Universidad de las Ciencias Informáticas

Facultad # 3

Diseño y aplicación de pruebas al módulo Misiones del sistema Convenio Cuba-Venezuela.

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

Autora: Daymara Eva Fariñas González

Tutores: Ing. Daylín María Fariñas González
Ing. Juan Carlos Montané Izaguirre

Junio, 2011

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la facultad 3 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Daymara Eva Fariñas González

_____

(Autor)

Daylín María Fariñas González

_____

(Tutor)

Juan Carlos Montané Izaguirre.

_____

(Tutor)

AGRADECIMIENTOS

A mis padrecitos, por ser la razón de mi existencia y darme todo el amor del mundo. Los amo.

A mi hermana, la mejor tutora del mundo, con este trabajo también termina uno de tus dolores de cabeza je je, no sabes cuánto te agradezco tu apoyo. Todos mis méritos son también los tuyos. Te quiero mucho.

A Juan Carlos, alias el pepino, mi segundo tutor, por aguantarme estos cinco años y ayudarme en este trabajo.

A mi noviecito lindo, ya somos dos ingenieros más en la familia. ¡I Lub U!

Gracias a todos por dar su granito de arena.

Como todo queda en familia, los quiero mucho!!!

DEDICATORIA

A mi mamá y papá, no me alcanzaría esta página si pusiera todo lo que siento por ustedes, porque me dieron todo para alcanzar mis sueños, consejos para hacerme una mejor persona, ánimos para no dejarme caer, besos para mimarme. Porque sé que para ustedes siempre seré la chiquitica de la casa, los amo con todo mi corazoncito.

A mi hermanita del alma querida, por estar a mi lado cuando más lo necesité, quiero que sepas que sin ti no hubiese podido llegar hasta aquí y que tus regaños no fueron en vano, por ellos hoy me convierto en ingeniera. Todos los días le agradezco a la vida por haberme dado un regalo tan lindo, nunca olvides que te quiero del tamaño del infinito.

A mi abuelita Eva, la más enérgica de todas las abuelitas, gracias por darme todo de ti y malcriarme como lo has hecho, aquí está la nenita que más te quiere y más que eso te adora.

A mi viejito lindo, bibi donde quieras que estés, aquí está tu nietecita preferida echa toda una profesional, por dedicar tus últimos momentos a aconsejarme y guiarme por el buen camino, siempre estarás en mi corazón.

A mi abuela Reyna y abuelo Hugo, espero que se sientan orgullosos de mí, ustedes también son parte de mis logros. Los quiero.

A mi tía Lurdes y a mi bellas primas Lisbe y Pachy. Hoy quisiera tenerlas a mi lado para compartir este gran sueño hecho realidad, porque son parte de mis alegrías, las quiero con toda mi alma.

A mi tía Reinita, Meme, Lise y Luisi. Gracias por haber formado parte de momentos lindos de mi vida, soy la sobrina y prima que más los quiere.

A mi puchunguito lindo, no por estar al final de estas líneas eres el menos importante. Mi amor quiero que sepas que te quiero mucho. Sin ti estos cinco años no hubiesen tenido el mismo sentido. Te Amo!!

¡A todos los quiero mucho!

RESUMEN

La Secretaría Técnica (ST) de Cuba consciente de los beneficios que brindan los adelantos tecnológicos solicitó a principios del curso 2007-2008 al equipo del proyecto Convenio Cuba-Venezuela (CCV) de la Universidad de las Ciencias Informáticas la automatización de los trámites referentes a las misiones colaborativas de Cuba con otros países. Al módulo concebido se le denominó *"Misiones"*. El mismo luego de pasar por un primer desarrollo como un módulo más del sistema CCV, se sometió a un segundo desarrollo, pues era necesario un refinamiento de los requisitos y cambio de la plataforma arquitectónica por una más ligera.

Para evitar mayores desentendidos con la ST de Cuba y conociendo además que durante el proceso de creación de software se cometen errores propios del desempeño humano, se procurará que el módulo salga con la mayor calidad posible y un mínimo de defectos. Para ello entre otras acciones se estudian los conceptos relacionados con las pruebas de software, se elabora la estrategia de prueba a seguir, se diseñan los Casos de Prueba necesarios para probar las funcionalidades del módulo e identifican las no conformidades para su corrección y con ello obtener un producto con calidad.

Palabras Clave: pruebas de software, artefactos, herramientas de pruebas de software.

Índice de Figuras

Índice

INTRODUCCIÓN

Actualmente los sistemas informáticos constituyen sin lugar a dudas una evidencia del impacto de la tecnología en cualquier sociedad y se convierten en una alternativa válida para la sofisticada gestión de los procesos de negocio. Cada vez más el trabajo cotidiano de las personas y las empresas se optimiza con el procesamiento de información en computadores.

La Secretaría Técnica (ST) de Cuba consciente de los beneficios que brindan los adelantos tecnológicos solicitó con énfasis a principios del curso 2007-2008 al equipo del proyecto Convenio Cuba-Venezuela(CCV), de la Universidad de las Ciencias Informáticas la automatización de los trámites referentes a las misiones colaborativas de Cuba con otros países. Al módulo concebido se le denominó *"Misiones"*.

Para este entonces no se contaba con la organización y productividad adecuada en las actividades que se llevaban a cabo. La creación de una solicitud de misión se realizaba manualmente por el responsable de un ente. El proceso de aprobación era un poco ineficiente y demoraba algunos días debido al envío de la información física a los rectores (Ministerios y Secretarías Técnicas) y respectiva retroalimentación de estos sobre la aceptación o rechazo de la solicitud.

La totalidad de las deficiencias del negocio de las ST fueron resueltas en Noviembre de 2008 cuando se implanta la primera versión del sistema CCV tanto en Cuba como en Venezuela. Aunque las expectativas de la contraparte venezolana no fueron satisfechas, entre otras, por problemas de alcance, comprensión y rendimiento del mismo. Por esta razón se negoció realizar un refinamiento de los requisitos y cambiar la plataforma arquitectónica por una más ligera.

Ante esta decisión Misiones se afectó y tuvo que someterse de igual manera a un nuevo desarrollo. Por lo que los documentos hasta ese momento generados: Visión V1, Especificación de Requisitos de Software V1, Modelo de Casos de Usos del Sistema V1, Arquitectura del Sistema V1 y Diseño de Casos de Prueba quedaron obsoletos. Otros como el Plan de Pruebas no fueron generados fundamentalmente por la premura del desarrollo.

Para evitar mayores desentendidos con la ST de Cuba y conociendo además que durante el proceso de creación de software se cometen errores propios del desempeño humano, se procurará que el módulo salga con la mayor calidad posible y un mínimo de defectos.
Debido a la problemática planteada se deriva el siguiente diseño metodológico:

Problema de investigación: ¿Cómo contribuir a la detección y corrección de errores del módulo Misiones del sistema CCV que permita la obtención del producto con calidad?

Objeto de estudio: Gestión de la calidad.

Objetivo general: Realizar el diseño y aplicación de pruebas de software al módulo Misiones del sistema CCV que permita la obtención del producto con calidad.

Campo de acción: Pruebas de software.

Idea a defender: Realizando el diseño y aplicación de pruebas de software al módulo Misiones del sistema CCV permitirá la detección y corrección de errores y con ello la calidad del producto.

Objetivos específicos:

1. Elaborar el marco teórico para la justificación del trabajo.
2. Realizar el proceso de pruebas de software para el módulo.
3. Evaluar los resultados de la ejecución de las pruebas.

Tareas de la investigación:

1. Estudio de los conceptos relacionados con las pruebas de software.
2. Investigación acerca de las herramientas de pruebas para seleccionar la más adecuada.
3. Estudio del flujo de trabajo de pruebas de la metodología RUP.
4. Elaboración de la estrategia de prueba a seguir.
5. Identificación de requerimientos a probar.
6. Definición del entorno de prueba, detallando los recursos de hardware y software a utilizar.

7. Realización del diseño de Casos de Prueba necesarios para probar las funcionalidades del módulo.
8. Elaboración de los Datos de Prueba.
9. Aplicación de las pruebas funcionales mediante el método de prueba de Caja Negra.
10. Identificación de las no conformidades.
11. Evaluación de los resultados obtenidos.

Métodos de investigación

Métodos Científicos

Teóricos:

- ✓ Histórico – Lógico: Para el estudio del estado del arte de las pruebas de software y herramientas para pruebas más utilizadas.
- ✓ Analítico – Sintético: Para resumir las ideas de los diferentes autores prestigiosos sobre pruebas de software.

Empíricos:

- ✓ Revisión bibliográfica: Para fundamentar el estudio del estado del arte mediante consultas de un conjunto de fuentes de información referidas al tema de pruebas, como libros, artículos, revistas, publicaciones, boletines y toda una variedad de materiales escritos y digitales.
- ✓ Experimentación: Para evaluar y mejorar el software una vez aplicadas las pruebas y detectadas las no conformidades.

Estructura de la tesis

Para el desarrollo del trabajo se proponen tres capítulos, los cuales están estructurados de la siguiente manera:

En el capítulo 1 se hace un estudio y caracterización de las pruebas de software. Se tratan los métodos de pruebas que existen y las diferentes técnicas de cada uno, se presentan niveles de pruebas y se caracterizan herramientas para la automatización de las mismas. Se describen también algunas metodologías de desarrollo, en cada caso se asume una posición como fundamentación a la realización del trabajo.

En el capítulo 2 se elaboran diferentes artefactos como Plan de Pruebas que abarca la estrategia a utilizar durante el proceso, el escenario donde se desarrollarán las pruebas entre otros. Se generan además los artefactos Datos de Prueba, Casos de Prueba y No Conformidades.

En el capítulo 3 se aplican listas de chequeo al Plan de Pruebas y Casos de Prueba para verificar que los mismos cumplan con indicadores requeridos, se le otorga de esta forma una evaluación a cada uno de ellos así como al proceso de pruebas aplicado al módulo en cuestión.

Capítulo 1

Fundamentación Teórica

1.1 Introducción

En el presente capítulo se muestra un estudio de los conceptos de prueba de software dados por autores prestigiosos así como principios por lo que se puede regir a la hora de llevar a cabo este proceso. Se exponen los métodos de pruebas existentes, los cuales ayudan a detectar con más facilidad lo errores en el sistema y las técnicas de cada uno de ellos. También se tratan metodologías de desarrollo de software, herramientas para la automatización de las pruebas así como niveles de prueba. En cada caso se asume una posición como cimiento a la realización del trabajo.

1.2 Pruebas de software

Durante la creación de un producto de software se llevan a cabo disímiles actividades en las que es muy posible la aparición de errores, ya sea en los primeros o posteriores momentos del desarrollo. La significativa participación de las personas en este proceso impone la realización de acciones que contribuyan a la calidad del producto debido a la imposibilidad del humano de comunicarse y trabajar de forma perfecta. Para lograr este objetivo tiene un papel fundamental la aplicación de pruebas para la detección y corrección de errores.

Para Myers probar (o la prueba) es el *"proceso de ejecutar un programa con el fin de encontrar errores"* (Myers, 1979).

Según el glosario de términos de la IEEE 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"* (IEEE, 1990).

Según Jacobson *"La prueba (flujo de trabajo) es el flujo de trabajo fundamental cuyo propósito general es comprobar el resultado de la implementación mediante las pruebas de cada construcción, incluyendo tanto construcciones internas como intermedias, así como las versiones finales del sistema que van a ser entregadas a terceras partes"* (Jacobson, y otros, 2003).

Pressman a su vez enuncia que *"Las pruebas del software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación"* (Pressman, 2005).

De las anteriores definiciones se puede concluir que las pruebas son un conjunto de actividades necesarias e iterativas que se realizan con el fin de identificar posibles fallos o errores durante el ciclo de vida de un producto de software y corregir los mismos para contribuir con su calidad.

1.3 Principios de las Pruebas de software

En consonancia con las definiciones anteriores existen diferentes objetivos de las pruebas de software tales como: diseñar pruebas que saquen a la luz diferentes errores con la menor cantidad de tiempo y recursos (Pressman, 2005) y encontrar un error no descubierto hasta entonces (Myers, 2004). La correcta puesta en práctica de los mismos ayuda a verificar entre otras el cumplimiento de los requisitos funcionales y no funcionales así como la lógica interna del programa. Teniéndolos en cuenta Davis A propone una serie de principios que ayudan y guían durante la ejecución del proceso de prueba:

- ✓ La prueba puede ser usada para mostrar la presencia de errores, pero nunca de su ausencia.
- ✓ La principal dificultad del proceso de prueba es decidir cuándo parar.
- ✓ Evitar Casos de Prueba no planificados, no reusables y triviales a menos que el programa sea verdaderamente sencillo.
- ✓ Una parte necesaria de un caso de prueba es la definición del resultado esperado.
- ✓ Los Casos de Prueba tienen que ser escritos no solo para condiciones de entradas válidas y esperadas sino también para condiciones no válidas e inesperadas.
- ✓ Los Casos de Prueba tienen que ser escritos para generar las condiciones de salida deseadas.
- ✓ El número de errores sin descubrir es directamente proporcional al número de errores descubiertos.
- ✓ Las pruebas deberían empezar por lo pequeño y progresar hacia lo grande.
- ✓ Asigna el programador más creativo a la prueba (Davis, 1995).

Luego del estudio de los principios se puede afirmar que recogen un conjunto de ideas y buenas prácticas a tener en cuenta, que sin lugar a dudas guían la ejecución de la etapa de pruebas de software.

1.4 Niveles de prueba

Todo producto antes de ser entregado al cliente debe pasar por un proceso de pruebas para tratar de que el mismo salga con la máxima calidad requerida y la menor cantidad de defectos. Lo más común es que vayan de lo más pequeño hasta lo más grande, comenzando con la prueba de unidad, luego se progresa con la prueba de integración, después la prueba de validación y termina probando el software como un todo mediante la prueba de sistema.

