Universidad de las Ciencias Informáticas

Facultad # 6

Título: *"Propuesta de un procedimiento para la evaluación del rendimiento de las bases de datos basadas en PostgreSQL para software de gestión".*

Trabajo de Diploma para optar por el título de Ingeniero Informático

Autor(es): *Mariela Valdés Fernández*

Tutor(es): *Ing. Daimi Bretones Lorenzo*

Ing. José Rolando Lafaurie Olivares

Co-tutor: *MSC. Michaell González Jorrín*

Junio, 2010

DECLARACIÓN DE AUTORÍA

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

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

_____
Mariela Valdés Fernández
Firma del Autor

_____
Ing. Daimi Bretones Lorenzo
Firma del Tutor

_____
Ing. José Rolando Lafaurie Olivares
Firma del Tutor

DATOS DE CONTACTO

Ing. Daimí Bretones Lorenzo.

E-mail: dbretones@uci.cu, Edif. 52104, Teléf. 837-2166.

Ing. José Rolando Lafaurie Olivares.

E-mail: jrlafaurie@uci.cu , Edif. 77206, Teléf. 837-2261.

AGRADECIMIENTOS

Después de la realización de este trabajo de diploma quisiera agradecer a todo aquel que tuvo que ver de una forma directa e indirecta con la realización del mismo. Le quiero agradecer de todo corazón a mi tutora Daimí que siempre estuvo ahí cada vez que me hacía falta, cada vez que le mandaba el documento me lo enviaba en menos de 24 horas con todos los errores y siempre me dio el apoyo que me hizo falta. De mi tutor José Rolando a pesar de que no tuvimos mucha comunicación me ayudó mucho cada vez que me revisaba el documento la parte de la redacción. En DATEC tengo mucho que agradecerle a: Yasmany el novio de mi tutora, Michael, Juan Carlos, Lobo, Fran, Marien, Maikel Zúñiga, entre otros. También tengo que agradecerle a Delvis Echeverría que trabaja en Calisoft que me brindó mucha información de cómo hacer las pruebas de carga, stress y volumen. No puedo dejar de agradecerle al profesor Orlando Martínez que gracias a él es que he llegado a discutir esta tesis. Tampoco puedo dejar de agradecerle a todos mis compañeros de estudio y del aula que de una forma u otra me ayudaron y me dieron fuerzas para seguir adelante. Y por último quiero agradecerle a toda mi familia y especialmente a mi papá, a mi hermano, a mi novio, a la chinita y a mis tíos que están aquí presentes y a mi madre y a mis abuelos a pesar de que ya no estén entre nosotros que fueron los que me impulsaron cada día desde que entre en la escuela y me seguirán impulsando hasta que alcance todas mis metas trazadas, y voy aclarar algo muy importante que cité a mi familia de última y no por eso deja de ser la más importante porque como dice Martí: "La familia es la primera escuela del hombre".

DEDICATORIA

A mi padre por siempre estar a mi lado y a mi mamá que siempre está conmigo.

RESUMEN

En la Universidad de Ciencias Informáticas después de una indagación con los especialistas en el tema de bases de datos que laboran directamente en la producción del software se descubrió que no existe experiencia en la realización de pruebas a las bases de datos en función del rendimiento. Este atributo de calidad no se tiene en cuenta en la etapa inicial de los proyectos. Es por ello que surge la necesidad de evaluar el rendimiento de las bases de datos para realizar un diseño adecuado de las mismas.

En este trabajo de diploma se propone un procedimiento que permita la evaluación del rendimiento de las bases de datos basadas en PostgreSQL para software de gestión. Con la aplicación del procedimiento se espera que el producto de software proporcione apropiados tiempos de respuestas y procesamiento haciendo un uso adecuado de los recursos de hardware y software.

PALABRAS CLAVE

Rendimiento, Evaluación, Bases de Datos, Sistemas Gestores de Bases de Datos, PostgreSQL, Software de Gestión.

TABLA DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La competitividad en el terreno de la producción de software, está fuertemente marcada por la calidad que tenga intrínseca el software elaborado. La calidad de software es definida según Pressman como: "Concordancia del software producido 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". (1)

Un requisito es una capacidad o condición que debe cumplir un sistema de software. En ocasiones se centra la comprobación del producto al estado de operación de los requisitos funcionales y se obvia la comprobación de los requisitos de rendimiento.

La mayoría de los sistemas informáticos que se desarrollan actualmente necesitan hacer persistir información para lo cual se utilizan bases de datos para su almacenamiento y manipulación y un atributo de calidad a tener en cuenta es precisamente el rendimiento de estas bases de datos, pues determinan en gran medida la rapidez operacional, la escalabilidad y la disponibilidad.

En una breve indagación hecha a los especialistas en el tema de Bases de Datos que laboran en el Centro de Tecnologías de Gestión de Datos (DATEC) directamente en el campo de la producción de software, confirman que en este momento no existen experiencias en la realización de pruebas a las Bases de Datos en función del rendimiento, además no se tiene en cuenta este requisito en la etapa inicial de los proyectos. Entre las principales problemáticas identificadas se encuentran:

1. No se identifican las tablas de gran tamaño y los procesos más complejos que realizará la Base de Datos.
2. No se define la cuota máxima de usuarios con acceso a la Base de Datos.
3. No se realiza un análisis de concurrencia de usuarios a la Base de Datos.
4. No se realiza un análisis de los requisitos de hardware para el montaje de la Base de Datos.
5. Existen problemas en el levantamiento de los requisitos de rendimiento.
6. No se realizan pruebas de carga, stress y volumen a las Bases de Datos.

Las mejoras en el rendimiento apuntan al coste y la calidad del producto. Cuanto menor sea el tiempo que lleve obtener el resultado deseado, más productivo es el sistema.

Un error que comúnmente se comete es realizar la evaluación a los productos software en la fase de pruebas dentro del proyecto. Una prueba es un conjunto de procesos que permiten verificar y reflejar la calidad de un producto de software, por tanto, no se puede dejar para después de la materialización del

software, la evaluación debe comenzar a diseñarse en paralelo al ciclo de desarrollo, empleando para ello casos de pruebas, aunque la aplicación como tal de las pruebas de rendimiento sean después de las pruebas de integración del sistema.

Para obtener un buen rendimiento de las bases de datos se deben tener presente un conjunto de factores: Hardware, Software, Sistemas Gestores de Bases de Datos y el Negocio así como el tiempo de respuesta de cada operación realizada.

Es por ello que surge la necesidad de evaluar el rendimiento de las Bases de Datos en DATEC para realizar un diseño adecuado de las mismas, así como la selección de los recursos apropiados para lograr un beneficio mayor a los clientes en el uso de Bases de Datos rápidas y precisas y un uso eficiente de los recursos instalados.

Teniendo en cuenta lo anteriormente expuesto y la poca experiencia que existe en la universidad sobre el tema de la evaluación del rendimiento de las bases de datos, los esfuerzos estarán encaminados a resolver el siguiente **Problema Científico**: ¿Cómo evaluar el rendimiento de las bases de datos basadas en PostgreSQL para software de gestión en la Línea de Integración de Soluciones de DATEC?

Por lo que se define como **Objeto de estudio:** La evaluación del rendimiento de las bases de datos, centrándose más al **Campo de Acción:** La evaluación del rendimiento de las bases de datos basadas en PostgreSQL.

Para darle solución al problema científico se define como **Objetivo General:** Elaborar un procedimiento para la evaluación del rendimiento de las bases de datos basadas en PostgreSQL para software de gestión en la Línea de Integración de Soluciones de DATEC que permita la optimización de este atributo de calidad.

Para dar cumplimiento al objetivo general se definieron los siguientes **Objetivos Específicos:**

1. Elaborar el marco teórico de la Investigación.
2. Realizar un diagnóstico del proceso de evaluación del rendimiento de las Bases de Datos basada en PostgreSQL para software de gestión en la UCI y en la Línea de Integración de Soluciones de DATEC.
3. Definir un procedimiento para efectuar la evaluación del rendimiento de las Bases de Datos basada en PostgreSQL para software de gestión de la Línea de Integración de Soluciones de DATEC.

4. Aplicar el procedimiento propuesto para efectuar la evaluación del rendimiento de las Bases de Datos basada en PostgreSQL para software de gestión de la Línea de Integración de Soluciones de DATEC.
5. Validar los resultados de la Investigación mediante dos casos de estudio.

Para dar cumplimiento a estos objetivos se definieron las siguientes **Tareas de investigación:**

- Selección y revisión bibliográfica para actualizar los logros y limitaciones existentes sobre la evaluación del rendimiento de las Bases de Datos basadas en PostgreSQL para software de gestión.
- Realización de entrevistas con personas especializadas en el tema de rendimiento.
- Análisis de modelos de calidad tales como ISO/IEC 9126, FURPS, Dromey.
- Análisis de los tipos y métodos de pruebas que se realizan para medir el rendimiento de las Bases de Datos basada en PostgreSQL de los productos de la Línea de Integración de Soluciones.
- Análisis de las líneas de productos de software (LPS).
- Definición de rasgos característicos de software de gestión.
- Estudio de las herramientas de software que soportan el proceso de pruebas.
- Análisis de riesgos potenciales asociados con el atributo de calidad estudiado, los métodos y tipos de pruebas y las herramientas.
- Evaluación de la información obtenida y definir la posición como investigador.
- Identificación de los involucrados potenciales en el diagnóstico y caracterización de su marco de actuación respecto al tema de investigación.
- Estudio de los productos liberados y en desarrollo, ambientes de comprobación, herramientas software de soporte al desarrollo y procesos relacionados con el tema estudiado.
- Confección de síntesis de la información relacionada con la evaluación de rendimiento.
- Adaptación de los tipos y métodos de pruebas estudiados en función de las Bases de Datos basada en PostgreSQL para software de gestión en la Línea de Integración de Soluciones de DATEC.
- Realización del procedimiento a seguir acorde a las condiciones del entorno analizado.
- Aplicación del procedimiento de evaluación del rendimiento de las Bases de Datos basada en PostgreSQL para software de gestión en la Línea de Integración de Soluciones de DATEC.
- Validación de los resultados a partir del procedimiento aplicado a dos o más casos de estudio.

La estructuración del contenido de la investigación ha sido conformada en tres capítulos:

Capítulo 1: En este capítulo se va a trabajar todo el marco conceptual necesario para la realización de la investigación. Se tratan conceptos y definiciones que son necesarios para llevar a cabo esta investigación. Además, se evidencia la situación real del tema principal, la evaluación del rendimiento de las bases de datos.

Capítulo 2: En este capítulo se va a proponer un procedimiento que se obtenga después de una investigación concreta de todos los detalles, es decir, las herramientas y métodos a utilizar. El procedimiento permite una ordenada forma de evaluar el rendimiento de las bases de datos.

Capítulo 3: En este capítulo se va a realizar la validación del procedimiento, exponiendo los resultados de dos casos de estudio, donde fue aplicado. Arrojando los resultados exactos de la situación real de estos productos.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción.

En el presente capítulo se refleja la fundamentación teórica que se ha llevado a cabo en esta investigación. Se explican las características de algunos modelos de calidad que existen para así poder definir la posición que como investigador se va a seguir. Se enuncian un grupo de conceptos importantes en el área de las bases de datos, Líneas de Productos de Software (LPS) y pruebas de software que son claves en el desarrollo de este trabajo. Por último, se podrá encontrar la evaluación del rendimiento actual en este entorno y las tendencias hacia el punto central donde se dirige la investigación.

1.1. Modelos de Calidad de Software.

A lo largo de los años se han definido diferentes modelos que pretenden caracterizar la calidad del software haciendo una división clara entre modelos de calidad del proceso y modelos de calidad del producto.

Los modelos de calidad del producto pretenden definir las características que deben cumplir un producto software para obedecer con determinados criterios de calidad, de forma tal que su calidad se pueda cuantificar a través de atributos medibles. La diferencia básica entre los diferentes modelos consiste en la clasificación taxonómica que realiza cada uno basada en niveles, los cuales pueden variar en cuanto a la relación, cantidad y concepto. (2)

La ventaja de los modelos de calidad es que la calidad se convierte en algo concreto, que se puede definir, que se puede medir y, sobre todo, que se puede planificar. Ayudan también a comprender las relaciones que existen entre diferentes características de un producto software.

En los modelos de calidad, la calidad se define de forma jerárquica. Resuelven la complejidad mediante la descomposición. Es un concepto que se deriva de un conjunto de sub-conceptos, cada uno los cuales se van a evaluar a través de un conjunto de indicadores o métricas.

En esta investigación se va a centrar más en los modelos de calidad del producto y en el atributo de calidad rendimiento o eficiencia según como esté definido acorde a la bibliografía.

1.1.1. Modelo ISO/IEC 9126.

El estándar o modelo ISO/IEC 9126 ha sido desarrollado en un intento de identificar los atributos clave de calidad para un producto de software. Este estándar es una simplificación del Modelo de McCall, e identifica seis características básicas de calidad que pueden estar presentes en cualquier producto de

software. El estándar provee una descomposición de las características en sub-características, que se muestran en la Tabla 1.

Característica	Sub-características
Funcionalidad	➢ Adecuación. ➢ Exactitud. ➢ Interoperabilidad. ➢ Seguridad.
Confiabilidad	➢ Madurez. ➢ Tolerancia a fallas. ➢ Recuperabilidad.
Usabilidad	➢ Entendibilidad. ➢ Capacidad de aprendizaje. ➢ Operabilidad.
Eficiencia	➢ Comportamiento en tiempo. ➢ Comportamiento de recursos.
Mantenibilidad	➢ Analizabilidad. ➢ Modificabilidad. ➢ Estabilidad. ➢ Capacidad de pruebas.
Portabilidad	➢ Adaptabilidad. ➢ Instalabilidad. ➢ Reemplazabilidad.

Tabla 1: Características y sub-características de calidad – Modelo ISO/IEC 9126

Es interesante destacar que los factores de calidad que contempla el estándar ISO/IEC 9126 no son necesariamente usados para mediciones directas, pero proveen una valiosa base para medidas indirectas y una excelente lista para determinar la calidad de un sistema.(3)

1.1.2. **Modelo de FURPS**.

El modelo de McCall ha servido de base para modelos de calidad posteriores, y este es el caso del modelo FURPS, producto del desarrollo de Hewlett-Packard (1987). En este modelo se desarrollan un conjunto de factores de calidad de software, bajo el acrónimo de FURPS: funcionalidad (Functionality),

usabilidad (Usability), confiabilidad (Reliability), desempeño (Performance) y capacidad de soporte (Supportability). La Tabla 2 presenta la clasificación de los atributos de calidad que se incluyen en el modelo, junto con las características asociadas a cada uno.

Factor de Calidad	Atributos
Funcionalidad	➢ Características y capacidades del programa. ➢ Generalidad de las funciones. ➢ Seguridad del sistema.
Facilidad de uso	➢ Factores humanos. ➢ Factores estéticos. ➢ Consistencia de la interfaz. ➢ Documentación.
Confiabilidad	➢ Frecuencia y severidad de las fallas. ➢ Exactitud de las salidas. ➢ Tiempo medio de fallos. ➢ Capacidad de recuperación ante fallas. ➢ Capacidad de predicción.
Rendimiento	➢ Velocidad del procesamiento. ➢ Tiempo de respuesta. ➢ Consumo de recursos. ➢ Rendimiento efectivo total. ➢ Eficacia.
Capacidad de Soporte	➢ Extensibilidad. ➢ Adaptabilidad. ➢ Capacidad de pruebas. ➢ Capacidad de configuración. ➢ Compatibilidad. ➢ Requisitos de instalación.

