Universidad de las Ciencias Informáticas

Facultad 5

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Título: “Desarrollo de una estrategia de prueba para el Sistema Integral de perforación de Pozo (SIPP)”.

Autor: Rosa Mª. Ching Zuñiga.

Tutor: Ing. Mariela Cepero Núñez.

Co-tutor: Ing. Annierys Martínez González.

Ciudad de La Habana, Junio 2012

“Año 54 de la Revolución”

FRASE

El optimismo es el peligro ocupacional de la programación, la prueba, el tratamiento".

Kent Beck.

DECLARACIÓN DE AUTORÍA

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

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

Rosa Mª. Ching Zuñiga Ing. Mariela Cepero Núñez Ing. Annierys Martínez González

_____ _____ _____

Firma del Autor Firma del Tutor Firma del Co-Tutor

DATOS DE CONTACTO

Mariela Cepero Núñez: Ingeniera en Ciencias Informáticas, perteneciente al Departamento Integración y Despliegue del Centro de Informática Industrial. Profesor instructor con 5 años de experiencia docente y 6 años de experiencia en la producción.

Annierys Martínez González: Ingeniera en Ciencias Informáticas, adiestrada, desempeña el rol de probadora en el Departamento Integración y Despliegue del Centro de Desarrollo de Informática Industrial (CEDIN).

AGRADECIMIENTOS

Ante todo agradecer a todas las personas que de una forma u otra me han ayudo en todo momento a cumplir todos mis sueños. No puedo dejar de mencionar a mi abuela, a mis padres hermana, ttos y ttas primos mi familia en general, mis padrinos amigos vecinos a mis profesores desde los inicios de mi vida estudiantil hasta mis tutores y oponente, tribunal en general, a mis compañeros de apartamento, de aula, también agradecer a todos los trabajadores de la dieta que sin su apoyo y amor no hubiese sido posible permanecer tan lejos de mi hogar, amiga y hermana Dailé Osdany que ha sido para mi más que un amigo.

DEDICATORIA

A mis padres Isabel y Roberto. A alguien muy especial en mi vida que es mi abuela Rosa y a la memoria de mis abuelos Felipe y Nena.

RESUMEN

La calidad de software es un aspecto muy importante a la hora de comercializar un producto, pues está en juego el prestigio y moral de la institución desarrolladora. El proceso de prueba es fundamental para que un producto cuente con calidad. Durante la creación de un software se le aplican distintos tipos de pruebas, de esta manera se garantiza que cuente con una mayor calidad.

En estos momentos la Universidad de las Ciencias Informática (UCI) específicamente el Centro de Informática Industrial (CEDIN) se encuentra en la realización de varios productos, particularmente en la construcción del producto MaximusDrillPro. El mismo surge después de la solicitud planteada al centro por la Empresa Cuba-Petróleo (CUPET) al tener la necesidad de gestionar la información generada durante el proceso de perforación de los pozos de petróleo, por lo que dicho proceso refiere a la economía del país es de vital importancia su correcto funcionamiento con la calidad máxima.

Esta investigación se centra en los conceptos relacionados con la calidad y pruebas del software. Se aplica una estrategia de prueba para que el proceso tuviese un resultado óptimo. Al aplicar la estrategia al producto se realizaron pruebas funcionales, pruebas de carga, pruebas de stress, pruebas de volumen, pruebas de replicación, pruebas de seguridad, obteniéndose los artefactos generados en el proceso plan de pruebas, diseños de casos de pruebas, evaluaciones de pruebas, registros de pruebas y un listado con las no conformidades detectadas.

Palabras clave
Calidad, pruebas de software, estrategia.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE TABLAS

INTRODUCCIÓN

En el mundo existen disímiles organizaciones que se dedican al aseguramiento de la calidad del software, algunas se ocupan de brindar servicio de supervisión y consultoría sobre el tema. En ocasiones las grandes compañías productoras de software como Apple o Microsoft tienen más de un grupo destinado al control de la calidad. La definición de calidad varia según el ámbito al cual esta guiado algunos se basa en el cumplimento de una característica específica, otra a la satisfacción de los clientes a los que está destinado el producto.

En Cuba el desarrollo de la informática es uno de los sectores priorizados, ejemplo de ello es la creación de los Joven Club y el surgimiento de la Universidad de las Ciencias Informática (UCI), por el Comandante en Jefe Fidel Castro, durante la llamada Batalla de Ideas en el 2002. Es la única de su tipo en el país, al tener vinculado el carácter Productivo - Formativo, en la misma se desarrollan distintos proyectos los cuales realizan softwares para la comercialización tanto a nivel nacional como internacional, por esta razón la calidad de los mismos constituyen un factor clave durante su desarrollo. Unos de los principales aspectos a tener en cuenta para que el producto tenga la calidad requerida es la ejecución de las pruebas durante todo el ciclo de vida del software, algunas de las pruebas que pueden ser aplicadas son: Funcionalidad, Usabilidad, Fiabilidad, Rendimiento, Soportabilidad, entre otras muchas.

La UCI no está exenta de velar por la calidad con que se realizan sus productos, por tal motivo cuenta con el Centro de Calidad para Soluciones Informáticas (Calisoft) que se dedica a liberar cada producto que se desarrolla en la universidad y en el país. Por su parte el Centro de Informática Industrial (CEDIN) que pertenece a la facultad 5, cuenta con un departamento de Integración y Despliegue en el que labora un grupo de pruebas que se ocupa de la detección de errores en los productos antes de ser enviados a Calisoft, con el objetivo de que tengan la mayor calidad posible y de esta manera reducir el período de liberación.

En el CEDIN se desarrollan varios proyectos entre ellos se encuentra el Sistema Integral de Perforación de Pozos (SIPP) que desarrolla el producto llamado MaximusDrillPro para gestionar y supervisar el flujo de información generada en el proceso de perforación de los pozo. Dicha solución ha sido desplegada como piloto en algunos pozos, pero aun no cubre todas las expectativas de los clientes, pues en varias ocasiones han detectado mal funcionamiento, el producto no responde de manera correcta ante la afluencia de varios usuarios al mismo tiempo. También como es un sistema distribuido la comunicación entre las bases de datos se demora y en ocasione existe perdida de información, el producto presenta vulnerabilidades pues los permisos no están centrados por roles posibilitando que todos los usuarios accedan a cualquier tipo de información, los clientes han redefinido los requisitos para próximas iteraciones, todos estos problemas son comunicados al equipo de desarrollo para su corrección, por lo que se hace necesario evitar el descontento de los mismos y proporcionarles mayor seguridad de que le sistema realmente funciona como ellos esperan. Ante los problemas existentes tiene la interrogante ¿Cómo lograr que el producto MaximusDrillPro cumpla con los requerimientos especificados y esté apta para satisfacer a los clientes? Constituyendo el **Problema Científico** de la investigación.

Se tiene como **Objeto de estudio** los procesos de pruebas de software enmarcando el **Campo de acción** en las pruebas de software para sistemas de gestión.

Para darle solución al problema propuesto se formula como **Objetivo General**: Desarrollar una estrategia de prueba para el producto MaximusDrillPro satisfagan al cliente.

Para dar cumplimiento al objetivo de este trabajo se plantean las siguientes **tareas de investigación**:

- Elaboración del marco teórico de la investigación a partir del estado del arte existente sobre el tema actualmente.
- Elaboración de la estrategia de prueba al producto MaximusDrillPro.

- Definición del conjunto de pruebas de software a ser implementadas para la aplicación MaximusDrillPro para el Sistema (SIPP).
- Aplicación de la estrategia definida.
- Valoración de los resultados obtenidos.

Idea a defender:

Con el desarrollo y aplicación de la estrategia de pruebas al producto MaximusDrillPro se logrará aumentar la calidad del mismo y la satisfacción del cliente.

Métodos Científicos de Investigación:

Para el desarrollo de esta investigación fue necesario utilizar algunos métodos teóricos y empíricos, ya que facilitan y guían la investigación de una forma más cómoda caracterizando el tema a tratar. Los métodos a utilizar durante la investigación son los siguientes:

Métodos Teóricos:

- Análisis - Síntesis: Permite analizar, estudiar, caracterizar e interpretar las teorías y documentos relacionados con el producto MaximusDrillPro, las pruebas de software, las estrategias de pruebas y sistemas de gestión con el fin de extraer los elementos más importantes de interés para la investigación.
- Histórico – Lógico: Para el estudio y análisis del estado del arte de las pruebas de software, herramientas, estrategias y otros aspectos de las pruebas más utilizadas. También se realizó un estudio de sobre la trayectoria de creación de la aplicación MaximusDrillPro y el proyecto SIPP.

Métodos Empíricos:

- Entrevista: La entrevista no es más que el acto informal donde se elaboran preguntas para realizarlas a especialistas y de esta manera obtener la información necesaria para validar la propuesta. Con la ayuda de este método se hace un acta donde se especifican las conformidades o no de los clientes, para de esta manera medir el cumplimiento del objetivo. Se entrevistaron especialistas en pruebas de software y bases de datos, así como al líder del proyecto Sipp. Para saber las recomendaciones de los tipos y que pruebas aplicar, en el caso del especialista en pruebas de software. El especialista de base de datos recomienda la herramienta para realizar las pruebas de replicación ya que es con la que se comunica las bases de datos del producto, también facilitó el entendimiento de la estructura de las bases de datos. Por medio de este método se puedo conocer la estructura de subsistema.

- Experimentación: Para evaluar y mejorar el software una vez aplicadas las pruebas y detectadas las no conformidades.

Estructura de la investigación:

La investigación está diseñada en tres capítulos

- Capítulo 1: Se aborda las principales definiciones de interés para la investigación, relacionadas con la calidad de software, pruebas de software, sistemas de gestión y tipos de prueba más comunes para los sistemas de gestión. Se decide definir una estrategia de prueba.

- Capítulo 2: Se define y obtiene la estrategia de pruebas con todos sus aspectos correspondientes para ser aplicada posteriormente.

- Capítulo 3: Se aplica la estrategia, así como las pruebas definidas en la misma. Se valorarán los resultados obtenidos.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se exponen los principales conceptos asociados a la investigación para lograr una mejor comprensión del tema a tratar. La calidad y las pruebas de software están definidas en diferentes niveles y tipos, por lo que se realiza una caracterización de dichas pruebas con el objetivo de profundizar y esclarecer los términos empleados durante el desarrollo del trabajo, dejando esclarecidos las posibles dudas que pudieran surgir con algunos términos utilizados.

1.1 Calidad

En el proceso creativo de un software la calidad es un factor de vital importancia, ya que en este aspecto se centran la correcta funcionalidad, seguridad y confiabilidad del producto. La calidad depende de muchos factores, es un concepto que se ha generalizado en los últimos tiempos en la industria del software, a continuación se presentan algunas definiciones.

La Real Academia de la Lengua Española (RAE), define que la calidad es “la propiedad o conjunto de propiedades inherentes a una cosa que permiten apreciarla como igual, mejor o peor que las restantes de su misma especie”.[1]

La Organización Internacional para la Estandarización (ISO) define calidad como “el conjunto de propiedades y de características de un producto o servicio, que le confieren aptitud para satisfacer necesidades explícitas o implícitas”.[2]

En cuanto a la calidad dentro en el proceso de desarrollo de software existen definiciones más específicas que involucran elementos de este dominio, entre ellas:

La calidad del software se define como la “concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente

documentados, y con las características implícitas que se espera de todo software desarrollado profesionalmente" Según Roger S. Pressman.[3]

La definición propuesta por Watts S. Humphrey en su libro Introducción al Proceso de Software Personal (PSP), plantea que la Calidad de Software "consiste en satisfacer las necesidades de los usuarios haciendo el trabajo de los mismos de una forma fiable y consistente. Esto requiere que el software que hagas tenga pocos defectos".[4]

Teniendo en cuenta las diferentes definiciones del término calidad, su adaptación estará en dependencia al medio en el que se esté enmarcando, puede concluirse que no es más que el indicador de satisfacción del cliente con el producto final y para esto es importante la realización de pruebas al software.

