Universidad de las Ciencias Informáticas

Facultad 15

Título: Perfeccionamiento de una estrategia basada en pruebas de Caja Blanca para los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II.

Autores:

Anel Alfonso Febles

Denia Francisca Alomá Benitez

Tutor: Ing. Yanet Edghill Martínez

Ciudad de la Habana, junio del 2010

¨Año 52 de la Revolución¨

Frase

La calidad nunca es un accidente; siempre es el resultado de un esfuerzo de la inteligencia.

John Ruskin.

Dedicatoria

A mis padres le dedico este trabajo porque nadie más que ellos se lo merecen, por apoyarme en las buenas y en las malas y confiar siempre en mí. A mi hermana por estar siempre ahí y cuidar de mí. A mi novio por esperarme y brindarme su amor incondicional.

Y a todas mis amistades que de una forma u otra contribuyeron a la realización de este trabajo.

Anel Alfonso Febles

A mi familia querida y amigos por ser lo más grande que tengo en la vida. Especialmente a mi bisabuela que es la niña de mis ojos y a mi bisabuelo que donde quiere que este, espero que se sienta orgulloso de mí, viendo que su crianza no fue en vano.

Denia F. Alomá Benitez

Agradecimientos

Le dedico este trabajo a toda mi familia por hacerme la persona que soy y porque un logro mío es un logro de ellos. A Luis A. Alegrant por apoyarme en todo lo posible desde que lo conocí.

A mi compañera de tesis Denia por desarrollar este trabajo a la par conmigo y a mi tutora Yanet por ayudarnos en cuanto a lo posible.

A todos mis compañeros de estudio que ayudaron a mi formación y al desarrollo de este trabajo.

A todas las personas que de una forma u otra aportaron al desarrollo de este trabajo.

Anel Alfonso Febles

A toda mi familia, por su preocupación, apoyo y cariño. A mi abuela, a mi papa, a mis tíos, a mis primos y a mis hermanas a los cuales quiero con todo mi corazón. Especialmente a mi bisabuela y a mi mama por quererme tanto y estar cuando las necesité, sin dejar de mencionar a mi bisabuelo que gracias a su presencia espiritual hoy soy lo que soy.

A mis mejores amigos de ahora y siempre, compañeros de batallas Ivelisse, Dollys, Damaidys, Ricardo, Jonnier, Pedro Carlos y el Indio que han estado en los buenos y malos momentos de mi vida, y por su apoyo a lo largo de estos cinco años. Gracias por su cariño.

A ti papito por darme tanto amor y estar siempre que te necesité.

A mi compañera de tesis Anel por esforzarse tanto.

A mi tutora Yanet por su ética y profesionalidad.

Al profesor Addel y al profesor Cadet por su apoyo y preocupación.

En general, a todos los que de una forma u otra han ayudado en la realización del trabajo.

Denia F. Alomá Benitez

Declaración de la Autoría

Declaro ser la autora del presente trabajo de diploma y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma son exclusivos.

Para que así conste firmamos la presente a los ____ días del mes _____ del año 20__.

_____ _____

Denia Francisca Alomá Benitez Anel Alfonso Febles

Autor Autor

_____

Yanet Edghill Martínez

Tutor

Resumen

El proceso de pruebas de software se hace con el objetivo de detectar la mayor cantidad de errores posibles para lograr satisfacer las necesidades del cliente, y entregarle el producto con la mayor calidad requerida. Este objetivo es alcanzado con el uso de metodologías, herramientas y métricas que contribuyen a la calidad del desarrollo de software. El presente trabajo de diploma se enmarca en el perfeccionamiento de una estrategia de pruebas de Caja Blanca realizada para el proyecto Registro y Notarías Fase I. Esta se aplicará a los sistemas Registros Principales y Notarías Públicas del proyecto Registro y Notarías Fase II apoyada en la metodología RUP y basada en el método de camino básico. En la misma se mantiene los principales procesos como es la planificación, el diseño, implementación y ejecución de los casos de prueba y evaluación de los resultados. A través del perfeccionamiento de la estrategia se alcanzó ajustar las actividades que hicieron necesarias para el desarrollo de la segunda fase del proyecto. Estuvieron enfocados los mismos principalmente a los roles que intervienen y las actividades a desarrollar para contribuir a la calidad del proceso de prueba.

La evaluación de los resultados obtenidos durante la aplicación de la estrategia, verificó el aporte realizado al aseguramiento de la calidad mitigando el mayor número de defectos en etapas iniciales del proceso de prueba.

Palabras claves

Proceso de pruebas de software, Estrategia de prueba, Caja Blanca.

Índice

Introducción

El avance de las tecnologías hoy en día está en constante evolución y desarrollo, y existen diversos sectores de la sociedad que interactúan de forma activa en dicho avance. La informatización es un proceso que ha tomado auge en los últimos tiempos debido al desarrollo de las ciencias de la computación y a las ventajas y comodidades que brinda el utilizar hoy todo un conjunto de herramientas producidas tras los resultados de estudios de esta ciencia.

El fruto de la informatización brinda una gran variedad de herramientas Software, que han hecho posible un trabajo más eficientemente ejecutado. Pero para garantizar un software fiable y con calidad se requiere de pruebas durante todo su ciclo de vida realizadas por el personal encargado. Las pruebas de software son un conjunto de actividades que se llevan a cabo metódicamente, pueden planificarse por adelantado y ejecutarse una vez construido el código con el objetivo de asegurar efectividad, adecuada funcionalidad y validez del mismo.

Para realizar pruebas de software se necesita una estrategia donde se utilicen métodos que analicen todos los posibles caminos y que asegure que el programa se encuentre libre de errores. En la actualidad contamos con herramientas para realizar pruebas al software de Caja Blanca como son: NUnit para .Net o el Rational Robot para .Net, Java y aplicaciones Web.

"*En un estudio realizado por la Asociación de Técnicos de Informática sobre la implementación y la eficacia de las pruebas de software en España se demostró que se considera 20 prácticas fundamentales para la realización de las pruebas de software para alcanzar los 5 niveles en función de la eficacia. De las 20 prácticas establecidas la mayoría solo aplican 8 y solo un 15,79% de las empresas alcanzan el 5 y más alto nivel de calidad. En cuanto al tipo de pruebas realizadas un 62% de las empresas optan por aquellas relacionadas con la satisfacción del cliente, que se reducen a la validación del cumplimiento de sus expectativas, y menos de un 10% son las que invierten en la gestión del proceso de pruebas en sí*."[*Tecnobloggers, 2009*]

Demostrando con la estadística anterior referenciada, que en el desarrollo de un software no se emplea un proceso de prueba profundo, pues conlleva tiempo y se tienen que hacer sistemáticamente, pero es importante aclarar que cuanto más se demore en encontrar un error mayor es el costo y el tiempo en solucionarlo. Es por ello la necesidad de una estrategia robusta la cual exige herramientas y conocimientos destinados a dicha tarea. También es de gran importancia que este proceso comience

desde el principio del desarrollo del software, desde la misma fase de inicio de la metodología RUP, muy conocida en el mundo del desarrollo de software.

El pueblo cubano en los últimos años se ha introducido en el mundo del desarrollo de software. Para esto cuenta con algunas empresas de software como la Empresa Nacional de Software, Desoft, con el fin de informatizar la sociedad cubana. Estas empresas siguen el ciclo de desarrollo de software y cuentan con una estructura de calidad, que bien planificada, detecta errores en tiempo para ser solucionados y no causen grandes pérdidas, tanto en tiempo como en recursos a la hora del despliegue del producto. Estas aplican tanto pruebas de Caja Blanca como de Caja Negra.

Un papel importante en la industria cubana de desarrollo de software es la Universidad de las Ciencias Informáticas (UCI). "*En el actual año la universidad tiene como objetivo alcanzar una certificación internacional del 2do nivel del modelo CMMI y la convertiría en la primera empresa cubana certificada con este modelo. La Dirección de Calidad de Software de la UCI es un centro de referencia de calidad reconocida y acreditada a nivel nacional*" [*Calisoft, 2009*], donde garantizan una producción de software con la calidad requerida. En este centro se están trazando estrategias de pruebas de Caja Blanca para verificar el código y a la vez optimizarlo, para esto se han realizado investigaciones sobre el tema pero no han tenido mucha aceptación debido a la falta de experiencia.

La universidad cuenta con disímiles proyectos productivos nacionales e internacionales que aportan considerables ingresos al país, organizados por facultades dentro de los cuales se encuentra el proyecto Registros y Notarías de la Facultad 15. Durante el desarrollo de la Fase I en el módulo Requisiciones del sistema Administración Financiera se realizó una estrategia de prueba que incluía pruebas de Caja Blanca, en dicha estrategia se obtuvieron resultados satisfactorios pero no estaba contemplada en la arquitectura del proyecto ni en el cronograma de trabajo. Esta estrategia no fue aplicada a los restantes módulos, por lo que se generó durante el proceso de pruebas de software un total de 165 no conformidades que afectaron al sistema Administración Financiera, de las cuales 149 se referían a la funcionalidad del software. Debido a que varios de estos errores se detectaron luego de la integración de todos los módulos que componían el sistema, le provocó al proyecto un retraso en las realizaciones de las tareas ingenieriles, como fueron las pruebas funcionales con los clientes, puesto que un error cuanto más tarde es detectado más perdida provoca. Ya en la arquitectura de los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II se realizan cambios en el entorno de desarrollo hacia el Visual Studio 2008 y queda integrada la sección de las pruebas en cada una de las

capas, por lo que surge la necesidad de perfeccionar la estrategia de prueba ya existente de pruebas de Caja Blanca que apoye la calidad del software, puesto que el equipo de calidad hace más énfasis en pruebas de Caja Negra dejándose de revisar el código y verificar cuán óptimo este puede ser.

Después de un análisis de la situación problemática planteada anteriormente se identifica el siguiente **problema de investigación**: El procedimiento que se utiliza actualmente en los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II para la detección de defecto no tiene en cuenta la revisión del código lo que afecta el proceso de prueba.

Donde se enmarca como **objeto de estudio:** Proceso de prueba de software.

El **Objetivo general** es: Perfeccionar una estrategia para la detección de defectos basada en las pruebas de Caja Blanca y contribuir a la mejora del proceso de prueba para los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II.

Siendo abarcado como el **campo de acción**: Proceso de pruebas de Caja Blanca para los sistemas Registros Principales y Notarías Públicas para el proyecto Registros y Notarías Fase II.

Hipótesis

Perfeccionando una estrategia basada en las pruebas de Caja Blanca se logrará mejorar el proceso de prueba para los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II.

Las **Tareas de la investigación** concebidas:

1. Estudio de los temas Gestión de Calidad de software, Proceso de Prueba de software, Estrategia de Pruebas de software.
2. Perfeccionamiento de estrategia de prueba de Caja Blanca para los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II.
3. Aplicación de la estrategia de pruebas de Caja Blanca en los sistemas de Registros Principales y Notarías Públicas para el proyecto Registros y Notarías Fase II.
4. Análisis de los resultados obtenidos mediante los resultados de la aplicación de las pruebas de Caja Negra y la aplicación de métrica.

Capítulo 1: Fundamentación Teórica.

1.1 Introducción.

Este capítulo se enfoca en el fundamento teórico del objeto de estudio; donde se abordan puntos elementales sobre el proceso de prueba. Se realiza un estudio detallado sobre las pruebas y de la estrategia ya existente, así como las características de las pruebas de Caja Blanca y las herramientas que se pueden utilizar para su automatización.

1.2 Calidad del Software.

En la actualidad uno de los problemas más frecuentes que se presentan durante el ciclo de desarrollo del software es el impacto de la calidad en la eficacia y eficiencia. Esto ha sido una gran preocupación a la que se ha dedicado mucho esfuerzo, e independientemente de esto los software casi nunca son perfectos. Muchos han investigado y analizado sobre el tema llegando a elaborar conceptos sobre la calidad de software.

"*La calidad de cualquier proceso del ciclo de vida del software influye en la calidad del producto software que, a su vez, contribuye a mejorar la calidad en el uso del producto*."[*Estándar ISO 12.207*]

"*Calidad del software: es el desarrollo de software basado en estándares con la funcionalidad y rendimiento total que satisfacen los requerimientos del cliente*."[Soto, 2010]

"*La calidad del software es el conjunto de cualidades que lo caracterizan y que determinan su utilidad y existencia. La calidad es sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad*. " [Oscar M., 1995]

El proceso de calidad de software es un conjunto de teorías, métodos y herramientas para perfeccionar a los procesos de desarrollo, artefactos, análisis, diseño, especificación de los requerimientos, arquitecturas, que son componentes que conforman la ingeniería de software. Este proceso no se enmarca dentro en un momento determinado sino que se realiza durante todo el ciclo de desarrollo y se puede definir parámetros, indicadores o criterios de medición para medir el grado de calidad de un producto. Estas mediciones pueden realizarse al producto final pero conllevaría un costo alto si se detectan errores

relacionados con las etapas anteriores por lo que se recomienda realizarse este proceso de forma continua durante todo el ciclo de vida.

Para llevar a cabo el proceso de calidad de software se necesitan metodologías o procedimientos estándares para el análisis, diseño, implementación y prueba del desarrollo del software en aras de alcanzar el mayor grado de calidad del producto final.

1.2.1 Gestión de la Calidad de Software.

La Gestión de la calidad es un conjunto de actividades que se realizan a nivel de empresa o dentro de la gestión de cada proyecto. Tiene como propósito entender las expectativas del cliente en términos de la calidad y llevar a cabo un plan para satisfacer dichas expectativas. Para una práctica factible de esta gestión se cuentan con 4 etapas.

La Planificación de la Calidad de Software es la encargada de administrar los objetivos a desarrollar y los recursos humanos con que cuentan la organización. Los aspectos que se tienen en cuenta en la planificación son modelos y estándares a utilizar, costo de la calidad de software, recursos humanos y materiales necesarios. En esta etapa se definen las particularidades más importantes del desarrollo del producto y se define el proceso de evaluación de la calidad.

El Control de la Calidad de Software son las técnicas y actividades que se realizan en función de los requisitos de la Calidad. Esta etapa tiene varios objetivos fundamentales en el proceso de la calidad los cuales son: mantener bajo control un proceso y eliminar las causas de los defectos en las diferentes fases del ciclo de vida, pero el aspecto más importante que tiene esta etapa son las Pruebas de Software.

El Aseguramiento de la Calidad de Software no es más que el conjunto de actividades planificadas y sistemáticas para asegurar la confianza de que el sistema responde a los requisitos de la calidad. Esta está presente en:

- ✓ Métodos y herramientas de análisis, diseño, programación y prueba.
- ✓ Inspecciones técnicas formales.
- ✓ Estrategias de pruebas.
- ✓ Control de la documentación del software y de los cambios realizados.

- ✓ Procedimientos para ajustarse a los estándares.
- ✓ Mecanismos de medidas o sea las llamadas métricas.
- ✓ Registros de auditorias y realización de informes.

La **Mejora de la Calidad del Software** contribuye mediante análisis de los datos, auditorías y mediciones a efectuar mejoras en la calidad del software. En esta etapa se tiene como objetivo mostrar la situación real para aportar confianza y descartar las áreas que puedan afectar a esta confianza. Otro objetivo es suministrar una evaluación objetiva de los productos y procesos para ratificar la conformidad con los estándares, guías, especificaciones y procedimientos.

1.3 Pruebas de Software.

La IEEE plantea que "la prueba es la actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan y almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente.*"[*IEEE, 1991*]*

Según RUP la prueba es una disciplina en el proceso de ingeniería de software cuyo objetivo es integrar y poner a prueba el sistema.

Este es un proceso destructivo que describe un caso de prueba, se considera exitoso si detecta defectos no descubiertos aún. La prueba demuestra hasta que punto el software tiene un buen funcionamiento pero no puede demostrar la ausencia de defectos en un software, por lo que la Calidad de un Software no depende solamente de un plan de pruebas ya que estas nos aseguran que el producto se encuentre libre de defectos pero si ayuda en gran medida a la Calidad de Software.

Por tanto, una estrategia de pruebas debe incluir pruebas de bajo nivel que se centran en que el código este correctamente implementado y pruebas de alto nivel que se dirigen un funcionamiento correcto del sistema. Este proceso es una actividad de evaluación del producto pero no se realiza al final sino que acompaña al desarrollo del ciclo de vida de un software y aporta información sobre la calidad del producto para luego tomar decisiones respecto al resultado. Es importante conocer que el costo asociado a las pruebas y corrección aumenta a medida que avanza el proceso de desarrollo del software.

1.3.1 Objetivos de Pruebas de Software.

Las pruebas de software son un proceso que ofrece métodos y técnicas para contribuir a la calidad del propio software, para lograr que este proceso sea satisfactorio se tienen que trazar objetivos fundamentales de la eficacia requerida. Estos objetivos son:

- ✓ Contribuir a la calidad del producto desarrollado.
- ✓ Descubrir errores de un programa no detectados hasta entonces.
- ✓ Manejar los resultados sistemáticamente, de forma que los defectos importantes puedan ser arreglados.

1.3.2 Niveles de Pruebas de software.

Las pruebas son aplicadas con diferentes objetivos y en diferentes escenarios. Por lo que se agrupan por niveles como:

- ✓ Pruebas de Unidad
- ✓ Pruebas de Integración
- ✓ Pruebas del Sistema
- ✓ Pruebas de Aceptación
- ✓ Pruebas Funcionales
- ✓ Prueba de Validación

"***Las pruebas de unidad*** *es el nombre que reciben los procedimientos de pruebas locales a un módulo del sistema. Por definición dichas pruebas cubren la funcionalidad propia del módulo tanto con una perspectiva de Caja Blanca como de Caja Negra; pero prestando poca o ninguna atención a la integración con otros módulos. "[*Garcerant, 2008*]* Combinando ambas perspectivas se obtiene una mayor fiabilidad del producto final.

