógicos del sistema "SystExp"

FICHEROS LÓGICOS		
Pages	**regist:** Archivo php para gestionar los datos una de las tres técnicas, pueden acceder los participantes.	addCodeReading1 addCoddeReading2 addWhiteBox1 addWhiteBox2 addBlackBox1 addBlackBox2
	registAdministrator: Archivo php para gestionar un administrador	addNewAdministrator editAdministrator deleteAdministrator listAdministrator active desactive

(Continua)

	registExperimenter: Archivo php para gestionar un experimentador	addNewExperimenter editExperimenter deleteExperimenter listExperimenter active desactive
	registParticipant: Archivo php para gestionar un participante	addNewParticipant editParticipant deleteParticipant listParticipant active desactive addExperiment
	registExperiment: Archivo php para gestionar un experimento	addNewExperiment editExperiment execute listExperiment active desactive

Realizado por: G. Hernández, 2018.

2.7.3.2. Estándar de interfaces

La interfaz de usuario es el medio con que el usuario puede comunicarse con un equipo, en general, el estándar de interfaces está dado por un menú lateral al costado izquierdo y un área de contenido.

Como pedido del cliente se utilizó Ajax (Asynchronous JavaScript and XML) y CSS. Ajax se refiere a un grupo de tecnologías utilizadas para desarrollar aplicaciones web con el objeto XMLHttpRequest para manipular los datos de forma asíncrona con el servidor web. XML, HTML y XSLT para el intercambio y la manipulación de datos.

Mientras que CSS es un lenguaje usado en la presentación de documentos HTML o páginas web, CSS permite organizar la presentación y aspecto de una página web.

El diseño de las interfaces de usuario del sistema “SystExp”, inicialmente se lo realizo como un bosquejo hecho en Word, con el mismo se comenzó a definir el estándar que se utilizó para la implementación en el sistema de apoyo a la ejecución de experimentos en el área de IS.

En el **Gráfico 9-2** se puede observar la pantalla principal del sistema, y se puede apreciar un menú lateral al costado izquierdo y un área para el contenido para desplegar las funciones del sistema.

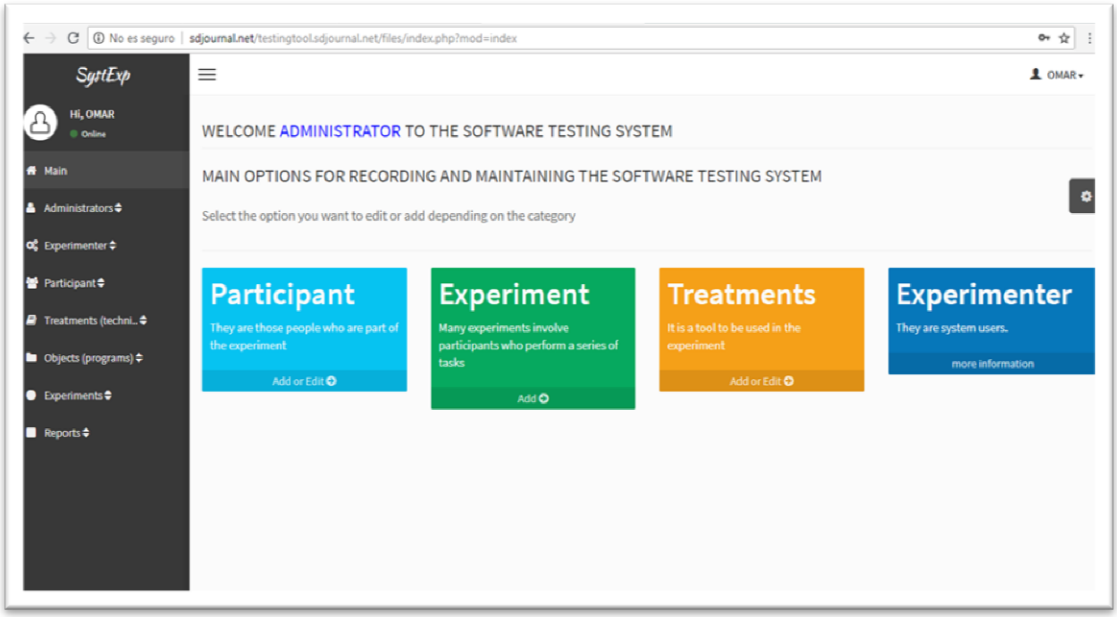

Gráfico 9-2: Pantalla principal del sistema “SystExp”
Realizado por: G. Hernández, 2018.

A continuación se describe el estándar utilizado en el desarrollo del sistema:

- En general para la presentación de las páginas se usó la gama de colores grises claros y azules.
- Los botones están en la parte inferior del formulario y están centrados.
- Los botones tienen la propiedad de resizable, es decir pueden adaptarse a cualquier tamaño de pantalla incluso es posible ver la aplicación en un celular.
- Los botones tienen un color azul, el valor en hexadecimal es “#5DADE2”
- Las letras de las etiquetas label estarán con letra calibre y tamaño 12.

- La letra de los botones de los formularios estarán con letra calibre, tamaño de 12 y en color celeste.
- Los títulos de los formularios tienen la letra calibre, tamaño 18 y en negrita
- La presentación de listas se encuentra dado por una hoja de estilo denominada datatables.css
- Los textboxs, labels, comboboxs tienen la propiedad resizable es decir se ajustan al formulario.
- Para el color de fondo de los formularios se usó los colores en la gama de grises claros.

- Los mensajes de aceptación están en color verde claro, en el **Gráfico 10-2** se observa un claro ejemplo de este tipo de mensaje.

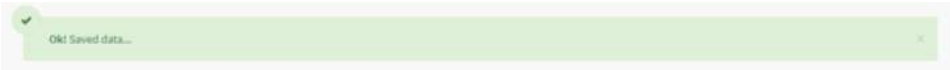

Gráfico 10-2: Mensaje afirmativo

Realizado por: G. Hernández, 2018.

- Los mensajes de error y advertencia se encuentran en color salmón, en el **Gráfico 11-2** se observa el diseño de un mensaje de error.

Gráfico 11-2: Mensaje de error

Realizado por: G. Hernández, 2018.

Estándar de interfaz para los formularios en general, en el **Gráfico 12-2** se puede apreciar el uso del estándar, se muestra un menú lateral izquierdo y en el área de contenido se muestra el formulario para el ingreso de una cuenta de administrador. Este tipo de formulario es utilizado para el ingreso y edición de datos. Los botones le muestran al usuario las acciones que puede realizar.

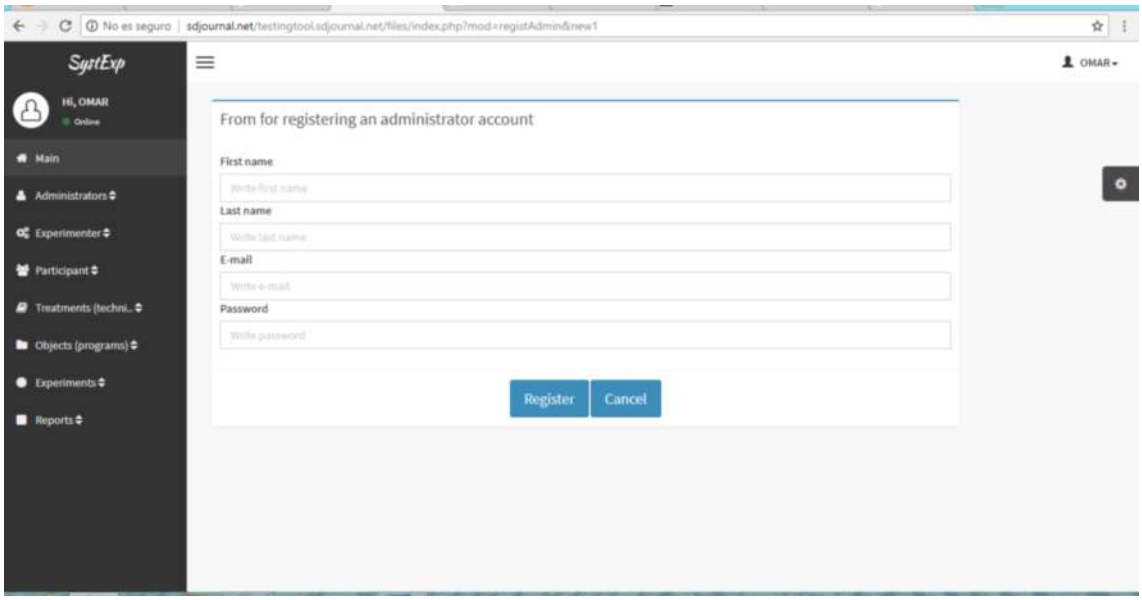

Gráfico 12-2: Formulario de ingreso de una cuenta administrador

Realizado por: G. Hernández, 2018.

Estándar de interfaz para la lista de datos, las listas se presentan en una tabla de datos, en el **Gráfico 13-2** se puede observar cómo se muestran los datos de los administradores además en la columna action el usuario puede seleccionar acciones sobre cada una de las filas que se presenta, existe un botón de acceso rápido para registrar. Además se conserva el estándar de interfaces al presentar una pantalla con el menú lateral izquierdo y un área de contenido donde se muestra la lista de datos.

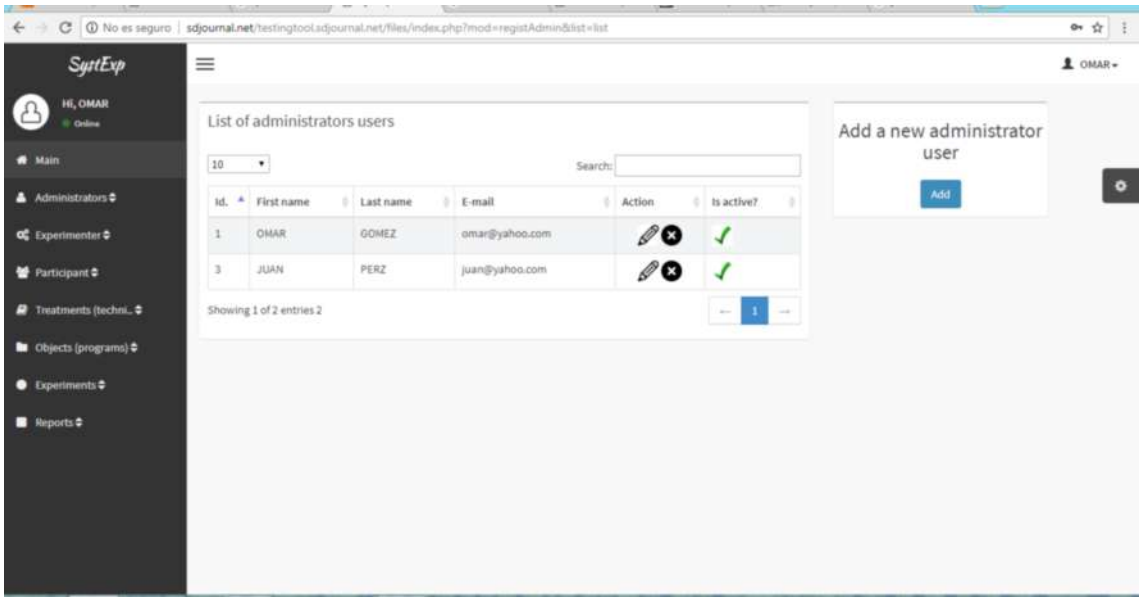

Gráfico 13-2: Lista de administradores

Realizado por: G. Hernández, 2018.

Estándar de interfaz para las tareas de una técnica, un participante accede a la plataforma para realizar una práctica. Por cada uno de los tratamientos existe una lista de tareas que el participante debe realizar. En el **Gráfico 14-2** se puede apreciar que cada tarea siguió el mismo estándar y por ello se utilizó un menú lateral con los instrumentos necesarios para realizar la práctica, un área de contenido que se encuentra dividida en tres partes.

En la parte superior se observa el formulario para el registro de datos y un botón denominado “instructions” dicho botón muestra al participante las instrucciones para desarrollar la práctica.

En la parte inferior se puede apreciar una tabla con la información ingresada por el participante y en el lado superior derecho los datos relevantes para la realización de la práctica, cabe recalcar que en la primera tarea de cada una de las tres técnicas se omite estas cajas de datos.

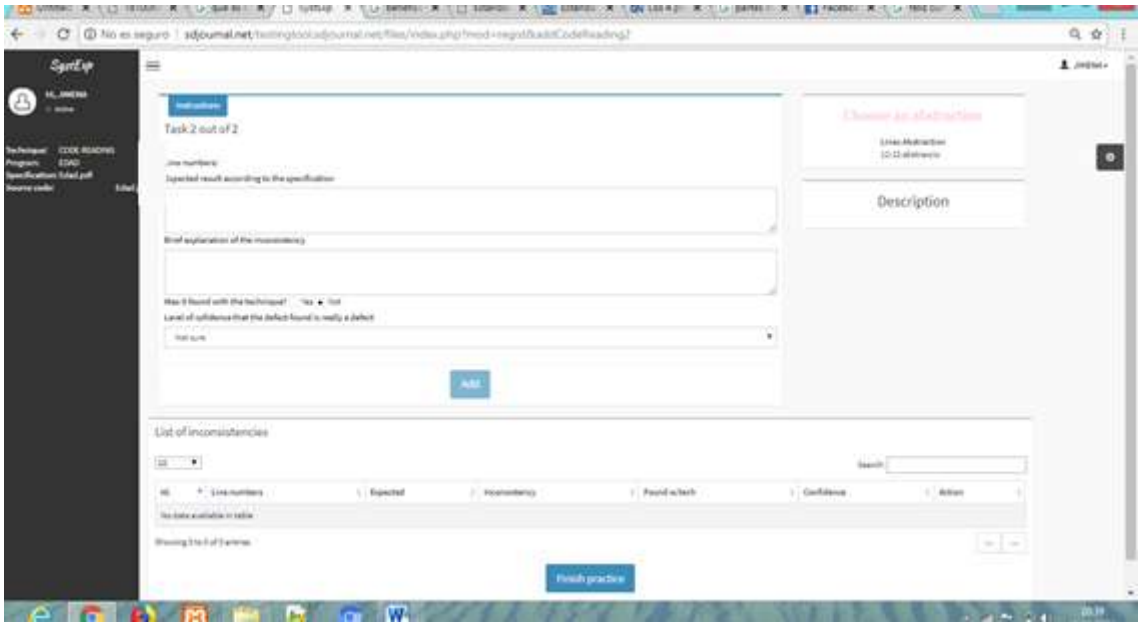

Gráfico 14-2: Página para desarrollar una práctica

Realizado por: G. Hernández, 2018.

2.7.3.3. Diseño de la base de datos

Modelo Entidad Relación, es una herramienta para el modelado de datos por que permite representar las entidades más relevantes de un sistema así como sus interrelaciones y propiedades, dicho modelo se encuentra documentado en la arquitectura del software.

Normalización de los datos, la normalización de los datos se lo realiza con el objetivo de eliminar redundancias e inconsistencias de dependencia en el diseño de las tablas por lo que se analizó los datos del diagrama entidad relación; tablas, atributos y relaciones, con ello se ha podido minimizar el impacto que estos tendrían en el sistema, de esta manera la información es guardada de manera eficaz, fiable, flexible y apropiada.

En total sean aplicado las tres formas de normalización y son descritas a continuación:

La primera forma normal que se aplicó se obtuvo una única fila; es decir que se impide a un campo contener múltiples valores, para ello se estableció un identificador para cada tabla, lo que garantiza unicidad e integridad de los datos. La segunda forma normal fue aplicada para que no exista redundancia de los datos en alguna tabla, lo que garantiza la inserción de un registro sin un exceso de datos en las tablas.

La tercera forma normal se aplicó con el fin de eliminar la dependencia transitiva evitando errores de lógica cuando se insertan o borran registros, lo que garantiza la facilidad de trabajo y expansión.

Finalmente se obtuvo el diagrama lógico de la base de datos como se muestra en el **Gráfico 15-2**, dicho diagrama está compuesto por 16 tablas obtenidas del proceso de normalización antes mencionado, para la creación del diagrama se empleó la herramienta StartUML, el nombre de cada tabla y atributo están en el idioma Inglés como parte de uno de los requerimientos no funcionales del usuario.

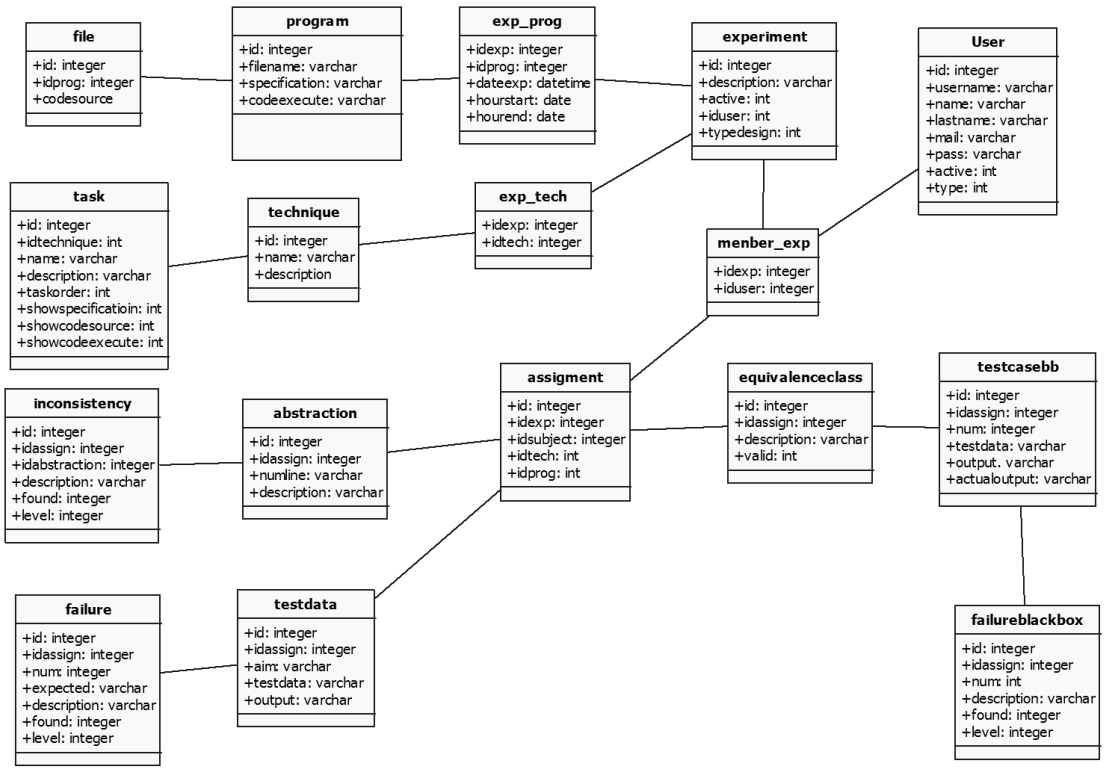

Gráfico 15-2: Diagrama de clases del sistema "SystExp"
Realizado por: G. Hernández, 2018.