1.2 Pruebas de Software

Para que un producto software tenga la calidad requerida tiene un papel fundamental la aplicación de pruebas para la detección y corrección de errores durante todo el proceso de creación.

Según el glosario de términos de la IEEE[1] las pruebas constituyen "una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente".[5]

Pressman[2] a su vez enuncia que "las pruebas del software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación".[3]

Muchas pruebas se desarrollan basadas en casos de pruebas, los cuales no son más que una especificación de un caso para probar el sistema, en el cual se explica los pasos a seguir durante

[1] IEEE: Institute of Electrical and Electronics Engineers(Instituto de Ingenieros Eléctrico y Electrónica,)
[2]Roger S. Pressman: Es una autoridad reconocida internacionalmente por mejorar los procesos de prueba, ente otros aspectos.

la realización de las pruebas, incluyendo qué probar, con qué entradas y resultados y bajo qué condiciones. Como resultado de las pruebas se tienen las no conformidades que son error detectado durante el proceso de prueba ya sea de documentación o en el funcionamiento del software.

Los Objetivos de las Prueba son:

- Encontrar y documentar los defectos que puedan afectar la calidad del software.
- Validar que el software trabaje como fue diseñado.
- Validar y probar los requisitos que debe cumplir el software.
- Validar que los requisitos fueron implementados correctamente.
- La prueba no puede asegurar la ausencia de defectos, solo pueden demostrar que existen defectos en el software.[3]

Las pruebas resumiendo es un proceso que se le aplica a un software con la intención de encontrar la mayor cantidad de errores, siendo un buen caso de prueba aquel que tiene alta probabilidad de encontrar un error no descubierto antes.

1.2.1 Principios de las Pruebas de Software

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

- Las pruebas deberían empezar por lo pequeño y progresar hacia lo grande.
- Una buena prueba no es redundante.
- La principal dificultad del proceso de prueba es decidir cuándo parar.

- Evitar Casos de Prueba no planificados, no reusables y triviales a menos que el programa sea verdaderamente sencillo.
- Una parte necesaria de un caso de prueba es la definición del resultado esperado.
- Los Casos de Prueba tienen que ser escritos no solo para condiciones de entradas válidas y esperadas sino también para condiciones no válidas e inesperadas.
- Los Casos de Prueba tienen que ser escritos para generar las condiciones de salida deseadas.
- El número de errores sin descubrir es directamente proporcional al número de errores descubiertos.[6]

1.2.2 Niveles de Prueba

Las pruebas se encuentran en cada etapa del desarrollo del software. En la medida que el trabajo de los desarrolladores va aumentando el volumen de la aplicación, las pruebas van cambiando de estrategias y técnicas, presentándose como una nueva etapa, estas están definidas en niveles que tienen nuevos objetivos, entornos y resultados.

Los niveles de pruebas son:

Prueba de Unidad: Pruebas a nivel de clases. Las pruebas unitarias sólo son efectivas si se usan en conjunto con otras pruebas de software.

Prueba de Integración: Pruebas a nivel de componentes. Donde existen dos tipos: las ascendentes que son basadas en hilos y las descendentes con regresión basadas en uso.

Prueba de Validación o Aceptación: Pruebas pilotos realizadas por los clientes. Este tipo de prueba está enfocada en el nivel de requisitos, o sea se valida que el software cumpla con las especificaciones solicitadas.

Prueba de sistema: Pruebas al software aseguran el correcto funcionamiento del sistema, ingreso de datos, procesamiento y recuperación.[7]

1.2.3 Tipos de Pruebas

Las pruebas pueden clasificarse en dos grandes grupos, pruebas funcionales y no funcionales. Estas pruebas además cuentan con diferentes clasificaciones.

Funcionalidad	Función: Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso.
	Seguridad: Asegurar que los datos o el sistema solamente es accedido por los actores deseados.
	Volumen: Enfocada en verificar las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la Base de Datos.
Soportabilidad	Configuración: Enfocada a asegurar que el sistema funciona en diferentes configuraciones de hardware y software. Esta prueba es implementada también como prueba de rendimiento del sistema.
	Instalación: Enfocada a asegurar la instalación en diferentes configuraciones de hardware y software bajo diferentes condiciones (insuficiente espacio en disco, tecnología acorde, entre otras).
Rendimiento	Benchmark: Es un tipo de prueba que compara el rendimiento de un elemento nuevo o desconocido a uno de carga de trabajo de referencia conocido.
	Contención: Enfocada a la validación de las habilidades del elemento a probar para manejar aceptablemente la demanda de múltiples actores sobre un mismo recurso (registro de recursos, memoria).

	Carga: Usada para validar y valorar la aceptabilidad de los límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba permanece constante. La variación en carga es simular la carga de trabajo promedio y con picos que ocurre dentro de tolerancias operacionales normales.
	Performance profile: Enfocadas a monitorear el tiempo en flujo de ejecución, acceso a datos, en llamada a funciones y sistema para identificar y direccional los cuellos de botellas[3] y los procesos ineficientes.
Fiabilidad	Integridad: Enfocada a la valoración de la robustez (resistencia a fallos).
	Estructura: Enfocada a la valoración a la adherencia a su diseño y formación. Este tipo de prueba es hecho a las aplicaciones Web asegurando que todos los enlaces están conectados, el contenido deseado es mostrado y no hay contenido huérfano.
	Stress: Enfocada a evaluar cómo el sistema responde bajo condiciones anormales (extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible, recursos compartidos no disponible).
Usabilidad	Usabilidad: Prueba enfocada a factores humanos, estéticos, consistencia en la interfaz de usuario, ayuda sensitiva al contexto y en línea, asistente documentación de usuarios y materiales de entrenamiento.

Tabla 1 Tipos de pruebas[7]

Las pruebas funcionales se desarrollan para validar si el sistema cumple los requisitos establecidos. A continuación se analizan los dos métodos de pruebas de que se pueden aplicar a un software durante su ciclo de creación.

[3] Es cuando se realizan muchas solicitudes que no pueden ser atendidas al mismo tiempo formándose una lista de espera hasta que.se solucione.

1.2.4 Métodos de pruebas de software

Los métodos de pruebas no son más que una guía de procedimientos para realizar una prueba de software, tienen como objetivo proporcionar una vía más fácil para buscar errores garantizando la obtención de un producto de mayor calidad. Tradicionalmente se dividen en prueba de caja blanca y de prueba de caja negra.

1.2.4.1 Prueba de Caja Negra.

Las pruebas de Caja Negra se centran fundamentalmente en las funciones, entradas y salidas, por lo que no es necesario conocer la lógica del programa. Se llevan a cabo sobre la interfaz del software a probar, donde los Casos de Prueba pretenden demostrar que el software es operativo, que las entradas se aceptan y las salidas se producen de forma correcta y que la integridad de la información externa se mantiene.

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

Pressman muestra diferentes técnicas dentro de este método, entre las que se encuentran:

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

- **Partición equivalente**: se dirige a la definición de Casos de Prueba que descubran clases de errores, reduciendo así el número total de Casos de Prueba en uno más pequeño que sea manejable y que evalúe bien el software. Se toma un riesgo porque se escoge no probar todo, por lo que se necesita ser cuidadoso a la hora de escoger las clases.

- **Análisis de Valores Límite (AVL)**: es una técnica de diseño de Casos de Prueba que completa a la partición equivalente, seleccionando los Casos de Prueba en los extremos de la clase. En lugar de centrarse solamente en las condiciones de entrada, el AVL obtiene Casos de Prueba también para el campo de salida.

- **Prueba de comparación**: se utiliza en situaciones críticas en las que el software debe ser sumamente fiable. En estos casos se desarrolla una serie de versiones del programa siguiendo las mismas especificaciones a las cuales se les realiza pruebas con los mismos datos de entrada para asegurar la misma salida, en caso que se obtenga la misma salida, significa que todas las implementaciones son correctas, en caso contrario, es necesario revisar todas las versiones para encontrar la causa de las diferencias.[3]

Estas pruebas permiten encontrar:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las Bases de Datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.[8]

1.2.4.2 Prueba de Caja Blanca.

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

Este método cuenta con técnicas como:

- **Prueba del camino básico**: permite obtener una medida de la complejidad de un diseño procedimental, y utilizar esta medida como guía para la definición de una serie de caminos básicos de ejecución, diseñando Casos de Prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control que garanticen que cada camino se ejecuta al menos una vez.[3]

- **Prueba de condición**: se centra en la prueba de cada una de las condiciones del programa, diseña además Casos de Prueba que permiten ejercitar condiciones lógicas contenidas en el módulo de un programa, asegurando así que ninguna condición del programa contenga errores.[9]

- **Prueba del flujo de datos**: se basa en seleccionar diferentes caminos de prueba de forma que se pueda probar todas las definiciones, variables y estructuras de datos de los programas.[10]

- **Prueba de bucles**: es una prueba complementaria a la del camino básico y se basa en estudiar la validez de todos los bucles del programa.[10]

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

1.2.5 Metodologías en el proceso de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos y técnicas que ayudan a la documentación para el desarrollo de productos de software. Estas van indicando paso a paso todas las actividades a realizar para lograr el producto deseado. Indicando además,

qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener cada una. Existen varias metodologías aplicables para el desarrollo de un software se clasifican en ágiles y tradicionales. Las metodologías tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo del software. Enfatizando en la planificación total de todo el trabajo a realizar. Se centran especialmente en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas, notaciones para el modelado y documentación detallada. Por su parte las metodologías ágiles surgen como respuesta a problemas reales, se caracterizan por la efectividad que muestran en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad. Entre las metodologías tradicionales se encuentra RUP (Rational Unified Process).[11]

1.2.5.1 Proceso Unificado de Desarrollo de Software (RUP)

La metodología RUP sugiere su uso para proyectos nuevos o actualizaciones de sistemas existentes, y recomienda adoptarlo en forma gradual. Este es uno de los procesos más generales que existe, está enfocado a cualquier tipo de proyecto sea de desarrollo de software o no, se basa en la documentación generada en cada una de sus cuatro fases: 1.Inicial (puesta en marcha), 2. Elaboración (definición, análisis y diseño), 3. Construcción (desarrollo) y 4. Transición (fin del proyecto y puesta en producción) en las cuales se ejecutarán varias iteraciones (según el tamaño del proyecto).[12]

RUP durante su ciclo de vida define claramente quién, cómo, cuándo y qué hacerse en cada momento en el proyecto, para poderlo aplicar antes se tiene que adaptar a las características de la empresa, haciendo un minucioso análisis de factores como el tiempo, costos y todos los demás recursos involucrados en el proceso.

Una de las cualidades que se destacan de la metodología RUP es que todo el proceso se documenta mediante UML (Unified Modeling Language - Lenguaje de Modelado Unificado), está orientado a la arquitectura del sistema a realizar, indicando en qué orden debe de ser desarrollado, basándose en casos de uso para describir lo que se debe y espera del software.

Adicionalmente es iterativo e incremental, logrando en cada iteración el incremento de las funcionalidades hasta alcanzar el objetivo final.

RUP define un flujo de trabajo de prueba que será realizado durante todo el desarrollo del software, siendo más relevante en las etapas finales del ciclo de creación.