Las pruebas de **Caja Negra** se le aplican a la interfaz del software pero es independiente de la estructura del código. Estas pruebas permiten detectar:

- ✓ Funcionamiento incorrecto o incompleto.

- ✓ Errores en la interfaz.
- ✓ Errores en el acceso de estructura de datos externa.
- ✓ Problemas de rendimientos.
- ✓ Errores de inicialización y terminación.

Estas pruebas tratan de demostrar que:

- ✓ Las funciones del software son operativas.
- ✓ Las entradas se aceptan de forma adecuada.
- ✓ Se produce una salida correcta.
- ✓ La integridad de la información externa se mantiene.

"*Las pruebas de **Caja Blanca** examinan la estructura interna del programa y pretenden garantizar que:*

- ✓ *Se ejerciten todos los caminos independientes de cada módulo.*
- ✓ *Se ejerciten todas las decisiones lógicas.*
- ✓ *Se ejecuten todos los bucles.*
- ✓ *Se ejecuten las estructuras de datos internas. "[* E-Software, 2010*]*

Las Pruebas de Integración es para asegurar que los componentes en el modelo de implementación funcionen correctamente una vez integrados para ejecutar un caso de uso.

Las Pruebas del Sistema verifican que se alcance la funcionalidad y el rendimiento del sistema total. Estas tienen como objetivo ejercitar profundamente el sistema con diferentes tipos de prueba como:

- ✓ Prueba de validación: proporciona una seguridad final de que el software satisface todos los requerimientos funcionales. Durante la validación se usan exclusivamente técnicas de prueba de Caja Negra.
- ✓ Prueba de recuperación: fuerza un fallo del software y verifica que la recuperación se lleva a cabo apropiadamente.
- ✓ Prueba de seguridad: verificar los mecanismos de protección.

- ✓ Prueba de resistencia: enfrenta a los programas a situaciones anormales.
- ✓ Prueba de rendimiento: prueba el rendimiento del software en tiempo de ejecución.
- ✓ Prueba de instalación: se centra en asegurar que el sistema desarrollado se puede instalar en diferentes configuraciones de hardware y software, y bajo condiciones excepcionales, por ejemplo con espacio de disco insuficiente.

Las Pruebas Funcionales: como su propio nombre lo indican, prueban una funcionalidad completa, donde pueden estar implicadas una o varias clases y hasta la propia interfaz de usuario.

Las Prueba de Validación: se prueba el software totalmente ensamblado como un todo para comprobar si cumple los requisitos funcionales y de rendimiento, facilidad de mantenimiento y recuperación de errores.

Las Pruebas de Aceptación son las pruebas finales que se hace antes del despliegue. Su objetivo es verificar que el software esté listo y que puede ser usado por los usuarios finales para cumplir con las funciones y tareas para las cuales fue construido.

1.3.3 Estrategia de Pruebas de Software.

*"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: la planificación, el diseño de casos de prueba, la ejecución y los resultados, tomando en consideración cuánto esfuerzo y recursos se van a requerir, con el fin de obtener como resultado una correcta construcción del software."[*Pressman, 2002]

La estrategia de pruebas se hace con el objetivo de que el producto de software que se encuentre en desarrollo, reúna con todos los requisitos planteados por el cliente mediante la lógica del negocio. La estrategia de prueba es la línea base para las personas que están capacitadas para realizarles pruebas a un proyecto de desarrollo de software. La misma debe ser definida claramente ya que comprende las ideas más representativas del proceso de pruebas que se llevará a cabo.

Si se tiene en cuenta una correcta planificación para la estrategia de prueba, la misma puede proporcionar menor tiempo de ejecución y disminuir los costos que se generan. Esto proceso se hace con el mayor grado de responsabilidad, para obtener un desarrollo de prueba con calidad.

Básicamente la estrategia de prueba busca dejar claros los siguientes aspectos que corresponden al proceso de pruebas de un desarrollo de software.

- ✓ Metodología de pruebas que se va a usar: esta es una colección de métodos de pruebas para verificar y validar el software en su desarrollo. O sea, su finalidad fundamental es permitir establecer los pasos para ejecutar pruebas a cualquier sistema antes de se ejecute frente al usuario final.
- ✓ Cómo serán evaluados los riesgos del software: la evaluación de riesgos es utilizada para identificar, medir y priorizar los mismos, con el fin de que el mayor esfuerzo sea realizado para identificar los riesgos de mayor relevancia.
- ✓ Qué técnicas específicas de pruebas que serán usadas para probar el software: estas técnicas ayudarán a definir un conjunto de casos de prueba aplicando un cierto criterio. Estas técnicas pueden ser de Caja Blanca donde sus criterios son basados en el contenido de los módulos, y las de Caja Negra con sus criterios basados en las interfaces.
- ✓ Qué herramientas y recursos serán necesarios: para realizar esta estrategia son utilizadas herramientas con el fin de realizarles pruebas al software, por lo que se requiere de un estudio para seleccionar la herramienta idónea. Para probar el sistema en desarrollo es muy importante tener en cuenta los recursos necesarios que contribuyan al avance de la estrategia de pruebas de software.
- ✓ En cuanto al equipo de pruebas se debe tener en cuenta las siguientes preguntas:
    - ¿Cuántas personas serán necesarias para realizar el proceso de pruebas?
    - ¿Qué habilidades deben tener?
    - ¿Serán necesarias capacitaciones para dichas personas?

1.3.4 Análisis de la estrategia del proyecto Registros y Notarías Fase I

En el proyecto Registros y Notarías Fase I se desarrolló una estrategia para las prueba de Caja Blanca, dicha estrategia solo se aplicó al módulo de Requisiciones del sistema Administración Financiera pero de forma aislada, siendo algo de lo que se encargaron algunos miembros del equipo de calidad y estas no estaban contempladas dentro de la arquitectura del proyecto.

En esta estrategia teniendo en cuenta los procesos de prueba se definieron roles y responsabilidades, así como, artefactos y herramientas de automatización, necesarios en el desarrollo la misma.
La estrategia anterior estaba estructurada de la siguiente forma:

Procesos de prueba

De los procesos asociados a las pruebas de software se propone la realización de los procesos planificación, diseño, implementación, ejecución y evaluación de las pruebas. Estos consisten básicamente en una serie de pasos para la realización de pruebas al software y tienen como objetivo demostrar que el software cumpla con los requisitos y tareas para el cual fue diseñado y que los artefactos generados se encuentran libres de errores.

Planificación de las pruebas: en este subprocesos se planifica todo lo referente al proceso de pruebas, como definir una estrategia de prueba con el fin de mejorar dicho proceso. Establecer un plan de prueba y su alcance donde se especifiquen recursos humanos, herramientas a utilizar, ámbito del plan de prueba, roles a participar conjunto con sus responsabilidades, precisar si se necesita capacitación para el uso de la herramienta y en caso de serlo planificarla, junto con otros procesos que sean necesarios o sean propios del desarrollo del software en específico.

Diseño de las pruebas: este subproceso donde se diseñan casos de pruebas con el objetivo de lograr encontrar la mayor cantidad de defectos posibles en procesos posteriores, el diseño se basa en la descripción de casos de usos y en la tabla de datos para obtener un mayor entendimiento sobre las funcionalidades del código.

Implementación de las pruebas: este subproceso se basa en el uso de la herramienta de automatización donde se implementan las pruebas anteriormente diseñadas.

Ejecución de las pruebas: luego de la implementación de las pruebas se procede a su ejecución con el objetivo de comparar los resultados obtenidos con los resultados esperados.

Evaluación de las pruebas: una vez ejecutadas las pruebas se procede a su evaluación mediante métrica y se elabora el documento de sumario de evaluación de estas.

Roles

- ✓ ***Administrador de prueba****: Es el máximo responsable del éxito de las pruebas, pues es el encargado de verificar y asegurar que se realice una correcta planificación y administración de los recursos. Comprueba el progreso y efectividad de las pruebas, evalúa los resultados de cada iteración, y debe dar solución a los problemas que impiden el buen desarrollo del proceso de prueba.*
- ✓ ***Analista de Pruebas****: Es responsable de identificar y definir las pruebas requeridas, supervisando el progreso de las pruebas, así como el resultado por cada iteración. Tiene como responsabilidad la obtención de los datos de pruebas necesarios para obtener todas las posibles respuestas del sistema.*
- ✓ ***Diseñador de Pruebas****: Es responsable de identificar las técnicas apropiadas, herramientas y pautas para llevar a cabo las pruebas requeridas, y define los recursos necesarios.*
- ✓ ***Probador****: Este rol es responsable de las principales actividades y el mayor esfuerzo de las pruebas. Es quien ejecuta las pruebas y obtiene directamente los resultados. Se encarga de hacer un registro donde se plasman todos los defectos encontrados como resultados de la ejecución de las pruebas.*
- ✓ ***Responsable de Calidad Interna:*** *Es el encargado de planificar y organizar las revisiones, orientando los artefactos a revisar así como su entrega, también planifica los cursos de capacitación para el equipo y la aceptación del plan de pruebas. Se encarga de conformar el informe único de No Conformidades con todos los defectos encontrados y se lo hace llegar a los responsables de darle solución a los problemas. Este también debe realizar un resumen de los resultados a través de métricas archivándose los resultados para su posterior uso, finalizando así las revisiones.*
- ✓ ***Desarrollador del software****: Participa junto al Analista de Pruebas en el diseño de los casos de prueba para las pruebas de unidad y de integración. También participa en la implementación y ejecución de las pruebas empleando la herramienta NUnit junto al probador y es el encargado de realizar la depuración.¨[González, Durán, 2008]*

Artefactos

Los artefactos generados en la estrategia realizada en la Fase I del proyecto fueron los siguientes:

Plan de prueba: Es una visión general del proceso de prueba y es un artefacto como resultado del proceso de planificación de pruebas. Este tiene en cuenta el alcance, requerimientos a probar, definición

de la estrategia, herramienta que se utilizan para la realización del proceso de prueba, los recursos tanto humanos como del sistema, el cronograma de actividades y entregables.

Estrategia de prueba: El artefacto estrategia de prueba describe los objetivos generales de las pruebas. Incluye las fases de prueba que se deben seguir y los tipos de pruebas que se deben realizar. En esta se definen propósito, alcance, entregables y técnicas de prueba.

Configuración del entorno de prueba: En este artefacto se detalla la distribución de hardware y software para llevar a cabo las pruebas, periféricos de entrada y salida y sus drivers correspondientes, así como herramientas que faciliten la prueba. El objetivo de este es separar el ambiente de pruebas del ambiente de desarrollo y especificar las herramientas necesarias.

Datos de pruebas: Para el diseño y ejecución de los casos de pruebas se necesita datos que apoyen a la ejecución de los mismos, permitiendo que el sistema se ejecute por todas sus variantes. Los datos de prueba no son más que datos reales referentes al negocio correspondiente, los cuales se utilizan como entrada, salida o precondiciones. Estos se escogen atendiendo a las especificaciones del negocio.

Casos de pruebas: Los casos de prueba constituyen la especificación formal donde quedan registrados los datos de entrada de la prueba, las condiciones para su ejecución, así como los resultados previstos. Un caso de prueba incluye la verificación del resultado de la interacción entre los actores y el sistema, las precondiciones y poscondiciones especificadas por el caso de uso, así como la secuencia de acciones para la ejecución de las pruebas.

Procedimiento de prueba: El artefacto procedimiento de prueba constituye el conjunto de instrucciones detalladas para la disposición y ejecución paso a paso de uno o más casos de prueba. Este puede constituir una instrucción sobre cómo ha de realizarse un caso de prueba manualmente o puede ser una especificación de cómo interactuar con una herramienta de automatización de pruebas. En el caso de las pruebas de Caja Blanca, cada caso de prueba tendría asociado un procedimiento de prueba con los pasos a seguir para utilizar la herramienta NUnit, herramienta que se propuso para su automatización.

Sumario de evaluación de las pruebas: En este artefacto se expone un resumen de los resultados de las pruebas aplicadas a los componentes operacionales del sistema, lo cual permite revisar y evaluar la calidad del producto que se está desarrollando. Este artefacto puede contener observaciones y recomendaciones para futuros ciclos de pruebas, a lo que se suman los siguientes aspectos:

- Breve descripción del contenido del Sumario de Evaluación.
- Resumen de los resultados de las pruebas.
- Análisis de la cobertura de las pruebas.

Herramienta de automatización

Para la automatización de las pruebas en el proyecto Registros y Notarías Fase I, en la estrategia de prueba de Caja Blanca se empleó la herramienta NUnit que se ajusta a la plataforma .Net para llevar a cabo dicho proceso.

1.3.5 Métodos de pruebas de Caja Blanca.

"*Los métodos de prueba del software tienen el objetivo de diseñar pruebas que descubran diferentes tipos de errores con menor tiempo y esfuerzo*."[Díaz, 2010]

Las pruebas de Caja Blanca es un método utilizado para el diseño de casos de pruebas que se basan en un análisis detallado de la estructura del código, se comprueban los caminos lógicos, bucles y condiciones examinando así el estado del programa para informar de la situación real de la calidad del software.

En este tipo de pruebas se utilizan diferentes métodos con el objetivo de analizar el código por diferentes criterios como son:

- ✓ Pruebas de camino básico.
- ✓ Pruebas de condición.
- ✓ Pruebas de bucles.

El método de **camino básico** consiste en obtener una medida de la complejidad de un código y utilizar esta medida de guía para la definición de los caminos básicos y diseños de casos de pruebas que garanticen que se ejecuten al menos una vez cada uno.

En este método se realiza un grafo que muestra los caminos de un determinado programa, donde utiliza una notación específica cómo se muestra a continuación:

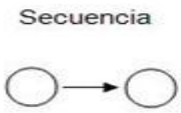


Figura 1

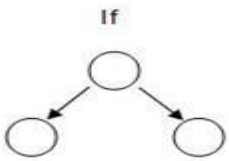


Figura 2

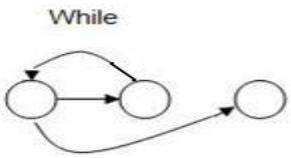


Figura 3

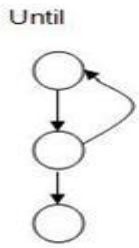


Figura 4

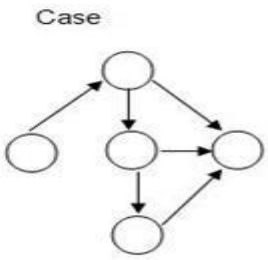


Figura 5

Simbología:

- ✓ Cada nodo representa una sentencia de código.
- ✓ Un nodo puede responder a una secuencia de pasos o a una decisión (If).
- ✓ Las flechas o aristas representar el flujo de control.

En este tipo de grafo se puede calcular la complejidad ciclomática con el objetivo de conocer la medida de la complejidad lógica de un programa. Existen 3 formas de calcular la complejidad:

- ✓ La complejidad ciclomática coincide con el número de regiones del grafo de flujo.
- ✓ La complejidad ciclomática es igual al número de aristas(A) – el número de nodos(N) + 2.
- ✓ La complejidad ciclomática es igual a los nodos de predicados + 1.

A partir de la complejidad obtenemos el número de caminos independientes que nos ayuda a saber la cantidad de pruebas que debemos diseñar para el programa.

Los pasos para diseñar las pruebas de camino básico son:

- ✓ Obtener el grafo de flujo, a partir del código del módulo.
- ✓ Obtener la complejidad ciclomática del grafo de flujo.
- ✓ Definir el conjunto básico de caminos independientes.
- ✓ Determinar los casos de prueba que permitan la ejecución de cada uno de los caminos anteriores.
- ✓ Ejecutar cada caso de prueba y comprobar que los resultados son los esperados.

El método de pruebas de **bucles** se centra en la validez de las construcciones del bucle.

El bucle en programación, es una sentencia que se realiza repetidas veces a un trozo aislado de código, hasta que la condición asignada a dicho bucle deje de cumplirse.

Se pueden distinguir 4 tipos de bucles diferentes:

- ✓ Bucles simples.
- ✓ Bucles concatenados.
- ✓ Bucles anidados.
- ✓ Bucles no estructurados.

Los bucles simples se representan como se muestra en la siguiente figura:

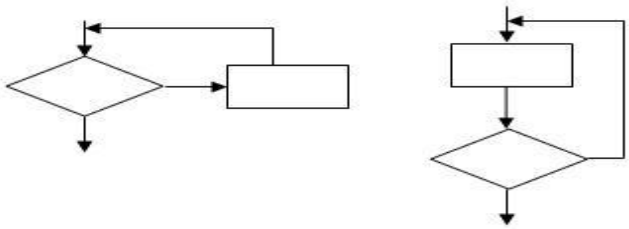


Figura 6 Método de bucle simple.

A los bucles simples de n iteraciones se le pueden aplicar las siguientes pruebas:

- ✓ Saltar el bucle.
- ✓ Pasar sólo una vez por el bucle.
- ✓ Pasar dos veces por el bucle.
- ✓ Hacer *m* pasos del bucle con *m* < *n.*
- ✓ Hacer *n-1, n* y *n+1* pasos por el bucle.

Los bucles anidados se representan como se muestra en la siguiente figura:

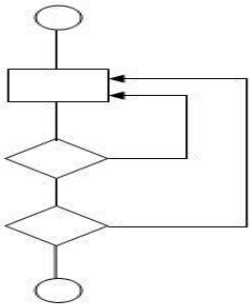


Figura 7 Método de bucle anidado.

