Universidad de las Ciencias Informáticas

Facultad 2

Estrategia de clúster de base de datos para el Sistema de Gestión Hospitalaria del Centro de Informática Médica

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

Autores: Yanislei Matos Faure

Arel Luis Fleitas Cruz

Tutor: Ing. Yusmilaidi Causse Ascanio

Co-Tutor: MSc. Yovannys Sánchez Corales

Curso: 2013-2014

“Año 56 de la Revolución”

"Un científico debe tomarse la libertad de plantear cualquier cuestión, de dudar de cualquier afirmación, de corregir errores."

Robert Oppenheimer

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 _____.

Yanislei Matos Faure

_____

Firma del Autor

Arel Luis Fleitas Cruz

_____

Firma del Autor

Ing. Yusmilaidi Causse Ascanio

_____

Firma del Tutor

MSc. Yovannys Sánchez Corales

_____

Firma del Co-Tutor

Ing. Yusmilaidi Causse Ascanio: Graduado de ingeniero en Ciencias Informáticas en el año 2008, en la Universidad de las Ciencias Informáticas (UCI), incorporada a trabajar en la misma en el año 2010, como especialista general, actualmente como profesor instructor impartiendo la asignatura de gestión de software. Ha tutorado otras tesis y se ha desempeñado también en tribunales de tesis y como oponente.

Correo electrónico: ycausse@uci.cu

MSc. Yovannys Sánchez Corales: Profesor graduado de Ingeniero en Ciencias Informáticas en el año 2005 en la CUJAE. Categoría docente: Asistente. Ha impartido las asignaturas Programación III, Inteligencia Artificial, Arquitectura de Software y Administración de Servidores Linux. Se desempeña como arquitecto principal del Sistema Integral de Atención Primaria para la Salud (alas SIAPS) perteneciente al Centro de Informática Médica (CESIM).

Correo electrónico: yscorales@uci.cu

Agradecimientos Arel

A mi tía mili, mi otra madre, aunque estés lejos te siento muy cerca.

A cabeza, sin tu ayuda no hubiera sido posible este viaje.

A mi mamá por ser mi faro, mi guía, mi luz y mi oscuridad, ahora me toca a mí solo.

A mi papá por enseñarme a ser un hombre de bien, nunca, nunca te defraudaré.

Los Amo.

A mi primo Luisi por ser mi hermano.

A mi abuela por darme la sabiduría de la vida.

A mi primo Brailis por quererme tanto.

A Babi por darme su tiempo.

A mi primo David por aconsejarme.

A Neni por tenderme la mano cuando más lo he necesitado, siempre.

A Miriam por ser tan buena conmigo, te quiero mucho.

A mi tía Izqui, nunca salgas de mi vida.

A la Guajira, (eres lo mejor que puedo pedir para mi hermano) no puedes faltar en esta página.

A todas las personas que han contribuido a mi desdicha en estos 5 años, a esos, que son los que más me han enseñado.

Agradecimientos Yanislei

A mi abuela porque la amo con locura y porque es uno de los cosas más grandes que tengo en la vida.

A mi mamá por darme la vida, porque la amo, porque es mi todo, por ser madre y padre, y por su dedicación estos 24 años.

A mis tías Marie y Cari porque han sido como mis madres, por todo el amor que me han dado y porque yo las amo.

A mi primito Ery porque es mi tesorito y porque es una de las personas que más amo en el mundo.

A mi prima Norvis porque sé que ve en mí la hija que no tuvo y por todo lo que ha hecho por mí…Te quiero mucho.

A mi papa porque a pesar de sus defectos también ha hecho mucho por mí y porque sé que me quiere.

A José y familia por todas las cosas que han hecho por mí en estos 6 años.

A su mamá Mireyda porque ha sido más que una madre para mí, por sus consejos y por darme tanto amor.

A sus tías Migdalia y Marlenis porque también han sido como mis madres y porque las quiero mucho.

A su abuelo Antonio, su papa José Ángel y su padrastro Raudel que sé que me adoran y yo lso quiero mucho.

A sus tíos Mario, Marvin, Miroel, sus primos Yudi, Beti, Yesica, Andresito, en fin, a todos por tanto amor y cariño.

A José que gracias a que lo conocí, llegue a formar parte de esta familia a la que adoro, también porque el llego a ser mi amigo, mi compañero, mi consejero y por todo lo que hiso por mí, quiero que sepa que te estoy muy agradecida.

A Esteban porque en tan poco tiempo me ha demostrado que soy muy especial para él y que me quiere mucho, gracias por todas las cosas que has hecho por mí y por convertir mis problemas en tuyos

A mis amistades Claudia, Daysi, Susej, Yaneisi, Dami, Tati, Lisi, Liset, Amanda, Yarianna, Chang, Digmari, Heydi, Marcos, Yosuani, Beti, Rodrigo, los quiero mucho.

A mis tutores y mi compañero de tesis porque fueron mi todo en estos meses de preparación para esta exposición. Muchas gracias.

A mi hermano Juan Carlos y mi hermana Dunia. Ustedes son la razón de mi vida.

Dedicatoria Yanislei

A mi abuela, mi mamá, mis tías Marie y Cari, y a mi primito Ery porque son la luz de mi vida, porque si me falta su presencia, amor, ternura, comprensión y sus consejos no sabría qué hacer con mi vida… Quiero que sepan que los amo mucho y que son mi razón de existir.

Uno de los objetivos de los sistemas informáticos es mantener la disponibilidad de la información y posibilitar su rápido acceso. Los sistemas informáticos para la salud no están exentos de este fenómeno, al contrario, por lo crítico de sus servicios, la disponibilidad y el rendimiento son pilares de confianza especialmente en este sector. La investigación describe, fundamenta y verifica la construcción de una estrategia de clúster de base de datos sobre el sistema gestor PostgreSQL que garantiza la disponibilidad y el rendimiento del Sistema Informático para la Gestión Hospitalaria del Centro de Informática Médica. Para su construcción se utiliza Pgpool como herramienta de balanceo de carga, de réplica de datos, pool de conexiones y Heartbeat como herramienta de disponibilidad. El diseño físico es de dos nodos sobre una arquitectura maestro-esclavo que soporta fallas.

Palabras clave: PostgreSQL, clúster, base de datos, Pgpool, Heartbeat, clúster de base de datos.

One of the goals of computer systems is to maintain the availability of information and enable quick access. The health systems are not expenses of this phenomenon, unlike so critical of their services, the availability and performance are the pillars of trust especially in this sector. The research describes and verifies a strategy cluster database on the PostgreSQL system manager guaranteeing the availability and increasing the performance of the Hospital Information System for Management Center on Medical Informatics. For its construction Pgpool is used as load balancing, pool connection and replication tool, with in Heartbeat as high availability disposure tool. The physical design is two nodes on a master-slave architecture where performed failover techniques.

Keywords: PostgreSQL, cluster, database, Heartbeat, Pgpool, database cluster.

Introducción

Las Tecnologías de la Información y las Comunicaciones (TIC) constituyen el resultado de las posibilidades creadas por la humanidad en torno a la digitalización de datos, productos, servicios y procesos. Forman parte de la cultura tecnológica y son ampliamente utilizadas en los sectores sociales (Filippi, 2009), especialmente en la salud por la acumulación de información y la importancia de los servicios que ofrece.

Como parte de las TIC, la tecnología de clúster se ha convertido en la estrategia inmediata que utilizan los grandes monopolios como Google y Facebook para el manejo de grandes volúmenes de datos y como garantía de la disponibilidad de sus recursos. Se propone como una alternativa novedosa para mejorar el rendimiento de sistemas, enmarcado en dominios de aplicación tales como: sistemas de bases de datos (Rodríguez, 2001), centros de procesos de datos (González, 2009), servidores de aplicaciones (Chévez, 2012) o sistemas web (Ramírez, 2010).

Por las ventajas que brinda el uso de la tecnología de clúster, son varios los sistemas informáticos en el sector de la salud, que la utilizan para apoyar investigaciones científicas que requieren de cálculos avanzados. Un ejemplo de esto es el Centro de Ingeniería Genética y Biotecnología en Cuba (CUJAE, 2003) donde se configuró un clúster que se utiliza con fines investigativos. También en la Universidad de las Ciencias Informáticas (UCI) se utiliza esta tecnología, es en el software del Sistema Integral para la Atención Primaria de Salud (SIAPS) (Germán & Suau, 2013) donde se concreta su uso, mejorando la disponibilidad del servicio de la atención sanitaria en este nivel.