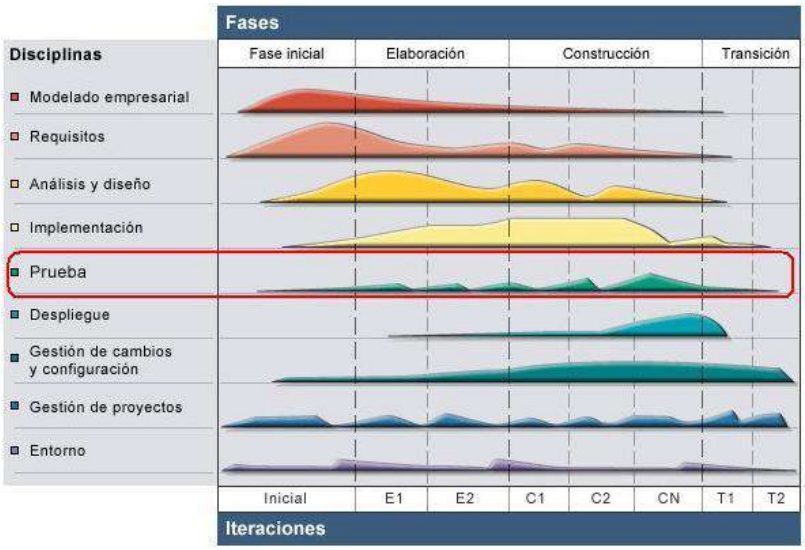


Figura 1 Arquitectura global de RUP

1.2.5.2 Pruebas en la Metodología RUP

El flujo de trabajo de prueba es el último dentro de los flujos de ingeniería (lo que no significa que se realice al final) y es el encargado de evaluar la calidad del producto que se desarrolla, pero no para aceptar o rechazar el producto al final del proceso de desarrollo, sino que debe ir integrado en todo el ciclo de vida. De forma general la disciplina de Prueba actúa como un proveedor de servicio a las otras disciplinas en muchos aspectos.

RUP propone que en cada fase las pruebas se comporten de la siguiente forma:

En la Fase de Inicio:

Se comienzan a considerar qué pruebas se requerirán y se van desarrollando algunos planes provisionales de prueba. No se realiza en esta etapa un trabajo significativo de pruebas. Se puede generar un modelo de pruebas algo primario en esta fase.

En la Fase de Elaboración:

El objetivo es asegurarse de que los subsistemas de todos los niveles y de todas las capas funcionen. Sólo se pueden probar los componentes ejecutables. Al empezar por las capas más bajas de la arquitectura se prueban los mecanismos de distribución, almacenamiento, recuperación y concurrencias de objetos, así como otros mecanismos de las capas inferiores del sistema. Al planificar las pruebas se seleccionan los objetivos que evaluarán la línea base de la arquitectura. Cuando se diseñan las pruebas se toman como base estos objetivos para identificar los casos de pruebas necesarios y organizar los procedimientos de pruebas para comprobar la sucesiva integración de subsistemas hasta completar la línea base. Con la comprobación de los componentes quedará listo para realizar las pruebas de integración. Concluyendo la integración del sistema, como queda definido por los casos de uso más importante se realizan las pruebas de sistema.

En la Fase de Construcción:

En esta fase las pruebas de unidad son una actividad fundamental. Al planificar las pruebas se seleccionarán los objetivos que comprueben las sucesivas construcciones y el propio sistema. Al diseñar las pruebas se determina cómo probar los requisitos en el conjunto de construcciones y se preparan casos y procedimientos de pruebas con este fin. Se realizan las pruebas de integración informando los resultados para tomar las medidas necesarias en casos de errores. Se realizan también pruebas del sistema al alcanzarse el estado de versión parcial del sistema, informando los resultados para tomar las medidas necesarias en casos de errores. Las pruebas se deben evaluar a medida que transcurren las pruebas de integración y del sistema comprobando que éstas alcancen los objetivos del plan de pruebas. Si una prueba no alcanza sus objetivos, los casos y procedimientos de pruebas deberán ser modificados para lograrlos.

En la Fase de Transición:

Se pueden realizar pruebas Beta (prueba realizada en organizaciones representativas “clientes beta”) y/o pruebas Alfa (se realizan en la empresa que desarrolla el software; pero fuera de la organización de desarrollo) y/o validaciones por terceros (una empresa especializada en pruebas

realiza pruebas de aceptación por encargo del cliente). Se recopilan y analizan los resultados de estas pruebas con el objetivo de llevar a cabo acciones. En esta fase se buscan pequeñas deficiencias que pasaron desapercibidas durante la fase de construcción y que pueden ser corregidas en el marco de la línea base de la arquitectura existente. La prueba está enfocada principalmente en la evaluación y determinación de la calidad del producto.[3]

Según RUP un software se va perfeccionando mediante las iteraciones que se van realizando durante su ciclo de vida en varios niveles o escenarios de trabajos, que permiten probar el producto desde la menor unidad creada hasta que se ha finalizado la construcción.

1.3 Estrategia de prueba de Software

Una estrategia de prueba de software íntegra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados. Es una parte fundamental del proceso de validación y verificación del software. Este proceso debe llevar implícito pruebas de bajo nivel, así como de pruebas de alto nivel por lo que es útil aplicar distintas técnicas de pruebas en diferentes momentos.[13]

En todo proyecto es necesario establecer una estrategia para probar el software. Si se efectúan las pruebas sin un plan se malgasta el tiempo y el esfuerzo es consumido innecesariamente y en el peor de los casos, los errores inadvertidos quedarán sin detectar. A medida que se descubren, los errores deben diagnosticarse y corregirse empleando un proceso llamado depuración.

La estrategia define:

- Técnicas de pruebas (manual o automática) y herramientas a ser usadas.
- Qué criterios de éxito y culminación de la prueba serán usados.
- Consideraciones especiales afectadas por requerimientos de recursos o que tengan implicaciones en la planificación.

Objetivos de la estrategia de prueba

- Planificar las pruebas necesarias en cada iteración, incluyendo las pruebas de unidad, integración y las pruebas de sistema. Las pruebas de unidad y de integración son necesarias dentro de la iteración, mientras que las pruebas de sistema son necesarias sólo al final de la iteración.

- Diseñar e implementar las pruebas creando los casos de prueba que especifican qué probar, cómo realizar las pruebas y creando, si es posible, componentes de prueba ejecutables para automatizar las pruebas.

- Realizar diferentes pruebas y manejar los resultados de cada prueba sistemáticamente. Los productos de desarrollo de software en los que se detectan defectos son probados de nuevo y posiblemente devueltas a otra etapa, como diseño o implementación, de forma que los defectos puedan ser arreglados.

Para conseguir estos objetivos el flujo de trabajo de la etapa de Pruebas consta de las siguientes etapas:

- Planificación de las pruebas: se define en el período que se ejecutarán las pruebas y quiénes serán los responsables de cada una.

- Diseño de las pruebas: se diseñan las pruebas definidas basándose en los diseños de caso de uso.

- Implementación de las pruebas: se generan los scripts para las pruebas automatizadas.

- Ejecución de las pruebas: se realizan las pruebas diseñadas o implementadas con anterioridad.

- Evaluación de las pruebas: se realiza un registro de los errores y los con los resultados obtenidos de las pruebas se verifican si las no conformidades anteriormente fueron solucionadas.

1.4 Práctica de prueba en la actualidad

En el mundo existen grandes compañías rectoras en todo el proceso de prueba e incluso se dedican a brindar servicios de consultorías a otras empresas que requieran velar por la calidad de sus productos, son en Colombia (CCC-Comunidad Colombiana de Pruebas y Calidad de Software, SQA-Software Qualiti Assurance), Bélgica (EvoTest), entre otras muchas. Cuba por su parte no tiene mucha experiencia en lo que desarrollo de software se refiere, aunque ya se avanza con paso firme una clara evidencia es la creación de los institutos politécnicos de informáticas con el fin de preparar jóvenes en dicha rama; por otro lado se han creado varias organizaciones para el desarrollo del software en el país como son: DESOFT, empresa encargada de ofrecer Soluciones Integrales en Tecnologías de la Información. Softel es una Empresa de Software Cubana cuyo objeto social principal es ofrecer soluciones informáticas para el Sistema de Salud. SoftCaribe es un equipo de profesionales, con un efectivo sistema de gestión empresarial, dichas organizaciones cuentan con un grupo de prueba el cual es el encargado de velar por la calidad de los productos.

Otro logro de la revolución cubana es el surgimiento de la UCI en el 2002, con el objetivo de formar y preparar ingenieros en ciencias informáticas para fomentar el desarrollo en el país, la misma cuenta con varios centros que se dedican a la creación de software para satisfacer las necesidades de diferentes organizaciones y el Centro de Calidad para Soluciones Informáticas (Calisoft) que se especializa en el aseguramiento de la calidad. En el resto de los centros laboran grupos de prueba con el objetivo de que el producto cuente con una calidad adecuada al llegar a Calisoft que se encarga de la realización de las pruebas de liberaciones finales, para ello los producto transita por varios departamento donde se le realizan procesos de pruebas algunas que se aplican son funcionales, rendimiento, seguridad, usabilidad, aceptación, entre otras. Luego de un análisis minucioso de los resultados si los mismos son satisfactorios, estarán listos para salir

al mercado bajo la licencia de nivel II de CMMI,[4] que es con la que cuenta el centro, esto le brinda mayor confiabilidad a las empresas productoras, principalmente a la UCI.

1.5 Sistemas de Gestión

Un sistema de gestión es una estructura probada para la gestión y mejora continua de los procedimientos y procesos de una organización, permitiendo aprovechar y desarrollar el potencial existente en la organización.

La implementación de un sistema de gestión eficaz puede ayudar a:

- Aumentar la satisfacción de clientes y partes interesadas.
- Reducir costos.
- Mejorar la efectividad operativa.
- Lograr mejoras continuas.[22]

La gestión de la información es el proceso de analizar y utilizar la información que se ha obtenido y registrado para permitir a los administradores (de todos los niveles) tomar decisiones documentadas.

La gestión de la información implica:

- Determinar la información que se precisa.
- Recoger y analizar la información.
- Registrar y recuperar la información cuando sea necesario.
- Utilizar la información.

[4] CMMI: Capability Maturity Model Integration(Modelo de Integración de Madurez y Capacidad)

Los procesos de gestión de la información se logran de una manera más rápida o eficaz cuando se hace uso de sistemas informáticos que faciliten estos procesos ya sea para una empresa de distintos ramas de la economía.

1.6 Sistema de Gestión en la Industria Petrolera. Producto MaximusDrillPro

En el Centro de Informática Industrial (CEDIN) se encuentra funcionado el proyecto Sistema Integral de Perforación de Pozos (SIPP) el cual desarrolla el producto llamado MaximusDrillPro para gestionar y supervisar la información consolidada por los expertos en los pozos durante el proceso de perforación. El proyecto surge debido a la necesidad de informatizar los procesos de negocio de le empresa de Cuba – Petróleo (Cupet), luego el centro le dio la tarea al equipo de trabajo para darle cumplimiento al mismo, el resultado de varias reuniones con los clientes es el levantamiento y especificación de requisitos. Luego de un análisis minucioso de los mismos se llega a la conclusión de crear un sistema distribuido, esto permite tener parte del sistema en el Pozo y la otra en la Dirección de Intervención y Perforación de Pozos (Dipp)[5] haciendo el sistema más ligero y seguro.

El subsistema pozo trabaja en su propio entorno local, utiliza la replicación de datos como solución para comunicar a los pozos con la Dipp, es multiplataforma y de fácil configuración, si bien es un software a medida del cliente, por sus características puede ser reestructurado para operar en negocios con tipos de necesidades específicas del cliente. El sistema permite visualizar la información en la aplicación a manera de reportes y partes de la perforación, permite hacer representaciones gráficas, así como exportar/importar archivos (txt o exel).