- ✓ **Prueba de Unidad:** Mediante esta prueba se comprueba que una unidad funciona correctamente por sí misma, sin tener en cuenta las relaciones que pueda tener con otras partes del sistema. Probando uno a uno los diferentes módulos que lo constituyen con el fin de descubrir errores dentro del ámbito de dicho módulo (Mañas, 1994). Es aplicable a componentes representados en el modelo de implementación para verificar que los flujos de control y de datos están cubiertos, y que ellos funcionen como se espera. Esta prueba siempre está orientada a Caja Blanca (EVA, 2011).

- ✓ **Prueba de Integración:** se ejecutan durante la construcción del software, implican varios módulos y culminan con la realización de las pruebas al sistema como un todo (Mañas, 1994). Tienen como objetivo tomar los componentes probados en unidad y descubrir errores que se pueden producir en la interacción entre dichos módulos (Javier Tuya, 2007). Estas pruebas se realizan por etapas, abarcando de forma gradual más módulos en cada prueba. Se pueden empezar en cuanto se tienen pocos módulos, aunque no terminan hasta que se encuentren todos terminados (Juristo, 2006).

- ✓ **Prueba de Validación:** validan los requisitos establecidos comparándolos con el sistema que ha sido construido (Guzmán, 2008). Básicamente estas pruebas se centran en asegurar que dichos requisitos satisfagan las expectativas del cliente (2003). Se realizan sobre un software completamente integrado en vista a evaluar el cumplimiento de los requisitos tanto funcionales como los de rendimientos, seguridad y demás especificados por el usuario final (Prado, 2007).

- ✓ **Prueba de Sistema:** es la prueba global del sistema como unidad de ejecución (2003) . Prueban el software como un todo para comprobar entre otras que se cumpla con los requisitos funcionales, la correcta función y rendimiento en las interfaces hardware, software y de usuario además de verificar que el sistema no presente problemas de rendimiento a la hora de ejecutarlo en

condiciones límite y de sobrecarga. Para dirigir estas pruebas pueden usarse las pruebas de Caja Negra (2005) (Juristo, 2006). Está constituida por otros tipos de pruebas que tienen como objetivo verificar que todos los elementos del sistema están integrados correctamente y realizan las funciones adecuadas, entre ellas se encuentran las de Rendimiento, Resistencia, Robustez, Seguridad, Usabilidad e Instalación (Prado, 2007).

1.5 Métodos de pruebas de software

Los métodos de pruebas del software tienen como objetivo proporcionar una vía más fácil para buscar errores garantizando la obtención de un producto de mayor calidad. Tradicionalmente se dividen en Prueba de Caja Blanca y Prueba de Caja Negra.

1.5.1 Prueba de Caja Negra. Técnicas

Las pruebas de Caja Negra no están basadas en el conocimiento del código o diseño interno sino en comprobar la funcionalidad del sistema. Se centran fundamentalmente en las funciones, entradas y salidas. Se llevan a cabo sobre la interfaz del software, donde los Casos de Prueba pretenden demostrar que el software es operativo, que las entradas se aceptan y las salidas se producen de forma correcta y que la integridad de la información externa se mantiene.

El método de Caja Negra deriva un conjunto de Casos de Prueba. Para la elaboración de los mismos se necesita cierta cantidad de datos que ayuden a su ejecución y que permitan que el sistema se ejecute en todas sus variantes, pueden ser datos válidos o inválidos según si lo que se desea es hallar un error o probar una funcionalidad. Los datos se escogen atendiendo a las especificaciones del problema, sin importar los detalles internos del programa, a fin de verificar que el programa funcione correctamente.
Pressman muestra diferentes técnicas dentro de este método, entre las que se encuentran (Pressman, 2005):

- ✓ **Métodos de prueba basados en grafos:** se crea un grafo de objetos importantes y sus relaciones y propone el diseño de una serie de pruebas que cubran el grafo de manera que se ejerciten todos los objetos y sus relaciones para descubrir errores.

- ✓ **Partición equivalente:** se dirige a la definición de Casos de Prueba que descubran clases de errores, reduciendo así el número total de Casos de Prueba en uno más pequeño que sea

manejable y que evalúe bien el software. Se toma un riesgo porque se escoge no probar todo, por lo que se necesita ser cuidadoso a la hora de escoger las clases (Patton, 2005).

- ✓ **Análisis de valores límite (AVL):** es una técnica de diseño de Casos de Prueba que completa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el AVL lleva a la elección de Casos de Prueba en los extremos de la clase. En lugar de centrarse solamente en las condiciones de entrada, el AVL obtiene Casos de Prueba también para el campo de salida.

- ✓ **Prueba de comparación:** se utiliza en situaciones críticas en las que el software debe ser sumamente fiable. En estos casos se desarrolla una serie de versiones del programa siguiendo las mismas especificaciones a las cuales se les realiza pruebas con los mismos datos de entrada para asegurar la misma salida, en caso que se obtenga la misma salida, significa que todas las implementaciones son correctas, en caso contrario, es necesario revisar todas las versiones para encontrar la causa de las diferencias.

1.5.2 Prueba de Caja Blanca. Técnicas

No basta con realizar pruebas que demuestren la buena funcionalidad del software, también es necesario realizar pruebas para comprobar que el código que se ha escrito funciona correctamente y de esta forma cumpla con los estándares requeridos. Para contribuir a ello se aplican las Pruebas de Caja Blanca. Es en este tipo de prueba donde se comprueban los caminos lógicos del software.

Las Pruebas de Caja Blanca realizan un seguimiento del código fuente según va ejecutando los Casos de Prueba, de manera que se determinan de manera concreta las instrucciones, bloques y otros en los que existen errores (Usaola, 2006).

"El objetivo fundamental de las Pruebas de Caja Blanca es asegurar el correcto funcionamiento de las interfaces, o flujo de datos entre componentes" (Oré, 2009).

Este método cuenta con técnicas como:

- ✓ **Prueba del camino básico:** permite obtener una medida de la complejidad de un diseño procedimental, y utilizar esta medida como guía para la definición de una serie de caminos básicos de ejecución, diseñando Casos de Prueba a partir de un conjunto dado de caminos independientes

por los cuales puede circular el flujo de control que garanticen que cada camino se ejecuta al menos una vez (Pressman, 2005).

- ✓ **Prueba de condición:** se centra en la prueba de cada una de las condiciones del programa, diseña además Casos de Prueba que permiten ejercitar condiciones lógicas contenidas en el módulo de un programa, asegurando así que ninguna condición del programa contenga errores (Jimenez, 2004).
- ✓ **Prueba del flujo de datos:** se basa en seleccionar diferentes caminos de prueba de forma que se pueda probar todas las definiciones, variables y estructuras de datos de los programas (María Garzón Villar, 2007).
- ✓ **Prueba de bucles:** es una prueba complementaria a la del camino básico y se basa en estudiar la validez de todos los bucles del programa (María Garzón Villar, 2003).

Aplicar los métodos de prueba es de gran importancia, pues ayudan a encontrar defectos que atentan contra la correcta puesta en marcha del software en construcción. Teniendo en cuenta que cada cual cumple funciones diferentes, es buena práctica utilizarlos de forma complementaria, es decir aplicar uno y el otro en vista a descubrir nuevos tipos de errores. Esto trae consigo que al final del desarrollo se adquiera un producto que cumpla con los requisitos definidos por el cliente.

En el presente trabajo se utilizará para las pruebas funcionales el método de prueba de Caja Negra debido al poco tiempo planificado para las pruebas a nivel de proyecto. Se hará uso además de la técnica partición equivalente mediante la definición de clases válidas e inválidas que permiten probar bien el software. No se hará uso de otra técnica como complemento de la anterior entre otras razones porque el producto no cuenta con campos que requieren variedad en los datos de entrada.

1.6 Automatización de las pruebas

Realizar pruebas manuales a una aplicación puede conllevar muchas veces a la aparición de errores, los cuales podrían ser evitados mediante el uso de herramientas para su automatización. Esto permite la obtención de mejores y satisfactorios resultados.

Esta forma es más rápida y eficaz pues no requiere la introducción de datos de prueba varias veces procurando así un mayor ahorro en cuanto a tiempo y recursos (Toretti, 2010). Es importante tener en cuenta la correcta configuración de la herramienta a utilizar para evitar inconvenientes o mal funcionamiento de la misma (Rueda, 2001).

1.6.1 Selenium IDE

Selenium IDE es un plugin de Firefox que permite la realización de pruebas funcionales sobre aplicaciones Web. Ahorra tiempo cada vez que se resuelve alguna incidencia o se genera una versión nueva. Automatiza la realización de las pruebas específicas (una acción en particular) o bien juegos de pruebas (un conjunto de acciones). Forma parte de las herramientas de código abierto y es multiplataforma (Selenium, 2010).

Esta herramienta graba y reproduce los escenarios de las pruebas automatizadas. Es muy fácil de usar, permite agregar validaciones, además de poder escribir, editar y ejecutar los scripts grabados durante la ejecución del escenario a automatizar. Una vez creadas las pruebas los resultados pueden ser visualizados en diferentes formatos lo cual facilita las labores de análisis para el probador y la realización de un análisis exhaustivo de las pruebas realizadas.

1.6.2 JMeter

JMeter es una herramienta de software libre, multiplataforma, utilizada para la ejecución de pruebas de carga y estrés. Permite probar la concurrencia de usuarios y velocidad de carga en un sitio. Es una aplicación de escritorio 100 por ciento Java. Permite realizar pruebas distribuidas en distintos ordenadores que actuarán como clientes simulando varios hilos que harán función de usuarios, así como generar un caso de prueba a través de una navegación de usuario (Jakarta, 2010).

Posee la capacidad de realizar desde una solicitud sencilla hasta secuencias de ellas que permiten diagnosticar el comportamiento de una aplicación en condiciones de producción (Pello, 2005). Permite conocer los tiempos de respuesta experimentados por la aplicación cuando se tiene un determinado número de usuarios y el número real de transacciones procesadas por unidad de tiempo (Pruebas, 2011). Da la posibilidad de visualizar la respuesta del servidor en varios formatos como XML y HTML y generar gráficos con los resultados obtenidos. Su interfaz gráfica da la posibilidad de realizar las operaciones más rápido, además puede cargar y realizar pruebas sobre distintos tipos de servidores (Jakarta, 2010).

1.6.3 JUnit

JUnit es un conjunto de clases java que el usuario extiende para crear un entorno de pruebas automatizado. Permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera (Sommerville, 2005).

Posibilita detectar nuevos bugs que puedan surgir a consecuencia de haber cambiado el código fuente inicial. De esta manera, permite comprobar que la aplicación cumple con los requisitos y que no se ha alterado su funcionalidad inicial.

Esta herramienta visualiza los resultados en modo texto y en modo gráfico. Una de las mejores características de ella es que existen plugins para Eclipse y Netbeans que generan automáticamente las clases java necesarias para la creación de la pruebas, de manera que el programador solo tenga que centrarse en si los resultados han sido correctos o no (Unix, 2010).

1.7 Selección de la herramienta a utilizar

Por las características anteriormente mencionadas se ha decidido utilizar para la realización de las pruebas funcionales la herramienta **Selenium IDE** ya que permite crear y mantener las pruebas web automatizadas de forma gratuita, es multiplataforma, proporciona facilidad a la hora de ejecutar y registrar las pruebas, las acciones pueden ser ejecutadas paso a paso lo que hace que sea de fácil uso, y puede ser utilizada para varios lenguajes entre los que se encuentra Java.

1.8 Metodologías de desarrollo de software

Los procesos de desarrollo de software son la base para que todo proyecto se realice de forma correcta, entendible y con la mayor calidad posible. Estos definen las actividades necesarias para transformar los requisitos de un usuario en un sistema software, especificando *quién* hace *qué*, *cuándo* y *cómo* alcanzar un determinado objetivo (Jacobson, y otros, 2003).

No es óptimo ajustarse exactamente a un proceso de desarrollo, sino que se debe adaptar a las necesidades y características de cada empresa, equipo de trabajo o proyecto en específico. Con el fin de minimizar la inversión requerida y obtener los resultados esperados en el tiempo acordado. Elegir un proceso de desarrollo adecuado a la hora de elaborar un software contribuye a resultado exitoso.

1.8.1 Extreme Programming (XP)

La metodología XP o Programación Extrema, es una metodología ágil utilizada para desarrollar software de la manera más rápida posible y con el mayor beneficio para el cliente, siendo este una parte más del equipo de desarrollo. Se caracteriza por tener ciclos de desarrollo extremadamente breves, integración constante, retroalimentación continua entre el cliente y el equipo de desarrollo, reutilización de código, pruebas automatizadas regulares y enfoque de equipo (Borrero, 2003).

Se distingue por ser concreta y dependiente de las pruebas automatizadas, las cuales son escritas por programadores y clientes para monitorizar el progreso de la aplicación y para encontrar los defectos en una etapa temprana del desarrollo, permitiendo la retroalimentación de los mismos (Beck, 2000).

Consta de cuatro fases Planificación, Diseño, Desarrollo y Prueba. Esta última constituye el pilar de la metodología. Ella se integra en el proceso de construcción e integración continua, lo que permitirá obtener en el desarrollo futuro una plataforma estable y de buena calidad. Las pruebas se hacen todo el tiempo, no sólo de cada nueva clase (pruebas unitarias) sino que también los clientes irán comprobando que el proyecto va satisfaciendo los requisitos (pruebas aceptación (funcionales)) (Programming, 2009).

