Procedimiento para pruebas de software con herramientas automatizadas en el Departamento de Pruebas de Software.

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

Autor(es): Ariannis Marzán Matos.
Juan Carlos Villar Bravo.

Tutor(es): Ing. Roig Calzadilla Díaz.
Ing. Delvis Echeverría Pérez.
Ing. Delmys Pozo Zulueta.

Ciudad de la Habana, Junio del 2010

DECLARACIÓN DE AUTORÍA

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

Para que así conste firmo la presente a los días___ del mes de_____ del año 2010.

Ariannis Marzán Matos

_____

Firma del Autor

Ing. Roig Calzadilla Díaz

_____

Firma del Tutor

Juan Carlos Villar Bravo

_____

Firma del Autor

Ing. Delvis Echeverría Pérez

_____

Firma del Tutor

Ing. Delmys Pozo Zulueta

_____

Firma del Tutor

Les dedico este trabajo a aquellas personas que han sido mi fuerza e inspiración:

A Dios por ser mi fortaleza y mi pronto auxilio, por guiarme y levantar estandartes en mi vida, y sobre todo por darme la victoria, porque todo se lo debo a Él.

A mis padres por ser un precioso regalo de Dios, por estar siempre conmigo, por apoyarme en los momentos difíciles, por ser los mejores padres del mundo.

A mi hermanita del alma por ser parte de mí.

A mis mejores amigos por ser tan pacientes conmigo, apoyarme y siempre estar presentes en los momentos de alegría y tristeza.

Ariannis.

Este trabajo lo dedico a mi hijo Carlos Daniel, quien se puede decir que no ha comenzado a vivir y ya me ha dado vida y alegría.

Juan Carlos.

Resumen

La realización de las pruebas de software constituye un factor esencial para garantizar la calidad del producto, implicando que se debe de cumplir con todas las exigencias y perspectivas del cliente. La siguiente investigación se centra en crear un procedimiento que permita realizar pruebas con herramientas automatizadas en el Departamento de Pruebas de Software (D.P.S.). De manera que al automatizar dichas pruebas el proceso sea más rápido y flexible, sin pérdida de tiempo, detectando todas las No Conformidades de forma correcta. Se hizo imprescindible la investigación del uso de herramientas para mejorar la calidad y eficiencia del flujo de prueba. Durante la investigación se dan a conocer algunos tipos de pruebas, así como su funcionamiento, mediante ejemplos reales, de pruebas realizadas a una determinada aplicación web de la Universidad de las Ciencias Informáticas.

Palabras claves: pruebas de software; procedimiento; herramientas

Introducción

El desarrollo de software es uno de los pilares fundamentales de la informática que ha tomado un gran auge a nivel mundial, demandando la producción de mejores productos en menos tiempo, aplicando el uso de las técnicas y metodologías que permiten una mayor productividad. No siempre se obtienen buenos resultados en la producción de software, pues en el transcurso del desarrollo surgen algunos inconvenientes como: grandes retrasos en la programación, inconsistencia en su funcionamiento y principalmente la falta de calidad.

Los desarrolladores de software hoy en día piensan que la calidad sólo es aplicable al producto sin pensar en muchos factores previos a esta fase. La calidad de un producto parte desde que se define la idea inicial del mismo hasta su entrega al usuario. La entrega de un producto con calidad considera actividades como:

- Administración de la calidad, asegurando minimizar las diferencias entre los recursos presupuestados y los recursos realmente utilizados en las distintas etapas.
- Uso de tecnología de Ingeniería de Software eficiente, considerando métodos de desarrollo y herramientas.
- Aplicación de técnicas formales a lo largo de todo el proceso.
- Minimización de las variaciones entre los productos, disminuyendo las diferencias y defectos entre versiones.
- Testeo acucioso en diferentes etapas del desarrollo.
- Control de la documentación, tanto de apoyo al desarrollo como la entregada al usuario final generado en cada etapa, y verificación de los posibles cambios y modificaciones que pudiera sufrir.
- Correcta mantención y servicios de post-venta.

Cuando es necesario elegir un producto, se opta siempre por aquel que sea flexible, eficiente, que brinde más servicios y que tenga mayor calidad, influyendo esto positivamente en la decisión de un cliente a la hora de escoger el producto que necesita. La obtención de un software con calidad implica la utilización de metodologías o procedimientos estándares para el análisis, diseño, programación y prueba de software que permitan uniformar la filosofía de trabajo, en aras de lograr una mayor confiabilidad y facilidad de prueba, a la vez que eleven la productividad, tanto para la labor del desarrollo como para el control de la calidad de software. (ISO 900-200). [1]

Las pruebas de software son unas de las actividades fundamentales para la garantía de la calidad, estas identifican posibles fallos que puedan tener los productos, representando una revisión parcial o final de las especificaciones, del diseño, y de la codificación de los elementos, que juntos componen la aplicación computacional.

En la actualidad el uso de las herramientas se ha hecho imprescindible, para automatizar muchos procesos. El flujo de prueba específicamente es muy importante en el desarrollo de un software, en el cual se hace importante el uso de herramientas para mejorar la calidad y eficiencia de dicho flujo, proporcionando un análisis detallado de la calidad y rendimiento de las aplicaciones propiciando a los desarrolladores un experto automatizado capaz de advertirles, localizar y corregir sus problemas.

El gobierno cubano se ha propuesto desarrollar la Industria de Software, no solamente por los beneficios que se aportan al desarrollar sistemas informáticos para uso interno, sino también con el fin de aunar los esfuerzos individuales que han venido realizando diversas instituciones del país en este campo para alcanzar una fortaleza que permita incursionar, con más efectividad, en los mercados extranjeros. La Industria Cubana del Software tiene la ardua tarea de lograr que los productos desarrollados en el país cumplan con las normas y estándares internacionales de calidad, tarea en la que se encuentran enfrascadas todas las instituciones vinculadas al desarrollo de software.

La Universidad de la Ciencias Informáticas (UCI) no solo forma futuros profesionales sino también se dedica al desarrollo de software y servicios informáticos, y se ve obligada a crear un Centro de Calidad para Soluciones Tecnológicas (CALISOFT) dedicado al diseño y aplicación de pruebas de software para verificar la calidad de estos productos desarrollados; CALISOFT está estructurado en 4 áreas fundamentales: Revisiones y Auditoría, Normalización, Métrica, y el Departamento de Pruebas de Software; este último tiene como misión principal lograr que todo producto elaborado en la UCI y en la Industria Cubana del Software (InCuSoft), que se presente al departamento, sea comprobado y evaluado según normas y estándares de calidad, antes de ser entregado al cliente, siendo esta evaluación confiable para los equipos de desarrollo y para los clientes.

Los tipos de pruebas que se realizan en el Departamento de Pruebas de Software de la universidad son los siguientes:

- Pruebas Funcionales.
- Pruebas de Volumen.
- Pruebas Carga y Estrés.
- Pruebas de Regresión.
- Pruebas Exploratorias.
- Pruebas de Seguridad.

Debido a la complejidad de probar un producto, la etapa de prueba puede llegar a ser de las más lentas del proceso de desarrollo de software. Sin embargo, si un proceso de prueba se desarrolla siguiendo una planificación, una metodología y unas herramientas adecuadas se pueden conseguir importantes disminuciones en los costos de desarrollo y mantenimiento del software, así como una reducción en el número de errores.

A pesar de los esfuerzos realizados por la dirección de calidad, todavía las pruebas de software constituyen un problema vigente ya que la mayoría de las pruebas se realizan de forma manual, que conlleva a la pérdida de tiempo y existe el riesgo de que no se detecten todas las No Conformidades existentes en el software. Además, se han identificado algunas herramientas para diferentes tipos de pruebas, pero no se cuenta con una organización adecuada de las actividades para realizar dicho proceso.

Por todo lo antes descrito, se determina que el **Problema científico** del presente trabajo es: ¿Cómo planificar el proceso de pruebas automatizadas en el Departamento de Pruebas de Software?

La investigación tiene como **Objeto de estudio:** Las pruebas de software.

El estudio se centra en el **Campo de acción definido por:** Las pruebas automatizadas para el Departamento de Pruebas de Software.

Para dar solución a lo antes expuesto se define como **Objetivo general:** Desarrollar un procedimiento para las pruebas de software con herramientas automatizadas en el Departamento de Pruebas de Software.

Para dar cumplimiento al objetivo general del trabajo se trazaron los siguientes **Objetivos específicos:**

- Fundamentar los principales conceptos y herramientas, relacionados con los procesos de pruebas automatizadas.
- Definir el procedimiento para las pruebas automatizadas de software.
- Ejecutar las pruebas utilizando herramientas automatizadas.
- Analizar los resultados de la ejecución de las herramientas.
- Evaluar los resultados de la ejecución de las herramientas.

Para dar cumplimiento al objetivo general, se definieron las siguientes **Tareas investigativas:**

- Búsqueda bibliográfica de los conceptos relacionados con las pruebas automatizadas.
- Investigación de las herramientas automatizadas para pruebas de software.
- Selección de las herramientas automatizadas a desarrollar en el procedimiento.
- Definición de los artefactos necesarios por el procedimiento.

- Descripción de los artefactos a utilizar en el procedimiento.
- Definición de los roles y sus responsabilidades en el procedimiento.
- Descripción de un procedimiento para las pruebas automatizadas.
- Selección de la aplicación para realizar las pruebas automatizadas.
- Realización de las pruebas utilizando las herramientas seleccionadas.
- Análisis de los resultados obtenidos en las pruebas.

En la investigación se definen como **Posibles resultados**: Con la descripción de las herramientas utilizadas para cada una de las pruebas y con el análisis de los resultados se contribuirá a la realización de las pruebas automatizadas de una manera eficiente, minimizando el tiempo de ejecución y haciendo un mejor uso de los recursos involucrados.

En el desarrollo del trabajo se aplican diferentes métodos científicos: los métodos teóricos y los métodos empíricos.

Los métodos teóricos.

Analítico–Sintético: Se emplea para conocer las teorías y documentos que ocupan el objetivo de investigación, extrayendo los aspectos más importantes relacionados con la temática.

Se busca la esencia de los procesos de pruebas automatizadas, identificando las principales actividades llevadas a cabo en el proceso de ejecución de las pruebas en los proyectos con las herramientas escogidas. En este método se utiliza la búsqueda y consultas bibliográficas.

Histórico-Lógico: En la primera fase de la investigación se desarrolla un estudio del estado del arte de la problemática analizada. Se manejan las principales etapas del proceso de prueba y conocer los resultados de ellos en la universidad. Se tiene en cuenta la trayectoria histórica real así como la evolución y desarrollo de las herramientas automatizadas para la realización de las pruebas.

Los métodos empíricos

Observación: La observación es un elemento fundamental de todo proceso investigativo, parte del acervo de conocimientos que construye la ciencia ha sido lograda mediante la misma, este método científico es consciente y se orienta hacia un fin o un objetivo determinado.

Entrevistas: La entrevista da la posibilidad de interacción verbal. Se realiza de forma flexible, para permitir que el entrevistado ahonde en los aspectos de interés para el desarrollo de la investigación.

Este método ayuda a obtener información, comprender y precisar bien el problema a resolver, así como a validar la propuesta que se presenta al cliente.

El trabajo está constituido por tres capítulos que conforman el desarrollo de la investigación.

Capítulo 1: "Fundamentación teórica."

Resume todo lo referente a los conceptos vinculados con la temática sobre las características de las pruebas automatizadas, con el fin de lograr una mejor comprensión del problema planteado. Se realiza una descripción de las herramientas, relacionadas con los procesos de pruebas automatizadas.

Capítulo 2: "Descripción del procedimiento."

Se desarrolla una propuesta del procedimiento para la organización de las actividades del proceso que puede ser utilizado para cualquier tipo de prueba.

Capítulo 3: "Evaluación del procedimiento."

Se realizará la validación del procedimiento y la evaluación de los resultados de las pruebas automatizadas con las herramientas escogidas.

Capítulo 1
Fundamentación Teórica

1. Introducción

En el presente capítulo se describen los fundamentos teóricos que sustentan la investigación, haciendo una exposición detallada de los principales conceptos relacionados con el campo de acción, así como identificar las características imprescindibles de las herramientas automatizadas relacionadas con la realización de pruebas de software. Antes de iniciar el desarrollo de dicha solución; es necesario hacer un estudio previo de los temas relacionados con las herramientas que se utilizan para el desarrollo de estas pruebas.

1.1 Calidad de software.

La calidad de software es el grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario. (IEEE, Std. 610-1990). [2]
Según (Pressman) se define como calidad de software a la concordancia del software producido con los requerimientos explícitamente establecidos, con los estándares de desarrollo prefijados y con los requerimientos implícitos no establecidos formalmente, que desea el usuario.[3]

Actualmente en el mundo se buscan software que tengan calidad en su funcionamiento general, aunque en ocasiones no se tomen en cuenta. Muchas empresas dedicadas al desarrollo de software poseen procesos de pruebas manuales o automatizados, cuyo propósito general es encontrar errores para su posterior corrección con el fin de lanzar al mercado productos que tengan una alta eficiencia que le confiera al cliente o usuario, un elevado grado de confiabilidad y seguridad, proporcionando al producto una alta calidad. Las ideas de calidad de proceso y calidad de producto extraídas de los procesos de manufactura son hoy en día lo políticamente correcto en el ciclo de vida del desarrollo software. Obviamente un marco de proceso es esencial, los Modelos de Integración de Madurez de la Capacidad (CMMI), Librería de Infraestructura de Tecnologías de la Información (ITIL), gestión de Tecnología de la Información (TI) están ciertamente de moda. Pero lo que se observa es que la existencia de un proceso definido no garantiza la calidad del producto final y del uso que los usuarios finales hacen de las aplicaciones y sistemas de software desarrollados. El mejor proceso puede producir un producto defectuoso, irrelevante, ineficaz, ineficiente o imposible de mantener. Existen ciertos parámetros que se tienen presente en la calidad de un producto, la Organización Internacional para la Estandarización (ISO) ha emitido algunas normas que definen un modelo de calidad del software, durante muchos años se buscó en la Ingeniería de Software un modelo único para expresar calidad en varios contextos de uso, la ventaja era obvia: poder comparar productos entre sí.

Según las normas ISO 9126-1 (Software Product Evaluation: Quality Characteristics and Guidelines for their Use) es el nombre formal, se reconocen 6 factores de calidad y 27 sub atributos: (Ver Anexo # 18)

❖ **Factores de calidad según ISO 9126-1:**

- **Funcionalidad:** conjunto de atributos que relacionan la existencia de un conjunto de funciones con sus propiedades especificadas. Las funciones satisfacen necesidades especificadas o implícitas:

  - **Adecuación o Idoneidad:** atributos que determinan si el conjunto de funciones son apropiadas para las tareas especificadas.
  - **Exactitud o Precisión:** atributos que determinan que los efectos sean los correctos o los esperados.
  - **Seguridad:** atributos que miden la habilidad para prevenir accesos no autorizados, ya sea accidentales o deliberados, tanto a programas como a datos.
  - **Interoperabilidad:** atributos que miden la habilidad de interactuar con sistemas especificados.
  - **Cumplimiento o Conformidad:** atributos que hacen que el software se adhiera a estándares relacionados.

- **Fiabilidad**: conjunto de atributos que se relacionan con la capacidad del software de mantener su nivel de performance bajo las condiciones establecidas por un período de tiempo.
  - **Madurez:** atributos que se relacionan con la frecuencia de fallas por defectos en el software.
  - **Tolerancia a los fallos:** atributos que miden la habilidad de mantener el nivel especificado de performance en caso de fallas del software.
  - **Recuperación:** atributos que miden la capacidad de restablecer el nivel de performance y recuperar datos en caso de falla, y el tiempo y esfuerzo necesario para ello.
  - **Cumplimiento o Conformidad:** atributos que hacen que el software se adhiera a estándares relacionados.