MaximusDrillPro se encuentra dividido en 3 subsistemas principales los que a su vez están compuestos por varios módulos. Subsistema Seguridad (Módulo Inicio, Módulo Administración, Módulo Administración de Pozo), Subsistema DIPP (Módulo Directivo, Módulo Secretaria, Módulo Técnico), Subsistema Pozo (Módulo Supervisor y Módulo Geólogo), además cuenta con los

[5] Empresa radicada en la provincia de Matanzas, dedicada a la administración de los procesos industriales en la perforación de pozos de petróleo.

Módulo de Gestión de Nomencladores de la perforación y un Módulo de Graficación de indicadores de perforación. Por la importancia de esta aplicación para el sector del petróleo, para gestionar información y tomar decisiones oportunas durante el proceso de perforación. Al producto MaximusDrillPro del proyecto Sistema Integral de perforación de Pozo (SIPP) se le aplica pruebas asociadas a la base de datos, seguridad, y otras para lograr un buen funcionamiento del sistema desarrollado.

1.7 Conclusiones Parciales

Se obtiene la base fundamental de la investigación a través de la definición y elementos importante del objeto de estudio. Se abordaron elementos que caracterizan a los procesos de pruebas de software, proporcionando los elementos teóricos necesarios para guiar el proceso de definición de una estrategia para el producto MaximusDrillPro en el próximo capítulo.

CAPÍTULO 2: DESCRIPCIÓN DE LA ESTRATEGIA PROPUESTA

Introducción

En dicho capítulo se desarrollara una estrategia de prueba para ser aplicada al producto MaximusDrillPro para evaluar la calidad del producto. Especificando la planificación, los artefactos y herramientas que serán utilizadas.

2.1 Herramienta en el proceso de pruebas

Con el avance de las tecnologías, el proceso de las pruebas de software también ha evolucionado por lo que se hace necesario la creación de herramientas para apoyar, agilizar y facilitar dicho proceso. El flujo de prueba específicamente es muy importante en el desarrollo de un software por lo que se hace imprescindible la investigación del uso de herramientas para mejorar la calidad y eficiencia del mismo.

Las pruebas de software cuentan con una serie de herramientas que automatizan y ayudan a un mejor resultado de las mismas existen herramientas tales como:

Para las pruebas de carga y stress:

JMeter, es una herramienta la cual permite probar la afluencia de usuarios y la velocidad de carga de la aplicación web, software libre y multiplataforma, realizada con el lenguaje de Java en su totalidad. Permitiendo conocer los tiempos de respuestas experimentadas por la aplicación al realizarse la prueba con un número de usuarios determinados y un número real de transferencias procesadas en una unidad de tiempo. Brinda la posibilidad de visualizar la respuesta del servidor en varios formatos como XML y HTML, generar gráficos con los resultados obtenidos. La interfaz gráfica da la posibilidad de realizar operaciones más rápido, además puede cargar y realizar pruebas sobre distintos tipos de servidores.[14]

VPerformer, es un producto de pruebas de rendimiento y de carga, que puede ser usado para evaluar el rendimiento y escalabilidad de las aplicaciones web. Revela el comportamiento de la aplicación web cuando está sometido a una tensión extrema. Posibilita medir las características de rendimiento de la aplicación mediante la generación de scripts de pruebas automatizadas y reproducir scripts de prueba que simulan miles de usuarios virtuales concurrentes. Permitiendo monitorear el estado de cualquier componente externo asociado que influya en el comportamiento de su aplicación, permitiéndole identificar los cuellos de botella. Entre sus principales beneficios se destacan: una interfaz fácil de usar, no requiere conocimientos de programación y aporta flexibilidad a las pruebas de distribución.[15]

Para las pruebas de Volumen:

Benchmark Factory es una herramienta de evaluación del rendimiento que le permite realizar reproducciones del volumen de trabajo de la base de datos, evaluaciones de referencias estándares de la industria y evaluaciones de escalabilidad. Le permite implementar cambios en el entorno de su base de datos con confianza, disminuyendo los riesgos asociados a parches, actualizaciones, migraciones y ajustes en las configuraciones de máquinas virtuales mediante las herramientas de evaluación de carga incorporadas. Benchmark Factory es una alternativa más flexible, sencilla y menos costosa a las evaluaciones de Oracle Real Application. Con este software de programación de reproducción del volumen de trabajo, puede eliminar el bajo rendimiento de las bases de datos SQL y simplificar considerablemente la gestión de bases de datos de alto rendimiento, así como realizar la evaluación de la escalabilidad.[16]

EMS Data Generator para PostgreSQL es una potente herramienta para la generación de datos de prueba a varias tablas de base de datos PostgreSQL de una vez. La aplicación asistente permite definir tablas, configurar valores de campos y muchas otras características para generar datos de prueba de una forma sencilla. Presenta una interfaz amigable y fácil de utilizar, maneja todos los tipos de datos de PostgreSQL incluyendo matriz, direcciones de red, y tipos geométricos.[17]

Para las pruebas de Replicación:

Slony-I. Es un software que permite hacer replicaciones maestro/esclavo asíncrono, realizando actualizaciones en cascada. Slony-I es un maestro para varios sistemas de replicación esclavos en cascada de apoyo (por ejemplo, un nodo puede alimentar a otro nodo que se alimenta de otro nodo) y de conmutación por error. El panorama para el desarrollo de Slony- I es un esclavo de replicación del sistema principal que incluye todas las características y las capacidades necesarias para replicar base de datos de gran tamaño a un número razonable limitado de los sistemas esclavistas. Entre sus principales desventajas se puede encontrar que su instalación, configuración y administración es compleja. Además no es viable para conexiones inestables o configuraciones variables.[18]

SymmetricDS, es una herramienta de replicación de datos asincrónica apoya varios suscriptores y sincronización bi-direccional, maneja tecnologías web y base de datos para replicar tablas entre bases de datos relacionales en tiempo casi real. Puede ser instalado como una aplicación web en un servidor de aplicaciones Java o puede ser instalado independientemente incrustado en otra aplicación Java. Permite que los cambios de entrada y de salida sean sincronizadas con otras bases de datos. Partiendo que la sincronización está configurada para extraer o enviar en cualquier dirección, el mismo nodo puede actuar a la vez como cliente o como servidor en diferentes circunstancias. Cuenta con una configuración sencilla para el usuario. Es una herramienta fácil de aprender y con más de una posibilidad para su despliegue, lo que permite usar la forma más adecuada según las características del hardware que se posea.[19]

Para las pruebas de Seguridad:

La herramienta Snort es un detector de intrusos basado en red. Es un software con licencia GPL, muy flexible, robusto y multiplataforma que ofrece capacidades de almacenamiento en archivos de texto como en bases de datos abiertas como lo es MySQL. Implementa un motor de detección de ataques y barrido de puertos que permite registrar, alertar y responder ante cualquier anomalía previamente definida, Snort utiliza la biblioteca estándar libcap y tcpdump como registro

de paquetes en el fondo. Dispone de una gran cantidad de filtros o patrones ya predefinidos, así como actualizaciones constantes ante casos de ataques, barridos o vulnerabilidades que vayan siendo detectadas a través de los distintos boletines de seguridad.[20]

PenTBox, es una mini suite de Seguridad orientada a pruebas de penetración, es multiplataforma, libre bajo la licencia (GNU/GPLv3) y programada en Ruby, destinada a testear la seguridad y estabilidad de las redes. Está compuesta por herramientas como: crackeadores de claves, herramientas para realizar denegación de servicio (DoS y DDoS), generadores de claves seguras, entra otras.[21]

2.2 Estrategia para aplicar al producto MaximusDrillPro

La siguiente estrategia de prueba se fundamenta en la ejecución de pruebas de Caja Negra al producto MaximusDrillPro. Para el desarrollo de la misma se analizan los procesos de pruebas como planificación, diseño, ejecución y evaluación de las pruebas, así como los artefactos que se deben generar y los roles que deben intervenir.

2.2.1 Objetivos de las pruebas

- Pruebas Funcionales: se aplican dichas pruebas para verificar que los requisitos planteados por los clientes de Cupet para el producto MaximusDrillPro estén cumplidos y que funcionen correctamente.

- Pruebas de Carga: se aplican dichas pruebas para analizar la respuesta del producto MaximusDrillPro ante el grado de concurrencia de usuarios en el mismo instante de tiempo.

- Pruebas de Stress: se aplican para observar el punto clave donde el producto MaximusDrillPro colapsaría en un momento de carga extrema, ante varias peticiones de usuarios y las respuestas emitidas por el mismo.

- Pruebas de Volumen: se aplican dichas pruebas para determinar el tiempo de respuesta de la base de datos del producto MaximusDrillPro en el momento de realizar una tarea en condiciones particulares de trabajo.

- Pruebas de Replicación: se aplican dichas pruebas para verificar la comunicación entre las bases de datos del producto MaximusDrillPro (de los Dipp y los Pozo) desplegados en el campo con comunicación en tiempo real.

- Pruebas de Seguridad: se aplican al producto MaximusDrillPro para detectar los puntos más vulnerables que puedan tener.

2.2.2 Alcance de la estrategia

En el alcance de la estrategia se define como se aplicaran las pruebas, hasta qué momento seguir probando y cuando finalizar una iteración o ciclo de prueba.

En el proceso de prueba se le aplicará como máximo tres iteraciones en caso de que persistan las No Conformidades (NC), significativa el proceso de prueba será abandonado, hasta que el equipo de desarrollo logre solucionarlas, reiniciando el mismo. En la segunda o tercera iteración se comenzará chequeando las NC detectadas con antelación. Finalmente el producto será liberado por el grupo de prueba y estará en condiciones de pasar a Calisoft, donde le darán la liberación final para el caso de las pruebas funcionales.

Las pruebas de carga y stress se realizarán iteraciones redoblando en cada una el número de usuarios tratando de lograr colapsar el sistema y de esta manera identificar el máximo soportado por el mismo. Así mismo las pruebas de volumen se definirán los rangos de rendimiento para diferentes volúmenes que se incrementarán en cada iteración mientras la base de datos lo soporte.

Para la prueba de replicación se realizará de una a dos iteraciones para verificar la comunicación entre las bases de datos. En el caso de las pruebas de seguridad se realizarán dos iteraciones

una para detectar las vulnerabilidades, la otra para verificar que se hayan solucionado las mismas.

2.2.3 Tipos de pruebas. Herramientas

Existen varios tipos de pruebas en este caso se aplican pruebas funcionales las cuales se realizan de forma manual. Las pruebas de rendimiento por su parte se utilizan herramientas como se describe a continuación:

Para pruebas las de carga y stress se realizarán con la utilización la herramienta JMeter ya que permite probar la afluencia de usuarios y la velocidad de carga de la aplicación web. La interfaz gráfica da la posibilidad de realizar operaciones más rápido.

Las pruebas de volumen se aplicarán con la utilización de la herramienta EMS Data Generator para PostgreSQL 2005 que es una potente herramienta para la generación de datos de prueba a varias tablas de base de datos PostgreSQL de una vez, también maneja todos los tipos de datos de PostgreSQL incluyendo matriz, direcciones de red, y tipos geométricos.

Las pruebas de replicación utilizará la herramienta SymmetricDS la cual maneja tecnologías web y base de datos para replicar tablas entre bases de datos relacionales en tiempo casi real. Permite que los cambios de entrada y de salida sean sincronizadas con otras bases de datos. Cuenta con la posibilidad de aumentar su despliegue.

Para aplicar las pruebas de seguridad se utilizará la herramienta PenTBox que está destinada a testear la seguridad y estabilidad de las redes.