Algunos sistemas gestores de bases de datos como Oracle (oracle, 2009), MySQL (mysql, 2011) y PostgreSQL (postgresql, 2011) pueden aplicar tecnología de clúster nativo o por medio de herramientas externas que se lo permiten. Por su parte, PostgreSQL al ser escalable y robusto lo hace idóneo para su uso en sitios web que posean una alta concurrencia de usuarios. Entre sus características (Aliaga, 2008) se destaca el soporte de distintos tipos de datos, la declaración de funciones propias y el uso de índices (Acosta, 2008).

En el marco del proceso de informatización de la sociedad, se desarrolla en la Universidad de las Ciencias Informáticas, un sistema para la gestión hospitalaria por parte del Centro de Informática Médica (CESIM) como parte de un convenio específico de cooperación para el mejoramiento de la calidad de la atención médica entre PDVSA industrial – Guardián del Alba . Dicho sistema, contiene como herramienta de persistencia de datos físicos un servidor PostgreSQL en su versión 9.1, que aloja aproximadamente treinta y nueve esquemas, tres mil veintinueve tablas y ocho millones setecientos mil

registros donde almacena información relacionada con los pacientes y medios básicos del hospital. Además se comprobó que la salva de la base de datos ocupa un gigabyte y medio. También, se ha evidenciado mediante los ficheros de acceso que como promedio, por cada funcionalidad en la aplicación utilizada por un usuario, se realizan cinco transacciones a la base de datos.

Este sistema de Gestión Hospitalaria incluye un framework de Mapeo Objeto-Relacional (ORM) (hibernate, 2010) como intermedio entre la capa de la lógica del negocio y la de persistencia de datos, evitando excesivas normalizaciones que ralenticen las consultas complejas. También se tuvo en cuenta el paginado del lado del servidor de base de datos para minimizar sobrecargas de la memoria del servidor de aplicaciones. Se optó por un bloqueo optimista de los datos, permitiendo mayores niveles de concurrencia pues disminuyen los tiempos de bloqueos en la base de datos así como el número total de conexiones a la misma.

El sistema informático para la gestión hospitalaria utiliza una estrategia de despliegue de dos nodos y dos niveles mediante una arquitectura cliente-servidor, es decir; un nodo para el servidor de aplicaciones encargado de la lógica de negocio y otro con una instancia para el servicio de base de datos; cada uno ubicado en ordenadores distintos. Dicha estrategia puede traer como consecuencia que, ante un posible fallo del servidor de bases de datos, ya sean procesos congelados o rotura del hardware, el sistema colapsaría y dejaría de prestar servicios vitales para los pacientes.

El hecho de que exista una única instancia del servidor de bases de datos, provoca que ante el aumento del número de transacciones del sistema causadas por numerosas funcionalidades; a pesar del apoyo del ORM, o la gran cantidad de nuevas peticiones generadas por los usuarios que se suman al sistema, se tenga que ampliar los parámetros de configuración en el servidor en aras de lograr un mejor rendimiento de la aplicación. Comprometiendo la escalabilidad vertical del sistema por la sobrecarga relativa al hardware.

La dificultad de mantener en línea el servicio de base de datos en el sistema informático, trae como consecuencia la disminución de la operatividad del servicio, disminuyendo en gran medida la eficacia y eficiencia de los servicios sanitarios. En caso de la no caída del sistema, el retraso de la aplicación interfiere en los procesos de consulta de datos de los pacientes, atrasa los análisis estadísticos para las entidades de salud encargadas de controlar las epidemias y obstruye la toma de decisiones de los especialistas con respecto al diagnóstico de los pacientes.

La situación anterior arroja como **problema de la investigación**: ¿Cómo garantizar la disponibilidad y aumentar el rendimiento del servidor de bases de datos en el Sistema Informático para la Gestión Hospitalaria del Centro de Informática Médica?

El **objeto de estudio** se centra en: Clúster de bases de datos en PostgreSQL.

El **campo de acción** lo constituye: Clúster de bases de datos en PostgreSQL para Sistemas de Gestión Hospitalaria.

Como **objetivo general** se tiene: Diseñar una estrategia de clúster de bases de datos que garantice la disponibilidad y aumente el rendimiento del Sistema de Gestión Hospitalaria del Centro de Informática Médica.

Para lograr el objetivo propuesto se plantean las siguientes **tareas de la investigación**:

- ✓ Realización de un análisis crítico y valorativo de clústeres de bases de datos en el ámbito de la salud existentes a nivel nacional e internacional, estableciendo similitudes con la investigación en curso.
- ✓ Selección de las herramientas y tecnologías en las que se apoya la construcción de la estrategia de clúster.
- ✓ Elaboración de una estrategia de clustering en el Sistema de Gestión Hospitalaria del Centro de Informática Médica.
- ✓ Validación de la propuesta de solución mediante las herramientas identificadas y sobre la base de los resultados obtenidos en trabajos relacionados.

A lo largo de la investigación es necesario utilizar los siguientes **métodos científicos**:

Métodos teóricos

Analítico-Sintético: A través de este método se desarrolló la investigación y posibilitó descomponer la esencia de las estrategias de clúster y comprender las relaciones entre sus componentes (técnicas de clúster, tipos de clúster, arquitectura).

Inductivo-Deductivo: Permitió realizar un razonamiento de los diferentes aspectos relacionados con las estrategias de clúster, llegando a un grupo de conocimientos abarcadores, desde diferentes puntos de vistas para realizar el análisis y resumir las principales particularidades de estas.

Hipotético-Deductivo: Se concretó en el planteamiento de la hipótesis y juega un papel fundamental en la validación de la solución.

Análisis Histórico-Lógico: Se utilizó en la exploración de soluciones o investigaciones de estrategias de clúster y la reproducción de sus elementos de importancia para la investigación.

Genético: Se sintetizó al generalizar la solución a los Sistemas de Gestión Hospitalaria.

Modelación: Se puntualizó a través de la presentación de modelos o diagramas para el mejor entendimiento de la solución y enriquecimiento de la investigación.

Métodos empíricos

Observación: Posibilitó la identificación del problema de la investigación.

Experimento: Permitió realizar las pruebas a la solución, brindando los datos necesarios para comprobar si la investigación es factible.

Entrevista: A través de él se desarrolló la obtención de información en una conversación no planificada con directivos del Centro de Informática Médica para la elaboración de la problemática.

Actualidad y necesidad del trabajo

El Sistema de Gestión Hospitalaria del CESIM en su servicio de bases de datos, no es capaz de mantener la disponibilidad de sus recursos según el crecimiento de la información y la cantidad de usuarios nuevos del sistema. La arquitectura de despliegue del sistema presenta una sola instancia del servidor, lo cual obstruye la capacidad de re-establecerse ante un fallo de hardware o de software; por lo que se hace necesaria la implementación de una estrategia de clustering que subsane estos problemas y permita la disponibilidad de los servicios sanitarios.

Antecedentes

La tecnología de clustering ha sido utilizada por empresas nacionales e internacionales, entre las que se destacan: el Departamento de Salud de Inglaterra; que se propuso implementar una solución de clúster (Fleming, 2004) en el Servicio Nacional de Salud con el objetivo de alcanzar un registro único

de atención electrónica centralizado para los pacientes. Se encuentra también el Centro de Ingeniería Genética y Biotecnología de Cuba que cuenta con un clúster (CUJAE, 2003) que se dedica a las tareas dentro de la investigación científica de la institución. Además, se tiene en la UCI, en el centro CESIM, una implementación de una solución de clúster sobre el servidor de aplicaciones JBoss.

Aportes prácticos esperados de la investigación

Al concluir la investigación el servicio de bases de datos del sistema informático para la Gestión Hospitalaria debe haber adquirido la capacidad de: aumentar la disponibilidad de los servicios ofrecidos, manejar las cargas de los nuevos usuarios que se agreguen al sistema, ser transparente para el usuario y durable en el tiempo.