- **Usabilidad o Facilidad de uso**: conjunto de atributos que se relacionan con el esfuerzo necesario para usar, y en la evaluación individual de tal uso, por parte de un conjunto especificado o implícito de usuarios.
  - **Entendimiento o Comprensión:** atributos que miden el esfuerzo del usuario en reconocer el concepto lógico del software y su aplicabilidad.
  - **Aprendizaje:** atributos que miden el esfuerzo del usuario en aprender la aplicación (control, operación, entrada, salida).
  - **Operabilidad:** atributos que miden el esfuerzo del usuario en operar y controlar el sistema.

- **Atractivo.**
- **Cumplimiento o Conformidad:** atributos que hacen que el software se adhiera a estándares relacionados.

- **Eficiencia**: conjunto de atributos que se relacionan con el nivel de performance del software y la cantidad de recursos usados, bajo las condiciones establecidas. Está indicado por los siguientes sub atributos:
  - **Tiempo de uso o Comportamiento temporal:** atributos que miden la respuesta y tiempos de procesamiento de las funciones.
  - **Uso de recursos:** atributos que miden la cantidad de recursos usados y la duración de tal uso en la ejecución de las funciones.
  - **Cumplimiento o Conformidad:** atributos que hacen que el software se adhiera a estándares relacionados.

- **Mantenibilidad**: Facilidad con que una modificación puede ser realizada. Está indicada por los siguientes sub atributos:
  - **Facilidad de análisis:** atributos que miden el esfuerzo necesario para el diagnóstico de deficiencias o causas de fallas, o para identificación de las partes que deben ser modificadas.
  - **Facilidad de cambio o Cambiabilidad:** atributos que miden el esfuerzo necesario para realizar modificaciones, remoción de fallas o cambios en el contexto.
  - **Estabilidad:** atributos que se relacionan con el riesgo de efectos no esperados en las modificaciones.
  - **Facilidad de prueba :** atributos que miden el esfuerzo necesario para validar el software modificado
  - **Cumplimiento o Conformidad:** atributos que hacen que el software se adhiera a estándares relacionados.

- **Portabilidad**: La facilidad con que el software puede ser llevado de un entorno a otro. Está referido por los siguientes sub atributos:
  - **Facilidad de instalación:** atributos que miden el esfuerzo necesario para instalar el software en el ambiente especificado.
  - **Adaptabilidad:** atributos que miden la oportunidad de adaptación a diferentes ambientes sin aplicar otras acciones que no sean las previstas para el propósito del software.

- **Conformidad:** atributos que miden si el software se adhiere a estándares o convenciones relacionados con portabilidad.
- **Reemplazo:** atributos que se relacionan con la oportunidad y esfuerzo de usar el software en lugar de otro software en su ambiente.
- **Coexistencia:** atributo que indica la capacidad del software de coexistir con otro software independiente en un entorno común compartiendo recursos. [4]

1.2 Pruebas de software

1.2.1 Concepto de prueba de software

Las pruebas de software son un conjunto de herramientas, técnicas y métodos que hacen a la excelencia del desempeño de un programa, así como también la mejor publicidad que una empresa dedicada a la producción de software pueda tener. ¨Las técnicas para encontrar problemas en un programa son extensamente variadas y van desde el uso del ingenio por parte del personal de prueba hasta herramientas automatizadas que ayudan a aliviar el peso y el costo de tiempo de esta actividad.¨ [5]

Los procesos de pruebas aseguran que un sistema hace lo que tiene que hacer. Probar es una práctica habitual de todo proceso productivo, que básicamente consiste en comprobar que un producto tiene las características deseadas. Prácticamente todos los productos que llegan al mercado son probados, de manera que el producto final tenga la resistencia adecuada. Es decir, a lo largo del proceso productivo de cualquier producto se realizan comprobaciones que hacen que el producto final sea el adecuado.

En el caso del software ocurre lo mismo. El proceso de desarrollo de software consta de una serie de etapas, en cada etapa se van desarrollando y ensamblando partes que al terminar van a conformar el producto final. Al igual que ocurre en un proceso productivo normal, cada una de estas partes debe ser probadas. Al igual que la naturaleza y el tipo de prueba a realizar con el software, varía a medida que el desarrollo avanza. Se puede decir que el desarrollo de software es un proceso productivo como otro cualquiera, con sus características y particularidades. Dentro de cada una de las etapas de desarrollo de un software, las pruebas son fundamentales, ya que a partir de ellas es posible controlar que los productos cumplan requisitos mínimos de operabilidad además de garantizar la calidad de estos productos.

¨Las pruebas constituyen una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados, y se realiza una evaluación de algún aspecto del sistema o componente.¨ [6]

Se conocen además como verificación y validación (V&V). Con la verificación de software se determina si los productos de una fase dada, satisfacen las condiciones impuestas al inicio de la fase.

La validación de software se refiere a la evaluación de un sistema o uno de sus componentes durante o al final de su desarrollo para determinar si satisface los requisitos. Se puede definir que las pruebas de software son unas de las actividades fundamentales para la garantía de la calidad, estas identifican posibles fallos que puedan tener los productos, y que son encargadas de demostrar que un sistema que es ejecutado bajo ciertas condiciones está incapacitado para ser liberado, demostrando hasta qué punto las funciones del software parecen funcionar de acuerdo con las especificaciones.

1.2.2 Objetivos de las pruebas de software

- Los objetivos de las pruebas de software son los siguientes:
  - Encontrar y documentar los defectos que puedan afectar la calidad del software.
  - Validar que el software trabaje como fue diseñado.
  - Validar y probar los requisitos que debe cumplir el software.
  - Validar que los requisitos fueron implementados correctamente.

Es importante recalcar que de acuerdo a los objetivos anteriormente expuestos, una prueba tiene éxito cuando descubre errores. Las pruebas demuestra hasta qué punto las funciones del software parecen funcionar de acuerdo con las especificaciones. Los datos que se van recogiendo a medida que se llevan a cabo las pruebas, proporcionan una buena indicación de la fiabilidad del software y, de alguna manera indican la calidad como un todo. Sin embargo, y citando a Pressman, hay una cosa fundamental que una prueba no puede hacer:

“La prueba no puede asegurar la ausencia de defectos, sólo puede demostrar que existen defectos en el software”. [7]

1.2.3 Principios de las pruebas de software

Los principios básicos de pruebas de software a menudo parecen obvios pero por lo general son siempre ignorados, lo cual genera serias consecuencias en el proceso. Entre los principios básicos de pruebas de software se tienen los siguientes aspectos:

- A todas las pruebas se les debería hacer un seguimiento hasta los requerimientos del cliente/usuario.
- Las pruebas deberían planificarse antes de que empiecen. La planificación de las pruebas pueden empezar cuando estén completos los requerimientos.
- Las pruebas deberían empezar de lo individual a lo general.
  - Módulos individuales.

- Grupos de módulos integrados.
- Sistemas totales.

Las pruebas deberían ser conducidas por un equipo independiente. El equipo que creó el sistema no es el más adecuado para llevar a cabo las pruebas.

1.2.4 Tipos de pruebas de software

Dimensión Calidad/Riesgos de Calidad	Tipos de Pruebas
Usabilidad.	Prueba enfocada a factores humanos, estéticos, consistencia en la interfaz de usuario, ayuda sensitiva al contexto y en línea, asistente, documentación de usuarios y materiales de entrenamiento.
Fiabilidad.	**Integridad:** Enfocada a la valoración de la robustez (resistencia a fallos). **Estructura:** Enfocada a la valoración a la adherencia a su diseño y formación. Este tipo de prueba es hecha a las aplicaciones web asegurando que todos los enlaces están conectados, el contenido deseado es mostrado y no hay contenido huérfano. **Stress:** Enfocada a evaluar cómo el sistema responde bajo condiciones anormales (extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible, recursos compartidos no disponibles).
Rendimiento.	**Benchmark:** Es un tipo de prueba que compara el rendimiento de un elemento nuevo o desconocido a uno de carga de trabajo de referencia conocido. **Contención:** Enfocada a la validación de las habilidades del elemento a probar para manejar aceptablemente la demanda de múltiples actores sobre un mismo recurso (registro de recursos, memoria). **Carga:** Usada para validar y valorar la aceptabilidad de los límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba permanece constante. La variación en carga es simular la carga de trabajo promedio y con picos que ocurre dentro de tolerancias operacionales normales.
Soportabilidad.	**Configuración:** Enfocada a asegurar que funciona en diferentes configuraciones de hardware y software. Esta prueba es implementada también como prueba de rendimiento del sistema.

	Instalación: Enfocada a asegurar la instalación en diferentes configuraciones de hardware y software bajo diferentes condiciones (insuficiente espacio en disco, etc.).
Funcionalidad.	**Función:** Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso. **Seguridad:** Asegurar que los datos o el sistema solamente es accedido por los actores deseados. **Volumen:** Enfocada en verificar las habilidades de los programas para manejar grandes cantidades de datos, de entrada, salida o residente en la BD.

Tabla 1: Tipos de pruebas de software.

1.2.5 Ventajas de las pruebas de software

Principalmente, las ventajas que trae la realización de pruebas, en un desarrollo de software son las siguientes:

- Reducen la posibilidad de agregar defectos al software: Si hay que realizar una adición de características requeridas por el cliente y se ve que ya no funcionan bien, algunas de las cosas que anteriormente servían, se puede inferir que la nueva funcionalidad es la que contiene defectos, por lo que no hay necesidad de realizar modificaciones a los componentes realizados anteriormente.
- Reducen la posibilidad de encontrar defectos en funcionalidades ya implementadas.
- Las pruebas son buena documentación: Es preferible ver unas pequeñas líneas de código de prueba, que son concisas y realmente fáciles de entender, a revisar línea por línea de código para poder entender que hace determinado componente de software.
- Reducen el costo del cambio: ya que evitan que se descubran los defectos hasta el final del desarrollo de software.
- Permiten realizar reimplementación: se puede llegar a necesitar reimplementar determinada funcionalidad en un sistema; debido a fallos de seguridad, rendimiento, o simplemente porque no reunía lo que el cliente esperaba de este. Entonces dada tal situación, las pruebas que se realizan a dicha funcionalidad, van a permitir que se vuelva a desarrollar de manera segura; ya que las pruebas encierran el criterio de aceptación sobre la funcionalidad, permitiendo de esta forma que se vuelvan a desarrollar algo acorde con las especificaciones.
- Restringe las características a implementar: muchas veces los programadores pierden tiempo en detalles que la especificación no pedía. Con las pruebas el programador sabe que tiene que programar

dicha funcionalidad y también probarla, por lo que se restringe a lo que los diseñadores de las pruebas hayan realizado.

- Hacen que el desarrollo sea más rápido: ya que a medida que se agregan características al software, las funcionalidades anteriores pueden fallar, pero si se han hecho las pruebas pertinentes a las funcionalidades anteriores, se puede descartar inmediatamente que existan defectos en éstas, por lo que se puede concentrar tranquilamente en la funcionalidad nueva.

1.2.6 Estrategia de prueba

Una estrategia de prueba de software integra las técnicas de diseño de casos de pruebas en una serie de pasos bien planificados que dan como resultado una correcta construcción del software. La estrategia proporciona un mapa que describe los pasos que hay que llevar a cabo como parte de la prueba, cuándo se deben planificar y realizar esos pasos, y cuánto esfuerzo, tiempo y recursos se van a requerir. Por lo tanto cualquier estrategia debe incorporar la planificación de las pruebas, el diseño de los casos de pruebas, la ejecución de las pruebas y la agrupación y evaluación de los datos resultantes. Para aplicar pruebas al software se deben seguir un conjunto de estrategias para lograr que estas se hagan en el menor tiempo posible y con la calidad requerida, además de lograr que arrojen los resultados esperados. Existen características que se pueden aplicar a todas las estrategias de pruebas, muchas de ellas definidas por Pressman.

1.2.7 Procedimiento de prueba

Un procedimiento de prueba especifica cómo realizar uno o varios casos de pruebas. Este puede ser una instrucción para un individuo sobre cómo realizar un caso de prueba manualmente o puede ser una especificación de cómo interactuar manualmente con una herramienta automatizada para crear componentes ejecutables de pruebas.

1.3 Pruebas según el grado de automatización

Las pruebas se pueden categorizar según el grado de automatización, éstas pueden ser:

Pruebas manuales: son las que se hacen normalmente al programar o las que ejecuta una persona con la documentación generada durante la codificación. Por ejemplo comprobar cómo se visualiza el contenido de una página web en dos navegadores diferentes.

Pruebas automatizadas: se usa un determinado software para sistematizar las pruebas y obtener los resultados de las mismas. Por ejemplo verificar un método de ordenación.

1.3.1 Pruebas manuales de software

Las pruebas manuales son el método de pruebas de software más antiguo y riguroso. Requieren a un probador que ejecute de manera manual, operaciones en la aplicación sin ayuda de herramientas de automatización. Estas pruebas ayudarán a descubrir cualquier problema relacionado con la funcionalidad del producto, especialmente defectos relacionados con la usabilidad y la interfaz gráfica de la aplicación.

Las herramientas de automatización ejecutan únicamente scripts diseñados para testear una funcionalidad o requisitos específicos y no poseen la habilidad de toma de decisiones, ni grabación de discrepancias no incluidas en el script.

1.3.2 Pruebas automatizadas de software

La automatización de pruebas es la parte de ciclo de calidad, en la que el software de automatización es utilizado para controlar la ejecución de pruebas, comparar los resultados, preparar las condiciones y realizar informes. Las pruebas automatizadas son efectivas en entornos donde los cambios son frecuentes o en aplicaciones en las que se esperan releases críticos, estos son versiones del producto.
Existe una gran variedad de herramientas automatizadas de pruebas y frameworks disponibles, este último es una estructura conceptual y tecnológica de soporte definida como artefactos y módulos de software, que puede incluir soportes de programas, biblioteca y un lenguaje interpretado. Esto, junto con las nuevas metodologías de desarrollo de aplicaciones, ha creado un reto a la hora de diseñar frameworks de pruebas que sean mantenibles y suites de pruebas destinadas a regresión.

1.3.3 Propósitos de las pruebas automatizadas

Entender los propósitos generales y conocer porque se automatizan las pruebas, posibilitará al desarrollador identificar en qué pueden serles útiles en cada momento. En muchos casos, se tiene la idea de que las pruebas son algo bueno, sólo que a veces no se está consciente del porqué. A esta idea se pueden aplicar los criterios siguientes:

- Las pruebas deben ayudar a mejorar la calidad.
- Las pruebas deben ayudar a reducir riesgos.
- Las pruebas deben ser fáciles de ejecutar.
- Las pruebas deben ser fáciles de escribir y mantener.
- Las pruebas deben requerir el mínimo mantenimiento mientras evoluciona el sistema.

1.3.4 Tipos de pruebas que se automatizan

Existen muchos puntos de vistas por las cuales se puede o no definir tipos de pruebas. Uno de estos puntos es dividir las pruebas en las siguientes categorías:

Pruebas de funcionalidades cruzadas: verifican varios aspectos del comportamiento del sistema que interrelacionan funcionalidades específicas. Generalmente son manuales y se realizan para criticar y valorar el producto.

Pruebas por funcionalidad: verifican el comportamiento del sistema, en respuesta a un estímulo en particular. Prueban el sistema a varios niveles. Estas pruebas son usualmente automatizadas y forman parte del apoyo al desarrollo del producto.

En el caso de las pruebas por funcionalidad verifican el comportamiento directamente observable de una pieza de software en respuesta a un estímulo específico.

La funcionalidad puede estar relacionada con el negocio o con requerimientos operacionales, incluyendo el mantenimiento del sistema y escenarios específicos de tolerancia a fallas. Estos requerimientos pueden ser expresados como casos de uso, historias de usuarios o escenarios de pruebas.

1.3.5 Retos de automatización del proceso de prueba

- Falta de herramientas, debido fundamentalmente a su elevado precio o a que las existentes no se ajusten al propósito o entorno para el que se necesitan.
- Falta de compatibilidad y interoperabilidad entre herramientas.
- Falta de proceso de gestión de la configuración. Igual que las diferentes versiones del código fuente, las pruebas, especialmente las de regresión, deben someterse a un control de versiones.
- Falta de procesos básicos de pruebas y de conocimiento de que es lo que se debe probar.
- Falta de uso de las herramientas de pruebas que ya se poseen, bien por su dificultad de uso, por falta de tiempo para aprender a manejarla, por falta de soporte técnico, etc.
- Formación inadecuada en el uso de las herramientas.
- Las herramientas no cubren todos los tipos de pruebas que se desean (corrección, fiabilidad, seguridad, rendimiento, etc.). Obviamente, a la hora de elegir la herramientas, deberían tenerse priorizados los tipos de pruebas, y entonces hacer la elección de la herramienta basado en esto. A veces también es necesario utilizar no una, sino varias herramientas de pruebas, así como tener en cuenta que es imposible automatizar el 100% de las pruebas.
- Falta de soporte o comprensión por parte de los jefes, debido a la escasa importancia que habitualmente se le da a la fase de prueba.