Las pruebas unitarias, dirigen la producción de código y se establecen antes de escribir el mismo, son ejecutadas constantemente ante cada modificación del sistema. Las pruebas de aceptación son creadas y usadas por los clientes para comprobar que el software cumple con las especificaciones y requerimientos deseados, además significan el final de una iteración y el comienzo de la siguiente.

Existen tres roles que juegan un papel fundamental en el proceso de Prueba, ellos son:

- ✓ **El programador**: escribe las pruebas unitarias y produce el código del sistema.
- ✓ **El cliente** escribe las pruebas funcionales para validar su implementación.
- ✓ **El encargado de pruebas** ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.

1.8.2 Rational Unified Process (RUP)

La metodología Rational Unified Process conocida por sus siglas RUP es un proceso para el desarrollo de un software. Es orientada a objetos que constituye uno de los procesos más generales, ya que está pensado para adaptarse a cualquier proyecto (Jacobson, y otros, 2003). Describe cómo aplicar efectivamente enfoques comprobados comercialmente para el desarrollo de software (Internacional, 2007).

Se divide en cuatro fases: Inicio, Elaboración, Construcción y Transición, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto. Se centra en las actividades que tiene lugar durante el proceso de desarrollo denominadas flujos de trabajo y los cuales pueden estar activos en todas las etapas del proceso. Son nueve los flujos definidos, entre ellos se encuentra el de Prueba el cual se debe poner en práctica desde la fase de inicio (Kruchten, 2004). En la siguiente figura se muestra la relación de fases y flujos de esta metodología, es decir su arquitectura global.

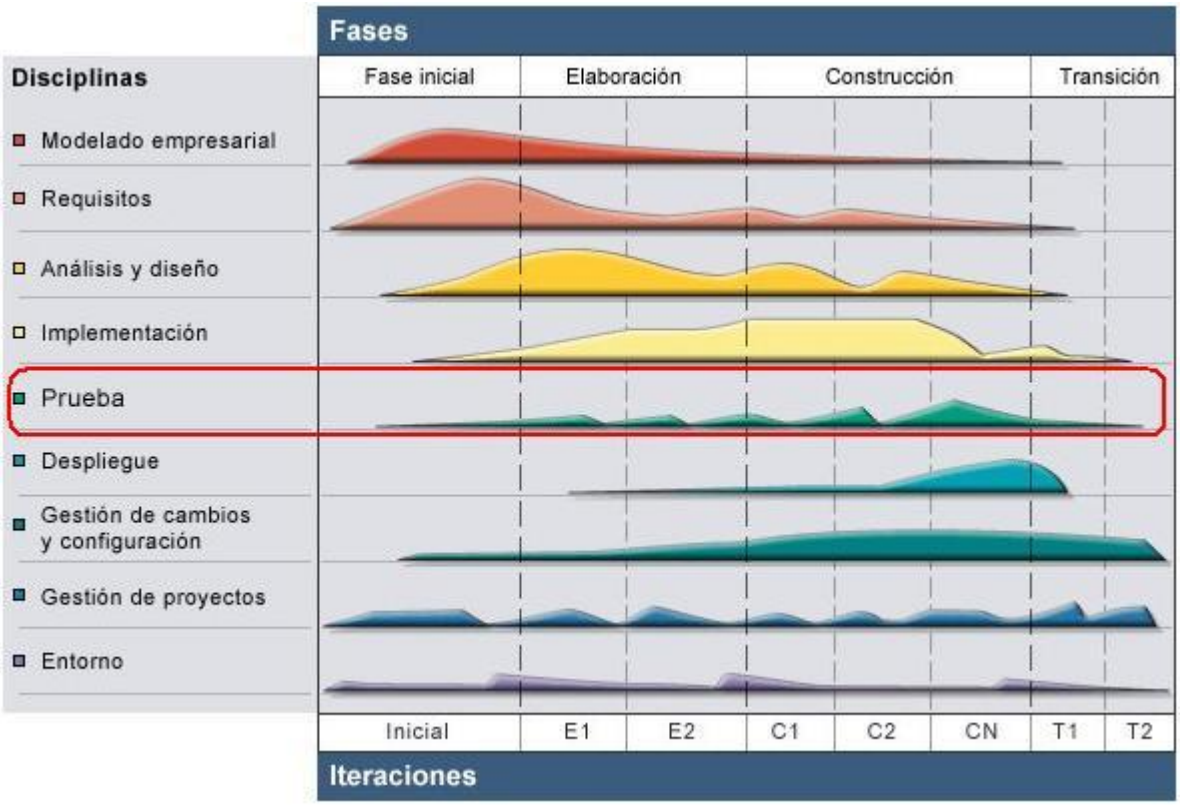

Figura 1 Arquitectura global de RUP

El flujo de trabajo de Prueba proporciona orientación sobre cómo evaluar y valorar la calidad del producto a lo largo de todo el ciclo de vida. Entre sus objetivos están, 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 y validar que los requisitos fueron implementados correctamente (IBM, 2006).

Esta disciplina requiere de un mayor esfuerzo en la fase de construcción ya que es el momento en el que se valida el producto de software. Antes, durante y después de esta fase se realizan actividades de verificación (Edumilis Méndez, 2007). En ella intervienen una serie de roles que son los encargados del desarrollo de las pruebas y de generar una serie de artefactos (Figura 2) (RUP, 2006).

Figura 2 Roles y artefactos generados en el flujo de trabajo de Prueba

Se designará para desempeñar los cuatro roles correspondientes a la etapa de Prueba al ingeniero de pruebas, el cual será el encargado de generar los artefactos Datos de prueba, Casos de Prueba, No Conformidades y Plan de Pruebas que recoge la estrategia de pruebas, configuración del entrono de pruebas, evaluación de las pruebas entre otros aspectos.

1.9 Selección de la metodología de desarrollo de software a utilizar.

Se selecciona para esta investigación RUP como metodología de desarrollo de software, ya que es adaptable para proyectos pequeños, como es el caso de CCV. Los requisitos funcionales determinan la funcionalidad del sistema mediante la elaboración de casos de uso, lo cual facilita el diseño de Casos de Prueba y automatización de las mismas; es iterativo e incremental lo que posibilita que las pruebas no se apliquen al final del proyecto ya que en cada iteración puede ser aplicado al menos un ciclo de pruebas, tiene el proceso de pruebas bien definido. Por tener el equipo de desarrollo conocimiento en su aplicación y por contar con clientes que no tendrán una relación directa con el equipo del proyecto.

1.10 Conclusiones

En este capítulo se realizó un estudio de la etapa de pruebas, especificando una serie de principios que ayudan al buen desarrollo del proceso de pruebas, los métodos de pruebas de software con sus

respectivas técnicas, herramientas para la automatización de las pruebas así como diferentes metodologías de desarrollo de software.

De acuerdo a este estudio se arribaron a las siguientes conclusiones:

- ✓ Aplicando un sólido proceso de pruebas de software como vía óptima para la detección y prevención de errores se obtendrá un sistema con la mayor calidad requerida y que satisfaga las necesidades del cliente.
- ✓ Se aplicarán las pruebas de tipo funcionales a la aplicación utilizando el método de Caja Negra y dentro de este la técnica partición equivalente con el propósito de diseñar Casos de Prueba que saquen a la luz la mayor cantidad de errores.
- ✓ Todas las metodologías tienen sus características. Para elegir alguna de estas es necesario identificar cuál se adapta más al medio. Lo que si no es correcto es no utilizarlas. En el presente trabajo se aplicará RUP por las razones antes expuestas.
- ✓ Para desempeñar los cuatro roles correspondientes a la etapa de prueba de RUP se designó el rol ingeniero de pruebas el cual será el encargado de llevar a cabo todas las actividades que se requieren durante este proceso.
- ✓ En el proceso de pruebas definido por la metodología de desarrollo de software RUP se generan una gran cantidad de artefactos. Los artefactos que se han decidido generar en el presente trabajo, teniendo en cuenta su importancia son: Plan de Pruebas, Casos de Prueba, Datos de Prueba y No Conformidades.
- ✓ Para la automatización de las pruebas funcionales de software es de gran utilidad auxiliarse de herramientas como Selenium IDE.

Capítulo 2 Solución Propuesta

2.1 Introducción

En el presente capítulo se definen los diferentes artefactos correspondientes a la etapa de Prueba para el módulo Misiones del sistema CCV. Inicialmente se describe el módulo a probar, se elabora el Plan de Pruebas, que recoge los roles con sus respectivas responsabilidades, el escenario donde serán llevadas a cabo las pruebas, teniendo en cuenta el diagrama de despliegue y los recursos del sistema necesarios. Se describe además la estrategia de trabajo a seguir para garantizar el éxito de todo el proceso. Se muestran algunos de los Casos de Prueba diseñados, los datos de prueba así como las no conformidades detectadas durante todo el proceso.

2.2 Descripción del módulo a probar

El proyecto CCV informatiza los procesos que se desarrollan en el marco del Convenio Cuba-Venezuela en materia de colaboración. Consta de ocho módulos de desarrollo entre los que se encuentra Misiones. Este es una funcionalidad desarrollada solamente para la parte cubana, la cual se encarga de la gestión y procesamiento de las solicitudes de misiones del personal cubano que viajarán no solo a Venezuela sino hacia otros países.

En la realización de todo este proceso intervienen una serie de actores como son Coordinador de Ente Ejecutor, Coordinador de Ministerio, Coordinador, Embajada, Especialista principal, Especialista y Técnico que son los que llevan a cabo los trece casos de uso por los que está compuesto Misiones, entre los que se encuentran: Gestionar solicitud donde se crea, modifica o elimina la misma. Procesar solicitud realizada una vez que haya sido enviada a los ministerios. Enviar solicitud, este se aplica a todos los niveles, ministerios, secretarías técnicas y embajadas. Generar informe de viaje para las embajadas creado cuando el país destino no es Venezuela y Dar entrada al informe de viaje que se ejecuta cuando los colaboradores regresan a Cuba.

2.3 Artefacto: Plan de Pruebas

En el flujo de trabajo de Prueba definida por RUP y la cual fue estudiada en el capítulo anterior se generan una serie de artefactos fundamentales que ayudan al buen desarrollo de dicho flujo. Entre estos se encuentra el Plan de Pruebas el cual orienta y dirige al equipo encargado de realizar las operaciones que

requiere esta etapa. Forma parte de la planificación del esfuerzo de trabajo por lo que debe dejar bien claro el alcance para definir las fases en las que se llevarán a cabo las diferentes actividades, los roles y responsabilidades de las personas implicadas, el escenario donde se ejecutarán las pruebas, los recursos del sistema como servidores y PC clientes que se utilizarán para el despliegue de la aplicación, los requerimientos a probar, la estrategia a seguir, la descripción de la evaluación de las pruebas así como el cronograma de ejecución de las mismas. (Ver Anexo 1).

2.4 Plan de Pruebas. Alcance

El alcance que tendrán las pruebas está dado por la intención de revisar las funcionalidades asociadas a los grupos de requisitos del sistema definidas para el módulo Misiones, demostrando además que los datos de entradas sean aceptados y las salidas que se produzcan sean las correctas. Estas pruebas se proponen realizar en tres iteraciones.

A continuación se definen los tipos de pruebas que serán realizadas.

Tipo	Descripción de tipo de prueba	Herramientas y técnicas de Pruebas
Funcional (método Caja Negra)	Prueba centrada en validar las funciones que son objeto de prueba ofreciendo los servicios, métodos o casos de uso requeridos.	Como herramienta manual se utiliza el diseño de Casos de Prueba mediante la técnica partición equivalente donde se tratan todas las posibles entradas y parámetros que pueden tomar los campos de las diferentes interfaces del módulo, y se utilizan las clases para probar un amplio rango de posibilidades. Como herramienta automatizada se utiliza Selenium IDE.

2.5 Plan de Pruebas. Roles y Responsabilidades

Para realizarle pruebas a un producto de software se debe contar con un equipo encargado de ejecutar este proceso. El mismo debe estar formado por el analista, diseñador y gestor (administrador) de pruebas así como el verificador (probador). Destacar que en el caso de CCV debido a la sencillez del módulo Misiones, no se cuenta con todo este equipo para efectuar las pruebas, por lo que una sola persona será la encargada de desempeñar estos cuatro roles y desarrollar todas las actividades correspondientes a cada uno de ellos.

En la siguiente tabla se muestra el rol principal con sus respectivas responsabilidades.

Rol	Responsabilidades
Ingeniero de pruebas	• Definir la estrategia de pruebas. • Especificar la configuración del entorno de pruebas. • Diseñar Casos de Prueba. • Detallar los Datos de Prueba. • Definir el Plan de Pruebas. • Resumir la evaluación de las pruebas. • Ejecutar las pruebas. • Elaborar documento de No Nonformidades.

2.6 Plan de Pruebas. Escenario de pruebas

El escenario de pruebas tiene como objetivo principal crear un ambiente cercano al real, de forma tal que las pruebas no se lleven a cabo en el entorno de desarrollo del software, esto es muy importante ya que permite ver cómo se va a comportar el software en su ambiente verdadero, detectar los defectos y evaluar las pruebas con mayor facilidad.

Para la puesta en marcha de las pruebas al módulo Misiones el ingeniero de pruebas cuenta con una computadora dispuesta a tiempo completo, lo que ayudaría a agilizar todo el proceso. Se dispone además de un servidor web, un servidor de aplicación y dos servidores de base de datos, uno de estos últimos cumpliendo la función de backup para copias de respaldo a la información. En los siguientes sub-epígrafes se dará una información detallada de las características tanto de hardware como de software de cada uno de los recursos a utilizar.

2.6.1 Despliegue del sistema

A continuación se muestra el diagrama de despliegue del sistema que describe cómo van a estar distribuidos la aplicación y el equipamiento utilizado para desplegar la misma.