El presente documento sigue la **estructura** expuesta a continuación:

Capítulo 1. Fundamentación teórico-metodológica sobre los clústeres de bases de datos: Se hace una introducción a la tecnología de clustering, específicamente en sistemas de bases de datos. Además, se exponen las características que deben poseer las estrategias de clustering. Se presentan técnicas de esta tecnología, los tipos de clúster existentes, tipos de prueba de rendimiento, carga y las herramientas para su medición.

Capítulo 2. Propuesta de la estrategia de clúster de base de datos para el Sistema de Gestión Hospitalaria: Abarca la solución de la estrategia, diagramas para mejor entendimiento del diseño lógico, físico y de despliegue. Se muestra la gestión de la seguridad y aspectos de importancia para la interconexión de las herramientas escogidas.

Capítulo 3. Validación de la propuesta de solución: Contiene la descripción del proceso de pruebas de rendimiento y carga, recogida de medidas y comparación con el entorno actual.

Cada capítulo ampara una conclusión parcial donde se sintetizan los resultados semifinales permitiendo una mejor apreciación de los resultados generales.

2.1. Breve reseña de la tecnología de clúster

Enlazar varias computadoras con un propósito común fue un concepto que comenzó en las décadas de los 50 y 60. Tuvo sus inicios cuando la Fuerza Aérea de los Estados Unidos estableció la red de defensa más grande hasta ese momento (Ibañez, 2009), el sistema SAGE (Semi-Automatic Ground Enviroment). Este sistema coordinaba varias estaciones de radar con el objetivo de detectar amenazas aéreas (aviones bombarderos soviéticos o misiles intercontinentales). En 1967 Gene Amdahl [1] publica lo que se considera como las bases del procesamiento en paralelo. El artículo definía la "Ley de Amdahl" (Diaz, 2010), empleada para calcular la mejora máxima de un sistema cuando solo una parte de éste es mejorado.

Fue en el año 1984 cuando la compañía estadounidense DEC *(Digital Equipment Corporation)* utilizó por primera vez el término clúster para representar un conjunto coordinado de computadoras. VAXclusters (Lugo, 2011) fue el nombre que recibió su sistema operativo, enfocado a proveer redundancia de disco. Estos sistemas mantienen los datos en dos discos, cualquier escritura de datos es realizada en ambos discos, pero las operaciones de lectura son consultadas indistintamente, ya que contienen la misma información. En el VAXCluster todos los procesadores tienen acceso al HSC *(Hierarchical Storage Controllers)* a través del acoplador estrella, por lo que no hay ningún punto de falla en el hardware.

Para 1994 y bajo el patrocinio del proyecto ESS *(Earth and Space Sciences)*, un grupo de investigadores del CESDIS *(Center of Excellence in Space Data and Information Sciences) en Estados Unidos*, construye un clúster que consiste en dieciséis equipos (Yáñez, 2011) con procesadores Intel DX4 interconectados por una red tipo Ethernet de canal múltiple, con el objetivo de realizar cálculos paralelos de alto rendimiento. A los clústeres de este tipo se les conoce genéricamente como clústeres Beowulf.

[1] Eugene Myron Amdahl estadounidense de origen noruego, arquitecto computacional y una de las personalidades más importante y excéntricas en la historia de la informática y la computación.

2.2. Tecnología de clúster

Se le denomina clúster *(del inglés: cluster, traducción al español: grupo)* a las agrupaciones de diferentes elementos que guardan similitud entre ellos (Arboníes, 2000). En la informática, se puede expresar como un conjunto de computadoras interconectadas que actúan de forma transparente para el usuario. En un clúster es posible aumentar el rendimiento del sistema en general al poder utilizar el poder de cómputo de varias CPU *(Central Processing Unit, traducción español: Unidad Central de Procesamiento)*. Constan de varios componentes para su funcionamiento (Ibañez, 2009):

- ✓ Nodos: simples computadores, sistemas multi procesador o estaciones de trabajo.
- ✓ Conexiones de red: comunicación entre los nodos del clúster (*Ethernet, Fast Ethernet, Gigabit Ethernet, Myrinet, Infiniband, SCI*).
- ✓ Middleware: software que generalmente actúa entre el sistema operativo y las aplicaciones.

Un clúster se conforma de elementos de software, hardware y middleware (Paulo, 2010). El hardware está conformado por los dispositivos físicos disponibles en cada uno los nodos que lo compone y el software, por los programas que se ejecutan sobre los nodos del clúster. El middleware es el encargado de la comunicación entre la aplicación *(programa)* y los nodos *(hardware)*; es considerada la capa más importante de esta tecnología porque contiene la arquitectura utilizada para el despliegue del clúster.

Los clústeres tienen la propiedad de afrontar mayores volúmenes de datos a medida que crece el sistema, a esta capacidad se le denomina escalabilidad. Existen dos tipos de escalabilidad (Castro, 2012), la escalabilidad vertical y la escalabilidad horizontal. La vertical se basa en añadir más recursos a un nodo particular del sistema y la horizontal en agregar más nodos al sistema.

Los clústeres se clasifican (Duran, 2009) según las características de hardware y sistema operativo de sus nodos en: homogéneo *(mismo hardware y sistema operativo)*, semi-homogéneo *(diferentes dispositivos de hardware pero mismo sistema operativo)* y heterogéneo *(diferente sistemas operativos y hardware)*.

2.3. Beneficios de la tecnología de clúster

La tecnología de clúster juega un papel decisivo en la solución de problemas de las ciencias, la ingeniería y en la ejecución de aplicaciones comerciales. Con el tiempo ha ido evolucionando para apoyar actividades como: la supercomputación, el comercio electrónico y las bases de datos de alto rendimiento. Entre sus beneficios se encuentran:

- ✓ Incrementar la capacidad de procesamiento.
- ✓ Automatización del monitoreo de los recursos y las actividades del servidor.
- ✓ Disponibilidad y confiabilidad del sistema.
- ✓ Carga de trabajo repartida por los nodos del clúster.
- ✓ Enfrentar cargas de trabajo cada vez mayores, sin dejar por ello de prestar un rendimiento aceptable.

2.4. Tipos de clúster

La implementación de un tipo de clúster está arraigado a las características del entorno y a los procesos del negocio que se gestionen en el ambiente de uso. Diciendo esto es válido aclarar que un tipo de clúster es independiente de los clústeres que se generan en distintas capas, por lo que no importa en qué nivel se clusterice un servicio, un tipo de clúster es genérico para cualquier subtipo especifico de uso de clúster y cada uno de los subtipos puede presentar varias implementaciones de tipo de clúster. La presente investigación, opta por la construcción de una estrategia que responda a un clúster de base de base de datos de alta disponibilidad y balance de carga ya que el rendimiento aumentado al final de su construcción no se considera dentro del umbral de un clúster de alto rendimiento

2.4.1. Clúster de alta disponibilidad

Un clúster de disponibilidad alta es aquel que brinda recursos de forma ininterrumpida y garantiza un funcionamiento correcto de los servicios (cala, 2014) proporcionando flexibilidad y robustez al sistema. Tradicionalmente la alta disponibilidad ha sido un requerimiento exigido a aquellos sistemas que realizan misiones críticas. Sin embargo, se hace cada vez más importante esta cualidad en sistemas comerciales y en áreas académicas, donde el objetivo es prestar los servicios ininterrumpidamente.

Un clúster que brinde alta disponibilidad se hace indispensable en ambientes de almacenamiento de datos sensibles o de acceso constante. Los nodos de un clúster de alta disponibilidad se pueden poner fuera de línea, apagar, actualizar o reparar sin comprometer los servicios que brinda el clúster.

2.4.2. Clúster de alto rendimiento

Un clúster de **alto rendimiento** surge frente a la necesidad que demandan las aplicaciones que necesitan potencia computacional. Persigue proveer a los sistemas con una alta capacidad de ejecución de procesos para el cómputo de grandes volúmenes de datos (Durán, 2009). Entre las áreas más comunes de clústeres de alto rendimiento se encuentran (Ibañez, 2009): el cálculo del estado del tiempo, la astronomía, la investigación en criptografía, la simulación militar, la simulación de recombinaciones entre moléculas naturales y el análisis de imágenes.