Tabla 2: Atributos de calidad – Modelo FURPS

El modelo FURPS incluye, además de los factores de calidad y los atributos, restricciones de diseño y requerimientos de implementación, físicos y de interfaz. Las restricciones de diseño especifican o

restringen el diseño del sistema. Los requerimientos de implementación especifican o restringen la codificación o construcción de un sistema (por ejemplo, estándares requeridos, lenguajes, políticas). Por su parte, los requerimientos de interfaz especifican el comportamiento de los elementos externos con los que el sistema debe interactuar. Por último, los requerimientos físicos especifican ciertas propiedades que el sistema debe poseer, en términos de materiales, forma, peso, tamaño (por ejemplo, requisitos de hardware, configuración de red). (3)

1.1.3. Modelo de Dromey.

Dromey (4) propuso un marco de referencia – o meta modelo - para la construcción de modelos de calidad, basado en cómo las propiedades medibles de un producto de software pueden afectar los atributos de calidad generales, como por ejemplo, confiabilidad y mantenibilidad. El problema que se plantea es cómo conectar tales propiedades del producto con los atributos de calidad de alto nivel. Para solventar esta situación, Dromey sugiere el uso de cuatro categorías que implican propiedades de calidad, que son: correctitud, internas, contextuales y descriptivas.

La Tabla 3 presenta la relación que establece Dromey entre las propiedades de calidad del producto y los atributos de calidad de alto nivel.

Propiedades del producto	Atributos de Calidad
Correctitud	➢ Funcionalidad. ➢ Confiabilidad.
Internas	➢ Mantenibilidad. ➢ ***Eficiencia.*** ➢ Confiabilidad.
Contextuales	➢ Mantenibilidad. ➢ Reusabilidad. ➢ Portabilidad. ➢ Confiabilidad.
Descriptivas	➢ Mantenibilidad. ➢ Reusabilidad. ➢ Portabilidad. ➢ Usabilidad.

Tabla 3: Relación entre propiedades del producto y atributos de calidad – Modelo de Dromey.

El proceso de construcción de modelos de calidad propuesto por Dromey consta de 5 pasos, basados en las propiedades mencionadas (3). Los pasos del marco de referencia propuesto son:

- Especificación de los atributos de calidad de alto nivel (por ejemplo: confiabilidad, mantenibilidad).
- Determinación de los distintos componentes del producto a un apropiado nivel de detalle (por ejemplo: paquetes, subrutinas, declaraciones).
- Para cada componente, determinación y categorización de sus implicaciones más importantes de calidad.
- Proposición de enlaces que relacionan las propiedades implícitas a los atributos de calidad, o, alternativamente, uso de enlaces de las cuatro categorías de atributos propuestas.
- Iteración sobre los pasos anteriores, utilizando un proceso de evaluación y refinamiento.

Después de haber analizado estos modelos de calidad se puede definir que el rendimiento no es más que una: “Capacidad del producto de software para proporcionar apropiados tiempos de respuesta y procesamiento haciendo un uso adecuado de los recursos de hardware y software”.

Teniendo en cuenta este concepto de rendimiento, se plantea que el modelo que se va a utilizar para la investigación es el modelo de FURPS, pues es el modelo que más se acerca a esta definición y es donde se explican mejor los atributos en los que es caracterizado el rendimiento.

1.2. Líneas de Productos de Software (LPS).

¿Qué es una Línea de Productos de Software?

Se puede decir que la idea básica es:

- Ensamblaje de partes de software previamente elaboradas.
- Inspirada en los procesos de producción de sistemas físicos.
- Fundamentada en la Reutilización de Software.
- Asume la existencia de una industria de partes.

Pero según el criterio de diversos autores se define como:

- **Krueger:**"... técnicas de ingeniería para crear un portafolio de sistemas de software similares, a partir de un conjunto compartido de activos de software, usando un medio común de producción". (5)
- **Gomma:**"... una familia de sistemas de software que tienen una funcionalidad común y alguna funcionalidad variable". (6)

1.2.1. Rasgos Característicos.

Modelo básico de un LPS (7):

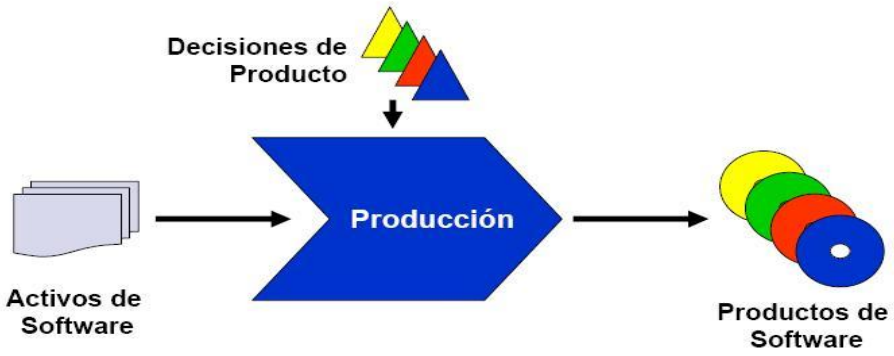


Figura 1: Modelo Básico de una Línea de Productos de Software.

Donde:

- **La entrada: Activos de Software** no es más que una colección de partes de software (requisitos, diseños, componentes, casos de prueba) que se configuran y componen de una manera prescrita para producir los productos de la línea.
- **El control: Modelos de Decisión y Decisiones de Productos es:**

  Los Modelos de Decisiones describen los aspectos variables y opcionales de los productos de la línea.

  Cada producto de la línea es definido por un conjunto de decisiones (decisiones del producto).
- **El proceso de producción:**

  Establece los mecanismos o pasos para componer y configurar productos a partir de los activos de entrada.

  Las decisiones del producto se usan para determinar qué activos de entrada utilizar y cómo configurar los puntos de variación de esos activos.
- **La salida: Productos de software** es el conjunto de todos los productos que pueden o son producidos por la línea de productos.

Este modelo trae como **beneficios** (7):

- La entrega de productos de software de una manera más rápida, económica y con una mejor calidad.
- Las LPS producen mejoras en:

  Tiempo de entrega del producto (time to market).

Costos de ingeniería.

Tamaño del portafolio de productos.

Reducción de las tasas de defectos.

Calidad de los productos.

1.2.2. Tipos de Productos (Software de Gestión).

Antes de conocer qué es software de gestión, primero se debe empezar conociendo qué es software. El software es la parte lógica o intangible del sistema, es decir, lo que no se puede tocar ni ver del computador.

Software de Gestión: Es un programa que sirve como herramienta desarrollado especialmente para adecuarse a los diferentes requerimientos de las empresas. Es una solución diseñada para empresas medianas y grandes dinámicas con necesidades de alta competitividad, que buscan la eficiencia en sus procesos internos y en la gestión con terceros. (8)

Clasificación del software de gestión (8):

- **Gestión Dinámica:** Esta gestión suministra la información directiva para la toma de decisiones en una empresa.
- **Gestión de Ventas:** (Clientes, pedidos, Facturación, Cobranzas, Logística). El objetivo de este módulo es ordenar la administración de las ventas.
- **Gestión de Compras:** (Proveedores, Cuentas a Pagar, Requisiciones, Cumplimientos). El objetivo de este módulo es ordenar la administración de las compras locales y de importación.
- **Gestión de Finanzas:** (Planificación, Control Presupuestario, Flujo de Caja). El objetivo de este módulo es integrar toda la información de los otros módulos.
- **Gestión de Contabilidad:** (Plan de Cuentas, Impuestos, Bienes de uso). El objetivo de este módulo es integrar el plan de cuentas en donde se apoyan todos los registros contables.
- **Gestión de Producción Industrial:** (Procesos, Órdenes, Plan de Producción, Costos). El objetivo de este módulo es poder optimizar la administración de la producción, integrando los procesos productivos con los módulos de compras e inventarios.
- **Gestión de Recursos Humanos:** (Liquidación de sueldos y Gestión Personal). Este módulo permite principalmente, procesar las liquidaciones de sueldo y jornales generando la contabilización correspondiente.

1.2.3. Ejemplos de Aplicación Real.

El Centro de Tecnologías de Gestión de Datos (DATEC), trazó como objetivo central de su proceso productivo el concepto de reutilización, siguiendo el paradigma del desarrollo de Líneas de Productos de Software. Durante la implantación de este modelo comúnmente aparecen muchos obstáculos convirtiéndolo en un proceso complejo, sobre todo por la tradición de desarrollar *"Proyectos"* de la manera clásica. Para ellos se organizaron tres Líneas de Productos:

- PostgreSQL.
- Almacenes de Datos
- Integración de Soluciones.

Para esta investigación se hace énfasis en la línea de Integración de Soluciones porque es la línea que hace software de gestión, y este se clasifica en gestión dinámica ya que todos los productos que desarrollan sirven para la toma de decisiones.

1.3. Sistema Gestor de Bases de Datos (SGBD).

Un **Sistema Gestor de Bases de Datos (SGBD) o DBMA (DataBase Management System)** es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos. Algunos ejemplos de SGBD son: Oracle, DB2, PostgreSQL, MySQL, MS SQL Server. (9)

Un SGBD debe permitir: (9)

- **Definir una base de datos:** especificar tipos, estructuras y restricciones de datos.
- **Construir la base de datos:** guardar los datos en algún medio controlado por el mismo SGBD.
- **Manipular la base de datos:** realizar consultas, actualizarla, generar informes.

Las características de un SGBD son: (9)

- **Abstracción de la información.** Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. Da lo mismo si una base de datos ocupa uno o cientos de archivos, este hecho se hace transparente al usuario. Así, se definen varios niveles de abstracción.
- **Independencia.** La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.

- **Redundancia mínima.** Un buen diseño de una base de datos logrará evitar la aparición de información repetida o redundante. De entrada, lo ideal es lograr una redundancia nula; no obstante, en algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancias.
- **Consistencia.** En aquellos casos en los que no se ha logrado esta redundancia nula, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea.
- **Seguridad.** La información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentra asegurada frente a usuarios malintencionados, que intenten leer información privilegiada; frente a ataques que deseen manipular o destruir la información; o simplemente ante las torpezas de algún usuario autorizado pero despistado. Normalmente, los SGBD disponen de un complejo sistema de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías de permisos.
- **Integridad.** Se trata de adoptar las medidas necesarias para garantizar la validez de los datos almacenados. Es decir, se trata de proteger los datos ante fallos de hardware, datos introducidos por usuarios descuidados, o cualquier otra circunstancia capaz de corromper la información almacenada.
- **Respaldo y recuperación.** Los SGBD deben proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada en ellos y de restaurar a partir de estas copias los datos que se hayan podido perder.
- **Control de la concurrencia.** En la mayoría de entornos (excepto quizás el doméstico), lo más habitual es que sean muchas las personas que acceden a una base de datos, bien para recuperar información o para almacenarla. Y es también frecuente que dichos accesos se realicen de forma simultánea. Así pues, un SGBD debe controlar este acceso concurrente a la información, que podría derivar en inconsistencias.

1.3.1. Gestor de Bases de Datos PostgreSQL. Características Distintivas.

PostgreSQL (10) es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD (Berkeley Software Distribution) y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

A continuación se expone un gráfico que ilustra de manera general los componentes más importantes en un sistema PostgreSQL.

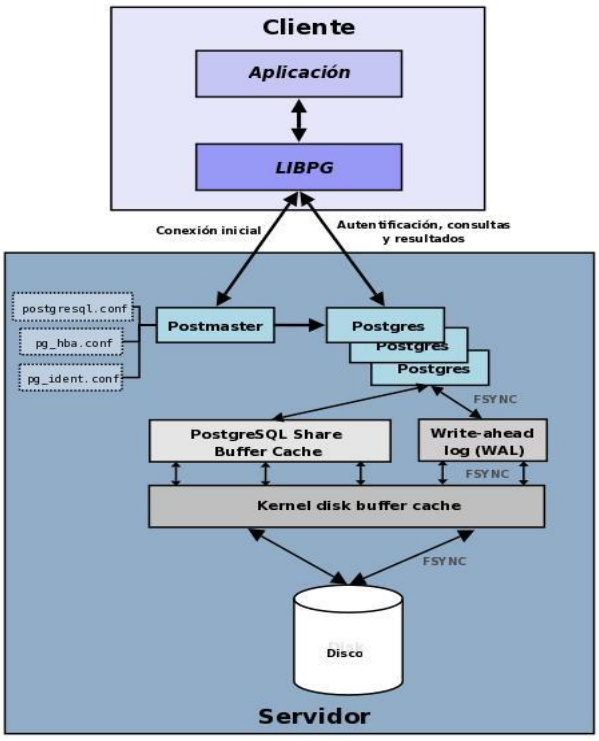


Figura 2: Componentes más importantes en un sistema PostgreSQL.

- **Aplicación cliente:** Esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos. La conexión puede ocurrir vía TCP/IP ó sockets locales.
- **Demonio postmaster:** Este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket por conexiones entrantes de clientes. También es el encargado de crear los procesos hijos que se encargarán de autentificar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes.
- **Ficheros de configuración:** Los 3 ficheros principales de configuración utilizados por PostgreSQL son: postgresql.conf, pg_hba.conf y pg_ident.conf.

- **Procesos hijos postgres:** Procesos hijos que se encargan de autentificar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes.
- **PostgreSQL share buffer cache:** Memoria compartida usada por PostgreSQL para almacenar datos en caché.
- **Write-Ahead Log (WAL):** Componente del sistema encargado de asegurar la integridad de los datos (recuperación de tipo REDO).
- **Kernel disk buffer cache:** Caché de disco del sistema operativo.
- **Disco:** Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

Características

La última serie de producción es la 8.4, siendo la última versión disponible actualmente la 8.4.3.

Sus características técnicas la hacen una de las bases de datos más potentes y robustas del mercado. Su desarrollo comenzó hace más de 15 años, y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.

A continuación se tiene algunas de las características más importantes y soportadas por PostgreSQL en función del rendimiento:

Generales

- Es una base de datos 100% ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad).
- Integridad referencial.
- Tablespaces.
- Copias de seguridad en caliente (Online/hot backups).
- Unicode.
- Juegos de caracteres internacionales.
- Multi-Version Concurrency Control (MVCC).
- Acceso encriptado vía SSL (Secure Sockets Layer).
- Licencia BSD.

Algunos de los límites de PostgreSQL son:

Límite	Valor
Máximo tamaño base de dato.	Ilimitado (Depende de tu sistema de almacenamiento).
Máximo tamaño de tabla.	32 TB.
Máximo tamaño de fila.	1.6 TB.
Máximo tamaño de campo.	1 GB.
Máximo número de filas por tabla.	Ilimitado.
Máximo número de columnas por tabla.	250 - 1600 (dependiendo del tipo).
Máximo número de índices por tabla.	Ilimitado.

Tabla 4: Límites de PostgreSQL.

1.3.2. El rendimiento en el SGBD PostgreSQL.

PostgreSQL se diseñó como una base de datos orientada a objetos, es decir, una ORDBMS (Object Relational DataBase Management System). Esto significa, que las tablas no son tablas, sino objetos y las tuplas son instancias de ese objeto. Se pueden crear nuevos tipos de datos, hacer herencias entre objetos. PostgreSQL tiene transacciones, integridad referencial, vistas, y multitud de funcionalidades, pero es lento y pesado. Esta última característica hay que detallarla bien por versiones porque este problema se ha tratado de darle solución entre una versión y otra. Por ejemplo en la versión de PostgreSQL 8.2 las tablas crecen significativamente, ya entrando en la versión 8.3 las tablas crecen un poco menos antes de la aparición de HOT (Heap Only Tuples). En PostgreSQL 8.3 con HOT, se observa un orden de magnitud diferente porque disminuye el tamaño, aumenta la cantidad de transacciones y disminuye la fragmentación. Ya en PostgreSQL 8.4 que es la versión más actual que existe debido a que su lanzamiento fue el 1ro de julio del 2009 se encuentran las siguientes mejoras:

- Restauración de bases de datos en procesos paralelos, que acelera la recuperación de un respaldo hasta 8 veces.
- Privilegios por columna, que permiten un control más granular de datos confidenciales.
- Configuración de ordenamiento adaptable por base de datos, lo cual hace a PostgreSQL más útil en entornos con múltiples idiomas.