Los programas también son conocidos como objetos experimentales, se puede observar en el **Gráfico 15-2** que existe una clase denominada program, donde se almacena la descripción, el nombre del archivo con la especificación y el nombre del archivo con el código ejecutable además está clase se relaciona con la clase file porque se detectó que un programa está compuesto por uno o varios archivos con el código fuente.

La clase technique también es conocida como treatments, está clase está compuesta por un nombre, la descripción y posee varias tareas que son almacenadas en la clase task. Por cada tarea se necesita mostrar al participante la especificación, el código ejecutable o el código fuente, en la clase task estos tres atributos son de tipo booleano porque el experimentador debe seleccionar que instrumentos pretende mostrar en cada tarea.

La clase program y la clase technique son primordiales en la preparación del experimento. El experimentador debe preparar el experimento que posee un nombre, una descripción, un tipo de diseño experimental, las técnicas y los objetos experimentales. Un diseño experimental puede ser factorial o de medidas repetidas. El sistema registra el propietario del experimento y solo el administrador puede asignar a

un experimentador el experimento para ser ejecutado. Todos estos datos son almacenados en la clase experiment.

La fecha de ejecución se registra en la clase exp_prog, porque en un diseño experimental de medidas repetidas el experimentador deberá asignar la fecha con la que el participante ejecutará la técnica y uno de los programas. Cada programa se ejecuta en sesiones diferentes pero con la misma técnica.

Una vez que el experimento está listo, los participantes deben registrarse en el sistema e ingresar los datos como: el nombre, el apellido, el mail con el que ingresarán al sistema, su contraseña y finalmente el nombre del experimento. Si el participante tiene alguna dificultad para registrarse el experimentador o administrador puede realizar esta acción. Los participantes registrados no podrán ingresar al sistema hasta que el experimentador ejecute las asignaciones. Los datos de los participantes son almacenados en la clase user y en la clase member_exp se registra el id del participante y el id del experimento.

Cuando todos los participantes se han registrado en el sistema, el experimentador ejecuta las asignaciones que son realizadas por el sistema de manera aleatoria, respetando el tipo de diseño. En un diseño experimental factorial el sistema asigna una técnica y un programa, en cambio cuando tenemos un diseño experimental de medidas repetidas el sistema asigna una técnica y varios programas que serán ejecutados en las fechas establecidas.

Dichas asignaciones son almacenadas en la clase assigment. El sistema activa a los participantes, finalmente ellos pueden acceder al sistema con el mail y la contraseña. Depende de la asignación que tenga cada participante se almacenaran los datos.

En la técnica de lectura de código, la primera tarea registra las abstracciones en la clase abstraction, las mismas que serán usadas en la segunda tarea para registrar las inconsistencias en la clase insonsistency además se registra si las inconsistencias fueron encontradas con la técnica y cuál es el nivel de seguridad. Al igual que la técnica anterior, la técnica de caja blanca tiene dos pasos en los que se registra los casos de prueba en la clase testdata, y las fallas en la clase failures.

Sin embargo para la técnica de caja negra se registra las clases de equivalencia que son válidas o no válidas. Dentro del sistema dichas clases son almacenadas en la clase equivalenceclass. Los casos de prueba son almacenados en la clase testcasebb y

finalmente se registra las fallas en la clase failureblackbox. Se realizó el diccionario de datos que sirve de referencia para trabajar bajo un mismo dominio de datos a fin de evitar confusiones y aumentar la integridad del repositorio de datos.

Diccionario de datos. Es el conjunto de metadatos que contiene las características lógicas y puntuales de los datos tales como la descripción, validaciones, tipos de datos y tamaño de cada campo que se van a utilizar en el sistema. El esquema de la base de datos se lo obtiene a partir de la problemática a resolver y los requerimientos de usuario planteados anteriormente. Para lo cual se parte del modelo lógico se genera el código correspondiente.

En la **Tabla 17-2** se observa la entidad experiment donde se registran los datos de un experimento, cada atributo está compuesto por la descripción, el tipo de dato, si necesita ser validado y si este atributo es obligatorio o no.

Tabla 17-2: Diccionario de la tabla experiment

Tabla: experiment				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Id	Serial	Código del experimento PK	Null	True
description	Varchar(100)	Nombre del experimento	No	True
Active	Boolean	Si está activo el experimento	(1-0)	True
Iduser	Int	Código del usuario que creo el experimento	Solo números	True
Typedesign	Int	Tipo de diseño experimental	(1-2)	True

Realizado por: G. Hernández, 2018.

En la tabla user se almacenan los datos de los usuarios del sistema, existen tres tipos de usuarios como son: administrador, experimentador y participante. En la **Tabla 18-2** se puede apreciar los atributos de la tabla user. En esta tabla todos los atributos son obligatorios.

Tabla 18-2: Diccionario de la tabla user

Tabla: user				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Id	Serial	Código del usuario	Null	True
Username	Varchar(100)	Nombre del usuario	Solo letras	True
Name	Varchar(50)	Nombre del usuario	Solo letras	True
lastName	Varchar(50)	Apellido del usuario	Solo letras	True
Mail	Varchar(50)	Correo del usuario	Formato correo electrónico	True
Pass	Varchar(50)	Contraseña	No	True
Active	Boolean	Si está activo el experimento	(1-0)	True
Type	Int	Tipo de usuario	1: Administrador 2: Experimentador 3: Participante	True

Realizado por: G. Hernández, 2018.

En la tabla exp-prog se registra la relación entre los objetos experimentales y el experimento, todos los atributos son obligatorios, cada uno de ellos es validado a nivel de la interfaz del sistema, en la **Tabla 19-2** se observa más a fondo lo dicho anteriormente.

Tabla 19-2: Diccionario de la tabla exp_prog

Tabla: exp_prog				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Idexp	Int	Pk	Null	True
Idprog	Int	Pk	Solo letras	True
Dateexp	Datetime	Fecha de ejecución	Fecha	True
Hourstart	Date	Hora de inicio	Hora	True
Hourend	Date	Hora de finalización	Hora	True

Realizado por: G. Hernández, 2018.

En la **Tabla 20-2** se observa la entidad program donde se registra el objeto experimental o programa y cada uno de sus atributos.

Tabla 20-2: Diccionario de la tabla program

Tabla: program				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Id	Serial	Código	Null	True
Filename	Varchar(50)	Nombre	No	True
Specification	Varchar(50)	Nombre de la especificación	No	True
Codeexecute	Varchar(50)	Nombre del código ejecutable	No	True

Realizado por: G. Hernández, 2018.

Un programa u objeto experimental puede tener uno o varios archivos con el código fuente, es por ello que en la **Tabla 21-2** se evidencia la entidad file y sus atributos, donde cada uno de ellos son obligatorios.

Tabla 21-2: Diccionario de la tabla file

Tabla: file				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Id	Serial	Código	Null	True
idProg	Int	PK	Solo números	True
codeSource	Varchar(50)	Nombre del código fuente.	No	True

Realizado por: G. Hernández, 2018.

En la **Tabla 22-2** se observa la entidad exp_tech, representa la relación de varios a varios, porque una técnica puede estar dentro de varios experimentos y un experimento tiene varias técnicas.

Tabla 22-2: Diccionario de la tabla exp_tech

Tabla: exp_tech				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Idexp	Int	Pk	Null	True
Idprog	Int	Pk	Solo letras	True

Realizado por: G. Hernández, 2018.

Una técnica es parte de un experimento, es por ello que se necesita almacenar sus datos y se ha empleado la tabla technique, los atributos y características de la entidad se muestran en la **Tabla 23-2.**

Tabla 23-2: Diccionario de la tabla technique

Tabla: technique				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Id	Serial	Código	Null	True
Name	Varchar(50)	Nombre	No	True
description	Varchar(100)	Descripción de la técnica	No	True

Realizado por: G. Hernández, 2018.

Una técnica está compuesta por varias tareas, y una tarea pertenece solamente a una técnica. En la **Tabla 24-2** se puede evidenciar el diccionario de datos de la entidad task.

Tabla 24-2: Diccionario de la tabla task

Tabla: task				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Id	Serial	Código	Null	True
idTechnique	Int	Pk	No	True
Name	Varchar(100)	Nombre de la tarea	No	True
description	Varchar(1000)	Instrucciones de la tarea	No	True
taskorder	Int	Orden de la tarea	Solo numeros	True
showEspecification	Boolean	Mostrar la especificación	(1-0)	True
showCodeSource	Boolean	Mostrar el código fuente	(1-0)	True
showCodeExecute	Boolean	Mostrar código ejecutable	(1-0)	True

Realizado por: G. Hernández, 2018.

Una vez que los participantes se han registrado en el sistema, el experimentador corre las asignaciones estas son almacenadas en la entidad assigment, cada uno de los atributos son obligatorios.

Tabla 25-2: Diccionario de la tabla assigment

Tabla: assigment				
Columna	**Tipo**	**Descripción**	**Validación**	**Obligatoriedad**
Id	Serial	Código	Null	True
Idexp	Int	PK id experimento	Solo números	True
Idsubject	Int	PK id del participante	Solo números	True
Idtech	Int	PK id de la técnica	Solo números	True
Idprog	Int	PK id del programa	Solo números	True

Realizado por: G. Hernández, 2018.

Finalmente se obtuvo el modelo físico de la base de datos. Además se generó la "Historia técnica" para diseñar la base de datos, como parte de la metodología SCRUM. En las siguientes tablas podemos apreciar el formato utilizado para la documentación de las historias técnicas y de las pruebas de aceptación.

2.7.3.4. Código fuente

El código fuente de un sistema software, es un conjunto de líneas de texto con los pasos que debe seguir el computador para ejecutar dicho programa. Para el desarrollo del sistema se siguió la metodología ágil SCRUM, una de las ventajas de dicha metodología es mostrar al cliente partes del sistema, es decir se entrega al cliente un prototipo implementado y funcional.

A partir del Sprint 3 se entregó parte del sistema con la funcionalidad para gestionar administradores, de ahí en adelante hasta el sprint 14 se implementaron los requerimientos del sistema, en cada reunión del equipo con el cliente se mostró avances del mismo conforme la planificación del Sprint planning, dicha tabla se puede apreciar en el apartado de la gestión del proyecto. Actualmente el código fuente se encuentra en el siguiente dominio: http://github.com/omargomez2/SystExp.git

2.7.4. Verificación, pruebas de aceptación

La verificación busca comprobar que el sistema cumpla con los requerimientos especificados. Las pruebas de aceptación se realizaron como parte de la metodología, para el desarrollo del sistema se aplicó la metodología SCRUM, dicha metodología utiliza los sprints que son intervalos prefijados durante el cual se crea un incremento del producto. En primer lugar se utilizó la técnica de tallas de camisetas o T-shirt, la misma que se encuentra detallada en la gestión del proyecto, se utilizó esta técnica para calcular los tiempos estimados en el desarrollo de cada uno de los requerimientos.

En total se realizaron 15 Sprints cada Sprint fue valorado en 40 puntos y se realizó la verificación de cada uno de ellos, se utilizó las historias técnicas, tareas de ingeniería las cuales fueron documentadas y verificadas mediante las pruebas de aceptación, el desarrollo de toda la metodología SCRUM se encuentra en el documento adjunto (manual técnico)**.** En la reunión con el cliente se expuso la historia técnica y se procedió a realizar la prueba de aceptación.

En la siguiente **Tabla 26-2** se muestra un ejemplo de la historia técnica para diseñar la base de datos.

Tabla 26-2: Historia técnica para diseñar la base de datos

HISTORIA TÉCNICA	
Número: HT-04	**Nombre de la historia técnica:** Diseñar la base de datos
Usuario: Desarrollador	**Sprint Asignado:** 01
Fecha inicial: 21/11/2017	**Fecha de finalización:** 05/12/2017
Descripción: Implementar la base de datos a partir del DER y llevarlo a la tercera normalización.	
Pruebas de aceptación: El diseño de la base de datos está acorde a los requerimientos del cliente	

Realizado por: G. Hernández, 2018.

A continuación en la **Tabla 27-2** se muestra la prueba de aceptación realizada para la historia técnica HT-04.

Tabla 27-2: Prueba de aceptación 4 para HT-04

PRUEBA DE ACEPTACIÓN	
Código: 4	**Nombre de la historia técnica:** Diseñar la base de datos
Nombre de la prueba: El diseño de la base de datos está acorde a los requerimientos del cliente	
Programador Responsable: Hernández J.	**Fecha de ejecución**: 05/12/2017
Descripción: Se realizó el DER y ejecutó hasta la tercera forma normal.	
Condiciones de ejecución Analizar los procesos del sistema.	
Resultado esperado: La base de datos fue implementada en MySQL. Y se ejecutó hasta la tercera forma normal.	
Evaluación de la prueba: Satisfactoria	

Realizado por: G. Hernández, 2018.

La tarea de ingeniería ejecutada para HT-02 se define en la siguiente **Tabla 28-2**.

Tabla 28-2: Actividades realizadas para el Sprint-02

Fecha	Actividad	Tiempo
05/12/2017	**Tarea 1.Diesñar la base de datos** El diseño de la base de datos está acorde a los requerimientos del cliente	40 horas

Realizado por: G. Hernández, 2018

Cada tarea de ingeniería nos permite registrar las actividades realizadas en el sprint, al finalizar se realiza la prueba de aceptación, como se muestra en la **Tabla 29-2.**

Tabla 29-2: Tarea de ingeniería 2 para el Sprint-02

TAREA DE INGENIERÍA	
Sprint: 02	
Nombre de la historia técnica: Desarrollo del Sprint 02	
Tipo de tarea: Análisis y desarrollo	**Programador Responsable:** Hernández J.
Fecha inicial: 21/11/2017	**Fecha final:** 05/12/2017
Descripción: • Realizar el diagrama entidad relación. • Realizar el diagrama lógico de la base de datos. • Realizar el diagrama físico de la base de datos.	
Prueba de aceptación: • El diseño de la base de datos está acorde a los requerimientos del cliente.	

Realizado por: G. Hernández, 2018.

En conclusión se desarrollaron 15 Sprints con un total de 15 pruebas de aceptación, siendo evaluadas de manera satisfactoria, sin embargo existieron 2 pruebas evaluadas de manera insatisfactoria, porque cuando se desarrolló el Sprint 8 para gestionar experimentos, se presentó el prototipo al cliente con el tipo de diseño al final, en ese momento el cliente manifestó que en el último paso no debía ir el diseño experimental. El cliente requería que en el primer paso se ubicara la descripción del experimento y el tipo de diseño.

La otra prueba no aceptada se dio en el Sprint 14 para desarrollar la asignación de la técnica de caja negra. Dicha dificultad se presentó cuando la desarrolladora no comprendió que en la segunda tarea de la técnica, un caso de prueba puede estar compuesto por una o varias clases de equivalencia.

Finalmente se gestionaron las tareas para ajustarlas al siguiente Sprint con el fin de cumplir a cabalidad con la planificación hecha y evitar algún tipo de desfase en el proyecto.

2.8. Gestión del proceso de desarrollo

La gestión de proyectos de desarrollo es un enfoque metódico para planificar y orientar los procesos del proyecto de principio a fin.

2.8.1. Análisis y gestión de riesgos

El análisis y gestión de riesgos se realizó con el fin de identificar los riesgos, y así determinar los controles adecuados para disminuir, supervisar o evitar la ocurrencia de los riesgos que podrían afectar a los objetivos del proyecto de manera negativa.

El proceso comprendió el reconocimiento de los riesgos, así como también su probabilidad de ocurrencia y el impacto que puedo ocasionar dichas situaciones negativas. El análisis y gestión de riegos cuenta con varias etapas a cumplirse como son:

- Identificación de los riesgos
- Análisis de riesgos
- Hojas de gestión de riesgos.
- Priorización de riesgos.

A continuación se muestra el desarrollo de cada una de las etapas.

2.8.1.1. Identificación de los riesgos

En la etapa de identificación fue necesario definir los posibles riesgos que podrían producirse durante el desarrollo del proyecto, con ello también se logró plantear medidas de prevención y gestión ante la posible ocurrencia de alguno de ellos.

Los riesgos fueron clasificados en tres tipos que son: "Del proyecto", "Técnico" o "Del negocio", un riesgo del proyecto es aquel que amenaza con la planificación del proyecto, mientras que un riesgo técnico amenaza la calidad del sistema y amenaza la factibilidad del proyecto, dependiendo de esto, se describe y analiza las posibles consecuencias de cada uno de ellos.

Se han identificado un total de siete riesgos, siendo dos riesgos del proyecto, dos riesgos técnicos y tres riesgos del negocio, con ello se procedió a realizar el análisis correspondiente por cada uno de ellos. Hasta el momento no se ha identificado otro riesgo adicional a los planteados inicialmente, tal como se evidencia en el documento adjunto del manual técnico.

2.8.1.2. Análisis De Riesgos

En la siguiente etapa del análisis de riesgos se determina el grado de probabilidad, el impacto y la exposición de cada uno de los riesgos en el desarrollo del proyecto. Se tiene un identificador (ID) del Riesgo que se representa con la letra mayúscula R y un número secuencial, así como la descripción del mismo.

Después se analiza la probabilidad de ocurrencia del riesgo en una escala: alta, media o baja. Se asigna un porcentaje en intervalos de 1% a 33%, 34% a 67% y del 68% a 99% cada intervalo corresponde a una valoración de 1, 2, o 3 respectivamente. Dichos valores fueron analizados por los miembros del equipo.

Los riesgos pueden causar un retraso en el desarrollo del proyecto, es por ello la importancia de determinar su impacto, para la valoración se tomó en cuenta que una semana de retaso es un impacto bajo con un valor de 1, al tener dos semanas de retaso se produce un impacto moderado con una valoración de 2, mientras que un mes de retraso produce un impacto alto con un valor de 3 y más de un mes de retraso provoca un impacto crítico con un valor de 4.

Al multiplicar la valoración de la probabilidad y el impacto del riesgo se determinó la exposición a la ocurrencia del riesgo, se categorizó como baja al obtener un producto de

1 o 2, media con un producto de 3 o 4 y alta si se tiene un producto mayor a 6, lo dicho anteriormente se evidencia en el documento adjunto del manual técnico.

Este análisis permite verificar qué tan expuesto está el desarrollo del sistema a estos posibles riesgos, tomando en cuenta un porcentaje promedio de probabilidad de ocurrencia del 28.57% el cual es relativamente bajo, la viabilidad del proyecto no se vio afectada y por ello se procedió a realizar las hojas de gestión de cada uno de los riesgos para proponer acciones de prevención de los mismos.

2.8.1.3. Hojas de gestión de riesgos

Cada riesgo es registrado en una hoja de gestión del riesgo que permite obtener un plan de medidas correctivas y preventivas para la reducción, supervisión y gestión del mismo, es decir en el caso de ocurrencia se toman las medidas necesarias.

Las hojas de gestión de riesgos contiene un identificador (ID) del Riesgo que se compone por una letra mayúscula R y un número secuencial, la fecha de realización de dicho documento, la probabilidad, el impacto, la exposición y prioridad con las diferentes valoraciones del riesgo a tratar, así como su descripción; para el refinamiento se expusieron las causas y consecuencias que el riesgo ocasionaría en caso de ocurrencia, todo esto se puede verificar en el documento adjunto (manual técnico).