- Organización inadecuada del equipo de prueba.
- Adquisición de una herramienta inadecuada. [8]

1.3.6 Razones por la que se automatizan las pruebas

- El ciclo de prueba manual es muy largo.
- El proceso de prueba manual es propenso a errores.
- Liberar a la gente para realizar tareas creativas.
- Generar un ambiente de confianza soportado por las pruebas.
- Obtener realimentación de forma temprana y con alta frecuencia.
- Generar conocimiento del sistema en desarrollo a partir de las pruebas.
- Generar la documentación del código consistente.
- Generar una mejor utilización de los recursos a partir de menores costos. [9]

1.3.7 Ejemplos de pruebas automatizadas en el DPS.

Existen algunos tipos de pruebas automatizadas que actualmente se realizan en el Departamento de Pruebas de Software de la Universidad de las Ciencias Informáticas y estas son las siguientes:

- Pruebas Funcionales.
- Pruebas Carga y Estrés.
- Pruebas de Regresión.
- Pruebas Exploratorias.
- Pruebas de Volumen.
- Pruebas de Seguridad.

Las pruebas que se están automatizando en el DPS son las pruebas de carga y estrés y las pruebas funcionales. Dicha automatización radica en una herramienta capaz de detectar todas las posibles No Conformidades de una aplicación o sistema. Logrando resultados reales que permiten mejorar la calidad de los productos probados.

1.4 Pruebas funcionales

Las pruebas funcionales son consideradas como la parte más importante del desarrollo. Ellas están desarrolladas bajo la perspectiva del usuario, confirmando que el sistema hace lo que los usuarios esperan que haga. Un error funcional en su aplicación puede tener consecuencias catastróficas, desde la pérdida de credibilidad de los clientes, hasta grandes pérdidas económicas.

De acuerdo con Pressman, las pruebas funcionales pueden realizarse en base a dos enfoques principales: [10]

- Prueba de caja blanca.
- Prueba de caja negra.

La prueba de caja blanca del software, se basa en el minucioso examen de los detalles procedimentales. Se comprueban los caminos lógicos del software proponiendo casos de prueba que ejerciten un conjunto de condiciones. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el estado esperado. La prueba de caja blanca es un método de diseño de casos de pruebas que usa la estructura de control del diseño procedimental para obtener los casos de pruebas.

La prueba de caja negra se centra en los requisitos funcionales del software, es decir la prueba de caja negra permite obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales. La prueba de caja negra intenta encontrar errores de las siguientes categorías:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

A diferencia de la prueba de caja blanca, que se lleva a cabo previamente en el proceso de prueba, las pruebas de caja negra tienden a aplicarse durante fases posteriores del proceso de pruebas.

❖ **Objetivo**

Las pruebas funcionales tienen como objetivo principal:

- Asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada y salida de datos, procesamiento y obtención de resultados.

❖ **Metas**

Las metas de estas pruebas son:

- Verificar el procesamiento, recuperación e implementación adecuada de las reglas del negocio.
- Verificar la apropiada aceptación de datos.

❖ **Enfoque**

El enfoque de las pruebas funcionales es:

- Los requisitos funcionales (Casos de Uso).
- Las reglas del negocio.

❖ **Técnica**

- **Caja Negra**

Se ejecuta cada caso de uso, flujo de caso de uso, o función, usando datos válidos e inválidos, para verificar lo siguiente:

- Que se aplique apropiadamente cada regla de negocio.
- Que los resultados esperados ocurran cuando se usen datos válidos.
- Que sean desplegados los mensajes apropiados de error y precaución cuando se usan datos inválidos.

❖ **Automatización de las pruebas funcionales**

Para la automatización de las pruebas funcionales son especialmente indicadas las herramientas de ejecución de las pruebas de captura y reproducción. Estas herramientas permiten al probador capturar y grabar pruebas, para luego editarlas, modificarlas y reproducirlas en distintos entornos.

Las herramientas que graban la interfaz de usuario a nivel de componentes y no de bitmaps son más útiles. Durante la grabación se capturan las acciones realizadas por el probador, creando automáticamente un script en algún lenguaje de alto nivel. Luego el probador modifica el script para crear una prueba reusable y mantenible. En general estas herramientas vienen acompañadas de un comparador, que compara automáticamente la salida en el momento de ejecutar el script con la salida grabada.

❖ **Un caso de prueba funcional es bien elegido si se cumple que:**

- Reduce el número de otros casos necesarios para que la prueba sea razonable. Esto implica que el caso ejecute el máximo número de posibilidades de entrada diferentes para así reducir el total de casos.

- Cubre un conjunto extenso de otros casos posibles, es decir, no indica algo acerca de la ausencia o la presencia de defectos en el conjunto específico de entradas que prueba, así como de otros conjuntos similares.

1.5 Prueba de estructura

Las pruebas de estructura se realizan a aplicaciones web, evaluando el estado de sus enlaces y contenido. Esta prueba se basa en evaluar los siguientes aspectos de una aplicación:

❖ Existencia de enlaces rotos:

El objetivo de este aspecto es escanear la aplicación en busca de enlaces que no estén conectados a ningún recurso o página.

- Existencia de contenido huérfano:

El objetivo de este aspecto es escanear el servidor de la aplicación en busca de archivos a los cuales no se pueda acceder, es decir que estén en el servidor pero que no existen formas de llegar a él.

- Mostrar el contenido deseado:

El objetivo de este aspecto es escanear la aplicación, para comprobar que los enlaces que referencien a archivos o páginas externas estén mostrando el contenido que realmente se desea que observe el usuario.

Esta prueba se debe efectuar con el apoyo de alguna herramienta, porque ejecutarla manualmente es muy difícil y consume mucho tiempo y además se puede correr el riesgo que pase algún vínculo por alto. Para realizar una prueba de estructura es necesario conocer varias características de la aplicación para luego seleccionar la herramienta que se adapte a las condiciones del artefacto, se debe tener dominio de:

Tipo de aplicación: este aspecto recoge si la aplicación a probar es una aplicación web o desktop, en caso de ser del último tipo no se puede realizar la prueba, por lo menos automatizada debido a que las herramientas existentes solamente son compatibles con aplicaciones web.

Autenticación: este aspecto recoge si para acceder a las páginas de la aplicación es necesaria la autenticación del usuario, en este caso hay que conocer el par usuario/contraseña y los nombres de estos campos en la página de autenticación.

URL: este aspecto recoge la URL para acceder a la aplicación.

Directorio ftp: este aspecto recoge la dirección del directorio ftp o local donde se encuentre alojada la aplicación en el servidor, para ser utilizado en caso que se desee buscar archivos huérfanos.

Conociendo estas propiedades de la aplicación solamente falta que el probador decida que herramienta es la que se adapta y comenzar la prueba. [11]

1.6 Pruebas de carga y estrés

1.6.1 Prueba de carga

- **Objetivo**

Verificar el rendimiento del software en tiempo de ejecución, dentro del contexto de un sistema integrado. Comprobar la capacidad del sistema para manejar volúmenes de datos extremos de acuerdo al tiempo de respuesta establecido para el sistema.

- **Metas**

Validar en la aplicación:

- Comprobar los tiempos de respuestas del sistema en una cantidad limitada de escenarios de trabajo (a nivel de número de usuarios y número de transacciones), bajo una configuración de hardware y software constante.
- Comprobar el tiempo de respuesta al realizar una función.
- Comprobar el tiempo de respuesta al realizar accesos concurrentes a una determinada información.
- Atender múltiples solicitudes de parte de los actores que acceden a un mismo recurso.

1.6.2 Prueba de estrés

❖ **Objetivo**

Verificar cómo se comporta el sistema bajo condiciones anormales.

❖ **Metas**

Validar en la aplicación:

- Carencia de sistemas externos con los que interactúa el sistema.
- Aplicar carga excesiva de trabajo al sistema (extrema sobrecarga).
- Hardware no disponible.
- Recursos compartidos no disponibles.
- Además verifica que hace el sistema cuando el usuario no hace lo que supuestamente debe hacer. [12]

1.7 Herramientas automatizadas

1.7.1 Herramientas para la automatización de las pruebas

Una herramienta es un objeto elaborado, a fin de facilitar la realización de una tarea automatizada que requiere de una aplicación correcta funcionalmente. En el mundo el uso de las herramientas se ha hecho tan imprescindible, para automatizar muchos procesos. El flujo de prueba específicamente es un flujo muy importante en el desarrollo de un software, que necesita el uso de herramientas automatizadas para un mejor resultado a la hora de hacer las pruebas al producto.

Existen herramientas especiales para cada una de las etapas de un proyecto de desarrollo de software. Algunos proveedores de este tipo de herramientas ofrecen una serie integrada que da soporte tanto a las pruebas como al desarrollo de software durante todo un proyecto, desde que se reúnen los requisitos hasta que se inicia el funcionamiento en vivo del sistema. Con estas herramientas de pruebas se proporciona un análisis detallado de la calidad y rendimiento de las aplicaciones.

1.7.2 Clasificación de las herramientas automatizadas

Por regla general, las herramientas automatizadas se agrupan en tres categorías:

- Front-end.
- Back-end.
- Integrales.

Esta clasificación recalca las actividades del proceso de desarrollo donde las herramientas tienen su mayor papel. Cada categoría es de utilidad y ninguna es más valiosa que otra.

- **Herramientas de tipo front-end**

Las herramientas de tipo front-end, automatizan las primeras actividades del proceso de desarrollo de sistemas. Entre los muchos aspectos que se toman en cuenta al desarrollar herramientas para esta fase, se hallan las técnicas de soporte para ayudar al analista a preparar especificaciones formales que carezcan de ambigüedades. Validar las descripciones del sistema con el objeto de determinar su consistencia, y seguir la evolución de los requerimientos de la aplicación, en características que formen parte del sistema que finalmente será implantado. A menudo, las herramientas de tipo front-end proporcionan soporte para el desarrollo de modelos gráficos de sistemas y procesos. Los diagramas de flujo de datos son representativos de este tipo de herramienta. Los diagramas de flujo de datos representan en forma gráfica (más que por escrito) los procesos y flujos de datos del sistema.

- **Herramientas de tipo back-end**

Las herramientas de tipo back-end tienen como finalidad ayudar al analista a formular la lógica del programa, los algoritmos de procesamiento y la descripción física de datos, también ayudan a la interacción con los dispositivos (para entrada y salida). Estas actividades convierten los desafíos lógicos del software en un código de programación, que es el que finalmente da existencia a la aplicación. Dado que su empleo está destinado al desarrollo de software, este tipo de herramienta también se conoce como herramientas para programación asistida por computadora.

- **Herramientas integrales o asistidas por computadora para la ingeniería de sistemas (CASE)**

Las siglas CASE se emplean con bastante frecuencia en la comunidad de sistemas de información, para denotar la ingeniería de sistemas asistida por computadora o la ingeniería de software asistida por computadora. Aunque el uso de este último término está más diseminado, el primero es más exacto ya que el objetivo a largo plazo de las herramientas CASE es automatizar los aspectos claves de todo el proceso de desarrollo, desde el principio hasta el final. Para aquellos que emplean el término ingeniería de software asistida por computadora, se hace mención de que el desarrollo de una aplicación comienza con la

especificación de requerimientos, no con la codificación del software. Es así como las extensiones de CASE hacen referencia al mismo proceso. Esta sección examina los componentes que forman parte de las herramientas CASE y los métodos utilizados para integrar las herramientas dentro de un sistema de información.

1.7.3 Herramientas para pruebas funcionales

❖ **Selenium IDE**

Es un plugin de Mozilla Firefox que pertenece al juego de herramientas SeleniumHQ, permite realizar juegos de pruebas sobre aplicaciones web. Para ello realiza la grabación de la acción seleccionada (navegación por una página) en un "script", el cual se puede editar y parametrizar para adaptarse a los diferentes casos, y lo que es más importante su ejecución se puede repetir tantas veces como se quiera. El principal objetivo de este plugin es crear pruebas funcionales, aunque no se puede pasar por alto que este tipo de herramientas permite automatizar tareas que requieren un cierto procesamiento mental básico:

- Rellenar formularios (autenticación o cualquier otro tipo).
- Navegación web.
- Acciones de gestión (CRUD de comentarios blog / correos / noticias / etc.).

Esta herramienta permite al desarrollador web ahorrarse mucho esfuerzo (mucho esfuerzo = muchas horas), cada vez que se resuelve alguna incidencia o se genera una versión nueva. Además permite automatizar la realización de las pruebas, ya sean pruebas especificas (una acción en particular) o juegos de pruebas (un conjunto de acciones). Las pruebas con Selenium se pueden escribir como tablas HTML o codificadas en varios idiomas como C #, PHP, Perl, Python y se pueden ejecutar directamente en la mayoría de las modernas IDE browsers.

Características

- Facilidad de registro y ejecución de las pruebas.
- Referencia a objetos DOM en base al ID, nombre o a través de XPath.
- Autocompletado para todos los comandos.
- Las acciones pueden ser ejecutadas paso a paso.
- Herramientas de depuración y puntos de ruptura (breakpoints).
- Los test pueden ser almacenados como HTML entre otros formatos.
- Soporte para Selenium user-extensions.js.
- Ejecución en varios navegadores.

- Uso de diferentes API's, en diferentes lenguajes (PHP, Ruby, JAVA, JavaScript). [13]

1.7.4 Herramienta para prueba de carga y estrés

❖ **JMeter**

Es una herramienta desarrollada dentro del proyecto Jakarta, que permite realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones web. Es un software de código abierto, originalmente se diseñó para probar aplicaciones web, pero desde entonces se ha expandido a otras funciones de pruebas. JMeter permite realizar pruebas web clásicas, pero también permite realizar pruebas de FTP, JDBC, JNDI, LDAP, SOAP/XML-RPC, y WebServices (en Beta). Permite la ejecución de pruebas distribuidas entre distintos ordenadores, para realizar pruebas de rendimiento. Además permiten activar o desactivar una parte de la prueba, lo que es muy útil cuando se está desarrollando un test largo, y se desea deshabilitar ciertas partes iniciales que sean muy pesadas o largas. Tiene la forma de generar un caso de prueba a través de una navegación de usuario. Se necesita tener una Máquina Virtual de Java 1.3 o superior instalada en el sistema operativo Windows XP Professional en este caso.

Estas son otras funcionalidades que brinda la herramienta JMeter:

- Ofrece la posibilidad que el propio usuario desarrolle en Java un “Controller” a medida, cumpliendo una interfaz Java, y depositando el archivo con extensión .jar correspondiente al desarrollo en el directorio “lib” de JMeter, lo que permite reducir el consumo de rendimiento del CPU.
- Permite realizar pruebas distribuidas en distintos ordenadores, que actuarán como clientes simulando varios hilos que harán función de usuarios.
- Permite generar un caso de prueba a través de una navegación de usuario.
- Posee vario tipos de informes.
- Permite envío de parámetros.
- Permite envío de un fichero adjunto.
- Se puede cargar y probar el rendimiento de diferentes tipos de servidores:
  - Web - HTTP, HTTPS.
  - SOAP.
  - Base de datos a través de JDBC.
  - LDAP.
  - JMS.
  - Correo - POP3 (S) e IMAP (S) [14]

1.7.5 Herramienta de prueba de estructura

❖ **Xenu Link Sleuth**

Es un programa de lo más útil para la correcta utilización de los links en una página web. Este comprobará automáticamente todo link para evitar problemas como fallas en las direcciones de los links, tanto así como las redirecciones, es algo simple pero verdaderamente útil. Al final de todo nos presentará un informe con los errores que pudo haber tenido el programa. Link Sleuth funciona al actuar con un navegador web, cargando documentos mediante los protocolos HTTP web y / o HTTPS. [15]

Características

- Es gratuito.
- Simple interfaz de usuario.
- Posee un mejor informe de errores (no sólo error de red).
- El informe se puede ver fácilmente, incluso cuando tiene las URL largas.
- Utiliza mucho menos espacio en disco para archivos intermedios, mucho más pequeño que archivos ejecutables.
- Carga los archivos guardados mucho más rápidos.
- Soporta SSL sitios web ("http://").
- Realiza una búsqueda de archivos locales huérfanos.
- Tiene manejo especial de las redirecciones.
- Mapa del sitio.
- Comprueba el orden parcial, mediante peticiones simultáneas menos en un único servidor.