Es usual que un clúster de alto rendimiento sea utilizado para resolver problemas que requieren de grandes capacidades de cálculo. Están compuestos por varios nodos que ejecutan código de manera paralela, con el objetivo de obtener resultados en un menor tiempo. Es el tipo de clúster más popular y utilizado desde la aparición de los clúster tipo Beowulf.

2.4.3. Clúster de balance de carga

Un clúster de **balance de carga** (Ibañez, 2009) es el que distribuye su trabajo a través de múltiples nodos. Todos los nodos son capaces de responder a peticiones externas al ejecutar los mismos programas y en caso de que se presente la falla de un nodo, las peticiones son distribuidas entre los nodos activos restantes. Este tipo de clúster tiene la ventaja de que los nodos pueden encontrarse en distintos puntos geográficos. En este tipo de clúster, repartir el volumen de trabajo en un conjunto de servidores trae como consecuencia la mejora en el tiempo de acceso de los usuarios y la respuesta del sistema. Además, al ser un conjunto de servidores el que recibe las peticiones, la caída de uno de ellos no ocasiona una caída total del sistema.

2.5. Tecnología de clúster aplicada a base de datos

Un clúster de base de datos (Aycock, 2006) es un conjunto de ordenadores que se disponen en grupo para resolver un servicio de este tipo. Su clasificación arquitectónica según el compartimiento de la información entre nodos se divide en, shared-nothing (no compartir los datos) y shared-all (compartir los datos). En la arquitectura shared-nothing la instancia y los datos están separados lógicamente, mientras que en la shared-all se encuentran en el mismo contexto de ejecución.

Las soluciones de clúster de base de datos más representativas son:

- ✓ **Oracle Real Application Clusters** (Oracle RAC) (oracle, 2009) es una versión en clúster de Oracle Database basada en una pila integral de alta disponibilidad que se puede utilizar como plataforma de un sistema de base de datos en la nube, así como una infraestructura compartida, asegurando una alta disponibilidad, escalabilidad y agilidad. Se distribuye bajo una licencia comercial bastante cara (oracle, n.d.) si se compara con la competencia. Este sistema distribuido de memoria compartida implementa la técnica de mirroring *(replicado),* aunque *el replicado* también se le puede agregar, sería un costo adicional a la licencia por concepto de cores por nodo y nuevo nodo además. El funcionamiento de esta solución se basa en diferentes instancias Oracle que acceden a la misma base de datos Oracle concurrentemente (Brien, 2007).

- ✓ **MySQL Cluster Carrier Grade Edition** como estrategia se compone de una arquitectura shared-nothing y cuenta con tres tipos de nodos: Data node (nodo de almacenamiento; solo contiene los datos), Management node (nodo encargado de la configuración y monitoreo del cluster) y SQL node (nodo que ejecuta una instancia de MySQL server). Este sistema implementa técnicas de sharding y replicado.

  MySQL es un gestor de base de datos que tiene licencia dual; una gratis (GPL) y otra comercial bajo la guardia de Oracle que en el año 2010 compró la compañía Sun Microsystems y con ella este producto. Esta versión no cuenta con un equivalente gratuito y su licencia ronda los $10.000 USD sin incluir el pago por concepto de procesador.

- ✓ **Windows Server Failover Clustering** (technet, 2014) es un grupo de servidores independientes que trabajan en conjunto para aumentar la disponibilidad de las aplicaciones y servicios. Es un producto distribuido por Windows bajo una Licencia comercial que se despliega en una arquitectura shared-all. Esta solución se apoya en tres componentes que en conjunto permiten su funcionamiento. Microsoft Cluster Service (MSCS), es el encargado de los procesos y comunicación entre los nodos. Component Load Balancing (CLB), reparte las peticiones según el grado de balance de cada nodo. Network Load Balancing Services (NLB), distribuye el tráfico de IP entre múltiples nodos y redirecciona el tráfico en caso de caída de alguno.

Ademas, PostgreSQL *(epígrafe 1.11.1)* aunque no consta con un producto o versión en la cual el uso de esta tecnología sea nativa, posee aplicaciones de terceros *(epígrafe 1.8)* que le permiten hacer uso de las mismas. Por lo amplio del abanico de posibilidades para este sistema gestor de base de datos es posible enmarcar la solución lo más a la medida del entorno.

Una estrategia de clúster a la medida mejora el rendimiento del sistema por ejemplo: tiene la opción de realizar la réplica de los servidores esclavos sincrónica o asíncrona, o escoger usar un pool de conexiones externo al de PostgreSQL. Las herramientas que permiten estas funcionalidades por lo general tienen un objetivo único.

2.6. Beneficio de la tecnología de clúster de base de datos

Los gestores de base de datos (Ramírez, 2008) son los encargados del almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para añadir, borrar, modificar y analizar los datos. Hoy las empresas optan por estos programas de computadores para aumentar el rendimiento en general de sus procesos. Muchas veces no es suficiente con solo realizar operaciones básicas sobre estos datos y se necesita aumentar el rendimiento o aumentar la cantidad de personas que pueden acceder a estos datos concurrentemente.

La tecnología de clúster subsana estas cuestiones y permite (oracleclusters, 2014) entonces:

- ✓ Rapidez.
- ✓ Continuidad del negocio.
- ✓ Interoperabilidad.
- ✓ Flexibilidad.
- ✓ Concurrencia.
- ✓ Escalabilidad para las aplicaciones.
- ✓ Usa hardware de bajo costo.
- ✓ Capacidad de conmutación por error.

- ✓ Habilidad para incrementar la capacidad del sistema a través del tiempo mediante la adición de servidores.

2.7. Técnicas que se utilizan en la tecnología de clúster de base de datos

Las técnicas de clúster de base de datos, son el conjunto de tecnologías con que se dispone para llevar a cabo un clúster de este servicio. Su uso depende del dominio de aplicación y en gran medida de la complejidad de los procesos del negocio. Dentro de ellas se encuentra la réplica; proceso de copia de una misma base de datos en varios nodos.

El replicado puede ser sincrónico o asíncrono (postgresql, 2013), en el sincrónico el modificado de los datos no se realiza hasta que los servidores no terminen las transacciones y la asíncrona permite un retraso en el tiempo de la transacción. Esta técnica emplea un concepto arquitectónico donde existe uno o varios nodos maestros que mantienen a sus esclavos actualizados. Se utiliza como punto de recuperación en caso de que el nodo maestro falle.

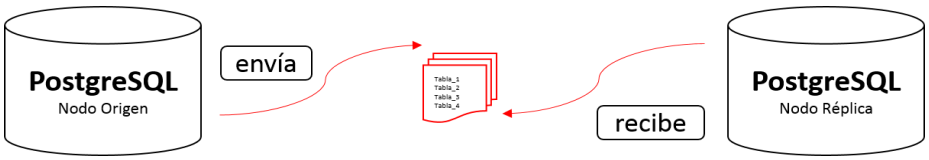


Figura 1.Replicado de Datos.

Fuente: Elaboración propia.

Su contraparte, el particionado de la base de datos, es una técnica factible, siempre que se escoja un criterio de división consecuente con respecto a la probabilidad de consulta de la información. Las consultas que más accedan a una parte de los datos, pueden optimizarse al separarlos hacia otro servidor.

Se puede conseguir la creación de particiones de datos sin dividir las tablas si estas se colocan íntegramente en localidades individuales (postgresql, 2013). La creación de particiones horizontales *(sharding)* divide una tabla en varias, cada una contiene el mismo número de columnas, pero menos filas. El particionado vertical divide una tabla creando nuevas, cada una con menos columnas y la misma cantidad de filas.

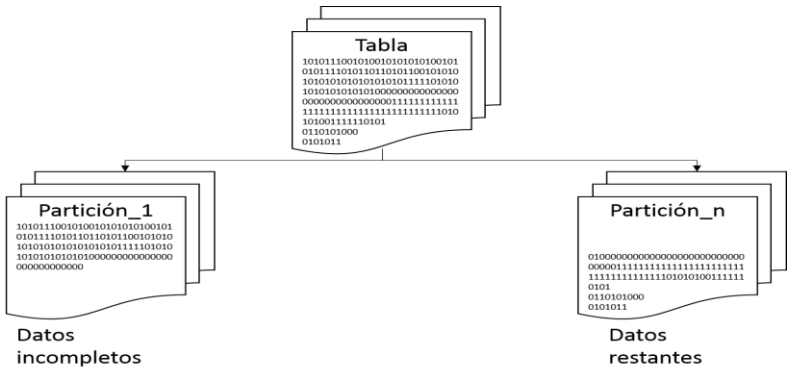


Figura 2: Particionado de la base de datos.