Se realizó las hojas de gestión de riesgos para cada uno de ellos y se obtuvo un riesgo con un impacto alto y con la exposición alta. Cuando existe un mal diseño en la base de datos se convierte en un riesgo que puede generar la redundancia de información. En las hojas de gestión del documento adjunto (manual técnico) se definió las medidas de reducción, supervisión y gestión de los riesgos, se cumplió de manera correcta y se evitó inconvenientes en el desarrollo del proyecto; los demás riesgos se encontraban con niveles de prioridad, impacto y exposición de media a baja, por lo que se continuó con la priorización de riesgos.

2.8.1.4. Priorización de riesgos

La priorización de riesgos tiene como objetivo determinar el orden en el que deben ser considerados cada uno de los riesgos para su reducción, supervisión o gestión. Para ello se tiene un identificador (ID) del riesgo que se compone con la letra mayúscula R y un número secuencial además la descripción del riesgo.

El estudio se basó en el análisis de los posibles riesgos, que se pensó que ocurrirían en la etapa de inicio, desarrollo o finalización del proyecto, por lo que se priorizaron los riesgos de acuerdo a su rango de exposición (alta, media o baja).

Posteriormente se asignó la prioridad conforme a los valores obtenidos en la exposición al riesgo, siendo ordenados y categorizados desde aquellos que poseen una exposición alta con la prioridad valorada en 1, la exposición media con una prioridad de 2 y exposición baja con prioridad de 3. Con ello se consigue tener mayor control de cuál de ellos puede causar un efecto negativo en cuanto al desarrollo del proyecto en el caso de ocurrencia. En la **Tabla 30-2** se observan los riesgos identificados y priorizados.

Tabla 30-2: Riesgos valorados y priorizados

ID RIESGO	DESCRIPCION	EXPOSICIÓN	
		VALOR	EXPOSICIÓN
R02	Mal diseño de la Base de la base de datos	8	ALTA
R01	Falta de explicación de parte del usuario	4	MEDIA
R06	Retirada transitoria o terminante de algún miembro del equipo de trabajo.	4	MEDIA
R03	Falta de presupuesto para la elaboración del proyecto.	3	MEDIA
R04	Cambios continuos de requerimientos por parte del usuario	3	MEDIA
R07	Falta de apoyo por parte de los miembros del proyecto	2	BAJA
R05	El hardware sea inconsistente con el sistema.	1	BAJA

Realizado por: G. Hernández, 2018.

Finalmente se obtuvo un riesgo con prioridad alta, cuatro con prioridad media y dos con la prioridad baja. Se prestó mayor atención al riesgo de un "Mal diseño en la base de datos" se realizó las medidas de prevención y corrección de aquellos riesgos de

prioridad alta y media por lo que podrían provocar el fracaso del proyecto. Existió un mal entendido con respecto a las necesidades del cliente, por falta de explicación del usuario, es por ello que se realizó las medidas pertinentes y se evitó algún tipo de desfase en el proyecto.

2.8.2. Gestión del proyecto

La gestión de proyectos es un enfoque metódico para planificar y orientar los procesos del proyecto de principio a fin. En la fase de planificación para determinar los puntos estimados sé tomó una técnica denominada "Tallas de camiseta" (McConnell, 2006), porque se ha difundido de gran manera en la gestión de proyectos a través de las metodologías ágiles. Está permite la estimación del esfuerzo que el equipo de trabajo necesita para cumplir con la tarea. En la **Tabla 31-2** se puede apreciar los valores con los que se trabajó.

Tabla 31-2: Aplicación del método T-Shirt

Talla	**Puntos estimados**	**Horas de trabajo**
S	10	10
M	20	20
L	40	40
XL	80	80

Realizado por: G. Hernández, 2018.

Una vez aplicado el método T-Shirt se obtuvo la siguiente tabla denominada Product Backlog, es una lista de objetivos priorizados con el fin de desarrollar el sistema. Dicha lista está compuesta por los requerimientos, los puntos estimados y la prioridad. Se tomó como referencia para las tareas de ingeniería los requerimientos documentados en los casos de uso. En la Tabla 32-2 se visualiza el product backlog, se identificaron los requisitos, se priorizaron y se estimaron a partir de ello se construirá el Sprint planning.

Tabla 32-2: Product Backlog

ID	Requisitos	Prioridad	Puntos estimados	Talla
HT-01	Diseñar de la arquitectura del sistema	1	10	S
HT-02	Establecer el estándar de programación	1	10	S
HT-03	Definir el estándar de interfaz del sistema	2	20	M
HT-04	Diseñar la Base de Datos	2	40	L
CU-01	Gestionar administradores	3	40	L
CU-02	Gestionar experimentadores	2	40	L
CU-03	Gestionar participantes	1	40	L
CU-04	Gestionar técnicas	3	40	L
CU-05	Gestionar objetos experimentales	3	40	L
CU-06	Gestionar experimentos	2	40	L
CU-07	Generar reportes	1	40	L
CU-08	Gestionar participantes con el rol experimentador.	2	20	M
CU-09	Gestionar experimentos con el rol experimentador.	2	20	M
CU-10	Generar reportes con el rol experimentador.	2	20	M
CU-11	Registrarse en un experimento.	3	20	M
CU-12	Desarrollar asignación de la técnica Lectura de código.	2	40	L
CU-13	Desarrollar asignación de la caja blanca.	2	40	L
CU-14	Desarrollar asignación de la caja negra.	2	40	L
HT-05	Implementación y pruebas	2	40	L

Realizado por: G. Hernández, 2018.

Plan de entrega, permite obtener los sprints y fechas de realización de cada versión del proyecto. Un sprint es el conjunto de periodos de tiempo en el que se realiza avances del proyecto, los cuales producen un incremento del mismo.

En la **Tabla 33-2** se muestra la lista de tareas que el equipo elaboró en la reunión de planificación de la iteración (Sprint planning) como plan para completar los

objetivos/requisitos seleccionados para la iteración y que se compromete el equipo a entregar al cliente al finalizar la iteración, en forma de incremento del producto.

Tabla 33-2: Sprint planning

Sprint	**ID**	**Requisitos**	**Total**	**Fecha de inicio**	**Fecha Final**
Sprint 1	HT-01	Diseñar de la arquitectura del sistema	40	07/11/2017	21/11/2017
	HT-02	Establecer el estándar de programación			
	HT-03	Definir el estándar de interfaz del sistema			
Sprint 2	HT-04	Diseñar la Base de Datos	40	21/11/2017	05/12/2017
Sprint 3	CU-01	Gestionar administradores	40	05/12/2017	19/12/2017
Sprint 4	CU-02	Gestionar experimentadores	40	19/12/2017	02/01/2018
Sprint 5	CU-03	Gestionar participantes	40	02/01/2018	16/01/2018
Sprint 6	CU-04	Gestionar técnicas	40	16/01/2018	30/01/2018
Sprint 7	CU-05	Gestionar objetos experimentales	40	30/01/2018	13/02/2018
Sprint 8	CU-06	Gestionar experimentos	40	13/02/2018	27/02/2018
Sprint 9	CU-07	Generar reportes	40	27/02/2018	13/03/2018
Sprint 10	CU-08	Gestionar participantes con el rol experimentador.	40	13/03/2018	27/03/2018
	CU-09	Gestionar experimentos con el rol experimentador.			
Sprint 11	CU-10	Generar reportes con el rol experimentador.	40	27/03/2018	10/04/2018
	CU-11	Registrarse en un experimento.			
Sprint 12	CU-12	Desarrollar asignación de la técnica Lectura de código.	40	10/04/2018	24/04/2018
Sprint 13	CU-13	Desarrollar asignación de la caja blanca.	40	24/04/2018	08/05/2018
Sprint 14	CU-14	Desarrollar asignación de la caja negra.	40	08/05/2018	22/05/2018
Sprint 15	HT-05	Implementación y pruebas	40	22/05/2018	05/06/2018
Total de puntos de estimados			600		

Realizado por: G. Hernández, 2018.

Se estableció para cada Sprint un rango de dos semanas y de 4 horas diarias, por lo que se obtuvo un total de 15 Sprints y 600 puntos estimados, los cuales se cumplieron en su totalidad para culminar con la ejecución y documentación de las acciones propuestas en el plan de entrega de manera organizada y respetando el tiempo establecido, se ejecutaron cada una de las iteraciones planteadas.

Es por ello que las funcionalidades y metáforas del sistema se realizaron gradualmente cumpliendo con las especificaciones de la metodología SCRUM, se documentó las tareas de ingeniería y pruebas de aceptación al finalizar cada sprint. No se realizaron las historias de usuario porque se documentó los requerimientos mediante diagramas de casos de uso.

En el proyecto se ejecutaron hitos transitorios pequeños y consolidados. Cada sprint proveyó un entregable plenamente funcional del producto, de esta forma el cliente tuvo una versión del sistema.

Además con el equipo de trabajo cada semana se valoró el desarrollado del sistema y se comunicó cuáles son las dificultades presentadas, con ello se actualizó el estado de la lista de tareas del Sprint Planning. El desarrollo de los 15 Sprints se encuentra en el documento adjunto (manual técnico).

Finalmente, el cierre del proyecto es la última de las fases que componen el proceso de gestión del mismo, en esta fase se realizaron todas las pruebas finales del sistema con los usuarios finales y se hizo la liberación total del producto para el cliente. Los usuarios finales fueron los estudiantes de sexto y noveno semestre de la facultad de informática y electrónica (FIE) de la Escuela Superior Politécnica de Chimborazo (ESPOCH).

Las reuniones y entregables, dentro de la metodología empleada se establecen mediante una continua comunicación con el cliente (Product Owner, PO), permitiendo así el desarrollo de las funcionalidades del proyecto en base a las necesidades del Dr. Omar Gómez que tuvo el rol de PO, cuando se trabaja de esta manera se consigue que el PO conozca más a fondo del software y su forma de trabajar, se aclaran los posibles requerimientos mal entendidos en etapas tempranas del proyecto.

En el proyecto se trabajó en bloques temporales cortos y fijos de quince días, cada sprint proporciona un resultado completo de un caso de uso, así como también un

incremento del producto que sea potencialmente entregable, es decir que el producto esté disponible para ser utilizado por el PO.

El Dr. Omar Gómez se desenvolvió como tester, porque fue la persona que ayudó a planificar y llevar a cabo las pruebas de software y así comprobar la funcionalidad del sistema, es decir la verificación, con el fin de identificar el riesgo de sufrir errores o fallos en el software y tomar las medidas correspondientes para corregir y evitar su propagación.

La gestión realizada permitió planificar, controlar y dirigir el desarrollo del proyecto, cumpliendo con el tiempo establecido para la culminación del mismo. El progreso de un proyecto Scrum fue medido por un medio de un Burn down chart.

CAPÍTULO III

3. MARCO DE RESULTADOS Y DISCUSIÓN

En el siguiente apartado se muestran los resultados obtenidos en el transcurso del desarrollo de la herramienta web de apoyo a la ejecución de experimentos en el área de ingeniería de software (IS).

3.1. Ingeniería del producto

A continuación se muestra de manera concisa los resultados obtenidos, en la ingeniería de requisitos se realizaron 3 diagramas de contexto, en donde el primer diagrama (administrador) se especificó 8 casos de uso y 22 escenarios. En el segundo diagrama correspondiente al rol de experimentador se especificó 4 casos de uso y 8 escenarios en total. Para el diagrama de contexto de un participante se registraron 5 casos de uso y 11 escenarios.

En la arquitectura de software se consiguió diseñar la vista lógica con un total de 13 clases, se pudo identificar que una asignación depende de los programas, técnicas, usuarios y experimentos. En el sistema se implementó las tres técnicas y es así que para la técnica de lectura de código se genera dos objetos el primero permite gestionar abstracciones y el segundo las inconsistencias. En la segunda técnica caja blanca se tiene dos objetos que son los casos de prueba y después se crea el objeto para gestionar las fallas. Finalmente en la técnica caja negra se tiene tres objetos siendo el primero para la gestión de las clases de equivalencia, el segundo para los casos de prueba y el último objeto para gestionar las fallas de la técnica.

En la vista de procesos se identificó dos fases que son: la preparación del experimento y la de ejecución del experimento. En la primera fase el experimentador selecciona el diseño experimental, las técnicas, programas, y la hora de ejecución de la práctica. Cuando los participantes se inscriben en el sistema, ellos deben esperar la asignación

del tratamiento. El sistema se encarga de la aleatorización de los tratamientos. En la segunda fase los participantes deben acceder al sistema y desarrollar el tratamiento dado.

En la vista física se describe el mapeo del software en el hardware y refleja los aspectos de distribución del mismo, en el servidor se encuentra la capa de presentación que está dividida en 7 subcapas, Ajax es la primera capa que contiene el archivo php para hacer páginas más interactivas, la subcapa de css contiene las hojas de estilo, la subcapa inc es donde se aloja todos los archivos php para el manejo de la base de datos y las cabeceras de la página web. En la subcarpeta pages se encuentran los archivos php que permiten la gestión de los administradores, experimentadores, participantes, técnicas, programas y experimentos. En la subcapa js se encuentra los archivos java script utilizados en la aplicación, en la subcapa img se localiza las imágenes utilizadas en el sistema, y en la subcapa pdf se alojan los reportes.

Finalmente en la vista de desarrollo se observan los archivos php, es así que en la subcapa pages están las páginas regist, registAdministrator, registExperimenter, registParticipant, registExperiment, registProgram y registTechnique. Dichos archivos contienen el código fuente de la lógica de negocio y las líneas código fuente (KLOCs) mismas que suman 7.677 líneas. Otros archivos como las hojas de estilo, data-tables, java script tienen alrededor de 14.024 líneas de código. El sistema está compuesto por un total de 21.701 KLOCs.

3.2. Gestión del proceso de desarrollo

En la gestión del proyecto se trató la gestión de riesgos donde se detectó un total de 7 riesgos, los mismos que fueron documentados, para ello se identificaron, priorizaron y se realizaron las hojas de gestión de cada uno de los riesgo. Finalmente se obtuvo un riesgo con prioridad alta, cuatro con prioridad media y dos con la prioridad baja. Se prestó mayor atención al riesgo de un “Mal diseño en la base de datos” se realizó las medidas de prevención y corrección de aquellos riesgos de prioridad alta y media, por lo que se evitó el fracaso del proyecto ya que en el transcurso del desarrollo del proyecto sobre todo en los sprints 8 y 14 existió una mala interpretación de lo que el cliente quería sin embargo se realizaron las gestiones necesarias para que el impacto sea bajo.

En la gestión del proyecto se realizaron las actividades necesarias en cada una de las fases de la metodología SCRUM, en la primera fase se documentó la especificación de los requerimientos a partir de los 14 casos de uso, y se desarrolló el Product Backlog, es una lista ordenada de los requerimientos. En el product backlog se obtuvo un total de 5 historias técnicas y 14 casos de uso.

En la segunda fase de desarrollo se estableció la medida de cada Sprint en un rango de dos semanas y de 4 horas diarias, para determinar las tareas de un sprint se hizo uso de la técnica T-shirt o técnica de tallas de camiseta en español. Se obtuvo un total de 15 Sprints y 600 puntos, los cuales se cumplieron en su totalidad. Se ejecutó y documentó las acciones propuestas en el plan de entrega de manera organizada y respetando el tiempo establecido, se ejecutaron de manera satisfactoria cada una de las iteraciones planteadas.

Al finalizar cada sprint se ejecutó la verificación del mismo mediante las pruebas de aceptación. Dichas pruebas casi en su totalidad fueron evaluadas de manera satisfactoria, sin embargo dos ellas fueron insatisfactorias por lo que se realizaron las gestiones necesarias para evitar el retraso del proyecto. En la fase de cierre se constató el desarrollo de los 600 puntos estimados y finalmente se obtuvo el siguiente **Gráfico 16-2**. En el eje horizontal del burn down chart muestra los sprints; el eje vertical muestra la cantidad de trabajo pendiente por realizar al inicio de cada sprint, este trabajo restante se ha expresado en puntos de función. En el **Gráfico 16-2** se puede visualizar los puntos de función del proyecto con un total de 600 puntos y medida que se concluye un sprint la cantidad de estos se reducen, al culminar el proyecto los puntos de función están en 0, es decir todos los puntos se cumplieron conforme a lo planificado.

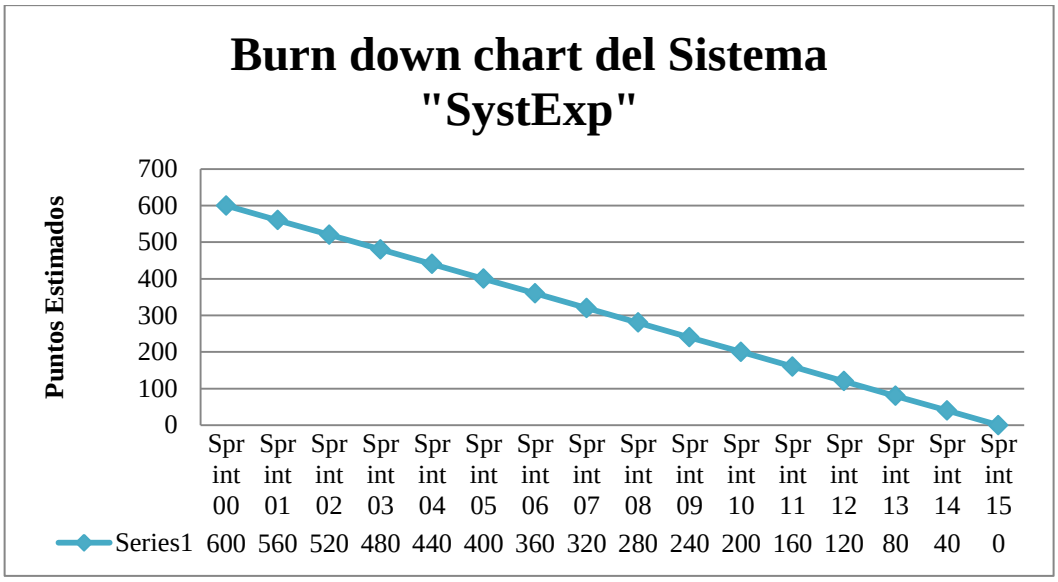

Gráfico 16-2: Burn down chart del Sistema"SystExp"
Realizado por: G. Hernández, 2018.

3.3. Validación de la propuesta

Las pruebas de validación son procesos de revisión que comprueban que el sistema cumple con las especificaciones y que logra su cometido. La validación es el proceso de comprobar que lo especificado por el usuario es realmente lo que él quería.

Para la evaluación de la usabilidad se ha hecho uso de la estadística descriptiva (recolecta, analiza y caracteriza un conjunto de datos) y la estadística inferencial que trata de extraer inferencias en poblaciones a partir del estudio de muestras, porque dichos estudios pretenden deducir (inferir) propiedades o características de una población a partir de una muestra representativa.

En la inferencia, uno de los aspectos principales de la estimación de parámetros estadísticos es la media muestral, porque es un estimador de la media poblacional. Por ejemplo: para averiguar la media de las estaturas de todos los niños de 10 años, se tomará una población aleatoria y se procederá a tomar las medidas. Finalmente se realiza el promedio de las mediciones obtenidas y se logra la media muestral.