2.2.4 Método de prueba a utilizar

Solo se utiliza la técnica de caja negra ya que no se entrara en el detalle del código, centrándose en las pruebas funcionales y en otras que pretenden demostrar que el software es operativo, que las entradas se aceptan y las salidas se producen de forma correcta y que la integridad de la información externa se mantiene.

2.2.5 Roles y Responsabilidades

El rol no es más que el papel que juega un actor en una actividad, encargado de cumplir con el deber asignado.

- **Diseñador de pruebas**: el diseñador de pruebas es el encargado de realizar los casos de pruebas basándose en los casos de uso del sistema.
- **Probador**: Es el encargado de realizar las pruebas diseñadas anteriormente, además de otras que se deseen aplicar que no requieran de diseños.

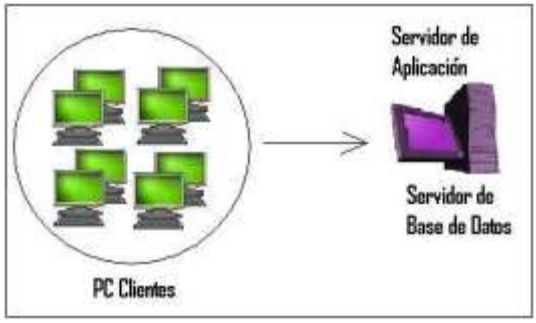


Figura 2 Escenario de pruebas

Recursos del sistema

El sistema debe de contar con un servidor de base de datos y aplicación, Postgres versión 8.4 como gestor de base de datos, un servidor web Apache 1.7.3, para su visualización es necesario la utilización de un navegador web FirexFox versión 3.5.5.

1 PC que funcionara como Cliente

Artefactos de apoyo a las pruebas

Artefactos a probar	Documentación necesaria
Aplicación Web	Casos de Prueba
	Modelos de Caso de Uso del sistema.

Tabla 2 Artefactos de apoyo a las pruebas

2.2.6 Cronograma de trabajo

No.	Tarea	Duración	Fecha Inicio	Fecha Fin
1.	Seleccionar las técnicas de pruebas	6 días	23/03/2012	30/03/2012
2.	Diseño de prueba	25 días	02/04/2012	04/05/2012
3.	Pruebas Funcionales	13 días	02/04/2012	18/04/2012
4.	No Funcionales	12 días	19/04/2012	04/05/2012
5.	Carga y Stress	4 días	19/04/2012	24/04/2012
6.	Seguridad	4 días	25/04/2012	30/04/2012
7.	Bases de Datos	4 días	01/05/2012	04/05/2012
8.	Ejecución de las pruebas	11 días	07/05/2012	21/05/2012
9.	Análisis de los resultados	4 días	22/05/2012	25/05/2012

Tabla 3 Cronograma de trabajo

2.2.7 Diseño de las pruebas

Para las pruebas funcionales se diseñan casos de pruebas por subsistemas para facilitar el trabajo de la siguiente manera:

El Subsistema Seguridad a su vez está compuesto por los módulos Inicio, Administración y Administración de pozo:

- El módulo de Inicio tiene 3 diseños de casos de pruebas.
- El módulo de Administración y Administración de Pozo tiene 4 diseños de casos de pruebas comunes.

El Subsistemas DIPP a su vez está compuesto por los módulos Directivo, Secretaria, Técnico:

- El módulo Directivo tiene 6 diseños de casos de prueba.
- El módulo Secretaria tiene 5 diseños de casos de prueba.
- El módulo Técnico tiene 11 diseños de casos de pruebas.

El Subsistemas Pozo está compuesto por los módulos Supervisor, Geólogo:

- El módulo Supervisor tiene 18 diseños de casos de pruebas.
- El módulo Geólogo tiene 12 diseños de casos de pruebas.

El módulo Gestión de Nomencladores tiene 12 diseños de casos de prueba.

El módulo Graficación tiene 1 diseño de caso de prueba.

El módulo Manejo de Datos tiene 1 diseño de caso de prueba.

Para las pruebas de carga y stress se simulan una afluencia de 25 usuarios por módulos.

Las pruebas de Volumen se aplican según la siguiente distribución primero las tablas que en las Tablas de Seguridad, Tablas de Nomencladores y las Tablas del proceso de Perforación, se realizará 5 iteraciones.

Durante las pruebas de Replicación se aplican en la misma distribución que las de Volumen. Se insertan y eliminan datos en algunas de las tablas.

El resto de las pruebas que se realizan no necesitan de diseños de caso de pruebas ya que tan solo serán entrar juegos de datos nada más.

2.3 Conclusiones parciales

Al finalizar esta parte de la investigación se tiene lista la estrategia de prueba para ser aplicada al producto MaximusDrillPro. De esta forma se establece una guía que garantiza la organización del trabajo, ganando en tiempo y asegurando que el producto final tendrá un número reducido de defectos, contribuyendo al aseguramiento de la calidad del mismo.

CAPÍTULO 3: ANÁLISIS DE RESULTADOS

Introducción

En el actual capítulo se describen los resultados obtenidos durante la ejecución de las pruebas definidas en la estrategia anteriormente descripta. Se caracterizan las no conformidades encontradas y se obtiene la respuesta de equipo de desarrollo.

3.1 Ejecución de las pruebas

3.1.1 Resultado de pruebas Funcionales.

Durante el proceso de las pruebas funcionales el Subsistema de Seguridad en general tuvo 10 No Conformidades (NC) desglosadas de la siguiente manera por módulos:

- El módulo de Inicio presento 3 NC las cuales se solucionaron satisfactoriamente quedando de esta forma listo para pasar a la próxima etapa.
- El módulo de Administración y Administración de Pozo se le detecto 7 NC las cuales se solucionaron satisfactoriamente quedando liberado por el grupo de prueba del CEDIN.

El Subsistemas DIPP en resumen conto 45 NC desglosadas por módulos de la siguiente manera:

- El módulo Directivo tuvo 11 NC de ellas se solucionaron satisfactoriamente 9, finalmente concluyo el período de prueba.
- El módulo Secretaria tuvo 15 NC de ellas se solucionaron satisfactoriamente 14 finalmente concluyo dicho período de prueba.
- El módulo Técnico tuvo 19 NC las cuales se solucionaron satisfactoriamente concluyendo dicho período de prueba.

Al aplicarle las pruebas al Subsistemas Pozo obtuvieron 8 NC en general y por módulos quedando de la siguiente manera:

- El módulo Supervisor contó con 3 NC las cuales se solucionaron.
- El módulo Geólogo arrojo 5 NC de ellas se solucionaron 4 que fueron las más significativas para poder liberar el producto por el grupo de prueba.

El proceso de dichas pruebas arrojó un total de 56 NC para el módulo Gestión de Nomencladores de ellas se solucionaron 50 las restantes no son significativas, de esta manera se termina el ciclo de prueba.

El módulo Graficación presentó 2 NC las cuales fueron solucionadas con éxito.

El módulo Manejo de Datos resalto 6 NC de ellas se solucionaron todas para terminar de esta forma las pruebas.

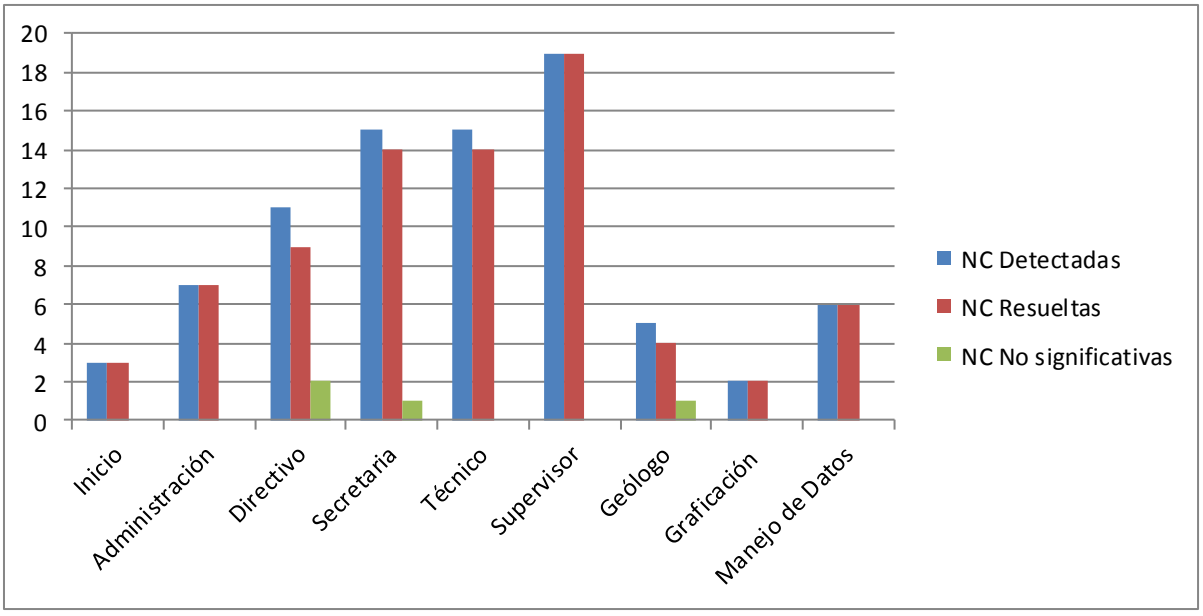
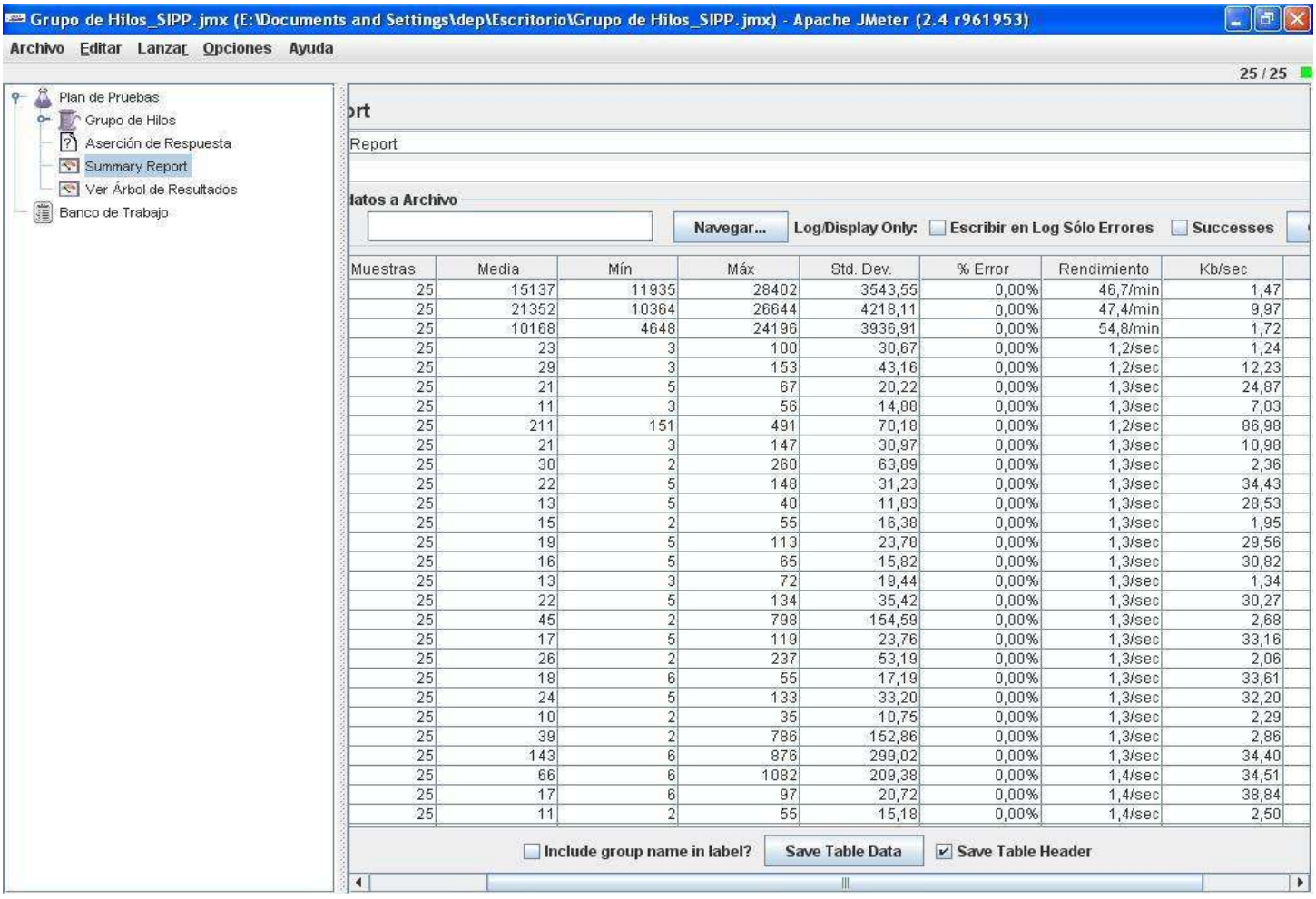


Figura 3 Gráfica con los resultados de las Pruebas Funcionales

Las pruebas funcionales terminaron su proceso con éxito, ya que las NC que quedaron pendiente no son significativas para el sistema, sino de documentación, por lo que el grupo de pruebas las notifica al equipo de analista para que de solución y le da paso a la liberación por parte del departamento quedando apto para pasar a Calisoft.