Fuente: Elaboración propia.

La agrupación de conexiones *(connection pool)* es otra técnica dentro de las técnicas de clustering de base de datos. Se encarga de mantener y gestionar un número de conexiones físicas que se reutilizan automáticamente para aumentar la eficiencia del sistema (eliax, 2012). De esta forma cuando se crea una nueva conexión desde la aplicación, se obtiene una conexión lógica manejada por el gestor de conexiones *(connection manager)* que al cerrarla es devuelta a dicho manejador; siendo capaz de ofrecer múltiples conexiones lógicas utilizando un reducido número de conexiones reales.

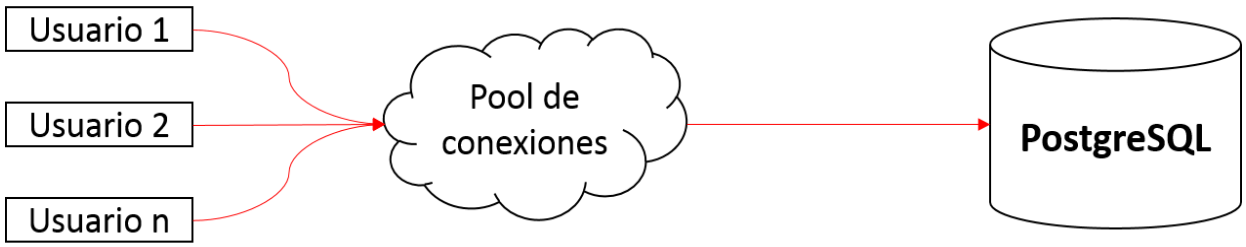


Figura 3: Pool de conexiones.

Fuente: Elaboración propia.

Usualmente, dentro de la solución de clustering se incluye la ejecución de consultas en paralelo (Oberto, 2012), esta técnica hace posible distribuir una consulta en más de un nodo, asignando pasos a computadoras diferentes. Una vez terminada la consulta, se ensamblan los resultados y se envía la información de vuelta al usuario. Cada computadora trabaja en una parte de la labor y en conjunto terminan una solicitud SQL en menos tiempo.

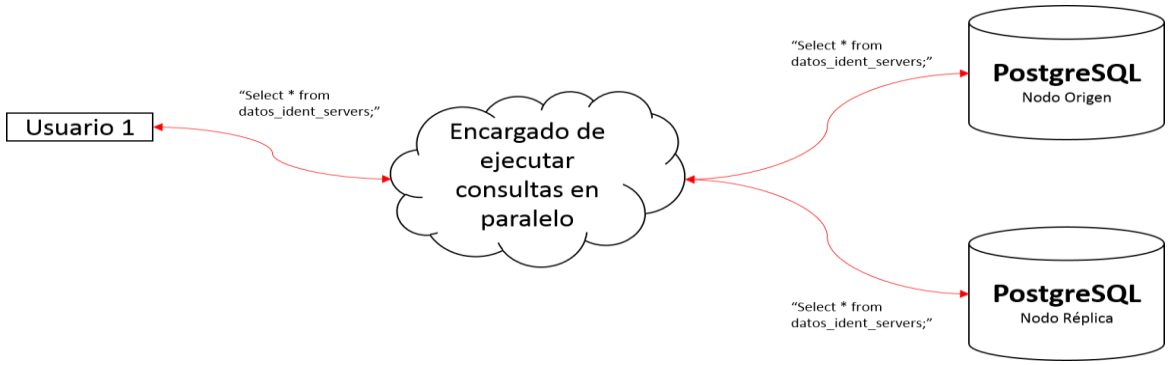


Figura 4: Consultas en paralelo.

Fuente: Elaboración propia.

En la construcción de la estrategia de clúster para el Sistema de Gestión Hospitalaria del Centro de Informática Medica es imprescindible no sobrecargar al clúster con innecesarias técnicas que desdibujen el rendimiento, por lo que se identificó según las causas de caída del HIS las siguientes técnicas a utilizar:

- ✓ Pool de conexiones
- ✓ Balance de carga
- ✓ Réplica de datos

Las técnicas anteriores atienden a situaciones específicas del sistema HIS, en el caso de pool de conexiones, responde al problema del incremento de los usuarios al sistema y su baja capacidad de gestionarlos. La réplica de datos se corresponde al problema de la poca disponibilidad del servicio y la falta de escalabilidad del sistema actual. Por último el balanceo de carga responde a ambas situaciones y presenta un cimiento sólido al repartir las obligaciones entre los servidores.

2.8. Tecnologías que implementan técnicas de clustering en PostgreSQL

Son variadas las tecnologías que se utilizan para la construcción de clúster de base de datos. Según las técnicas que implementan, se separan para su mejor clasificación. Teniendo en cuenta que hay herramientas que realizan más de una función, en cada sub-epígrafe se señalan los casos correspondientes.

1.8.1. Tecnologías que implementan balanceo de carga

DBBalancer

DBBalancer es un middleware (dbbalancer, 2011) que se asienta en medio de los clientes de bases de datos y el servidor de base de datos. Actualmente el único servidor soportado es PostgreSQL, pero presenta una arquitectura abierta que podría implementar más servidores en un futuro. Uno de sus puntos más fuertes es que puede ser encausado sin cambiar una línea de código existente, ya que el equilibrio se hace a nivel del protocolo de PostgreSQL. Este software actualmente, se distribuye bajo una licencia comercial.

Pgpool-II (versión 3.3)

Pgpool –II, es un middleware que funciona entre el servidor y el cliente de base de datos PostgreSQL. Distribuido bajo la licencia BSD *(Berkeley Software Distribution)* proporciona la gestión y manejo de *pool de conexiones* a los servidores PostgreSQL y las reutiliza cada vez que llega una nueva conexión con las mismas propiedades *(nombre de usuario, bases de datos, versión del protocolo),* eliminando conexiones redundantes.

Cuenta con características de *replicado* que permite crear copias de seguridad sincrónicas, por lo que el servicio continúa sin detener servidores en caso de un fallo en disco. El replicado de la base de datos brinda la posibilidad de que al ejecutar una consulta en cualquier servidor devolverá el mismo resultado, reduciendo la carga en los nodos mediante la distribución de consultas entre varios servidores.

Una consulta se puede ejecutar en todos los servidores a la vez (pgpool, 2013), para reducir el tiempo total de ejecución, a esto se le llama paralelizado de consultas que funciona mejor en la búsqueda de datos a gran escala. Pgpool -II es totalmente transparente para el cliente y para el servidor PostgreSQL.

La gestión de la administración se facilita con la herramienta PgPoolAdmin (pgpool, 2013) que es una interfaz web que permite gestionar la configuración del Pgpool de forma rápida y eficiente.

1.8.2. Tecnologías que implementan replicado

Slony (versión 2.2.2)

Slony es un sistema de replicado (slony, 2010) asíncrona maestro-esclavos tolerante a fallos y desarrollado para PostgreSQL. Incluye todas las características y capacidades necesarias para replicar grandes bases de datos haciéndolo idóneo para data centers y sitios de respaldo (Calderón, 2011).

PL/Proxy *(versión* 2.5*)*

PL/Proxy es un lenguaje compacto (plproxy, 2011) para llamadas remotas entre bases de datos PostgreSQL. PL/Proxy no implementa en si el particionado ni réplica de base de datos, sino que permite la ejecución de funciones remotas, lo que al particionar la base de datos o replicarla puede ser utilizado como comunicador entre los servidores. Es utilizado con fines de proxy lo que brinda una extra seguridad y protección de los datos.

Bucardo *(versión* 4.4.0*)*

Bucardo, es una herramienta de replicado de datos para el SGBD PostgreSQL. Este sistema es asíncrono, puede realizar replicado de tipo maestro-esclavo y maestro-maestro (este tipo de replicación solo soporta dos servidores maestros). Está desarrollado en el lenguaje de programación Perl. Es Software Libre y está liberado bajo la licencia BSD. (*Berkeley Software Distribution*) (Delgado, 2013).