1.8 Conclusiones

En el presente capítulo se han descrito los elementos teóricos sobre los cuales se sustentará la propuesta del procedimiento, tomando como ejemplo las pruebas funcionales, de estructura y de carga y estrés. Se definen los conceptos generales de pruebas, sus tipos, su automatización. Se destaca un estudio de las principales herramientas. De la investigación realizada se puede concluir, que no existe actualmente un procedimiento para llevar a cabo las pruebas automatizadas en el Departamento de Pruebas de Software de la Universidad de las Ciencias Informáticas.

Capítulo 2

Descripción del Procedimiento

2. Introducción

En el presente capítulo se describe el procedimiento de las pruebas automatizadas diseñado para ser aplicado en el Departamento de Prueba de Software de la Universidad de las Ciencias Informáticas; dicho procedimiento cuenta con aspectos que organizan y especifican las actividades que se realizarán, definiendo el alcance, los objetivos de cada unas de las tareas, seleccionando los roles que intervienen y los artefactos generados durante el proceso.

2.1 Procedimiento para la realización de las pruebas de software

Procedimiento: es una serie común de instrucciones, operaciones y pasos definidos, que permiten realizar un trabajo de forma correcta, que presenta artefactos, actividades y roles.

2.1.1 Resumen general

Con la propuesta del procedimiento se definirá de forma detallada los pasos a seguir para ejecutar las pruebas de software utilizando herramientas automatizadas.

2.1.2 Alcance

El procedimiento puede ser aplicado por los especialistas y probadores del Departamento de Prueba de Software y por los probadores de los grupos de calidad de las facultades de la Universidad.

2.1.3 Artefactos de entrada

Los artefactos de entrada no son más que objeto con un propósito definido, en este caso, creación de documentos y aplicaciones con la finalidad de desempeñar una función o realizar algún trabajo.
Los artefactos de entrada que se deben proporcionar para ser utilizados en la ejecución de las pruebas son los siguientes:

- Aplicación.
- Herramientas.
- Requisitos Funcionales.
- Requisitos No Funcionales.
- Casos de Uso.

- Pre Plan de Prueba.
- Plan de Prueba.
- Casos de Pruebas.
- Documento de No Conformidades.
- Criterio de Criticidad.

2.1.4 Artefactos de salida

Los artefactos de salida son el resultado o respuesta de un proceso, que generalmente es utilizado por otro proceso hasta cumplir el objetivo final. Los artefactos de salida generados por el procedimiento en la realización de las pruebas con herramientas automatizadas son los siguientes:

- Pre Plan de Prueba.
- Casos de Pruebas.
- Plan de Prueba.
- Documento de No Conformidades.
- Documento Final de No Conformidades.
- Informe de Evaluación de las Pruebas.

2.1.5 Descripción de los artefactos

Los artefactos generados que se proponen para llevar a cabo el procedimiento son los siguientes:

Aplicación: es un programa diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de trabajo.

Requisitos Funcionales: define lo que el sistema debería de hacer en cuanto a su comportamiento específico.

Requisitos No Funcionales: especifica los criterios que se deben usar para juzgar el funcionamiento de un sistema, estos son llamados “cualidades de un sistema”.

Plan de Prueba: describe las estrategias, recursos y planificación de la prueba. Incluyendo la definición del tipo de prueba a analizar para cada iteración y sus objetivos, el nivel de cobertura y de código necesario además del porcentaje de pruebas que deberían ejecutarse con un resultado específico.

Casos de Uso: es una técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización de software. Cada caso de uso proporciona uno o más escenarios, que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico.

Casos de Pruebas: especifica una forma de probar el sistema, incluyendo la entrada o resultados de datos con los que se ha de verificar y las condiciones para esto.

Documento Final de las No Conformidades: documento que recoge todos los incumplimientos o errores que aparecen en el producto.

Informe de Evaluación de las Pruebas: documento que se encarga de recoger la evaluación de los resultados de las pruebas, y el estado de los defectos.

Criterio de Criticidad: documento que establece los parámetros para declarar un producto en estado crítico de terminación. Son aplicables en tres momentos fundamentales: cuando se hace la revisión de la solicitud (RS); durante el proceso de pruebas exploratorias (PE) y durante las pruebas formales.

2.1.6 Actividades

Código	Actividades	Breve Descripción
1	Estudio Preliminar.	Se estudian las principales funcionalidades del producto y los requerimientos especificados por el cliente.
2	Planificación.	Se planifica el proceso de prueba.
3	Configuración de Entorno.	Se configura el ambiente de pruebas, se instalan las herramientas necesarias y el software a probar en la versión correspondiente.
4	Diseño de Prueba.	Se diseñan los casos de pruebas que se van a utilizar durante el proceso de prueba.
5	Ejecución de las Pruebas.	Se verifica que la herramienta seleccionada fue instalada correctamente, se ejecutan las pruebas, y se detectan las No Conformidades.
6	Gestión de las No Conformidades.	Se gestionan todas las No Conformidades. Las No Conformidades reportadas son validadas.
7	Evaluación de las Pruebas.	Se realiza un resumen del proceso, teniendo en cuenta el personal que participa, la cantidad de No Conformidades por clasificaciones, el cronograma real de prueba y se evalúa el comportamiento de la asistencia de los probadores y especialistas.

Tabla 2: Actividades del Procedimiento.

2.1.7 Representación gráfica

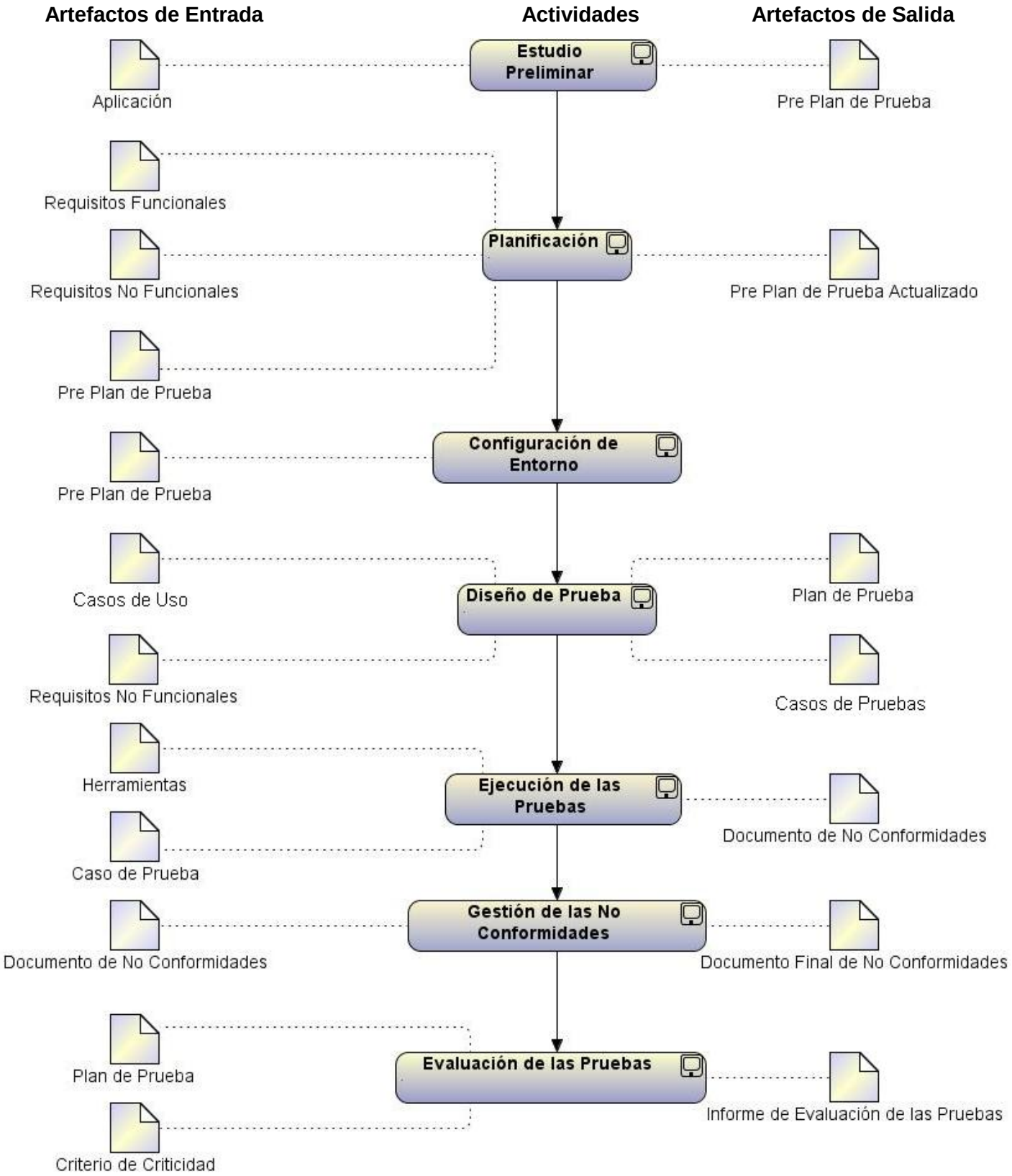

Figura 1: Esquema del proceso.

2.1.8 Relación entre actividades del proceso

Código	Actividades	Responsables	Participantes	Resultados a obtener
1	Estudio Preliminar.	Especialista de Prueba.	✓Especialista de Prueba. ✓Cliente.	Se obtiene un Pre Plan de Prueba y una lista priorizada de las funcionalidades más importantes.
2	Planificación.	Especialista de Prueba.	✓Especialista de Prueba.	Los requisitos funcionales y los requisitos no funcionales.
3	Configuración de Entorno.	Especialista de Prueba.	✓Especialista de Prueba.	Se crea el entorno de prueba, instalándose la herramienta seleccionada.
4	Diseño de Prueba.	Diseñador.	✓Diseñador.	Se crean los casos de pruebas y se obtiene el Plan de Prueba.
5	Ejecución de las Pruebas.	Probador.	✓Probador.	Se verifica la instalación de las herramientas con las que se van a ejecutar las pruebas.
6	Gestión de las No Conformidades.	Probador.	✓Probador ✓Especialista de Prueba	Se genera un documento final de las No Conformidades.
7	Evaluación de las Pruebas.	Especialista de prueba.	✓Especialista de Prueba. ✓Probador.	Se obtiene un Informe de Evaluación de las Pruebas.

Tabla 3: Relación entre actividades y procesos.

2.2 Descripción de las actividades

2.2.1 Estudio Preliminar

Procedimiento de la ejecución de las actividades	
Código	1.1
Actividad	Estudio Preliminar.
Proceso	Procedimiento de pruebas automatizadas en el Departamento de Pruebas de Software.
Responsable	Especialista de Prueba.
Participantes	Especialista de Prueba. Cliente.

Tabla 4: Procedimiento de la ejecución de la actividades: Estudio Preliminar.

❖ **Objetivo**

Estudiar la factibilidad de realizar las pruebas y definir el alcance de estas.

❖ **Descripción**

En esta actividad se estudian las principales funcionalidades del producto. A partir de estos datos se realiza la propuesta de servicio de prueba. Que incluye las reuniones que se realizan con el cliente en las que se explica la aplicación por completo y sus principales funcionalidades; haciéndose una lista de las mismas basándose en el menú de la aplicación y las funcionalidades de cada menú. Para poder realizar estas pruebas, se debe:

- Definir dónde se realizará la ejecución de las pruebas, si en el laboratorio del equipo de prueba, en las instalaciones del cliente o se usará una modalidad de trabajo mixta.
- Definir los atributos de calidad externa que serán tenidos en cuenta para la prueba del producto, según el modelo de calidad [ISO9126]. Se definen cuáles de estas características de calidad se debe verificar que estén presentes en el producto y con qué relevancia deben ser verificadas.

Después de ponerse al tanto con las especificaciones del cliente sobre el producto, se determinará si es posible generar los casos de pruebas a partir de estas especificaciones, en caso que no exista o de que estén incompletas, se deben definir junto con el cliente la forma en que serán generadas o mejoradas.

Al realizar pruebas funcionales se debe decidir si la salida observada al ejecutar el programa es la salida esperada. La salida esperada está expresada en las especificaciones del producto, se deben validar los

requerimientos para poder usarlos como guía para las pruebas. Obteniendo una lista priorizada de las funcionalidades más importantes, así como identificar las funcionalidades que no serán verificadas.

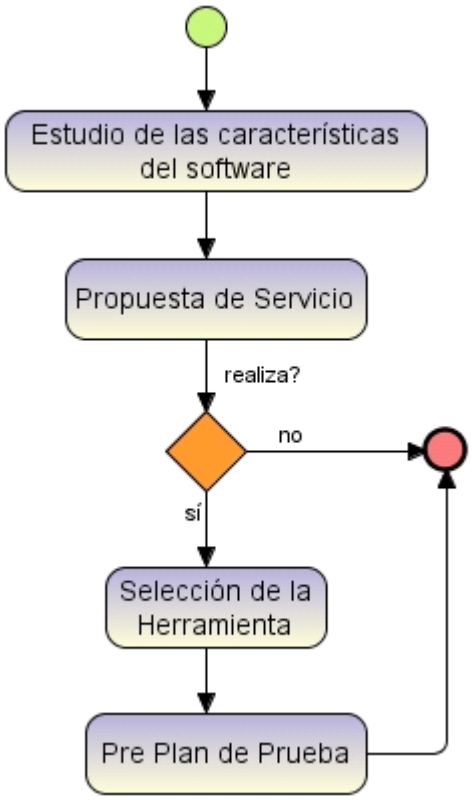

Figura 2: Actividades en Estudio Preliminar.

Durante el Estudio Preliminar se conformará el Pre Plan de Prueba el cual no será un documento final, si no que estará sometido a cambios durante la etapa inicial del procedimiento hasta llegar a conformar el Plan de Prueba. El Pre Plan de Prueba se actualizará en cada una de las fases donde se le añadirá información significativa para realizar las siguientes actividades.

2.2.2 Planificación

Procedimiento de la ejecución de las actividades	
Código	1.2
Actividad	Planificación.
Proceso	Procedimiento de pruebas automatizadas en el Departamento de Pruebas de Software.
Responsable	Especialista de Prueba.
Participantes	Especialista de Prueba. Cliente.

Tabla 5: Procedimiento de la ejecución de la actividades: Planificación.

❖ **Objetivos**

El objetivo de esta etapa es planificar el proceso de prueba.

❖ **Descripción**

Luego de hacer un Estudio Preliminar se van a seleccionar los recursos humanos y tecnológicos con los que se va a contar durante el desarrollo del proceso de prueba. Además en esta actividad se revisa junto al cliente los atributos de calidad que serán tenidos en cuenta durante las pruebas, a partir de lo definido durante el Estudio Preliminar, donde se definieron los atributos y la relevancia de los mismos. En cuanto a las funcionalidades se realiza análisis del riesgo para cada una. Luego se gestionan las pruebas, es decir quién, cómo y cuándo se realizan y se obtiene el Pre Plan de Prueba actualizado hasta ese momento.

❖ **Selección de roles**

Para la aplicación del procedimiento teniendo como ejemplo las pruebas automatizadas se conformará un equipo de trabajo seleccionado por el especialista.

En el transcurso del procedimiento se hace necesario contar con 3 roles por el equipo de trabajo, estos son:

- Especialista de Prueba.
- Diseñador de Prueba.
- Probador.