3.1.2 Resultado de pruebas Carga y Stress

Al ejecutar las pruebas de Cargas y Stress se tuvo en cuenta los requisitos no funcionales de la BD comenzando por 25 usuarios en un período de 1 hora. Los resultados arrojados son:



Muestras	Media	Mín	Máx	Std. Dev.	% Error	Rendimiento	Kb/sec
25	15137	11935	28402	3543,55	0,00%	46,7/min	1,47
25	21352	10364	26644	4218,11	0,00%	47,4/min	9,97
25	10168	4648	24196	3936,91	0,00%	54,8/min	1,72
25	23	3	100	30,67	0,00%	1,2/sec	1,24
25	29	3	153	43,16	0,00%	1,2/sec	12,23
25	21	5	67	20,22	0,00%	1,3/sec	24,87
25	11	3	56	14,88	0,00%	1,3/sec	7,03
25	211	151	491	70,18	0,00%	1,2/sec	86,98
25	21	3	147	30,97	0,00%	1,3/sec	10,98
25	30	2	260	63,89	0,00%	1,3/sec	2,36
25	22	5	148	31,23	0,00%	1,3/sec	34,43
25	13	5	40	11,83	0,00%	1,3/sec	28,53
25	15	2	55	16,38	0,00%	1,3/sec	1,95
25	19	5	113	23,78	0,00%	1,3/sec	29,56
25	16	5	65	15,82	0,00%	1,3/sec	30,82
25	13	3	72	19,44	0,00%	1,3/sec	1,34
25	22	5	134	35,42	0,00%	1,3/sec	30,27
25	45	2	798	154,59	0,00%	1,3/sec	2,68
25	17	5	119	23,76	0,00%	1,3/sec	33,16
25	26	2	237	53,19	0,00%	1,3/sec	2,06
25	18	6	55	17,19	0,00%	1,3/sec	33,61
25	24	5	133	33,20	0,00%	1,3/sec	32,20
25	10	2	35	10,75	0,00%	1,3/sec	2,29
25	39	2	786	152,86	0,00%	1,3/sec	2,86
25	143	6	876	299,02	0,00%	1,3/sec	34,40
25	66	6	1082	209,38	0,00%	1,4/sec	34,51
25	17	6	97	20,72	0,00%	1,4/sec	38,84
25	11	2	55	15,18	0,00%	1,4/sec	2,50

Figura 4 Prueba de Carga y Stress

El “Summary Report” (Informe Resumen) crea una fila por cada petición en la prueba. Por cada una de estas filas se muestra la siguiente información:

Label: El nombre de la muestra (conjunto de muestras).

Muestras: El número de muestras para cada URL.

Media: El tiempo medio transcurrido para un conjunto de resultados.

Mín: El mínimo tiempo transcurrido para las muestras de la URL dada.

Máx: El máximo tiempo transcurrido para las muestras de la URL dada.

Error %: Porcentaje de las peticiones con errores.

Rendimiento: Rendimiento medido en base a peticiones por segundo/minuto/hora.

Kb/sec: Rendimiento medido en Kilobytes por segundo.

Avg. Bytes: Tamaño medio de la respuesta de la muestra medido en bytes.

Analizados los resultados de las pruebas se observa se realizaron con éxito. Pues la columna respectiva al número de errores tiene valor 0%. El rendimiento se muestra que para casa simulación el servidor es capas de aceptar una media de 1,4 peticiones por segundo. Por todo lo ante planteado el proyecto es aceptable para la conexión.

3.1.3 Resultado de pruebas Volumen

Al ejecutar las pruebas de Volumen se puede decir que es una BD robusta y bien diseñada ya que los resultados de todas las pruebas fueron satisfactorios, para cada una:

Tablas Seguridad

Volumen	Tiempo Estimado	Tiempo Real	Cantidad de Datos Generados	Errores
100		0:00:02 min	500	0
1 000	0:00:30 min	0:00:12 min	5 000	0
10 000	0:03:30 min	0:01:43 min	50 000	0
25 000	0:08:30 min	0:04:00 min	125 000	0
50 000	0:16:59 min	0:09:41 min	250 000	0

Tabla 4 Pruebas Volumen. Tablas Seguridad

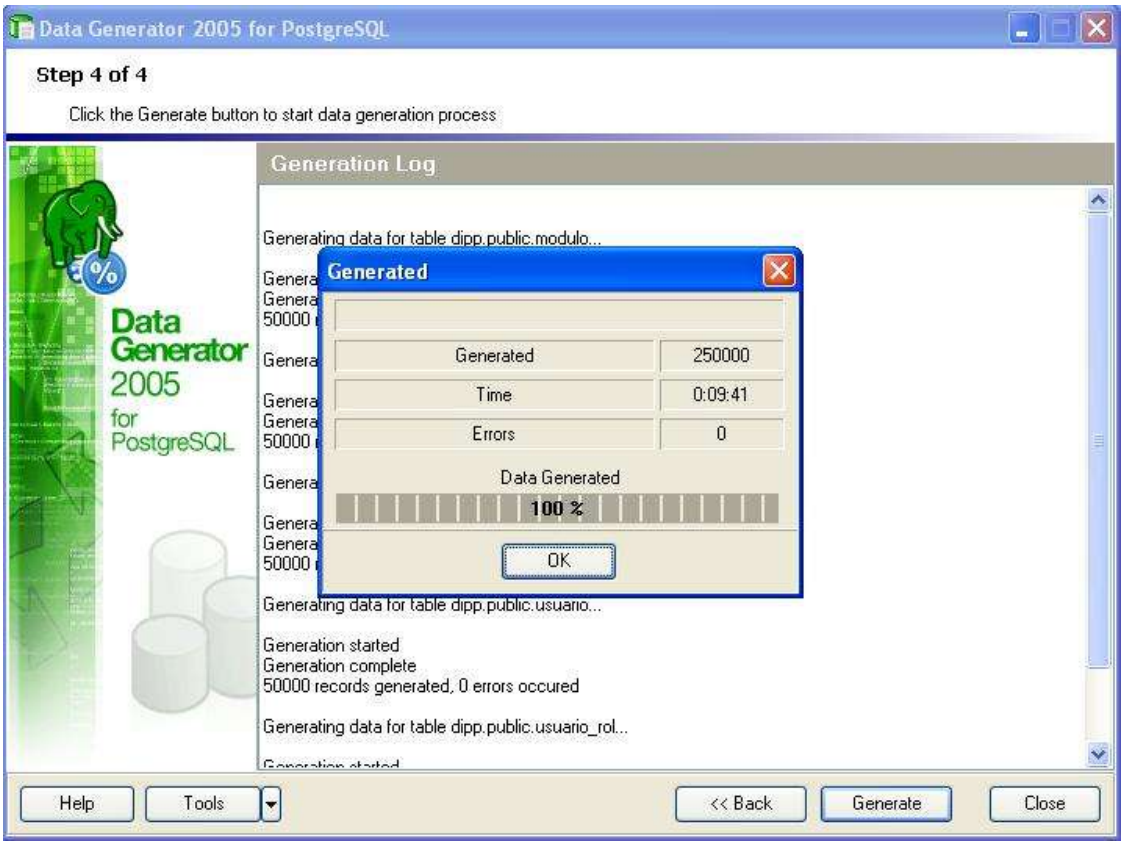


Figura 5 Prueba de Volumen Tablas Seguridad

Tablas Nomencladoras

Volumen	Tiempo Estimado	Tiempo Real	Cantidad de Datos Generados	Errores
100		0:00:06 min	3 100	0
1 000	0:00:01 min	0:00:51min	31 000	0
10 000	0:00:10 min	0:08:38 min	310 000	0
25 000	0:00:25 min	0:19:11 min	775 000	0
50 000	0:00:50 min	0:46:01 min	1550 000	0

Tabla 5 Pruebas Volumen. Tablas Nomencladoras

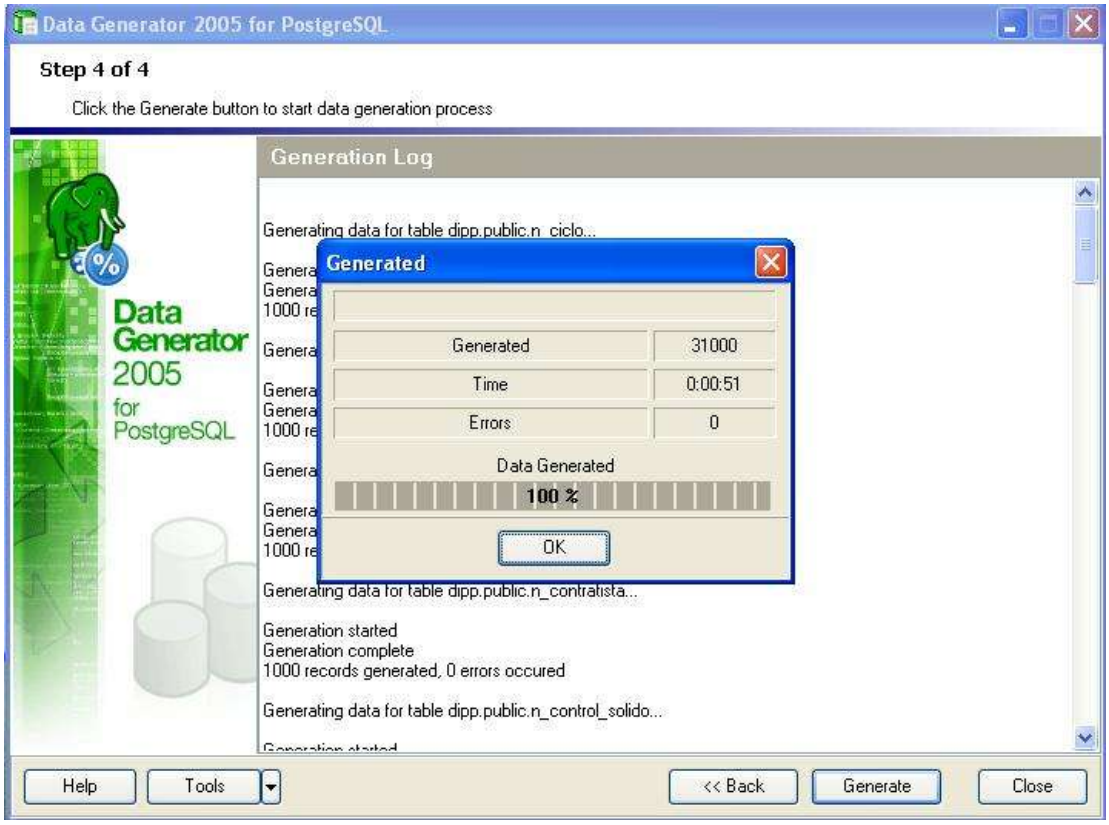


Figura 6 Prueba de Volumen Tablas Nomencladores

Tablas Perforación

Volumen	Tiempo Estimado	Tiempo Real	Cantidad de Datos Generados	Errores
100		0:00:20 min	6 300	0
1 000	0:03:04 min	0:02:36 min	63 000	0
10 000	0:33:04 min	0:27:19 min	630 000	0
25 000	1:23:03 min	1:03:31 min	1 575 000	0
50 000	2:46:06 min	2:01:33 min	3 150 000	0