Para comprobar la normalidad de los datos se utilizó el **test de Shapiro y Wilk,** se plantea como hipótesis nula que una muestra x1, ..., xn proviene de una población

normalmente distribuida. Fue publicado en 1965 por Samuel Shapiro y Martin Wilk. Dicho test se considera como uno de los más potentes para el contraste de normalidad de los datos, sobre todo para muestras pequeñas (n<30).

La fórmula estadística es:

$$W = \frac{\left(\sum_{i=1}^{n} a_i x_i\right)^2}{\sum_{i=1}^{n}(x_i - x)^2}$$

La hipótesis nula se rechazará si W es demasiado pequeño. El valor de W puede oscilar entre 0 y 1.

La interpretación es dada cuando la hipótesis nula que la población está distribuida normalmente, si el p-valor es menor a alfa (nivel de significancia) entonces la hipótesis nula es rechazada (se concluye que los datos no vienen de una distribución normal). Si el p-valor es mayor a alfa, no se rechaza la hipótesis y se concluye que los datos siguen una distribución normal. La normalidad se verifica confrontando dos estimadores alternativos de la varianza σ^2. (Shapiro & Wilk, 1965).

Con el fin de obtener el valor W y p del test Shapiro-Wilk se utilizó el software denominado R, ya que es un software para el análisis estadístico de datos. Además R es una implementación de software libre del lenguaje S pero con soporte de alcance estático. Se trata de uno de los lenguajes más utilizados en investigación por la comunidad estadística. (Kleiber & Zeileis, 2008).

Una vez que se comprueba la normalidad de los datos se hace uso de las pruebas o tests, son pruebas de significación estadística que cuantifican hasta qué punto la variabilidad de la muestra puede ser responsable de los resultados de un estudio en particular, en el presente estudio se ha seleccionado a la prueba t, para aquellos datos que se encuentran normalmente distribuidos y la prueba no paramétrica Wilcoxon.

Prueba t, es aquella que cuando se tiene solo dos grupos (es dicotómica), la comparación de medias entre dos grupos independientes inicia cuando se mira la magnitud de la diferencia de medias que hay entre los grupos a compararse. El proceso de aceptación o rechazo de la hipótesis lleva implícito un riesgo que se cuantifica con el

valor de la p, que es la probabilidad de aceptar la hipótesis alternativa como cierta (Visauta, 2007).

El valor p es una probabilidad que mide la evidencia en contra de la hipótesis nula. Un valor p más pequeño proporciona una evidencia más fuerte en contra de la hipótesis nula. El valor p señala si la asociación es estadísticamente significativa, es decir existe una "garantía de calidad". Este valor ha sido seleccionado y fijado en 0.05 ó 0.01. Una seguridad del 95% lleva implícitamente una p< de 0.05 y una seguridad del 99% lleva implícitamente una p< de 0.01 (Visauta, 2007).

Wilcoxón es una prueba no paramétrica para comparar el rango medio de dos muestras relacionadas y determinar si existen diferencias entre ellas. Para comprobar el supuesto se tomó los datos del pre y post test. Las diferencias iguales a cero son eliminadas y el valor absoluto de las desviaciones con respecto al valor central son ordenadas de menor a mayor. A los datos idénticos se les asigna el lugar medio en la serie. La suma de los rangos se hace por separado para los signos positivos y los negativos. S representa la menor de esas dos sumas. (Wilcoxon, 1945). Comparamos el supuesto según el nivel de significación elegido en este caso $p<0.05$

Una población estadística es un conjunto de sujetos o elementos que tienen características comunes. Sobre esta población se puede realizar un estudio estadístico con el fin de sacar conclusiones e inferir resultados. El tamaño poblacional es el número de individuos que constituyen la población. El estudio realizado se hizo con los estudiantes de sexto y noveno semestre de la FIE.

3.3.1. Instrumentos para la evaluación de la usabilidad

Un instrumento permite al investigador recopilar datos, en este caso se hizo uso de un cuestionario de investigación que consiste en una serie de preguntas e indicaciones con el propósito de obtener información de los participantes. A menudo los cuestionarios están diseñados para poder realizar un análisis estadístico a partir de las respuestas. Por ello este debe ser redactado de forma coherente, y organizado, secuenciado y estructurado de acuerdo con una determinada planificación, para que las respuestas puedan ofrecer toda la información.

Los cuestionarios se han utilizado durante mucho tiempo para evaluar las interfaces de usuario (Root & Draper, 1983), es por ello que para comprobar la usabilidad del sistema se ha realizado un cuestionario que ha sido validado por el instituto SociAl Power Intelligent agENTs (Sapient).

En el presente estudio se realizó una adaptación de dicho cuestionario: USE (Usefulness, Satisfaction, and Ease of use) (Lund, 2001), actualmente se encuentra en el dominio: http://garyperlman.com/quest/quest.cgi?form=USE. El cuestionario que se aplicó a los dos grupos se puede ver en el **Anexo A.**

El cuestionario fue diseñado para dos escenarios, dichos escenarios fueron encaminados a la aplicación manual y automatizada de la técnica de lectura de código por abstracciones sucesivas. En el primer escenario los estudiantes aplicaron la técnica y llenar el cuestionario del proceso manual para el segundo escenario debían utilizar el sistema, al finalizar la práctica debían llenar el cuestionario del proceso automatizado.

Los cuestionarios constan de 26 preguntas, las respuestas de las preguntas que aparecen están definidas en un rango de valores comprendido entre 1 y 7 donde el valor 1 equivale a la categoría "Totalmente en desacuerdo" y el valor 7 equivale a la categoría "Totalmente de acuerdo". Además está dividido en cuatro partes para medir la usabilidad del sistema las cuales son: Utilidad, uso, aprendizaje y satisfacción.

Una vez aplicado los cuestionarios a cada uno de los grupos se tabularon las preguntas y se obtuvieron los siguientes datos, aplicando la prueba-t del programa Excel. Se tomó un nivel de seguridad del 95% que conlleva implícitamente un valor de $p<0.05$.

3.4. Contexto del primer estudio

Para realizar el primer estudio se utilizó la técnica del pre-test y post-test, donde se seleccionó a los estudiantes de sexto nivel de la FIE de la ESPOCH, ellos debían aplicar la técnica de lectura de código por abstracciones sucesivas de manera manual y automatizada. La primera encuesta se aplicó al finalizar el proceso manual y la segunda al término del proceso automatizado. Se denominó a dicho grupo como Grupo A y los datos tabulados se encuentran el **Anexo C.**

3.4.1. Estadística descriptiva

En el primer estudio se contó con la presencia de los estudiantes de sexto semestre de la escuela de ingeniería en sistemas de la facultad de informática y electrónica (FIE) de la ESPOCH, son jóvenes con conocimientos de desarrollo de software. Este estudio fue realizado en dos sesiones, la primera sesión se ejecutó el 12 de junio del 2018 desde las 7 am hasta las 9 am. En la primera sesión se ejecutó el proceso manual con la presencia de 17 estudiantes.

Inicialmente los estudiantes aprendieron como debían aplicar la técnica de lectura de código por abstracciones sucesivas, después de ello para la ejecución de la práctica se les entregó el material preparado con anterioridad, dicho material se encuentra en el **Anexo B.** Para el estudio pre-test se entregó al grupo el material que consta de las instrucciones para desarrollar la técnica, un formulario para registrar las abstracciones y otro para el registro de las inconsistencias, además se preparó un programa con cuatro inconsistencias y la especificación.

En el primer paso se les entregó el código fuente impreso, las instrucciones de la tarea y el formulario F01 donde debían escribir las abstracciones a partir del código fuente, en la segunda tarea los estudiantes recibieron la especificación del programa y el formulario F02 para que registren las inconsistencias encontradas. Al finalizar la práctica se entregó a los estudiantes la encuesta del proceso manual.

La siguiente sesión fue realizada el 14 de junio del 2018 desde las 7 am hasta las 9 am, para ello los estudiantes se registraron en el sistema. Una vez que se verificó que estén registrados los participantes se procedió a la asignación del tratamiento. Se utilizó los mismos materiales del proceso manual pero en la sesión dichos materiales estaban digitalizados y están como parte del sistema. Es necesario recalcar que 3 de los 17 estudiantes no asistieron a la segunda sesión por ello no se tomó en cuenta las 3 encuestas del proceso manual para él estudio.

Después de realizar dichas actividades se procedió tabular las encuestas y se obtuvo los siguientes datos.

3.4.1.1.Resultados con respecto a la utilidad del grupo A

Para determinar la utilidad de la herramienta, se aplicó 5 preguntas encaminadas a medir la utilidad.

En la Tabla 1-3 se muestran los valores obtenidos en el estudio pre-test o proceso manual y post-test o proceso automatizado aplicado a los 14 estudiantes de sexto semestre de la Fie de la ESPOCH.

Tabla 1-3: Datos tabulados de la sub-característica utilidad

	N	Promedio	Mediana	Desviación estándar	Valor mínimo	Valor máximo
Pre-test	14	5,0286	5,2	1,4861	2,8	7
Post-test	14	6,1571	6,1	0,8121	4,4	7

Realizado por: G. Hernández, 2018.

Análisis: El promedio nos indica el valor característico de una serie de datos cuantitativos, es así que en el estudio pre-test se obtuvo un promedio de 5 por lo tanto se puede inferir que los estudiantes están parcialmente de acuerdo con el proceso manual mientras que en el post-test se obtuvo un valor de 6, dicho valor corresponde a que los estudiantes están de acuerdo con la utilidad del sistema para desarrollar la técnica.

La mediana nos muestra el número central de un grupo de números ordenados en el pre-test se puede evidenciar que el grupo está parcialmente de acuerdo con el proceso manual y que existe un nivel de aceptación en el proceso automatizado.

Con el valor de la desviación estándar podemos ver la dispersión con la que se encuentran los datos, por lo tanto los estudiantes en el pre-test tienen un porcentaje de dispersión menor al 95% mientras que el segundo estudio el grupo está dentro del rango del 68,2% de dispersión.

Se puede observar en el estudio pre-test que el valor mínimo se ubica en 2,8 dicho valor se obtuvo al calcular el promedio de las 5 preguntas implícitas para el estudio de la facilidad de uso. Este valor de 2,8 significa que el estudiante está parcialmente en desacuerdo con el proceso manual, mientras que para el proceso automatizado el valor

mínimo se ubica en 4,4 por lo que un estudiante no está ni en desacuerdo ni en acuerdo con la utilidad que puede brindar el sistema.

Finalmente se puede evidenciar que en ambos estudios existe por lo menos un estudiante que está totalmente de acuerdo con él proceso.

En el siguiente Gráfico 17-3 de barras se puede apreciar los valores obtenidos en los dos estudios, el primer estudio pre-test es de color negro mientras que el post-test tiene un color gris.

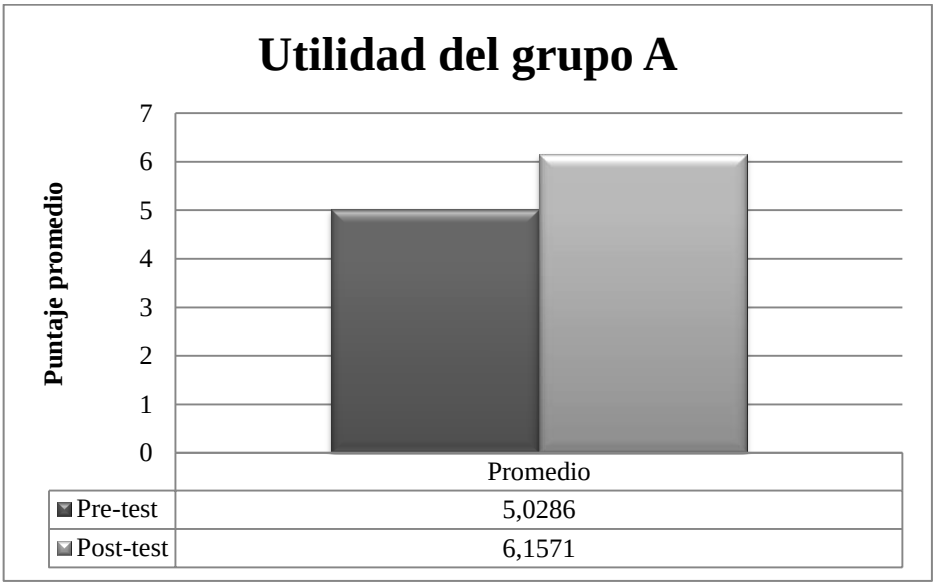

Gráfico 17-3: Gráfico de barras de la sub-característica utilidad
Realizado por: G. Hernández, 2018.

3.4.1.2. Resultados con respecto a la facilidad de uso del grupo A

Para determinar la facilidad con la que el usuario hace uso de la herramienta, con menos pasos o más naturales a su formación específica. Se aplicó 9 preguntas encaminadas a medir la facilidad de uso.

En la Tabla 2-3 se muestran los valores obtenidos en el estudio pre-test o proceso manual y post-test o proceso automatizado aplicado a los 14 estudiantes de sexto semestre de la Fie de la ESPOCH.

Tabla 2-3: Datos tabulados de la sub-característica facilidad de uso

	N	Promedio	Mediana	Desviación estándar	Valor mínimo	Valor máximo
Pre-test	14	4,4921	4,8333	1,2348	1,6667	6,3333
Post-test	14	6,0794	6	0,5363	5,2222	7

Realizado por: G. Hernández, 2018.

Análisis: En el estudio pre-test se obtuvo un promedio de 4 por lo tanto se puede inferir que los estudiantes no están ni en desacuerdo ni en acuerdo con el proceso manual mientras que en el post-test se obtuvo un valor de 6, dicho valor corresponde a que los estudiantes están de acuerdo con la facilidad de uso del sistema para desarrollar la técnica.

El número central de un grupo de números o mediana en el pre-test se puede evidenciar que el grupo no está ni de acuerdo ni en desacuerdo con el proceso manual y que existe un nivel de aceptación en el proceso automatizado.

En la desviación estándar podemos ver la dispersión con la que se encuentran los datos, en el proceso automatizado el valor no supera a uno es decir que la mayoría de los estudiantes optaron por el mismo valor.

En el estudio pre-test el valor mínimo se ubica en 1,6, dicho valor se obtuvo al calcular el promedio de las 9 preguntas implícitas para el estudio con respecto a la facilidad de uso. El valor de 1,6 significa que el estudiante está en total desacuerdo con el proceso manual, mientras que para el proceso automatizado el valor mínimo se ubicó en 5,2 por lo que un estudiante está en parcialmente de acuerdo con la facilidad de uso del sistema.

Al ver el valor máximo que se ubica en 7 para los dos estudios, se evidencia que existe por lo menos un estudiante que está totalmente de acuerdo con los dos procesos.

A continuación en el Gráfico 18-3 se puede observar los valores promedios obtenidos en los dos estudios, el primer estudio pre-test es de negro mientras que el post-test tiene un color gris.

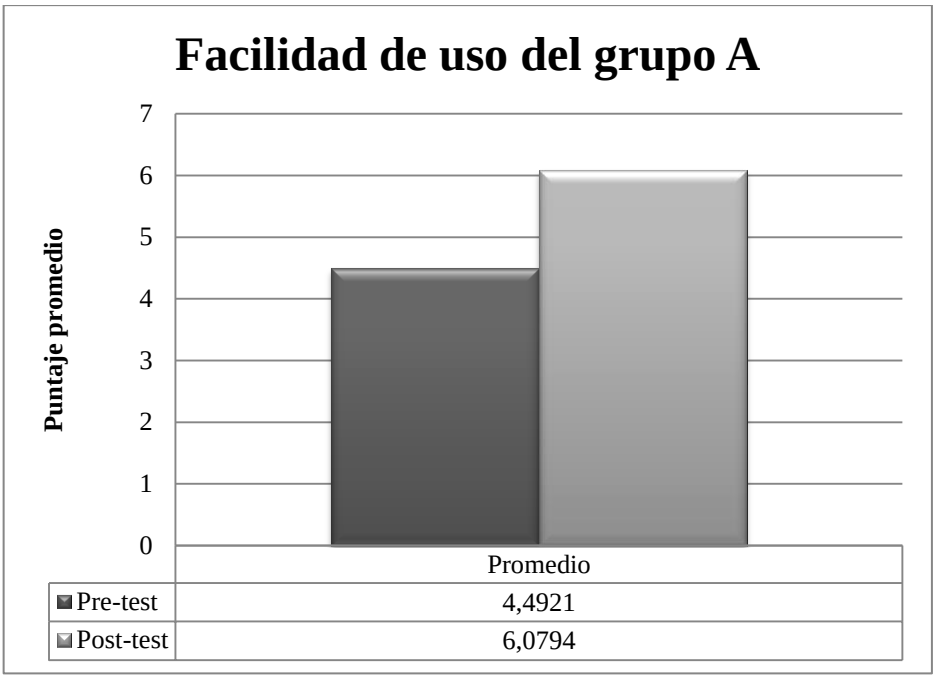

Gráfico 18-3: Gráfico de barras de la sub-característica facilidad de uso
Realizado por: G. Hernández, 2018.

3.4.1.3. Resultados con respecto a la facilidad de aprendizaje del grupo A

Para determinar la facilidad con la que nuevos usuarios desarrollan una interacción efectiva con el sistema, el nivel de predictibilidad, sintetización y familiaridad se aplicó 4 preguntas encaminadas a medir la facilidad de aprendizaje.

En la Tabla 3-3 se muestran los valores obtenidos en el estudio pre-test o proceso manual y post-test o proceso automatizado aplicado a los 14 estudiantes de sexto semestre de la Fie de la ESPOCH.

Tabla 3-3: Datos tabulados de la sub-característica facilidad de aprendizaje

	N	Promedio	Mediana	Desviación estándar	Valor mínimo	Valor máximo
Pre-test	14	5,125	5,5	1,4536	2,75	7
Post-test	14	6,125	6,125	0,7322	4,75	7

Realizado por: G. Hernández, 2018.

Análisis: El valor 5,12 corresponde al promedio del grupo del pre-test, con este valor se puede inferir que los estudiantes están parcialmente de acuerdo con el proceso manual mientras que en el post-test se obtuvo un valor de 6,12 dicho valor corresponde a que los estudiantes están de acuerdo con la facilidad de aprendizaje del sistema.

El valor medio se ubicó en 5,5 se puede evidenciar que el grupo está parcialmente de acuerdo con el proceso manual y para el post-test el valor se ubica en 6,12 es decir que el grupo está de acuerdo con el proceso automatizado.

La dispersión de los datos no es tan grande con respecto al estudio post-test, lo que significa que el grupo está de acuerdo con el proceso manual. Sin embargo en el estudio pre-test se evidencia que los estudiantes están parcialmente de acuerdo con el proceso manual.