Si aplicáramos el criterio de bucles simples a los bucles anidados el número de pruebas aumentarían en gran medida por lo que se recomienda el siguiente conjunto de pruebas para los bucles anidados:

- ✓ Comenzar con el bucle más interno, estableciendo los demás bucles a los valores mínimos.
- ✓ Llevar a cabo las pruebas de bucles simples para el bucle más interno, conservando los valores de iteración de los bucles más externos a los valores mínimos.
- ✓ Progresar hacia fuera en el siguiente bucle más externo manteniendo los bucles más externos a sus valores mínimos y continuar hasta que se hayan probado todos los bucles.

Los bucles concatenados se representan de la siguiente forma:

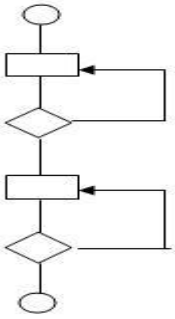


Figura 8 Método Bucles concatenados.

Este tipo de bucles se prueba aplicando el criterio de bucles simples mientras estos sean independientes, de lo contrario se utiliza el criterio de bucles anidados.

Por último, se muestra los bucles no estructurados que siempre que sea posible se deben rediseñar para que se ajusten a las construcciones de la programación estructurada y se representan como en la siguiente figura:

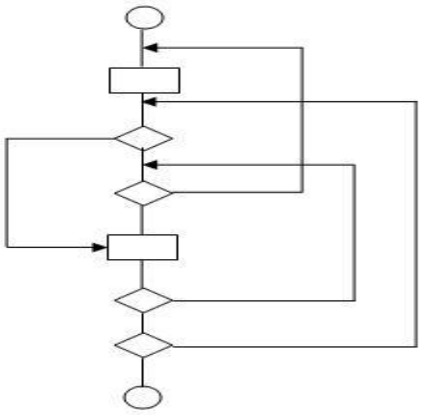


Figura 9 Método Bucles no estructurado.

El método de pruebas de **bucles de condición** diseña casos de pruebas que ejercitan las condiciones lógicas de la estructura de un código.

En este tipo de pruebas se manejan dos conceptos sobre las condiciones:

- ✓ Condición simple: es una variable lógica o una expresión relacional (E1 < operador - relacional > E2).
- ✓ Condición compuesta: está formada por dos o más condiciones simples, operadores lógicos y paréntesis.

Los tipos de errores más frecuentes en este tipo de prueba son:

- ✓ Error en operador lógico (existencia de operadores lógicos incorrectos, desaparecidos, sobrantes).
- ✓ Error en variable lógica.
- ✓ Error en paréntesis lógico.
- ✓ Error en operador relacional.
- ✓ Error en expresión aritmética.

Una vez conocidos los métodos de pruebas de Caja Blanca se procede a determinar el método idóneo para la estrategia. Teniendo en cuenta las características que dichos métodos analizan se escogen el método de camino básico para la realización de esta estrategia por ser el más completo, ya que dentro de este se analizan las condiciones y los bucles.

Los pasos a seguir para diseñar los casos de pruebas de Caja Blanca son:

- ✓ Obtener el grafo de flujo, a partir del diseño o del código del módulo.
- ✓ Obtener la complejidad ciclomática del grafo de flujo.
- ✓ Definir el conjunto básico de caminos independientes.
- ✓ Determinar los casos de prueba que permitan la ejecución de cada uno de los caminos anteriores.
- ✓ Ejecutar cada caso de prueba y comprobar que los resultados son los esperados.

1.4 Herramientas para las Pruebas de Caja Blanca.

El software de automatización es utilizado para controlar la ejecución de pruebas, el mismo es parte del ciclo de calidad, en la que realiza comparación de resultados, preparación de precondiciones y realización de informes. Estas pruebas son efectivas en entornos donde los cambios son frecuentes debido a la calidad con que se realicen.

Existe una gran variedad de herramientas que ayudan en gran medida a su realización, ya que en la actualidad los sistemas que se necesitan son cada vez más grandes y complejos, y con estas herramientas se puede reforzar el desarrollo de todo el proceso de pruebas, disminuyendo tiempo, esfuerzo y costo del proceso de prueba.

Entre las herramientas creadas para automatizar las pruebas, dentro de las cuales soportan la plataforma el Visual Studio se encuentran NUnit, Rational Robot, Parasoft .TEST entre otros. Estas herramientas tienen diferentes características que la distinguen y que a la vez son aplicadas para diferentes lenguajes. Por estas razones se hace una investigación de las herramientas con el fin de conocer a fondo sus características con el objetivo de escoger la idónea para la estrategia de prueba en cuestión.

Una de las características que deben cumplir la herramienta seleccionada es soportar al lenguaje C#.

1.4.1 ParaSoft.Test.

La ParaSoft .TEST trabaja con lenguajes de programación que utilizan el IDE de desarrollo Microsoft .NET y soporta el lenguaje C#. Los miembros del equipo de calidad lo pueden emplear para identificar problemas críticos antes de un inminente despliegue. Monitoriza la calidad del proyecto y el progreso a través de las metas de calidad. Para su funcionamiento requiere de:

- ✓ Windows XP o Windows 2003 Server.
- ✓ 512MB mínimo, 1GB recomendado.
- ✓ Procesador de 1GHz o más rápido.
- ✓ Visual Studio .NET 2003.

Parasoft.TEST ofrece servicios como:

- ✓ Optimización de la infraestructura.
- ✓ Aseguramiento de la Calidad de Software.
- ✓ Visibilidad y Control de Procesos.
- ✓ Monitoreo de la infraestructura.

1.4.2 Rational Robot

El Rational Robot es una de las herramientas para realizar pruebas sobre aplicaciones Java y Web, al igual que las aplicaciones basadas en GUI (Interfaz Gráfica de Usuario). Esta herramienta soporta múltiples tecnologías de interfaz de usuario desde Java y la web hasta todos los controles de VS.NET, incluidos VB.NET, J#, C# y C++.

Esta herramienta facilita al equipo de pruebas:

- ✓ Facilita la transición de los equipos de pruebas manuales a pruebas automatizadas.
- ✓ Identifica más defectos al ampliar sus scripts de pruebas con lógica condicional, para abarcar una mayor parte de la aplicación.
- ✓ Proporciona casos de prueba para objetos comunes, como menús, listas y mapas de bits y casos de prueba especializados para los objetos específicos del entorno de desarrollo.

Una de las desventajas que tiene es que necesita de una licencia para el uso del producto pero sin embargo incluye componentes de gestión de pruebas, seguimiento de defectos, gestión de cambios y rastreo de requisitos.

Sistemas Operativos y Plataformas de Hardware apropiadas:

- ✓ Windows 2000
- ✓ Windows 95/98
- ✓ Windows NT
- ✓ Windows XP

1.4.3 Framework NUnit

La NUnit es un framework el cual se integra con el Visual Studio con el objetivo de desarrollar pruebas de unidad en un proyecto. Las pruebas desarrolladas con esta herramienta son exitosas si los valores esperados y generados cumplen con las condiciones una vez comparados.

Dentro de sus características principales se encuentran:

- ✓ Es un framework opensource (marco de trabajo de código abierto) para pruebas unitarias de sistemas realizados con la plataforma Microsoft .NET.
- ✓ Sirve al mismo propósito que realiza JUnit en el mundo Java, y es uno de muchos en la familia XUnit.
- ✓ Consiste en un conjunto de meta atributos y manifestaciones que permiten probar los métodos de una clase especificada.
- ✓ Se puede ejecutar desde la consola o a través de una interfaz gráfica.
- ✓ Se puede integrar con el Visual Studio en cualquiera de sus versiones.
- ✓ NUnit soporta los lenguajes base de .NET como C#, J#, VB y C++.

Esta herramienta proporciona algunas ventajas en cuanto a pruebas:

- ✓ Automatiza las pruebas, por lo cual se hacen repetibles.

- ✓ Fomenta el cambio: ya que permiten probar cambios en el código y asegurar que en éstos no se hayan introducido errores funcionales; habilitan la refactorización del código.
- ✓ Simplifica la integración: permiten llegar a la fase de integración con un alto grado de seguridad sobre el código.
- ✓ Documenta el código.
- ✓ Separa la interfaz y la implementación.
- ✓ Los defectos están acotados y fáciles de localizar.
- ✓ Permiten al desarrollador pensar como el consumidor del código y no como el productor.

1.5 Conclusiones.

A partir del estudio de los principales conceptos del proceso de prueba y de la estrategia antes desarrollada se estableció la base de las actividades del perfeccionamiento de la estrategia de pruebas de Caja Blanca. De igual manera se utilizará el método de camino básico para la realización de las pruebas por ser flexible y abarcar todas las condiciones posibles de las funcionalidades desarrolladas. El perfeccionamiento de la estrategia será guiado por los mismos procesos de pruebas ya definidos teniendo en cuenta los roles, artefactos, actividades y herramienta de automatización ajustados al desarrollo del proyecto Registros y Notarías Fase II.

Capítulo 2: Perfeccionamiento de la estrategia de pruebas de Caja Blanca.

2.1 Introducción.

En este capítulo se perfeccionan las actividades de la estrategia de prueba de Caja Blanca utilizada en el proyecto Registros y Notarías Fase I. Se realiza un análisis de las actividades, roles y artefactos generados ajustándolos a las condiciones actuales del proyecto detectando mejoras que favorezcan el proceso de prueba de software.

Este capítulo se centra en el perfeccionamiento y descripción de los procesos fundamentales que conformarán la estrategia de prueba de Caja Blanca. Se efectúa un análisis de los diferentes roles ya definidos y responsabilidades que participan en la fase de prueba. Se establece la relación entre los roles que intervienen en el proceso de prueba, las actividades a desarrollar, los artefactos resultantes y el conjunto de métricas a aplicar. De igual forma se muestran algunos de las principales características del funcionamiento de la herramienta NUnit para la automatización de las pruebas.

2.2 Redefinición de la estrategia de prueba de Caja Blanca.

A partir de la estrategia existente de prueba para el proyecto Registros y Notarías Fase I y las características del desarrollo de software actual del proyecto se realizó un estudio con el objetivo de reorientar las actividades ya definidas hacia los objetivos trazados del proyecto y de la calidad.

Para el perfeccionamiento de esta estrategia se realizó un estudio sobre la situación actual del proyecto Registros y Notarías Fase II y de la estrategia ya existente para Registros y Notarías Fase I con el objetivo de poder luego redefinir dicha estrategia.

En una encuesta realizada al administrador de la calidad del proyecto Registros y Notarías Fase II se evidenció que cuentan con un equipo de 8 personas para las actividades de calidad interna. Estos son los que diseñan y aplican las pruebas obteniendo como resultado el artefacto caso de prueba, donde se diseñan y ejecutan las pruebas realizadas al software. Luego el administrador del equipo de calidad redacta el documento único de No Conformidades que es entregado a los desarrolladores para la corrección de errores.

Existe poca experiencia en la utilización de la herramienta para la automatización de las pruebas de Caja Blanca por la poca importancia que se le brinda a estas.

Otro cambio sustancial en el proyecto por lo que es necesario redefinir la estrategia es el cambio en la arquitectura del proyecto, ya que con el cambio del IDE de desarrollo hacia el Visual Estudio 2008 permite la integración de la NUnit sin necesidad de abandonar el mismo, por lo que se integra los componentes de las pruebas en las capas de presentación, negocio y acceso a datos.

Por tanto, para lograr un buen desarrollo de la aplicación de las pruebas de Caja Blanca en el proyecto Registros y Notarías Fase II se redefinen una serie de aspectos de la estrategia ya existente que garantice este proceso de forma organizada y eficiente.

2.2.1 Propósito de la estrategia de pruebas de Caja Blanca.

La estrategia de prueba de software garantiza que se realice las actividades de pruebas de Caja Blanca durante el desarrollo del proyecto con el objetivo de contribuir a la calidad de los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II.

2.2.2 Alcance.

La siguiente estrategia de prueba se fundamenta en la ejecución de pruebas de Caja Blanca a los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II utilizando el método camino básico. Para el desarrollo de la misma se analizan y se redefinen los procesos de pruebas como planificación, diseño, implementación, ejecución y evaluación de las pruebas de la estrategia ya existente, así como los artefactos que se deben generar y los roles que deben intervenir en cada uno de los procesos.

En la estrategia no se define la cantidad de iteraciones a realizar en el proceso de prueba, pero se le propone al administrador de la calidad realizar tantas iteraciones como sea necesario para mejorar el proceso de prueba.

2.2.3 Objetivos.

El objetivo de las pruebas de Caja Blanca es garantizar que una unidad por separado funcione correctamente antes de integrarla al sistema, por lo que se requiere planificar y ejecutar actividades que

contribuyan a la estabilidad incremental del software, siendo las pruebas de Caja Blanca un filtro antes de las pruebas de Caja Negra y de la integración de los casos de usos.

2.3 Metodología de Prueba de Software.

El proyecto Registros y Notarías Fase I fue guiado por la metodología RUP en su proceso de desarrollo al igual que Registros y Notarías Fase II.

¨El Proceso Unificado de Rational o RUP (Rational Unified Process), es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP es en realidad un refinamiento realizado por Rational Software del más genérico Proceso Unificado¨. [Fernandez, 2004]

RUP se caracteriza por ser iterativo e incremental, guiado por casos de usos y centrado en la arquitectura. Este divide el proceso de desarrollo en ciclos y cada ciclo se divide en fases obteniendo artefactos (productos tangibles del proceso como por ejemplo el modelo de casos de usos, código fuente, etc.) y asigna roles (papel que desempeña una persona en determinado momento) para una mayor organización y asignación de tareas.

RUP propone para cada fase que las pruebas se comporten de la siguiente manera:

- ✓ **Inicio**: tiene como objetivo de determinar la visión del proyecto. En esta fase no se requiere de elaboración de pruebas.
- ✓ **Elaboración**: en esta fase se prueban los componentes ejecutables que se han implementado y que deben corresponderse con la arquitectura básica de la aplicación.
- ✓ **Construcción**: en esta fase se desarrollan los casos de prueba y procedimientos de prueba a realizar.
- ✓ **Transición**: en esta última fase se obtiene el producto final que es probado por usuarios reales.

En este caso la estrategia se centra en la fase de construcción ya que aquí es donde se obtiene el código al cual se le aplicaran las pruebas de Caja Blanca. De igual manera como en la anterior fase la estrategia estará basada en las principales características de la metodología RUP como iterativa e incremental y guiada por casos de uso, lo que posibilita los diseños de casos de pruebas y su automatización.

2.3.1 Roles y responsabilidades.

En la estrategia anterior basada en la metodología RUP se utilizan los siguientes roles durante el proceso de prueba:

- ✓ **Administrador de prueba.**
- ✓ **Analista de prueba.**
- ✓ **Diseñador de prueba.**
- ✓ **Probador.**
- ✓ **Responsable de Calidad Interna.**
- ✓ **Desarrollador del software**.

En el caso de la estrategia de prueba de Caja Blanca para el proyecto Registros y Notarías Fase II los roles quedaron establecidos de la siguiente manera, tomando en cuenta las características propias y el funcionamiento del equipo de calidad del proyecto Registro y Notarías Fase II manteniéndose:

- ✓ **Probador**.
- ✓ **Diseñador de prueba**

El analista de prueba no se contempla dentro de los roles que participan en esta estrategia, ya que no está definido dentro de los roles y responsabilidades en el programa de mejora para alcanzar el segundo nivel de CMMI definido en el documento 0516_Roles y Responsabilidades. [Calisoft, 2010].

Y se le agregan o modifican los siguientes roles:

- ✓ **Jefe de sistema**: Es el encargado de integrar las funcionalidades implementada por los programadores, y una vez validada la estabilidad del código hace entrega del mismo al administrador de la calidad.
- ✓ **Programador**: Es el nuevo nombre que toma el desarrollador de software dentro de los roles definidos para alcanzar el segundo nivel de CMMI [Calisoft, 2010]. Este el responsable de desarrollar las funcionalidades de los casos de usos, brinda apoyo en la implementación de las pruebas y corrige los defectos.

- ✓ **Administrador de la calidad:** es la unión de los roles administrador de pruebas y responsable de la calidad interna. Es el máximo responsable del éxito de las pruebas, pues es el encargado de verificar y asegurar que se realice una correcta planificación y administración de los recursos. Comprueba el progreso y efectividad de las pruebas, evalúa los resultados de cada iteración, y debe dar solución a los problemas que impiden el buen desarrollo del proceso de prueba.

2.3.2 Artefactos que se obtienen en la estrategia de prueba.

Durante el desarrollo de la estrategia se generan diferentes artefactos con el objetivo de documentar cada proceso y plasmar los resultados obtenidos para un posterior análisis. Los artefactos generados en la estrategia anterior son los siguientes:

- ✓ Plan de prueba.
- ✓ Estrategia de prueba.
- ✓ Configuración del entorno de prueba.
- ✓ Datos de pruebas.
- ✓ Casos de pruebas.
- ✓ Procedimiento de prueba.
- ✓ Sumario de evaluación de las pruebas.

Como parte del perfeccionamiento de esta estrategia se decide mantener los artefactos para Registros y Notarías Fase II:

- ✓ Casos de pruebas
- ✓ Datos de pruebas
- ✓ Sumario de evaluación de las pruebas

Debido a las actualizaciones del expediente de proyecto establecido por la dirección de calidad se modifica el siguiente artefacto:

- ✓ Plan de pruebas: en dicho artefacto se contemplan además de los ya definidos como epígrafes los artefactos establecidos en la estrategia anterior como son: estrategia de prueba, entorno de configuración de prueba y procedimiento de prueba.