Roles	Responsabilidad
Especialista de Prueba.	✓ Asegurar la planificación apropiada y dirección de los recursos de las pruebas. ✓ Evaluar el progreso y efectividad del proceso de pruebas. ✓ Evaluar resultados obtenidos. ✓ Identifica tipos de pruebas a realizar.
Diseñador de Prueba.	✓ Identificar, priorizar, seleccionar y describir los casos de pruebas. ✓ Identificar las técnicas apropiadas y pautas para llevar a cabo las pruebas. ✓ Definir la configuración del entorno para realizar las pruebas.
Probador	✓ Ejecutar las pruebas. ✓ Conformar documentación al culminar el proceso.

Tabla 6: Roles y responsabilidades.

❖ **Selección de los recursos materiales**

Recursos Materiales	
Recursos	**Tipos**
Servidor de Base de Datos	[Características del software.] [Características de hardware.]
Servidor de Aplicación	[Características del software.] [Características de hardware.]
PC Clientes	[Características de hardware.]
Red o Subred	[Tipo de red]

Tabla 7: Selección de los recursos materiales.

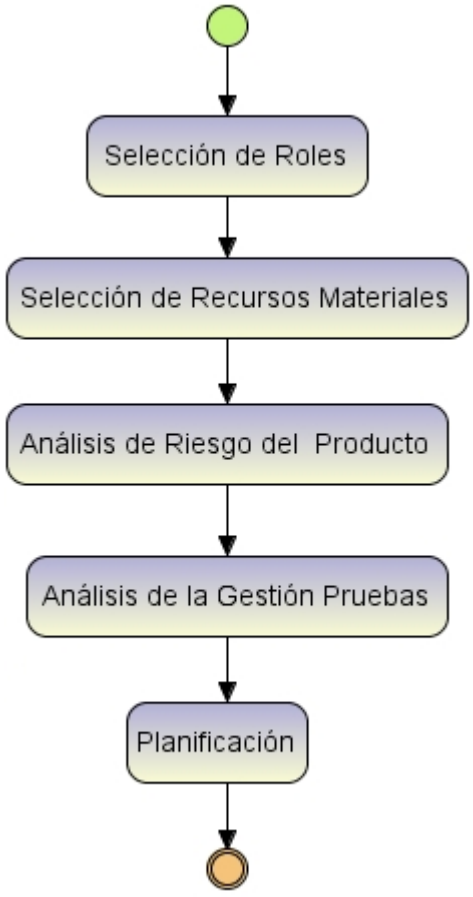

Figura 3: Actividades en Planificación.

2.2.3 Configuración de Entorno

Procedimiento de la ejecución de las actividades	
Código	1.3
Actividad	Configuración de Entorno.
Proceso	Procedimiento de pruebas automatizadas en el Departamento de Pruebas de Software.
Responsable	Diseñador de Prueba.
Participantes	Probador. Diseñador de Prueba.

Tabla 8: Procedimiento de la ejecución de las actividades: Configuración de entorno.

❖ **Objetivo**

El sistema contará con los requerimientos mínimos para garantizar un entorno lo más cercano posible al entorno real de la aplicación para obtener un mejor resultado.

❖ **Descripción**

En dicha actividad se configura el ambiente de prueba, separando los ambientes de desarrollo y prueba, para una posterior instalación de las herramientas necesarias y el software a probar en la versión correspondiente.

Requerimientos del Entorno: Presenta los recursos tecnológicos que se requieren para las pruebas.

- Hardware Base: presenta los requerimientos de hardware necesarios para probar el sistema.
- Software Base: presenta los requerimientos de software necesarios para probar el sistema.
- Herramientas: presenta las herramientas necesarias para probar el sistema.

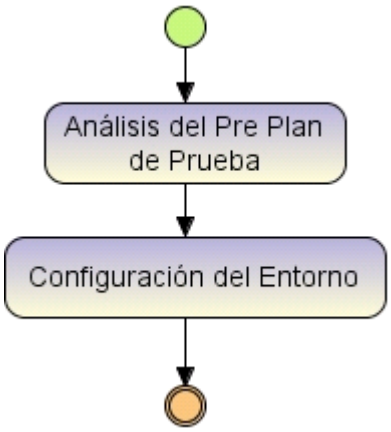

Figura 4: Actividades de la Configuración de Entorno.

2.2.4 Diseño de Prueba

Procedimiento de la ejecución de las actividades	
Código	1.4
Actividad	Diseño de Prueba.
Proceso	Procedimiento de pruebas automatizadas en el Departamento de Pruebas de Software.
Responsable	Diseñador de Prueba.
Participantes	Diseñador de Prueba.

Tabla 9: Procedimiento de la ejecución de las actividades: Diseño de Prueba.

❖ **Objetivo**

El objetivo de esta etapa es diseñar las pruebas que posteriormente van a ser ejecutadas, y los casos de pruebas que se derivan de las especificaciones de producto.

❖ **Descripción**

Esta actividad incluye diseñar e identificar los datos de prueba. Para cada funcionalidad a probar, se crea una matriz que muestra la correspondencia entre el identificador de prueba y el caso de prueba, de forma que se conozca qué caso de prueba cubren qué identificador.

Entre las técnicas posibles de caja negra para pruebas funcionales a utilizar para el diseño de los casos de prueba se encuentran: partición de equivalencia, valor límite, conjetura de errores, grafo causa efecto, partición en categorías, máquinas de estado finitas y escenarios. Actualmente en el Departamento de Pruebas de Software se utilizan las técnicas: partición de equivalencia, valor límite y escenarios, en dicho procedimiento se contará con las mismas.

En esta actividad se especifican los casos de prueba asociados a cada elemento que compone el inventario de pruebas.

Los casos de pruebas incluyen

- Identificar los elementos que deben ser probados.
- Analizar las prioridades de esos elementos dadas por el análisis de los riesgos del producto.
- Desarrollar el diseño de las pruebas de alto nivel para grupos de pruebas relacionados.
- Desarrollar los casos de pruebas individuales a partir del diseño de alto nivel.
- Crear una matriz de cubrimiento que muestre la correspondencia entre las funcionalidades a probar y los casos de pruebas.

Se deben identificar los ciclos funcionales, definiendo el orden en que las pruebas van a ser ejecutadas. Se prueba el ciclo de vida de los datos: crearlo, consultarlo, modificarlo y borrarlo.

Dado que es muy común que existan cambios tardíos en los requerimientos, el proceso de prueba debe funcionar bien con esto.

❖ **Plan de Prueba**

En esta misma etapa se va a obtener el Plan de Prueba, donde se define quién, cuándo, dónde y cómo se realizan las actividades del proceso. Realizar el Plan de Pruebas brinda la posibilidad de recolectar las ideas y cristalizarlas en tareas concretas, puede ser visto como un medio de comunicación entre el equipo de prueba y el cliente.

Un Plan de Prueba incluye

- Alcance del proyecto.
- Calendario.
- Ciclos y las condiciones que deben cumplirse para comenzar cada uno.
- Requerimientos necesarios de hardware y software para ejecutar las pruebas.
- Estrategia de prueba usada durante el proyecto.
- Técnicas usadas para el diseño de las pruebas.
- Roles y sus responsabilidades.
- Procedimientos a seguir para el reporte y seguimiento de No Conformidades.
- Análisis de los riesgos del proyecto de prueba.
- Clasificación de los defectos usados en el proyecto.
- Especificación de qué reportes y otros productos se entregarán al cliente durante el proyecto y al finalizar el mismo.

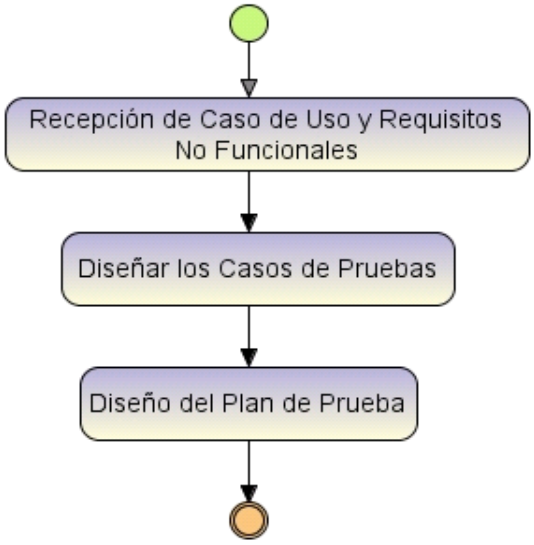

Figura 5: Actividades de Diseño de Prueba

2.2.5 Ejecución de Prueba

Procedimiento de la ejecución de las actividades	
Código	1.5
Actividad	Ejecución de Prueba
Proceso	Procedimiento de pruebas automatizadas en el Departamento de Pruebas de Software.
Responsable	Probador
Participantes	Probador

Tabla 10: Procedimiento de la ejecución de las actividades: Ejecución de Prueba.

❖ **Objetivo**

Su objetivo es contrastar el comportamiento esperado del software con su comportamiento real, analizar las diferencias y reportar los resultados.

❖ **Descripción**

Esta actividad incluye al probador seleccionar los casos de pruebas, los ejecuta, registra los resultados y determina si las No Conformidades encontradas son causadas por errores en el producto o errores en las pruebas. La ejecución de las pruebas se realiza automáticamente. Cuando se ejecuta un caso de prueba y el resultado no se corresponde con el real, el probador debe:

- Volver a ejecutar el caso de prueba, al menos una vez, para asegurarse que se trata de una No Conformidad.
- Ejercitar otros escenarios y otros datos, para poder aislar el error.

En dicha actividad se ejecutan las pruebas con las herramientas que anteriormente fueron instaladas, las cuales fueron definidas por el especialista. Dentro de las herramientas necesarias para realizar las pruebas se encuentran: sistemas operativos, herramientas para la automatización de las pruebas, manejadores de bases de datos, herramientas para la gestión de las pruebas, herramientas para la realización de reportes, sistema de gestión de No Conformidades, herramientas para la gestión de la configuración. Además debe instalarse la versión del producto a probar. Se obtienen los documentos de las No Conformidades.

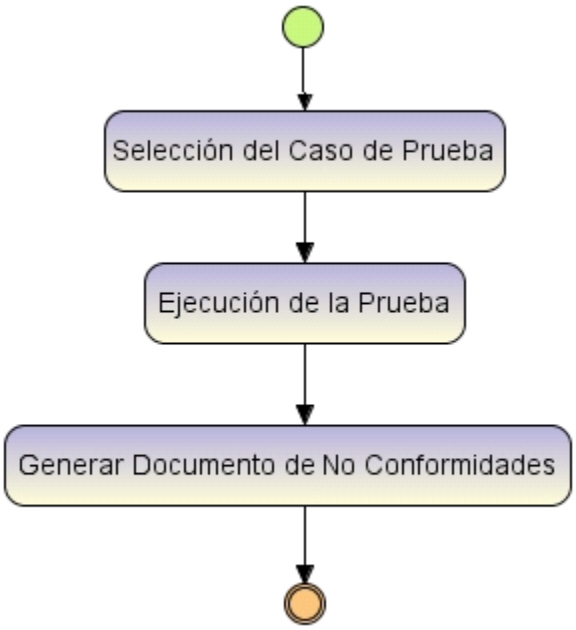

Figura 6: Actividades de Ejecución de las Pruebas.

2.2.6 Gestión de las No Conformidades

Procedimiento de la ejecución de las actividades	
Código	1.6
Actividad	Gestión de las No Conformidades
Proceso	Procedimiento de pruebas automatizadas en el Departamento de Pruebas de Software.
Responsable	Probador
Participantes	Probador

Tabla 11: Procedimiento de la ejecución de las actividades: Gestión de las No Conformidades.

❖ **Objetivo**

Se gestionan todas las No Conformidades existentes en el producto, realizando el documento final de las mismas.

❖ **Descripción**

Después que el probador haya ejecutado las pruebas con las herramientas seleccionadas, se van a generar posibles No Conformidades. Una vez encontrados todos los errores en el software estas se reportan. Cada No Conformidad debe referenciar que caso de prueba lo originó, debe poder reproducirse y estar acompañado de una valoración del impacto del mismo. Luego se gestionan estos errores, creándose un documento final de las No Conformidades, clasificándolas, es decir si es significativa o no significativa

Los elementos mínimos a documentar para cada No Conformidad son:

- Identificador: identificador único.
- Caso de prueba: identificador del caso de prueba donde se encontró la No Conformidad.
- Resumen: breve descripción de la No Conformidad.
- Pasos: pasos necesarios para la reproducción de la No Conformidad.
- Versión: versión del producto que se está probando.
- Prioridad: importancia de la No Conformidad encontrada, esta puede ser:
  - Crítica: la ejecución del sistema es interrumpida, no se puede seguir con la prueba.
  - Alta: la aplicación sigue funcionado pero el problema tiene un alto impacto.
  - Media: el problema tiene impacto medio en la aplicación.
  - Baja: el problema tiene poco impacto en la aplicación.
- Categoría: categoría a la que corresponde la No Conformidad.
- Probador: nombre del probador que reporta la No Conformidad.

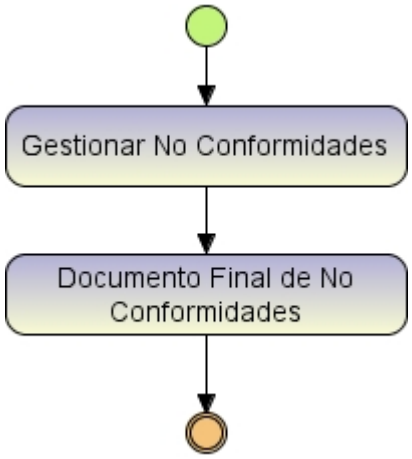

Figura 7: Actividades de la Gestión de las No Conformidades.

2.2.7 Evaluación de las Pruebas

Procedimiento de la ejecución de las actividades	
Código	1.7
Actividad	Evaluación de las Pruebas.
Proceso	Procedimiento de pruebas automatizadas en el Departamento de Pruebas de Software.
Responsable	Especialista de Prueba
Participantes	Especialista de Prueba Cliente

Tabla 12: Procedimiento de la ejecución de las actividades: Evaluación de las Pruebas.

❖ **Objetivo**

Esta actividad tiene como objetivo evaluar el proceso de prueba, logrando un criterio general del resultado de las mismas.

❖ **Descripción**

Al finalizar el proceso se debe conocer el resultado obtenido, la comunicación que se tuvo durante el proyecto, y su percepción del avance del proyecto. Estos elementos ayudan a ajustar y mejorar el proceso de prueba.

En esta etapa se genera el Informe de Evaluación de las Pruebas que no es más que indicar la evaluación que se le da al producto, así como lo referente a las No Conformidades y su solución, aplicando el Criterio de criticidad[1], este establece los parámetros para declarar un producto del desarrollo de software en estado crítico de terminación, o sea si se cumplen algunos de estos criterios se declara fallida la prueba.

La evaluación de las pruebas cubre los siguientes tres puntos:

- Evaluación del cubrimiento: evaluar los casos de pruebas según el cubrimiento de funcionalidades.
- Evaluación de las No Conformidades del producto: evaluar la calidad del producto respecto a la ejecución de las pruebas realizadas.
- Evaluación de la efectividad de las pruebas: evaluar las pruebas respecto al Criterio de completitud.

❖ **Informe de Evaluación de las Pruebas**

En el Informe de Evaluación de las Pruebas, se cuenta con los siguientes aspectos:

- **Introducción**
- **Objetivos**
- **Alcance**
- **Definiciones, Acrónimos y Abreviaturas**
- **Responsabilidades**
- **Datos del producto**
- **Detalle de los elementos probados y su estado final**

Artefactos	Estado Final
Nombre del artefacto	

Tabla 13: Artefactos y estado final.

[1] Ver Anexo #1

- **Cantidad de iteraciones**

Cantidad de Iteraciones	

Tabla 14: Cantidad de iteraciones.

- **Elementos revisados o probados y herramientas utilizadas**

Elementos	**Herramienta**
Nombre del elemento	Nombre de la herramienta

Tabla 15: Elementos revisados.