En el estudio pre-test se puede evidenciar que el valor mínimo es de 2,75 dicho valor se obtuvo al calcular el promedio de las 4 preguntas implícitas para el estudio de la facilidad de aprendizaje, significa que por lo menos existe un estudiante que está en desacuerdo con el proceso manual, mientras que para el proceso automatizado el valor mínimo es de 4,75 por lo que un estudiante no está ni de acuerdo ni en desacuerdo con la facilidad de aprendizaje del sistema, opta por una posición neutral.

Cuando analizamos el valor máximo es evidente que en ambos estudios existe por lo menos un estudiante que está totalmente de acuerdo con los dos procesos.

Para visualizar mejor la diferencia que existe entre los dos test se realizó el Gráfico 19-3 donde se plasman los valores promedios obtenidos en los dos estudios, el primer estudio pre-test es de color negro mientras que el post-test tiene un color gris

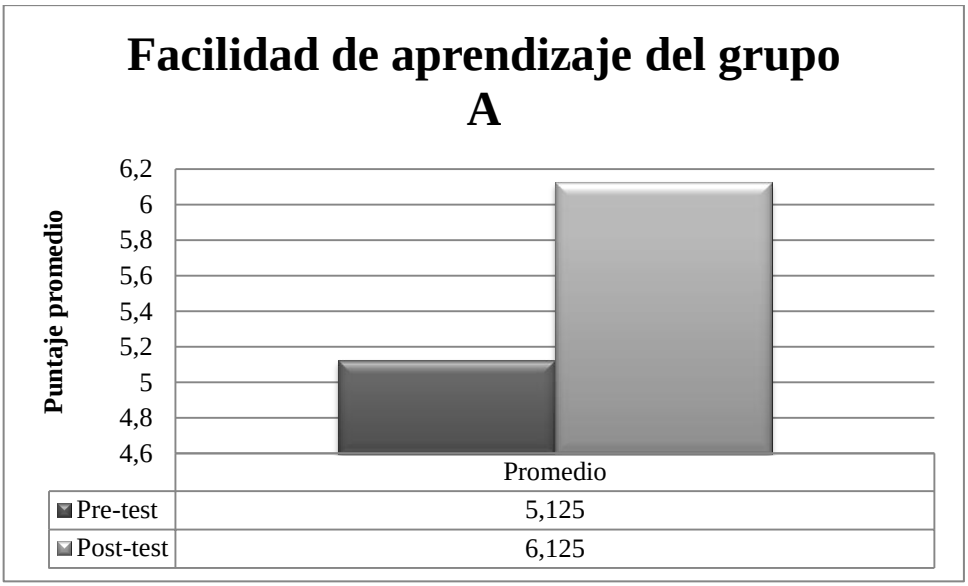

Gráfico 19-3: Gráfico de barras de la sub-característica facilidad de aprendizaje
Realizado por: G. Hernández, 2018.

3.4.1.4. Resultados con respecto al nivel de satisfacción del grupo A

Para determinar el nivel de satisfacción que perciben los usuarios al realizar el proceso manual o el proceso automatizado se aplicó 5 preguntas encaminadas a medir el nivel de satisfacción que percibieron los usuarios.

En la Tabla 4-3 se muestran los valores obtenidos en el estudio pre-test o proceso manual y post-test o proceso automatizado aplicado a los 14 estudiantes de sexto semestre de la Fie de la ESPOCH.

Tabla 4-3: Datos tabulados de la sub-característica nivel de satisfacción

	N	Promedio	Mediana	Desviación estándar	Valor mínimo	Valor máximo
Pre-test	14	5,0143	5,3	1,3999	1,6	7
Post-test	14	6,2143	6,2	0,5998	5,2	7

Realizado por: G. Hernández, 2018.

Análisis: El valor promedio para el nivel de satisfacción se ubicó en 5,01 en el estudio pre-test por lo tanto se puede inferir que los estudiantes están parcialmente de acuerdo con el proceso manual mientras que en el post-test se obtuvo un valor de 6,21 dicho valor muestra que los estudiantes están de acuerdo con el nivel de satisfacción

Se puede observar en el estudio pre-test que el valor mínimo se ubica en 1,6 dicho valor se obtuvo al calcular el promedio de las 5 preguntas implícitas para el estudio del nivel de satisfacción. Este valor de 1,6 significa que el estudiante está en desacuerdo con el nivel de satisfacción que le produce al realizar el proceso manual, mientras que para el proceso automatizado el valor mínimo se ubicó en 5,2 por lo que existe al menos un estudiante que está parcialmente satisfecho con el uso sistema.

En general los estudiantes se encuentran satisfechos con el proceso automatizado, en el Gráfico 20-3 se puede visualizar que el valor promedio del pre-test sobre pasa al valor promedio obtenido en el estudio pre-test.

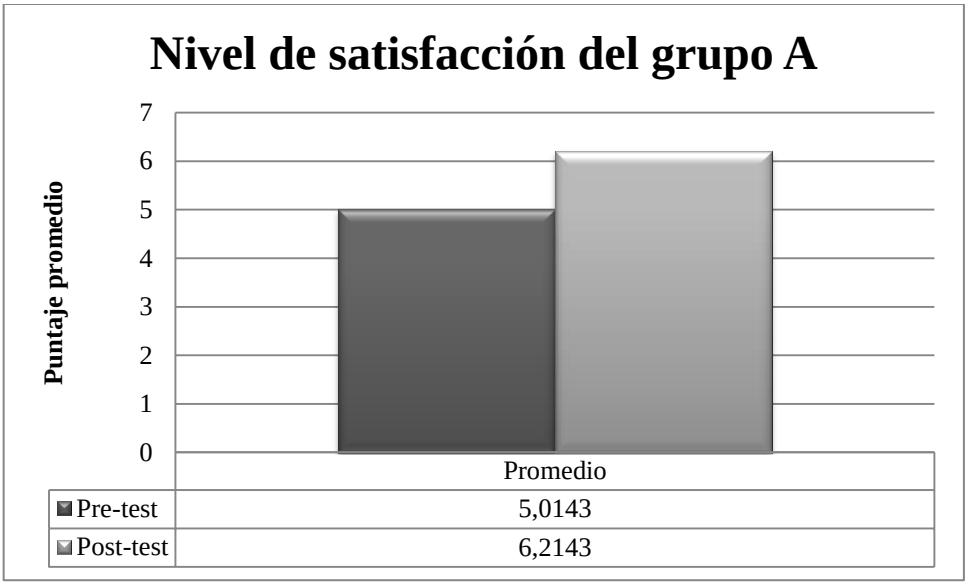

Gráfico 20-3: Gráfico de barras de la sub-característica nivel de satisfacción
Realizado por: G. Hernández, 2018.

3.4.1.5. Resultados con respecto al tiempo requerido en minutos del grupo A

Para demostrar que el sistema es útil al disminuir el tiempo requerido en la ejecución de una práctica, se calculó el tiempo empleado para resolverla. La técnica de lectura de código por abstracciones sucesivas requiere de dos tareas. En el estudio pre-test al aplicar la práctica se pidió a los estudiantes que ingresen la hora en la que iniciaban cada tarea y la hora en la que finalizaban la misma. Para calcular el tiempo requerido se realizó la diferencia entre la hora de inicio y la hora de finalización, después se sumó el tiempo empleado en cada tarea y se obtuvo el valor.

Para el estudio post-test se tomó los valores que el sistema calculó cuando los estudiantes realizaron la práctica. En la Tabla 5-3 se puede observar los valores obtenidos en minutos, del tiempo empleado en el proceso manual y automatizado.

Tabla 5-3: Datos tabulados del tiempo requerido en minutos

	N	**Promedio**	**Mediana**	**Desviación estándar**	**Valor mínimo**	**Valor máximo**
Pre-test	14	46,92	47	4,41	37	55
Post-test	14	35,66	34,79	9,83	12,6	48,6

Realizado por: G. Hernández, 2018.

Análisis: En el estudio pre-test se obtuvo un promedio de 46,92 minutos mientras que para el proceso automatizado se tuvo un promedio de 35,66 minutos por lo tanto se puede inferir que el proceso automatizado reduce en un 23,99% el tiempo requerido para la ejecución de una práctica.

Además se puede evidenciar que el grupo ocupa 47 minutos en el desarrollo del proceso manual con un valor de más o menos 4,41 minutos, sin embargo en el proceso automatizado los estudiantes ocupan un tiempo de 34,77 minutos.

En fin se puede observar que en el estudio pre-test el valor máximo es de 55 minutos mientras que en el proceso manual el valor máximo empleado es de 48,6 minutos, por lo tanto se infiere que existe una disminución del tiempo cuando se utiliza la herramienta.

En Gráfico 21-3 se puede observar el tiempo promedio del grupo A en el desarrollo del proceso manual y del proceso automatizado. En el gráfico de barras el estudio pre-test tiene un color más obscuro que el del estudio post-test.

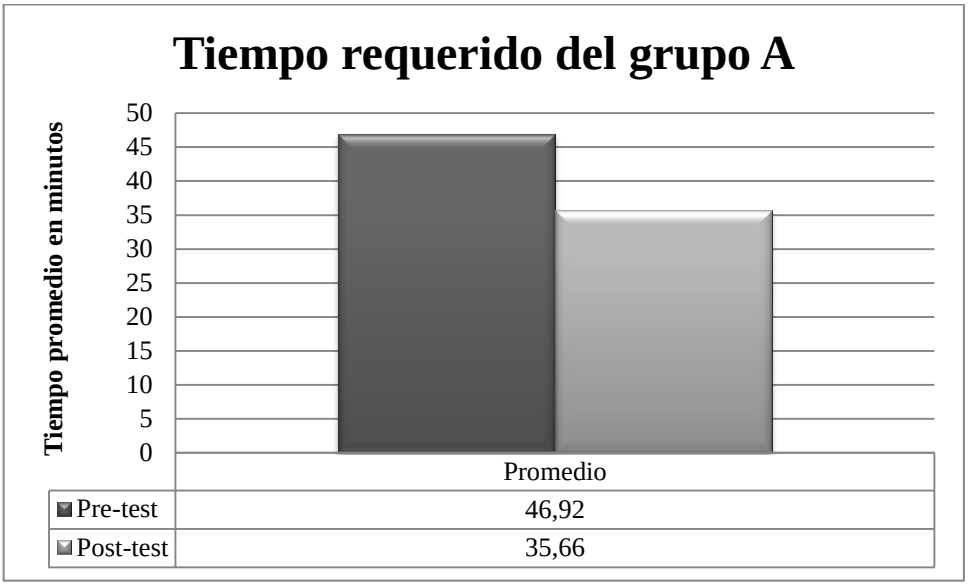

Gráfico 21-3: Gráfico de barras del tiempo requerido para desarrollar una práctica
Realizado por: G. Hernández, 2018.

3.4.2. Estadística inferencial

La estadística inferencial permite obtener inferencias a partir de la aplicación de una prueba, comprende métodos y procedimientos que a través de la inducción determina propiedades de una población estadística, a partir de una parte de esta.

3.4.2.1. Estudio del supuesto de normalidad de los datos del grupo A

Antes de aplicar cualquier tipo de prueba se debe analizar si los datos se ajustan al supuesto de normalidad para ello se aplicó el test de Shapiro Wilk, esto permitió comprobar la normalidad de los datos de cada aspecto estudiado, y se procedió aplicar la prueba Wilcoxon en aquellos aspectos que no se cumplía el supuesto de normalidad.

Los cálculos de los valores se realizaron mediante el software R. En primera instancia se descargó dicho programa, R es un software libre de análisis estadístico, en el **Gráfico**

22-3 se observa la consola de comandos de R y las líneas de código para ejecutar el test de Shapiro-Wilk.

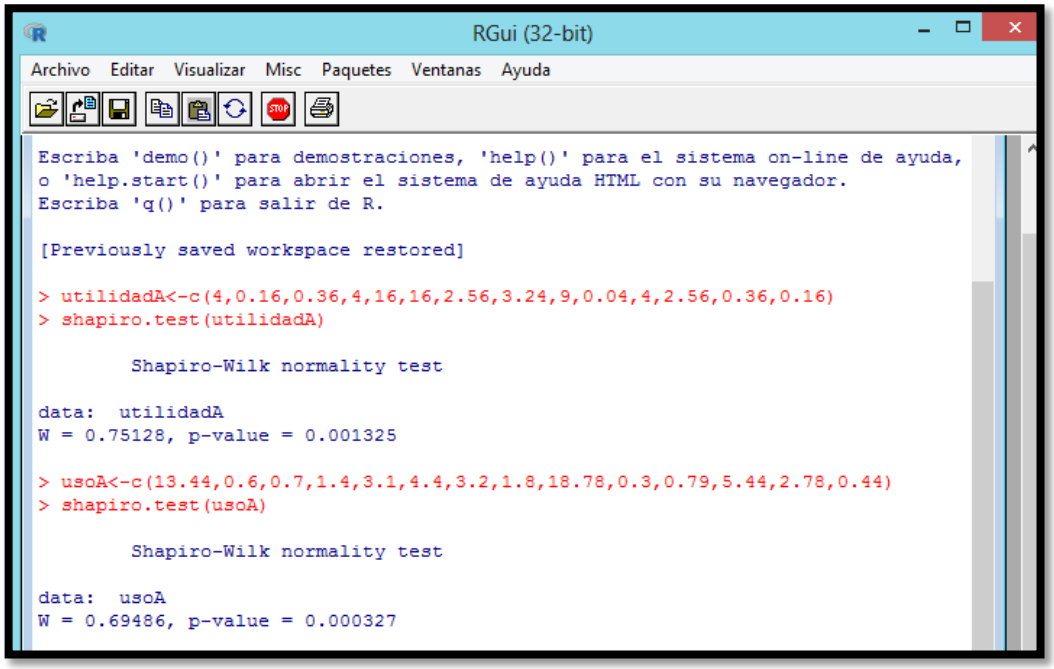

Gráfico 22-3: Consola del programa R.
Realizado por: G. Hernández, 2018.

En la siguiente tabla se encuentran los datos obtenidos a partir de los datos ingresados en el programa R.

Tabla 6-3: Valores obtenidos del test Shapiro-Wilk

Grupo A	**Utilidad**	**Uso**	**Aprendizaje**	**Nivel de Satisfacción**	**Tiempo Requerido**
W	0.75128	0.69486	0.65247	0.70391	0.74347
P<0.05	0.00132	0.00032	0.00012	0.00040	0.00108

Realizado por: G. Hernández, 2018.

Al aplicar el test de Shapiro-Wilk se evidencia que ninguno de los aspectos estudiados se ajustan al supuesto de normalidad, por ello se aplicó la prueba de Wilcoxon, es una prueba no paramétrica para comparar el rango medio de dos muestras relacionadas y

determinar si existen diferencias entre ellas. En el software R se encuentra desarrollada la prueba de Wilcoxon, por lo que basta aplicar los comandos como se muestra en el Gráfico 23-3.

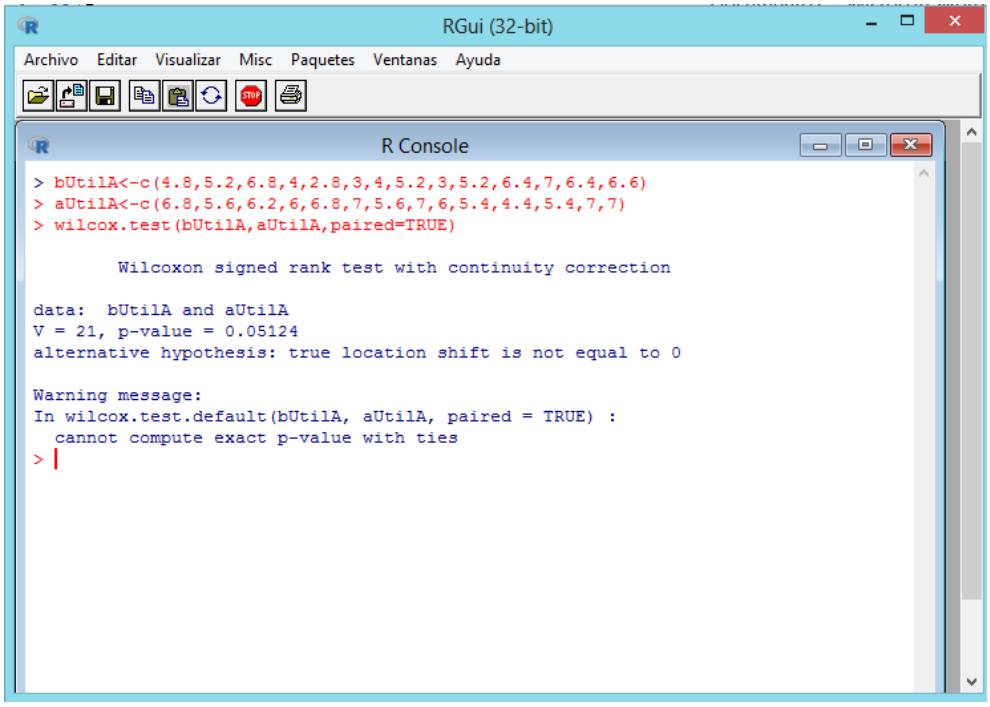

Gráfico 23-3: Consola del programa R, aplicación del test Wilcoxon

Realizado por: G. Hernández, 2018.

En la Tabla 7-3, se puede observar los valores que se obtuvieron en el software R a partir de la aplicación de la prueba de Wilcoxon para dos matrices. La primera matriz está compuesta por los valores correspondientes al proceso manual y la segunda matriz por los valores del proceso automatizado. El valor α está ubicado en $p<0,05$. Con la finalidad de demostrar que mientras más pequeño sea el valor obtenido, es más fuerte la evidencia que el proceso automatizado es más útil, fácil de usar, fácil de aprender y más satisfactorio que el proceso manual, además que el proceso automatizado es más rápido que el proceso manual, el valor p también significa que se obtuvo un mayor nivel de aceptación.

Tabla 7-3: Valores obtenidos de aplicar el test de Wilcoxon para el grupo A

Grupo A	Utilidad	Uso	Aprendizaje	Nivel de Satisfacción	Tiempo Requerido
V	21	4,5	15,5	6,5	97
P	0,05	0,002	0,03	0,001	0,005

Realizado: G. Hernández, 2018

Análisis: En general si p está por debajo 0.05 se puede decir que existen diferencias entre los dos estudios pre-test y post-test. Todos los valores obtenidos a partir de la prueba de Wilcoxon están por debajo del valor p. Con respecto a la utilidad se puede demostrar un valor de 0,05 una pequeña diferencia entre el proceso manual y el proceso automatizado, en el siguiente aspecto la facilidad de uso de obtuvo un valor de 0,002 por lo tanto hay una mayor evidencia de que el proceso automatizado es más fácil que el proceso manual.

Cuando se analiza la facilidad de aprendizaje del proceso automatizado se tiene un valor de 0,03 por lo que existe una diferencia pero no están significativa sin embargo, con ello se puede inferir que el proceso automatizado si es fácil de aprender. Mientras que en el nivel de satisfacción se obtuvo un valor de 0,001 muestra una mayor diferencia entre los dos estudios.

Conclusión: Se puede deducir que el proceso automatizado es mucho más útil, más fácil de usar, más fácil de aprender y más satisfactorio que el proceso manual. Además el valor p para el tiempo requerido de 0,001 permite conocer que el proceso automatizado es más eficiente que el proceso manual.