También se agregan artefactos con el objetivo de mejorar el proceso de prueba como:

- ✓ Lista de Chequeo: Es una técnica para evaluar aspectos estándares en un determinado ámbito. La lista de chequeo consiste en un formulario de preguntas a un determinado objeto sobre el cual se aplicarán. Para dar respuesta a cada pregunta se considera una relación de signo de – a +, donde (-) siempre es la respuesta menos significativa y (++) la más significativa (Ver Anexo 1).
- ✓ Documento de no conformidades: Es el documento donde se registran todas las no conformidades o defectos detectados durante todo el proceso de prueba. En dicho documento se especifica cual fue el error detectado, localización del mismo y su estado (Pendiente o Resuelto).
- ✓ Lista de riesgos: Es el documento donde quedan registrados los riesgos identificados que puedan afectar al proceso de prueba, para luego incluirlos dentro del Plan de gestión de riesgos del proyecto Registros y Notarías Fase II, donde se analizan los mismos para darle posteriormente seguimiento y control de riesgos, en este se definen el plan de mitigación para poder evitarlos y el plan de contingencia en caso de que ocurra para recuperarse lo antes posibles.

2.3.3 Procesos de la estrategia de prueba.

El desarrollo de la estrategia anterior estaba dirigía por procesos de prueba los cuales se muestran en la siguiente figura:

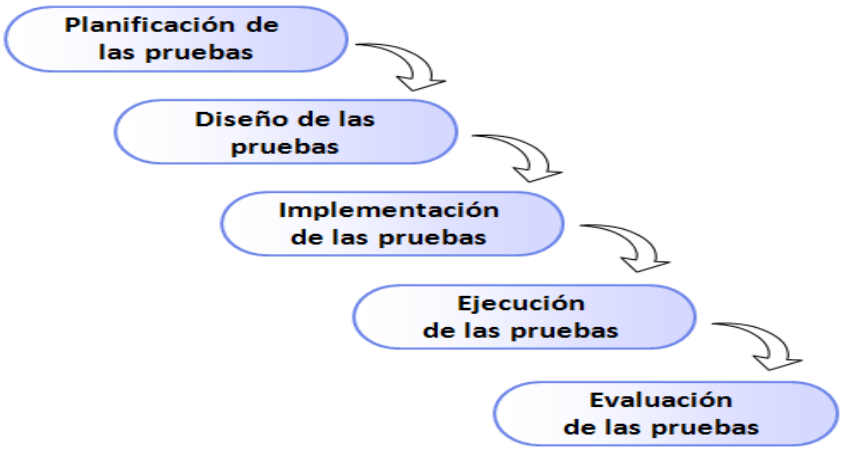


Figura 10 Proceso de prueba

En los procesos definidos se agrupan todas las actividades que deben desarrollarse durante la estrategia por lo que se propone mantenerlas y de ser necesario ajustarlas, modificando igualmente los roles que interviene, las entradas y las salidas.

2.3.3.1 Planificación de las pruebas.

Las pruebas de software deben ser planificadas con un tiempo anticipado antes de que comience su ejecución. En este proceso se precisan las pruebas que se les realizarán al software, se identifican sus roles y máximo responsable, así como la obtención del esfuerzo y tiempo necesario para las pruebas, artefactos que se generan, recursos requeridos, repositorios y ambientes de prueba.

Todas las planificaciones de las pruebas deben quedar reflejadas en el plan de prueba ya que es donde se describe y se deja claro como el equipo de calidad debe realizar las pruebas en un tiempo determinado según el cronograma de actividades. Así como la definición de los elementos de: escenario de pruebas, recursos del sistema, herramientas que serán utilizadas, estrategia de pruebas y evaluación de las mismas.

Objetivo

La planificación de las pruebas tiene como objetivo principal la definición de una estrategia apropiada, la planificación del esfuerzo y la definición de roles y recursos del sistema. Al igual que identificar y prevenir los diferentes riesgos que atenten contra el proceso de prueba.

Roles que participan

Administrador de la calidad

Diseñador de prueba

Entradas

Especificación de requisitos de software.

Modelo de caso de uso del sistema.

Cronograma de desarrollo del proyecto.

En este caso los artefactos de modelo de diseño y modelo de implementación no consistirán en entradas al proceso de planificación por no realizarse los mismos en el proyecto, En sustitución se realizan encuentro entre los diseñadores y programadores en aras de cubrir cualquier duda que exista al respecto.

Actividades

Las actividades establecidas en la estrategia anterior fueron:

- Identificar objetivos de las pruebas.
- Evaluar recursos necesarios.
- Definir los tipos de prueba a realizar.
- Definir las pautas de las pruebas.
- Definir la configuración del entorno de prueba.

Debido a los cambios establecidos en cuanto a la estrategia, las actividades del proceso de planificación se centran en la elaboración del artefacto plan de prueba por ser el artefacto principal del proceso de planificación. De igual manera se identifican los posibles riesgos para prever todas las posibles causas que puedan atentar contra la mejora del proceso, incorporando esta tarea como parte del proceso de planificación que no estaba inicialmente contemplada. Las actividades definidas en la planificación de pruebas son:

- Realizar el modelo de despliegue del sistema.
- Determinar los recursos del sistema, como son servidores y PC clientes.
- Definir los requerimientos a probar.
- Describir los criterios de evaluación para las pruebas en este caso el criterio de evaluación son las pruebas de Caja Blanca.
- Controlar roles y responsabilidades que deben participar en la realización de las pruebas.
- Identificar los posibles riesgos que puedan afectar al proceso de prueba.
- Definir la configuración del entorno de prueba.

Salidas

Artefacto Plan de Prueba.

Artefacto Lista de riesgos

Recursos

Para la realización del Plan de prueba se utiliza la planilla propuesta por la Dirección de Calidad de la Universidad.

Representación gráfica

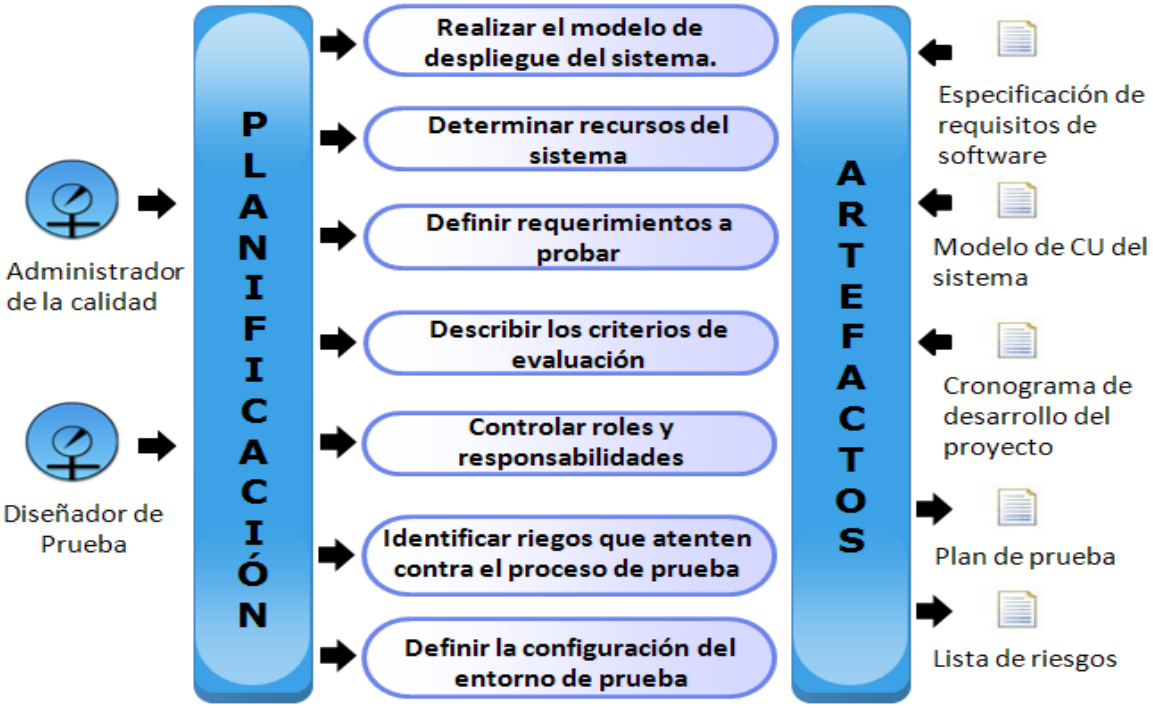


Figura 11 Proceso de planificación de las pruebas.

2.3.3.2 Diseño de las pruebas.

Una vez que se ha realizado la planificación de las pruebas se pasaría a realizar el diseño de las mismas, ya que el plan de pruebas es una de las entradas más importantes para este proceso. Para el diseño de las pruebas se identifican y describen los casos de pruebas, y los posibles datos de prueba que serán utilizados posteriormente en la ejecución.

Objetivos

La etapa de diseño de pruebas tiene como objetivos principales identificar los casos de prueba y organizar los procedimientos de pruebas de manera estructurada.

Roles que participan

Los roles que intervenían en este proceso (analista de prueba y desarrollador de software) fueron sustituidos por las razones expuestas en epígrafes anteriores, quedando de la siguiente forma:

Diseñador de pruebas

Jefe de sistema.

Administrador de la calidad.

Entradas

Especificación de requisitos de software

Modelo de casos de uso del sistema.

Plan de Pruebas.

Lista de chequeo

Elementos de prueba (Release)

Actividades

Las actividades establecidas en la estrategia anterior son:

- Identificar los datos de prueba.
- Identificar y describir los casos de prueba.
- Identificar y estructurar los procedimientos de prueba.

Las primeras dos actividades se mantienen por la importancia que presentan para este proceso ya que consisten en las acciones fundamentales a desarrollar. En el caso de identificar los procedimientos de prueba, estos se registran en el caso de prueba estableciendo los pasos a seguir para lograr una ejecución satisfactoria, este ajuste trae consigo que este procedimiento de prueba este contemplado en el producto entregable Caso de prueba

Y se agregan las siguientes actividades:

- Entrega de elementos de pruebas estables: entrega de los componentes de prueba persistentes por parte del jefe de sistema para que las pruebas una vez diseñas no sufran cambios y así no realizar esfuerzos innecesarios.
- Verificar el cumplimiento de las actividades referentes al diseño: verificar el cumplimiento y calidad de las actividades en el momento preciso para así evitar un retraso del proceso de prueba.
- Verificar cumplimiento de los estándares de codificación: aplicación de la lista de chequeo para verificar el cumplimiento del código generado con los estándares de codificación definidos

Salidas

Artefacto Casos de prueba.

Artefacto Datos de prueba.

Recursos

Actualmente en la Universidad se utilizan las plantillas definidas en el Expediente de proyecto 2.0 que estandariza la documentación en todos los proyectos. El empleo de la planilla Caso de prueba constituye una ventaja pues está mejor estructurada, ya que se especifica las condiciones para la ejecución de las pruebas, se describe el flujo de pasos o procedimiento a seguir para probar cada uno de los escenarios de las pruebas, contiene cada una de las variables que serán probadas durante todo el proceso. Se toman en cuenta además las no conformidades detectadas, y esto le sirve al programador para corregir todos los defectos encontrados, facilitándole su localización. Por estos beneficios se considera que para el diseño y aplicación de los casos de prueba en el proyecto se debe emplear esta planilla.

Representación gráfica

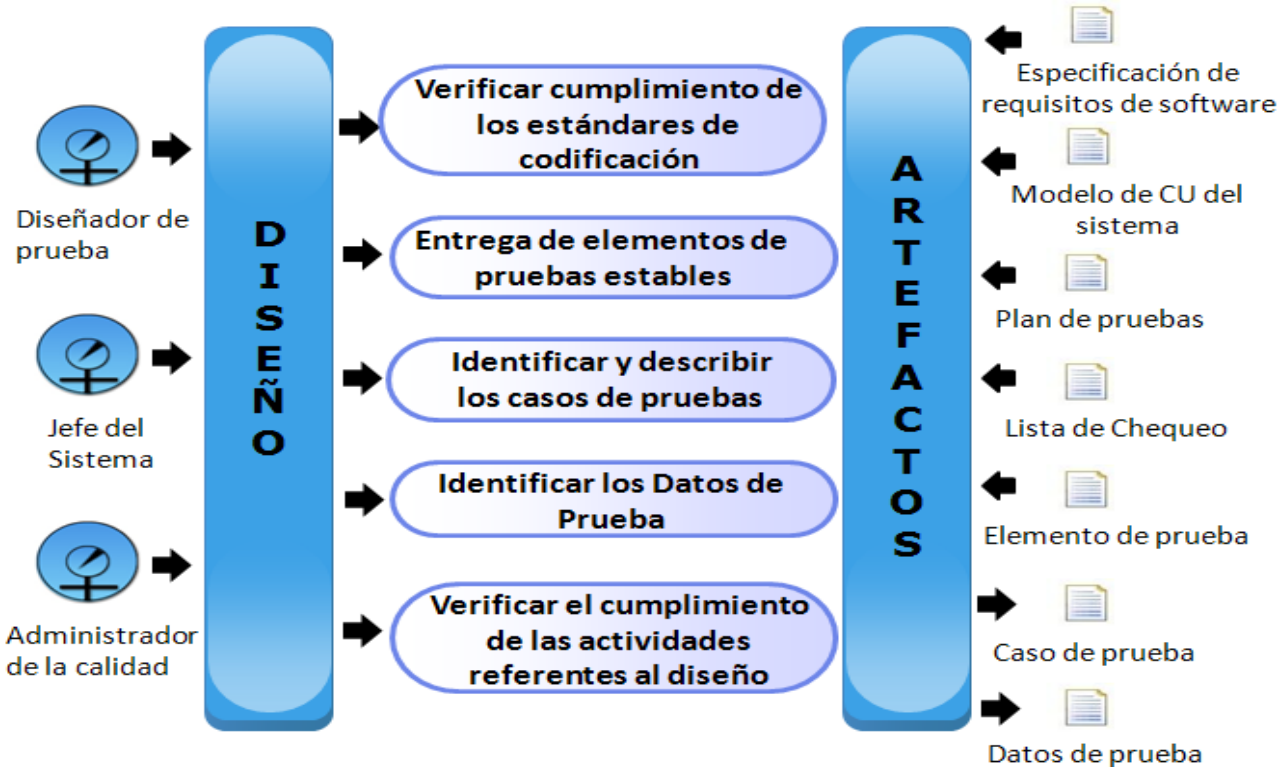


Figura 12 Proceso de diseño de las pruebas

2.3.3.3 Implementación de las pruebas.

Para la implementación de las pruebas se utilizará la herramienta Reshaper 4.5 la cual integra el Framework NUnit con el Visual Studio 2008, IDE utilizado para el desarrollo del proyecto. En la estrategia utilizada en la Fase I del proyecto las pruebas se implementaban de manera independiente al IDE de desarrollo limitación que se eliminó con la utilización de la herramienta Reshaper 4.5. El empleo de esta herramienta permite garantizar que las pruebas se desarrollen de forma rápida y eficiente además que puedan ser repetibles brindando al proceso de prueba disminución de tiempo y esfuerzo.

Objetivo

La etapa de implementación tiene como objetivo automatizar los casos de prueba.

Roles

Inicialmente estaba propuesto el rol de diseñador de pruebas como otro de los roles que debían intervenir en el presente proceso , tomando en cuenta la estructura actual del equipo de calidad interna, el probador

se encargará de realizar la implementación de los casos de pruebas que permite a través de la herramienta establecida generar los componentes de pruebas necesarios. Quedando establecidos los siguientes roles:

Probador

Programador

Entradas

Caso de prueba

Elemento de prueba (Release)

Datos de Prueba

Actividades

En la estrategia en cuestión se mantienen las siguientes actividades:

- Definir los datos de prueba.
- Definir los componentes de prueba.

No se realiza la actividad de identificar los mecanismos de prueba pues la estrategia está basada en el mecanismo de pruebas de Caja Blanca.

Y se agrega la actividad de:

Brindar apoyo a la realización de los componentes de pruebas la cual la realiza el programador.

Salidas

Artefacto Caso de prueba (Actualizado).

Artefacto Componente de prueba.

Recursos

Para la aplicación de pruebas de Caja Blanca se utiliza la herramienta de completamiento de código ReSharper 4.5 que permite integrar el framework NUnit con la plataforma Visual Studio 2008.

Representación Gráfica

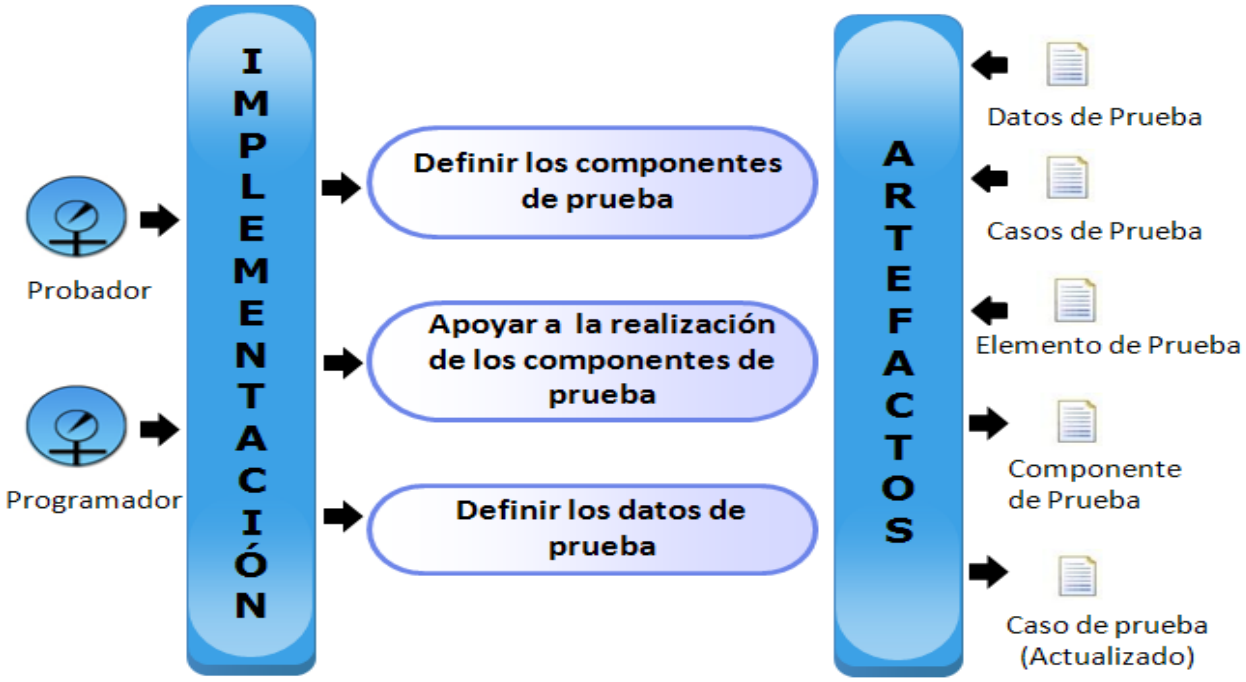


Figura 14 Proceso de Implementación de las pruebas.