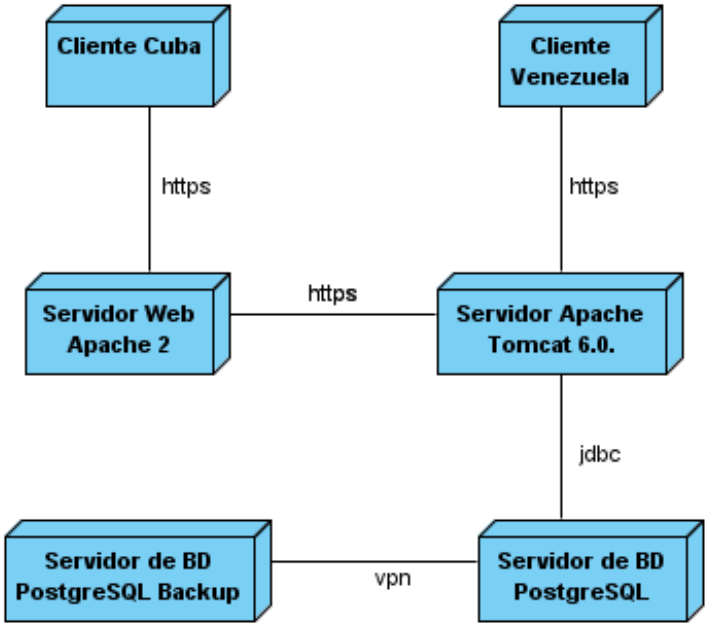

Figura 3 Diagrama de despliegue del sistema CCV

2.6.2 Recursos del sistema

A partir del diagrama anterior se describen los atributos con los que deberán contar las computadoras y servidores que se emplearán para realizar las pruebas, que son los mismos que tendrán que tener cuando se monte la aplicación en su entorno real. Por lo tanto se debe garantizar que ninguno de ellos se afecte o cambie sus características, ya que esto podría afectar e incluso detener el desarrollo de las pruebas, afectando el cronograma de ejecución de las mismas así como el costo por las modificaciones que traería consigo el cambio.

A continuación se muestran dos tablas que especifican las características de dichos recursos, una para los servidores y otra para las PC clientes requeridos.

Servidores:

Recurso	Tipo Hardware	Tipo Software
Servidor Web Apache 2 con función de proxy	Procesador Intel Xeon 3.0 GHz 4 Gb memoria RAM DDR3 40 Gb Disco Duro 10000 RPM	Sistema Operativo Ubuntu: Server 10.04 Máquina Virtual de Java : Openjdk 6 Apache Tomcat 6.0
Contenedor de Servlets	Procesador Intel Xeon 3.0 GHz 8 Gb memoria RAM DDR3 40 Gb Disco Duro 10200 RPM	Sistema Operativo Ubuntu: Server 10.04 Máquina Virtual de Java: Openjdk 6 Apache Tomcat 6.0
Servidor de Bases de Datos PostgreSQL 8.4	Procesador Inter Xeon 3.0 GHz 8 Gb memoria RAM DDR3. 40 Gb Disco Duro 10200 RPM.	Sistema gestor de base de datos PostgreSQL 8.4
Servidor de Bases de Datos PostgreSQL para respaldo (Backup)	Procesador Inter Xeon 3.0 GHz 8 Gb memoria RAM DDR3 40 Gb Disco Duro 10200 RPM	Sistema gestor de base de datos PostgreSQL 8.4

PC Clientes:

Cantidad	1
Descripción	Procesador Intel/AMD 1.6 GHz 1 GB memoria RAM DDR1 80 GB Disco Duro
Software base	Windows XP Professional

2.7 Plan de Pruebas. Requerimientos a probar

En esta sección se mostrarán los requisitos funcionales a probar. Por el módulo no ser tan complicado todos los requisitos de este serán objeto de prueba. Al finalizar las mismas se podrá tener una idea más clara de cuan correcto está el módulo, además de permitir conocer cuáles son las no conformidades

detectadas para poderlas resolver en la próxima iteración. Para una organización más detallada los requisitos se distribuyeron según la funcionalidad que cumplen en la aplicación.

Requisitos funcionales relacionados a las solicitudes de salidas de misión.

R02.001 Crear solicitud de salida de misión.

R02.002 Modificar solicitud de salida de misión.

R02.003 Eliminar una solicitud de salida de misión.

R02.004 Enviar solicitud de misión al ministerio.

R02.005 Enviar solicitud al MINCEX.

Requisitos funcionales relacionados al análisis de las solicitudes de viaje.

R02.006 Modificar solicitud de salida de misión.

R02.007 Procesar solicitudes de salida de misión.

R02.008 Aprobar solicitud de salida de misión.

R02.009 Cancelar solicitud de salida de misión.

R02.0010 Enviar informe a la embajada de Cuba en Venezuela.

R02.0011 Ver informe de solicitudes de salida de misión enviadas por el MINCEX.

R02.0012 Generar carta de aprobación de solicitudes de salida de misión.

R02.0013 Generar informe de viaje para las embajadas.

R02.0014 Exportar datos generados.

R02.0015 Visualizar solicitudes.

Requisitos funcionales relacionados al chequeo de correspondencia entre el listado del DIE y el registro de misiones.

R02.0016 Chequear correspondencia entre el listado del DIE y el registro de misiones.

R02.0017 Actualizar el listado de nombres.

Requisitos funcionales relacionados al informe de viaje.

R02.0018 Dar entrada al informe de viaje.

R02.0019 Cancelar la entrega de informe de viaje.

2.8 Plan de Pruebas. Estrategia de prueba de proyecto

La estrategia de prueba de software integra un conjunto de actividades que describen los pasos que hay que llevar a cabo en un proceso de prueba con el fin de obtener como resultado una correcta construcción

del software. A continuación se muestra un diagrama de flujo donde se muestran las acciones a realizar durante esta etapa.

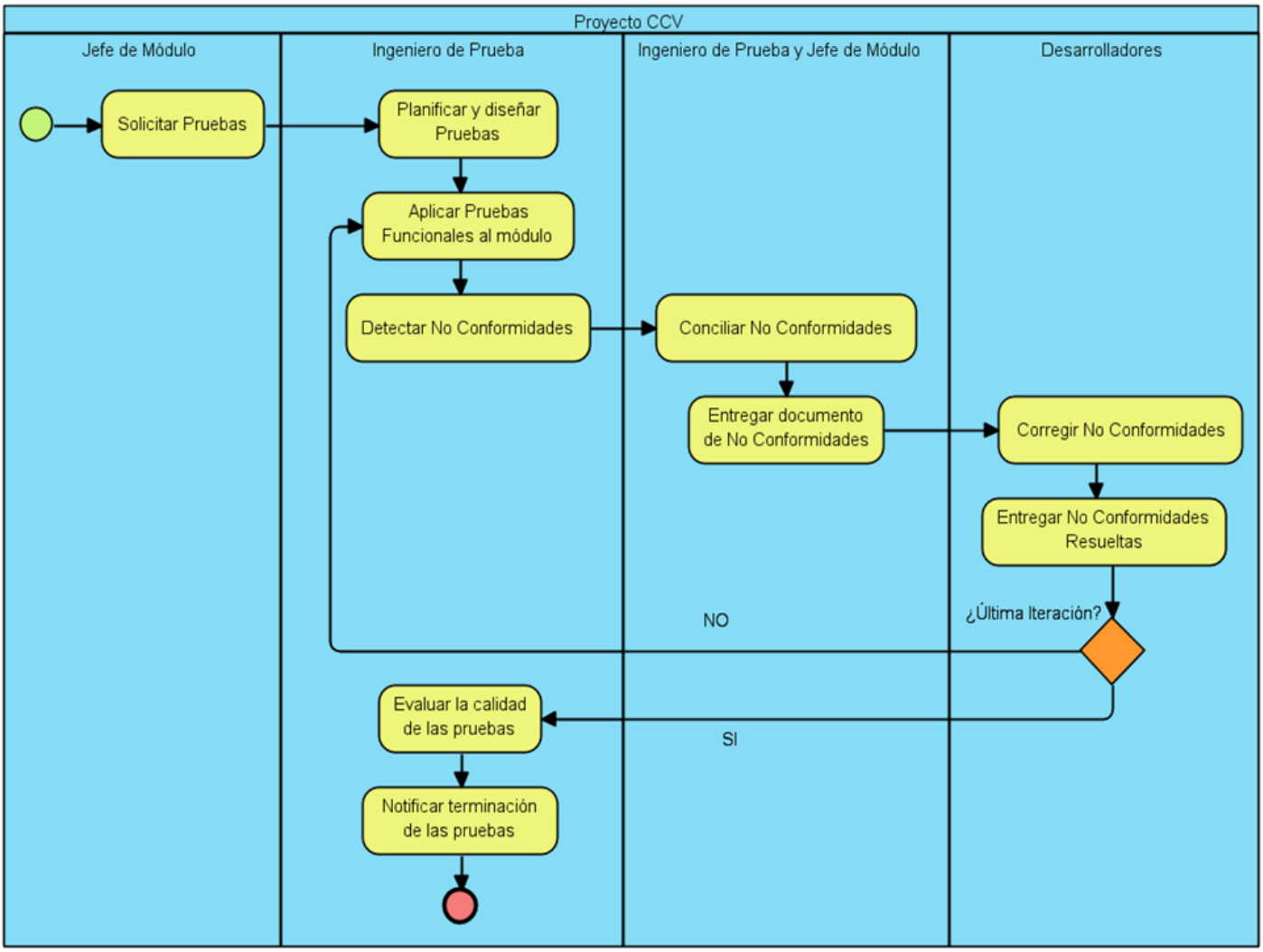

Figura 4 Flujo de trabajo para la etapa de prueba

La estrategia definida para Misiones inicia con la solicitud de las pruebas por parte del jefe de módulo al ingeniero de pruebas una vez que se haya culminado la implementación del mismo. Por su parte el ingeniero de pruebas lleva a cabo la planificación y diseño de las pruebas, lo que trae implícito una serie de actividades que la complementan como la definición del alcance de las pruebas y de los requerimientos a probar. Se seleccionan además la técnica y herramientas a emplear, en este caso se utilizará para las pruebas funcionales Selenium IDE como herramienta de automatización y como herramienta manual el diseño de Casos de Prueba.

Luego de haber organizado el proceso se procede a aplicar las pruebas funcionales a través de las pruebas diseñadas. A lo largo de esta actividad se van detectando las no conformidades, una vez culminada se conciliarán con el jefe de módulo para llegar a un consenso de los defectos encontrados para elaborar formalmente el artefacto No Conformidades, el cual será delegado a los desarrolladores. Los mismos corrigen los errores hallados y entregan las no conformidades resueltas.

Si esta actividad no constituye la iteración final se pasa nuevamente a la aplicación de las pruebas y los demás pasos correspondientes. Si es la última iteración se pasa a evaluar la calidad de las mismas, teniendo en cuenta los resultados obtenidos, la efectividad de las técnicas seleccionadas o si requiere de otras y ver si las pruebas realizadas cumplieron con los objetivos trazados.

Para culminar el proceso el ingeniero de pruebas notifica la culminación de las pruebas, esperando que el producto sea elevado a nivel de calidad de facultad y luego a nivel UCI en CALISOFT donde lo darían por liberado.

2.9 Plan de Pruebas. Evaluación de las pruebas

Una vez concluido el periodo de pruebas se procederá a evaluar dicho proceso, teniendo en cuenta una serie de criterios definidos por el ingeniero de prueba. Del cumplimiento de cada uno de ellos dependerá la evaluación final de las pruebas aplicadas al módulo.

Los criterios son los siguientes:

1. Asistencia de un 80 por ciento por parte del ingeniero de pruebas al laboratorio. Dando muestra del cumplimiento del cronograma de trabajo definido para las pruebas.

2. Cumplimiento de todas las actividades reflejadas en el Plan de Pruebas. Siendo de marcada importancia la correcta puesta en marcha de la estrategia de pruebas del proyecto ya que es la que influye de manera directa en el buen desarrollo de todo el proceso de pruebas.

3. Cantidad de no conformidades críticas detectadas al final de cada iteración. Estos defectos serán clasificados como Críticos si influyen de manera significativa en el sistema como el incumplimiento de un requisito y No Críticos si no afectan de forma directa al producto como el incumplimiento de un elemento de un requisito.

4. Impacto de los riesgos o inconvenientes aparecidos durante el desempeño de las actividades. Estos afectan directamente la planificación propuesta y pueden contribuir a la baja calidad en la realización de las tareas.

5. Resultado obtenido a partir de la aplicación de las listas de chequeo a los Casos de Prueba. Con dicha evaluación se verificará si realmente están bien elaborados y con ello la calidad de las pruebas aplicadas al módulo.

2.10 Plan de Pruebas. Cronograma

El cronograma previsto se define teniendo en cuenta las fechas en las cuales se llevará a cabo cada actividad del proceso de prueba, el mismo se modificará de existir algún elemento no previsto que provoque un cambio en el orden, duración o fecha de inicio de alguna tarea.

No.	Tarea	Fecha	Responsable	Observaciones
1	Planificación de las pruebas.	1-10-10	Daymara Eva Fariñas González	Realizar la planificación de las pruebas mediante la definición de su alcance, identificación de los requerimientos a probar entre otros aspectos.
2	Diseño de los Casos de Prueba.	10-10-10	Daymara Eva Fariñas González	Elaborar por cada caso de uso el diseño de caso de prueba correspondiente dejándolos listos para el comienzo de las pruebas.
3	Aplicación de las pruebas funcionales mediante los Casos de Prueba diseñados.	25-10-10	Daymara Eva Fariñas González	Aplicar las pruebas de tipo funcional al módulo según los Casos de Prueba diseñados, registrando los resultados obtenidos por cada una de ellas en cada iteración efectuada.
4	Evaluación de las pruebas.	10-11-10	Daymara Eva Fariñas	Emitir una evaluación de las pruebas según los

			González	resultados obtenidos para saber si se puede o no culminar el proceso.
5	Notificación de la culminación de las pruebas.	15-11-10	Daymara Eva Fariñas González	Notificar que se han culminado las pruebas para que el producto pueda pasar al siguiente nivel de revisión.