3.5. Contexto del segundo estudio

Para realizar el segundo estudio se utilizó la técnica del pre-test y post-test, donde se seleccionó a los estudiantes de noveno nivel de la FIE de la ESPOCH, ellos debían aplicar la técnica de lectura de código por abstracciones sucesivas de manera manual y automatizada. La primera encuesta se aplicó al finalizar el proceso manual y la segunda

al término del proceso automatizado. Se denominó a dicho grupo como Grupo B y los datos tabulados se encuentran el **Anexo 4.**

3.5.1. Estadística descriptiva

En el segundo estudio se contó con la presencia de los estudiantes de noveno semestre de la escuela de ingeniería en sistemas (EIS) de la facultad de informática y electrónica (FIE) de la ESPOCH, son jóvenes con conocimientos más bastos con respecto al desarrollo de software. Este estudio fue realizado en una sola sesión el día 18 de junio del 2018.

Inicialmente los estudiantes aprendieron como aplicar la técnica de lectura de código por abstracciones sucesivas. Para el estudio pre-test se les entregó el material preparado con anterioridad, dicho material se encuentra en el **Anexo 2.** El material consta de las instrucciones para desarrollar la técnica, un formulario para registrar las abstracciones y otro para el registro de las inconsistencias, además se preparó un programa con cuatro inconsistencias y la especificación del mismo.

En el primer paso se les entregó el código fuente impreso, las instrucciones de la tarea y el formulario F01 donde debían escribir las abstracciones a partir del código fuente, en la segunda tarea los estudiantes recibieron la especificación del programa y el formulario F02 para que registren las inconsistencias encontradas. Al finalizar la práctica se entregó a los estudiantes la encuesta del proceso manual.

En el estudio post-test del proceso automatizado los estudiantes se registraron en el sistema. Una vez que se verificó que estén registrados los participantes se procedió a la asignación del tratamiento. En el estudio participaron 10 estudiantes. Se utilizó los mismos materiales del proceso manual, pero dichos materiales se encuentran digitalizados y se encuentran en el sistema.

Después de realizar dichas actividades se procedió tabular las encuestas y se obtuvo los siguientes datos.

3.5.1.1. Resultados con respecto a la utilidad del grupo B

En el segundo grupo para determinar la utilidad de la herramienta, se aplicó 5 preguntas encaminadas a medir dicha característica.

En la Tabla 8-3 se muestran los valores obtenidos en el estudio pre-test o proceso manual y post-test o proceso automatizado aplicado a los 10 estudiantes de noveno semestre de la EIS de la Fie de la ESPOCH.

Tabla 8-3: Datos tabulados de la sub-característica utilidad del grupo B

	N	Promedio	Mediana	Desviación estándar	Valor mínimo	Valor máximo
Pre-test	10	3,82	3,82	1,83	1,4	6,2
Post-test	10	5,8	6	0,77	4,4	6,8

Realizado por: G. Hernández, 2018.

Análisis: En el grupo B el promedio del estudio pre-test es de 3 por lo tanto se puede inferir que los estudiantes están parcialmente en desacuerdo con el proceso manual mientras que en el post-test se obtuvo un valor de 5, dicho valor corresponde a que los estudiantes están parcialmente de acuerdo con la utilidad del sistema para desarrollar la técnica.

Los valores obtenidos no varían a más o menos un punto con respecto al estudio post-test, el valor mínimo se ubica en 4,4 por lo que un estudiante no está ni en desacuerdo ni en acuerdo con la utilidad que puede brindar el sistema. Y es evidente que en los dos estudios existe por lo menos un estudiante que está acuerdo tanto con el proceso manual como con él proceso automatizado.

En Gráfico 24-3 se observa un valor de 5,8 para el estudio post-test por lo tanto el grupo está parcialmente de acuerdo con el proceso automatizado.

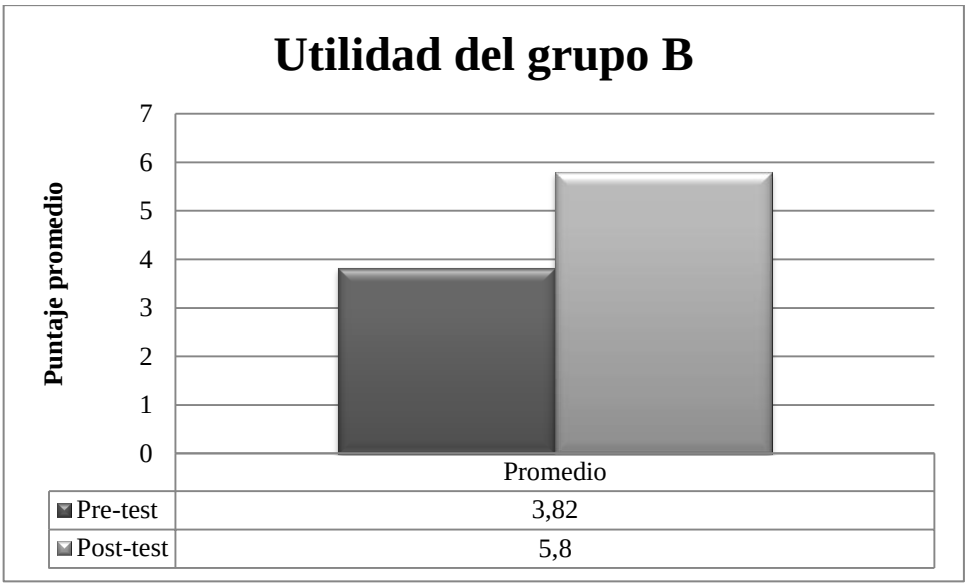

Gráfico 24-3: Gráfico de barras de la sub-característica utilidad del grupo B
Realizado por: G. Hernández, 2018.

3.5.1.2. Resultados con respecto a la facilidad de uso del grupo B

En el grupo B para determinar la facilidad con la que ellos hacen uso de la herramienta, con menos pasos o más naturales a su formación específica. Se aplicó 9 preguntas encaminadas a medir la facilidad de uso.

En la Tabla 9-3 se muestran los valores obtenidos en el estudio pre-test o proceso manual y post-test o proceso automatizado aplicado a los 10 estudiantes de noveno semestre de la Fie de la ESPOCH.

Tabla 9-3: Datos tabulados de la sub-característica facilidad de uso del grupo B

	N	Promedio	Mediana	Desviación estándar	Valor mínimo	Valor máximo
Pre-test	10	4,1667	4,1667	1,1139	2,667	6,3333
Post-test	10	5,54	5,54	0,6467	4,667	6,5556

Realizado por: G. Hernández, 2018.

Análisis: En el estudio pre-test se obtuvo un promedio de 4,16 por lo tanto se puede inferir que los estudiantes no están ni en desacuerdo ni en acuerdo con el proceso manual mientras que en el post-test se obtuvo un valor de 5,54 dicho valor corresponde

a que los estudiantes están parcialmente de acuerdo con la facilidad de uso del sistema para desarrollar la técnica.

En el Gráfico 25-3 se puede apreciar el promedio de los valores obtenidos en los dos estudios.

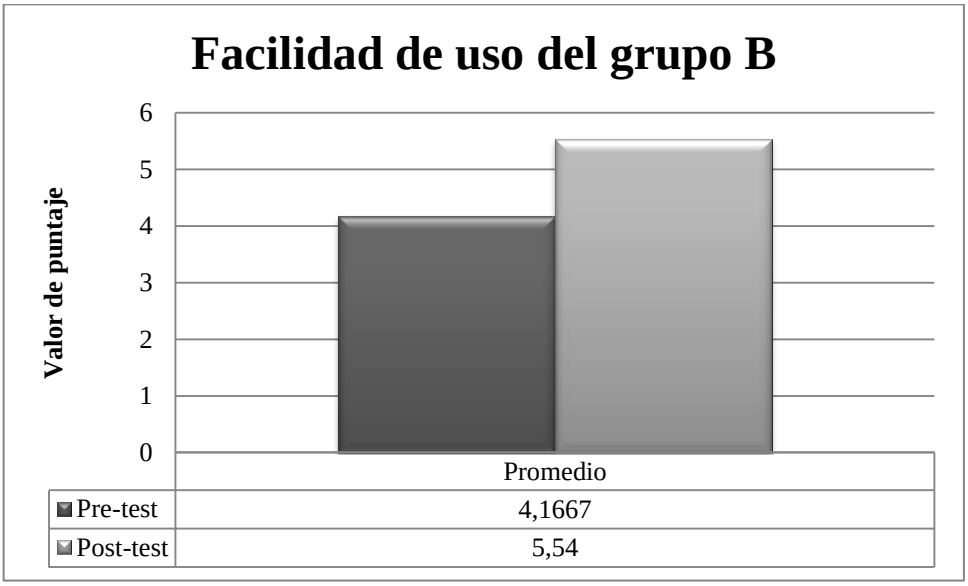

Gráfico 25-3: Gráfico de barras de la sub-característica facilidad de uso del grupo B
Realizado por: G. Hernández, 2018.

3.5.1.3. Resultados obtenidos con respecto a la facilidad de aprendizaje del grupo B

Con respecto al grupo para determinar la facilidad con la que nuevos usuarios desarrollan una interacción efectiva con el sistema, el nivel de predictibilidad, sintetización y familiaridad se aplicó 4 preguntas encaminadas a medir la facilidad de aprendizaje.

En el estudio pre-test o proceso manual y post-test o proceso automatizado aplicado a los 10 estudiantes de noveno semestre de la Fie de la ESPOCH se obtuvo los siguientes valores que se encuentran plasmados en la Tabla 9-3.

Tabla 10-3: Datos tabulados de la sub-característica facilidad de aprendizaje del grupo B

	N	Promedio	Mediana	Desviación estándar	Valor mínimo	Valor máximo
Pre-test	10	4,48	4,48	1,29	2,75	6,5
Post-test	10	5,65	5,65	0,72	4,5	6,75

Realizado por: G. Hernández, 2018.

Análisis: Para los estudiantes de noveno semestre se obtuvo un promedio de 4 por lo tanto se puede inferir que el grupo no está ni de acuerdo ni en desacuerdo con el proceso manual mientras que en el post-test se obtuvo un valor de 5 dicho valor corresponde a que los estudiantes están parcialmente de acuerdo con la facilidad de aprendizaje del sistema.

Se realizó un el Gráfico 26-3 donde se puede evidenciar que en el estudio post-test el grupo B está parcialmente de acuerdo con el proceso automatizado, dicho valor es mayor al valor promedio del estduio pre-test.

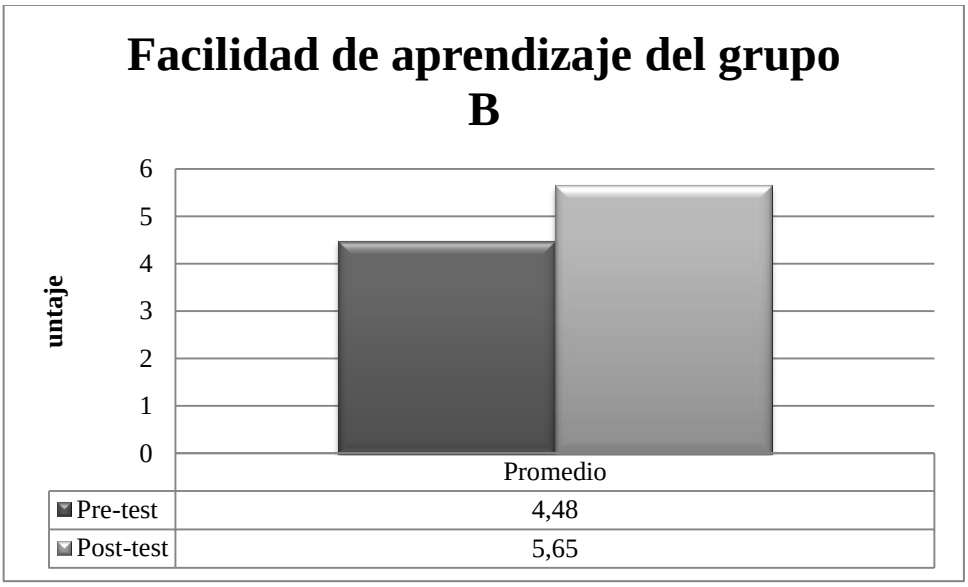

Gráfico 26-3: Gráfico de barras de la sub-característica facilidad de aprendizaje del grupo B

Realizado por: G. Hernández, 2018.

3.5.1.4. Resultado con respecto al nivel de satisfacción del grupo B

En el grupo B para determinar el nivel de satisfacción que perciben los usuarios al realizar el proceso manual o el proceso automatizado se aplicó 5 preguntas encaminadas a medir el nivel de satisfacción que percibieron los estudiantes al usar el sistema.

Los valores obtenidos en el estudio pre-test o proceso manual y post-test o proceso automatizado aplicado a los 10 estudiantes de noveno semestre de la EIS de la FIe de la ESPOCH, se encuentran en la Tabla 11-3.

Tabla 11-3: Datos tabulados de la sub-característica nivel de satisfacción del grupo B

	N	**Promedio**	**Mediana**	**Desviación estándar**	**Valor mínimo**	**Valor máximo**
Pre-test	10	4	4	1,68	1,4	6,2
Post-test	10	5,54	4,54	0,98	4,2	7

Realizado por: G. Hernández, 2018.

Análisis: El valor promedio de 4 se obtuvo para el estudio pre-test por lo tanto se puede inferir que los estudiantes no están ni de acuerdo ni en desacuerdo con el proceso manual mientras que en el post-test se obtuvo un valor de 5,54 dicho valor corresponde a que los estudiantes están parcialmente de acuerdo con el nivel de satisfacción

Además se puede evidenciar que en el pre-test por lo menos un estudiante está de acuerdo con el proceso manual, y existe un estudiante que está totalmente de acuerdo con el proceso automatizado.

Se puede evidenciar que el grupo en general se encuentra parcialmente satisfecho con respecto al uso de la herramienta, pero este valor es mayor al que se evidencia en el proceso manual. En el Gráfico 27-3 se puede observar la diferencia de los valores promedio de los dos estudios.

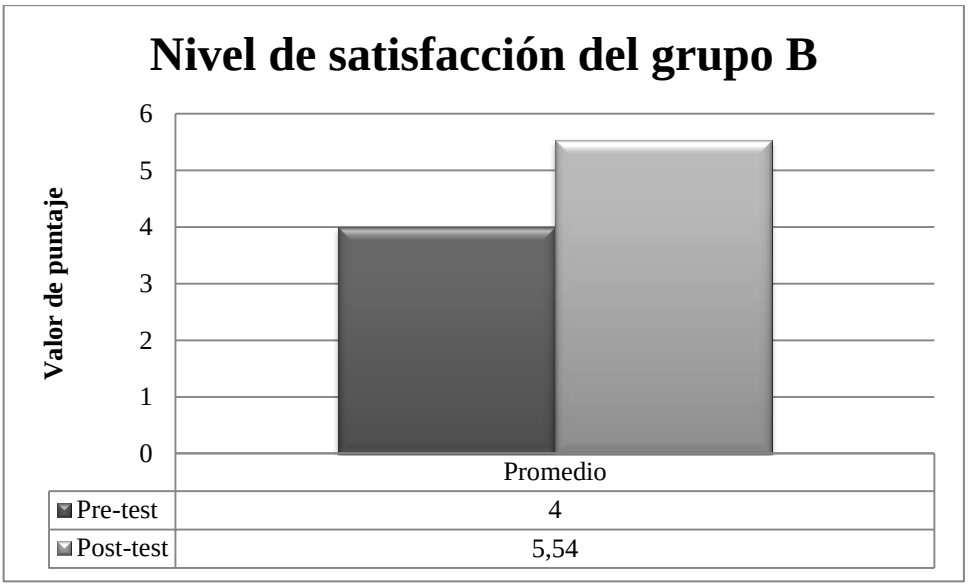

Gráfico 27-3: Gráfico de barras de la sub-característica nivel de satisfacción del grupo B

Realizado por: G. Hernández, 2018.

3.5.1.5. Resultados con respecto al tiempo requerido en minutos del grupo B

En el grupo B para verificar si el sistema es útil al disminuir el tiempo requerido en desarrollar una práctica, se calculó el tiempo empleado al resolver la práctica. La técnica de lectura de código por abstracciones sucesivas requiere de dos tareas. La suma del tiempo empleado para cada una es el tiempo utilizado para desarrollar la práctica.

En el estudio pre-test se pidió a los estudiantes que ingresen la hora de inicio de la primera tarea, al finalizarla ellos debían registrar en sus formularios la hora, al término de la tarea ellos debían nuevamente registrar la hora en la que concluyeron la segunda tarea. Existieron dos estudiantes que no registraron bien la hora, es por ello que se intuyó en esas dos métricas.

Para el estudio post-test se tomó los valores que el sistema calculó cuando los estudiantes realizaron la práctica, no existió ningún problema con respecto al registro del tiempo. En la Tabla 12-3 se puede observar los valores obtenidos en minutos del tiempo empleado en el proceso manual y automatizado.

Tabla 12-3: Datos tabulados del tiempo requerido del grupo B

	N	Promedio	Mediana	Desviación estándar	Valor mínimo	Valor máximo
Pre-test	10	21	20,75	4,55	15	31,5
Post-test	10	8,19	8,48	2,54	4,31	12,82

Realizado por: G. Hernández, 2018.

Análisis: Los datos obtenidos permiten deducir que existe una disminución en el tiempo requerido para realizar la práctica, con respecto al estudio pre-test el valor promedio se ubica en 21 minutos mientras que en el estudio post-test los estudiantes ocuparon menos de la mitad del tiempo para realizar la práctica, apenas 8,19 minutos, en este estudio el valor de la desviación estándar permite conocer que los estudiantes se demoraron entre más o menos 2,54 minutos.

Cuando se observa el valor mínimo los estudiantes se demoraron 15 minutos en el proceso manual en cambio en el proceso automatizado la ejecución de la práctica hubo un estudiante que la realizó en 4 minutos.

Mediante el Gráfico 28-3 se puede verificar que para el grupo B el tiempo promedio del proceso automatizado es menor al tiempo promedio del proceso manual.

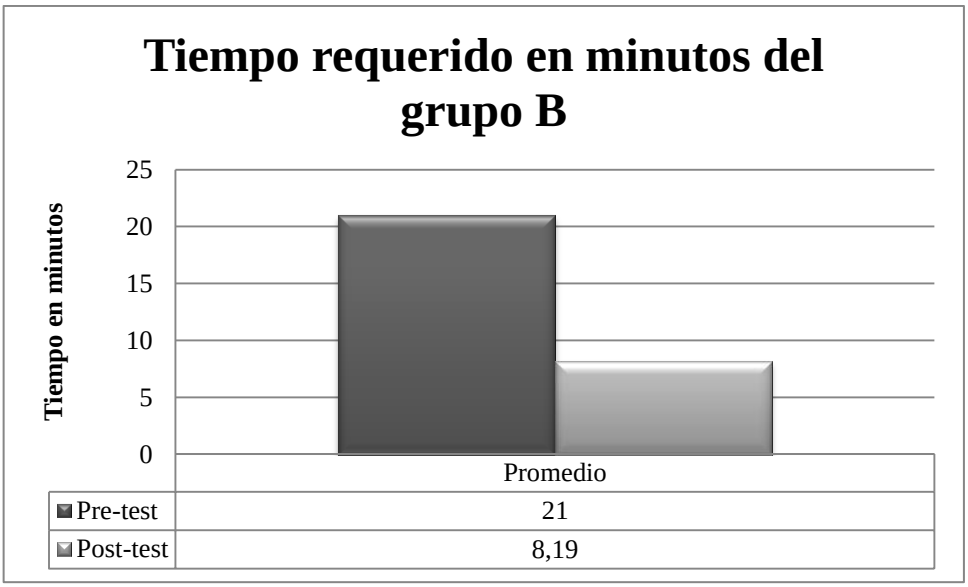

Gráfico 28-3: Gráfico de barras del tiempo requerido para desarrollar una práctica del grupo B

Realizado por: G. Hernández, 2018.