2.3.3.4 Proceso de ejecución de las pruebas.

Una vez definidos los casos de pruebas y los datos de pruebas el probador puede proceder a la ejecución de las mismas con el objetivo de descubrir defectos o no conformidades.

Aclarar que las pruebas se realizan en diferentes iteraciones en correspondencia con los defectos corregidos, verificando que las no conformidades detectadas sean resueltas con el fin de lograr una mayor estabilidad del software.

Objetivos

Ejecutar las pruebas ya implementadas (Componentes de pruebas) mediante la tabla de datos que brinda combinaciones de datos tanto válidos como inválidos para detectar los defectos y en correspondencia con el resultado de las pruebas se registran los resultados satisfactorios y no satisfactorios en el documento Caso de prueba.

Roles

Según los roles definidos para este proceso en la anterior estrategia de prueba de Caja Blanca los roles involucrados eran el desarrollador de software y analista de pruebas. En epígrafes anteriores se establecieron los roles que intervendrían durante el proceso de prueba y se asume que el probador será el encargado de ejecutar los componentes de prueba que fueron generados en el anterior proceso

Probador

Entrada

Casos de pruebas (Actualizado)

Datos de pruebas

Componentes de pruebas.

Actividades

Las actividades propuestas por la estrategia anterior se mantienen ya que se ajustan a las necesidades actuales del proyecto y están en correspondencia de los objetivos trazados.

- Ejecutar los casos de pruebas.
- Identificar los defectos o no conformidades.
- Determinar los resultados de las pruebas.
- Registrar las no conformidades.

Salidas

Componentes de pruebas (Actualizados) los definidos test scripts en la estrategia anterior.

Casos de pruebas (Actualizados)

Documento de No conformidades

Representación Gráfica

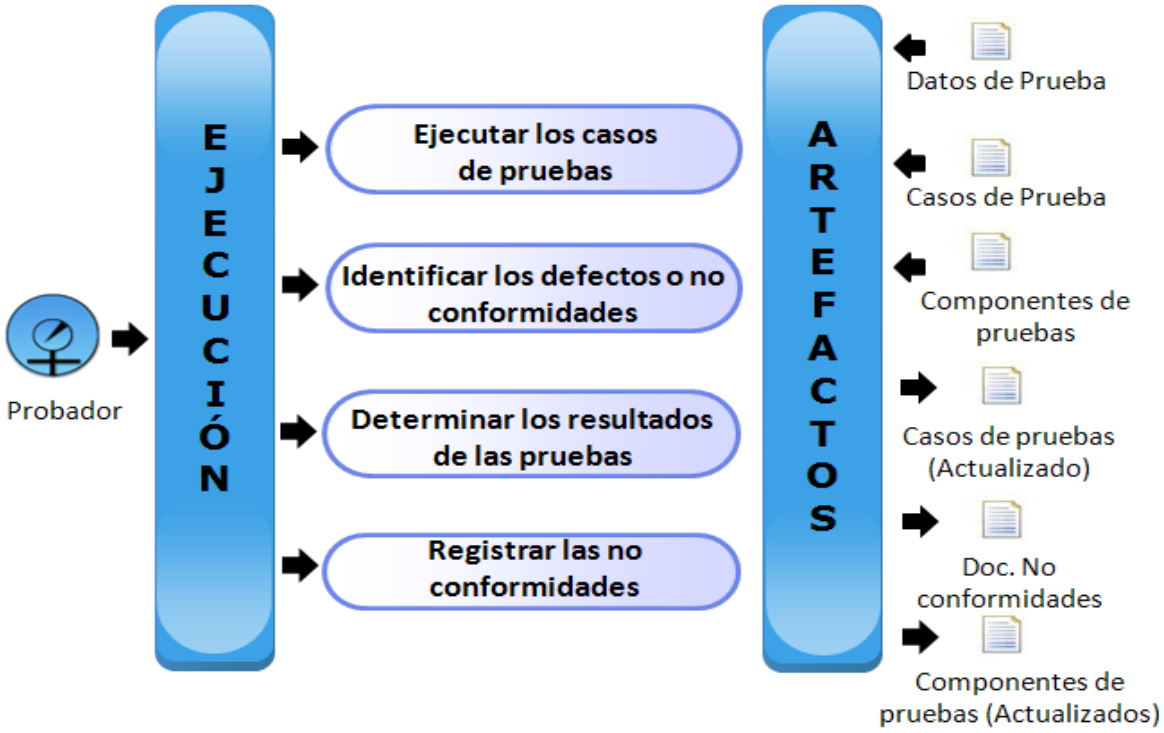


Figura 14 Proceso de ejecución de las pruebas.

2.3.3.5 Proceso de evaluación.

En el proceso de evaluación se comparan los resultados esperados con los resultados obtenidos a través de métricas, con el propósito de comprobar si las pruebas son factibles o no y además evaluar el desempeño de los probadores.

Objetivo

Evaluar los resultados obtenidos, registrar los defectos encontrados y en caso necesario recomendaciones y observaciones para pruebas posteriores.

Rol

Administrador de la calidad

Entradas

Documento de No Conformidades

Actividades

En este proceso también se mantienen las actividades definidas en la estrategia anterior.

- Evaluar la cobertura de las pruebas
- Analizar los defectos a través de métricas
- Elaborar un informe de evaluación

Salidas

Artefacto Sumario de evaluación de pruebas

Representación gráfica

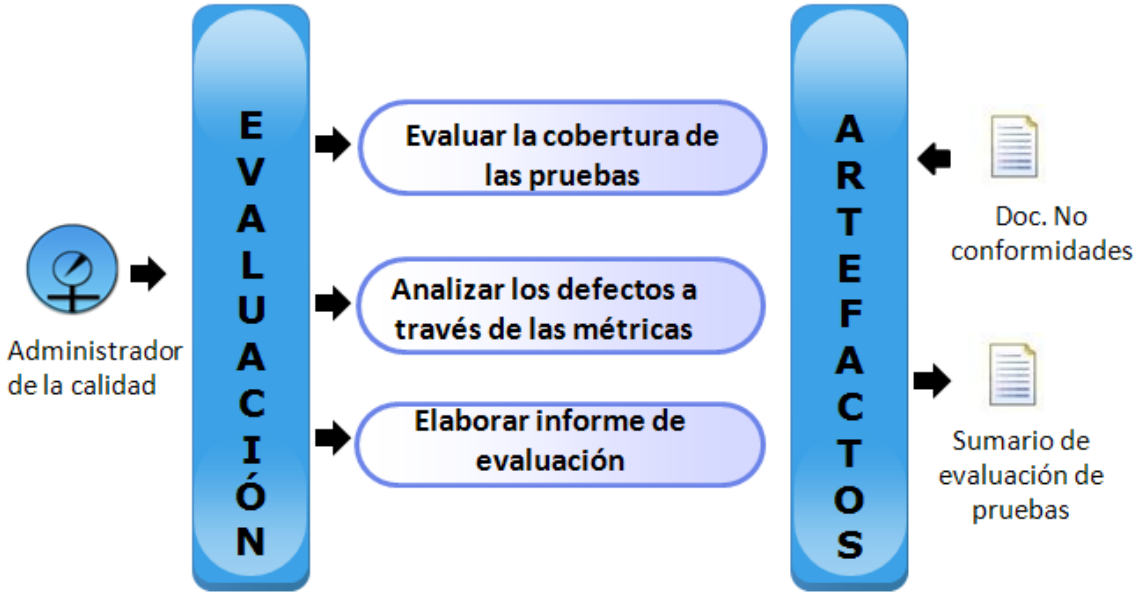


Figura 15 Proceso de evaluación de las pruebas.

2.4 Métricas a utilizar en la estrategia.

Las métricas de software proveen la información necesaria para la toma de decisiones técnicas. Estas son un medio útil para monitorizar, controlar y probar el desarrollo de un software.

Durante el desarrollo de la estrategia ya existente se estudiaron las siguientes métricas:

- Complejidad ciclomática.
- Cobertura de las pruebas
- Madurez de las pruebas
- Porciento del tiempo total dedicado a las pruebas
- Métricas EED (Eficacia de la Eliminación de Defectos)

De las cuales se aplicó solamente la métrica de Cobertura y ya que en la actualidad se exige un alto grado de calidad se requiere de varias medidas para demostrar el grado de aceptación con que cuenta el producto a través de varios criterios. Se mantienen las mismas métricas a utilizar definidas en la estrategia anterior [González, Durán, 2008] con el objetivo de validar los resultados y se agrega la siguiente métrica:

Métrica 6: Densidad de Defectos.

La densidad de defectos es una medida sobre la proporción de defectos con respecto a la cantidad de elementos de especificación. Permite realizar análisis estadísticos al finalizar las pruebas para valorar la integridad y madurez del software analizado.

Esta se calcula

$$DD = {}^{TD}/_{CER}$$

Dónde:

DD: densidad de defectos, TD: número total de defectos encontrados durante las pruebas y CER: número de elementos de especificación revisados.

Es recomendable para una alta calidad del software que la densidad de defectos tenga un valor mínimo.

2.5 Gestión de riesgo en el proceso de pruebas.

Tomando en cuenta los riesgos a los cuales puede estar expuesto el proceso de prueba se hace necesario identificar algunos ya que podrían afectar el éxito del proceso y deben ser tomados en cuenta para mitigarlos. En la estrategia anterior no se incluyó la gestión de riesgo en el proceso de prueba, lo que

se puede tomar como una deficiencia ya que el objetivo de esta es mejorar el proceso de prueba cosa que puede ser atentado si ocurre algún riesgo como los que se exponen a continuación:

- ✓ Poca disponibilidad de puestos de trabajo.
- ✓ Falla eléctrica.
- ✓ Actividades extra-productivas que impiden el continuo desarrollo del proceso de pruebas.
- ✓ Capacitación al equipo de pruebas de las herramientas a utilizar.
- ✓ Interrupción del desarrollo del proyecto.

Para una gestión de riesgo detallada donde se incluyen dichos riesgos ver el documento de la gestión de riesgo del proyecto Registros y Notarías Fase II (RN II Plan de Gestión de Riesgos)

2.6 Conclusiones.

En este capítulo se rediseñaron los procesos que guían la estrategia redefiniendo los roles que intervienen en las diferentes actividades y dando como resultado un conjunto de artefactos (Caso de Pruebas, datos de prueba, plan de prueba) del proceso de pruebas de software, y así optimizar el proceso de revisiones, debido a la necesidad de realizar diferentes pruebas al software para garantizarle una mayor calidad. Esta estrategia al igual que la anterior se apoya en la realización de prueba de software, utilizando la metodología RUP, ya que es la utilizada tanto en el proyecto Registros y Notarías Fase I como en Fase II al cual va dirigida.

Se especificaron los roles que se considera que deben intervenir en cada uno de los procesos de prueba de Caja Blanca, así como los artefactos necesarios para que estos se encuentren debidamente documentados, dando paso a cada una de las actividades a desarrollar.
Se procedió a la selección de la herramienta a emplear para la automatización de las pruebas de Caja Blanca, así como la selección de las métricas a utilizar en la estrategia, que se emplean para evaluar que el proceso está lo más completo posible. También se tuvo en cuenta los riesgos que puedan atentar al proceso de prueba incluyéndolos al Plan de gestión de riesgos del proyecto Registros y Notarías Fase II.

Capítulo 3: Aplicación de la estrategia de pruebas de Caja Blanca.

3.1 Introducción.

Este capítulo se centra en la aplicación de la estrategia perfeccionada de pruebas de Caja Blanca a los sistemas Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II. Como guía se utilizan los procesos establecidos, que enmarcan las actividades de diseño de casos de pruebas de unidad mediante el método de camino básico, aspecto fundamental para la ejecución de los casos de pruebas y se utiliza el framework NUnit para la automatización de las mismas. Por último, se realiza la evaluación de los resultados obtenidos con el objetivo de verificar la calidad del producto.

3.2 Plan de Pruebas.

El plan de prueba es el artefacto principal del proceso de planificación y tiene como objetivo principal definir el alcance, enfoque, recursos requeridos, cronograma de actividades, responsables e identificación de riesgos en el proceso de pruebas. El propósito del mismo es establecer la línea base por la cual el equipo de calidad se regirá para realizar pruebas en el tiempo determinado.

3.2.1 Alcance.

Se indican artefactos, recursos necesarios tantos humanos como de sistemas, así como los elementos del software a ser probado con la ejecución de pruebas de unidad, en los sistemas de Registros Principales y Notarías Públicas del proyecto Registros y Notarías Fase II.

3.2.2 Referencia

RN II Especificación de Requisitos (Registros Principales).

RN II Especificación de Requisitos (Notarías Públicas).

Modelo de casos de usos sistema de Registros Principales.

Modelo de casos de usos sistema de Notarías Públicas.

Cronograma de actividades de Registros Notarías Fase II.

3.2.3 Requerimientos a probar.

Los requerimientos a probar son las dos primeras fases del proceso de Inscripción, el cual incluye a los subprocesos Revisión y Cálculo. De los casos de usos incluidos en este proceso es de especial atención Gestionar Trámite de Inscripción que pertenece al subproceso de revisión y caso de uso Calcular Trámite perteneciente al subproceso de cálculo, los cuales se encuentran dentro de los casos de usos arquitectónicamente significativos de los sistemas Registros Principales y Notarías Públicas.

Ver Modelo de casos de usos sistema de Registros Principales.

Ver Modelo de casos de usos sistema de Notarías Públicas.

3.2.4 Estrategia de prueba.

La estrategia se fundamenta en la ejecución de pruebas de unidad específicamente en las pruebas de Caja Blanca a los sistemas Registros Principales y Notarías Públicas utilizando el método camino básico para diseñar casos de pruebas que garanticen que cada camino se ejecuta al menos una vez.

En la siguiente figura 16 se muestra el flujo de actividades donde se especifican los roles que participan en la estrategia de prueba y las tareas que realizan. En dicho flujo de actividades se muestran los pasos a seguir en la realización de las acciones que guían la estrategia, estas acciones fueron descritas detalladamente en los procesos de prueba.

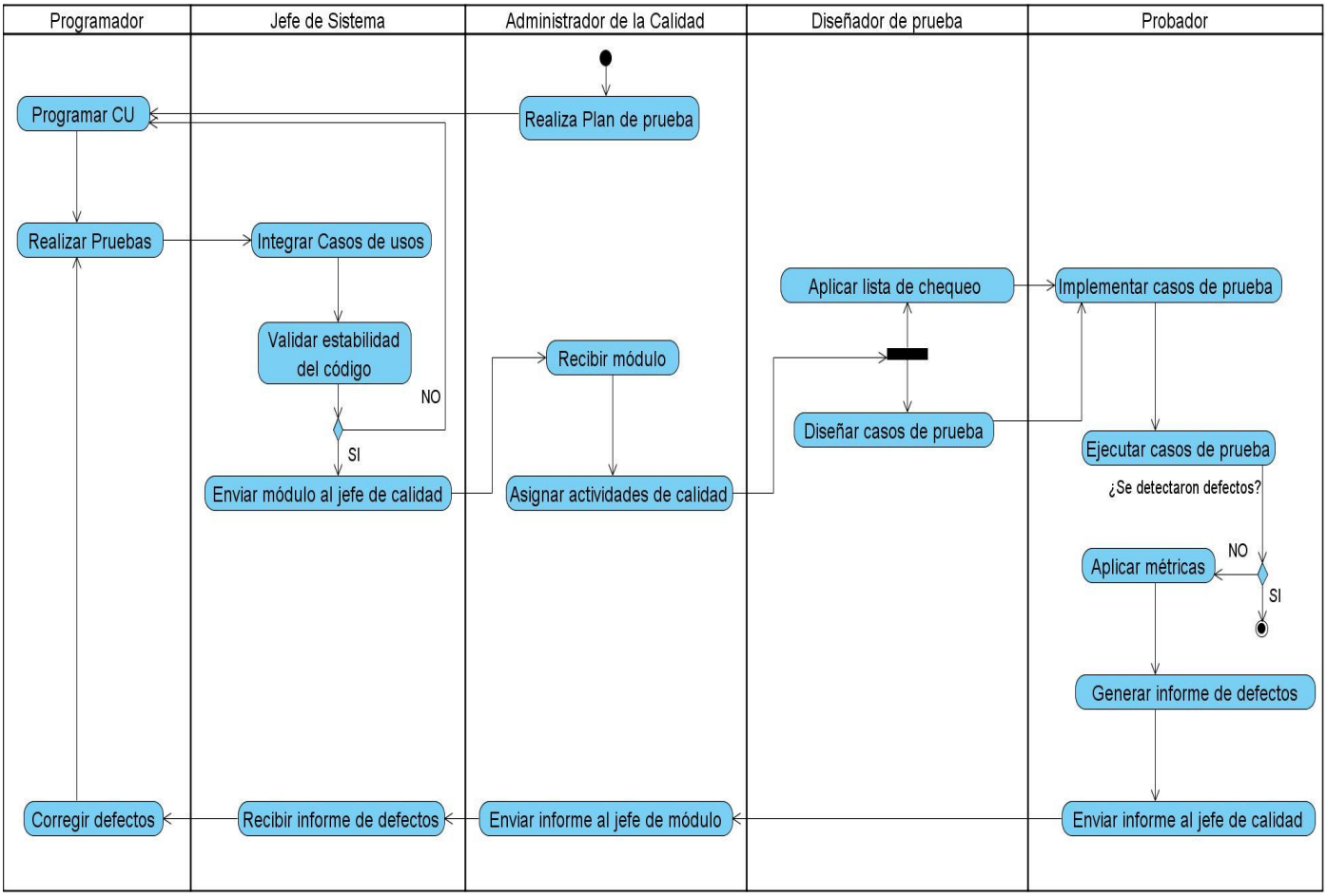


Figura 16 Flujo de actividades de la estrategia de prueba.