- **Cantidad de horas empleadas y rango de fecha**

Cantidad de horas	Rango de fecha
Cantidad de horas empleadas	Rango de fecha

Tabla 16: Cantidad de horas empleadas.

- **Estructura del equipo de prueba empleado y turnos de trabajo**

Equipo de Prueba	**Turnos**
Nombre de los roles	Turnos de Trabajo

Tabla 17: Estructura del equipo de prueba.

- **Resumen de las No Conformidades por entidades o persona responsable**
- **Análisis de los resultados**

Resultados	**Evaluación**	**Aspectos Significativos**

Tabla 18: Análisis de los resultados.

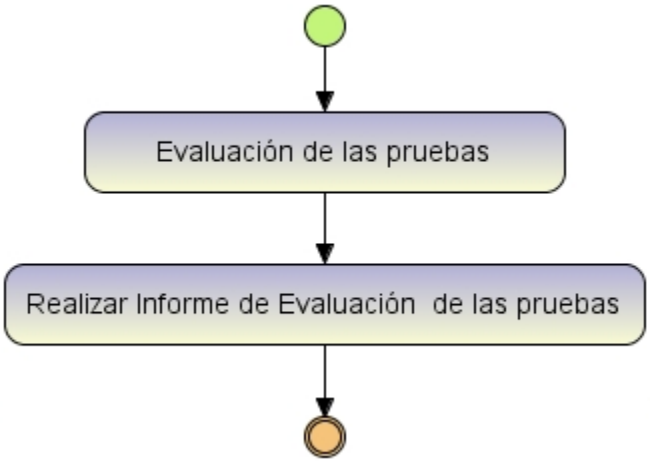

Figura 8: Actividades de Evaluación de Prueba.

2.3 Conclusiones

Durante el transcurso del capítulo se desarrolló la propuesta del procedimiento de las pruebas automatizadas para el Departamento de Pruebas de Software, para un posterior resultado en la utilización del mismo. Dicho procedimiento puede ser utilizado no solamente por los especialistas del centro de Calisoft sino también por los probadores de las diferentes facultades. Además no será utilizado para realizar pruebas únicamente a proyectos de la Universidad, sino también a otros proyectos que pertenecen a otras instituciones como Desoft, Softel entre otros.

Capítulo 3
Evaluación del Procedimiento

3. Introducción

Realizar pruebas a un sistema informático no constituye una tarea fácil, para ello es necesario un gran esfuerzo por parte de los responsables de la planificación y ejecución, además del tiempo invertido. A pesar de todo, es un punto de obligatorio cumplimiento para obtener buenos resultados. La utilización de herramientas para la automatización de pruebas es una tarea útil para economizar tiempo.

A partir de las necesidades de gestionar los procesos de pruebas con herramientas automatizadas, se creó un procedimiento para gestionar estas pruebas de forma eficiente. Se propone como ejemplos de validación utilizar las herramientas: Selenium IDE [2] para pruebas funcionales, JMeter[3] para pruebas de carga y estrés, Xenu Link Sleuth [4] para pruebas de estructura.

3.1 Proceso de selección de las herramientas

En el proceso de selección, se evalúan cuáles son las herramientas apropiadas para las necesidades del equipo de prueba. Para que el proceso tenga éxito no se puede comenzar la búsqueda centrándose en las herramientas que existen, si no que se deben identificar los requerimientos, características del grupo de prueba, metodología de desarrollo, proceso de prueba empleado, lenguajes y otros aspectos individuales de cada grupo que hacen que este tenga necesidades únicas. Una mala elección de las herramientas tendría un efecto indeseable, como hacer del proceso de prueba un período muy lento, tedioso y con menos rendimiento de lo esperado, lo que impactaría de forma significativa a la hora de realizar las pruebas al producto. Estos posibles aspectos negativos obstaculizan el correcto funcionamiento de dichas pruebas. Por lo que es necesario saber escoger la herramienta adecuada, donde el especialista debe contar con una excelente preparación con respecto a la misma, para luego dar una capacitación requerida a los respectivos probadores que posteriormente harán uso de la herramienta seleccionada.

[2] http://seleniumhq.org/projects/ide
[3] http://jakarta.apache.org/jmeter/
[4] http://home.snafu.de/tilman/xenulink.html

3.2 Gestión de pruebas automatizadas con: Selenium IDE

❖ **Descripción general**

Se propone utilizar Selenium IDE por ser un plugin del Mozilla Firefox que pertenece al juego de herramientas SeleniumHQ. Es un entorno de desarrollo integrado, ofreciendo una interfaz fácil de usar, posibilitando ejecutar casos de pruebas individuales o suites de pruebas completas para la creación de casos de prueba. Este permite realizar juegos de pruebas sobre aplicaciones web. Para ello realiza la grabación de la acción seleccionada (navegación por una página) en un script, el cual se puede editar y parametrizar para adaptarse a los diferentes casos, y lo que es más importante, su ejecución se puede repetir tantas veces como se quiera.

3.3 Gestión de pruebas automatizadas con: JMeter

❖ **Descripción general**

Se decide utilizar JMeter, porque permite realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones web, permitiendo bajar de forma gratuita el software desde la web que representa al proyecto Jakarta. Existiendo una amplia documentación sobre la utilización del modo de empleo de la herramienta y los requerimientos no funcionales que precisa ella. Se puede utilizar muchas de sus funcionalidades para probar aspectos de las pruebas de disponibilidad y red. Además se clasifica por su especialización en una herramienta de grabación y reproducción.

3.4 Gestión de pruebas automatizadas con: Xenu Link Sleuth

❖ **Descripción general**

Dicha herramienta analiza una aplicación web con la URL que se le indique buscando errores en los enlaces, genera un completo informe en formato HTML con información diversa acerca de los vínculos rotos ordenados por páginas y enlaces, la lista de URL redireccionadas y el mapa del sitio web en función de los títulos de las páginas.

3.5 Validación de actividades del procedimiento

3.5.1 Actividad # 1: Estudio Preliminar

❖ **Aspectos significativos que se tienen en cuenta en el Estudio Preliminar**

El especialista de prueba comenzó estudiando las principales funcionalidades del producto, analizando cada una de las posibles soluciones que pudieran tener; se recomienda que todas las observaciones realizadas sean informadas por escrito. Cuando la aplicación seleccionada es muy grande y tiene muchas

funcionalidades, el especialista de prueba debe escoger dos o más personas para que juntos logren hacer un estudio detallado del producto. Según el avance en el estudio de la aplicación y de sus funcionalidades se determina si se continúa con el estudio o si ya es suficiente para comenzar con el siguiente paso. Este conocimiento previo permite determinar si se puede dar solución a las funcionalidades básicas de la aplicación, y continuar así con todo el trabajo; de no quedar satisfechos se detendrá el estudio hasta dejar bien definido qué se quiere, y cómo se dará solución. Para lograr una mayor comprensión entre las partes, el especialista se reúne con el cliente donde se explica la aplicación por completo y sus principales funcionalidades. Después de ponerse al tanto con las especificaciones del cliente sobre el producto, se determina si es posible generar los casos de pruebas a partir de dichas especificaciones, en caso que no exista o de que estén incompletas, se deben definir junto con el cliente la forma en que serán generadas o mejoradas. Se considera importante concluir la reunión cuando ambas partes estén puestas de acuerdo, así como que todos los temas principales queden esclarecidos para el especialista de prueba que debe obtener una lista de funcionalidades priorizadas según su importancia. Durante el Estudio Preliminar el especialista de prueba conformará el Pre Plan de Prueba el cual constituye una primera versión del documento final, sometiéndose a cambios durante la etapa inicial del procedimiento hasta llegar a conformar el Plan de Prueba.

- **Validación del Estudio Preliminar**

El especialista realizó el estudio de las principales funcionalidades del sitio web http://calisoft.uci.cu/. Esta aplicación web no se considera compleja, pues no tiene un número significativo de vínculos asociados a ella, ni cuenta con muchas interfaces; por lo que se decide utilizar un solo especialista para realizar el estudio. El especialista se reunió con el cliente, en este caso el cliente es del mismo proyecto; ambas partes puestas de acuerdo seleccionaron las principales funcionalidades del producto y sus requerimientos, quedando todo bien especificado tanto para el especialista como para el cliente. Se conformó el Pre Plan de Prueba teniendo en cuenta todos los acuerdos tomados. (Ver Cap.3, epígrafe 3.5.4, Plan de Prueba)

3.5.2 Actividad # 2: Planificación

- **Aspectos significativos que se tuvieron en cuenta en la Planificación**

Teniendo en cuenta los resultados del Estudio Preliminar realizado, se comenzó a planificar el proceso de prueba, donde se tuvo en cuenta los recursos humanos y materiales para el desarrollo de la prueba. El cliente hizo entrega de los requisitos funcionales y los requisitos no funcionales al especialista, que fue el responsable de asegurar la planificación apropiada. Se seleccionaron tres roles principales con responsabilidades definidas: Especialista de Prueba, Diseñador de Pruebas y Probador. (Ver Cap. 2 epígrafe 2.2.2 Planificación).

El especialista vela por el progreso y efectividad del proceso de prueba, distribuyendo los recursos seleccionados para las pruebas y finalmente evaluando los resultados obtenidos. El diseñador debe identificar, describir, seleccionar y priorizar los casos de pruebas. Además de identificar las técnicas apropiadas para realizar dicho proceso. El probador es encargado de ejecutar las pruebas, y finalmente según los resultados se conforma la documentación de las No Conformidades.

Los recursos humanos pueden variar en correspondencia de las distintas situaciones posibles, pero los roles y sus responsabilidades no, por lo que es necesario que en cada momento se cumpla la planificación definida. Los recursos materiales con los que se realizan las pruebas no están predeterminados, ni definidas sus propiedades, sino que dependiendo de la organización y el lugar donde se realizan las pruebas, estos tienen que cumplir con los requerimientos mínimos para dar respuesta. Al final se debería actualizar el Pre Plan de Prueba teniendo en cuenta todas las actividades realizadas.

- **Validación de la Planificación**

Teniendo en cuenta el Estudio Preliminar realizado comienza la planificación del proceso de prueba. Primeramente se hizo recepción de dos documentos entregados al especialista de prueba:

- Documento de los requisitos funcionales.
- Documento de los requisitos no funcionales.

Se consideró que la complejidad de la aplicación a la cual se le realiza las pruebas no es alta, por lo que se utilizó solamente el mínimo de recursos humanos para dar respuesta.

Recursos Humanos	Cantidad
Especialista de Prueba	1
Diseñador de Prueba	1
Probador	2

Tabla 19: Recursos humanos validados.

- **Recursos materiales utilizados para el proceso de prueba.**

Recursos tecnológicos	Descripción
PC	✓ Pentium 4. ✓ 1 GB de RAM.

	✓ 256 de memoria de video interna.
Sistema Operativo	✓ Windows.

Tabla 20: Recursos tecnológicos validados.

En este caso la plataforma que se utiliza para realizar las pruebas es Windows, la herramienta seleccionada es el Selenium IDE. Se actualiza el Pre Plan de Prueba con todos los nuevos datos brindados por el cliente y por el equipo de trabajo.

3.5.3 Actividad # 3: Configuración del Entorno

- **Aspectos significativos que se tuvieron en cuenta en la Configuración del Entorno**

El software es una simulación de la realidad, donde a través de una buena configuración se logra que los resultados obtenidos sean los más fieles posibles. Para la realización de las pruebas se hace necesario crear un ambiente cercano a la realidad. Se debe tener en cuenta todas las características necesarias del hardware para la instalación de las herramientas seleccionadas. El software debe de estar en buen estado, preferentemente que no sea un demo y que tenga todas las funcionalidades que dice tener. Las computadoras utilizadas para el proceso de prueba deben contar con las propiedades mínimas para que el software funcione correctamente. El diseñador junto al probador deben pretender lograr resultados reales, por lo que se debe utilizar la herramienta correcta en el entorno correcto.

- **Validación de la Configuración del Entorno**

La computadora donde se ejecutaron las pruebas es verificada, cumpliendo con las propiedades mínimas para la instalación de la herramienta. La aplicación web que se utilizó se ejecuta rápidamente, posibilitando un buen rendimiento. En esta etapa se comprobó que el hardware y el software instalado están en buen estado, instalándose el Mozilla Firefox versión por encima de 1.5, en este caso se instaló la versión 3.0.3, el cual es uno de los requerimientos mínimos para que la herramienta seleccionada funcione.

Requerimientos técnicos	Descripción
Navegador	Mozilla Firefox 3.0.3.
Hardware	V
Software	V

Tabla 21: Requerimientos técnicos.

Leyenda: V (Válido, en buen estado).

3.5.4 Actividad # 4 Diseño de Prueba

❖ **Aspectos significativos que se tienen en cuenta en el Diseño de Prueba**

El cliente hace entrega de los casos de uso, los cuales serán utilizados para realizar cada una de las pruebas funcionales y entregar los requisitos no funcionales, y serán analizados a profundidad aquellos que están relacionados con la base de datos, para luego realizar pruebas en base a los requisitos especificados. Se identifican los datos de prueba para cada funcionalidad a probar. El diseñador tiene la responsabilidad de diseñar las pruebas que posteriormente van a ser ejecutadas y los casos de pruebas que se derivan de las especificaciones del producto. El diseñador realiza un estudio de las condiciones específicas que tiene el producto, luego de tener toda la información requerida se realiza el Plan de Prueba.

❖ **Validación del Diseño de Prueba**

Teniendo en cuenta los datos de prueba dados en la especificación del producto. Se diseñaron los casos de pruebas que fueron utilizados durante el proceso de prueba. Actualmente en el Departamento de Pruebas de Software se tiene el siguiente formato:

❖ **Diseño de Caso de Prueba: Autenticar Usuario**

- **Introducción**

Se le aplica el procedimiento con pruebas funcionales a la aplicación web de Calisoft.

- **Alcance**

Aplicación Web de Calisoft.

- **Descripción general**

El caso de prueba se realiza con el caso de uso Autenticar Usuario.

- **Condiciones de ejecución**

El caso de uso se divide en varias secciones.

- **Secciones**

Nombre de la sección	**Escenario**	**Descripción de la funcionalidad**	**Flujo Central**
Autenticar usuario	1.Autenticar	El usuario entra sus datos correctamente en la aplicación.	1 El usuario llena los campos. 2 Acepta
	2.Datos Incorrectos	El usuario entra los datos incorrectamente.	1 El usuario llena los

			2 Acepta
	3.Campos Vacios	El usuario deja campos vacíos.	1 El usuario llena los campos 2 Acepta

Tabla 22: Secciones.

- **Matriz de Datos**

ID Escenario	Escenario	Usuario	Contraseña	Respuesta del sistema	Resultado de la prueba
1	Autenticar usuario	*V* Administrador	*V* Administrador	El sistema comprueba los datos (usuario y contraseña) si son correctos.	Satisfactorio
2	Autenticar usuario	*I* Admin	*V* Administrador	El sistema muestra un mensaje indicando que los datos entrados son incorrectos	Satisfactorio
2	Autenticar usuario	*V* Administrador	*I* Admin	El sistema muestra un mensaje indicando que los datos entrados son incorrectos	Satisfactorio
3	Autenticar usuario	*NA* (Vacío)	*V* Administrador	El sistema muestra un mensaje indicando que el campo del nombre del usuario es obligatorio llenarlo.	Satisfactorio
3	Autenticar usuario	*V* Administrador	*NA* (Vacío)	El sistema muestra un mensaje indicando que el campo de la contraseña del usuario es obligatorio llenarlo.	Satisfactorio

Tabla 23: Matriz de Datos.

Las celdas de la tabla contienen V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante.

- **Registro de defectos y dificultades registradas**

Elemento	No.	No Conformidad	Aspecto Correspondiente	Etapa de detección	Significativa	No Significativa	Recomendación
Aplicación Web	1	El campo del usuario se encuentra vacío.	La No Conformidad se encontró en la Interfaz.	Prueba funcional	x		
	2	El campo del contraseña se encuentra vacío.	La No Conformidad se encontró en la Interfaz.	Prueba funcional	x		

Tabla 24: Registro de defectos y dificultades registradas.