2.11 Artefacto: Datos de Prueba

Dentro del método de Caja Negra la técnica partición equivalente es una de las más efectivas pues permite examinar los valores válidos e inválidos de las entradas existentes en el software. Descubre de forma inmediata una clase de errores que por otra vía requerirían la ejecución de muchos casos antes de detectar el error genérico. Se dirige a la definición de los Casos de Prueba, los cuales serán vistos en el siguiente epígrafe, reduciendo de esta forma el número de clases de prueba que hay que desarrollar.

En esta sección se hará uso de esta técnica, mostrando los tipos de datos antes comentados, que pueden tomar los campos de las diferentes interfaces del módulo para comprobar las funcionalidades del mismo, estos serán definidos como clase de equivalencia válida y clase de equivalencia no válida. Se utilizarán datos que pueden ser empleados por los usuarios, además de aquellos que en algún momento podrían alterar el comportamiento del programa. Seguidamente se muestran tablas con los juegos de datos para cada caso de uso que contenga campos a llenar.

Gestionar Solicitud

		Interfaz: Gestionar Solicitud	
Campo (condición de entrada)	**Tipo**	**Clase de Equivalencia Válida**	**Clase de Equivalencia No Válida**
País destino	Conjunto con comportamiento distinto.	Cualquiera de los países del mundo.	No tiene
Entidad de Misión	Conjunto con comportamiento distinto.	“-Seleccione-” “Albet S.A.”	No tiene

Nombre del proyecto	Valor	Puede ser cadena de caracteres que incluyan: • Desde la letra A-Z (mayúsculas o minúsculas) • Números • Espacios	Caracteres o símbolos extraños que no están entre los definidos como: • : · $ % & / ( ) = ¿? * _ , ;
Fecha de ida	Valor	Solo números separados por una barra con el siguiente formato: • ## / ## / #### (día/mes/año) • Debe ser menor que la fecha de regreso.	Formato distinto al definido.
Fecha de regreso	Valor	Solo números separados por una barra con el siguiente formato: • ## / ## / #### (día/mes/año) • Debe ser mayor que la fecha de ida.	Formato distinto al definido.
Nombre y Apellidos	Valor	Puede ser cadena de caracteres que incluyan: • Desde la letra A-Z (mayúsculas o minúsculas) • Espacios	Caracteres o símbolos extraños que no están entre los definidos como: • : · $ % & / ( ) = ¿? * _ , ; • Números

Motivo del viaje	Valor	Puede ser cadena de caracteres que incluyan: • Desde la letra A-Z (mayúsculas o minúsculas) • Números • Espacios	Caracteres o símbolos extraños que no están entre los definidos como: • : · $ % & / ( ) = ¿? * _ , ;
Instituciones o personalidades a contactar	Valor	Puede ser cadena de caracteres que incluyan: • Desde la letra A-Z (mayúsculas o minúsculas) • Números • Espacios	Caracteres o símbolos extraños que no están entre los definidos como: • : · $ % & / ( ) = ¿? * _ , ;

Actualizar registro de misiones

	Interfaz: Actualizar registro de misiones		
Campo	**Tipo**	**Clase de Equivalencia Válida**	**Clase de Equivalencia No Válida**
Número de referencia	Valor	El número debe estar en el siguiente formato: • Dos números separados por un guión (–) seguido por cuatro números más. ## - ####	Formato distinto al definido.
Nota de rechazo	Valor	Puede ser cadena de caracteres que incluyan: • Desde la letra A-Z (mayúsculas o	No tiene caracteres inválidos. Admite todo lo que se le introduzca al

		minúsculas) • Números • Espacios • Caracteres extraños (: · $ % & / ( ) = ¿? * _ , ;)	campo.

Dar entrada al informe de viaje

	Interfaz: Dar entrada al informe de viaje		
Campo	**Tipo**	**Clase de Equivalencia Válida**	**Clase de Equivalencia No Válida**
No viajó	Conjunto con comportamiento distinto.	Se debe seleccionar en caso que no se haya viajado.	No tiene
Entregó informe	Conjunto con comportamiento distinto.	Se debe seleccionar en caso que se haya entregado el informe de viaje.	No tiene

2.12 Artefacto: Casos de Prueba

Para realizar el diseño de los Casos de Prueba según la técnica partición equivalente hay que basarse en la evaluación de las clases de equivalencia, vistas en las tablas anteriores, para una condición de entrada. Estos tienen como propósito identificar y comunicar las condiciones que se llevarán a cabo en la prueba. Especifican además una forma de probar el sistema, incluyen la entrada y salida con la que se probará y las condiciones bajo las que ha de probarse. Se describen también las clases válidas e inválidas que se pueden utilizar para realizar las pruebas del software.

Se realizaron los trece Casos de Prueba según los casos de uso correspondientes, teniendo en cuenta los requisitos de cada uno de ellos. A continuación se muestran algunos de los Casos de Prueba diseñados para la ejecución de las mismas. (Ver Anexo 2).

Caso de Prueba 1:

Actualizar registro de misiones

Descripción General del caso de uso:

Una vez creada la solicitud, es enviada al ministerio para esperar por su aprobación, aprobada la misma pasa al Ministerio del Comercio Exterior. En este nivel es procesada y aceptada dicha solicitud. Es entonces cuando se procede a actualizar el registro de misiones, ya que así quedaría oficialmente incluida una nueva solicitud de misión en espera a ser enviada a niveles superiores.

Condiciones de ejecución:

El usuario debe estar autenticado con los permisos necesarios y debe existir una solicitud que haya sido aceptada o rechazada.

Secciones a probar en el caso de uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Actualizar Registro	EC 1.1: Seleccionar la solicitud a actualizar	Se selecciona la solicitud que se desea actualizar.
	EC 1.2 Aceptar la solicitud	Se acepta la actualización de la solicitud.
	EC 1.3: Insertar número de referencia	Se inserta el número de referencia.
	EC 1.4: Rechazar la solicitud	Se rechaza la actualización de la solicitud.
	EC 1.5: Introducir nota de rechazo de solicitud	Se introduce el motivo del rechazo y el número de referencia de la solicitud.
	EC 1.6 Cancelar	Se cancela(n) la(s) operación(es) realizadas.
	EC 1.7 Existen datos incorrectos	Se introducen datos incorrectos.

	EC 1.8 Existen datos incompletos	Se dejan campos obligatorios en blanco.

Descripción de la Variable:

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Número de referencia	Campo de texto	No	Dos números separados por un guión (–) seguido por cuatro dígitos más, como se muestra en el formato: ## - ####
2	Nota de rechazo	Campo de texto	No	Admite cualquier tipo de combinación de letras caracteres y números.

Matriz de datos:

Escenario	Nota de rechazo	Número de referencia	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 1.1: Seleccionar la solicitud a actualizar	NA	NA	Se selecciona la solicitud.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Actualizar

EC 1.2 Aceptar la solicitud	NA	V 65-1234	Se acepta la solicitud.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Actualizar 5. Dar clic en el ícono de la flechita azul de la solicitud que desea actualizar 6. Aceptar
EC 1.3: Insertar número de referencia	NA	V	Permite insertar el número de referencia con el formato requerido.		1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Actualizar 5. Dar clic en el ícono de la flechita azul de la solicitud que desea actualizar 6. Aceptar

EC 1.3: Rechazar la solicitud	NA	NA	Se rechaza la solicitud y aparece la nota de rechazo.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Actualizar 5. Dar clic en el ícono de la flechita azul de la solicitud que desea actualizar 6. Rechazar
EC 1.4: Introducir nota de rechazo	V 71!$·%·&jgshgd***//)=	V 65-1234	Se introduce el motivo de rechazo de la solicitud.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Actualizar 5. Dar clic en el ícono de la flechita azul de la solicitud que desea actualizar 6. Rechazar

EC 1.5 Cancelar	NA	NA	Se cancela el procesamiento de la solicitud.		1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Actualizar 5. Dar clic en el ícono de la flechita azul de la solicitud que desea actualizar 6. Cancelar
EC 1.6 Existen datos incorrectos	I	V 65-1234	Se muestra un mensaje indicando que existen datos incorrectos.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Actualizar 5. Dar clic en el ícono de la flechita azul de la solicitud que desea actualizar 6. Rechazar / Aceptar y aparece para introducir el número de referencia 7. Introducir algún dato con el formato incorrecto
	V 71!$·%·&jgshgd***//)=	I 52-54			

EC 1.7 Existen datos incompletos	I	V 65-1234	Se muestra un mensaje indicando que existen datos incompletos.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Actualizar 5. Dar clic en el ícono de la flechita azul de la solicitud que desea actualizar 6. Rechazar / Aceptar y aparece para introducir el número de referencia 7. Dejar alguno de los campos en blanco
	V 71!$·%·&jgshgd***//)=	I			

Caso de Prueba 2:

Gestionar Solicitud

Descripción General del caso de uso:

Se inicia cuando se necesita crear una nueva solicitud de misión o modificar o eliminar una ya existente.

Condiciones de ejecución:

El usuario debe estar autenticado con los permisos necesarios.

Secciones a probar en el caso de uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1 : Crear Solicitud	EC 1.1: Crear una nueva solicitud exitosamente	Se crea una nueva solicitud insertando datos válidos.

	EC 1.2: Seleccionar la opción Cancelar	Mediante este escenario se cancela la creación de una nueva solicitud.
	EC 1.3: Existen datos incompletos	Al insertar datos pertenecientes a una nueva solicitud, al menos uno se encuentra en blanco.
	EC 1.4: Existen datos incorrectos	Al insertar datos pertenecientes a una nueva solicitud, existen campos incorrectos.
SC 2 : Eliminar Solicitud	EC 2.1 Eliminar una solicitud exitosamente	Se elimina la solicitud exitosamente.
SC 3: Modificar Solicitud	EC 3.1:Modificar una solicitud exitosamente	Se modifica la solicitud seleccionada exitosamente.
	EC 3.2: Existen datos incompletos	Al insertar datos pertenecientes a una nueva solicitud, al menos uno se encuentra en blanco.

Descripción de variable.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	País destino	Lista desplegable	No	Seleccionar las opciones de países que se muestran.
2	Entidad de Misión	Lista desplegable	No	Seleccionar las opciones que se muestran (Albet S.A.).

3	Nombre del proyecto	Campo de texto	No	Pueden ser letras o números, palabras separadas correctamente.
4	Fecha de ida	Campo de Selección	No	Solo números separados por una barra con el formato ## / ## / #### (día/mes/año).
5	Fecha de regreso	Campo de Selección	No	Solo números separados por una barra con el formato ## / ## / #### (día/mes/año).
6	Nombre y Apellidos	Campo de texto	No	Cadena de caracteres separados por un espacio. No admite caracteres extraños ni números.
7	Motivo del viaje	Campo de texto	No	Letras separadas por un espacio, números. No admite caracteres extraños.
8	Instituciones o personalidades a contactar	Campo de texto	No	Letras separadas por un espacio, números. No admite caracteres extraños.

Matriz de Datos

SC 1 Crear solicitud

Escenario	País destino	Entidad de Misión	Proyecto	Fecha ida	Fecha de Regreso	Nombre y Apellidos	Motivos del viaje.	Instituciones o personalidades a	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 1.1: Crear una nueva solicitud exitosamente	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc	El sistema permite crear una solicitud.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Nuevo
EC 1.2: Seleccionar la opción Cancelar	NA	NA	NA	NA	NA	NA	NA	NA	El sistema permite cancelar las operaciones realizadas.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Cancelar
EC 1.3: Existen datos incompletos	I	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc	El sistema muestra un mensaje informativo que faltan datos por llenar.	Satisfactorio.	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Seleccionar la opción Guardar dejando campos sin llenar

	V Venezuela	I	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	I	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	V CCV	I	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	I	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	I	V Snvasa	V Dfjfjbsd2324scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	I	V Dfjfjbsd2324scjsbc			

	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	I		Satisfactorio	
EC1.4 Existen Datos Incorrectos	NA	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc	El sistema muestra un mensaje indicando el error.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Seleccionar la opción Guardar dejando campos escritos de forma incorrecta
	V Venezuela	NA	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	I *foew //jk/*ef	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	V CCV	I Hfdf, 12/31/2010,	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd2324scjsbc		Satisfactorio	

	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	I Hfdf, 12/31/2010,	V Daymara Fariñas	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	I 45454, */()-hkhk	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfactorio	
	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	I */()-hkhk	V		Satisfactorio	
	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	I */()-hkhk		Satisfactorio	

SC 2 Eliminar Solicitud

Escenario	País destino	Entidad de Misión	Proyecto	Fecha ida	Fecha de Regreso	Nombre y Apellidos	Motivos del viaje.	Instituciones o personalidades a contactar.	Respuesta Esperada	Resultado de la Prueba	Flujo Central

Escenario	País destino	Entidad de Misión	Proyecto	Fecha ida	Fecha de Regreso	Nombre y Apellidos	Motivos del viaje.	Instituciones o personalidades a contactar.	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 2.1: Eliminar una solicitud exitosamente	NA	NA	NA	NA	NA	NA	NA	NA	El sistema elimina la solicitud deseada.	Satisfactorio.	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Seleccionar solicitud deseada 6. Eliminar
EC 2.2: Seleccionar la opción Cancelar	NA	NA	NA	NA	NA	NA	NA	NA	El sistema permite seleccionar la opción cancelar.	Satisfactorio.	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Cancelar