3.5.2. Estadística inferencial

Como parte de la estadística se halla la estadística inferencial que permite obtener inferencias a partir de la aplicación de una prueba, existen métodos y procedimientos que permiten deducir determinadas propiedades de una población estadística. Se utilizó el test de Shapiro–Wilk con el objetivo de contrastar la normalidad de un conjunto de datos. El valor alfa que se tomó para el test es de $p>0,05$.

3.5.2.1. Estudio del supuesto de normalidad de los datos del grupo B

El grupo B está comprendido por los estudiantes de noveno semestre de la EIS de la Fie de la ESPOCH, en primer lugar se llevó a cabo el estudio de los datos para conocer si se ajustan al supuesto de normalidad para ello se aplicó el test de Shapiro-Wilk, con lo que se comprobó la normalidad de los datos de cada aspecto estudiado, y se procedió aplicar la prueba-t en aquellos aspectos con una distribución normal y en los que no se cumplía el supuesto de normalidad se procedió aplicar la prueba de Wilcoxon.

Todos los valores se calcularon por medio de R, software libre de análisis estadístico, en la Tabla 13-3 se encuentran los datos obtenidos a partir de los datos ingresados en el programa R para el grupo B.

Tabla 13-3: Valores obtenidos del test Shapiro-Wilk del grupo B

Grupo A	**Utilidad**	**Uso**	**Aprendizaje**	**Nivel de Satisfacción**	**Tiempo Requerido**
W	0.755511	0.77383	0.86615	0.68825	0.86615
P<0.05	0.004118	0.00687	0.09013	0.00063	0.09013

Realizado por: G. Hernández, 2018.

Con el test de Shapiro-Wilk se puede observar que el aspecto de utilidad, facilidad de uso y el nivel de satisfacción no se ajustan al supuesto de normalidad, por ello se aplicó la prueba de Wilcoxon, es una prueba no paramétrica para comparar el rango medio de

dos muestras relacionadas y determinar si existen diferencias entre ellas. Para los aspectos de facilidad de aprendizaje y tiempo requerido para realizar la práctica cumplen con el supuesto de normalidad de los datos entonces se aplicó la prueba-t.

Para calcular los valores de Wilcoxon se hizo uso del software R, en la Tabla 14-3, se puede observar los valores que se obtuvieron por cada aspecto estudiado. La primera matriz está compuesta de los valores correspondientes al proceso manual y la segunda matriz de los valores del proceso automatizado. El valor p está ubicado en $p<0,05$. El objetivo fue demostrar que mientras más pequeño sea el valor obtenido, es más fuerte la evidencia que el proceso automatizado es más útil, fácil de usar y más satisfactorio que el proceso manual.

Tabla 14-3: Valores obtenidos de aplicar el test de Wilcoxon para el grupo B

Grupo A	**Utilidad**	**Uso**	**Nivel de Satisfacción**
V	4	0	8
P	0.032	0.009	0.05

Realizado: G. Hernández, 2018

Análisis: El valor de p está por debajo 0.05 y se puede decir que existen diferencias entre el pre-test y post-test para los tres aspectos estudiados. Con respecto a la utilidad se puede evidenciar un valor de 0.032 por lo cual hay una diferencia significativa a favor del proceso automatizado, en el siguiente aspecto de facilidad de uso de obtuvo un valor de 0,009 es decir existe una mayor evidencia de que el proceso automatizado es más fácil que el proceso manual. En el nivel de satisfacción se obtuvo un valor de 0,05 muestra una no muy significativa diferencia.

Para continuar con el estudio inferencial se analizó los dos aspectos para los cuales el valor p existe el supuesto de normalidad de los datos. En este estudio se utilizó la prueba-t para dos matrices. La primera matriz está compuesta por los valores del estudio pre-test y la segunda matriz tiene los valores del estudio post-test. En Tabla 15-3 se observa el valor p para la facilidad de aprendizaje y el tiempo requerido en la práctica.

El valor alfa está ubicado en p<0,05 porque mientras más pequeño es dicho valor, es más fuerte la evidencia que el proceso automatizado es más fácil de aprender y más eficiente que el proceso manual.

Tabla 15-3: Valores obtenidos de la prueba-t del grupo B

	Valor P	**Aprendizaje**	**Tiempo requerido**
Grupo B	P<0,05	0,0308	0,00044

Realizado por: G. Hernández, 2018.

Los valores obtenidos a partir del estudio de la prueba-t permiten conocer que el grupo B tiene una mayor aceptación al proceso manual con respecto a la facilidad de uso. Con respecto al tiempo requerido se puede evidenciar que el valor es de 0,0004 por lo cual se puede inferir que el proceso automatizado es más eficiente que el proceso automatizado.

Conclusión: Se puede deducir que para el segundo grupo el proceso automatizado es mucho más útil, más fácil de usar, más fácil de aprender y más satisfactorio que el proceso manual. Además el valor p para el tiempo requerido de 0,0004 permite conocer que el proceso automatizado es más eficiente que el proceso manual.

CONCLUSIONES

En el presente trabajo de titulación se han desarrollado varias actividades con el fin de cumplir los objetivos planteados en primera instancia y contestar a las sub-preguntas de investigación. Las siguientes conclusiones son argumentos y afirmaciones del trabajo realizado.

- Se investigó sobre el área de la experimentación en IS, herramientas de soporte a la experimentación en IS, la familia de pruebas de software y se determinó que los procesos que se realizan en la ejecución de un experimento son manuales por lo que existe una pérdida de tiempo y la falta de control en la recolección de métricas, mediante el desarrollo del sistema se determinó en un 95% que el uso del sistema web ayuda al desarrollo de los procesos y disminuye el tiempo de los mismos.

- Se analizó los procesos para la ejecución de experimentos en IS y se determinó dos fases las cuales son: La fase de preparación del experimento y la fase de ejecución, las mismas que fueron automatizados mediante el sistema web, para el desarrollo del sistema se aplicó la metodología ágil SCRUM, con un total de 600 puntos de función divididos en 15 Sprints, además se obtuvo 5 historias técnicas y se adaptaron las historias de usuario a diagramas de casos de uso con lo que se obtuvo 14 requerimientos documentados.

- Se investigó sobre las técnicas de pruebas de software más comunes, se determinaron las variables y las métricas que se debían recoger por cada una de ellas, es así que para la técnica de lectura de código por abstracciones sucesivas se guardan las abstracciones y las inconsistencias; para la técnica caja blanca se almacenan los casos de pruebas y los defectos; finalmente para la técnica caja negra se guardan las clases de equivalencia, los casos de prueba y los defectos encontrados, en las tres técnicas se mide el tiempo que tardan los participantes en terminar la práctica.

- Se aplicó un estudio a los estudiantes de sexto y noveno semestre de la FIE de la ESPOCH, dicho estudio se lo realizo para determinar la usabilidad del sistema determinando la facilidad con que las personas pueden utilizar la herramienta. Para ello se aplicó una encuesta para evaluar las cuatro sub-características del sistema

como son: Utilidad, uso, aprendizaje y nivel de satisfacción. Sé determinó que el uso de una herramienta ayuda a desarrollar las tareas en menor tiempo.

RECOMENDACIONES

En el presente trabajo de titulación se ha desarrollado el sistema web de apoyo a la ejecución de experimentos, con el fin de proponer mejoras o que el sistema funcione de acuerdo a lo planteado se realizan las siguientes recomendaciones.

- Se recomienda el uso de la herramienta para ejecutar experimentos en IS, además se puede investigar otras pruebas de software para ser implementadas en el sistema, se debe tener en cuenta que en el sistema están desarrolladas las técnicas de pruebas de software tales como: lectura de código con abstracciones sucesivas, caja blanca y caja negra. Y se debe tener en cuenta que el sistema ha sido desarrollado con php, MySQL y Ajax.

- Se recomienda que el administrador del sistema sea una persona experta en el área de la experimentación en ingeniería de software (IS) porque dicho rol tiene el control sobre los tratamientos (técnicas) y los objetos experimentales (programas).

- El trabajo puede ser complementado al determinar otras características de calidad que forman parte del estándar ISO/IEC 9126.

- Se recomienda utilizar el manual de usuario (documento adjunto) para que el sistema funcione de manera correcta y el manual técnico **(**documento adjunto) para visualizar el desarrollo de la metodología.

BIBLIOGRAFÍA

1. **Arisholm, E., Sjoberg, D. I., Carelius, G. J., & Lindsjorn, Y.** (2011). SESE – An Experiment Support Environment for Evaluating Software Engineering Technologies. Madrid: KompetanseWeb AS, Tullinsgate 6, NO-0166 Oslo, Norway yngve@kompetanseweb.no. [Consultado: 02-15-2017], disponible en: https://pdfs.semanticscholar.org/5a89/1448302d3d103e4399334983505356fa6b9c.pdf

2. **Basili, V.** (1996). The Role of Experimentation in Software Engineering: Past, Current, and Future. Berlin, Alemania: Proceedings of the 18th International Conference on Software Engineering.[Consultado: 01-15-2017]. Disponible en: https://ieeexplore.ieee.org/document/493439

3. **Basili, V., & Selby, R.** (1987)**.** Comparing the effectiveness of software testing strategies. *IEEE Trans. Softw. Eng., 13*(12), págs. 1278–1296. [Consultado 02-13-2017] Disponible en: https://ieeexplore.ieee.org/document/1702179

4. **Basili, V., & Selby, R.** (1987)**.** Comparing the efficiency of technical software testting. IEEE Transactions on Software Engineering.[Consultado: 01-15-2017] Disponible en: https://academic.oup.com/nar/article/30/9/e36/1089004

5. **Basili, V., Caldiera, G., & Rombach, H.** (1994). Goal question metric paradigm. *Encyclopedia of Software*, pp: 528-532. [Consultado: 02-23-2017], Disponible en: http://www.cs.umd.edu/users/basili/publications/technical/T87.pdf

6. **Basili, V., Selby, R., & Hutchens, D.** (1986). Experimentation in software engineering. IEEE Transactions on Software Engineering. [Consultado: 01-16-2017]. Disponible en: https://books.google.com.ec/

7. **Beizer, B.** (1990). *Software testing techniques (2nd ed.).* New York,USA: Van Nostrand Reinhold Co. [Consultado: 01-24-2017], disponible en: https://www.amazon.com/Software-Testing-Techniques/185032

8. **Casanova, H., Legrand, A., & Quinson, M**. (2008). Singrid: A generic framework for large-scale distributed experiments. 10 th IEEE International Conference on Computer Modeling an Simulation- EUROSIM/UKSIM. [Consultado: 03-02-2017]. Disponible en: https://ieeexplore.ieee.org/abstract/document/4488918/

9. **Del Valle Rodriguez, A. N.** (2009). Metodologías de Diseño usadas en Ingeniería Web. *Su vinculación con la NTics. (Postgrado). Universidad Nacional de la Plata.* [Consultado: 01-10-2017]. Disponible en: http://sedici.unlp.edu.ar/handle/10915/4172

10. **Farooq, & Quadri.** (2013). An externally replicated experiment to evaluate software testing methods. *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*, págs. 72-77. Disponible en: https://dl.acm.org/citation.cfm?id=2461009

11. **Fisher, R.** (1918). The correlation between relatives on the supposition of mendelian inheritance. *Transactions of the Royal Society Edinburgh 52* , págs. 399–433. Disponible en: https://www.cambridge.org/core/journals/

12. **Glenford, M.** (1979). The Art of Software Testing. Nueva York: John Wiley & Sons. Disponible en: https://www.amazon.com/Art-Software-Testing/1118031962

13. **Gómez, O., Aguilar, R., & Ucán, J.** (Nov, 2014). Efectividad de técnicas de prueba de software aplicadas por sujetos novicios de pregado. *Encuentro Nacional de Ciencias de la Computación.* Oxaca, Mexico: M.D. Rodríguez, A.I. Martínez, and J.P. Disponible en: https://pdfs.semanticscholar.org/8571/0ed477220dbbc9119cb31dce7b6ead88b174.pdf

14. **Gómez, O., Cortés, K., & Pardo, C.** (2017). *Efficiency of Software Testing Techniques:A Controlled Experiment Replication and Network Meta-analysis* (Vol. 11). e-Informatica Software Engineering Journal. Disponible en: www.e-informatyka.pl/attach/e.../eInformatica2017Art4.pdf

15. **Gómez, O., Ucán, J., & Gómez, G.** (2013). Aplicación del proceso de experimentación a la Ingeniería de Software. *Abstraction & Application Vol. 8*, págs. 26-37. Disponible en: https://intranet.matematicas.uady.mx/journal/descargar.php?id=64

16. **Hochstein, et al.** (2007). An Environment of Conducting Families of Software Engineering Experiments. University of Nebraska – Lincoln: Technical Report CS-TR-4877. Disponible en: https://www.sciencedirect.com/science/.../S0065245808006050

17. **Howden, W.** (2003). Functional testing structural testing and code reading: What fault type do they each detect?" in Empirical Method ods and Studies in Software Engineering. Págs. *2765*. Lecture Notes in Computer Science, R. Conradiand A. Wang, Eds. Berlin: IEEE Transactions on Ingeniering Software. Disponible en: https://www.researchgate.net/publication/220772316.

18. **Humphrey, W.** (1995). A discipline of Software Engineering. Addison-Wesley. [Consultado: 01-23-2018]. Disponible en: https://dl.acm.org/citation.cfm?id=526070.

19. **IEEE.** (1983). Estándar Glosario de software de Ingeniería Terminología de 1983. Disponible en: https://es.slideshare.net/luispuntoexe/ieee-610-and-ieee-1002.

20. **ISO, 9.-1.** (1998). Ergonomic requirements for office work with visual display terminals (VDTs)- Part 11: Guidance on usability. Disponible en: https://www.iso.org/standard/16883.html.

21. **Juristo, N., & Moreno, A**. (2001). Basics of software engineering experimentation . *Kluwer Academic Publishers*. Disponible en: https://www.springer.com/us/book/9780792379904.

22. **Juristo, N., & Vegas, S.** (2003). Functional testing, structural testing and code reading: What fault type do they each detect?" in Empirical Methodos and Studies

in Software Engineering. *Lecture Notes in Computer Science*, *2765*, págs. 208-232. Disponible en: https://link.springer.com/chapter/10.../978-3-540-45143-3_12

23. **Kamsties, E., & Lott, C.** (1995). An Empirical Evaluation of Three Defect-Detection Techniques. *IEEE*. Kaiserslautern, Germany: Software Technology Transfer Initiative. Disponible en: https://www.sciencedirect.com/science/.../S0950584997000281

24. **Kappel, G., Pröll, B., Reich, S., & Retschitzegger.** (2006). The Discipline of Systematic Development of Web Applications. . John Wiley & Sons. Disponible en: https://www.amazon.es/Web...Discipline-Development-Applications/dp/0470015543

25. **Kruchten, P.** (1994). Planos Arquitectónicos: El Modelo de "4+1" Vistas de la arquitectura de software. *In Proceedings of the TRI-Ada'94 C*, págs. 262–271. Disponible en: https://es.calameo.com/books/00514847377f4c1452490

26. **Linger, R., Mills, H., & Witt, B.** (1979). Structured Programming: Theory and Practice. *Addison-Wesley Publishing Company*. [Consultado: 07-13-2018] Disponible en: http://trace.tennessee.edu/cgi/viewcontent.cgi?article=1008&context=utk_harlan

27. **Loucopoulos, P., & Karakostas, V.** (1995). *System Requirements Engineering*. Inc. New York: McGraw-Hill. Disponible en: http://mcs.open.ac.uk/ban25/papers/sotar.re.pdf

28. **Lund, M.** (2001). Measuring Usability with the USE Questionnaire. *STC Usability SIG Newsletter*. [Consulta: 05-10-2018] Disponible en: http://garyperlman.com/quest/quest.cgi?form=USE.

29. **McConnell, S.** (2006). Software Estimation – Demystifying the Black Art. *WA: Microsoft Press*. [Consultado: 03-10-2017]. Disponible en: https://ptgmedia.pearsoncmg.com/images/.../9780735605350.pdf

30. **Oliveros, A**., Wehbe, R., Rojo, S. d., & Rousselot, J. (2011). *Requerimientos para aplicaciones web.* XIII Workshop de Investigadores en Ciencias de la Computación. Disponible en: http://sedici.unlp.edu.ar/bitstream/handle/10915/20125/Documento_completo.pdf%3Fsequence%3D1

31. **Potts, C.** (1993). *Software-Engineering Research Revisited.* IEEE Software. [Consultado: 01-20-2017] Disponible en: https://ieeexplore.ieee.org/abstract/document/232392/

32. **Quintero, R.** (2008). Desarrollo dirigido por modelos de aplicaciones web que integran. Sevilla-Valencia. [Consultado: 02-14-2017]. Disponible en: https://riunet.upv.es/handle/10251/1988?show=ful

33. **Reggia, J.** (1982). Computer-assisted medical decision making in Application of Computers. En M. Schwartz, *Application of Computers.* New York. Págs. 198-213. Disponible en: https://www.jstor.org/stable/29512219

34. **Richard, L., Harlan, M., & Bernard, W.** (1979). Structured Programming: Theory and Practice. Addison-Wesley Publishing Company. Disponible en: https://dl.acm.org/citation.cfm?id=578547

35. **Root, & Draper.** (1983). Questionnaires as a software evaluation tool. *Proceedings of CHI 83*. New York: NY ACM. Disponible en: https://dl.acm.org/citation.cfm?id=801586

36. **Roper, M., Wood, & Miller, J**. (1997). An empirical evaluation of defect detection techniques. *Information and Software Technology, 39*(11), págs. 763–775. Disponible en: https://www.sciencedirect.com/science/.../S0950584997000281

37. **Schwaber, K.** (2004). Agile Project Management with Scrum. *ISBN 0-7356-1993-X*, 163.