3.2.5 Herramientas utilizadas.

Para la realización de las pruebas se requiere de un conjunto de herramientas para lograr una mayor rapidez y eficacia durante este proceso como son:

	Herramientas
Gestión de proyecto.	Microsoft Project 2007 Microsoft Word 2007
Entrono de Configuración de prueba.	Microsoft Visio 2007

Pruebas de Caja Blanca.	ReSharper 4.5.1274.1 Framework NUnit

3.2.6 Recursos del sistema

Servidores de Base de datos:

- ✓ Servidor de base datos del sistema Registros Principales, con 2 procesadores Intel Xeon, 1GB RAM.
- ✓ Servidor de base datos del sistema Notarías Pública, con 2 procesadores Intel Xeon, 1GB RAM.
- ✓ Servidor de la Aplicación, con 2 procesadores Intel Xeon, 1 GB RAM.
- ✓ Servidor del Repositorio, con 2 procesadores Intel Xeon, 1 GB RAM.

3.2.7 Recursos Humanos

Rol	Recursos	Responsabilidades
Administrador de la calidad	1	- Definir el plan de prueba - Planificar los cursos de capacitación - Revisar los artefactos a entregar - Conformar el documento único de no conformidades
Diseñador se prueba		- Verificar cumplimiento de los estándares de codificación - Diseñar los casos de pruebas - Identificar los datos de prueba - Definir entorno de configuración de prueba
Probador		- Realizar los componentes de prueba - Ejecutar las pruebas - Evaluar el resultado de las pruebas

Jefe de sistema	2	- Validar la estabilidad de los módulos - Proporcionar los elementos de prueba
Programador	2	- Apoyar la implementación y ejecución de las pruebas de unidad.

3.2.8 Cronograma de las actividades del proceso de pruebas.

En esta sección se especifican las actividades, los responsables de dichas actividades, los participantes y fechas de realización, que están sujetas a cambio en correspondencia con el nivel de desarrollo de software. A continuación se muestra el cronograma de actividades que conforma la estrategia de prueba.

No.	Tarea	Fecha	Responsable	Participantes	Observaciones
1	Estudio de las herramientas de prueba	2/12/09	Administrador de la Calidad	Equipo de Calidad	
2	Elaboración del plan de prueba	18/1/10	Administrador de la Calidad	Jefe del equipo de prueba	
3	Capacitación del Sistema	11/2/10	Jefe de sistema	Equipo de Calidad	
4	Capacitación del negocio	12/2/10	Jefe de sistema	Equipo de Calidad	
5	Elaboración de los casos de prueba	10/5/10	Administrador de la Calidad	Equipo de Calidad	
6	Aplicación Pruebas de Caja blanca	20/5/10	Administrador de la Calidad	Equipo de Calidad	

7	Informe de No conformidades	26/5/10	Administrador de la Calidad	Administrador de la Calidad	
8	Entrega de no conformidades al equipo de desarrolladores	26/5/10	Administrador de la Calidad	Administrador de la Calidad	
9	Evaluación de los resultados.	4/6/10	Equipo de Calidad	Administrador de la Calidad	
10	Aplicación de métricas	5/6/10	Administrador de la Calidad	Equipo de Calidad	

3.2.9 Artefacto que se generan.

Cada uno de los siguientes artefactos constituye al resultado de una actividad específica.

- ✓ Plan de prueba.
- ✓ Datos de prueba.
- ✓ Diseño de casos de pruebas de Caja Blanca.
- ✓ Documento de no conformidades.
- ✓ Lista de Chequeo.
- ✓ Sumario de evaluación de las pruebas.

3.2.10 Configuración del entorno de prueba.

La configuración del entorno de prueba se detalla en la figura 17 mostrando la distribución de hardware y software para llevar a cabo las pruebas, se muestran además periféricos de entrada y salida, así como herramientas que faciliten el proceso de prueba:

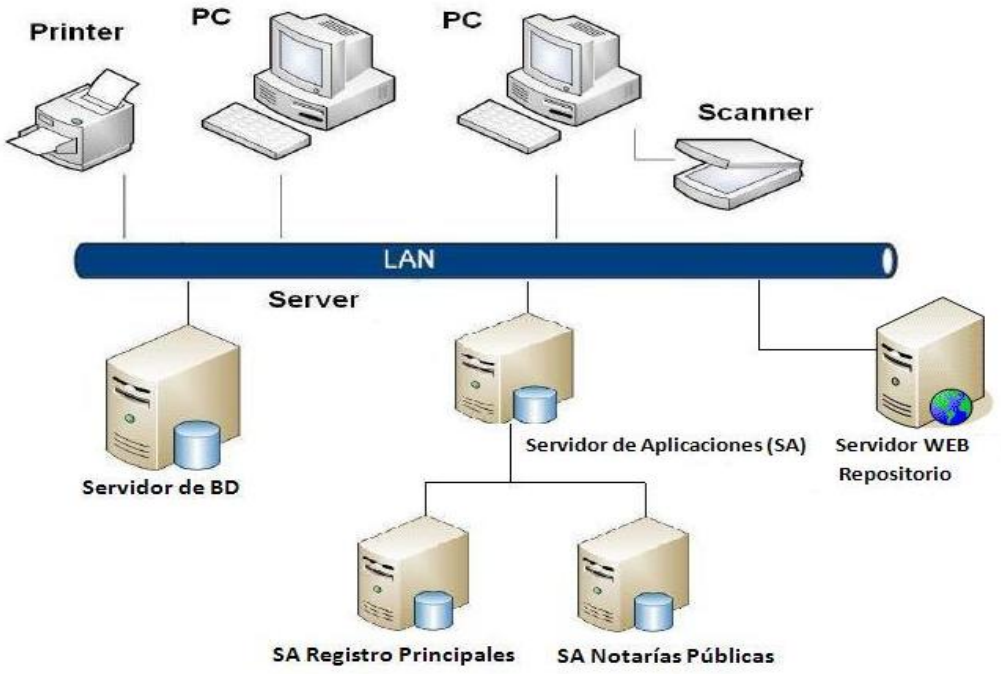


Figura 17 Configuración del entorno de prueba.

3.3 Datos de Pruebas.

Los datos de pruebas son de gran importancia para poder realizar los casos de prueba, ya que especifican un listado de datos reales referentes al negocio. En el anexo 2 se muestran los datos válidos que pueden ser utilizados en los sistemas Registros Principales y Notarías Públicas, especificándose también datos inválidos para comprobar las funcionalidades de los sistemas.

3.4 Diseño de pruebas de Caja Blanca.

Para el diseño de los casos de pruebas se analizan las funcionalidades y se le realizan una serie de pasos como se muestra a continuación. Es válido destacar que basada en la estructura arquitectónica del sistema, se le hicieron pruebas a cada uno de los casos de uso mostrados en las capas más sensibles y de mayor importancia según sus funcionalidades específicas. En este epígrafe se muestran ejemplos de prueba del caso de uso Buscar Persona Natural y Generar Número de Trámite en la capa de Negocio.

3.4.1 Listas de chequeo.

La lista de chequeo fue aplica antes de desarrollar los casos de prueba como técnica para filtrar los errores introducidos en la generación del código, verificando así la estructura del código obteniéndose como resultado final una calificación de Correcto (Ver Anexo 1).

3.4.2 Diseño de casos de prueba.

3.4.2.1 Caso de Prueba Buscar Persona Natural.

La siguiente funcionalidad es común para los dos sistemas (Registros Principales y Notarías Públicas).

Diseño del caso de prueba.

Descripción de la funcionalidad: Método encargado de devolver la persona natural.

Condiciones de Ejecución:

- ✓ El usuario debe tener permiso de escritura para realizarlas pruebas.
- ✓ Debe estar configurado el [SetUp] (se configura para tener acceso a la base de datos).
- ✓ Se debe cumplir con la estructura de datos definida.

En primera instancia se construye el **grafo de flujo**:

Luego se calcula la **complejidad ciclomática** y se puede verificar con la opción del Visual Studio 2008 de calcular métricas (Figura 14):

Complejidad = aristas – nodos + 2

Complejidad = 1 - 2 + 2 = 1

Complejidad ciclomática en Visual Studio:

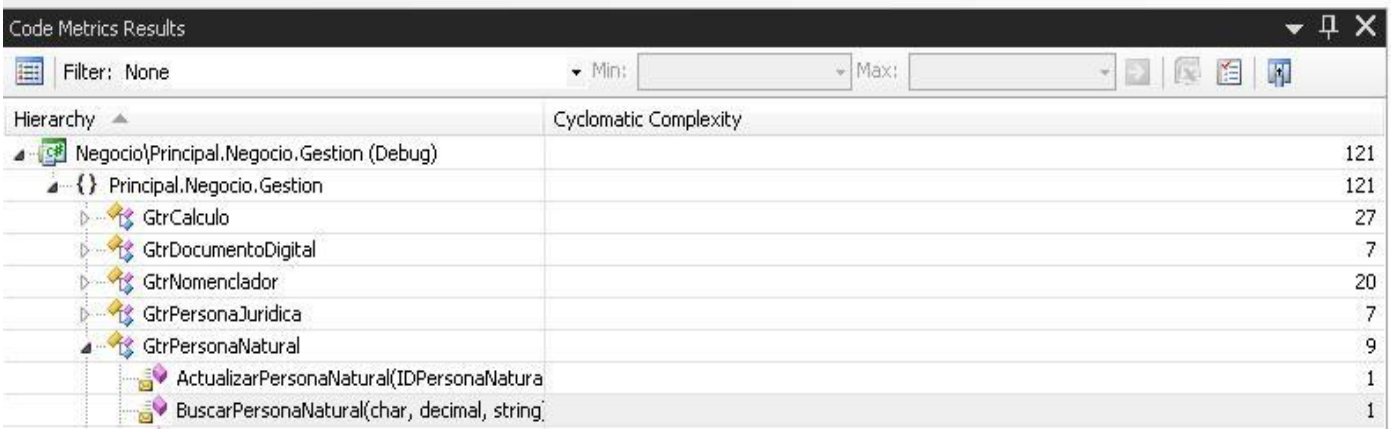


Figura 18. Cálculo de métrica complejidad ciclomática en Visual Studio 2008.

A continuación se muestra el procedimiento de prueba a seguir para la implementación y ejecución de la misma:

Flujo Central

1. El usuario con permiso de escritura accede a la aplicación donde realizará la clase de prueba.
2. El usuario crea la clase de prueba con las bibliotecas necesarias para realizar las mismas.
3. El sistema guarda los cambios satisfactoriamente.
4. El usuario realiza los métodos de prueba con meta atributo [test] encima de cada uno, para implementar las pruebas al código.
5. El sistema guarda los cambios satisfactoriamente.
6. El usuario especifica los datos correspondientes a la prueba:
   - letraCedula
   - numeroCedula
   - pasaporte
7. El usuario ejecuta (run) las pruebas con el compilador del Nunit.
8. El sistema compila las pruebas generando el archivo de prueba.
9. El sistema muestra los resultados de las pruebas.

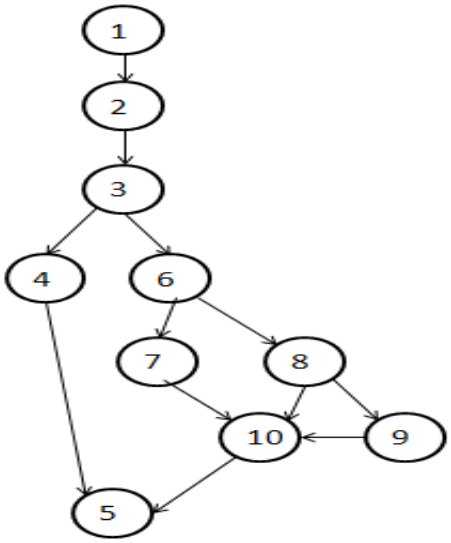
3.4.2.2 Caso de Prueba Generar número del trámite.

Esta funcionalidad se encuentra dentro del caso de uso gestionar trámite de inscripción.

Diseño del caso de prueba.

Descripción de la funcionalidad: Este método cuando se crea un nuevo trámite es el encargado de crearle un número único para el trámite.

Grafo de flujo:



Luego de la construcción del grafo de flujo se calcula la complejidad ciclomática:

Complejidad = aristas – nodos + 2

Complejidad = 12 - 10 + 2 = 4

Se compara con el cálculo de la complejidad ciclomática del Visual Studio la cual coincide como se muestra en la siguiente figura.

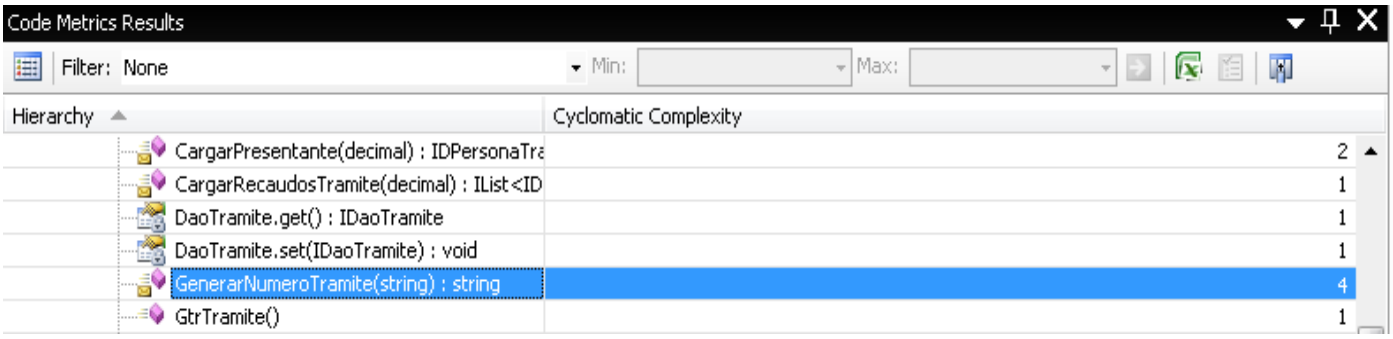


Figura 19 Cálculo de la complejidad ciclomática en el Visual Studio 2008.

Flujo Central:

1. El usuario con permiso de escritura accede a la aplicación donde realizará la clase de prueba.
2. El usuario crea la clase de prueba con las bibliotecas necesarias para realizar las mismas.
3. El sistema guarda los cambios satisfactoriamente.
4. El usuario realiza los métodos de prueba con meta atributo [test] encima de cada uno para implementar las pruebas al código.
5. El sistema guarda los cambios satisfactoriamente.
6. El usuario especifica los datos correspondientes a la prueba:
    - consecutivo
7. El usuario ejecuta (run) las pruebas con el compilador del Nunit.
8. El sistema compila las pruebas generando el archivo de prueba.
9. El sistema muestra los resultados de las pruebas.

3.5 Implementación de las pruebas.

Una vez instalado el ReSharper se crea una clase y se agrega la librería NUnit.Framework. En esta nueva clase se debe especificar el atributo [TestFixture] que indica que se va implementar un código de pruebas unitarias, este atributo se posiciona antes del nombre de la clase y para la implementación del método de prueba se le especifica el atributo [Test].

En la implementación de las pruebas se utilizan métodos propios de clase Assertion (Assert) que se utilizan para la comparación de valores, como AreEqual que compara dos tipos datos y el método Assert.AreSame que compara si dos valores son parecidos.

Durante el proceso de implementación primeramente se establece la conexión con la base de datos como se muestra en la figura 20.

```
[SetUp]
public void IniciarContexto()
{
    string conxString =
        "Data Source=RP213;User Id=usuario_213;Password=Data Source=RP213;User Id=usuario_213;
         Password=CslNMWU73oQbONy1dFos3361tWO=;;";
    Base.IniciarContexto(conxString, Base.TipoConexion.Oracle, true);
}
```

Figura 20 Establecer conexión con la Base de Datos.

En el caso de la implementación de la prueba Buscar Persona Natural se definen los datos de prueba para validar que la búsqueda de la persona sea correcta, para ello se pasa por parámetro (V, 22212732," ") y comprueba si la persona "BARBARA" es igual al valor esperado de la persona (Ver Figura 21).

```
[Test]
public void ProbandoBuscarPersonaNatural()
{
    persona = Base.Resolver<IGtrPersonaNatural>();
    var letra = char.MinValue;
    IDPersonaNatural pers = persona.BuscarPersonaNatural('V', 22212732, "");

    Assert.AreEqual("BARBARA", pers.PrimerNombre);

}
```

Figura 21. EC Buscar Persona Natural.

En el caso de la prueba Generar los Números de Trámites en los diferentes trimestres definidos (1,2, 3, 4) se le pasa al método como parámetro un # consecutivo para generar el número como se muestra en la Figura 22.

```
[Test]
public void ProbandoGenerarNumeroTramite()
{
    obj = Base.Resolver<IGtrTramite>();
    string num = obj.GenerarNumeroTramite("1");

    Assert.AreNotEqual("", num);
}
```

Figura 22 Implementación del método Generar número de trámite.