SC 3 Modificar Solicitud

Escenario	País destino	Entidad de Misión	Proyecto	Fecha ida	Fecha de Regreso	Nombre y Apellidos	Motivos del viaje.	Instituciones o personalidades a contactar.	Respuesta Esperada	Resultado de la Prueba	Flujo Central

EC 3.1 Modificar una solicitud exitosamente	NA	NA	NA	NA	NA	NA	NA	NA	El sistema permite modificar una solicitud correctamente.	Satisfactorio.	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Dar clic en el ícono del lapicito amarillo 6. Realizar las modificaciones correctamente y seleccionar la opción Guardar
EC 3.2: Existen datos incompletos	NA	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd 2324 scjsbc	El sistema muestra un mensaje indicando que existen campos por llenar.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Dar clic en el ícono del lapicito amarillo 6. Seleccionar la opción Guardar dejando campos sin llenar

	V Vene zuela	NA	V CC V	V 10/1 0/20 10	V 09/1 1/20 10	V Daym ara Fariñ as	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	
	V Vene zuela	V Albet S:A	I * foe w // jk/* ef	V 10/1 0/20 10	V 09/1 1/20 10	V Daym ara Fariñ as	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	
	V Vene zuela	V Albet S:A	V CC V	I Hfdf , 12/3 1/20 10, 12/1 3/20 10	V 09/1 1/20 10	V Daym ara Fariñ as	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	
	V Vene zuela	V Albet S:A	V CC V	V 10/1 0/20 10	I Hfdf , 12/3 1/20 10, 12/1 3/20 10	V Daym ara Fariñ as	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	
	V Vene zuela	V Albet S:A	V CC V	V 10/1 0/20 10	V 09/1 1/20 10	I 4545 4, */()- hkhk	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	

	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	I */()-hkhk	V			
	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	I */()-hkhk		Satisfactorio	
EC1.4 Existen Datos Incorrectos	NA	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd 2324 scjsbc	El sistema muestra un mensaje indicando el error.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Administrar 5. Dar clic en el ícono del lapicito amarillo 6. Seleccionar la opción Guardar dejando campos escritos de forma incorrecta
	V Venezuela	NA	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfactorio	

	V Vene zuela	V Albet S:A	I * foe w // jk/* ef	V 10/1 0/20 10	V 09/1 1/20 10	V Daym ara Fariñ as	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	
	V Vene zuela	V Albet S:A	V CC V	I Hfdf , 12/3 1/20 10, 12/1 3/20	V 09/1 1/20 10	V Daym ara Fariñ as	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	
	V Vene zuela	V Albet S:A	V CC V	V 10/1 0/20 10	I Hfdf , 12/3 1/20 10, 12/1	V Daym ara Fariñ as	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	
	V Vene zuela	V Albet S:A	V CC V	V 10/1 0/20 10	V 09/1 1/20 10	I 4545 4, */()- hkhk	V Snvasa	V Dfjfjbsd 2324 scjsbc		Satisfac torio	
	V Vene zuela	V Albet S:A	V CC V	V 10/1 0/20 10	V 09/1 1/20 10	V Daym ara Fariñ as	I */()- hkhk	V		Satisfac torio	

	V Venezuela	V Albet S:A	V CCV	V 10/10/2010	V 09/11/2010	V Daymara Fariñas	V Snvasa	I */()-hkhk		Satisfactorio	

Caso de Prueba 3:

Chequear correspondencia entre el listado del DIE y el registro de misiones

Descripción General del caso de uso:

El caso de uso inicia cuando el técnico recibe el listado del DIE y necesita comprobar que las personas que aparecen en él se correspondan con las que están archivadas en el registro de misiones.

Condiciones de ejecución:

El usuario debe estar autenticado con los permisos necesarios.

Secciones a probar en el Caso de Uso.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Chequear correspondencia entre el listado del DIE y el registro de misiones.	EC 1.1: Cargar fichero	Mediante este escenario se carga el fichero con el listado de correspondencia entre el listado del DIE y el registro de misiones.
	EC 1.2: Seleccionar opción Cancelar	Mediante este escenario el sistema cancela la opción de subir el documento.
	EC 1.3: Procesar	Mediante este escenario el sistema muestra el listado de las personas que realizaron las solicitudes de viajes y las de la correspondencia DIE.

	EC 1.4: Terminar	Mediante este escenario el sistema te permite abrir el fichero actualizado con las personas que han viajado y no está registrada en el sistema.
	EC 1.5: Existen campos incompletos	Se muestra un mensaje indicando que se debe seleccionar el país destino.

Descripción de variable.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	País de destino	Lista desplegable	No	Nombre propio de un país.

Matriz de Datos

Escenario	País de destino	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 1.1: Cargar fichero	NA	El sistema carga el fichero.	Satisfactorio	1. Módulos del sistema 2. Misiones 3. Correspondencia 4. Cargar fichero 5. Examinar 6. Seleccionar el fichero 7. Cargar fichero

Escenario	País de destino	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 1.2: Seleccionar opción Cancelar	NA	El sistema cancela la operación de cargar el fichero.	Satisfactorio	1. Módulos del sistema 2. Misiones 3. Correspondencia 4. Cancelar
EC 1.3: Procesar	V Venezuela	El sistema muestra el listado de las personas que solicitan el viaje y de las de la correspondencia DIE.	Satisfactorio	1. Módulos del sistema 2. Misiones 3. Correspondencia 4. Procesar
EC 1.4: Terminar	V Venezuela	El sistema muestra un fichero.	Satisfactorio	1. Módulos del sistema 2. Misiones 3. Correspondencia 4. Procesar 5. Terminar
EC 1.4: Existen campos incompletos	I	El sistema muestra un mensaje indicando el error.	Satisfactorio	1. Módulos del sistema 2. Misiones 3. Correspondencia 4. No se selecciona algún país de destino

Caso de Prueba 4:

Dar Entrada al informe de viaje

Descripción General del caso de uso:

El caso de uso inicia cuando llega un informe de viaje a la Secretaría Técnica y el técnico tiene que darle entrada.

Condiciones de ejecución:

El usuario debe estar autenticado con los permisos necesarios y debe existir al menos una solicitud que requiera informe de viaje y que no se haya entregado.

Secciones a probar en el caso de uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Dar entrada al informe de viaje	EC 1.1: Seleccionar la solicitud	Se selecciona la solicitud deseada para dar entrada al informe.
	EC 1.2: Seleccionar la(s) opción(es) que indica(n) la acción realizada	Se selecciona la opción que indica la acción que se realizó.
	EC 1.3: Aceptar	Se acepta el informe y se le da la entrada.
	EC 1.4: Cancelar	Se cancela el informe y no se le da entrada.

Descripción de la Variable

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	No viajó	Campo de selección	Sí	Se marca si viajó.
2	Entregó informe	Campo de selección	Sí	Se marca si entregó el informe.

Matriz de Datos

Escenario	No viajó	Entregó informe	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 1.1: Seleccionar la solicitud	NA	NA	Se selecciona la solicitud(es).	Satisfactorio.	1. Ir a Módulos del Sistema 2. Misiones 3. Informe 4. Dar entrada 5. Dar clic en el ícono del lapicito amarillo de la solicitud que se desea
EC 1.2: Seleccionar la(s) opción(es) que indica(n) la acción realizada(s).	V	V	Se selecciona(n) la(s) opción(es) que indica(n) la acción(es) realizada(s).	Satisfactorio.	1. Ir a Módulos del Sistema 2. Misiones 3. Informe 4. Dar entrada 5. Dar clic en el ícono del lapicito amarillo de la solicitud que se desea 6. Marcar al menos una de las acciones realizadas

Escenario	No viajó	Entregó informe	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 1.3: Aceptar	NA	NA	Se acepta la operación realizada para dar entrada al informe.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Informe 4. Dar entrada 5. Dar clic en el ícono del lapicito amarillo de la solicitud que se desea 6. Marcar al menos una de las acciones realizadas 7. Seleccionar opción Aceptar
EC 1.4: Cancelar	NA	NA	Se cancela la operación realizada para dar entrada al informe.	Satisfactorio.	1. Ir a Módulos del Sistema 2. Misiones 3. Informe 4. Dar entrada 5. Dar clic en el ícono del lapicito amarillo de la solicitud que se desea 6. Marcar al menos una de las acciones realizadas 7. Seleccionar opción Cancelar

Caso de Prueba 5:

Enviar solicitud a la embajada

Descripción General del caso de uso:

El caso de uso inicia cuando el técnico necesita enviar un informe de las solicitudes aprobadas a la embajada de Cuba en Venezuela.

Condiciones de ejecución:

El usuario debe estar autenticado con los permisos necesarios y deben existir solicitudes en estado 'aprobadas'.

Secciones a probar en el caso de uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Enviar solicitud	EC 1.2: Seleccionar una solicitud	Se selecciona una solicitud para enviarla a la embajada.
	EC 1.3: Enviar solicitud al ministerio exitosamente	Mediante este escenario se envía la solicitud a la embajada correctamente.
	EC 1.4: Cancelar el envío de la solicitud	Se cancela el envío de la solicitud a la embajada.

Matriz de datos

Escenario	Respuesta Esperada	Resultado de la Prueba	Flujo Central

Escenario	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 1.1 Seleccionar una solicitud	El sistema permite seleccionar una solicitud.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Enviar 5. Dar clic en el ícono del lapicito amarillo de la solicitud que desea enviar
EC 1.2: Enviar solicitud al ministerio exitosamente	El sistema permite enviar solicitud a la embajada.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Enviar 5. Dar clic en el ícono del lapicito amarillo de la solicitud que desea enviar 6. Selecciona la opción Enviar
EC 1.3: Cancelar el envío de la solicitud	El sistema permite cancelar el envío de una solicitud a la embajada.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Solicitud 4. Enviar 5. Dar clic en el ícono del lapicito amarillo de la solicitud que desea enviar 6. Selecciona la opción Cancelar

Caso de Prueba 6:

Generar carta de aprobación

Descripción General del caso de uso:

El caso de uso inicia cuando el técnico necesita generar una carta de aprobación de solicitudes de salida de misión.

Condiciones de ejecución:

El usuario debe estar autenticado con los permisos necesarios y debe de existir al menos una solicitud en estado en 'revisión'.

Secciones a probar en el caso de uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1 Generar carta de aprobación	EC 1.1: Seleccionar la(s) solicitud(es) a reportar	Se selecciona la solicitud que se le quiere hacer el reporte.
	EC 1.2 Generar el reporte de la(s) solicitud(es) exitosamente.	Se genera el reporte correctamente.

Matriz de datos

Escenario	Respuesta Esperada	Resultado de la Prueba	Flujo Central

Escenario	Respuesta Esperada	Resultado de la Prueba	Flujo Central
EC 1.1: Seleccionar la(s) solicitud(es) a reportar	Se selecciona la(s) solicitud(es) a reportar.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Reporte 4. Carta de aprobación 5. Seleccionar la(s) solicitud(es) a reportar
EC 1.2 Generar el reporte de la(s) solicitud(es) exitosamente	Se acepta la solicitud.	Satisfactorio	1. Ir a Módulos del Sistema 2. Misiones 3. Reporte 4. Carta de aprobación 5. Seleccionar la(s) solicitud(es) a reportar 6. Seleccionar la opción Generar

2.13 Artefacto: No Conformidades

Las no conformidades son defectos encontrados durante la etapa de ejecución de las pruebas. La detección de los mismos es de gran importancia para el software, ya que mediante su corrección se podría asegurar una mejor calidad en el producto que se desarrolla. Mientras menos defectos se encuentren en el software mayor será la satisfacción del cliente.

Para recoger todas las fallas descubiertas de una forma más organizada existe el documento No Conformidades. En el mismo se describen los aspectos a tener en cuenta a la hora de realizar el diseño de

las pruebas, incidencias en el momento de su desarrollo y otros aspectos relevantes. De los elementos probados se muestra una lista y de los no probados las causas o incidencias que impidieron esta acción. También cuenta con la sección Registro de defectos y dificultades detectados.

Este artefacto es fundamental ya que el hecho de tener todos los defectos agrupados significa que con solo tomar la planilla se tendrán los errores a ser corregidos en una próxima iteración y sin lugar a dudas un ahorro en tiempo.

Luego de haber aplicado los Casos de Prueba para probar las funcionalidades del sistema, se realizó un análisis de los errores encontrados. La revisión del sistema se realizó de forma minuciosa para encontrar una mayor cantidad de estos y evitar un incorrecto funcionamiento del módulo que en parte imposibilitarían una precisa comprensión y entendimiento del sistema. Una vez obtenidos los errores, fueron recogidos en el documento No Conformidades antes detallado.

A continuación se muestra la tabla Registro de defectos y dificultades detectados como parte de los resultados obtenidos en las iteraciones de las pruebas realizadas al módulo Misiones. (Ver Anexo 3).