Tabla 6 Pruebas Volumen. Tablas Perforación

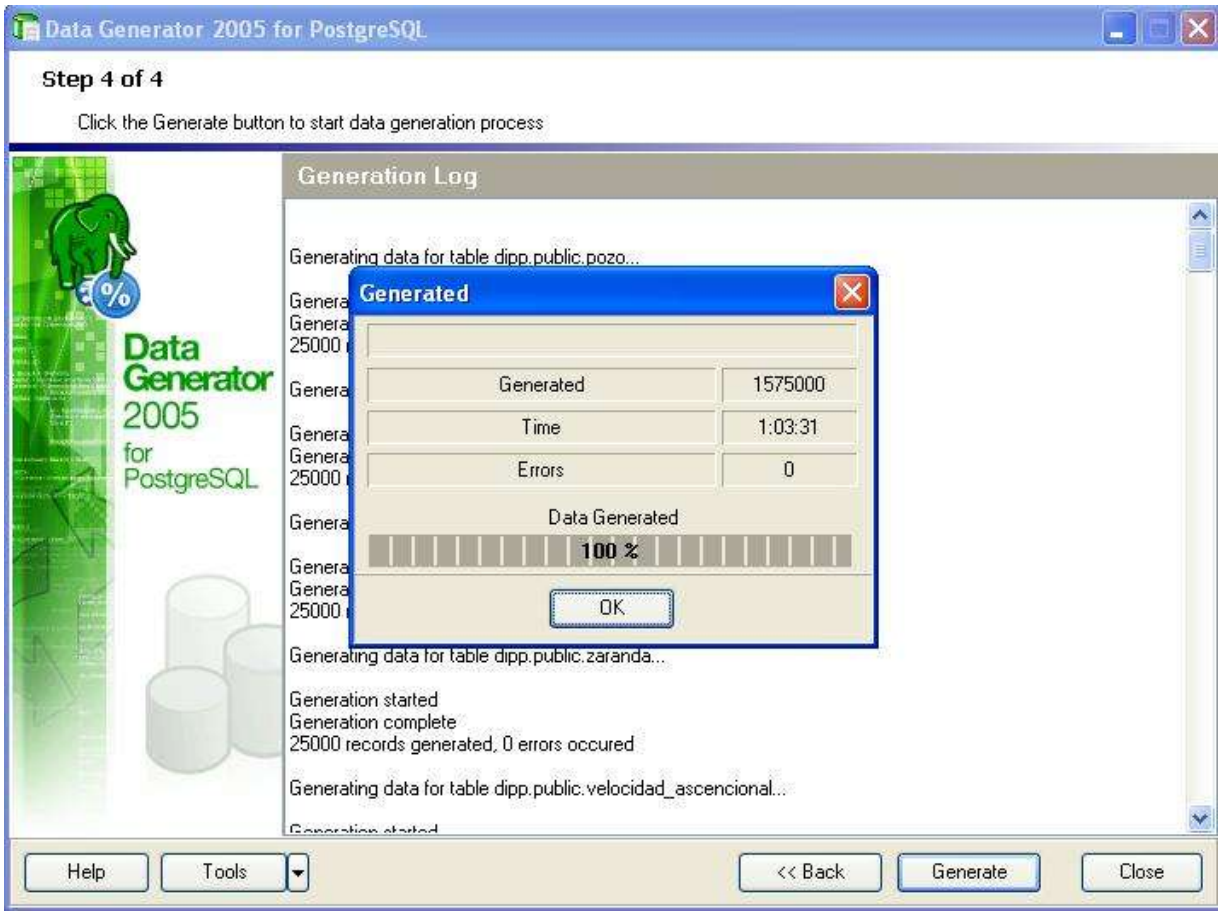


Figura 7 Prueba de Volumen Tablas Perforación

El tiempo esperado se determinar por regla de 3, Ejemplo: se toma como muestra el tiempo de la primera iteración 6 seg para 100 datos. ¿Qué tiempo demoraría para el volumen deseado?

100→6

1000→X

1000*6/100= 60 seg =1 min.

Luego de esto se define una escalabilidad de eficiencia en caso de que el tiempo real sea mayor que el esperado no sería Eficiente, de ser iguales estaría Normal y de ser menor Eficiente. Como se puede ver el tiempo real siempre permaneció por debajo del esperado.

3.1.4 Resultado de pruebas Replicación

Al ejecutar las pruebas de Replicación se tuvo como resultado una buena, rápida y constante comunicación entre las bases de datos del sistema es decir la Dipp y el Pozo, a esta conclusión se llega al analizar los resultados de las pruebas ejemplo las pruebas, se debe decir que la comunicación desde la Dipp hacia el Pozo es mejor que del Pozo hacia la Dipp.

Replicación de la BD Dipp hacia la BD Pozo

Tablas Nomencladora	Puerto 8080 Tiempo	Puerto 9090 Tiempo	Demora
Insertar	11:39:59	12:01:12	00:01:06
Eliminar	11:19:56	11:14:12	00:05:44
Tablas Perforación	Puerto 8080 Tiempo	Puerto 9090 Tiempo	Demora
Insertar	13:39:59	13:41:12	00:02:13
Eliminar	12:19:59	12:41:12	00:22:13
Tablas Seguridad	Puerto 8080 Tiempo	Puerto 9090 Tiempo	Demora
Insertar	17:42:30	17:43:20	00:01:10
Eliminar	16:09:59	16:11:12	00:02:13

Tabla 7 Pruebas Replicación. Dipp Pozo

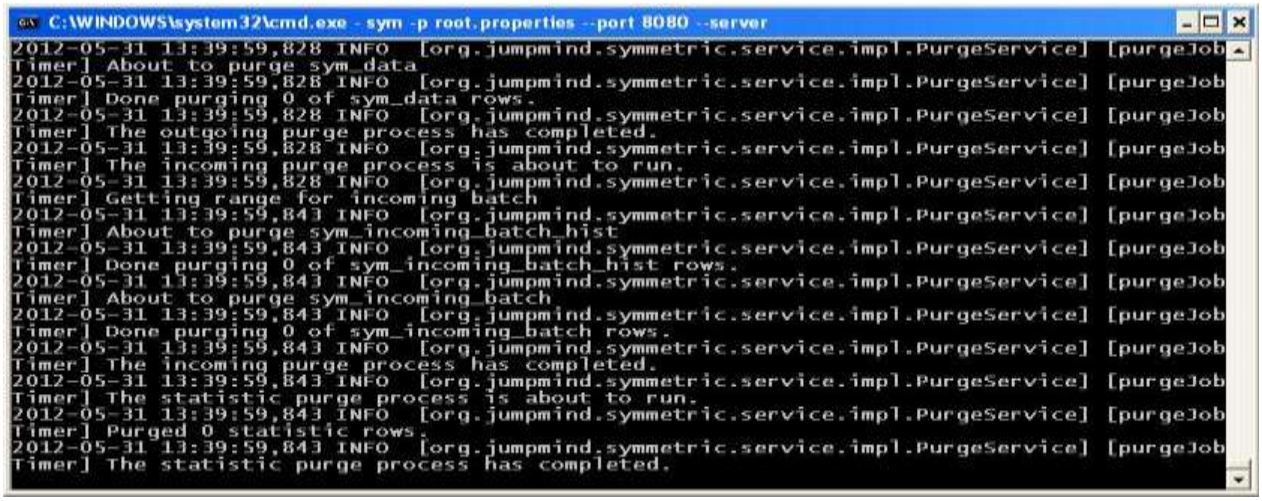


Figura 8 Prueba Replicación Dipp Pozo (8080)

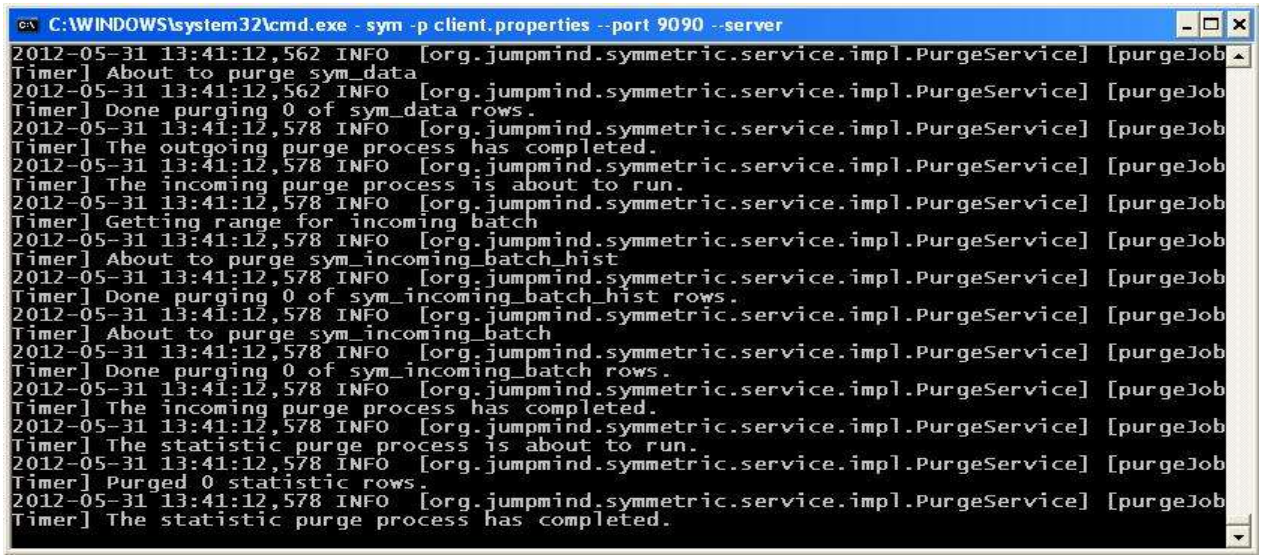


Figura 9 Prueba Replicación Dipp Pozo (9090)

Replicación de la BD Pozo hacia la BD Dipp

Tablas Nomencladora	Puerto 8080 Tiempo	Puerto 9090 Tiempo	Demora
Insertar	10:19:59	10:31:12	00:12:43
Eliminar	10:09:59	10:11:12	00:02:43
Tablas Perforación	Puerto 8080 Tiempo	Puerto 9090 Tiempo	Demora
Insertar	12:09:59	12:11:12	00:02:43