3.6 Ejecución de las pruebas.

3.6.1 Ejecución del Caso de Prueba Buscar Persona Natural.

Durante el proceso de ejecución se lleva a cabo la prueba, se registra el resultado obtenido como se muestra en la figura 23. Donde se observa que el resultado de la prueba es satisfactorio, mostrándose en la pantalla el color verde.

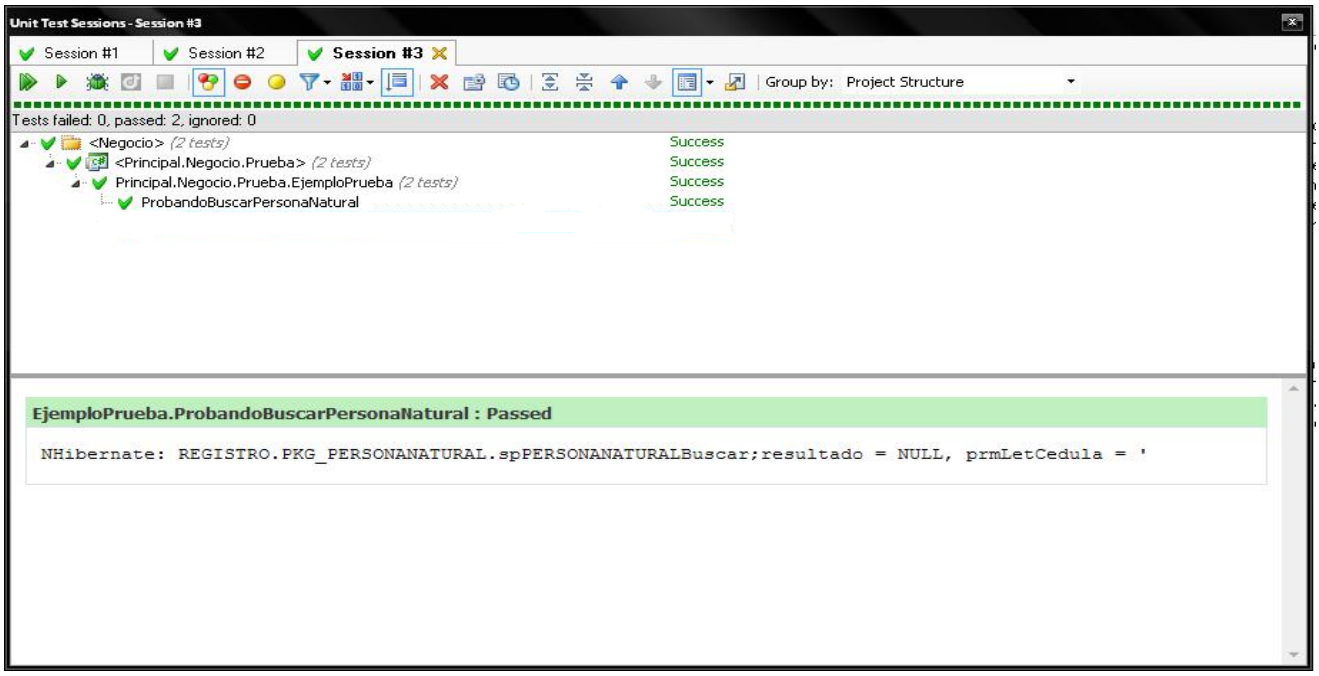


Figura 23 Pantalla de ejecución del NUnit.

Luego se procede a registrar los resultados de la prueba en la siguiente tabla de la planilla del caso de prueba.

Id del escenario	Escenario	letraCedula	numeroCedula	pasaporte	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Buscar persona natural.	V	V	V	El sistema debe buscar una persona natural.	Satisfactoria
		I	V	V	El sistema debe devolver error.	Satisfactoria
		V	I	V	El sistema debe devolver error.	Satisfactoria
		I	I	I	El sistema debe devolver error.	Satisfactoria

3.6.2 Ejecución del Caso de Prueba Generar Número del Trámite.

Durante el proceso de ejecución se lleva a cabo la prueba como se muestra en la figura 24 y se registran los resultados en la tabla de la planilla de casos de pruebas. Donde se observa que el resultado de la prueba es no satisfactorio, mostrándose en la pantalla el color rojo.

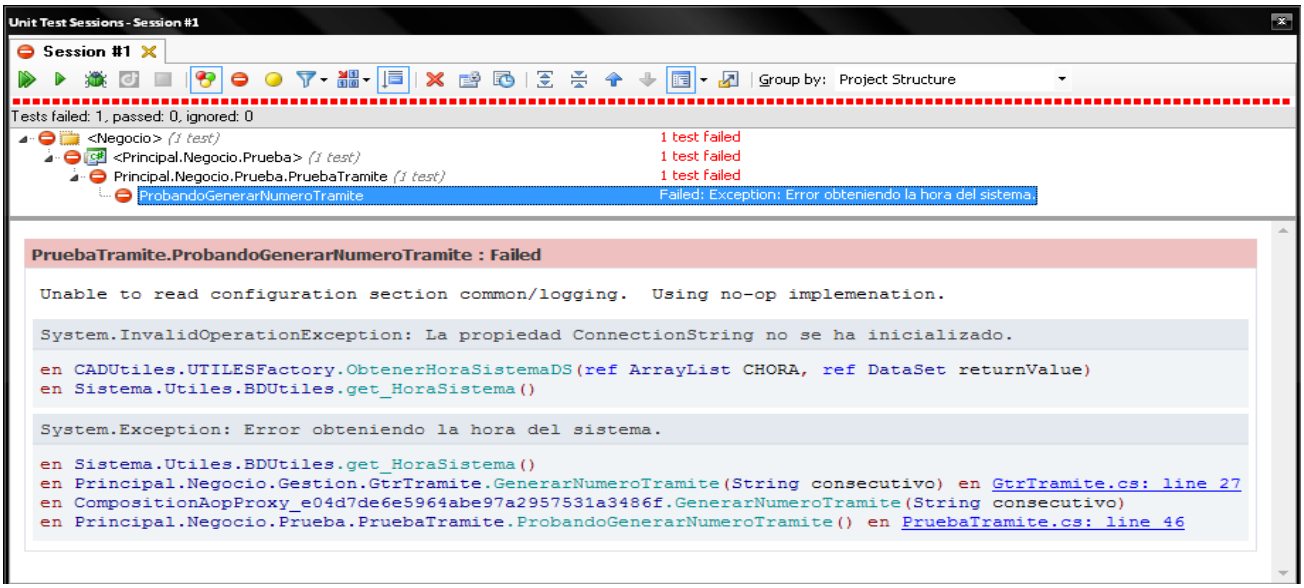


Figura 24 Resultado de la prueba.

Id del escenario	Escenario	Número consecutivo	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Generar número de trámite_ trimestre 1.	V	El sistema debe generar el número de trámite.	No Satisfactoria
EC 1.2	Generar número de trámite_ trimestre 2.	V	El sistema debe generar el número de trámite.	No Satisfactoria
EC 1.3	Generar número de trámite_ trimestre 3.	V	El sistema debe generar el número de trámite.	No Satisfactoria
EC 1.4	Generar número de trámite_ trimestre 4.	V	El sistema debe generar el número de trámite.	No Satisfactoria

3.7 Evaluación de las pruebas.

Durante el proceso de prueba de Caja Blanca de los procesos Revisión y Cálculo se aplicaron un total de 32 pruebas tanto a la capa del negocio como a la de acceso a datos, de las cuales 26 funcionalidades procedieron satisfactoriamente y 6 presentaron errores, resaltar que a los nomencladores utilizados se les realizo pruebas de unidad. (Ver Anexo 3)

Obteniendo los siguientes resultados en el proceso de pruebas:

- ✓ Casos de pruebas diseñados: 32
- ✓ Casos de pruebas aplicados: 32
- ✓ Casos de pruebas con resultados satisfactorios: 26

3.7.1 Resumen de No Conformidades de pruebas de Caja Blanca.

Durante las pruebas de Caja Blanca de los procesos de Revisión y Cálculo, en la primera iteración de se obtuvieron las siguientes no conformidades:

Elemento	No	No conformidad	Aspecto correspondiente	Etapa de detección	Clasificación	Estado NC	Respuesta del Equipo Desarrollo
Código	1	Permite entrar vacío el atributo descripción municipio.	Se encuentra en la funcionalidad Salvar y actualizar persona natural del CU Persona natural.	Prueba	Significativa	PD: 19/05/2010	
Código	*2*	Permite entrar 0 en l variable identidad federal sea.	Se encuentra en la funcionalidad Salvar y actualizar persona natural del CU Persona natural.	Prueba	Significativa	PD: 19/05/2010	

Código	3	Índice fuera de intervalo	Se encuentra en la funcionalidad Buscar persona jurídica dentro del CU Buscar persona jurídica.	Prueba	Significativa	PD: 20/05/2010	
Código	4	Permite entrar una sola letra sin los números y permite entrar números sin la letra que debe tener delante el atributo RIF.	Se encuentra en la funcionalidad Actualizar persona jurídica del CU Buscar Persona Jurídica.	Prueba	Significativa	PD: 20/05/2010	
Código	5	Permite la entrada del atributo identificador del trámite faltándole un 0.	Se encuentra en la funcionalidad Cargar exenciones del CU Gestionar Trámite.	Prueba	Significativa	PD: 20/05/2010	
Código	6	Error obteniendo la hora del sistema.	Se encuentra en la funcionalidad Generar número de trámite del CU Gestionar Trámite.	Prueba	Significativa	PD: 20/05/2010	
Código	7	No se ha inicializado la conexión Connecting string.	Se encuentra en la funcionalidad Cargar persona por trámite del CU Gestionar Trámite.	Prueba	Significativa	PD: 20/05/2010	

Código	8	No se ha inicializado la conexión Connecting string.	Se encuentra en la funcionalidad Cargar presentante del CU Gestionar trámite.	Prueba	Significativa	PD: 20/05/2010	
Código	9	No se encuentra el objeto actualizado.	Se encuentra en la funcionalidad SalvarActualizarNiñosNiñasAdolecente del CU SalvarActualizarNiñosNiñasAdolecente	Prueba	Significativa	PD: 29/05/2010	

3.7.2 Análisis de los resultados obtenidos en las pruebas de Caja Blanca y las pruebas de Caja Negra.

Durante las pruebas de Caja Negra de los procesos de Revisión y Cálculo, en la primera iteración se obtuvieron las siguientes no conformidades:

Elemento	No	No conformidad	Aspecto correspondiente	Etapa de detección	Clasificación	Estado NC	Respuesta del Equipo Desarrollo
Interfaz	1	El sistema permite insertar un correo electrónico sin @.	Inscripción/Revisión/Opción Buscar Tramite/botón siguiente/Representación Jurídica/Personas Naturales/botón Adicionar/Campo Correo electrónico	Prueba	Significativa	PD: 22/05/2010	
Docume	2	Los campos	Inscripción/revisión,/Inte	Prueba	Significativa	PD:	

ntación PIU		existentes en la aplicación no coinciden con los mostrados en el prototipo de interfaz de usuario.	rfaz Buscar Trámite/opción Buscar/ botón Siguiente/Opción representación jurídica/Adicionar persona jurídica/Llenar datos/Botón adicionar.			22/05/2010	
Interfaz	3	Al llenar todos los campos de la interfaz para buscar a la persona jurídica después de una búsqueda general se cierra la aplicación	Inscripción/revisión,/Interfaz Buscar Trámite/opción Buscar/ botón Siguiente/Opción representación jurídica/Adicionar persona jurídica/Llenar datos/Botón Buscar.	Prueba	Significativa	PD: 22/05/2010	
Interfaz	4	Al adicionar la representación jurídica el nombre de la interfaz tiene una falta ortográfica: Representación jurídica.	Inscripción/revisión/Interfaz Buscar Trámite/opción Buscar/ botón Siguiente/Opción representación jurídica/Interfaz Representación Jurídica/Personas Naturales.	Prueba	Significativa	PD: 22/05/2010	
Interfaz	5	En la interfaz de adicionar representación jurídica el campo numero tiene falta ortográfica debe ser número.	Inscripción/revisión,/Interfaz Buscar Trámite/opción Buscar/ botón Siguiente/Opción representación jurídica/Interfaz Representación Jurídica/Personas Jurídicas.	Prueba	Significativa	PD: 22/05/2010	
Interfaz	6	Al asociar una Persona Jurídica no se carga en la	Inscripción/revisión,/Interfaz Buscar Trámite/opción Buscar/ botón Siguiente/Opción	Prueba	Significativa	PD: 22/05/2010	

		interfaz de Representación Jurídica el tomo y el número asociado al mismo.	representación jurídica/Interfaz Representación Jurídica/Personas Jurídicas.				
Interfaz	7	Al asociar una persona jurídica al trámite el sistema no permite eliminarla.	Inscripción/revisión,/Interfaz Buscar Trámite/opción Buscar/ botón Siguiente/Opción representación jurídica/Interfaz Representación Jurídica/Personas Jurídicas.	Prueba	Significativa	PD: 22/05/2010	
Interfaz	8	Al insertar una persona natural él forma de correo que admite no es correcta.	Inscripción/revisión/Interfaz Buscar Trámite/opción Buscar/ botón Siguiente/Opción representación jurídica/Adicionar persona natural/Llenar datos/correo electrónico.	Prueba	Significativa	PD 12/06/2010	
Interfaz	9	En el campo Cédula no permite entrar números solo letras. Ver anexo 1	Notarias/ Revisión/ interfaz Trámites para revisión/seleccionar Buscar trámites/Introducir los criterios de búsqueda (Número de trámites, Fecha desde, Fecha hasta, Cédula, Pasaporte, Concluido, Revisión, Pendiente).	Prueba	Significativa	PD: 2/06/2010 Resuelta 12/06/2010	
Interfaz	10	No permite especificar la cantidad de folios del trámite y siempre aparece 18.5 como cálculo de	Interfaz seleccionar acto notarial	Prueba	Significativa	PD: 2/06/2010 PD 12/06/2010	

		los timbres fiscales					

Se puede verificar que de las no conformidades (NC) detectadas durante el proceso de prueba de Caja Negra solo 4 NC (1, 3, 7, 8) tienen trazabilidad con el código, este resultado permite constatar el funcionamiento de la estrategia de pruebas de Caja Blanca y entre sus objetivos está detectar la mayor cantidad de defectos en los sistemas, en una etapa anterior a las pruebas de Caja Negra. Es importante acotar que solamente se hizo una primera iteración de ambos tipos de prueba, donde a medida que las pruebas abarquen mayor cantidad de escenarios mayor cantidad de no conformidades se podrán eliminar, en el menor tiempo y menor costo de desarrollo para el proyecto.

3.7.3 Aplicación de las métricas.

3.7.3.1 Complejidad ciclomática.

La complejidad ciclomática aporta una valoración sobre la complejidad lógica del código definiendo los caminos independientes que posee un fragmento de código determinado.

En la siguiente figura se observa el cálculo de la complejidad ciclomática de cada método por la plataforma Visual Studio 2008.

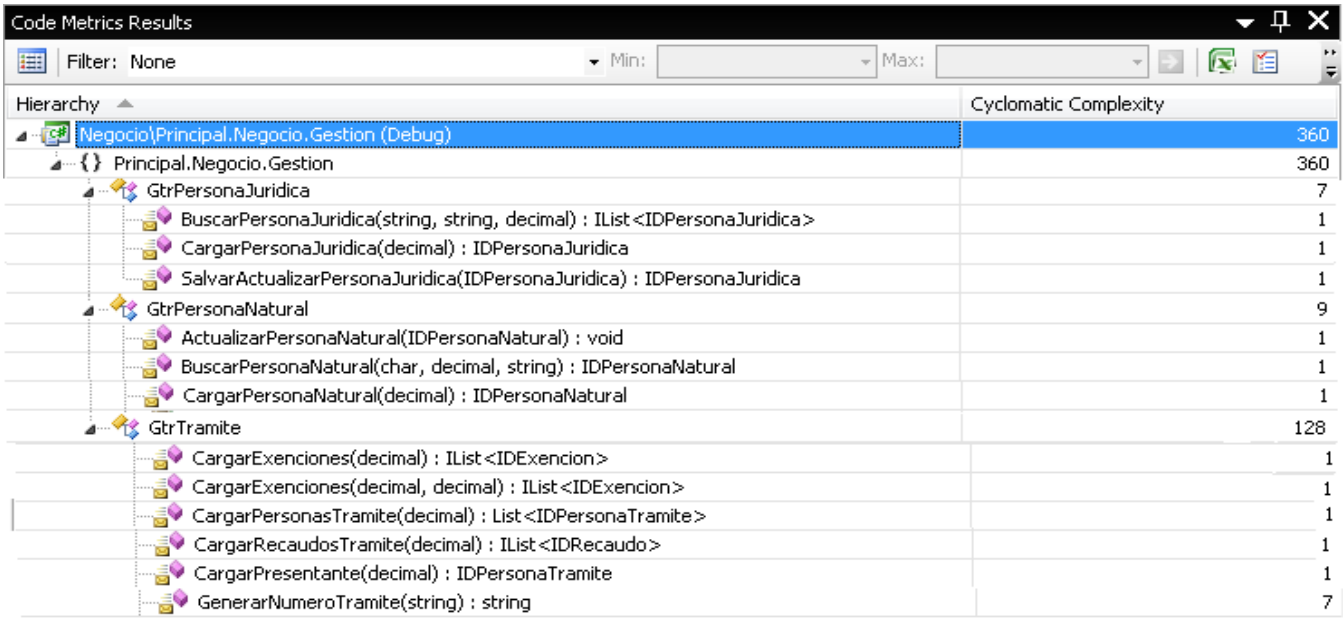


Figura 25. Complejidad ciclomática para el sistema Registros Principales.