- Actualizaciones “en el lugar” desde 8.3 a 8.4 con muy bajo downtime, gracias al uso de pg_migrator beta.
- Nuevas herramientas de monitoreo de consultas que le otorgan a los administradores mayor información sobre la actividad del sistema.

Muchos se preguntarán por qué PostgreSQL y no otro sistema gestor. La respuesta está en que DATEC utiliza este gestor de base de datos porque migró a tecnologías de bases de datos libres y este gestor libre es el más potente hasta el momento. Además, tienen la idea de crear su propio gestor de bases de datos con la base de PostgreSQL lo que contribuye a lograr la soberanía tecnológica en Cuba.

1.4. Tipos, Métodos y Herramientas de Pruebas.

Pruebas (test): «una actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto». (11)

Pruebas de Software: Ejecución de un programa con la intención de descubrir un error. Técnica experimental para la búsqueda de errores en los programas. (12)

Las pruebas de software son un conjunto de herramientas, técnicas y métodos que hacen a la excelencia del desempeño de un programa, así como también la mejor publicidad que una empresa dedicada a la producción de software pueda tener. Las técnicas para encontrar problemas en un programa son extensamente variadas y van desde el uso del ingenio por parte del personal de prueba hasta herramientas automatizadas que ayudan a aliviar el peso y el costo de tiempo de esta actividad. (13)

1.4.1. Tipos de Pruebas.

1.4.1.1. Pruebas de Carga.

Las pruebas de carga son el tipo más sencillo de pruebas de rendimiento. Una prueba de carga se realiza generalmente para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperada. Se utiliza para encapsular pruebas no manuales, es decir, unitarias, Web, genéricas, por orden y luego ejecutarlas simultáneamente a través de usuarios virtuales. La ejecución de estas pruebas con carga genera resultados, incluidos contadores de rendimiento y de otros tipos, en tablas y en gráficos. (14)

Por ejemplo, esta carga puede ser el número esperado de usuarios concurrentes utilizando la aplicación y que realizan un número específico de transacciones durante el tiempo que dura la carga. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación. Si la

base de datos y el servidor de aplicaciones también se monitorizan, entonces esta prueba puede mostrar el cuello de botella en la aplicación. (14)

1.4.1.2. Pruebas de Stress.

Las pruebas de stress son pruebas de carga y rendimiento basadas en la funcionalidad del sistema bajo cargas pesadas, un gran número de repeticiones, manejo de grandes datos y demasiadas preguntas a bases de datos grandes (13). Esta prueba se utiliza normalmente para romper la aplicación. Se va doblando el número de usuarios que se agregan a la aplicación y se ejecuta una prueba de carga hasta que se rompe. Este tipo de prueba se realiza para determinar la solidez de la aplicación en los momentos de carga extrema y ayuda a los administradores para determinar si la aplicación rendirá lo suficiente en caso de que la carga real supere a la carga esperada.

1.4.1.3. Pruebas de Volumen.

Las pruebas de volumen verifican las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la BD. (15)

Las pruebas de volumen están sujetas a grandes cantidades de datos para determinar si cuando los límites son alcanzados causan que el software falle. Las pruebas de volumen también identifican la carga continua máxima o el volumen que las pruebas pueden manejar durante un período dado. Por ejemplo, si las pruebas procesan un conjunto de registros de base de datos para generar un informe, una prueba de volumen debería usar una base de datos grande de prueba y comprobar que el software se comportó normalmente y produjo el informe correcto. (16)

1.4.2. Métodos de Pruebas.

1.4.2.1. Caja Blanca.

El método de prueba de Caja Blanca se basa en el minucioso examen de los detalles procedimentales, requieren del conocimiento de la estructura interna del programa y son derivadas a partir de las especificaciones internas de diseño o el código.

Mediante los métodos de prueba de la caja blanca, el ingeniero de software puede obtener casos de prueba que garanticen que:

- Se ejerciten por lo menos una vez todos los caminos independientes para cada módulo.
- Se ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsas.
- Ejecuten todos los bucles en sus límites y con sus límites operacionales.
- Se ejerciten las estructuras internas de datos para asegurar su validez.

Es por ello que se considera a la prueba de Caja Blanca como uno de los tipos de pruebas más importantes que se le aplican a los software, logrando como resultado que disminuya en un gran por ciento el número de errores existentes en los sistemas y por ende una mayor calidad y confiabilidad. (15)

1.4.2.2. Caja Negra.

El método de prueba de Caja Negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del software, por lo que los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta, así como que la integridad de la información externa se mantiene. Esta prueba examina algunos aspectos del modelo fundamentalmente del sistema sin tener mucho en cuenta la estructura interna del software.

Este método se centra principalmente en los requisitos funcionales del software. Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos.

La prueba de Caja Negra no es una alternativa a las técnicas de prueba de la Caja Blanca, sino un enfoque complementario que intenta descubrir diferentes tipos de errores a los encontrados en los métodos de la Caja Blanca.

Muchos autores consideran que estas pruebas permiten encontrar: (15)

- 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.

1.4.3. Técnicas de prueba de Caja Negra.

Para desarrollar la prueba de caja negra existen varias técnicas, entre ellas están: (17)

- **Técnica de la Partición de Equivalencia:** esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- **Técnica del Análisis de Valores Límites:** esta técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- **Técnica de Grafos de Causa-Efecto:** es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

1.4.4. Herramientas de Pruebas.

Antes de hacer énfasis en JMETER y SELENIUM que son las herramientas de pruebas que más se conocen en la UCI, se exponen otras herramientas que existen para el entorno de prueba de rendimiento que dan soporte a las actividades de QA (Aseguramiento de la calidad).

Las herramientas de mayor utilidad para los entornos de pruebas de rendimiento son:

- **Open Load Tester de Open Demand Systems:** Es la primera solución rápida de optimización de rendimiento basada en navegador, fácil de usar, para las pruebas de carga y stress de aplicaciones y sitios web dinámicos. (18)
- **QALoad de Compuware:** Es una herramienta de pruebas de carga que ayuda a los equipos de pruebas, desarrolladores y jefes de proyecto a realizar pruebas de carga efectiva a aplicaciones distribuidas. (18)
- **SOATest de Parasoft:** Proporciona las pruebas y verificación instantánea de Web services, simplifica los desarrollos SOA, automatiza las pruebas funcionales cliente/servidor, pruebas de regresión, pruebas de carga y rendimiento. (18)
- **DataGenerator:** La idea del generador de datos es que es un guión libre, de código abierto escrito en Java Script, PHP y MySQL que le permite generar rápidamente grandes volúmenes de datos personalizados en diversos formatos para su uso en pruebas de software y poblar bases de datos. (19)
- **Nagios:** Es un sistema de monitorización de redes de código abierto ampliamente utilizado, que vigila los equipos (hardware) y servicios (software) que se especifiquen, alertando cuando el comportamiento de los mismos no sea el deseado. Entre sus características principales figuran la monitorización de servicios de red, la monitorización de los recursos de sistemas hardware (carga del procesador, uso de los discos, memoria, estado de los puertos), independencia de sistemas operativos, posibilidad de monitorización remota mediante túneles SSL cifrados ó SSH. (20)

1.4.4.1. JMETER.

JMeter es un proyecto de Apache Jakarta que puede ser utilizado como una herramienta de prueba de carga para analizar y medir el desempeño de una variedad de servicios, con énfasis en aplicaciones web.

JMeter puede ser usado como una herramienta de pruebas unitarias para conexiones de bases de datos con JDBC, FTP, LDAP, Servicios web, JMS, HTTP y conexiones TCP genéricas. JMeter puede también

ser configurado como un monitor, aunque es comúnmente considerado una solución ad-hoc respecto a soluciones avanzadas de monitoreo.

Mientras que JMeter es clasificado como una herramienta de "generación de carga", no es una descripción completa de la herramienta. JMeter soporta aserciones para asegurarse que los datos recibidos son correctos, por cookies de hilos, configuración de variables y una variedad de reportes. (21)

1.4.4.2. SELENIUM.

Selenium es un set de herramientas que permiten desarrollar scripts para pruebas de aplicaciones Web en diversos lenguajes como Java, Ruby, Python, Perl, .Net o PHP. Es un producto ofrecido como Open Source que está permanentemente siendo optimizado. La versión más reciente es Beta 2 lanzada la primera quincena de enero del 2009. Existen 3 tipos de Herramientas: Selenium IDE, Selenium RC, y Selenium Grids. (22)

- **Selenium IDE** es un plugin para Firefox que permite grabar y ejecutar scripts directamente desde tu navegador.
- **Selenium RC** es una biblioteca (library) y servidor escrito en lenguaje Java que permite ejecutar scripts en forma local o remota a través de comandos.
- **Selenium Grids** permite coordinar múltiples servidores Selenium para así poder ejecutar scripts en múltiples plataformas y equipos al mismo tiempo.

Las pruebas de software son un punto clave en esta investigación porque son las que van ayudar a dar un criterio de evaluación. Para esta investigación hace falta realizar pruebas de carga, stress y volumen porque las pruebas de carga y stress permiten saber la cantidad de usuarios que laboran concurrentemente, lo que las diferencian es que las pruebas de carga se hacen aumentando la carga lentamente mientras que las de stress se realiza aumentando la carga más rápido y las pruebas de volumen se realizan para llenar la base de datos con una gran carga de datos. Como método de prueba se realizará el de caja negra con la técnica partición de equivalencia que es la que permite interactuar con la interfaz usando un juego de datos para así ejercitar determinadas funciones del software y como herramientas de pruebas se utilizarán JMeter y DataGenerator; se escoge la herramienta JMeter porque es la herramienta mas completa que hay ya que permite hacer pruebas de carga y stress juntas, además es la mas utilizada en el mundo aunque consume gran cantidad de recurso.

1.5. Evaluación del rendimiento para software de gestión.

Según Pere Marqués la evaluación es: “Proceso de información, interpretación y valoración para la toma de decisiones y para la mejora”. (23)

Enfocado a este concepto se realizó una encuesta a los arquitectos (Ver anexo 1) y diseñadores de base de datos (Ver anexo 2) de distintos proyectos de la universidad y a DATEC, para saber si la evaluación del rendimiento para software de gestión utilizando el gestor de base de datos PostgreSQL, se lleva a cabo como un proceso para ser valorado a la hora de tomar decisiones en los proyectos. La encuesta se realizó con el objetivo de verificar la situación actual de los proyectos y del centro en función del conocimiento de pruebas a las bases de datos en función del rendimiento.

1.5.1. En la Universidad de las Ciencias Informáticas (UCI).

En la universidad la encuesta hecha para saber el conocimiento de los proyectos en la realización de pruebas a las bases de datos en función del rendimiento fue realizada a 7 arquitectos y 3 diseñadores de base de datos pertenecientes a los proyectos de: Atención Primaria de la Salud (Facultad 7), Hospitales (Facultad 7) y la Línea de Integración de Soluciones de DATEC; donde se desarrollan soluciones de tipo software de gestión y se utiliza además el SGBD PostgreSQL.

Los resultados obtenidos (Anexo 3) por parte de los 3 diseñadores de bases de datos lo cual representa el 100% fue que los índices de los diseños de la base de datos en los proyectos tienen un 66.7% con nivel medio y un 33.3% con nivel alto de dichos índices. En cuanto al nivel de normalización de las bases de datos están en tercera forma normal un 66.7% de los proyectos encuestados y un 33.3% en segunda forma normal de su nivel de normalización.

Por parte de los 7 arquitectos que representa el 100% se tiene que:

- El 42.8% de los arquitectos tiene un conocimiento de la existencia de herramientas de pruebas para el análisis del rendimiento de base de datos, mientras que el 28.6% saben que existen y el otro 28.6% desconoce del tema.
- En el conocimiento de las pruebas de carga, stress, tensión, volumen y rendimiento se tiene que: hay un 50% que no conoce las pruebas de carga y stress, un 100% que no tiene conocimiento de las pruebas de tensión y volumen y existe un 71.4% que desconoce las pruebas de rendimiento. Analizando estos por cientos de desconocimiento de estas pruebas se arriba a la conclusión que no se aplican.

- En cuanto a las técnicas de evaluación del rendimiento existe un 57.1% de conocimiento parcial y un 42.9% de conocimiento nulo sobre el tema, por tanto, se arriba a la conclusión que no se aplican técnicas para la evaluación del rendimiento.
- En el monitoreo del servidor cuando se ejecuta una consulta a la base de datos se tiene en cuanto al por ciento de uso del CPU hay un 71.4% que no lo hacen y un 28.6% que lo realizan. En el monitoreo del % de uso del disco duro hay un 85.7% que no lo hacen y un 14.3% que sí. En cuanto al monitoreo del aumento de consumo de memoria RAM hay un 71.4% que no lo hacen y un 28.6% que sí.
- Existe un 71.4% que conoce la velocidad de la red y un 28.6% que no tiene ese conocimiento.
- En el conocimiento de los sistemas de ficheros que tiene el sistema operativo hay un resultado de un 85.7% que si conocen del tema, en cuanto a utilizar el adecuado para las bases de datos hay un 57.1% positivo y un 42.9% que no saben.
- Hay un resultado de un 85.7% positivo en cuanto a la utilización de un sistema operativo versión servidor donde se encuentra físicamente la base de datos. En cuanto a la utilización del pool de conexiones en el servidor hay un 71.4 % que lo utiliza. En el análisis de la complejidad del negocio hay un 57.1 % que no lo tiene en cuenta para la selección del SGBD a utilizar y un 42.9% que si lo tiene en cuenta.
- En la configuración del uso de la caché y la cantidad de conexiones máximas para el SGBD PostgreSQL hay un 71.4% que si lo configuran, un 14.3% que no lo hacen y un 14.3% que no saben si se realiza.
- Por último, tenemos que un 71.4% no realiza el particionamiento de la base de datos y un 28.6% que si lo hacen.

1.5.2. En el Centro de Tecnologías de Gestión de Datos (DATEC).

En DATEC la encuesta fue hecha a los cinco arquitectos y al diseñador de base de datos de la Línea de Integración de Soluciones.

Según el diseñador de base de datos se tiene que el centro tiene un nivel medio de utilización de índice en el diseño de la base de datos y que el nivel de normalización de la base de datos se realiza hasta la segunda forma normal.

Según los 5 arquitectos de la línea que representa el 100% se tiene los siguientes resultados:

- En el centro existe un 60% de los encuestados que conoce la existencia de herramientas de pruebas para el análisis del rendimiento de las bases de datos pero no las utilizan, mientras que hay un 20% que saben que existen y otro 20% que no tienen idea de lo que le están preguntando.
- Se tiene un 60% de los encuestados que no conocen las pruebas de carga, stress y rendimiento y un 100% que no conocen las pruebas de tensión y volumen, por tanto, se arriba a la conclusión que no se aplican.
- Existe un 60% parcial de conocimiento de técnicas para la evaluación del rendimiento y un 40% nulo de conocimiento de dichas técnicas de evaluación del rendimiento por lo que se arriba a la conclusión de que no se aplican.
- Hay un 80% de certeza que cuando se ejecuta una consulta a la base de datos no se monitorea el por ciento de uso del CPU, tampoco se monitorea el por ciento de uso del disco duro y mucho menos se monitorea el aumento de consumo de memoria RAM. También existe un 80% de certeza que conocen la velocidad de la red.
- Un 80% de los encuestados afirman saber el sistema de fichero que tiene el sistema operativo en el servidor de base de datos; pero hay un 60% que no saben si se utiliza el adecuado y un 40% que afirman que se utiliza el indicado para el servidor.
- En el servidor donde se encuentra físicamente la base de datos hay instalado un sistema operativo versión servidor obtenido de un 80% de respuestas positivas y se utiliza el pool de conexiones sacado de un 60% de respuestas positivas y un 40% que no saben.
- Para la selección del SGBD no se tuvo en cuenta el análisis de la complejidad del negocio ya que existe un 80% de respuestas negativas.
- En el SGBD PostgreSQL si se configura el uso de la caché y la cantidad máxima de conexiones con 80% de respuestas positivas.
- Existe cierta contradicción en la realización del particionamiento de la base de datos porque hay 40% que dice que sí, otro 40% que no y el 20% restante no sabe.