Eliminar	10:39:59	10:41:12	00:02:43

Tabla 8 Pruebas Replicación. Pozo Dipp

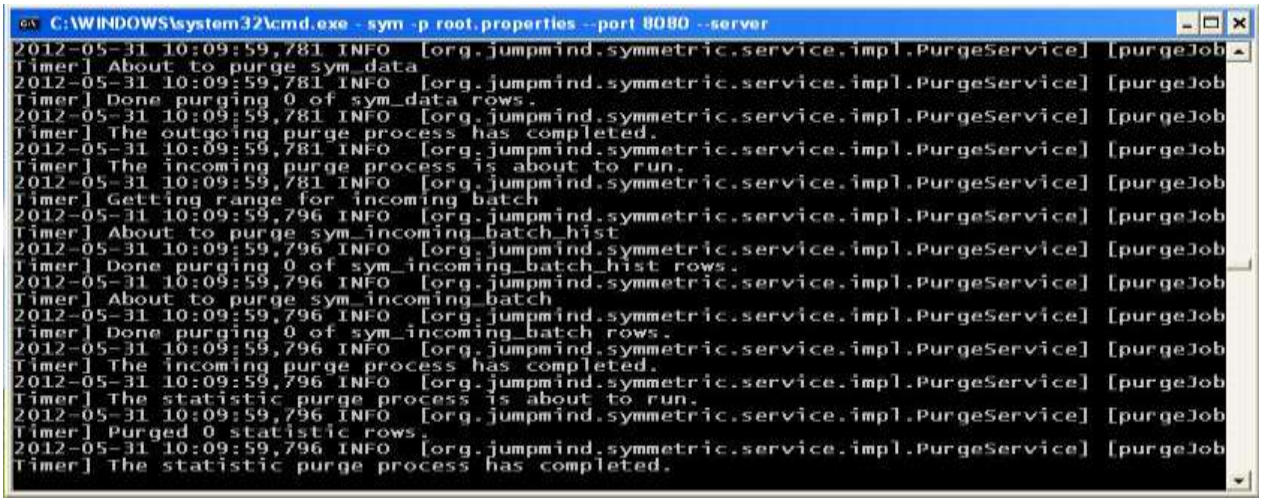


Figura 10 Prueba Replicación Pozo Dipp (8080)

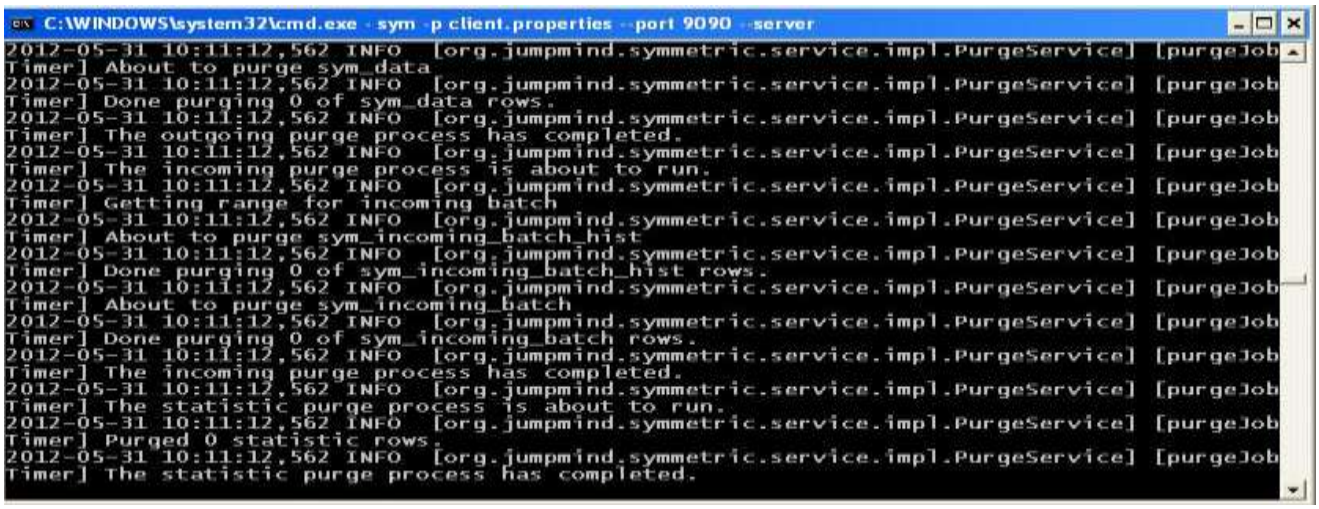


Figura 11Prueba Replicación Pozo Dipp (9090)

3.1.5 Resultado de pruebas Seguridad

Al ejecutar las pruebas de Seguridad no se encontraron vulnerabilidades significativas en el sistema.

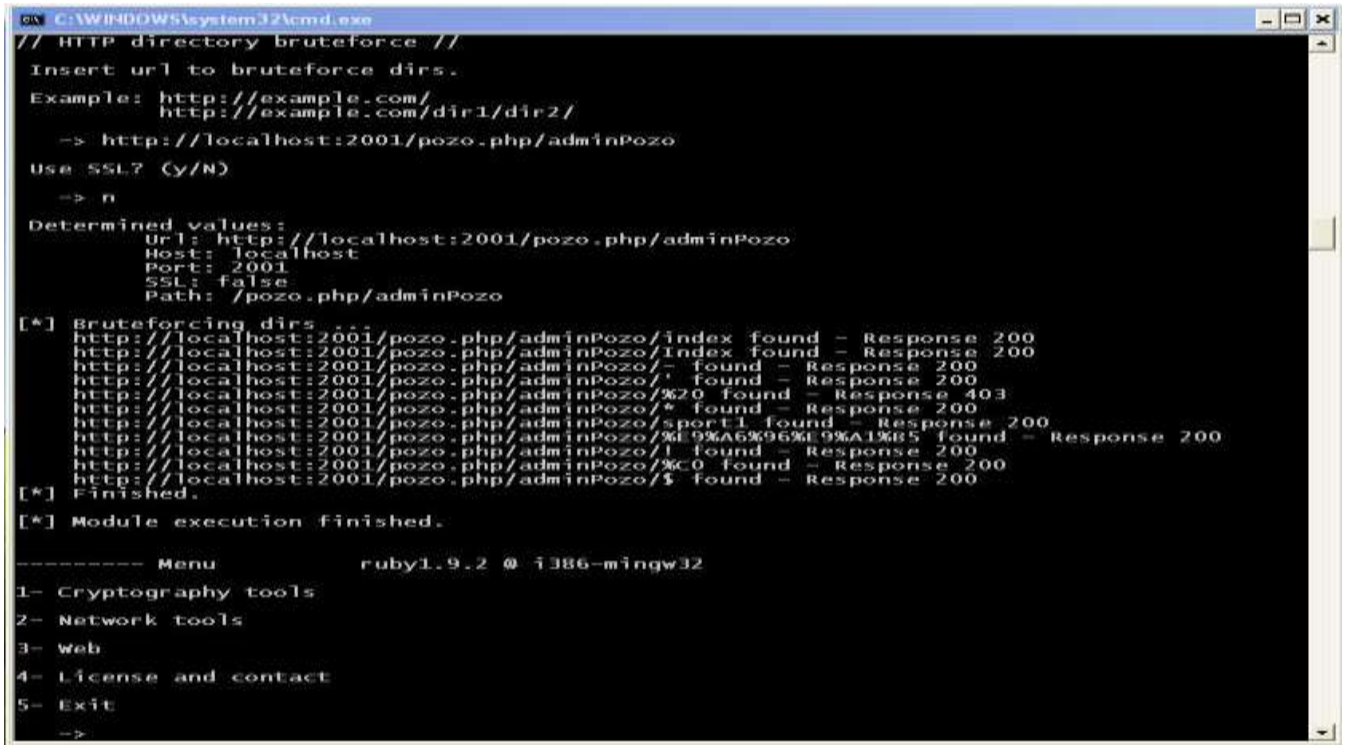


Figura 12 Seguridad Pozo

Vele destacar que las pruebas de rendimiento varían según las características y procesos que estén presentes durante el desarrollo de las mismas.

3.1.6 Conclusiones parciales

Con la aplicación de la estrategia de prueba se logro medir la calidad y rendimiento del producto MaximusDrillPro, mejorando aspectos con iteraciones antecedentes como son la seguridad, tiempo de respuesta del producto, así como la funcionalidad del mismo.

CONCLUSIONES

Como resultado de la investigación desarrollada, se puede arribar a las siguientes conclusiones:

- ✓ Se esclarecieron conceptos que estaban erróneos.
- ✓ Se logro definir una estrategia de prueba.
- ✓ Con la aplicación de la estrategia se organizó la forma de trabajo del equipo de prueba.
- ✓ Se logró mejorar el producto en diferentes aspectos como la seguridad y el rendimiento del producto al aplicar las pruebas definidas en la estrategia.
- ✓ Los resultados obtenidos en las pruebas sirvió para elevar la calidad del producto MaximusDrillPro.

RECOMENDACIONES

Se sugiere al grupo de prueba que en posteriores ciclos de prueba del producto MaximusDrillPro:

- ✓ Aplicar pruebas de Caja Blanca ya que en la investigación no alcanzo el tiempo para emplearlas.
- ✓ Aplicar pruebas de Integración.
- ✓ Agregar otras pruebas que se pueden realizar como las pruebas de Aceptación y de Confiabilidad.

REFERENCIAS BIBLIOGRÁFICAS

1. RAE, *Diccionario de la Real Academia de la Lengua Española.* 1999.
2. ISO_8402, *Gestión de la Calidad y Garantía.* 1994.
3. Pressman, R.S., *Ingeniería de Software. Un enfoque práctico.* 2005.
4. Humprhey, W.S., *Introducción al Proceso de Software Personal.* 2005.
5. IEEE, *Standard Glossary of Software Engineering Terminology.* 1990.
6. Davis, A., *Principles of Software Development.* 1995.
7. Garcerant, I., *Niveles de Pruebas.* 2008.
8. *L/T Ingenieria de Software. Un enfoque practico. Capitulo 17 (Técnicas de Software) y Capitulo 18 (Estrategias de Pruebas de Software)*
9. Jiménez, D., *Modelos de Pruebas de Software.* 2004.
10. María Garzón Villar, M.S.d.l.T., Esteban Leyva Cortés, Ignacio Prieto Tinoco, *Informática. Temario A. Volumen II. Profesores de Educación Secundaria.* 2007.
11. Karenny, A.B., *Metodologías tradicionales y metodologías ágiles.* 2009.
12. Galves, J., *Fundamento de la Metodología RUP RATIONAL UNIFIED PROCESS.* 2010.
13. EcuRed, *Estrategia de prueba de software.*
14. Jakarta, C., *Apache JMeter.* 2011.
15. vPerformer, 2011.
16. *Herramientas de escalabilidad, prueba de rendimiento y de carga de bases de datos.* 2012.
17. WIN, P.E.D.G.f.P., *EMS Data Generator 2005 for PostgreSQL.* 2012.
18. Salvador, U.d., *Replicación modelado y minería de datos .Replicación en PostgreSQL Slony-I.* 2011.
19. Henson, E.L.-C., *SymmetricDS User guide.* 2008.

20. *Snort Office*. 2012.

21. PentBox, *Herramienta de Pentesting*. 2012.

22. *Que son los sistemas de gestión.*

ANEXOS

Anexo 1

Los anexos serán entregados.

GLOSARIO DE TÉRMINOS

AVL – Análisis de Valores Límite

CEDIN - Centro de Informática Industrial.

Cupet - Empresa Cuba-Petróleo.

Calisoft - Centro de Calidad para Soluciones Informáticas.

Dipp - Dirección de Intervención y Perforación de Pozos.

ISO - Organización Internacional para la Estandarización (International Organization for Standardization).

NC - No Conformidades.

RAE – La Real Academia de la Lengua Española.

SIPP - Sistema Integral de Perforación de Pozo.