❖ **Plantilla para el diseño del Plan de Prueba**

- **1 Introducción**

En la siguiente plantilla se realizará una previa planificación del proceso de prueba teniendo en cuenta el procedimiento en general.

- **1.1 Alcance**

Se realiza la planificación del procedimiento que es aplicado al sitio web del centro de Calisoft.

- **1.2 Definiciones, Acrónimos y Abreviaturas**

- **1.3 Referencias**

[Lista de documentos a los que se hace referencia]

Código	Titulo
[1]	Documento 1

Tabla 25: Referencias.

- **2 Roles y responsabilidades**

Rol	Cantidad	Responsabilidad	Nombres
Probador	2	✓ Verificar la correcta instalación de la	✓ Ariannis Marzán Matos.

		herramienta: Selenium IDE. ✓ Verificar el entorno de prueba. ✓ Ejecución de Prueba. ✓ Gestionar las No Conformidades.	✓ Juan Carlos Villar Bravo.

Tabla 26: Descripción del equipo de probadores.

- **3 Escenario de pruebas.**

Los escenarios generados se encuentran en la plantilla del diseño de caso de prueba para pruebas funcionales.

- **4 PC Clientes**

PC Clientes	
Cantidad	1
Descripción	1 Gb de RAM. 256 Mb de Memoria interna.
Software base	Windows.

Tabla 27: Recursos del Sistema: PC Clientes.

- **5 Requerimientos a probar**

[Listado de requerimientos a probar. Puede hacerse referencia al documento de requerimientos]

- **6 Estrategia de pruebas**

El flujo de trabajo se inicia cuando el especialista realiza un estudio de las condiciones específicas que tiene el producto, luego de tener toda la información requerida se realiza un Pre Plan de Prueba en el cual se tiene presente el expediente de proyecto, llegando a una posterior planificación y luego a un diseño previo de los casos de pruebas y el Plan de Prueba en su estado general por parte del diseñador. Creadas las condiciones se empiezan a ejecutar las primeras iteraciones de las pruebas, en caso de que se encuentren errores, se puntualizan formando un documento de las No Conformidades y luego pasan a ser revisadas por el especialista. Al concluir las iteraciones de las pruebas se realizará el Informe Final de Evaluación. Una vez aprobado es entregado al líder del proyecto para que este comience a ejecutar los arreglos acordados.

- **7 Cronograma**

En el cronograma de trabajo de la plantilla del Plan de Prueba, se cuenta con los identificadores de cada una de las tareas del procedimiento, recogiendo las fechas en que se realizaron dichas actividades, es decir el período de duración de cada una de estas; emitiendo los responsables y participantes de dicho proceso, expresando las observaciones correspondientes a las actividades. (Ver Anexo # 19)

3.5.5 Actividad # 6: Ejecución de Prueba

- **Aspectos significativos que se tienen en cuenta en la Ejecución de Prueba**

En esta actividad se compara el comportamiento esperado del software con su comportamiento real, donde se analizan las diferencias y se reportan los posibles resultados. Se verifica que la herramienta seleccionada ya se encuentra instalada para realizar las pruebas. El probador selecciona los casos de pruebas, los ejecuta, registra los resultados y determina los tipos de No Conformidades encontradas. Todo este proceso da como resultado el Documento Final de las No Conformidades donde se especifica el tratamiento para cada una de las No Conformidades encontradas.

- **Validación de Ejecución de las Pruebas**

Se verificó que el Selenium IDE versión 1.0-beta-2 estuviera instalado correctamente, se trabajó con el caso de prueba propuesto Autenticar Usuario, se ejecutaron las pruebas y se obtuvieron las No Conformidades.

3.5.6 Actividad # 6: Gestión de las No Conformidades.

- **Aspectos significativos que se tienen en cuenta en la Gestión de las No Conformidades**

Una vez realizadas las pruebas se debe gestionar todas las No Conformidades teniendo en cuenta cual fue el caso de prueba que la originó para así poder tener referencia y darle tratamiento. El Documento de las No Conformidades es estudiado detalladamente, se valora el impacto de cada una de ellas, se gestionan los posibles errores encontrados, y se clasifican, se documenta cada uno de los pasos realizados dando como resultado la respuesta del equipo de desarrollo. En caso de no poder dar solución a las No Conformidades encontradas estas pasarán a un equipo de desarrollo más grande.

- **Validación de la Gestión de las No Conformidades**

Después de realizadas las pruebas, se obtuvo una lista de No Conformidades la cual se organizó por su impacto dentro del producto. Detectándose los casos de pruebas que las originan. En el Documento Final se recogen todas las No Conformidades que no tuvieron solución con el tratamiento dado por los probadores. Actualmente se utiliza la plantilla de No Conformidades del Departamento de Pruebas de Software de Calisoft.

❖ **Planilla de las No Conformidades**

- **Descripción general**

Las No Conformidades detectadas son tratadas en este documento.

- El campo usuario y contraseña no pueden estar vacio.
- El campo usuario y contraseña no pueden ser incorrectos.

- **Elementos probados**

Aplicación Web del centro de Calisoft, específicamente el caso de uso Autenticar Usuario.

- **Registro de defectos y dificultades detectados**

Elemento	No.	No Conformidad	Aspecto correspondiente	Etapa de detección	Clasificación	Estado NC	Respuesta del Equipo Desarrollo
Aplicación Web.	1	Insertar datos Inválidos.	El usuario insertó los datos incorrectamente en la interfaz de usuario.	Prueba funcional	Significativa	9/4/2010. PD: Pendiente	
	2	Dejar campos vacios	El usuario deje los campos vacíos.	Prueba funcional	Significativa	9/4/2010. PD: Pendiente	

Tabla 28: Registro de defectos.

3.5.7 Actividad # 7: Evaluación de las Pruebas

❖ **Aspectos significativos que se tienen en cuenta en la Evaluación de las Pruebas**

La evaluación de cada una de las pruebas aplicadas al sitio web http://calisoft.uci.cu/ son realizadas en esta parte. Estas son de gran importancia para el proyecto ya que brinda un conocimiento cuantificado de los resultados obtenidos. Se validó la utilización del Plan de Prueba que es diseñado con anterioridad en la actividad Diseño de Prueba, se tiene en cuenta el Criterio de criticidad para evaluar cada una de las pruebas. El especialista como responsable de la evaluación de las pruebas debe emitir un criterio general de los resultados obtenidos, plasmándolos en el Informe de Evaluación.

- **Validación de la Evaluación de las Pruebas**

Se tienen en cuenta tres aspectos esenciales para la evaluación de las pruebas. (Ver Capítulo 2 epígrafe 2.2.7 Evaluación).

En este caso se utilizó un solo caso de prueba, Autenticar Usuario. No es necesario utilizar varios casos de pruebas para validar el procedimiento para pruebas funcionales, pues la evaluación no es para las herramientas sino para el procedimiento propuesto. Es por eso que se toma un caso de prueba genérico para mostrar el uso del procedimiento.

Se evalúan tres escenarios diferentes para un mismo caso de prueba, brindando conocimiento de la funcionalidad Autenticar Usuario. Estos escenarios son los siguientes:

Casos	Usuario	Contraseña
1	Correcto	Correcto
2	Vacío	Correcto
3	Correcto	Vacío

Tabla 29: Escenario del Autenticar Usuario.

La ejecución de las pruebas realizadas está descrita en el Informe de Evaluación de las Pruebas, allí se detalla cómo no se cumplió el Criterio de criticidad por lo que la prueba no fue fallida. Los resultados esperados fueron iguales a los resultados obtenidos lo que da a entender que la prueba seleccionada para este caso tuvo éxito. Se cumplió el Criterio de completitud y se decide parar y no realizar más pruebas.

- **Validación del Informe de Evaluación de las Pruebas**

En el Informe de Evaluación de las Pruebas se cuenta con los siguientes aspectos:

- **Introducción**
- **Objetivos**

Este Informe de Evaluación, tiene como meta evaluar la aceptación por parte de los usuarios, además de trasmitir un criterio útil para la toma de decisiones.

- **Alcance**

Aplicación Web de Calisoft.

- **Definiciones, Acrónimos y Abreviaturas**

- **Responsabilidades**

Ariannis Marzán Matos: Diseñador de Prueba.

Juan Carlos Villar Bravo: Probador.

- **Datos del producto**

Clasificado como: Aplicación Web.

- **Detalle de los elementos probados y su estado final:**

Artefactos	**Estado Final**
Aplicación Web (caso de uso: Autenticar Usuario)	2 No Conformidades.

Tabla 30: Artefactos y Estado validados.

- **Cantidad de iteraciones**

Cantidad de Iteraciones	1

Tabla 31: Cantidad de Iteraciones validados.

- **Elementos revisados o probados y herramientas utilizadas**

Elementos	**Herramienta**
Aplicación Web: http://calisoft.uci.cu	Selenium-ide-1.0-beta-2.

Tabla 32: Elementos revisados validados.

- **Cantidad de horas empleadas y rango de fecha**

Cantidad de horas	**Rango de fecha**
5hrs	15-18/4/2010

Tabla 33: Cantidad de horas empleadas validadas.

- **Estructura del equipo de prueba empleado y turnos de trabajo**

Equipo de Prueba	**Turnos**
Especialista de Prueba	5hrs
Diseñador de Prueba	5hrs
Probador de Prueba	5hrs

Tabla 34: Estructura del equipo de prueba validado.

- **Resumen de las No Conformidades por entidades o persona responsable**

Se analizaron cada uno de los resultados obtenidos, según las combinaciones del caso de prueba, así como los avances alcanzados, según el criterio evaluativo se trabajaron en cada uno de los casos, se especificaron cuántas No Conformidades fueron detectadas. En este caso de prueba se detectaron solamente 2 No Conformidades, los responsables de solucionar cada una de las No Conformidades, y en caso de no poder darle respuesta se pasarán para un grupo de trabajo donde se tratarán cada una de ellas.

- **Análisis de los resultados**

Los resultados obtenidos son evaluados de Bien, pues son iguales a los resultados esperados.

Resultados	Evaluación	Aspectos Significativos
Se realizaron todas las posibles combinaciones de los escenarios, comprobándose y mostrándose los resultados esperados	Bien	Se realizó todos los escenarios correspondientes al caso de uso Autenticar Usuario satisfactoriamente.

Tabla 35: Análisis de los resultados validados.

- **Conclusiones del Informe de Evaluación de las Pruebas.**

En el transcurso del proceso de prueba, utilizando el procedimiento con ayuda de la herramienta automatizada del Selenium IDE, se obtuvieron resultados reales y significativos, detectándose las No Conformidades correspondientes, evaluando todas las posibles combinaciones que podía tener el caso de uso correspondiente, obteniendo buenos resultados durante el proceso en general.

3.6 Funcionalidades del Selenium IDE

Para visualizar la herramienta Selenium IDE, solamente hay que seleccionar el plugin en el menú del Mozilla Firefox, se muestra el Selenium IDE, con una secuencia de comandos. En este caso la aplicación que se utilizó para realizarle pruebas funcionales y validar la propuesta de solución fue el sitio web: http://calisoft.uci.cu, se escogió esta aplicación por interés de los especialistas del centro, además de ser un producto que se le realizan pruebas de liberación en el Departamento de Pruebas de Software, dicha aplicación incluye funcionalidades para probar. (Ver Anexo # 4)

Para realizarles pruebas a una aplicación determinada sería necesario introducir la dirección URL de dicha aplicación en la Base de URL del Selenium IDE. (Ver Anexo # 5)

Se activa el botón de grabación y luego con los casos de pruebas que se tienen presente se realiza la navegación por el sitio web. Se puede crear un caso de prueba o simplemente una suite de prueba que no es más que varios casos de pruebas dentro de esta suite. Estos van a verificar la funcionalidad del software. (Ver Anexo # 6)

Caso de Prueba 1: Autenticar Usuario.

En este caso de prueba, se probó el caso de uso autenticar usuario, donde se definieron tres posibles combinaciones, primeramente se entró el usuario y la contraseña de forma que se han validos. Luego en las otras combinaciones se cambiaron los valores, por ejemplo en la segunda combinación el usuario está vacío y la contraseña sea válida y posteriormente en la tercera combinación, el usuario sea válido y la contraseña esté vacía, para luego observar las No Conformidades que se generan.

A partir de crear las condiciones especificas para empezar la grabación de la prueba, se introduce la primera combinación (usuario y contraseña sean válidos). (Ver Anexo # 7)

La segunda combinación (usuario (vacío) y contraseña (correcta)) (Ver Anexo # 8). El campo inválido del usuario que se observa en la herramienta se le puede cambiar su valor y poner correctamente si lo deseas los datos correctos en el campo Value. Estas pruebas se pueden guardar, la herramienta de pruebas funcionales Selenium IDE posee en su archivo la opción de Guardar Caso de Prueba (Save Test Case).

La tercera combinación (usuario (correcto) y contraseña (vacía)). (Ver Anexo # 9)

La herramienta Selenium IDE va ejecutando comando por comando, recargando la página, y escribiendo en los cuadros de textos los datos que le indicamos. En el panel de comandos se irán marcando en verde llevando un registro de forma visual de los sucesos, teniendo en cuenta el orden de los eventos los que se ejecutaron correctamente y en rojo los que no. A su vez en la parte de Log (abajo) aparecerán los mismos pasos pero más detallados. En este caso al final de la prueba el sistema mostrará un mensaje de error que vendría siendo las No Conformidades de las pruebas.

3.7 Funcionalidades del JMeter

Lo primero que se debe hacer para trabajar en esta herramienta es realizar la instalación de la Máquina Virtual Java 1.3 o superior instalada en el sistema operativo, se necesita que la aplicación web a la cual se le va a realizar las pruebas esté funcionando correctamente. Luego se graban los escenarios de prueba (Ver Anexo # 10) y se documentan cada uno ellos en la plantilla de diseño de casos de pruebas de la aplicación web http://calisoft.uci.cu. (Ver Anexo # 3).

Luego de grabar todos los escenarios, se ejecuta la herramienta en este caso se prueba con 50 usuarios conectados concurrentemente. La prueba realizada arrojó algunos errores, uno de los más frecuentes es el código 404 el cual nos dice que la petición realizada no fue encontrada en el servidor (Ver Anexo # 11).Se demostró que con 50 usuarios conectados concurrentemente no es factible, el análisis de estas pruebas mostró como resultados que la interfaz index.php fue solicitada cada una de las veces que se realizaban

peticiones a páginas que tienen vínculos dentro de esta. Por lo que el servidor colapsó emitiendo un mensaje de error. (Ver Anexo # 12)

Los resultados finales de la prueba realizada usando la herramienta JMeter a la aplicación web http://calisoft.uci.cu fueron satisfactorios, ya que se logró cargar y estresar el sistema hasta su punto máximo, emitiendo los siguientes resultados. (Ver Anexo # 13)

En el informe agregado de los resultados se muestran unas series de datos los cuales exponen el estado en el que se encuentra en software con respecto a los módulos probados.

- Muestras: cantidad de páginas (Hilos) que simulan la cantidad de usuarios. Que están interactuando con el sistema desde la misma URL.
- Media: media de páginas que se cargaron de manera satisfactoria.
- Mediana: tiempo promedio que han tardado en cargarse las páginas.
- Min: tiempo mínimo que ha demorado en cargarse una página.
- Max: tiempo máximo que ha tardado en cargarse una página.
- Línea 90 %: 90 por ciento de las páginas que se cargaron de manera satisfactoria.
- %Error: por ciento de error de las páginas que no se llegaron a cargar de manera satisfactoria.
- Kb/Seg: velocidad de carga de las páginas.
- Tiempos de Respuestas: total del tiempo que demoró en cargarse la cantidad de hilos de esa prueba.