Se puede concluir que en la universidad tanto en DATEC como en otras facultades no se tiene en cuenta el atributo rendimiento a la hora de planificar un proyecto. Esto se debe a que no existe un procedimiento por donde se pueden apoyar a la hora de planificar la evaluación del rendimiento.

1.6. Conclusiones.

En el presente capítulo se realizó un análisis sobre el atributo de calidad rendimiento o eficiencia (en dependencia de las bibliografías consultadas), en los modelos de calidad de ISO/IEC 9126, FURPS y Dromey; donde se arribó a una definición de este atributo de calidad. También se definió qué son las LPS, específicamente las de software de gestión y sus diferentes clasificaciones. Se habló a grandes rasgos del SGBD PostgreSQL enfocado al rendimiento y de los tipos, métodos y herramientas de pruebas que existen. Y por último se trató la situación de la evaluación del rendimiento para software de gestión en la universidad y en DATEC.

CAPÍTULO 2: PROPUESTA DE LA SOLUCIÓN

Introducción.

En el presente capítulo se muestra un procedimiento para la evaluación del rendimiento de las bases de datos basadas en PostgreSQL, la cual será aplicada al Centro de Tecnologías de Gestión de Datos (DATEC) en la Línea de Integración de Soluciones.

2.1. Necesidad del procedimiento.

La necesidad de este procedimiento es tener bien claro la importancia que se le atribuye a la evaluación del rendimiento. El rendimiento se puede ver afectado por:

- Parámetros del sistema operativo (tipos de sistemas operativos, políticas de planificación y gestión de procesos, configuración del sistema de memoria virtual).
- Componentes hardware del sistema (calidad y velocidad).
- Diseño de los programas (localidad en las referencias).
- Distribución de la carga.

Sabiendo esto entonces hay que pensar cómo se puede mejorar el rendimiento. Este se mejora por la actualización de los componentes (reemplazos de dispositivos más rápidos y añadir nuevas unidades) y el ajuste o sintonización (parámetros del sistema operativo y parámetros de las aplicaciones informáticas).

Pueden existir problemas prácticos con estas mejoras. En la actualización de componentes por ejemplo tenemos la compatibilidad con los existentes o la facilidad del sistema (biprocesadores, disco agrupados por matrices RAID) y en la sintonización puede ocurrir que el conocimiento profundo del sistema operativo no sea suficiente y también es posible una alteración de la fiabilidad.

El papel del rendimiento es importante para el funcionamiento de un sistema por tanto hay que controlarlo. La evaluación del rendimiento es importante porque:

- Los computadores tienen un precio que dependen de los costes de diseño y fabricación.
- Hace falta relacionar precio y calidad.
- El análisis de prestaciones y precios es un mecanismo que permite elegir entre productos.
- Terminología básica sobre rendimiento.

La evaluación nos sirve para optimizar el diseño de un sistema informático, seleccionar y ajustar un sistema informático y predecir la carga máxima aceptable que nos sirve de gran ayuda ya que el rendimiento siempre depende de la carga.

2.2. Procedimiento propuesto.

Un procedimiento es una sucesión cronológica de operaciones entre sí, que se constituyen en una unidad de función para la realización de una actividad específica dentro de un ámbito predeterminado de aplicación. Los procedimientos implican actividades, tareas del personal y tiempo de realización (24). A continuación se presenta un esquema de cómo va a estar estructurado el procedimiento propuesto:

Figura 3: Esquema del procedimiento.

2.3. Descripción del procedimiento.

2.3.1. Objetivo.

El objetivo de este procedimiento es evaluar el rendimiento de las bases de datos basadas en PostgreSQL, para los productos de software de gestión.

2.3.2. Alcance.

El procedimiento es aplicable a todos los productos de software de gestión que se desarrollen en la Línea de Integración de Soluciones en DATEC.

2.3.3. Definiciones, Acrónimos y Abreviaturas.

DATEC: Centro de Tecnologías de Gestión de Datos.

CP: Casos de Pruebas.

NC: No Conformidades.

PSC: Presencia de la sub-Característica.

PCCR: Presencia de la Característica de Calidad Rendimiento.

2.3.4. Roles.

Administrador de Pruebas: Responsable del éxito que tendrá la prueba que dirige y organiza. Responsable de elaborar el Plan de Pruebas y el Cronograma de Pruebas.

Analista de Pruebas: Es responsable de identificar y definir las pruebas requeridas en una iteración, supervisando el progreso de las pruebas, así como el resultado para cada ciclo de prueba.

Diseñador de Pruebas: Se centra en identificar las técnicas apropiadas, herramientas y pautas para llevar a cabo las pruebas y usar los recursos correspondientes para el esfuerzo de las mismas. Especifica el enfoque, así como las funcionalidades que serán probadas en la presente iteración.

Probador: Ejecuta o produce las principales pruebas del software y es quien obtiene directamente los resultados.

2.3.5. Métodos y Herramientas de Pruebas.

El método a utilizar es el de caja negra con la técnica partición de equivalencia explicada anteriormente en el capítulo 1 en el epígrafe 4.2.2 Caja Negra.

Las herramientas a utilizar son JMeter y DataGenerator explicadas también en el capítulo 1 en el epígrafe 4.3.1 JMeter y epígrafe 4.3 Herramientas para las pruebas simultáneamente.

2.3.6. Fases del procedimiento.

2.3.6.1. Fase Inicio.

La Fase Inicio tiene como objetivo definir y preparar las pruebas a realizar. Para ello se definen las actividades que se listan a continuación:

- Actividad 1: Planificación de las pruebas.
- Actividad 2: Propuesta de instrumento de medición: Lista de Chequeo.
- Actividad 3: Diseño de casos de pruebas.

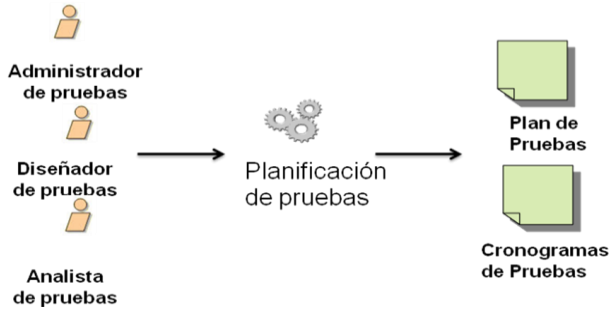


Figura 4: Actividad 1. Planificación de las pruebas.

Actividad 1: Planificación de las pruebas.

Paso 1: Se reúnen los trabajadores involucrados: Administrador de Pruebas, Analista de Pruebas y Diseñador de Pruebas para planificar las Pruebas de Caja Negra. Se describe la estrategia de prueba de la iteración definiendo claramente el método apropiado para ejecutarlas, los objetivos, las posibles pruebas a realizar, las pautas a seguir y el enfoque de las mismas.

Paso 2: Realizar el Plan de Prueba que describe los recursos y la planificación de las pruebas. Se realiza con el objetivo de esquematizar y comunicar la intención del esfuerzo de prueba de una planificación determinada, para obtener la aceptación y la aprobación de los interesados en el esfuerzo de prueba.

El Plan de Prueba forma la infraestructura dentro de la cual el equipo que realiza las pruebas trabajará durante la planificación determinada. Dirige, orienta y restringe el esfuerzo de prueba, centrándose el trabajo en los entregables útiles y necesarios. Como tal, el Plan de Prueba debe evitar detalles que no se entenderán o que los interesados considerarían irrelevantes en el esfuerzo de prueba. Para el procedimiento se adapta el Plan de Prueba definido en la Dirección de Calidad. (Ver Anexo 4 Plan de Prueba).

El Plan de Prueba está estructurado por:

- **Introducción:** se da una breve introducción del objetivo principal del artefacto, además del alcance, definiciones, acrónimos y abreviaturas y referencias que pueda tener.
- **Organización del Equipo de Prueba:** se pone los roles implicados con las pruebas y los nombres de los mismos.
- **Especificaciones de Software y Hardware:** el diseñador de prueba hace un levantamiento de los requisitos no funcionales de hardware y software que hacen falta para la realización de las pruebas.
- **Descripción del Plan de Prueba:** se especifican los requisitos funcionales que son evaluados y la función que realizan dentro de la aplicación.
- **Casos de Pruebas:** se realizan los diseños de los casos de pruebas.
- **Estrategia de Prueba:** se plantea el objetivo, la técnica, el proceso, los casos de pruebas y las herramientas a utilizar en las pruebas a realizar.
- **Recursos Requeridos:** se pone todo lo que hace cada rol involucrado en la prueba y la cantidad que existe de cada uno, se recoge en una tabla con las columnas: roles, cantidad y responsabilidad.

- **Calendario y Plazos:** son las fechas de realización de las pruebas.
- **Definición de los Entregables:** son los artefactos que se van a generar.
- **Seguimiento y Reporte de Defectos:** se plantean todas las dificultades encontradas para que el equipo de desarrollo le den solución.
- **Aprobación del Plan:** se manifiesta si el plan fue aprobado o no por el equipo de pruebas.
- **Documentación de resultados:** se pone la dirección o donde están ubicados todos los resultados de las pruebas junto a la documentación.

Paso 3: El Administrador de pruebas elaborará un Cronograma de Pruebas. Esta será una plantilla nueva que se creará. Enunciará cuándo se planificarán, diseñarán y ejecutarán las pruebas a la base de datos. (Ver Anexo 5 Cronograma de Prueba).

El Cronograma de Pruebas (Tabla 5) se compone de una tabla con la siguiente estructura:

- **Módulo:** nombre del módulo al que se le realizarán las pruebas.
- **Funcionalidad :** son las consultas que se van a ejecutar y las tablas a las que se van aplicar las pruebas
- **Fecha Diseño:** fecha en que se diseñan los casos de pruebas.
- **Fecha Ejecución:** fecha que se ejecutan las pruebas.

Figura 5: Actividad 2. Propuesta de instrumento de medición: Lista de chequeo.

Actividad 2: Propuesta de instrumento de medición: Lista de chequeo.

El analista de pruebas identifica y define la Lista de Chequeo que está basada en la identificación de las técnicas de prueba para evaluar cada sub-características del atributo de calidad rendimiento. Una lista de chequeo es un formulario de preguntas, las cuales dependen del objetivo para el cual son usadas. Para una adecuada forma de uso se pasa a describir la función de cada columna:

Peso: Define si el indicador a evaluar es crítico o no.

Evaluación: Es la forma de evaluar el indicador en cuestión. El mismo tiene tres formas de establecer la evaluación:

- Tipo A: Se coloca un 1 en caso de respuesta negativa. Se coloca un 4 en caso de respuesta afirmativa.
- Tipo B: Se establece un rango de números entre 1 y 4. A continuación se explica el significado de cada posible respuesta:
    1. Inaceptable.
    2. Promedio.
    3. Bueno.
    4. Excelente.
- Tipo C: Existen dos posibles respuestas: 1 ó 4. Colocar 1 significa si se utiliza en Sistema Propio de PostgreSQL. Se coloca 4 si se utilizan Sistemas Externos Main Caches.

Comentario: Especifica los señalamientos del porqué de su respuesta en cada caso de no ser la máxima evaluación.

Elementos definidos por la metodología: Abarca todos los indicadores a evaluar según la metodología. A continuación se presenta la lista de chequeo definida para evaluar el rendimiento de las bases de datos basadas en PostgreSQL basada en el Modelo de FURPS.

Elementos definidos por la metodología			
Indicadores a Evaluar	**Evaluación**	**Comentarios**	**Peso**

Lista de Chequeo de Rendimiento (FURPS).

Velocidad de Procesamiento			
Indicadores a Evaluar	**Evaluación**	**Comentarios**	**Peso**
1. ¿Con la normalización actual, es necesario desnormalizar para mejorar el rendimiento?	*Tipo A*		Crítico.
2. ¿Las consultas son escritas dentro de funciones o de procedimientos almacenados? (Pues se pueden escribir en forma de bloques).	*Tipo A*		Crítico.
3. ¿Cómo es el uso de disparadores (triggers) en tablas con integridad referencial?	*Tipo B*		Crítico.
4. ¿En dependencia del tipo de operación que se realice en la base de datos, se hace un uso eficiente de las transacciones?	*Tipo B*		Crítico.
5. ¿Cuál es el nivel de pruebas que se le aplica a la base de datos? (pruebas de carga, stress y	*Tipo B*		

volumen).			
6. Cuando ejecutan una consulta a la base de datos, ¿Se monitorea el aumento de consumo de memoria RAM?	*Tipo A*		Crítico.
7. ¿A qué nivel se configura el uso de la caché para PostgreSQL?	*Tipo C*		

Tiempo de Respuesta			
Indicadores a Evaluar	**Evaluación**	**Comentarios**	**Peso**
1. ¿Con la normalización actual, es necesario des normalizar para mejorar el rendimiento?	*Tipo A*		Crítico.
2. ¿Se tiene en cuenta la complejidad de las consultas SQL y su uso debido de recursos de la base de datos?	*Tipo A*		Crítico.
3. ¿Las consultas son escritas dentro de funciones o de procedimientos almacenados? (Pues se pueden escribir en forma de bloques).	*Tipo A*		Crítico.
4. ¿Cómo es el uso de vistas? (Vistas materializadas).	*Tipo B*		
5. ¿Cuál es el nivel de pruebas que se le aplica a la base de datos? (pruebas de carga, stress y volumen).	*Tipo B*		

Consumo de Recursos			
Indicadores a Evaluar	**Evaluación**	**Comentarios**	**Peso**
1. ¿Con la normalización actual, es necesario des normalizar para mejorar el rendimiento?	*Tipo A*		Crítico.
2. ¿Se tiene en cuenta la complejidad de las consultas SQL y su uso debido de recursos de la base de datos?	*Tipo A*		Crítico.
3. ¿El SGBD y/o la Capa de Acceso a Datos y/o los programas que se conectan a la BD son eficientes en su gestión de conexión? Por ejemplo: Mantienen un grupo de conexiones abiertas y no inician y cierran conexión repetidamente.	*Tipo A*		Crítico.
4. ¿Cuál es el nivel de pruebas que se le aplica a la base de datos? (pruebas de carga, stress y volumen).	*Tipo B*		

5. Cuando ejecutan una consulta a la base de datos, ¿Se monitorea el % de uso del CPU?	*Tipo A*		Crítico.
6. Cuando ejecutan una consulta a la base de datos, ¿Se monitorea el % de uso del disco duro?	*Tipo A*		Crítico.
7. Cuando ejecutan una consulta a la base de datos, ¿Se monitorea el aumento de consumo de memoria RAM?	*Tipo A*		Crítico.
8. ¿Es adecuado el sistema de fichero que utiliza el servidor de base de datos?	*Tipo A*		Crítico.
9. ¿A qué nivel se configura el uso de la caché para PostgreSQL?	*Tipo C*		
10. ¿Es realizado un estudio previo para la configuración de la cantidad de conexiones máximas en PostgreSQL? En caso de que se tengan muchas ¿Se usa un pooling de conexiones?	*Tipo B*		Crítico.
11. ¿Tienen una red dedicada entre la aplicación y la base de datos?	*Tipo A*		Crítico.
12. ¿El sistema operativo está optimizado para la ejecución de tareas de tipo servidor?	*Tipo B*		Crítico.
Rendimiento Efectivo Total			
Indicadores a Evaluar	**Evaluación**	**Comentarios**	**Peso**
1. ¿Con la normalización actual, es necesario desnormalizar para mejorar el rendimiento?	*Tipo A*		Crítico.
2. ¿El SGBD y/o la Capa de Acceso a Datos y/o los programas que se conectan a la BD son eficientes en su gestión de conexión? Por ejemplo: Mantienen un grupo de conexiones abiertas y no inician y cierran conexión repetidamente.	*Tipo A*		Crítico.
3. ¿Las consultas son escritas dentro de funciones o de procedimientos almacenados? (Pues se pueden escribir en forma de bloques).	*Tipo A*		Crítico.
4. ¿Cómo es el uso de vistas? (Vistas materializadas).	*Tipo B*		
5. ¿Cómo es el uso de disparadores (triggers) en tablas con integridad referencial?	*Tipo B*		Crítico.
6. ¿En dependencia del tipo de operación que se realice en la base de datos, se hace un uso eficiente	*Tipo B*		Crítico.

de las transacciones?			
7. ¿Cuál es el nivel de pruebas que se le aplica a la base de datos? (pruebas de carga, stress y volumen).	*Tipo B*		

Eficacia			
Indicadores a Evaluar	**Evaluación**	**Comentarios**	**Peso**
1. ¿Las consultas son escritas dentro de funciones o de procedimientos almacenados? (Pues se pueden escribir en forma de bloques).	*Tipo A*		Crítico.
2. ¿Cómo es el uso de vistas? (Vistas materializadas).	*Tipo B*		
3. ¿Cuál es el nivel de pruebas que se le aplica a la base de datos? (pruebas de carga, stress y volumen).	*Tipo B*		

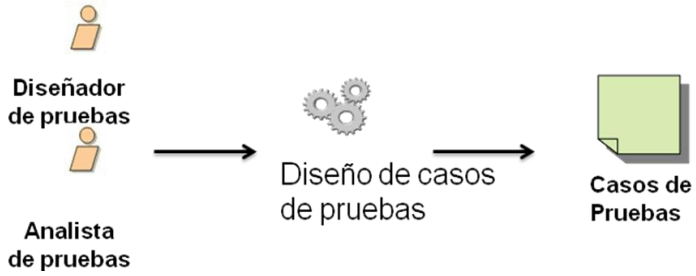


Figura 6: Actividad 3. Diseño de casos de pruebas.