Registro de defectos y dificultades detectados

Elemento	No	Descripción de la No Conformidad	Ubicación de la No Conformidad	Etapa de Detección	Significativa	No Significativa	Recomendación	Estado NC	Respuesta del Equipo de Desarrollo
Aplicación	1	Cuando se va a modificar una solicitud con estado rechazada por secretaría aparece un mensaje de error.	Misiones/solicitud/ Administrar/como usuario pmisiones.	Revisión Iteración 1	X			PD 7-11-2010	
	2	En el campo fecha solo debería tomar los datos del componente que da	Ir a Módulos del Sistema/Misiones/ Solicitud/Administrar/Nuevo	Revisión Iteración 1	X			PD 7-11-2010	

		la posibilidad de escoger la fecha.							
Aplicación	3	Se validan los demás campos de la solicitud cuando solo debería validar el nombre de la persona a insertar.	Ir a Módulos del Sistema/Misiones/ Solicitud/Administrar/Nuevo	Revisión Iteración 1	X			PD 7-11-2010	
Aplicación	4	No valida cuando en el campo nombre le introduces un dato numérico.	Ir a Módulos del Sistema/Misiones/ Solicitud/Administrar/Nuevo	Revisión Iteración 1	X			PD 7-11-2010	
Aplicación	5	En la aplicación al acceder por el usuario embmisiones, la opción reportes no muestra nada.	Módulos del Sistema/Misión/Reportes	Revisión Iteración 1	X			PD 7-11-2010	
Aplicación	6	Cuando se va a crear una nueva solicitud de misión y se selecciona la opción Aceptar da un error que muestra un mensaje “Error en flujo” y no deja crear la solicitud.	Ir a Módulos del Sistema/Misiones/ Solicitud/Administrar/Nuevo/Aceptar	Revisión Iteración 1	X			PD 7-11-2010	

Aplicación	7	Cuando se va a dar entrada al informe de viaje esta acción se realiza aunque no se seleccione alguna de las acciones realizadas obligatorias.	Ir a Módulos del Sistema/Misiones/Informe/Dar entrada	Revisión Iteración 1	X			PD 7-11-2010	
Aplicación	8	Al visualizar la solicitud no muestra el estado en que se encuentra la misma.	Ir a Módulos del Sistema/Misiones/ Solicitud/Ver Detalles	Revisión Iteración 1	X			PD 7-11-2010	
Aplicación	9	Cuando se va a actualizar registro el campo número de referencia admite caracteres extraños.	Ir a Módulos del Sistema/Misiones/ Solicitud/Actualizar /Aceptar	Revisión Iteración 1	X			PD 7-11-2010	
Aplicación	10	A la hora de cargar un fichero en la acción Chequear correspondencia entre el listado del DIE y el registro de misiones permite cargar un documento con cualquier formato.	Módulos del sistema/Misiones/Correspondencia/Cargar fichero	Revisión Iteración 2	X			PD 8-11-2010	
Aplicación	11	En la acción Chequear correspondencia	Módulos del sistema/Misiones/ Correspondencia/	Revisión Iteración 2	X			PD 8-11-2010	

		entre el listado del DIE y el registro de misiones, cuando se selecciona la opción Procesar no muestra el listado de las personas que solicitan el viaje.	Procesar						

Los resultados evidencian un total de once no conformidades significativas y de tipo aplicación, nueve de ellas detectadas en la primera iteración y dos en la segunda.

2.14 Conclusiones

Luego de la realización de este capítulo se pudo arribar a las siguientes conclusiones:

- ✓ El Plan de Pruebas sirve de orientación para el equipo de trabajo y constituye un artefacto fundamental para que se lleve a cabo un buen proceso de pruebas.
- ✓ El rol que lleva a cabo las actividades correspondientes al proceso de prueba debe tener responsabilidad con las tareas que le corresponden y mantener una buena organización en el trabajo para que el mismo se realice de forma eficiente y sin pérdida de tiempo.
- ✓ Un correcto escenario de pruebas ayuda a tener los recursos del sistema correctamente administrados y tener una visión más clara del desempeño del software en su entorno real.
- ✓ Se deben dejar plasmados los requisitos a probar en el Plan de Pruebas para evitar probar otros que no sean objetivo del proceso de prueba a realizar y que no se olvide ninguno de los planeados.
- ✓ Es de vital importancia una buena definición y aplicación de la estrategia de pruebas a seguir ya que la misma sirve de guía y ayuda para garantizar una mayor calidad del proceso y por ende una mayor calidad del producto final.
- ✓ El cronograma de prueba debe realizarse para tener presente los días que serán desempeñadas las diferentes tareas, ya que su cumplimiento en tiempo y forma posibilita una mejor organización de todo el personal involucrado en el proceso.
- ✓ Los datos de pruebas sirven para comprobar las funcionalidades del módulo y deben quedar bien claros, abarcando todas las posibles formas que el usuario pudiera introducir.

- ✓ Se diseñaron las pruebas de tipo funcional con el método de Caja Negra que cubrieron la mayor cantidad de combinaciones posibles para la detección de los defectos.

Capítulo 3

Análisis de los resultados

3.1 Introducción

En este capítulo se aplican listas de chequeo al Plan de Pruebas para garantizar que el mismo esté completo y a los Casos de Prueba para garantizar que se han especificado correctamente y que tratan todos los requisitos de prueba. Además se muestran los resultados obtenidos luego de un proceso de liberación y aceptación por parte del cliente.

3.2 Evaluación del Plan de Pruebas

Es de vital importancia que el Plan de Pruebas generado contenga todos los elementos que la plantilla requiere y que estén correctamente, pues el mismo constituye un factor clave para lograr un correcto proceso de pruebas. Para ello se le aplica listas de chequeo que tienen en cuenta una serie de indicadores de los cuales se muestra en el presente documento el atributo Evaluación pues es el que influye directamente en la evaluación final del artefacto.

En la revisión a cada indicador se le otorga uno (1) en caso de mal, cero (0) en caso de que el elemento revisado no presente errores o NP para especificar que el indicador a evaluar no se puede aplicar. (Ver Anexo 4).

Indicadores a evaluar de la lista de comprobación	Evaluación
ESTRUCTURA DEL DOCUMENTO	
¿Está el documento acorde con la plantilla estándar del proyecto o del expediente de proyecto?	0
¿Contiene las secciones obligatorias definidas en el expediente? (Ver Expediente de Proyecto 2.0)	0
ELEMENTOS DEFINIDOS POR LA METODOLOGÍA	
¿Están descritos en el alcance los Proyectos con los que se involucra?	NP
¿Se encuentran todas las Definiciones, Acrónimos y Abreviaturas?	1
¿Se encuentra la descripción del equipo de probadores, por quienes está compuesto y la responsabilidad de cada miembro?	0
¿Aparece la descripción del escenario en el que se ejecutarán las pruebas?	0

¿Aparece el Modelo de Despliegue del sistema?	0
¿Aparecen todos los recursos del sistema?	0
¿El listado de requerimientos a probar coincide con los identificados en el documento de requerimientos?	0
¿Se describe el flujo de trabajo que se utilizará para la ejecución de las pruebas?	0
¿Se describe la clasificación de las No Conformidades en la sección Evaluación de las pruebas?	0
¿Se describen los pedidos de cambios en la sección Evaluación de las pruebas?	NP
¿Se describen las listas de chequeo en la sección Evaluación de las pruebas?	0
¿El Cronograma de ejecución de las pruebas contiene las tareas, fecha, responsables, participantes y observaciones de la misma?	0
SEMÁNTICA DEL DOCUMENTO	
¿En el documento de Caso de Prueba se han identificado errores ortográficos?	1
¿Se entiende claramente lo que se ha especificado en el documento de Caso de Prueba?	0
¿El número de página que aparece en el índice coincide con el contenido que se refleja realmente en dicha página?	0
¿El total de páginas que aparecen en las reglas de confidencialidad coincide con el total de páginas que posee el documento?	1

Después de haber aplicado la lista de chequeo al artefacto Plan de Pruebas se puede concluir que el mismo cumple con la mayoría de los indicadores tales como los que evalúan la estrategia de pruebas, descripción de roles y responsabilidades, especificación de los requerimientos a probar, el diagrama y recursos del sistema, entre otros aspectos. Tres indicadores fueron evaluados de uno ya que no se encontraban explícitos en el documento como Definiciones, Acrónimos y Abreviaturas, errores ortográficos en algunos Casos de Prueba, y el total de páginas que aparecían en las reglas de confidencialidad no coincidían con el total de páginas del documento. Además dos de los indicadores fueron evaluados de No Procede.

Estos detalles al no afectar la correcta ejecución del proceso de pruebas no repercutieron en la evaluación general del artefacto, la cual fue de Bien.

3.3 Evaluación de los Casos de Prueba

Es importante que los Casos de Prueba cuenten con la calidad requerida en su diseño, ya que de ello dependen los resultados satisfactorios que ayudan a mejorar la funcionalidad del software. Para comprobar que los mismos se han especificado de forma correcta se aplica la lista de chequeo correspondiente al diseño de Casos de Prueba.

La lista de chequeo que a continuación se muestra verifica varios aspectos como la estructura del documento, los elementos definidos por la metodología (diseño de la prueba), la realización de la prueba y la semántica del documento. Permitiendo además recoger los puntos eficientes e ineficientes que tienen los elementos chequeados. La tabla que se presenta solo contiene el atributo Evaluación, el cual será medido con valores de uno (1) en caso de mal, cero (0) en caso de que el elemento revisado no presente errores o NP para especificar que el indicador no se puede aplicar en ese caso.

Indicadores a evaluar de la lista de comprobación	Evaluación
ESTRUCTURA DEL DOCUMENTO	
¿Está el documento de Caso de Prueba acorde con los aspectos siguientes? Nombre del Proyecto, Nombre del Módulo, Versión del proyecto, Nombre del Caso de Prueba, Versión de Caso de Prueba, Control de versiones, Tabla de contenido, Descripción General, Condiciones de Ejecución, Secciones a probar en el Caso de Uso, Descripción de variables, Matriz de Datos, Registro de defectos y dificultades detectados, Anexos.	0
ELEMENTOS DEFINIDOS POR LA METODOLOGÍA (DISEÑO DE LA PRUEBA)	
¿La descripción general del caso de prueba coincide con el resumen del Caso de Uso? (Ver descripción del Caso de Uso)	0
¿Las condiciones de ejecución del Caso de Prueba coinciden con las pre-condiciones del Caso de Uso? (Ver descripción del Caso de Uso)	0
¿En el Caso de Prueba se han especificado la sección o las secciones que tiene el Caso de Uso? (Ver descripción del Caso de Uso)	0
¿En el Caso de Prueba se han especificado todos los escenarios de pruebas por cada sección? (Ver descripción del Caso de Uso)	0
¿Si un escenario se relaciona con una entrada de datos, existe un escenario para validar que los datos sean correctos?	0
¿Si un escenario se relaciona con una entrada de datos, existe un escenario para	0

verificar que no falten datos obligatorios?	
¿En el Caso de Prueba aparece una descripción de la funcionalidad por cada escenario?	0
¿Si existen variables (datos de entrada o salida) se han definido sus nombres en la tabla de matriz de datos?	0
¿Si existen variables (datos de entrada o salida) se han definido sus clasificaciones (de acuerdo al tipo de dato)?	0
¿Si existen variables (datos de entrada o salida) se ha definido si aceptan valores nulos o no?	0
¿Si existen variables (datos de entrada o salida), se ha hecho una descripción de las mismas, especificando las condiciones que debe cumplir (ejemplo: La variable carnet de identidad debe tener 11 dígitos)?	0
¿Por cada escenario del Caso de Prueba se han especificado todas las variables (datos de entrada o salida) asociadas al mismo? (Ver descripción del Caso de Uso)	0
¿Por cada escenario que contenga variables (datos de entrada o salida) se ha especificado el estado que las mismas pueden tomar (Válido, Inválido, No Aplica)?	0
¿Si el escenario implica que las variables (datos de entrada o salida) tomen valor inválido, existe una sola variable con estado inválido y todas las demás con estado válido en cada combinación?	0
¿En el Caso de Prueba, el flujo central del escenario especifica el camino para probarlo?	0
¿Se ha especificado la respuesta del sistema para cada escenario del Caso de Prueba?	0
¿Se han colocado en la matriz de datos del Caso de Prueba los valores que toman las variables (datos de entrada o salida) en cada escenario del mismo?	0
SEMÁNTICA DEL DOCUMENTO	
¿En el documento de Caso de Prueba se han identificado errores ortográficos?	1
¿Se entiende claramente lo que se ha especificado en el documento de Caso de Prueba?	0
¿El número de página que aparece en el índice coincide con el contenido que se refleja realmente en dicha página?	0
¿El total de páginas que aparecen en las reglas de confidencialidad coincide con el total de páginas que posee el documento?	1

Durante la aplicación de la listas de chequeo a los trece Casos de Prueba realizados en el presente trabajo, se encontraron dos indicadores evaluados de mal, pues se detectó un error ortográfico en un caso de prueba y el total de páginas que aparecían en las reglas de confidencialidad no coincidían con el total de páginas del documento. Estos errores identificados fueron corregidos y en la próxima iteración no se volvieron a detectar. Los mismos no afectan la esencia del diseño de un caso de prueba (conjunto de entradas, condiciones de ejecución, y resultados esperados) por lo que la evaluación general del artefacto fue de Bien. (Ver Anexo 5).

3.4 Evaluación del proceso de pruebas aplicado al módulo Misiones

Teniendo en cuenta los criterios de evaluación definidos en el capítulo anterior para el proceso de pruebas, se puede concluir que el mismo fue satisfactorio. El ingeniero de prueba tuvo un 100 por ciento de asistencia al laboratorio y cumplió con todas las actividades reflejadas en el Plan de Pruebas. Como resultado de la aplicación de los Casos de Prueba a los casos de uso en una primera y segunda iteración quedaron registradas once no conformidades. Todas fueron clasificadas de tipo Aplicación pues evidenciaban problemas de validación en la entrada de datos y de las funcionalidades que no estaban correctamente implementadas y que influían en la ejecución de otras. Además fueron definidas como críticas (significativas) ya que afectaban el correcto funcionamiento del módulo. Todas las no conformidades fueron entregadas al equipo de desarrollo para su corrección. En una segunda iteración se pudo verificar que se le dio solución a todos los errores encontrados y que no aparecieron nuevos que pudieran introducirse en la etapa de corrección o no detectados en la revisión anterior. Durante el proceso de pruebas no aparecieron riesgos que pudieran afectar el desempeño de las actividades. La aplicación de las listas de chequeo a los Casos de Prueba diseñados arrojó resultado de Bien, pudiendo afirmar que el módulo cuenta con la mayoría de los requerimientos a probar y que gran parte de las funcionalidades se realizan de manera correcta.