3.8 Funcionalidades del Xenu Link Sleuth

Xenu Link Sleuth funciona al actuar con un navegador web bien acabado, es decir que todos sus links estén correctos, carga documentos mediante los protocolos HTTP web y / o HTTPS en este caso utilizamos la aplicación web http://calisoft.uci.cu. A continuación, pasa a comprobar los vínculos internos, especificados desde un inicio según el mapa del sitio. (Ver Anexo # 14)

El Xenu tiene como ventaja que genera un reporte detallado de cada uno de los link probados, dónde se especifica si la dirección URL se encuentra disponible y si no muestra el tipo de error por el cual no se pudo acceder. (Ver Anexo # 15)

Estos son los tipos de errores que el Xenu puede llegar a detectar, en este caso muestra la cantidad de URL con problemas y devuelve en por ciento la cantidad de errores. (Ver Anexo # 16)

Este es el informe final donde se especifican los tipos de archivos que fueron comprobados, la cantidad de URL y su por ciento, además de la sumatoria del tamaño de cada uno de ellos en Bytes y en KB. Muestra el por ciento del total y da un rango de tiempo mínimo de respuesta y un tiempo máximo. (Ver Anexo # 17)

Se han utilizado tres herramientas automatizadas de software libre:

- Selenium IDE: para pruebas funcionales.

- Apache JMeter: para pruebas de carga y estrés.
- Xenu: para pruebas de estructura.

Estas han sido utilizadas para validar el procedimiento para pruebas de software con herramientas automatizadas en el Departamento de Pruebas de Software (D.P.S).

❖ **Importancia del Procedimiento**

El procedimiento de prueba que se aplicó con herramientas automatizadas, descrito en el transcurso de la investigación posee una relevante importancia. Pues, es un procedimiento general aplicable a todos los tipos de pruebas automatizadas, demostrándose que los artefactos definidos durante el proceso fueron generados significativamente. Las herramientas son una fuente de apoyo que logran automatizar dicho proceso, haciendo más rápidas y flexibles las pruebas desarrolladas. Se tomaron como ejemplo a tres tipos de pruebas: pruebas funcionales, pruebas de estructura y pruebas de carga y estrés para llevar a cabo el cumplimiento del procedimiento descrito. Se detectaron las No Conformidades como fueron esperadas, plasmadas cada una en el Documento de No Conformidades. Las plantillas que se utilizaron como base del procedimiento recogen todas las informaciones correspondientes a cada una de ellas. Se generó el Informe de Evaluación de las Pruebas, en su primera versión, siendo este de gran importancia, pues avala todos los resultados obtenidos durante el proceso de prueba.

3.9 Conclusiones

En el desarrollo del capítulo, el análisis de los resultados del procedimiento en general se realizó con una buena efectividad, en el cual se diseñó el caso de prueba Autenticar Usuario como ejemplo de validación en el caso de las pruebas funcionales. Se le realizaron las pruebas a la aplicación escogida por parte del cliente, dando cumplimiento a cada requisito funcional definido; considerando conveniente clasificar las No Conformidades, pues de esta forma se definen aquellos que son urgentes de corregir ya que representan funcionalidades críticas de la aplicación probada.

Conclusiones Generales

Con el estudio y las pruebas que se han llevado a cabo durante el trabajo “Procedimiento para pruebas de software con herramientas automatizadas en el Departamento de Pruebas de Software” se ha cumplido el objetivo propuesto al inicio del mismo el cual consistió en desarrollar un procedimiento para las pruebas automatizadas. Con esto se logró una mejor organización de las actividades del procedimiento, obteniendo un proceso de prueba eficiente haciendo que el producto probado tenga una mayor calidad y a su vez se arribó a las siguientes conclusiones:

- Se realizó una investigación sobre los tipos de pruebas utilizados en el Departamento de Pruebas de Software de Calisoft, haciendo énfasis en las pruebas funcionales, estructura, carga y estrés; así como de las herramientas seleccionadas según sus características donde se obtuvieron resultados reales.

- El procedimiento se organizó de forma eficiente, planificando los recursos disponibles, roles, cronograma, requerimientos a probar, actividades a seguir para dar cumplimiento al proceso de prueba con ayuda de herramientas automatizadas.

- Se diseñó las pruebas de tipo funcional que cubrieron la mayor cantidad de combinaciones posibles en el caso de prueba propuesto para identificar las No Conformidades existente en la aplicación.

- Se logró desarrollar una adecuada documentación de todo el procedimiento en general que servirá para todo tipos de pruebas que se le desarrollen al producto.

- El procedimiento propuesto fue validado con ayuda de las herramientas: Selenium IDE para pruebas funcionales, JMeter para pruebas de carga y estrés, Xenu Link Sleuth para pruebas de estructura y aceptado por parte de los especialistas de pruebas del Departamento de Pruebas de Software de la Universidad.

- El procedimiento propuesto contribuye a elevar la calidad de las pruebas automatizadas en el Departamento de Pruebas de Software de Calisoft, haciendo uso de herramientas automatizadas con el fin de agilizar el proceso de prueba.

Recomendaciones

Una vez concluida la investigación en curso y dada la importancia que esta tiene para la realización de las pruebas automatizadas en el Departamento de Pruebas de Calisoft se recomienda que:

- Se aplique el procedimiento no solo en el Departamento de Pruebas de Software de Calisoft, sino también en los grupos de calidad de las diferentes facultades de la universidad.
- Se haga una propuesta de un sistema para automatizar el procedimiento.
- Los requisitos funcionales y no funcionales varíen de acuerdo al tipo de prueba que se realizará sobre un software determinado.
- Ampliar el estudio de las herramientas de apoyo a la realización de pruebas automatizadas donde se incluyan herramientas multiplataforma que permitan aplicar el procedimiento ajustándose al tipo de prueba que desee realizar el especialista de prueba.

Referencias Bibliográficas

1. **Fernández**, O. M., García León, D., & Beltrán Benavides, A. (s.f.). Enfoque actual de la calidad de software. [En línea][Septiembre-Diciembre,1995] [Citado el: 27 de 10 de 2009]. http://www.bvs.sld.cu/revistas/aci/vol3_3_95/aci05395.htm

2. **Vásquez**, Ing. Roberto Hugo. Introducción a la Calidad de Software. [En línea][Marzo del 2006] [Citado el: 3 de 12 de 2009.] http://gridtics.frm.utn.edu.ar/docs/Introduccion%20a%20la%20Calidad%20de%20Software%20Vazquez.pdf

3. **Vásquez**, Ing. Roberto Hugo. Introducción a la Calidad de Software. [En línea][Marzo del 2006]. [Citado el: 3 de 12 de 2009.] http://gridtics.frm.utn.edu.ar/docs/Introduccion%20a%20la%20Calidad%20de%20Software%20Vazquez.pdf

4. **Cueva**, Juan Manuel .Calidad de Software. [En línea][Abril del 2001]. [Citado el: 15 de 12 de 2009.] http://www.cs.man.ac.uk/~velascop/publ/Tesis.pdf

5. **Valdivia**, Daniel Rolando; Valdivia Espinosa Eduardo Geonías. Estándares de Calidad para las Pruebas de Software,Pruebas de Software. [En línea][2005]. [Citado el: 16 de 12 de 2009.] http://www.cybertesis.edu.pe/sisbib/2005/valdivia_ee/pdf/valdivia_ee-TH.5.pdf

6. **Técnicas** de prueba; [En línea][2004][Citado el: 16 de 12 de 2009.] http://indalog.ual.es/mtorres/LP/Prueba.pdf

7. **Etapa de Prueba;** [En línea][23 de Septiembre de 2007][Citado el: 16 de 12 de 2009.] http://ingpau.blogspot.com/2007/09/etapa-de-pruebas.html

8. **Usaola**. Macario Polo. Mantenimiento de Avanzado de Sistema de información de Prueba de Software. [En línea][2005]. [Citado el: 17 de 12 de 2009.] http://alarcos.inf-cr.uclm.es/doc/masi/doc/lec/parte5/polo-apuntesp5.pdf.

9. **Pruebas** de Software. [En línea][2004] [Citado el: 8 de 1 de 2010.] http://materias.fi.uba.ar/7548/Pruebas-Intro.pdf.

10. **Pruebas** funcionales y de campo. pruebas funcionales y de campo. [En línea] [Citado el: 16 de 12 de 2009.]

http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/fuentes_k_jf/capitulo4.pdf

11. **Pozo**, Delmys; Figueroa, Liane; Pruebas de estructura en aplicaciones web; Universidad de las Ciencias Informáticas [Citado el: 5 de 5 del 2010.]
12. **Sanchez**, Liudmila; Prueba automática de carga y estrés en el proyecto de CICPC. [Citado el: 5 de 5 de 2010.]
13. **Selenium IDE**; http://seleniumhq.org/projects/ide/ [Citado el: 2 de 2 de 2010.]
14. **JMeter**; http://jakarta.apache.org/jmeter/ [Citado el: 5 de 5 de 2010.]
15. **Pozo**, Delmys; Figueroa, Liane; Pruebas de estructura en aplicaciones web; Universidad de las Ciencias Informáticas [Citado el: 5 de 5 del 2010.]

Bibliografía

1. **Selenium.** Selenium. [En línea] 2008.

   http://seleniumhq.org/

2. **Software** Quality Systems S.A. Software Quality Systems S.A. [En línea] 2007.

   http://www.sqs.es/es/.

3. **Raja**, Elena; Casi todas las pruebas del software [En línea] 2007.

   http://www.sistedes.es/TJISBD/Vol-1/No-4/articles/pris-07-raja-ctps.pdf.

4. **Myers**, G. J; ¨The art of Software Testing¨ [2004].

5. **Raymond** McLeod, Jr. ¨Software Testing, Testing Across the Entire, Software Development Life Cycle. [2007].

6. **Burnstein**, Ilene; ¨Practical Software Testing¨; [En línea] 2002.

7. **PRESSMAN, R. S.** ¨Ingeniería del Software. Un enfoque práctico¨. Madrid: s.n., [En línea] 2005.

8. **Echeverría**, Delvis; Calzadilla, Roig; Herramienta Informática para automatizar los procesos en el Laboratorio de Calidad: Módulo Planificación, Universidad de las Ciencias Informáticas: Ciudad Habana [2007].

9. **Pruebas de Software**

   http://lsi.ugr.es/~ig1/docis/pruso.pdf

10. **Bedini** , Alejandro. Calidad Tradicional y de Software.

11. **Laitinen**, Mauri, Fayad, Mohamed and Ward, Robert. Software Engineering in the Small.Comunications of the ACM. [En línea] 2000.

12. **Letelier**, Patricio. Pruebas del Software. Valencia: Departamento de Sistemas Informáticos y Computación.

13. **López**, Ana M.; Cabrera, Cesar; Valencia, Luz E. ¨ INTRODUCCIÓN A LA CALIDAD DE SOFTWARE¨; [En línea] Septiembre de 2008 [Universidad Tecnológica de Pereira.].

http://www.utp.edu.co/php/revistas/ScientiaEtTechnica/docsFTP/111233326-331.pdf

14. **Automatizando** para Pruebas: Selenium IDE; [En línea] 8 de May, 2008; http://www.jourmoly.com.ar/automatizando-pruebas-selenium-ide/
15. **Víctor**, J. Primeros pasos con Selenium IDE.[En Linea] 31-12-2008. http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=seleniumIDE
16. **Automated** Testing Institute. http://www.automatedtestinginstitute.com/home/index.php?option=com_content&view=article&catid=49%3Afunctional-test-tools&id=362%3Aapodora&Itemid=1
17. **ALS. 2007.** Herramientas para el entorno de prueba. [En línea] 2007. http://www.als-es.com/home.php?location=herramientas/entorno-pruebas
18. **EVERETT**, G. D. and McLEOD, R. Software Testing. Across the Entire Software Development Life Cycle. s.l.: JOHN WILEY & SONS, INC., PUBLICATION, 2007.
19. **Escorial**, J. S. Calidad de Software: Medidas del Proceso. 2006.
20. **Lorenzo**, María Félix; Loro, Lisney. Procedimiento para Pruebas de Penetración en Aplicaciones Web. Universidad de las Ciencia Informáticas[2009]
21. **Barban**, Anay; Hernández, Paula. Procedimiento para pruebas de rendimiento de Carga y Estrés al Sistema Único de Aduanas. Universidad de las Ciencias Informáticas[2009]
22. **vTest**, Free Software Manager[En línea][26 de Abril del 2007] http://www.freedownloadmanager.org/es/downloads/vTest_38448_p
23. **Selenium IDE**; http://seleniumhq.org/projects/ide/
24. **Apodora**, ACULIS Software Testing Tools & Resources > Apodora Functional Testing Framework. [11/13/2009] http://www.aculis.com/software-testing/apodora-testing
25. **EMS Datagenerator para MySQL**. http://www.sqlmanager.net/products/mysql/datagenerator
26. **DB Monster**. http://dbmonster.kernelpanic.pl
27. **Software** Quality Systems S.A. Software Quality Systems S.A. [En línea] 2007. http://www.sqs.es/es/.

28. **Raja**, Elena; Casi todas las pruebas del software [En línea] 2007.

http://www.sistedes.es/TJISBD/Vol-1/No-4/articles/pris-07-raja-ctps.pdf.

29. **Myers**, G. J; ¨The art of Software Testing¨ [2004].

30. **Técnicas** de prueba; [En línea][2004]

http://indalog.ual.es/mtorres/LP/Prueba.pdf

31. **Raymond** McLeod, Jr. ¨Software Testing, Testing Across the Entire, Software Development Life Cycle. [2007].

32. **Burnstein**, Ilene; ¨Practical Software Testing¨; [En línea] 2002.

33. **Pruebas** funcionales y de campo. pruebas funcionales y de campo. [En línea] 2009

http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/fuentes_k_jf/capitulo4.pdf.

34. **Esmite**, Ignacio; Farías, Mauricio; Farías, Nicolás; Pérez, Beatriz. Automatización y Gestión de pruebas Funcionales usando herramientas Open Source. [En línea][Octubre del 2007] http://www.cacic2007.unne.edu.ar/papers/027.pdf

35. **Sánchez**, Liudmila; Tutorial Cómo realizar Pruebas de Carga y Estrés en JMeter. Universidad de las Ciencias Informáticas [2008].

36. **Pruebas** de Rendimiento; [En línea][2008].

http://es.testhouse.net/index.php/servicios/pruebas-no-funcionales/pruebas-de-rendimiento

37. **Etapa** de Prueba; [En línea][23 de 9 de 2007]

http://ingpau.blogspot.com/2007/09/etapa-de-pruebas.html

Anexos

Anexo # 1: Criterios de criticidad.

Establecen los parámetros para declarar un producto del desarrollo software en estado crítico de terminación. Son aplicables en tres momentos fundamentales: Cuando se hace la revisión de la solicitud (RS); Durante el proceso de pruebas exploratorias (PE) y Durante las pruebas formales (PF). Los criterios de criticidad son los siguientes:

a) No se presenta la última versión del producto de trabajo comprobada y avalada por la etapa anterior.

b) No están presentes todos los elementos componentes del sistema, producto o entregable. (Hardware, Software, Partes y Documentación). (RS) (PE)

c) No se corresponden totalmente los requisitos funcionales documentados con los implementados. (RS) (PE)

d) Supera el producto la taza de: 1 defecto significativo por CU para (PF) ó ½ defecto significativo por CU para (PE) y en caso de otros artefactos al menos 2 defectos significativos que estén relacionados con las pautas esenciales definidas por la metodología que sigue el proyecto. (PF)

e) Excede el producto el número máximo de iteraciones establecidas por plan (entre 2 y 3 iteraciones). (PF)

f) Se incumple más del 50% de las reglas para la documentación establecidas por el proyecto y/o expediente del proyecto. (RS) (PE) (PF)

g) Existe incoherencia significativa entre los artefactos que tienen relación o dependencia entre sí. (RS) (PE) (PF)

h) Existe incoherencia significativa entre lo documentado y lo implementado. (RS) (PE) (PF)

i) La cantidad de faltas de ortografías excede la cantidad de páginas o pantallas que tiene el artefacto en cuestión. (RS) (PE) (PF)

j) Durante las pruebas de regresión persisten al menos 2 defectos significativos de iteraciones anteriores. (PE) (PF)

k) Se observan los mismos tipos de defectos, ya señalados en iteraciones anteriores, en otros o los mismos lugares donde fueron detectados, para (PE): ½ de la cantidad de defectos tipo encontrados en etapa anterior, o para (PF): ¼ de la cantidad de defectos tipo encontrados en etapa anterior.

Si se aplica alguno de estos Criterios de Criticidad, se considera la prueba fallida y se para la prueba hasta el día siguiente como mínimo.