Actividad 3: Diseño de casos de pruebas.

El analista y el diseñador de prueba identifican y definen los casos de pruebas que se van a estructurar de acuerdo con los tipos de pruebas (carga, stress y volumen), se va a utilizar las plantillas definidas. (Ver Anexo 6 Diseño de Casos de Pruebas).

Para las pruebas de carga y stress se va a llenar una tabla con la siguiente estructura:

- **Consulta a ejecutar:** se escribe la consulta a ejecutar a la base de datos.
- **Carga de trabajo:** la cantidad de usuarios para simular la carga de trabajo.
- **Resultados (seg):** el tiempo que se demora la base de datos en ejecutar esa consulta con esa cantidad de usuarios ejecutándola al mismo tiempo.

Para las pruebas de volumen se va a llenar una tabla con la siguiente estructura:

- **Tabla:** el nombre de las tablas a las cuales se le van hacer las pruebas.
- **Cantidad de registros:** cantidad de datos a generar en esas tablas.

- **Generados a la BD (seg):** el tiempo que se demora la base de datos para generar esos datos concurrentemente.

2.3.6.2. Fase Intermedia.

La fase intermedia tiene como objetivo la ejecución de las pruebas preparadas en la fase anterior. Para ello se propone las actividades siguientes:

- Actividad 1: Aplicación de la Lista de Chequeo.
- Actividad 2: Tabulación de la Lista de Chequeo.
- Actividad 3: Montaje del entorno de prueba.
- Actividad 4: Ejecución de las pruebas.

Figura 7: Actividad 1. Aplicación de la Lista de Chequeo.

Actividad 1: Aplicación de la Lista de Chequeo.

Paso 1: El probador entrevista a los diseñadores de base de datos y a los arquitectos del proyecto para aplicarle la lista de chequeo.

Paso 2: El probador llena la lista de chequeo de acuerdo con las respuestas obtenidas.

Paso 3: El probador le pasa al analista de prueba los resultados que arroja la lista de chequeo para que este analice los mismos.

Figura 8: Actividad 2. Tabulación de la Lista de Chequeo.

Actividad 2: Tabulación de la Lista de Chequeo.

La lista de chequeo está organizada en preguntas por cada sub-característica en la que es desglosado el rendimiento atendiendo al Modelo FURPS. Para dar respuesta a cada pregunta se utiliza la notación Tipo A, Tipo B o Tipo C.

El analista de prueba una vez realizada la evaluación por cada sub-característica, procede al procesamiento de cada resultado. Los resultados dependen de la importancia que cada involucrado en la

evaluación le asigne a la correspondiente sub-característica. Los involucrados dan una evaluación en la escala del 1 al 4 en la Tabla de Ponderación de Resultados (Tabla 6), donde 1 es menos significativo y 4 más significativo. Se realiza marcando con una cruz la casilla adecuada al Valor Asignado a cada sub-característica.

Característica de Calidad: Rendimiento				
Sub-características	**Valor Asignado**			
	1	**2**	**3**	**4**
Velocidad de Procesamiento.				
Tiempo de Respuesta.				
Consumo de Recursos.				
Rendimiento Efectivo Total.				
Eficacia.				

Tabla 5: Tabla de ponderación de Resultados.

A partir del procesamiento de cada una de las posibles respuestas dadas en la Lista de Chequeo, se pueden generar dos tipos de resultados:

1. **Resultados de la Presencia de las Sub-Características (PSC) en cada etapa del proceso de prueba, según la característica de calidad: Rendimiento.**

   Paso 1: Se calcula el promedio aritmético de las respuestas de cada pregunta de la Sub-característica que se está evaluando a través de la fórmula:

   PSC = ∑ X / Y, donde PSC: Presencia de la sub-característica; X: Evaluación; Y: Cantidad de evaluaciones por sub-característica.

   Paso 2: Sobre la base de los promedios anteriores, la presencia de las sub-características podrá encontrarse entre los siguientes rangos:

   - PSC = 1: La sub-característica no está presente en esta etapa.
   - 1<PSC≤2: La sub-característica está presente deficientemente.

- 2<PSC≤3: La sub-característica está presente.
- 3<PSC≤4= La sub-característica se encuentra altamente presente.

2. **Resultados de la Presencia de la Característica de Calidad Rendimiento (PCCR) considerando la importancia dada por los involucrados.**

   Una vez obtenidos los resultados de todas las sub-características, se procederá a realizar los cálculos para obtener la evaluación de la característica de calidad rendimiento. Este cálculo se realizará de la siguiente manera:

   Paso 1: Se calcula el promedio ponderado de las sub-características tomando en cuenta los Valores Asignados que se han hecho corresponder a cada una de ellas.

   Paso 2: Para calcular este promedio ponderado (PP) se multiplican los valores obtenidos de cada sub-característica (PSC) por su Valor Asignado (VA) correspondiente, se realiza a través de la fórmula: **PP = PSC * VA.**

   Paso 3: Se suman los valores obtenidos de la multiplicación y se divide este valor entre la suma de todos los Valores Asignados. Este cálculo se representa a través de la siguiente fórmula:

   PCCR = $\sum$**PP /** $\sum$**VA**; donde PCCR: Presencia de la Característica de Calidad Rendimiento; PP: Promedio Ponderado; VA: Valor Asignado a cada sub-característica.

El resultado obtenido representa la evaluación de la presencia de la Característica de Calidad Rendimiento dentro de la fase de pruebas del proyecto. Esta evaluación será representada en un rango del 1 al 4, al igual que los resultados dados para las sub-características. El valor obtenido es llevado a porcentaje con el fin de identificar si el mismo tiene el nivel de aceptabilidad. El nivel de aceptabilidad es 75% y es el recomendado para decidir si una característica está presente o no.

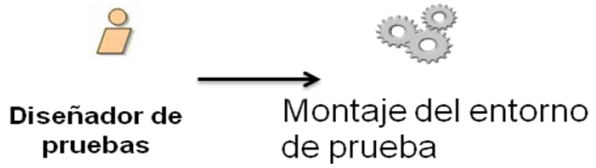


Figura 9: Actividad 3. Montaje del entorno de prueba.

Actividad 3: Montaje del entorno de prueba.

Paso 1: El diseñador de prueba precisa bien las computadoras donde se van a realizar las pruebas y con que cuentan para ejecutar las mismas.

Paso 2: El diseñador de prueba hace el levantamiento de los requisitos no funcionales de software y hardware que tiene instalado la computadora para realizar las pruebas. Estos requisitos tienen que corresponder al 100% con los descritos en el Plan de Pruebas e incluir las herramientas que se van a utilizar para las pruebas.

Figura 10: Actividad 4. Ejecución de las pruebas.

Actividad 4: Ejecución de las pruebas.

En esta actividad el probador desarrolla las pruebas diseñadas haciendo uso de los artefactos de entrada correspondientes y de las herramientas determinadas para cada tipo de prueba como son el JMeter y el DataGenerator. La ejecución de las pruebas debe estar estrictamente coordinada con lo establecido en el Plan de Pruebas, en el Cronograma de Pruebas, para evitar cualquier tipo de atraso. Durante este período se van a ejecutar los casos de prueba y se va a evaluar la ejecución del proceso de prueba, registrando los defectos o las no conformidades (NC) encontradas durante la ejecución. Se registrará las NC en la plantilla que pertenece al Expediente de Proyecto de la Dirección de Calidad. (Ver anexo 7 Registro de las NC).

El registro de las NC se recoge en una tabla con la estructura:

- **Elemento:** nombre del elemento.
- **No. :** número de la NC.
- **No Conformidad:** descripción de la NC.
- **Aspecto Correspondiente:** ubicación exacta de la NC.
- **Etapa de Detección:** etapa de detección del error.
- **Importancia:** como tal en la tabla se pone como columna significativa, no significativa y recomendación para darle la importancia que tiene el error detectado.
- **Estado NC:** se pone la fecha de la detección del error.
- **Respuesta del equipo de desarrollo:** esta columna se empieza a llenar después de la segunda iteración.

2.3.6.3. Fase Final.

La fase final tiene como objetivos comparar los resultados de la lista de chequeo y evaluar el resultado de las pruebas realizadas para así dar por terminado el procedimiento en su primera iteración. Para eso se cuenta con las actividades:

- Actividad 1: Comportamiento de la lista de chequeo.
- Actividad 2: Análisis de los resultados de las pruebas.

Figura 11: Actividad 1: Comportamiento de la lista de chequeo.

Actividad 1: Comportamiento de la lista de chequeo.

En esta actividad el administrador de prueba hace una comparación del comportamiento de las sub-características de rendimiento a partir del análisis de los resultados de la aplicación de la lista de chequeo. La aplicación de la lista de chequeo es como un diagnóstico de la situación real, o sea, un levantamiento de los problemas que existen.

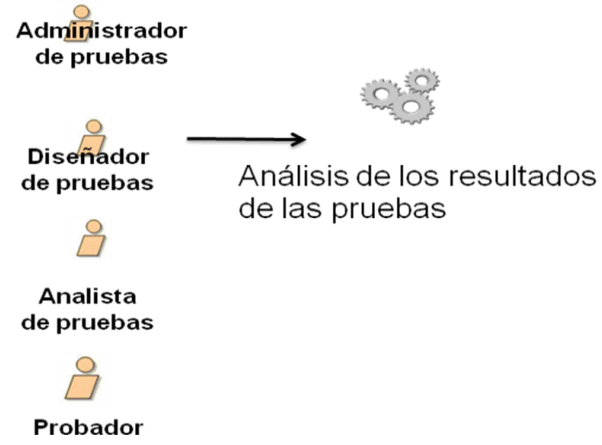


Figura 12: Actividad 2: Análisis de los resultados de las pruebas.

Actividad 2: Análisis de los resultados de las pruebas.

En esta actividad se reúnen el administrador, el diseñador, el analista de prueba y el probador para examinar todos los incidentes y anomalías de las pruebas. Es aquí donde se va a evaluar la realización de las pruebas, las condiciones de ejecución, si se presentó algún imprevisto, si el entorno de prueba no presentó problemas y cómo se desempeñaron los probadores. Se van a recopilar todas las no

conformidades (NC) encontradas y se van a entregar al equipo de desarrollo de la Base de Datos. El proceso de pruebas es evaluado de acuerdo a: la asistencia de los probadores, calidad del trabajo y cumplimiento del cronograma en un proceso de pruebas bueno, regular y malo según sus resultados.

2.3.7. Anexos.

Artefactos generados por el procedimiento:

- Plan de Prueba.
- Cronograma de Prueba.
- Lista de Chequeo.
- Casos de Pruebas.
- Tabla de Ponderación.
- Registro de NC.

2.4. Conclusiones.

En el presente capítulo se ha descrito la propuesta de un procedimiento para realizar la evaluación del rendimiento de las bases de datos que utilicen PostgreSQL como SGBD. Este muestra cómo se realizaran las diferentes actividades que comprenden cada fase de trabajo definida, enfocadas a las pruebas de carga, stress y volumen. Se especifica la actividad que realizará cada trabajador y los artefactos que son generados.

CAPÍTULO 3: VALIDACIÓN DE LA PROPUESTA

Introducción.

En el presente capítulo se realiza la evaluación del procedimiento propuesto a partir del análisis de dos casos de estudio. Para ello se le aplica todo el proceso de pruebas para la evaluación del rendimiento a los productos Generador Dinámico de Reportes (GDR) y al Sistema Integral para la Gestión Estadística (SIGE) de la Línea de Integración de Soluciones de DATEC.

3.1. Generador Dinámico de Reportes (GDR).

El GDR es un producto que permite la generación y administración de informes de forma dinámica. Fue diseñado e implementado a petición de las siguientes instituciones: MENPET (Ministerio de Energía y Petróleo), MIJ (Ministerio del Interior de Justicia), SAREN (Sistema de Registros y Notaria) pertenecientes a Venezuela y ERP (Planificación de Recursos Empresariales) que pertenece a Cuba. Tiene como principales funcionalidades el visor de reportes, diseñador de modelos, administrador de reportes, diseñador de reportes y diseñador de consultas.

3.2. Sistema Integral para la Gestión Estadística (SIGE).

SIGE fue creado a petición de la ONE (Oficina Nacional de Estadística) porque contaban con un software antiguo hecho en MS-DOS y ellos querían un software actualizado con funcionalidades nuevas para gestionar las estadísticas de Cuba. Tiene como principales funcionalidades: SIGE-MED (Módulo de Entrada de Datos), SIGE-MGM (Módulo Generador de Modelo) y SIGE-MRC (Módulo de Registro y Clasificadores).

3.3. Procedimiento de evaluación del rendimiento.

3.3.1. Fase Inicio.

Actividad 1: Planificación de las pruebas.

Se realizarán pruebas de Caja Negra al GDR y a SIGE con el objetivo de medir el rendimiento del producto para someterlo a una evaluación del mismo. Para ello se realiza el Plan de Pruebas en su primera iteración.

- Plan de Pruebas GDR y SIGE.

1. Introducción.

En el presente documento se planifican las pruebas para el GDR y para SIGE, a partir de las funcionalidades de los mismos.

1.1. Alcance.

Este Plan se elabora con el fin de planificar las pruebas al GDR y a SIGE de la Línea de Solución Integral de DATEC.

1.2. Definiciones, Acrónimos y Abreviaturas.

GDR: Generador Dinámico de Reporte.

SIGE: Sistema Integral para la Gestión Estadísticas.

2. Organización del Equipo de Pruebas.

Administrador de Pruebas: Responsable del éxito de la prueba. Dirige las actividades en el proceso de pruebas. (Estudiante: Mariela Valdés Fernández).

Analista de Pruebas: Responsable de identificar los tipos y técnicas de pruebas que se van a realizar. (Estudiante: Mariela Valdés Fernández).

Diseñador de Pruebas: Planifica, diseña y evalúa los resultados de la prueba. (Estudiante: Mariela Valdés Fernández).

Probador: Realiza las pruebas de carga, stress y volumen. (Estudiante: Mariela Valdés Fernández).