Por todo lo antes expuesto se puede comprobar que se cumplió el objetivo propuesto de desarrollar un correcto proceso de pruebas al módulo Misiones, con vista a la obtención del mayor número de faltas existentes en el mismo. Permitiendo lograr con su posterior eliminación la obtención de una aplicación con el mínimo de defectos y mayor calidad.

3.5 Proceso de liberación y aceptación con el cliente

Una vez concluida la etapa de pruebas del módulo Misiones a nivel proyecto, el producto de manera íntegra pasó a las revisiones de calidad a nivel de facultad y luego a la empresa CALISOFT (Centro

Nacional de Calidad de Software), única de su tipo en Cuba. Esta luego de proporcionar los servicios de pruebas emitió el Acta de Liberación de Productos Software del módulo (Ver Anexo 6), quedando el mismo listo para las pruebas de aceptación. Estas últimas se realizaron de manera satisfactoria donde ambas partes cumplieron con el flujo de trabajo definido en el Plan de Pruebas de Aceptación (Ver Anexo 7) firmado previamente por Janet Fernández Padilla, Jefa del Departamento de Venezuela, Asnier Enrique Góngora Rodríguez, Especialista de CALISOFT y Danaysa Macías Hernández, Jefa del Proyecto CCV.

3.6 Conclusiones

En este capítulo se realizó el análisis de los resultados obtenidos durante la realización de las pruebas al módulo Misiones del sistema CCV. Luego de culminar el mismo se arriba a las siguientes conclusiones:

- ✓ Las listas de chequeo permitieron otorgarle una evaluación a los artefactos Plan de Pruebas y Casos de Prueba, atendiendo a diversos parámetros de manera individual para verificar la efectividad de cada uno de ellos.
- ✓ La corrección de las no conformidades detectadas durante esta etapa permitió que el módulo fuera evaluado satisfactoriamente, quedando libre de errores y cumpliendo con los requerimientos deseados por el cliente. Esto permitió que el mismo fuera liberado a nivel UCI por la empresa CALISOFT.
- ✓ El Plan de Pruebas de Aceptación evidenció la satisfacción del cliente con el módulo Misiones.

CONCLUSIONES GENERALES

Al terminar el presente trabajo se arriba a las siguientes conclusiones:

- ✓ Los artefactos Plan de Pruebas, Casos de Prueba, Datos de Prueba y No Conformidades permitieron la realización de las pruebas al módulo de manera satisfactoria.
- ✓ La estrategia de prueba organizó la forma de trabajo del ingeniero de prueba y el equipo de desarrollo.
- ✓ La aplicación de las listas de chequeo a los artefactos Plan de Pruebas y Casos de Prueba evidenciaron algunos indicadores evaluados de mal lo que hizo posible la corrección de los mismos y con esto su completitud.
- ✓ Los resultados obtenidos en las pruebas sirvió para otorgarle una evaluación final al módulo Misiones.
- ✓ El proceso de pruebas a nivel de proyecto sirvió de filtro para las futuras revisiones a nivel de facultad y CALISOFT.
- ✓ El proceso de pruebas aplicado al módulo Misiones contribuyó a la obtención del producto CCV con calidad y con ello la aceptación del cliente.

RECOMENDACIONES

Debido a la importancia que lleva el proceso de prueba para la corrección de errores en una aplicación se recomienda:

- ✓ Fomentar en los proyectos productivos la realización de un proceso de pruebas desde el inicio del mismo hasta su culminación, para garantizar que se encuentren la mayor cantidad de defectos en todo el ciclo de vida del software.
- ✓ Tener en cuenta para próximos desarrollos la aplicación de métodos de satisfacción del cliente desde la etapa inicial del proyecto ya que el cliente estará satisfecho no sólo con la calidad de los servicios que recibe sino cuando los servicios cubren o exceden sus expectativas.
- ✓ Aplicar las pruebas de Caja Blanca a otras aplicaciones si cuentan con el tiempo suficiente y flexibilidad en el cronograma, ya que esto aseguraría de forma general la calidad del producto.

BIBLIOGRAFÍA

2003. adimen. *adimen.* [En línea] 2003. http://adimen.si.ehu.es/~rigau/teaching/EHU/ISHAS/Curs2007-2008/Apunts/IS.14.pdf.

2005. alarcos. [En línea] 2005. http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema09.pdf.

Beck, Kent. 2000. *Extreme Programming Explained.* Canadá : Adison-Wesley, 2000. ISBN.

Borrero, Lucia. 2003. *Tecnologias de la informacion en internet.* Colombia : Norma, 2003. ISBN.

Davis, Alan. 1995. *201 Principles of Software Development.* Universidad de California : McGraw-Hill, 1995. ISBN 0070158401, 9780070158405.

Edumilis Méndez, María Pérez, Luis Mendoza. 2007. *Aplicación de un Método para Especificar Casos de Prueba de Software en la Administración Pública.* 2007.

EVA, Entorno Virtual de Aprendizaje. 2011. Documentación sobre pruebas. *EVA, Entorno Virtual de Aprendizaje.* [En línea] 2011. [Citado el: 14 de enero de 2011.] http://eva.uci.cu/mod/resource/view.php?id=14103.

Guzmán, René Fernández. 2008. *VIRTUALIZACIÓN DE HARWARE COMO HERRAMIENTA PARA PRUEBAS DE SOFTWARE.* 2008.

IBM, Corporation. 2006. Rational Unified Process. [En línea] 2006.

IEEE. 1990. *IEEE Standard Glossary of Software Engineering Terminology.* 1990.

Internacional, Holding latinoamericano: Hista. 2007. Hista International. [En línea] 2007. [Citado el: 3 de enero de 2011.] http://www.histaintl.com/servicios/consulting/rup.php.

Jacobson, Ivar, Booch, G y Rumbaugh, J. 2003. *Proceso unificado del desarrollo del software.* Madrid : Pearson education, 2003. 84-7829-036-2.

Jakarta, Corporation. 2010. Jakarta. *Jakarta.* [En línea] Apache Software Foundation, 2010. [Citado el: 10 de enero de 2011.] http://jakarta.apache.org/jmeter/.

Javier Tuya, Isabel Ramos Román, Javier Dolado Cosín. 2007. *Técnicas cuantitativas para la gestión en la ingeniería del software.* España : Netbiblo, 2007. ISBN 8497452046, 9788497452045.

Jimenez, Darwin. 2004. *Modelos de Pruebas de Software.* 2004.

Juristo, Natalia. 2006. *TÉCNICAS DE EVALUACIÓN DE SOFTWARE.* 2006.

Kruchten, Philippe. 2004. *The Rational Unified Process an Introduction.* Canadá : Pearson Education, 2004. ISBN 0321197704.

Mañas, Jose A. 1994. *Prueba de Programas. Pruebas de Integracion .* 1994.

María Garzón Villar, María Sampalo de la Torre, Esteban Leyva Cortés, Ignacio Prieto Tinoco. 2007. *Informática. Temario A. Volumen II. Profesores de Educación Secundaria E-book.* Sevilla,España : MAD-Eduforma, 2007. ISBN 8466511504, 9788466511506.

—. 2003. *Informática. Temario A. Volumen Iv. Profesores de Educación Secundaria Ebook.* Sevilla, España : Mad, S.L, 2003. ISBN 8466506098, 9788466506090.

Myers, Glenford. 1979. *The art of software testing.* Canadá : Jhon Wiley & Sons, Inc, 1979. 1ra Edición.

—. 2004. *The art of sotware Testing.* Canadá : Jhon Wiley & Sons, Inc, 2004. 2da Edición.

Oré, Alexander. 2009. calidadysoftware. *calidadysoftware.* [En línea] 2009. [Citado el: 20 de enero de 2011.] http://www.calidadysoftware.com/testing/pruebas_unitarias2.php.

Patton, Ron. 2005. *Software Testing, Segunda Edición.* Universidad de California : Sams Publishing, 2005. ISBN 0672327988, 9780672327988.

Pello, Javier. 2005. SoftQaNetwork. *SoftQaNetwork.* [En línea] 2005. [Citado el: 10 de enero de 2011.] http://www.softqanetwork.com/jmeter.

Prado, Elena Raja. 2007. *Casi todas las pruebas del software.* s.l. : Quality Assurance & Regulatory Affairs, NTE S.A, 2007.

Pressman, Roger S. 2005. *Ingenieria del Software Un enfoque práctico.* La Habana : Félix Varela, 2005. 6ta Edición.

Programming, Extreme. 2009. Extreme Programming. [En línea] 2009. http://www.extremeprogramming.org/.

Pruebas, Proceso de. 2011. Mejoramiento del Proceso de Pruebas. [En línea] 2011. [Citado el: 11 de enero de 2011.] http://chie.uniandes.edu.co/~gsd/index.php?option=com_content&task=view&id=129&Itemid=183.

Rueda, Ana López-Mancisidor. 2001. *¿Por qué invertir en la automatización de pruebas Software?* España : IBM Software Group., 2001.

RUP. 2006. Ayuda de Rational. [En línea] 2006.

Selenium, Corporation. 2010. *seleniumhq.* [En línea] 2010. [Citado el: 22 de enero de 2011.] http://seleniumhq.org/docs/03_selenium_ide.html#introduction.

Sommerville, Ian. 2005. *Ingeniería del Software,Séptima Edición.* Universidad de Alicante, Madrid : PEARSON EDUCACIÓN S.A ,Adison Wesley, 2005.

Toretti, Gustavo A. 2010. ibm. [En línea] 2010. [Citado el: 20 de enero de 2011.] http://www.ibm.com/developerworks/ssa/rational/library/10/automateintegrationtestswithrationalfunctionaltester/index.html.

Unix. 2010. nosolounix. [En línea] 2010. [Citado el: 12 de enero de 2011.] http://www.nosolounix.com/2010/02/herramientas-test-software.html.

Usaola, Dr. Macario Polo. 2006. *Mantenimiento Avanzado de Sistemas de Información.Pruebas de software.* Ciudad Real : Universidad de Castilla-La Mancha, 2006.

APÉNDICES

Glosario de Abreviaturas

ALBET: Empresa cubana que posee los derechos comerciales de todos los productos y servicios que desarrolla la UCI y mediante la alianza con otras prestigiosas entidades ofrece soluciones integrales en la esfera de las tecnologías de la información y las comunicaciones.

CALISOFT: Centro Nacional de Calidad de Software

CCV: Convenio Cuba-Venezuela.

DIE: Departamento de Inmigración y Extranjería

GB: Giga Byte.

HTML: HyperText Markup Language, Lenguaje de Marcas de Hipertexto. Es un lenguaje de programación que se utiliza para el desarrollo de páginas de Internet.

IEEE: Institute of Electrical and Electronics Engineers, Instituto de Ingenieros Eléctricos y Electrónicos. Asociación de profesionales norteamericanos que aporta criterios de estandarización de dispositivos eléctricos y electrónicos.

MINCEX: Ministerio de Comercio Exterior

NC: No Conformidad.

NP: No Procede.

RAM: Random Access Memory (Memoria de Acceso Aleatorio). Es la que se encarga de almacenar la información que el computador está utilizando mientras se encuentra encendido.

PC: Personal Computer, Computadora Personal.

PD: Pendiente.

XP: Extreme Programming, Programación Extrema.

Glosario de Términos

Artefacto: Pieza de información tangible que es creada, modificada y usada por los trabajadores al realizar las actividades; representa un área de responsabilidad y es candidata a ser tenida en cuenta para el control de la configuración. Un artefacto puede ser un modelo, un elemento de un modelo, o un documento.

Backup: Copia de seguridad. Se utilizan con el fin de poder recuperar la información y las aplicaciones en caso de una avería en el disco duro, un borrado accidental o un accidente imprevisto.

Bugs: Error o defecto en el software o hardware que hace que un programa funcione incorrectamente.

Caso de Prueba: Especificación de un caso para probar el sistema, incluyendo qué probar, con qué entradas y resultados y bajo qué condiciones.

Código abierto: Es el término con el que se conoce al software distribuido y desarrollado libremente.

Defecto: Anomalía del sistema, por ejemplo un síntoma de error en el software descubierto durante las pruebas, o un problema descubierto durante una revisión.

Ente: Entidad o institución.

Escenario: Una secuencia específica de acciones que ilustran un comportamiento.

Interfaz gráfica: Componente de una aplicación informática que el usuario visualiza y a través de la cual opera con ella. Está formada por ventanas, botones, menús e íconos, entre otros elementos.

Metodología: Es un conjunto de procedimientos, técnicas, instrumentos y documentos que ayudan a los analistas y programadores en sus esfuerzos para obtener un nuevo sistema informativo.

Multiplataforma: Término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas.

Plan de Pruebas: Plan que describe las estrategias, recursos y programación de las pruebas

Plugin: Es un módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

Proxy: Es una aplicación o un dispositivo hardware que hace de intermediario entre los usuarios, normalmente de una red local e Internet.

Requisito Funcional: Requisito que especifica una acción que debe ser capaz de realizar el sistema, especifica comportamiento de entrada/salida de un sistema.

Rol: Papel que ejerce un actor en una actividad o proyecto.

Scripts: Pequeños programas incrustados en las páginas que nos permiten definir interactividades de cualquier tipo.