1.8.3. Tecnologías que implementan pool de conexiones

Pgbouncer (versión 1.5.4)

Pgbouncer (postgresql, 2012) es un manejador de conexiones ligero para PostgreSQL que reduce drásticamente el tiempo de procesamiento y los recursos en el mantenimiento de un gran número de conexiones a una o varias bases de datos. Pgbouncer suele utilizarse para ampliar el número de conexiones de usuario que se puede manejar en un entorno de altas prestaciones. Es parcialmente re-configurable on-line, y capaz de mantener las conexiones de los clientes aun cuando el servidor esté reiniciándose.

*Nota: La herramienta **pgpool** también implementa pool de conexiones.*

Tabla comparativa de tecnologías

Tabla 1: Tabla de tecnologías libres que implementan técnicas de clúster.

Herramienta	Licencia	Madurez	Arquitectura	Replicado	Pool de conexiones	Balance	Particionado de consultas
Bucardo	BSD	estable	multi-maestro	Si	NO	NO	NO
Slony	BSD	estable	maestro	SI	NO	NO	NO
Pgpool-II	BSD	estable	middleware	SI	SI	SI	SI
Pgbouncer	BSD	estable	middleware	NO	SI	NO	NO
PL/Proxy	-	estable	middleware	NO	NO	NO	NO
DBBalancer	comercial	beta	middleware	SI	SI	SI	NO

Para llevar a cabo la construcción de la estrategia de clúster es necesario realizar las siguientes técnicas:

- ✓ Pool de conexiones.
- ✓ Réplica de datos.
- ✓ Balance de carga.

Para la selección de las herramientas es necesario que se implementen la mayor cantidad de técnicas con un número reducido de tecnologías. Proponiendo la idea anterior con ánimo de facilitar las configuraciones dentro del clúster y construir una estrategia lo más óptima posible, se propone como herramienta escogida Pgpool porque aporta balanceo de cargas y pool de conexiones. En cuanto a la réplica de datos la única opción que se ajusta a la investigación es Bucardo porque aporta una arquitectura multi-maestro.

2.9. Valoración de clúster de base de datos en la salud

En la actualidad existen diferentes sistemas informáticos destinados a la salud, desplegados con el ánimo de aumentar la calidad de los servicios sanitarios. Luego de un estudio exhaustivo de la utilización de clúster de base de datos en estos sistemas, se identificó tres de ellas. A continuación se caracterizan estos sistemas.

- ✓ Laboratorios **CINFA** es una empresa farmacéutica española dedicada al desarrollo, fabricación y comercialización de medicamentos y productos sanitarios. Por la cantidad de información que gestiona esta entidad, aplica una estrategia de clúster (cuore, 2010) llamada **Oracle Real Application Clusters** *(epígrafe 1.5)*. El reto era proveer acceso a seiscientos usuarios las

veinticuatro horas del día, con un rendimiento excelente en un sistema estable con la capacidad de crecer y de fácil manejo.

- ✓ **Hospital de Clínicas Caracas** (HCC), fundado en marzo de 1985, actualmente (suprabt, 2011) cuenta con más de 1400 empleados entre personal administrativo, médicos, enfermeras, camilleros, mensajeros, mantenimiento, entre otros; operando en sus 8 pisos, con 147 consultorios, 12 quirófanos y varias oficinas y atendiendo a un promedio mensual de 1250 pacientes hospitalizados, 1375 emergencias atendidas (adultos y pediátricos), 1084 intervenciones quirúrgicas y 84 partos. El **Hospital de Clínicas Caracas** utiliza **Oracle Real Application Clusters** *(epígrafe 1.5)* con el objetivo de automatizar y centralizar las operaciones de las diferentes áreas, para así brindar un mejor servicio a los clientes y mayor control operativo, táctico y estratégico.

Ambos sistemas informáticos contribuyen con la misma estrategia de clúster a la investigación. La utilización de Oracle RAC como una solución integradora les permite gestionar todos los procesos e información que se genera desde la capa de negocio, sin tener que recurrir a ningún software externo como un balanceador de carga, un pool de conexiones o una herramienta de réplica. Oracle RAC presenta una arquitectura que comparte los datos físicos por todas sus instancias en un solo nodo *(epígrafe 1.5)*. Esto trae como consecuencia que se afecte el rendimiento de la base de datos por la cantidad de instancias activas en el clúster compartiendo un mismo recurso.

La investigación en curso, debe mantener la información almacenada anteriormente por lo que la portabilidad de los datos desde el antiguo sistema no es asegurable con esta estrategia, lo que ocasionaría perdida de información de los pacientes. Este sistema se distribuye bajo una licencia comercial. Hay soluciones que son gratuitas y pueden llegar a superar la anterior si se considera que una solución personalizada no obliga a desplegar el clúster con una arquitectura específica. El nivel de personalización está dado por la variedad de tecnologías que implementan técnicas de clúster *(epígrafe 1.6)* en su mayoría gratuitas y de código abierto.

- ✓ **CIIGMA Hospital** decidió implementar una solución de administración de hospital (HIS) para optimizar procesos, disminuir costos y conseguir la eficiencia operativa. Se optó por PALASH™ diseñado y desarrollado por Seed Healthcare Solutions. El HIS desplegado se integra con **Windows Server Failover Clustering** *(epígrafe 1.5)* (technet, 2014). Ambas soluciones permiten una información más precisa, aumenta la transparencia y la eficiencia operativa. Esto ha ayudado a la organización a reducir los gastos de funcionamiento de 25 a 35 por ciento.

Windows Server Failover Clustering es la nueva opción en que se apoya **CIIGMA Hospital** para el aumento de la calidad de sus servicios. Esta estrategia en realidad es la unión de tres componentes *(epígrafe 1.5)* los cuales brindan técnicas diferentes para la construcción del clúster. Uno de los inconvenientes de utilizar esta solución es que hay que pagar por costos de licencia lo que sería un costo adicional para la investigación. Este sistema se basa en **MSSQL** un sistema de base de datos creado por Microsoft. De forma general se puede decir que no cumple con la investigación por distribuirse bajo una licencia comercial.

2.10. Pruebas a realizar a la estrategia de clúster

Las pruebas (Sommervile, 2005) no son más que la forma de validación y verificación de un sistema mediante pasos organizados y planificados para el buen desempeño de las mismas. Se utilizan para encontrar errores de cualquier índole y es común que varíen sus tipos según la metodología de desarrollo.

En la investigación en curso no se desarrolla software y por consecuencia no hay una metodología de desarrollo, por lo que las pruebas no se definen de esta manera. Es común que a los sistemas de base de datos se les realice pruebas de rendimiento para detectar cuellos de botella o alguna errónea configuración.

En la ingeniería del software, las pruebas de rendimiento son las pruebas que se realizan, desde una perspectiva, para determinar lo rápido que realiza una tarea un sistema en condiciones particulares de trabajo. También puede servir para validar y verificar otros atributos de la calidad del sistema, tales como la escalabilidad, fiabilidad y uso de los recursos. Las pruebas de rendimiento son un subconjunto de la ingeniería de pruebas, una práctica informática que se esfuerza por mejorar el rendimiento, englobándose en el diseño y la arquitectura de un sistema.

Pruebas de rendimiento: Las pruebas de rendimiento de software (Sommervile, 2005) demuestran en qué medida se cumplen o no los requerimientos de disponibilidad del sistema. A menudo se utilizan para medir qué parte del software no se desempeña bien bajo la carga. Dentro de las pruebas de rendimiento se encuentran las pruebas de estrés:

- ✓ Estas pruebas se utilizan 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. Estas se realizan para determinar la solidez de la aplicación en los momentos de carga extrema

y ayuda a determinar si la aplicación rendirá lo suficiente en caso de que la carga real supere a la carga esperada.

Para llevar a cabo este tipo de prueba es necesario apoyarse en herramientas que permitan su desarrollo. Con la implementación de ellas, la investigación tendrá un resultado tangible y comprobable, capaz de sostener por si misma los resultados esperados.