3. Especificaciones del Software y Hardware.

Para el GDR y SIGE se necesitan los siguientes requisitos no funcionales de software y hardware:

Requisitos de Software.

Para que las pruebas tengan los resultados más exactos posibles las computadoras donde se realicen las mismas tienen que tener instalado una serie de programas que se listan a continuación:

- SO: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debian 4 GNU/Linux o superior.
- Paquetes: apache2, php5, libapache2-mod-php5, php5-cli, php5-mysql, php5-pgsql, php5-sqlite, php5-xsl, php5-gd.
- PostgreSQL versión 8.3 o superior.
- PGAdmin III.
- PostgreSQL debe estar correctamente configurado para aceptar conexiones vía TCP/IP utilizando el método de autenticación por md5.
- Navegador web basado en Mozilla Firefox 3 o superior

Requisitos de Hardware.

- La computadora tiene que tener un procesador Intel Pentium 4 1.7 GHz o AMD similar, 512 MB de RAM, 40 GB de espacio en disco duro, como requisitos mínimos.

4. Descripción del Plan de Pruebas.

Las pruebas tanto del GDR como de SIGE se van a realizar a la base de datos directamente mediante consultas ejecutándose concurrentemente y cargando las tablas de volúmenes de datos.

5. Casos de Prueba.

Los casos de pruebas están descrito en la Actividad 3: Diseño de casos de pruebas.

6. Estrategia de Prueba.

Este plan contempla pruebas de carga, stress y volumen, donde específicamente se verificó cómo se comporta el rendimiento a altos números de carga, de usuarios y de datos, usando el método de prueba de caja negra, siendo diseñados los casos de pruebas con el fin de verificar el rendimiento de las bases de datos.

6.1. Objetivo.

En esta iteración se comprueba cuánto se demora en recuperarse el sistema ante altas cargas de usuarios y ante altos volúmenes de datos.

6.2. Técnica.

Para realizar las pruebas de caja negra se utilizará la técnica de partición de equivalencia. Esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.

6.3. Proceso.

El proceso de pruebas funciona de la siguiente forma:

Cuando el equipo de prueba elabora el Plan de Prueba, el analista y el diseñador define y diseñan los casos de pruebas. Después el probador ejecuta esas pruebas para llenar el registro de no conformidades para redactar los resultados obtenidos. Estos son enviados al equipo de desarrollo y al líder de proyecto.

6.4. Casos de pruebas.

Los casos de pruebas se pueden ver en la Actividad 3: Diseño de los casos de pruebas.

6.5. Herramientas.

En la presente iteración se utilizaron las herramientas siguientes: JMeter y DataGenerator.

7. Recursos Requeridos.

Roles	Cantidad	Responsabilidad
Administrador de pruebas.	1	Identifica, prioriza e implementa los casos de prueba. Genera el Plan de prueba para la iteración. Negocia o acuerda las pruebas. Genera

		resumen de resultado de prueba. Verifica el progreso y efectividad de las pruebas. Genera informe de no conformidades.
Analista de pruebas.	1	Genera procedimientos de prueba. Identifica los casos prueba. Confecciona la lista de ideas para las pruebas. Genera documento con datos de pruebas.
Diseñador de pruebas.	1	Confecciona casos de pruebas.
Probador.	1	Ejecuta las pruebas. Registra los resultados de las pruebas. Documenta los pedidos de cambio.

8. Calendario y Plazos.

Para saber cuando la aplicación a probar estará disponible para realizar las pruebas ver la plantilla Cronograma de Pruebas.

9. Definición de los Entregables.
   - Plan de Prueba.
   - Cronograma de Prueba.
   - Lista de Chequeo.
   - Casos de Pruebas.
   - Tabla de Ponderación.
   - Registro de NC.

10. Seguimiento y Reporte de Defectos.

Los defectos encontrados se registrarán en la plantilla Registro de No Conformidades. Estas deben ser resueltas en un plazo de cinco (5) días. Luego se envía al equipo de Pruebas nuevamente para una nueva revisión. Este proceso de repetirá solo 3 veces.

11. Aprobación del Plan.

Se realiza una reunión donde están involucrados los miembros del Equipo de Calidad y ha quedado aprobado el Plan.

12. Documentación de Resultados.

Una vez concluido el plan se pasa a elaborar el Cronograma de Pruebas, el cual se muestra a continuación:

- Cronograma de Pruebas.

Módulo	Funcionalidad	Fecha Diseño	Fecha Ejecución
GDR	select * from tbreport where idreport = x; x = valor parametrizado.	28/04/2010	12/05/2010
	select * from tbmodel where idmodel = y; y = valor parametrizado.	28/04/2010	12/05/2010
	select * from tbdatasource;	28/04/2010	12/05/2010
	tbreport	28/04/2010	12/05/2010
	tbmodel	28/04/2010	12/05/2010
	tbdatasource	28/04/2010	12/05/2010
SIGE	SELECT tbcentroinformanteclasificacion.codcentroinformante, tbclasificacion.codclasificacion, tbclasificacion.nombredescriptivo, tbclasificador.codclasificador, tbclasificador.alias, tbclasificador.nombredescriptivo FROM mod_sige.tbcentroinformanteclasificacion, mod_sige.tbclasificacion, mod_sige.tbclasificador WHERE tbcentroinformanteclasificacion.codclasificacion = tbclasificacion.codclasificacion AND tbcentroinformanteclasificacion.codclasificador = tbclasificador.codclasificador AND tbcentroinformanteclasificacion.codcentroinformante = 103;	10/05/2010	12/05/2010
	tbcentroinformante	29/04/2010	12/05/2010

	tbclasificacion	29/04/2010	12/05/2010
	tbclasificador	29/04/2010	12/05/2010
	tbcentroinformanteclasificacion	29/04/2010	12/05/2010

Actividad 2: Propuesta de instrumento de medición: Lista de Chequeo.

La lista de chequeo a aplicar es la que se define en el capítulo anterior.

Actividad 3: Diseño de casos de pruebas.

Condiciones de ejecución

- El servidor de base de datos tiene que estar disponible.
- La velocidad de la red tiene que estar conectada y a 100.0 Mbps

CP Carga y Stress

Consulta a ejecutar	Carga de trabajo	Resultados (seg)
GDR		
select * from tbreport where idreport = x; x = valor parametrizado.	100	
select * from tbmodel where idmodel = y; y = valor parametrizado.	100	
select * from tbdatasource;	100	
SIGE		
SELECT tbcentroinformanteclasificacion.codcentroinformante, tbclasificacion.codclasificacion, tbclasificacion.nombredescriptivo, tbclasificador.codclasificador, tbclasificador.alias, tbclasificador.nombredescriptivo FROM mod_sige.tbcentroinformanteclasificacion, mod_sige.tbclasificacion, mod_sige.tbclasificador WHERE tbcentroinformanteclasificacion.codclasificacion = tbclasificacion.codclasificacion AND	100	

tbcentroinformanteclasificacion.codclasificador = tbclasificador.codclasificador AND tbcentroinformanteclasificacion.codcentroinformante = 103;		

CP Volumen

Tabla	Cantidad de registros	Generados a la BD (seg)
GDR		
tbreport	1000	
tbmodel	1000	
tbdatasource	1000	
SIGE		
tbcentroinformante	1000	
tbclasificacion	1000	
tbclasificador	1000	
tbcentroinformanteclasificacion	1000	

3.3.2. Fase Intermedia.

Actividad 1: Aplicación de la Lista de Chequeo.

La lista de chequeo fue aplicada a los arquitectos y los diseñadores de base de datos que conformaron el equipo de desarrollo de los productos de GDR y SIGE.

Actividad 2: Tabulación de la Lista de Chequeo.

GDR

Característica de Calidad: Rendimiento				
Sub-características	**Valor Asignado**			
	1	**2**	**3**	**4**
Velocidad de Procesamiento. (VP)			X	
Tiempo de Respuesta. (TR)			X	
Consumo de Recursos. (CR)			X	
Rendimiento Efectivo Total. (RET)			X	
Eficacia. (E)			X	

Resultados de la Presencia de las Sub-Características (PSC) en cada etapa del proceso de prueba, según la característica de calidad: Rendimiento.

PSC = ∑ X / Y

PSC (VP) = 12/7 = 1.7 (Presente deficientemente)

PSC (TR) = 9/5 = 1.8 (Presente deficientemente)

PSC (CR) = 24/12 = 2 (Presente deficientemente)

PSC (RET) = 15/7 = 2.1 (Presente)

PSC (E) = 4/3 = 1.3 (Presente deficientemente)

Resultados de la Presencia de la Característica de Calidad Rendimiento (PCCR) considerando la importancia dada por los involucrados.

PP = PSC * VA

PP (VP) = 1.7 * 3 = 5.1

PP (TR) = 1.8 * 3 = 5.4

PP (CR) = 2 * 3 = 6

PP (RET) = 2.1 * 3 = 6.3

PP (E) = 1.3 * 3 = 3.9

PCCR = ∑PP / ∑VA

PCCR = 26.7/15

PCCR = 1.7 = 42.5%. → No está presente

SIGE

Característica de Calidad: Rendimiento				
Sub-características	**Valor Asignado**			
	1	**2**	**3**	**4**
Velocidad de Procesamiento. (VP)	X			
Tiempo de Respuesta. (TR)	X			
Consumo de Recursos. (CR)			X	
Rendimiento Efectivo Total. (RET)	X			
Eficacia. (E)	X			

Resultados de la Presencia de las Sub-Características (PSC) en cada etapa del proceso de prueba, según la característica de calidad: Rendimiento.

PSC = ∑ X / Y

PSC (VP) = 11/7 = 1.5 (Presente deficientemente)

PSC (TR) = 5/5 = 1 (No está presente en esta etapa)

PSC (CR) = 20/12 = 1.6 (Presente deficientemente)

PSC (RET) = 14/7 = (Presente deficientemente)

PSC (E) = 3/3 = 1 (No está presente en esta etapa)

Resultados de la Presencia de la Característica de Calidad Rendimiento (PCCR) considerando la importancia dada por los involucrados.

PP = PSC * VA

PP (VP) = 1.5 * 1 = 1.5

PP (TR) = 1 * 1 = 1

PP (CR) = 1.6 * 3 = 4.8

PP (RET) = 2 * 1 = 2

PP (E) = 1 * 1 = 1

PCCR = ∑PP / ∑VA

PCCR = 10.3/7

PCCR = 1.4 = 35%. → No está presente

Actividad 3: Montaje del entorno de prueba.

Para las pruebas se utilizaron dos computadoras: una con el Sistema Operativo Ubuntu 10.4 y la otra con el Sistema Operativo Ubuntu-Server 10.4; todos los demás requisitos se corresponden con los descritos en el Plan de Prueba y se le agregan las herramientas JMeter y DataGenerator.

Actividad 4: Ejecución de las pruebas.

Las pruebas se realizaron para el GDR y SIGE en las dos computadoras, donde se obtuvieron los siguientes resultados:

GDR (PC con Ubuntu 10.4)

Las pruebas de carga y stress después de ejecutar las tres consultas concurrentemente se obtuvo un resultado de: throughput = 41.095.89/min. Throughput es la variable que da el JMeter que representa el rendimiento, capacidad de procesamiento y la velocidad, volumen de trabajo o de información que fluye a través de un sistema y su resultado representa la cantidad de datos o de registros por minuto, es la tasa promedio de mensajes entregados satisfactoriamente o la tasa de operaciones por unidad de tiempo.

En las pruebas de volumen se obtuvieron los siguientes resultados:

Tabla	Cantidad de registros	Generados a la BD (seg)
GDR		
tbreport	1000	12
tbmodel	1000	12

tbdatasource	1000	14

GDR (PC con Ubuntu-Server 10.4)

Las pruebas de carga y stress después de ejecutar las tres consultas concurrentemente se obtuvo un resultado de: throughput = 44.099.78/min.

En las pruebas de volumen se obtuvieron los siguientes resultados:

Tabla	Cantidad de registros	Generados a la BD (seg)
GDR		
tbreport	1000	10
tbmodel	1000	11
tbdatasource	1000	10

SIGE (PC con Ubuntu 10.4)

Las pruebas de carga y stress después de ejecutar la consulta se obtuvo un resultado de: throughput = 12.526.096/min.

En las pruebas de volumen se obtuvieron los siguientes resultados:

Tabla	Cantidad de registros	Generados a la BD (seg)
SIGE		
tbcentroinformante	1000	14
tbclasificacion	1000	9
tbclasificador	1000	8
tbcentroinformanteclasificacion	1000	10

SIGE (PC con Ubuntu-Server 10.4)

Las pruebas de carga y stress después de ejecutar la consulta se obtuvo un resultado de: throughput = 15.797.872/min.

En las pruebas de volumen se obtuvieron los siguientes resultados:

Tabla	Cantidad de registros	Generados a la BD (seg)
SIGE		
tbcentroinformante	1000	12
tbclasificacion	1000	8
tbclasificador	1000	6

tbcentroinformanteclasificacion	1000	9

En la ejecución de las pruebas no se obtuvieron NC, por lo tanto, el Registro de NC no se llenó.

3.3.3. Fase Final.

Actividad 1: Comportamiento de la Lista de Chequeo.

Después de haber analizado los resultados de la aplicación de la lista de chequeo se llega a la conclusión que el atributo rendimiento no está presente en el desarrollo de los productos analizados en los casos de pruebas (GDR y SIGE). Con los por cientos obtenidos teniendo en cuenta que el 100% representa 4 que es el valor más significativo se evidencia que este atributo tiene un bajo nivel de aceptabilidad puesto a que para alcanzar el nivel de aceptabilidad tiene que alcanzar el 75%. En caso que no alcance el valor mínimo de presencia, que es el caso, se recomienda que sea revisada la característica, es decir, el rendimiento en toda la etapa de desarrollo del software. Una vez más se demuestra que el rendimiento no se tiene en cuenta en el desarrollo de los proyectos.

Actividad 2: Análisis de los resultados de las pruebas.

Después de haber concluido con la realización de las pruebas de carga, stress y volumen se puede concluir que las pruebas hechas en la computadora con el sistema operativo Ubuntu-Server 10.4 son mucho más rápidas que las realizadas en la otra computadora con el sistema operativo Ubuntu 10.4. Las pruebas no se realizaron con otro sistema operativo como Debian debido a que los servidores están montados sobre Ubuntu. A continuación se representa estos resultados gráficamente para una mejor visión.

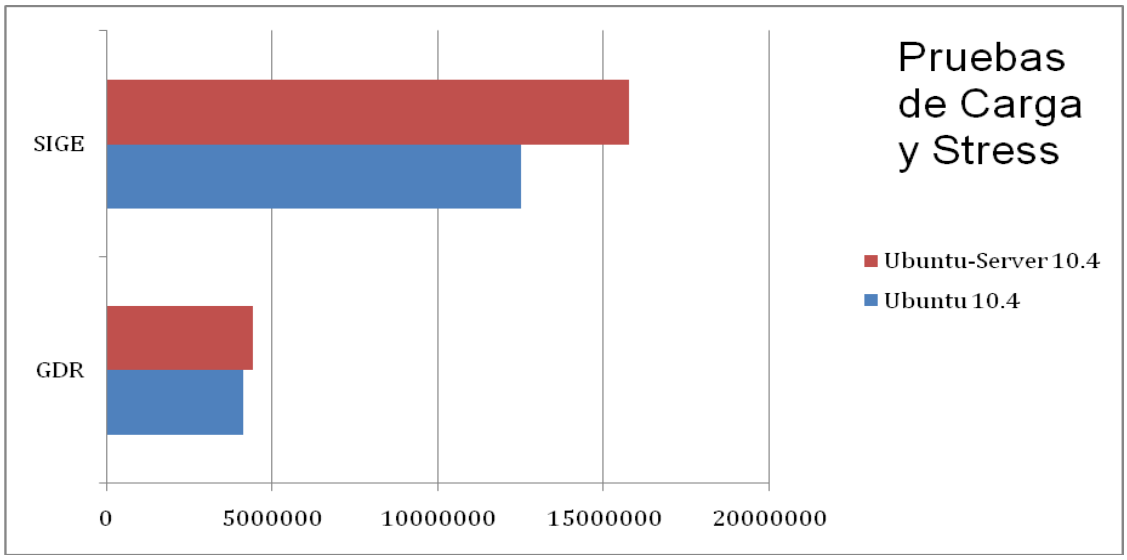


Figura 13: Resultados de las pruebas de carga y stress de GDR y SIGE.