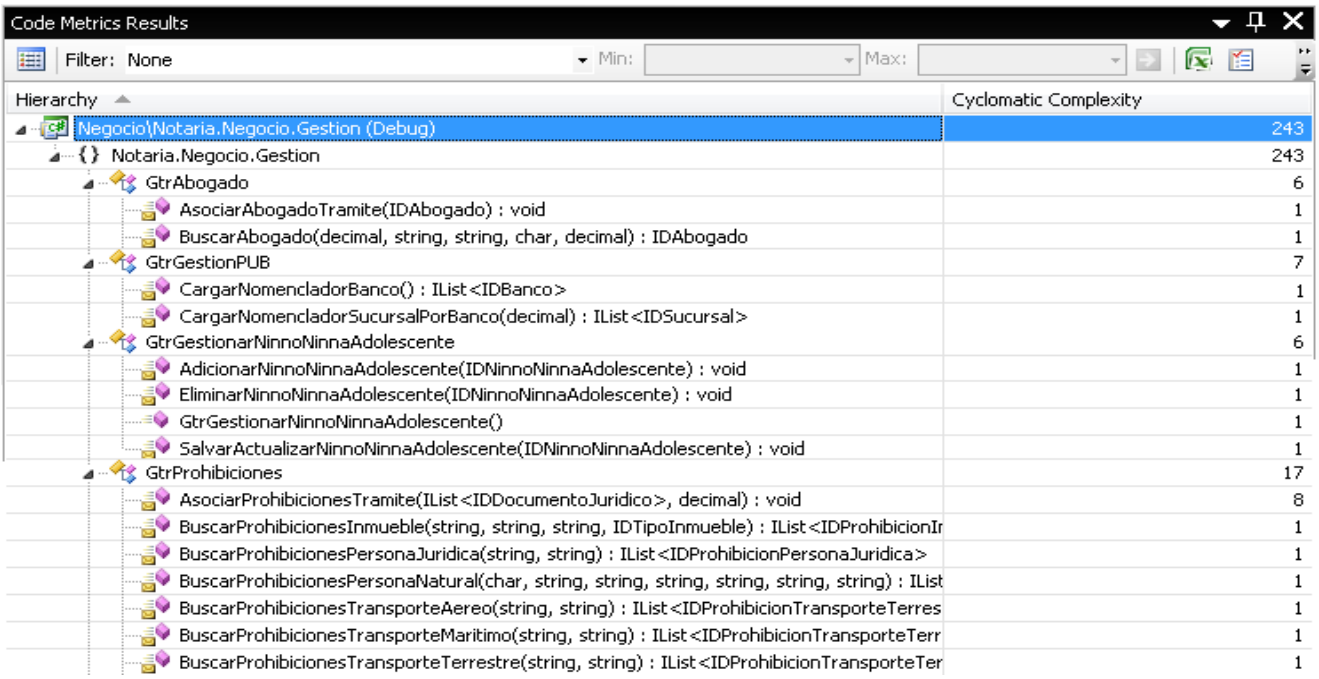


Figura 26. Complejidad ciclomática para el sistema Notarías Públicas.

3.7.3.2 Densidad de defecto (DD).

La siguiente métrica brinda la porción de los defectos con respecto a la cantidad de elementos probados. Permite un análisis estadístico atendiendo a la integridad y madurez del software. A continuación se presenta el cálculo de la métrica.

Densidad de Defecto de los sistemas Registros Principales y Notarías Públicas:

$$DD = {}^{TD}\!/_{CER} = {}^{9}\!/_{32} = 0.28$$

Se puede constatar que el nivel de densidad de defectos es de 0.28. Este resultado muestra como se ha demostrado madurez en el proceso de prueba, detectando la mayor cantidad de defectos en esta primera fase de la etapa de pruebas, antes de comenzar las pruebas de Caja Negra

3.7.3.3 Cobertura de las pruebas.

En el caso de esta métrica se obtiene un indicador de cuántos requisitos se han probado:

$$CP = {}^{CPE}\!/_{CPR} = {}^{32}\!/_{32} = 1$$

Cobertura de pruebas aplicadas: 100%

3.7.3.4 Madurez de las Pruebas

$$CP = {CPE}/{CPR} = {26}/{32} = 81.25$$

Madurez de las pruebas satisfactorias: 81.25 %

De forma general las métricas aplicadas mostraron que las pruebas de Caja Blanca cumplieron su principal objetivo hasta su ejecución de los primeros procesos de los sistemas Registros Principales y Notarías Públicas.

3.9 Conclusiones.

A partir de la aplicación de la estrategia de pruebas de Caja Blanca en los sistemas Registros Principales y Notarías Públicas, se logró verificar la correcta correspondencia con los estándares de codificación definidos a través de la lista de chequeo aplicada. La cobertura de pruebas respecto a la planificación fue total mostrando madurez en el proceso, y a partir del análisis de los resultados obtenidos mediante las métricas se evidenció que el 81.25 % de las mismas obtuvieron resultados satisfactorios. Este resultado se encuentra en estrecha relación con los obtenidos durante el proceso de prueba de caja negra mostrándose una disminución de las no conformidades que presentan trazabilidad con el código generado.

Mostrando así que el perfeccionamiento realizado teniendo en cuenta las características actuales del proyecto Registros y Notarías Fase II permitió mejorar el proceso de prueba.

Conclusiones Generales

Mediante este trabajo de diploma se logró dar cumplimiento a su principal objetivo el cual consistió en perfeccionar y aplicar una estrategia de prueba de Caja Blanca, apoyándose en la metodología RUP, ya que es la utilizada en el proyecto Registros y Notarías Fase II**.** Se redefinieron los roles que se considera que deben intervenir en cada uno de los procesos de prueba de Caja Blanca, así como los artefactos y herramientas necesarias que dieron paso a cada una de las actividades a desarrollar. Pudiendo llegar de esta forma a las siguientes conclusiones:

- Los casos de prueba diseñados fueron aplicados mediante las técnicas de prueba de Caja Blanca, combinándose la mayor cantidad clases posibles para la detección de los defectos. Logrando probar mediante la integración de la plataforma Visual Studio.Net con el framework NUnit el código de los procesos de Revisión y Cálculo.
- A partir de los resultados obtenidos se hizo un análisis mediante métricas donde se obtuvo el 81.25% de pruebas satisfactorias.
- Durante el proceso de prueba de Caja Negra se logró una disminución de las no conformidades que presentan trazabilidad con el código generado, contribuyendo de esta forma a la mejora del proceso de prueba.

Recomendaciones

A partir de la aplicación de la estrategia se recomienda:

1. Aplicar la estrategia perfeccionada de prueba de Caja Blanca en los restantes procesos de los sistemas Registros Principales y Notarías Públicas (Cálculo, Presentación, Procesamiento, Otorgamiento y Archivo).
2. Realizar capacitaciones al personal involucrado en el desarrollo de las pruebas para el correcto uso del framework NUnit para la detección de los errores en el código.
3. Aplicar la estrategia de pruebas a otros proyectos que presenten las mismas o semejantes características arquitectónicas de software.
4. Continuar con el desarrollo de la estrategia con el objetivo de enriquecerla y aportar soluciones novedosas que contribuyan a la mejora del proceso de prueba.

Bibliografía Referenciada

1 [Tecnobloggers, 2009]Tecnobloggers. (s.f.). Recuperado el 2009 de diciembre de 12, de Tecnobloggers: Tecnobloggers: http://blogs.cibersur.com

2 [*Calisoft, 2009*]*Calisoft*. (s.f.). Recuperado el 10 de diciembre de 2009, de Calisoft: http://calisoft.uci.cu/index.php?option=com_content&view=article&id=34&Itemid=27

[Estándar ISO 12.207, 1991] ISO. [Online] 1991.

3 [Soto, 2010]Soto, L. (s.f.). *Definición de la Calidad*. Recuperado el 10 de febrero de 2010, de Definición de la Calidad: http://www.mitecnologico.com/Main/DefinicionCalidadDeSoftware

4 [Oscar M., 1995] Oscar M., F. C., García León, D., & Beltrán Benavides, A. (diciembre de 1995). *Un enfoque actual sobre la calidad del software*. Recuperado el 10 de febrero de 2010, de Un enfoque actual sobre la calidad del software: http://www.bvs.sld.cu/revistas/aci/vol3_3_95/aci05395.htm

5 [IEEE, 1991] IEEE. ***Metrics***. [Online] 1991.

6 [Garcerant, 2008] Garcerant, I. (15 de marzo de 2008). *Tecnología y Synergix*. Recuperado el 11 de febrero de 2010, de Tecnología y Synergix: http://synergix.wordpress.com/2008/03/15/definimos-pruebas-de-unidad-como/

7 [E-Software, 2010]. E-Software. *E-Software.* [En línea] [Citado el: 10 de febrero de 2010.] http://www.e-software.com.ar/como_lo_hacemos.html#.

[Pressman, 2002] Pressman, R. (2002). Ingeniería del Software: Un enfoque Práctico. McGraw Hill.

8 [Díaz, 2010] Díaz, J. F. (s.f.). Métodos de Prueba de Programas. Recuperado el 11 de febrero de 2010, de Métodos de Prueba de Programas: http://www.galeon.com/neoprogramadores/met_test.htm

9 [Fernandez, 2004]Lanvin, Daniel Fernández. Desarrollo de una metodología para un nuevo paradigma de desarrollo de software. 2004.

10 [González, Durán, 2008]*Estrategia para la aplicación de Pruebas de Caja Blanca y Caja Negra al proyecto Registros y Notarías.*Habana, Cuba2008

11 [Calisoft, 2010] Programa de mejora, Roles y Responsabilidades, Cuba 2010

Bibliografía Consultada

1 [Facultad Informática, 2001] *Prueba y ciclo de vida*2001

2 [Mosquera, 2001]*Implementación de una metodología y herramienta de pruebas para el grupo de desarrollo de software*

3 [Cueva, 1999]*Calidad del Software* España1999

4 [Carrasco, 1995]Un enfoque actual sobre la calidad del software*Un enfoque actual sobre la calidad del software*http://www.bvs.sld.cu/revistas/aci/vol3_3_95/aci05395.htm

5 [Bañeres, 2005]*Visión ejecutiva de procesos y prácticas*2005

6 [Martin, 2005]*Prueba de software*2005

7 [Ginebra, 2000]*ISO 9001*2000

8 [Ruiz, 1999]*La norma ISO 14764*1999

9 [D'Onofrio, 2002]Probando software y números de versión*Probando software y números de versión*http://www.elguille.info/clipper/probando.htm

10 [Barrios, 2007]Métodos de prueba de caja blanca*Métodos de prueba de caja blanca*http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node26.html

11 [Vazquez, 2004]Ingenieria de Software XI*Ingenieria de Software XI*http://eclases.tripod.com/id21.html

12 [2005]*NORMATIVA SOBRE CALIDAD*2005

13[Mañas, 1994] Prueba de Programas*Prueba de Programas*http://www.lab.dit.upm.es/~lprg/material/apuntes/pruebas/testing.htm

14 [2009]Proceso de prueba*Proceso de prueba*http://blogs.cibersur.com/tecnobloggers/blog/2009/12/30/hacia-un-proceso-de-pruebas-mas-eficiente/

15 [2006]Gestión de la Calidad del Software*Gestión de la Calidad del Software*http://softqm.blogspot.com/2006/11/gestin-de-la-calidad-del-software.html

Anexo 1

Resultado de la aplicación de la lista de chequeo.

Evaluación Final: Correcto

Nivel	Criterio de evaluación	Evaluación	N.P.	Observaciones
!!	¿Cada IF tiene su llave de inicio y fin?	++		
!!	¿Las sentencias foreach tienen la siguiente forma: Foreach (instancia in Lista) { sentencias; }?	++		
!!	¿Las sentencias while tienen la siguiente forma: while (condición) { sentencias; }?		X	La sentencia especificada no se encuentra en el código.
!!	¿Las sentencias switch tiene la siguiente forma switch (condición) { case A: ... break; case B: ... break; default: ... break; }?		X	La sentencia especificada no se encuentra en el código.

!!	¿Las sentencias try-catch tienen la siguiente forma:
	try
	{
	...
	}
	catch (Exception) {}?
		++		

!!	¿Se usan paréntesis en expresiones que implican distintos operadores ejemplo if ((a == b) && (c == d))?	++		

!!	¿Se inicializan las variables locales a medida que son declaradas?	++		
!	¿Cada operación tiene un nombre que describe qué es lo que hace la operación?	++		
!!	¿El camino normal a través de cada operación es distinguible de otros caminos excepcionales o alternativos?	++		
!	¿Las variables están bien nombradas?	+		
!!	¿El nombre de los parámetros es significativo?	++		

El nivel de la pregunta será: (!!) Muy importante (!) Menos importante

Las evaluaciones serán: **#**

(--) **Incorrectos –** no se encuentran como están definidos. 0

(-) **Ambiguos –** no es claro. 0

(+) **Preciso –** se entiende lo que quiere decir. 1

(++) **Correctos –** está correctamente. 8

Total **9**

Anexo 2

Estructura de los datos

Tabla 1: Datos de pruebas.
Tabla 1.1 Buscar persona natural.

Campo	Datos Válidos	Datos Inválidos
letraCedula	Formato: alfabético.	Que los datos se inserten fuera del formato definido.
númeroCedula	Número con el siguiente formato: 8 dígitos 12345678/12345678.	Que los datos se inserten fuera del formato definido.
pasaporte	Formato: alfanumérico, longitud máxima: 16.	Longitud > 0 y con menos de 16 dígitos.

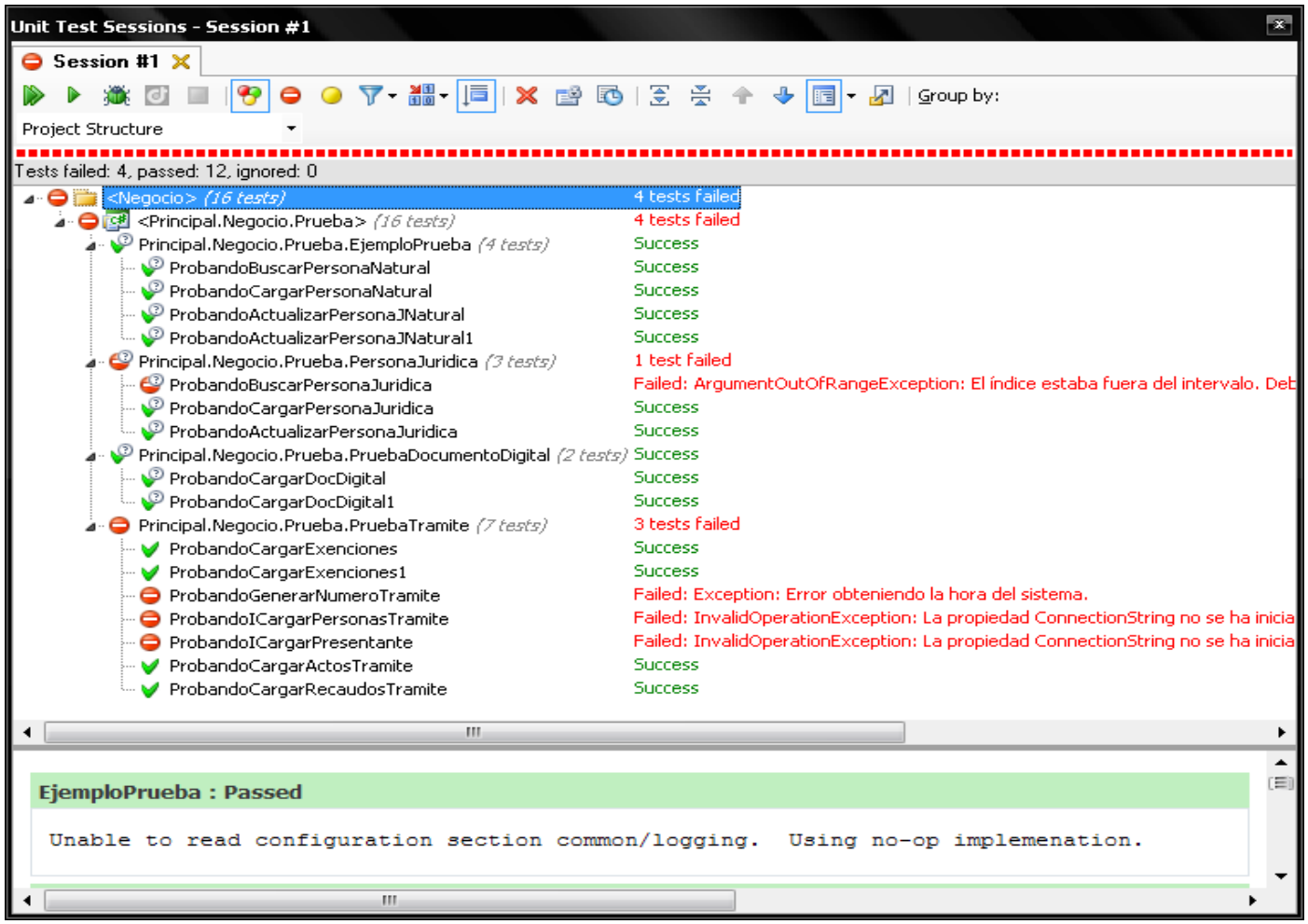
Tabla 1.2 Generar número de tramite 1,2,3,4 trimestre.

Campo	Datos Válidos	Datos Inválidos
consecutivo	Número mayor que 0	Números menores o iguales que 0

Anexo 3

Resumen de resultados de las pruebas de Caja Blanca de los sistemas Registros Principales y Notarías Públicas.

Sistema Registros Principales



Sistema Notarías Públicas.