2.11. Herramientas escogidas

La investigación se apoya en diferentes herramientas como base para el diseño de la estrategia de clúster. El sistema gestor de base de datos PostgreSQL en su versión 9.1, administrador de este gestor PgAdmin en su versión 1.12.1. Como herramienta que gestiona la disponibilidad entre los nodos del clúster Heartbeat versión 3.0. El servidor de aplicaciones que actúa como cliente de la estrategia JBoss AS en su versión 4.2.2 y como software para realizar las pruebas Jmeter en su versión 2.11. Para la virtualización de los nodos de la estrategia se utiliza VirtualBox en la versión 2.4.8.

2.11.1. Herramientas de base de datos

PostgreSQL *(versión 9.1)*

PostgreSQL es un Sistema Gestor de Base de Datos *(SGBD)* relacional de código abierto distribuido bajo licencia BSD *(del inglés, Berkeley Software Distribution)*. Cumple totalmente con: Atomicity, Consistency, Isolation, Durability *(características necesarias para que una serie de instrucciones puedan ser consideradas como una transacción: ACID).*

Este SGBD tiene interfaces de programación nativas para varios lenguajes de programación. Ofrece sofisticadas características de control concurrente multiversión *(Multi Version Concurrent Control)*, point in time recovery *(recuperación de la base de datos en cualquier momento desde el último respaldo)*, tablespaces *(lugar físico donde se aloja una base de datos)*, replicado asíncrono, transacciones anidadas, un sofisticado planificador-optimizador de consultas (postgresql, 2011) y es tolerante a fallos de hardware.

PostgreSQL soporta juegos de caracteres internacionales. Es altamente escalable tanto en la cantidad bruta de datos que puede manejar como en el número de usuarios concurrentes que puede atender. Hay sistemas activos en producción con PostgreSQL que manejan más de 4 terabytes de datos (rafaelma, 2009).

PgAdmin *(versión 1.12.1)*

PgAdmin es un administrador para el SGBD PostgreSQL de código abierto y multiplataforma. Está diseñado para responder a las necesidades de todos los usuarios, desde simples consultas SQL *(Lenguaje declarativo de bases de datos relacionales que permite especificar diversos tipos de operaciones)* hasta el desarrollo de complejas bases de datos.

Esta aplicación posee una interfaz gráfica compatible con todas las características de PostgreSQL facilitando la administración del gestor de base de datos. Entre las funcionalidades que incluye se encuentran un editor de sobresaltado de sintaxis SQL *(syntax highlighting SQL editor)* y un editor de código seguro del lado del servidor *(server-side code editor)*.

PgAdmin realiza conexiones no seguras con el SGBD mediante TCP/IP *(Protocolo de Control de Transmisión/Protocolo de Internet)* o cifradas mediante SSL *(Socket Security Language)* para mayor seguridad. Es un software distribuido bajo licencia BSD, desarrollado por una comunidad de expertos de todo el mundo y disponible en más de una docena de idiomas (pgadmin, n.d.).

2.11.2. Herramientas de disponibilidad

Heartbeat *(versión 3.0.0)*

Heartbeat es un producto del proyecto Linux High Availability que persigue proporcionar una herramienta que garantice la fiabilidad, disponibilidad y calidad de servicio a nivel de mensajes entre los nodos del clúster. Es exclusivo para plataformas Linux bajo licencia GPL. Esta herramienta permite implementar clústeres de control descentralizado *(más de un maestro)* siendo estable, flexible y eficiente. Requiere dos máquinas como mínimo para su implementación y su unidad de trabajo es el recurso *(por ejemplo: IP)*.

Heartbeat monitorea el nodo para el que fue configurado y es capaz de detectar su caída. Cuando transcurre un tiempo y el servidor maestro no responde, Heartbeat determina que está fuera de servicio o inactivo y automáticamente activa el servidor secundario que pasa a ser maestro asumiendo las peticiones de los clientes (linux-ha, 2011).

Corosync *(versión 1.4.2)*

Corosync es un proyecto de código abierto bajo la nueva licencia BSD y derivado del proyecto OpenAIS. La misión de Corosync es desarrollar una solución de alta disponibilidad para clústeres tanto comerciales como de código abierto.

Corosync proporciona una capa de comunicaciones fiable para clústeres de alta disponibilidad. Se asegura de que los nodos del clúster pueden enviar y recibir mensajes a través de múltiples vías de comunicación redundantes. Corosync es compatible con múltiples medios de transporte, tales como la multidifusión UDP, UDP unicast y InfiniBand nativo.

Es necesario recalcar que ambas herramientas proporcionan a la investigación la misma capacidad de disponibilidad. En realidad Corosync brinda más oportunidades pero no son necesarias para la investigación y no se consideraron parámetros de comparación entre estas tecnologías. El proyecto Heartbeat en el momento que se desarrolla la investigación está siendo mantenido por su comunidad pero no está recibiendo actualizaciones constantes que aumenten sus funcionalidades porque la comunidad ha votado por darle más soporte a Corosync para en un futuro este lo remplace. Ya Corosync implementa más funcionalidades que Heartbeat o sea que tiene más componentes que asegura una mejor disponibilidad. Hay que tener en cuenta que uno de los objetivos adyacentes de la investigación en curso es la habilidad de durabilidad de la solución. Por lo tanto la herramienta que se adecúa al trabajo es Corosync.

2.11.3. Herramientas para las pruebas

Tsung *(versión 1.5.0)*

Tsung es una herramienta de prueba de carga distribuida que está disponible de forma gratuita como un producto de software de código abierto. Es independiente del protocolo y se puede utilizar para XMPP, HTTP, SOAP, LDAP y servidores PostgreSQL. La herramienta puede simular un gran número de usuarios por cada servidor, lo que es ideal para analizar y probar el rendimiento de las aplicaciones a gran escala, tales como soluciones de mensajería instantánea. El propósito de Tsung es simular los usuarios con el fin de probar el rendimiento de las aplicaciones cliente / servidor basadas en IP. Se puede usar para hacer pruebas de carga y estrés de los servidores (Niclausse, 2013).

Jmeter *(versión* 2.11*)*

Es un proyecto de Apache Jakarta (jakarta, 2011) distribuido bajo licencia Apache (apache, 2012), puede ser utilizado como una herramienta de prueba de simulación de carga en servicios de base de

datos o como una herramienta de pruebas unitarias para conexiones de bases de datos con JDBC *(Java Database Connectivity)*, FTP *(File Transfer Protocol)*, LDAP *(Lightweight Directory Access Protocol)*, Servicios web *(Web Services)*, JMS *(Java Message Service)*, HTTP *(Hypertext Transfer Protocol)* y conexiones TCP *(Transmission Control Protocol)* genéricas. Es extensible y permite realizar pruebas distribuidas en distintos ordenadores (metanotion, 2006) (apache, 2012).

Desde el punto de vista de la investigación ambas herramientas aportan mediciones factibles necesarias en la investigación y soporte para PostgreSQL. Pero la herramienta Jmeter permite realizar pruebas de carga para analizar y medir el desempeño. Dicha herramienta posibilita verificar el rendimiento en recursos estáticos y dinámicos como pudieran ser archivos y bases de datos bajo diferentes tipos de carga concurrentemente, con la posibilidad de obtener una conclusión más acertada. También muestra las operaciones fallidas durante el proceso de pruebas. Además, posee un funcionamiento de hilos que permite realizar varias operaciones al mismo tiempo, lo que brinda diferentes estadísticas de la prueba que se realiza. Se tuvo en cuenta en la selección de la herramienta la facilidad de configuración, y muy importante la portabilidad tanto de sistemas Linux como Windows y Jmeter ofrece ambas características. Por lo que se selecciona Jmeter como herramienta para las pruebas.

2.11.4. Servidor de aplicaciones

JBoss Application Server *(versión 4.2.2)*

JBoss Application Server es un servidor de aplicaciones escrito en Java y registrado a nombre de Red Hat Inc (redhat, 2014) que se comercializa bajo licencia GNU LGPL *(GNU Lesser General Public License)*. Permite la integración de las tecnologías (jbossas, 2012) utilizadas por JBoss Seam *(Java Server Faces, Enterprise Java Beans 3, Java Persistence API, Business Process Modeling)* en un potente servidor de aplicaciones.