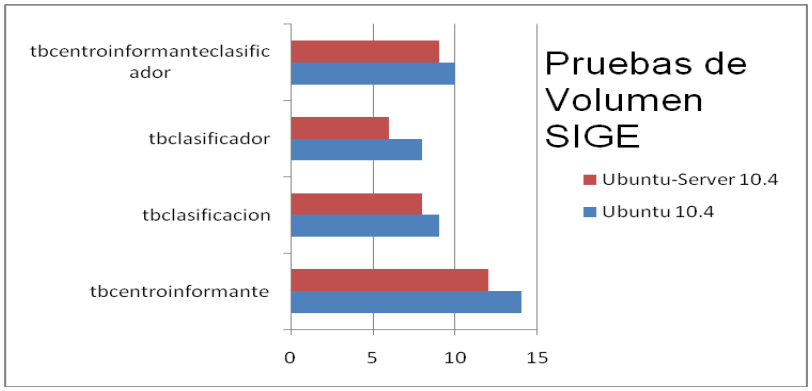


Figura 14: Resultados de las pruebas de volumen de SIGE.

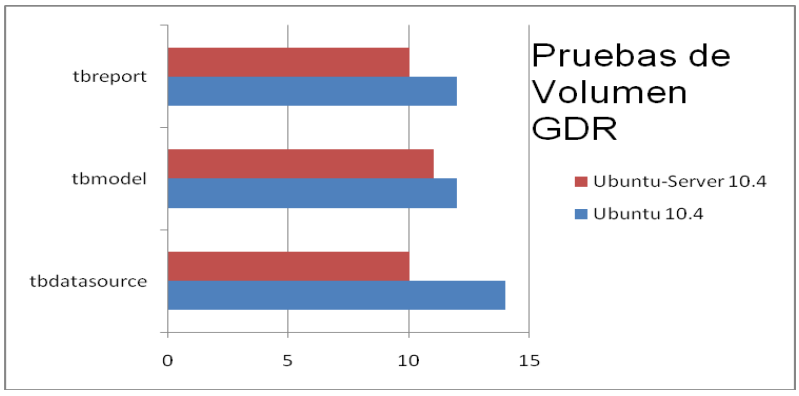


Figura 15: Resultados de las pruebas de volumen de GDR.

Después del proceso de pruebas hay que hacer una evaluación del mismo en cuanto a la asistencia de los probadores, calidad del trabajo y cumplimiento del cronograma. La evaluación que se le da al proceso de prueba es buena porque se obtuvieron los resultados exactos de la situación real de los casos de pruebas, además el entorno de prueba no presentó ningún imprevisto y las condiciones de ejecución fueron las que tenían que ser.

3.4. Conclusiones.

En el presente capítulo se ha validado la propuesta del procedimiento para la evaluación del rendimiento de las bases de datos basadas en PostgreSQL para software de gestión. Para la validación se utilizó dos productos desarrollados en la Línea de Integración de Soluciones de DATEC. Con estos dos casos de estudio se evidenció que el rendimiento es un atributo de calidad a tener en cuenta a la hora del desarrollo de un software.

CONCLUSIONES

La investigación realizada arrojó las siguientes conclusiones:

- Las encuestas realizadas demostraron que en la universidad y en DATEC no existen experiencias en la realización de pruebas de software a las bases de datos en función del rendimiento.
- Se propuso un procedimiento para la evaluación del rendimiento de las bases de datos basadas en PostgreSQL para software de gestión. El procedimiento propone un conjunto de actividades organizadas que generan una serie de artefactos que ayudan a la evaluación del rendimiento.
- La validación del procedimiento se realizó a dos casos de estudio, a los productos de GDR y SIGE de la Línea de Integración de Soluciones de DATEC y arrojó como resultado que el rendimiento tiene un nivel de aceptabilidad muy bajo.

Por tanto, la investigación dio cumplimiento a los objetivos y a las tareas propuestas para ayudar al centro DATEC específicamente a la Línea de Integración de Soluciones a desarrollar nuevos proyectos teniendo en cuenta la evaluación del rendimiento, para obtener un optimizado diseño del sistema informático.

RECOMENDACIONES

Una vez cumplido con el objetivo general propuesto, se recomienda:

- Generalizar el procedimiento de forma tal que se aplique en todos los proyectos que se desarrollen en la Línea de Integración de Soluciones de DATEC.
- Impartir cursos para la superación del equipo de desarrollo de la Línea de Integración de Soluciones de DATEC con el objetivo de que conozcan el atributo de calidad rendimiento, entiendan cuál será su papel dentro del proceso, su importancia, y las actividades que se realizarán, profundizando en cómo utilizar este procedimiento.

BIBLIOGRAFÍA

1- rojas@yahoo.es, "Comparación entre sistemas de gestión de bases de datos(SGBD) bajo licenciamiento libre y comercial", http://www.monografias.com/trabajos29/comparacion-sistemas/comparacion-sistemas.shtml, (17/11/2009)

2- Jaime Casanova, "PostgreSQL ha sido liberado", http://archives.postgresql.org/pgsql-es-fomento/2009-07/msg00000.php, (24/11/2009).

3- Tesis de doctorado de Edgar Henry Caballero Rúa. Título: Mejora de la calidad del software en el entorno de microempresas de TI. Caballero modelos de calidad.pdf. (2006-2007).

4- Erika Camacho, Fabio Cardeso y Gabriel Núñez. Abril 2004. Guía Arquitectura v.2.pdf. Arquitectura de Software. Guía de estudio.

5- ISO/IEC (1998). Information Technology Software product Quality Part 1: Quality Model. Obtenido de: http://www.usability.serco.com/trump/resources/standards.htm#9126-1

6- Lic. Cecilia Palazzolo. 2005. Calidad De SW- diap.pdf. Calidad de Software. Herramientas de Software.

7- Jonás A. Montila C., Ph.D. IEEE Member, 2006. Desarrollo de Software Basado en Líneas de Productos de Software.

8- Sitio de PostgreSQL: http://www.postgresql-es.org

9- Muñoz, G., Granja-Álvarez, J., Sempere, C. 2006. Sistema mixto MDA – Factorías de Software: Un caso práctico. CISTI 2006, Volume II.

10- Stephen H. Kan, Metrics and Models in Software Quality Engineering, 2002, Addison-Wesley.

11- Alejandro Gómez, Gustavo Muñoz, Juan Carlos Granja. Evaluación de los niveles de calidad en las transformaciones de modelos basados en el estudio de factores de éxito. Obtenida de: http://www.sistedes.es/sistedes/pdf/2007/JISBD-07-gomez-calidad.pdf

12- Sitio del DataGenerator: http://www.generatedata.com

13- Consorcio SIU. Monitoreando y midiendo el rendimiento de un servidor Postgresql en Sistemas Windows, Linux y Solaris. Parte 1. Obtenido de: http://www.siu.edu.ar/documentos/Monitoreando%20Servidor%20Postgresql.pdf

14- Ángel Cervera. El modelo de mccall como aplicación de la calidad a la revisión del software de gestión empresarial. Obtenido de: http://www.monografias.com/trabajos5/call/call.shtml

15- Marc Gibert Ginestá y Oscar Pérez Mora. Bases de datos PostgreSQL. Obtenido de: http://ggomez.files.wordpress.com/2008/09/postgresql.pdf

16- Ingeniería de Software. Pruebas. Obtenido de: http://delfin.mxl.uabc.mx/~angelica/Pruebas.pdf

17- Isabel Blank, Larissa Herrera y Miguel Ortiz. Pruebas de Funcionalidad. Obtenido de: http://carolina.terna.net/ingsw3/datos/Pruebas_Funcionales.pdf

18- Betzabeth Pereira, Farid Ayaach y Henry Quintero. Métricas de calidad de software. Obtenido de: http://subversion.assembla.com

19- I. Barrientos, J. Beites. Nagios, un sistema de monitorización de servicios de red. Obtenido de: http://www.asturlinux.org/archivos/jornadas2006/ponencias/Nagios/nagios.pdf

20- R.S. Pressman. Ingeniería de Software. Un enfoque práctico. 4ta Edición.

REFERENCIAS BIBLIOGRÁFICAS

(1) (Pressman 1998) R.S. Pressman. Ingeniería de Software. Un enfoque práctico. 4ta Edición.

(2) Caballero, E.H (2006). Doctorado: Mejora de la calidad del software en el entorno de microempresas de TI. Universidad Politécnica de Madrid.

(3) Camacho, E., Cardeso, F. y Nuñez, G. (2004). Arquitecturas de software. Guía de estudio.

(4) Dromey, G. (1996). Cornering the Chimena. IEEE Software. Vol 13, Nro. 1. URL: http://www.computer.org/software/so1996s1033abs.htm

(5) Krueger, Ch. (2006). Introducion to Software Product Lines. URL: www.softwareproductline.com

(6) Gomma, H. (2004). Designing Software Product Lines witch UML: From Use Cases to pattern-based Software Architectures. Adisson Wesley.

(7) Montila, J. (2006). Desarrollo de software basado en líneas de productos de software.

(8) Torrealba, R. y otros (2007). Software de Gestión y Programación. URL: http://ipdatos.files.wordpress.com/2007/06/ipd_exp_05.pdf.

(9) (En línea) CAVSI (Computer Audio Video Systems Integrator). URL: http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/

(10) (En línea) Sobre PostgreSQL. URL: http://www.postgresql-es.org

(11) Acuña, CJ. Pruebas de Software. Ingeniería del Software I. Universidad Rey Juan Carlos. URL: http://kybele.escet.urjc.es/Documentos/ISI/Pruebas%20de%20Software.pdf.

(12) Pruebas de Software. URL: http://lsi.ugr.es/~ig1/docis/pruso.pdf.

(13) Vásquez, M. Prueba de Software y Seguridad en entornos distribuidos. Editorial Prensa Técnica S.L. España. URL: http://pdf.rincondelvago.com/prueba-de-software.html

(14) (En línea). Seleccionar un tipo de prueba. Sitio: msdn. URL: http://msdn.microsoft.com/es-es/library/ms182415%28VS.80%29.aspx

(15) (curos 2006-2007). Conferencia: Flujo de trabajo Prueba. Ingeniería de Software II. Departamento de Ingeniería y Gestión de Software. UCI.

(16) (En línea). Pruebas. Ingeniería de Software. URL: http://delfin.mxl.uabc.mx/~angelica/Pruebas.pdf

(17) Pressman R. (2002) Ingeniería de Software. Un enfoque práctico. 5ta Edición. Mc Graw Hill.

(18) (En línea) ALS (Software Lifecycle Optimization). URL: http://www.als-es.com/home.php?location=herramientas/entorno-pruebas

(19) (En línea) Datagenerator.com. URL: http://www.generatedata.com

(20) Barrientos, I y Beites, J (2006). Nagios. Un sistema de monitorización de servicios de red.

(21) (En línea) Up to Down. URL: http://jmeter.uptodown.com/

(22) Tristain, M. (2009) ¿Qué es selenium? URL: http://softwarequality.bligoo.com/content/view/434556/Que-es-Selenium.html

(23) Pere, M. (2005). Glosario de Tecnología Educativa. URL: www.pangea.org

(24) (En línea) Guía para la elaboración de procedimientos. URL: http://www.euatm.upm.es/auditoria/PROCED%20DE%20PROCED.pdf.

ANEXOS

Anexo 1: Encuesta para los arquitectos

Encuesta para los arquitectos.

Partiendo de la definición de rendimiento: "Capacidad del producto de software para proporcionar apropiados tiempos de respuesta y procesamiento haciendo un uso adecuado de los recursos de hardware y software", responda marcando con una cruz las siguientes preguntas:

1. ¿Existe alguna herramienta de prueba para el análisis del rendimiento de Base de Datos? __Existe __No Existe __Desconoce el tema __Conoce el tema pero no las utiliza.
2. ¿Conoce y aplica a la Base de Datos, los tipos de pruebas que se listan a continuación?, Carga (__Sí. __No). Estrés (__Sí. __No). Tensión (__Sí. __No). Volumen (__Sí. __No). Rendimiento (__Sí. __No).
3. ¿Tiene conocimiento de técnicas de evaluación del rendimiento? (Métodos y herramientas para estimar los índices de prestaciones: Monitorización del sistema real; Referenciación (benchmarking) con sistemas reales o modelados; Modelado). __Parcial. __Total. __Nula.
4. En caso de conocer las técnicas para la evaluación del rendimiento, diga si las aplica. __Sí. __No.
5. ¿En el servidor de Bases de Datos se monitorea el % de uso del CPU ejecutando una consulta a la Base de Datos? __Sí. __No.
6. ¿En el servidor de Bases de Datos se monitorea el % de uso del disco duro ejecutando una consulta a la BD? __Sí. __No.
7. ¿En el servidor de Bases de Datos se monitorea el aumento de consumo de memoria RAM ejecutando una consulta a la BD? __Sí. __No.
8. ¿Conoce la velocidad de la red? __Sí. __No.
9. ¿Conoce el sistema de ficheros que tiene el Sistema Operativo en el servidor de Bases de Datos? __Sí. __No.
10. ¿El sistema de fichero del Sistema Operativo que utiliza el servidor de Bases de Datos es adecuado para Bases de Datos? __Sí. __No. __No sé.
11. Para el servidor donde se encuentra físicamente la Base de Datos, ¿se utiliza un Sistema Operativo versión servidor? __Sí. __No.
12. ¿Se utiliza el pool de conexiones en el servidor de Base de Datos? __Sí. __No.

13. ¿Se tuvo en cuenta el análisis de la complejidad del negocio, para la selección del SGBD a utilizar? __Sí. __No.
14. ¿Para el SGBD PostgreSQL se configura el uso de la caché? __Sí. __No.
15. ¿Para el SGBD PostgreSQL se configura la cantidad de conexiones máximas? __Sí. __No.
16. ¿Se realiza el particionamiento de la Base de Datos? (Poner tablas o esquemas de la BD almacenada físicamente en discos duros más rápidos) __Sí. __No.

Datos del Encuestado

Nombre y apellidos: _____.
Especialidad de Graduado: _____.
Rol que ocupa: _____.
Experiencia desempeñando ese rol: _____.
Facultad: _____.
Proyecto: _____.

Anexo 2: Encuesta para los diseñadores de base de datos.

Encuesta para los Diseñadores de Bases de Datos.

1. ¿Cuál es el nivel de utilización de índices en el diseño de la Base de Datos?

   __Alto. __Medio. __Bajo.

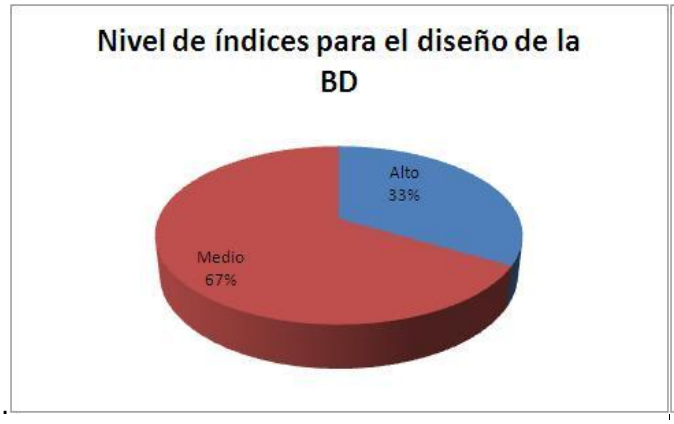
2. ¿Qué nivel de normalización tiene la BD?