38. **Shapiro, S., & Wilk, M.** (1965). An analysis of variance test for normality (complete samples). *Biometrika*, págs. 591-611. Disponible en: https://www.jstor.org/stable/2333709

39. **Spacco, J et al**. (2005). Software repository mining with Marmoset: an automated programming project snapshot and testing system. St. Louis, Missouri: Proceedings of the 2005 International Workshop on Mining Software Repositories. Disponible en: http://marmoset.cs.umd.edu/papers/spacco-msr2005.pdf

40. **Visauta, B.** (2007). *Estadística básica (3a ed.).* Madrid: McGraw-Hilll Interamericana. Disponible en: https://www.researchgate.net/publication

41. **Weyuker, E.** (1982). On testing non-testable programs. Computer Journal 25(4) [Consultado 04-12-2018]. Disponible en: https://academic.oup.com/comjnl/article/25/4/465/366384.

42. **Wilcoxon, F.** (1945). *Individual Comparisons by Ranking Methods.* Biometrics 1.

43. **Wilks, S.** (1962). Mathematical Statistics. John Wiley Section 8.1. Disponible en: https://archive.org/details/mathematicalstat032505mbp

44. **Zelkowitz, M. V., & Wallace, D.** (1998). Experimental models for validating computer technology. IEEE Computer 31, 5. Disponible en: https://ieeexplore.ieee.org/document/675630/

ANEXOS

Anexo A: Instrumentos de la evaluación de usabilidad.

Cuestionario del proceso manual

Objetivo:

- Recolectar datos para evaluar la facilidad de uso de un proceso manual y otro automatizado referente a la aplicación de la técnica de lectura de código por abstracciones sucesivas.

Instrucciones:

A continuación se lista una serie de preguntas encaminadas a medir la facilidad de llevar a cabo un escenario (aplicación de la técnica de lectura). Las respuestas de las preguntas que aparecen a continuación están definidas en un rango de valores comprendido entre 1 y 7 donde el valor 1 equivale a la categoría "Totalmente en desacuerdo" y el valor 7 equivale a la categoría "Totalmente de acuerdo". Marque cada pregunta con una "X" según el valor con el que usted está mayormente de acuerdo.

Sección 1	**Totalmente en desacuerdo**	**En desacuerdo**	**Parcialmente desacuerdo**	**Ni de acuerdo ni en descuerdo**	**Parcialmente de acuerdo**	**De acuerdo**	**Totalmente de acuerdo**
	1	**2**	**3**	**4**	**5**	**6**	**7**
1. En general, estoy satisfecho con la facilidad de completar las tareas de este escenario							
2. En general, estoy satisfecho con la cantidad de tiempo que me tomó completar las tareas de este escenario							
3. En general, estoy satisfecho con la información de soporte (instrucciones, mensajes, documentación) al completar las tareas							
Sección 2							
UTILIDAD	**1**	**2**	**3**	**4**	**5**	**6**	**7**
4. Me ayuda a ser más efectivo en el proceso manual.							
5. Me ayuda a ser más productivo en el proceso manual.							
6. Es útil hacerlo manualmente.							
7. El proceso manual hace que las tareas que quiero alcanzar sean más fáciles de hacer.							

8. El proceso manual me ha ahorrado más tiempo.							
	Totalmente en desacuerdo	**En desacuerdo**	**Parcialmente en desacuerdo**	**Ni de acuerdo ni en descuerdo**	**Parcialmente de acuerdo**	**De acuerdo**	**Totalmente de acuerdo**
FACILIDAD DE USO	**1**	**2**	**3**	**4**	**5**	**6**	**7**
9. El proceso manual es fácil de aplicar							
10. El proceso manual es simple de usar.							
11. El proceso manual es amigable con el usuario (formularios impresos).							
12. El proceso manual requiere el menor número de pasos posibles para lograr lo que quiero hacer con él.							
13. En el proceso manual no se necesita de esfuerzo para usarlo.							
14. El proceso manual puedo usarlo sin instrucciones escritas.							
15. En el proceso manual no noto ninguna incoherencia cuando lo aplico.							
16. En el proceso manual puedo recuperarme de los errores de forma rápida y fácil (por ejemplo modificar o corregir abstracciones)							
17. El proceso manual puedo usarlo exitosamente todo el tiempo.							
FACILIDAD DE APRENDIZAJE	**1**	**2**	**3**	**4**	**5**	**6**	**7**
18. Aprendí a usar rápidamente la técnica con el proceso manual.							
19. El proceso manual es fácil de recordar.							
20. El proceso manual es fácil de aprender a usarlo.							
21. Me volví rápidamente hábil con el uso del proceso manual							
SATISFACCIÓN	**1**	**2**	**3**	**4**	**5**	**6**	**7**
22. Estoy satisfecho con la forma de aplicar la técnica.							
23. El proceso manual se lo recomendaría a un amigo.							
24. El proceso manual funciona de la manera que yo quiero que funcione.							
25. El proceso manual es maravilloso.							
26. El proceso manual es agradable de usar.							

Comentarioadicional:……

……

Cuestionario del proceso automatizado

Objetivo:

- Recolectar datos para evaluar la facilidad de uso de un proceso manual y otro automatizado referente a la aplicación de la técnica de lectura de código por abstracciones sucesivas.

Instrucciones:

A continuación se lista una serie de preguntas encaminadas a medir la facilidad de llevar a cabo un escenario (aplicación de la técnica de lectura). Las respuestas de las preguntas que aparecen a continuación están definidas en un rango de valores comprendido entre 1 y 7 donde el valor 1 equivale a la categoría "Totalmente en desacuerdo" y el valor 7 equivale a la categoría "Totalmente de acuerdo". Marque cada pregunta con una "X" según el valor con el que usted está mayormente de acuerdo.

Sección 1	**Totalmente en desacuerdo**	**Parcialmente en desacuerdo**	**Parcialmente desacuerdo**	**Ni de acuerdo ni en**	**Parcialmente de acuerdo**	**Parcialmente de acuerdo**	**Totalmente de acuerdo**
	1	**2**	**3**	**4**	**5**	**6**	**7**
1. En general, estoy satisfecho con la facilidad de completar las tareas de este escenario							
2. En general, estoy satisfecho con la cantidad de tiempo que me tomó completar las tareas de este escenario							
3. En general, estoy satisfecho con la información de soporte (instrucciones, mensajes, documentación) al completar las tareas							
Sección 2							
UTILIDAD	**1**	**2**	**3**	**4**	**5**	**6**	**7**
4. Me ayuda a ser más efectivo en el proceso automatizado.							
5. Me ayuda a ser más productivo en el proceso automatizado.							
6. Es útil hacerlo automáticamente							
7. El proceso automatizado hace que las tareas que quiero alcanzar sean más fáciles de hacer.							
8. El proceso automatizado me ha ahorrado más tiempo.							

	Totalmente en desacuerdo	Parcialmente en desacuerdo	Parcialmente desacuerdo	Ni de acuerdo ni en	Parcialmente de acuerdo	Parcialmente de acuerdo	Totalmente de acuerdo
FACILIDAD DE USO	**1**	**2**	**3**	**4**	**5**	**6**	**7**
9. El proceso automatizado es fácil de aplicar							
10. El proceso automatizado es simple de usar.							
11. El proceso automatizado es amigable con el usuario (sistema).							
12. El proceso automatizado requiere el menor número de pasos posibles para lograr lo que quiero hacer.							
13. En el proceso automatizado no se necesita de esfuerzo para usarlo.							
14. El proceso automatizado puedo usarlo sin instrucciones escritas.							
15. En el proceso automatizado no noto ninguna incoherencia cuando lo aplico.							
16. En el proceso automatizado puedo recuperarme de los errores de forma rápida y fácil (por ejemplo modificar o corregir abstracciones)							
17. El proceso automatizado puedo usarlo exitosamente todo el tiempo.							
FACILIDAD DE APRENDIZAJE	**1**	**2**	**3**	**4**	**5**	**6**	**7**
18. Aprendí a usar rápidamente la técnica con el proceso automatizado.							
19. El proceso automatizado es fácil de recordar.							
20. El proceso automatizado es fácil de aprender a usarlo.							
21. Me volví rápidamente hábil con el uso del proceso automatizado							
SATISFACCIÓN	**1**	**2**	**3**	**4**	**5**	**6**	**7**
22. Estoy satisfecho con la forma de aplicar la técnica.							
23. El proceso automatizado se lo recomendaría a un amigo.							
24. El proceso automatizado funciona de la manera que yo quiero que funcione.							
25. El proceso automatizado es maravilloso.							
26. El proceso automatizado es agradable de usar.							

Comentario:……

……

…….

Anexo B: Material para el desarrolla de la técnica

Instrucciones para aplicar la técnica "Revisión de Código"

Lectura del Código

1. Escribe tu nombre, fecha y hora de inicio en el formulario de abstracciones (F01).
2. Lee por encima el código para tener una idea general del componente. Si te parece descubrir faltas mientras haces este paso, puedes marcarlas. No obstante, no pierdas demasiado tiempo en hacer un análisis preciso de ellas.
3. Determina las dependencias entre las funciones (o métodos) individuales del código de fuente, usando para ello un árbol de llamadas. Normalmente las funciones ya estarán ordenadas en el código, de tal modo que las funciones de bajo nivel (hojas) están al comienzo y las funciones de más alto nivel (raiz) aparecerán al final (método main o método principal). Comienza aplicando la técnica de revisión de código con las funciones hoja y sigue hacia la raíz.
4. Intenta entender la estructura de cada función individual identificando las estructuras elementales (secuencias condicionales, bucles) y marcándolas. Combina las estructuras elementales para formar estructuras más grandes hasta que hayas entendido la función entera.
5. Generación de abstracción. Trata de determinar el significado de cada estructura comenzando con la más interna y anota el resultado en el formulario para tal propósito (F01). Usa rango de números de línea (línea x-y) cuando hagas eso. Evita usar conocimiento implícito que no resida en la estructura, por ejemplo valores iniciales, entrada o valores de parámetros. Describe las partes lo más funcionalmente posible. Mientras hagas eso, usa principios generalmente aceptados del dominio de aplicación para mantener la descripción breve y entendible. Por ejemplo, en el caso de búsqueda en un árbol, menciona "búsqueda en profundidad" en lugar de describir lo que hace la búsqueda en profundidad.
6. Cuando hayas completado las abstracciones, incluye tu versión de la especificación del programa, registra la duración final de esta actividad e indica al instructor la finalización de esta actividad para continuar con los siguientes pasos.
7. Recoge el formulario para inconsistencias (F02), registra tus datos así como la fecha y la hora de inicio de esta actividad. Este formulario viene acompañado por la especificación del programa.

Búsqueda de inconsistencias

8. Lee con detenimiento la especificación que se te ha entregado. Dicha especificación describe el componente (programa) como un todo, no las funciones individuales. Detecta las posibles inconsistencias comparando la especificación con tu descripción del componente (abstracciones redactadas en lengua natural). Escribe las inconsistencias que hayas detectado en el formulario F02.

   Para ello, puedes enumerar las inconsistencias que hayas encontrado desde 1 hasta n del formulario para inconsistencias F02.Registra ls inconsistencias encontradas.
9. Una vez creas que has detectado todas las inconsistencias, registra la duración de esta actividad y entrega todo el material al instructor a cargo del ejercicio. Ya has terminado.

Formulario de trabajo para la técnica de revisión de código

Formulario F01

Abstracciones			
Id de usuario:	**Fecha:**	**Hora inicial:**	**Hora final:**
Líneas de código	**Descripción de las abstracciones**		

Formulario de trabajo para la técnica de revisión de código

Formulario F02

Inconsistencias detectadas			
Id de usuario:	**Fecha:**	**Hora inicial:**	**Hora final:**
Resultado esperado de acuerdo a la especificación	**Líneas de código de la abstracción**	**Escribe una breve descripción de la inconsistencia**	

Código fuente preparado para ambos estudios

```
1   package edad2;
2
3   /**
4    *
5    * @author Jimena Hernández
6    * @date 08/06/2018
7    */
8
9   import java.time.LocalDate;
10  import java.time.Period;
11  import java.time.format.DateTimeFormatter;
12  import java.util.ArrayList;
13  import java.util.Scanner;
14
15  class Persona {
16
17      private String nombre;
18      private LocalDate fecha;
19
20      public Persona() {
21      }
22
23      public Persona(String nombre, LocalDate fecha) {
24          this.nombre = nombre;
25          this.fecha = fecha;
26      }
27
28      public String getNombre() {
29          return nombre;
30      }
31
32      public void setNombre(String nombre) {
33          this.nombre = nombre;
34      }
35
36      public LocalDate getFecha() {
37          return fecha;
38      }
39
40      public void setFecha(LocalDate fecha) {
41          this.fecha = fecha;
42      }
43
44  }
45
46  public class Edad2 {
47
48      /**
49       * @param args the command line arguments
50       */
51      public static void calcular(LocalDate fechaNac) {
52          LocalDate ahora = LocalDate.now();
53          Period periodo = Period.between(fechaNac, ahora);
54          System.out.printf(" tiene: %s años, %s meses y %s días",
55                  periodo.getYears(), periodo.getDays(), periodo.getMonths());
56      }
57
58      public static void main(String[] args) {
```

```
59          ArrayList<Persona> personas = new ArrayList<Persona>();
60          Scanner sn = new Scanner(System.in);
61          boolean salir = false;
62          int opcion;
63          String nombre;
64          String fecha;
65          System.out.println("         Bienvenido");
66          while (!salir) {
67             System.out.println("         Menú");
68             System.out.println("1. Agregar una persona");
69             System.out.println("2. Imprimir listado de personas");
70             System.out.println("3. Salir");
71             System.out.println("Escriba una de las opciones");
72             opcion = sn.nextInt();
73             switch (opcion) {
74                case 1:
75                   System.out.print("Nombre:");
76                   nombre = sn.next();
77                   System.out.println("Utilice el siguiente "
78                         + "formato dd/mm/yyyy");
79                   System.out.print("Fecha de nacimiento:");
80                   fecha = sn.next();
81                   try {
82                      DateTimeFormatter fmt =
83                                        DateTimeFormatter.ofPattern("dd/MM/yyyy");
84                      LocalDate fechaNac = LocalDate.parse(fecha, fmt);
85                      personas.add(new Persona(nombre, fechaNac));
86                      System.out.println("Datos ingresados correctamente");
87                   } catch (Exception e) {
88                      System.out.println("La fecha es incorrecta");
89                   }
90                   break;
91
92                case 2:
93                   System.out.println("Listado de personas");
94                   for (Persona p : personas) {
95                      System.out.print(p.getNombre() + "  ");
96                      calcular(p.getFecha());
97                   }
98                   break;
99
100               case 3:
101                  salir = true;
102                  break;
103            }
104         }
105      }
106   }
107
108
109
110
111
112
```

Anexo C: Datos tabulados del primer estudio

La encuesta del proceso manual y automatizado fue aplicada a los estudiantes de sexto semestre de la FIE de la ESPOCH. Para los cálculos respectivos se tomó en cuenta el promedio del bloque de cada sub-característica, es así que para la utilidad se aplicaron 5 preguntas, en la facilidad de uso 9 preguntas, para la facilidad de aprendizaje se ocuparon 4 preguntas y para el nivel de satisfacción 5 preguntas.

En la siguiente tabla se muestran los datos tabulados de cada sub-característica como son: la utilidad, la facilidad de uso, la facilidad de aprendizaje, y el nivel de satisfacción.

Tabla: Valores tabulados del grupo A.

	UTILIDAD		**USO**		**APRENDIZAJE**		**NIVEL DE SATISFACCIÓN**		**TIEMPO REQUERIDO**	
Participantes del grupo A	**Pre-test**	**Pos-test**	**Pre-test**	**Pos-test**	**Pre-test**	**Pos-test**	**Pre-test**	**Pos-test**	**Pre-test**	**Pos-test**
Dario Janeta	4,8	6,8	3	6,667	2,75	6,25	3,6	6,2	47	48,0
José Ponce	5,2	5,6	5	5,777	5,75	6,25	5	6	50	44,2
Álvaro López	6,8	6,2	6,111	5,222	6,75	5,25	6,8	5,6	46	30,3
Jorge Jiménez	4	6	4,777	6	5,5	5,75	5,6	6,2	44	35,1
Bryon Sagnay	2,8	6,8	3,778	5,556	5,75	6,5	4	6,6	49	27,8
Rodrigo Lazo	3	7	4,222	6,333	2,75	7	3,8	6,8	53	40,7
Jonathan Valverde	4	5,6	3,778	5,556	4,75	5,25	5,4	5,4	55	32,1
Mirian Tipán	5,2	7	5	6,333	4,75	7	5,6	6,8	49	48,0
Israel Oña	3	6	1,666	6	2,75	4,75	1,6	5,8	47	48,0
Maria Jose Quinatoa	5,2	5,4	5,222	5,778	5,25	5,75	5,2	5,8	47	30,1
Viviana Tixi Gallegos	6,4	4,4	4,888	5,7778	6,75	6	5	5,2	44	34,4
Claudio Pilataxi	7	5,4	3,777	6,1111	5,5	6	6	6,6	37	37,1
Alex Moreno	6,4	7	5,333	7	7	7	7	7	46	12,6
Celio Tasna	6,6	7	6,333	7	5,75	7	5,6	7	43	30,4

Realizado por: G. Hernández, 2018.

Anexo D: Datos tabulados del grupo B

La encuesta del proceso manual y automatizado fue aplicada a los estudiantes de noveno semestre de la FIE de la ESPOCH. Para los cálculos respectivos se tomó en cuenta el promedio del bloque de cada sub-característica, es así que para la utilidad se aplicaron 5 preguntas, en la facilidad de uso 9 preguntas, para la facilidad de aprendizaje se ocuparon 4 preguntas y para el nivel de satisfacción 5 preguntas.

En la siguiente tabla se muestran los datos tabulados de cada sub-característica como son: la utilidad, la facilidad de uso, la facilidad de aprendizaje, y el nivel de satisfacción.

Tabla: Valores tabulados del grupo B

	UTILIDAD		**USO**		**APRENDIZAJE**		**NIVEL DE SATISFACCIÓN**		**TIEMPO REQUERIDO**	
Participantes del grupo B	**Pre-test**	**Post-test**	**Pre-test**	**Post-test**	**Pre-test**	**Post-test**	**Pre-test**	**Post-test**	**Pre-test**	**Post-test**
Dany Olmedo	6	6	6	6,3333	6,5	6,25	6,2	6	15	8,155
Clara Freire	1,4	6,2	3,1111	5,5556	2,75	6	1,6	5,6	20,5	6,25
Edwin Hernández	4,8	4,4	4,7778	4,8889	4,5	5,75	5	4,2	19	5,3
Anthony Pazmiño	1,8	6,8	2,8889	6,5556	4,75	6,75	1,4	7	20,5	10,265
Eduardo Hipo	1,4	6,4	2,6667	5,8889	4	6	2,8	6	22,5	8,795
Jasón Gómez	3,2	6,2	3,8889	5,6667	2,75	5,25	3,8	4,8	15,5	4,31
Cristian Gutiérrez	4,4	6	4	4,6667	3,75	4,5	3,6	4,4	21	9,68
Piedad Yumisaca	5,4	6	4,4444	4,6667	5,75	4,5	5,2	7	22,5	9,355
Adriana Quille	6,2	5,4	5,6667	5,6667	6	5,5	6	5	22	6,975
Grecia Torres	3,6	4,6	4,2222	5,5556	4	6	4,4	5,4	31,5	12,815

Realizado por: G. Hernández, 2018.