   __1FN. __2FN. __3FN. __FNBC. __Otras.

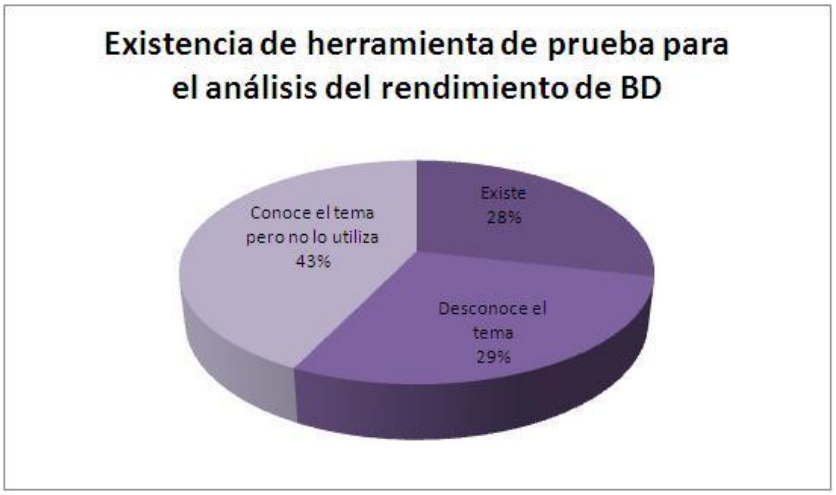
Datos del Encuestado

Nombre y apellidos: _____.
Especialidad de Graduado: _____.
Rol que ocupa: _____.
Experiencia desempeñando ese rol: _____.
Facultad: _____.
Proyecto: _____.

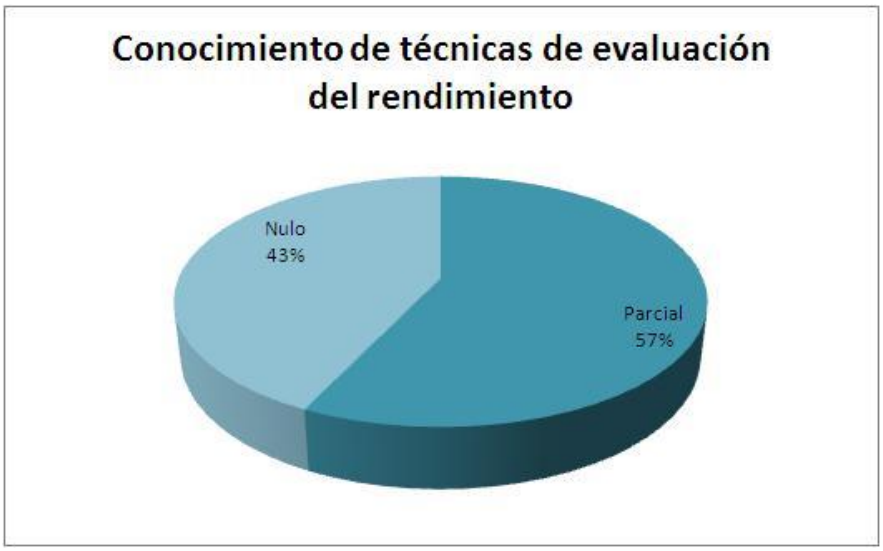
Anexo 3: Resultados de las encuestas graficados

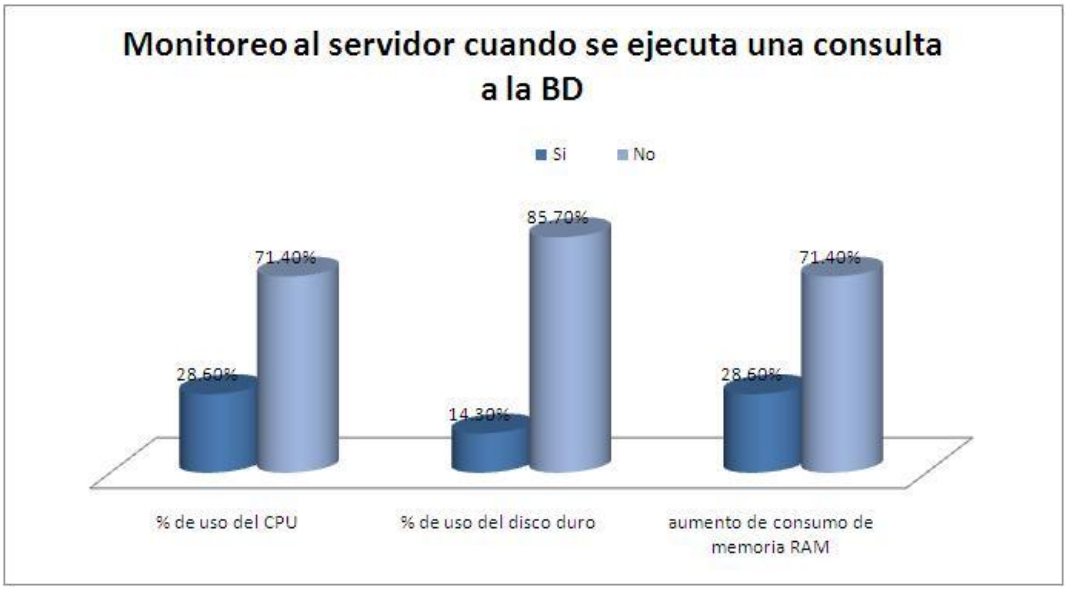

Nivel de índices para el diseño de la BD
Alto
33%
Medio
67%


Nivel de normalización
2FN
33%
3FN
67%

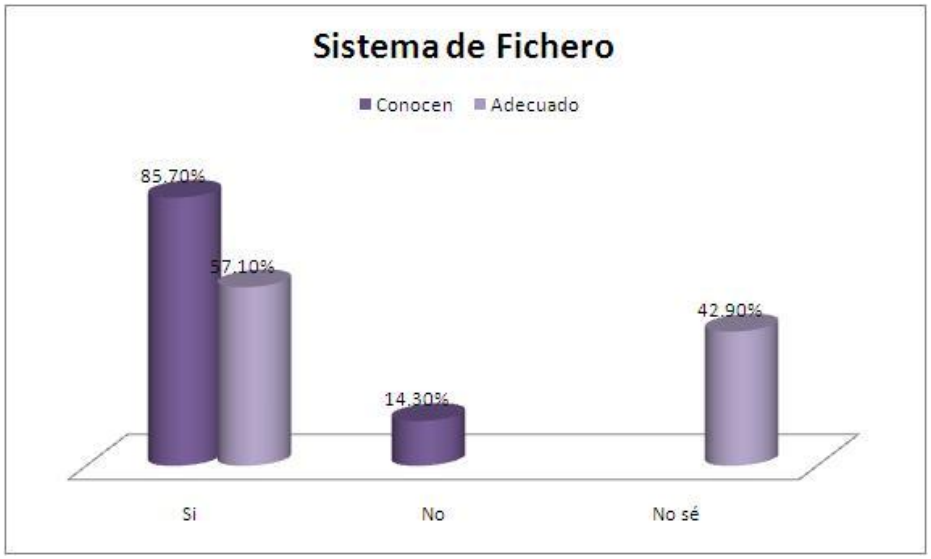

Existencia de herramienta de prueba para el análisis del rendimiento de BD
Existe
28%
Desconoce el tema
29%
Conoce el tema pero no lo utiliza
43%

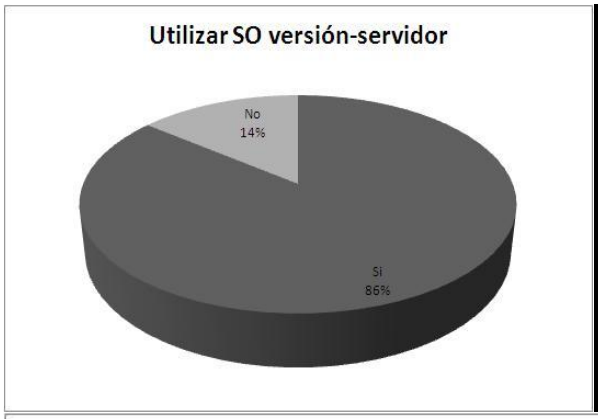

Conocimiento de tipos de pruebas que le hacen a las BD
Si
No
50% 50%
50% 50%
0%
100%
0%
100%
28.60%
71.40%
Carga
Stress
Tensión
Volumen
Rendimiento


Conocimiento de técnicas de evaluación del rendimiento
Nulo
43%
Parcial
57%

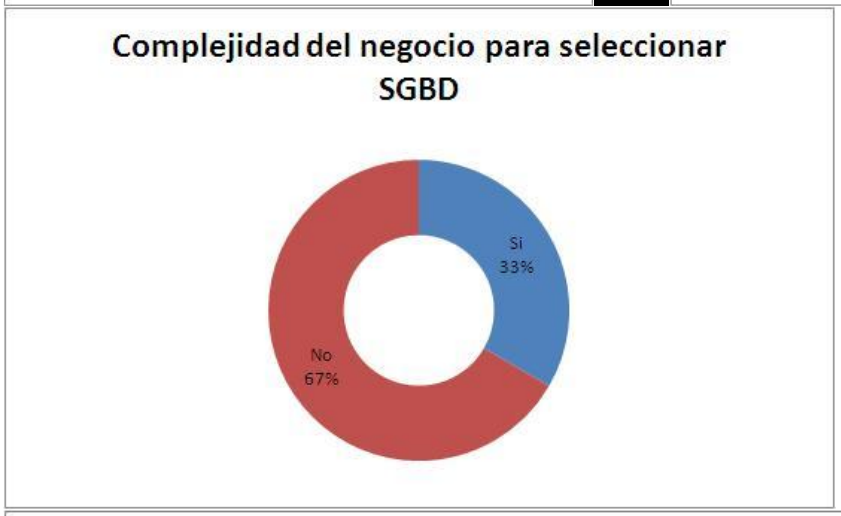

Monitoreo al servidor cuando se ejecuta una consulta a la BD
Si
No
28.60%
71.40%
14.30%
85.70%
28.60%
71.40%
% de uso del CPU
% de uso del disco duro
aumento de consumo de memoria RAM

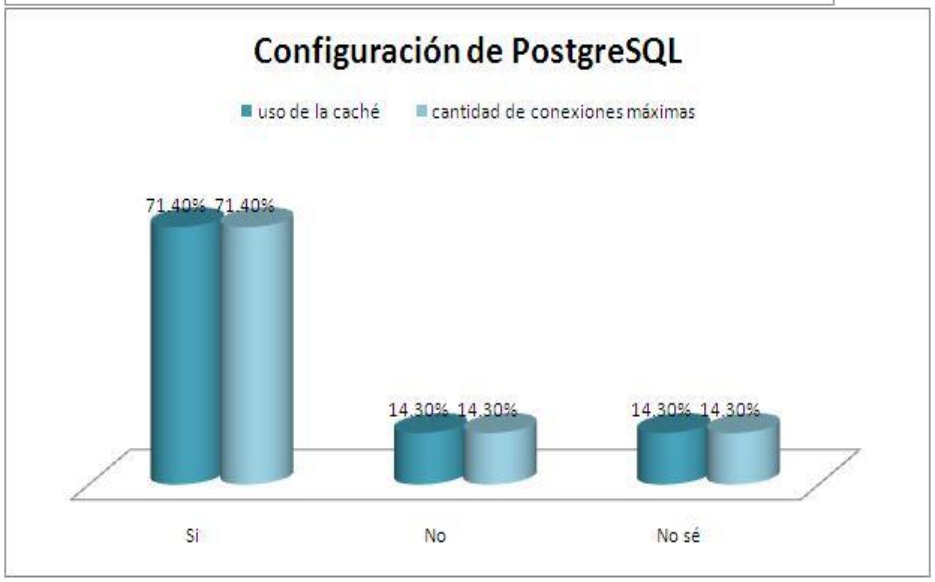

Conocimiento de la velocidad de la red
No
29%
Si
71%

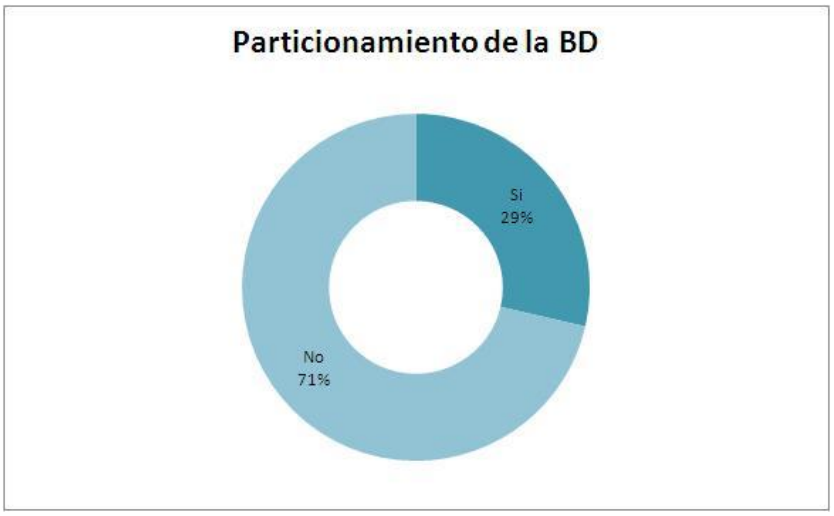

Sistema de Fichero
Conocen
Adecuado
85,70%
57,10%
14,30%
42,90%
Si
No
No sé


Utilizar SO versión-servidor
No
14%
Si
86%


Utilizar el pool de conexiones
No sé
29%
Si
71%


Complejidad del negocio para seleccionar SGBD
Si
33%
No
67%


Configuración de PostgreSQL
uso de la caché
cantidad de conexiones máximas
71,40%
71,40%
14,30%
14,30%
14,30%
14,30%
Si
No
No sé



Anexo 4: Plan de Prueba.

Plan de Pruebas

1. Introducción
1.1. Alcance
1.2. Definiciones, Acrónimos y Abreviaturas
1.3. Referencias
2. Organización del Equipo de Pruebas.
3. Especificaciones del Software y Hardware.
4. Descripción del Plan de Pruebas.
5. Casos de Prueba.
6. Estrategia de Prueba.
6.1. Objetivo.
6.2. Técnica.
6.3. Entorno de Prueba.
6.4. Proceso.
6.5. Casos de Prueba.
6.6. Herramientas.
7. Recursos Requeridos.

Roles	Cantidad	Responsabilidad

8. Calendario y Plazos.
9. Definición de los Entregables.
10. Seguimiento y Reporte de Defectos.
11. Aprobación del Plan.
12. Documentación de Resultados.

Anexo 5: Cronograma de Prueba.

Módulo	Funcionalidad	Fecha Diseño	Fecha Ejecución

Anexo 6: Diseño de Casos de Pruebas.

CP Carga y Stress

Consulta a ejecutar	Carga de trabajo	Resultados (seg)

CP Volumen

Tabla	Cantidad de registros	Generados a la BD (seg)

Anexo 7: Registro de No Conformidades.

Elemento	No	No Conformidad	Aspecto Correspondiente	Etapa de Detección	Significativa	No Significativa	Recomendación	Estado NC	Respuesta del Equipo de Desarrollo

GLOSARIO

ACID: Atomicity, Consistency, Isolation, Durability

RDBMS: Sistemas de Gestión de Bases de Datos Relacionales.

SQL: Lenguaje de Consultas Estructurado

ORDBMS: Object Relational DataBase Management System.

HOT: Heap Only Tuples.

LPS: Líneas de productos de software

DATEC: Centro de Tecnologías de Gestión de Datos.

CP: Casos de Pruebas.

NC: No Conformidades.

PSC: Presencia de la Sub-Característica.

PCCR: Presencia de la Característica de Calidad Rendimiento.

Ponderación: Peso que atribuye a una variable en función de la importancia que tenga dentro de un conjunto.