2.11.5. Herramienta de virtualización.

Oracle VM VirtualBox *(versión 4.2.8)*

VirtualBox es un software de virtualización disponible en arquitecturas x86 y x64. No solo es extremadamente rico en características de alto rendimiento para clientes empresariales sino que es también la única solución profesional que está libremente disponible como software de código abierto bajo los términos de la Licencia Pública General de GNU (GPL) versión 2.

VirtualBox se ejecuta en Windows , Linux , Macintosh y Solaris y soporta un gran número de sistemas operativos simulados incluyendo pero no limitado a Windows (NT 4.0, 2000 , XP , Server 2003 , Vista, Windows 7 , Windows 8) , DOS / Windows 3.x, Linux (2.4 , 2.6 y 3.x) , Solaris y OpenSolaris , OS / 2 , y OpenBSD .

VirtualBox está siendo desarrollado activamente con los lanzamientos frecuentes y tiene una lista creciente de características, con el apoyo de sistemas operativos que virtualizan y las plataformas que soporta. VirtualBox es un esfuerzo de la comunidad respaldada por una empresa donde Oracle garantiza que se cumpla con los criterios de calidad profesional.

2.11.6. Herramienta de firewall

Iptables *(versión 1.4.14)*

Iptables (Shrivastava, 2013) es el nombre de la herramienta de espacio de usuario (área de memoria donde todas las aplicaciones, en modo de usuario, pueden ser intercambiadas hacia memoria virtual cuando sea necesario) a través de la cual los administradores crean reglas para cada filtrado de paquetes y módulos de NAT. Iptables es la herramienta estándar de todas las distribuciones modernas de GNU/Linux de filtrado de paquetes incluida en el kernel (Barrios, 2011). En él se enumeran los contenidos del conjunto de reglas de filtrado de paquetes.

ConfigServer Security & Firewall *(versión 6.47)*

Es un software mantenido por Configserver (configserver, 2014), empresa especializada en soluciones para cPanel (*Herramienta de administración basado en tecnologías web para administrar sitios de manera fácil, con una interfaz limpia*). Este desarrollo mantiene una gran constancia de actualizaciones, su instalación es sumamente práctica y su configuración, a pesar de que incluye opciones a nivel de técnicas de defensa, está bien documentada.

Es actualizado periódicamente, de fácil instalación y dispone de varios documentos para su configuración. Soporta plataformas como RHEL y derivados, Ubuntu, Debian, entre otras. CSF Firewall, no solo es un firewall, es una suite de seguridad (hostingdiario, 2013) ya que también se verifican firmas md5, archivos modificados y puede interactuar con otras herramientas de seguridad como mod_security.

De las soluciones de firewall se escogió Iptables, una herramienta integrada en el kernel de sistema operativo GNU/Linux. Esto posibilita que el filtrado de paquetes se realice en un nivel bajo y con

tiempos de respuesta mayores a los de otras herramientas de firewall. Esta aplicación le brinda un nivel extra de rendimiento a la solución del clúster y cumple con el objetivo trazado.

2.12. Conclusiones del capítulo.

La incapacidad de cumplir con las necesidades que demanda el entorno del servicio de base de datos del sistema informático del CESIM, propicia el desarrollo de una estrategia de clúster que gestione de forma eficiente los recursos de la base de datos, sobre las premisas de rendimiento y disponibilidad.

En el capítulo se brindan los elementos teóricos necesarios que demuestran; teniendo en cuenta los problemas de disponibilidad y rendimiento, que la construcción de una estrategia de clúster es una solución factible para el Sistema de información del CESIM.

2.1. Caracterización del entorno

El sistema Alas HIS fue creado con el objetivo de Implementar un Sistema de Información de Salud que permita la gestión de los procesos administrativos y asistenciales de toda su red y la interoperabilidad con otros sistemas de salud, suministrando la información sanitaria de los pacientes hacia una Historia Clínica Electrónica y brindando información útil y oportuna para la toma de decisiones. Los datos se transmiten garantizando su cifrado al igual que la firma digital del autor de los distintos documentos clínicos.

La Solución Integral para la Gestión de la Información de Salud, incluye el desarrollo e implantación de varias Soluciones de Software. Estas son: Sistema de Información Hospitalaria (HIS), Sistema de Información Radiológica (RIS), Sistema para el Almacenamiento, Transmisión y Visualización de Imágenes Médicas (PACS) e Historia Clínica Electrónica (HCE).

El sistema estaría distribuido por todos los hospitales donde se encuentre implantado, se comunicarían a través de un centro de datos, que permitiría la existencia de una Historia Electrónica Única, concepto alrededor del cual funciona el resto del sistema. Además cuenta con quince módulos a los que el usuario tiene acceso, entre estos se encuentran los Módulos de Enfermería, Laboratorio, Visor de Historias Clínicas, Emergencias, Epidemiología, entre otros. El acceso a cada uno de los módulos está guiado por una página de inicio, informándole al usuario las características esenciales del módulo así como las funcionalidades que puede ejecutar.

Cuenta también con el módulo de Configuración que permite la administración del sistema, posibilitando configurar las funcionalidades de cada uno de ellos. Garantiza la seguridad de acceso al sistema, posibilitando a su vez la administración de usuarios y roles. Gestiona cada uno de los departamentos, servicios, habitaciones, camas, los datos generales de la institución hospitalaria y visionar la bitácora del sistema.

2.2. Requerimientos funcionales de la estrategia de clúster

Los requerimientos funcionales (Sommervile, 2005) son las capacidades o condiciones que debe cumplir el sistema, cómo el sistema reacciona a entradas particulares y cómo debe comportarse en situaciones específicas. En algunos casos, también pueden declarar explícitamente lo que el sistema no debe hacer. A continuación se muestran los requisitos funcionales del sistema:

- ✓ Balance de carga.
- ✓ Pool de conexiones.
- ✓ Réplica de datos asíncrona.
- ✓ Soportar fallas.

2.3. Requerimientos no funcionales de la estrategia de clúster

Los requisitos no funcionales (Sommervile, 2005) son, los requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este. Propiedades o cualidades que el producto debe tener, aplicándose al sistema como un todo.

Usabilidad:

- ✓ Los parámetros de configuración que se deben cambiar son mínimos.

Accesibilidad:

- ✓ La información y las funcionalidades estarán disponibles y se podrá acceder a ellas en todo momento.

Disponibilidad:

- ✓ Una vez que el sistema esté publicado estará siempre disponible y con la información que solicite la aplicación.

Rendimiento:

- ✓ La aplicación permitirá una alta concurrencia de usuarios.
- ✓ Los tiempos de respuestas serán rápidos.

Legales:

- ✓ Las herramientas seleccionadas para el desarrollo del producto están respaldadas por licencias libres, bajo las condiciones de software libre.
- ✓ La aplicación y toda la documentación generada pertenecen al grupo de proyecto Gestión Documental y Archivista de la UCI.

Software:

- ✓ Ambos nodos deben tener un sistema operativo GNU/Linux.
- ✓ PostgreSQL 8.4 o superior.
- ✓ Los nodos deben ser actualizados con los paquetes más recientes del repositorio.

Hardware:

Procesador

Para que el sistema se ejecute correctamente, debe contar con cualquiera de las siguientes especificaciones:

- ✓ Procesador Intel Dual-Core o superior.
- ✓ Procesador Intel® Xeon® 5140 Dual-Core 3.0 GHz o superior.

Memoria RAM

- ✓ 4 GB de memoria RAM o superior.

Almacenamiento

- ✓ 1199 GB de disco duro o superior.

Nota: Los requerimientos de la investigación en curso responden al documento "Especificación de Requisitos de Software_HIS.doc" que son la guía por la que se rige el Sistema de Gestión Hospitalaria.

2.4. Esbozo general de la propuesta de solución

En el clúster de la presente investigación se persiguen dos objetivos, la alta disponibilidad y el rendimiento. La funcionalidad que persigue el clúster es la de servidor de base de datos. Para llevar a cabo la construcción de la estrategia se necesitaron las siguientes herramientas:

- ✓ PostgreSQL, sistema gestor de bases de datos.
- ✓ Pgpool-II, middleware que gestiona la réplica, el pool de conexiones y balance de carga de los servidores de PostgreSQL.
- ✓ Heartbeat, software para dar alta disponibilidad a Pgpool y a la dirección IP de servicio.

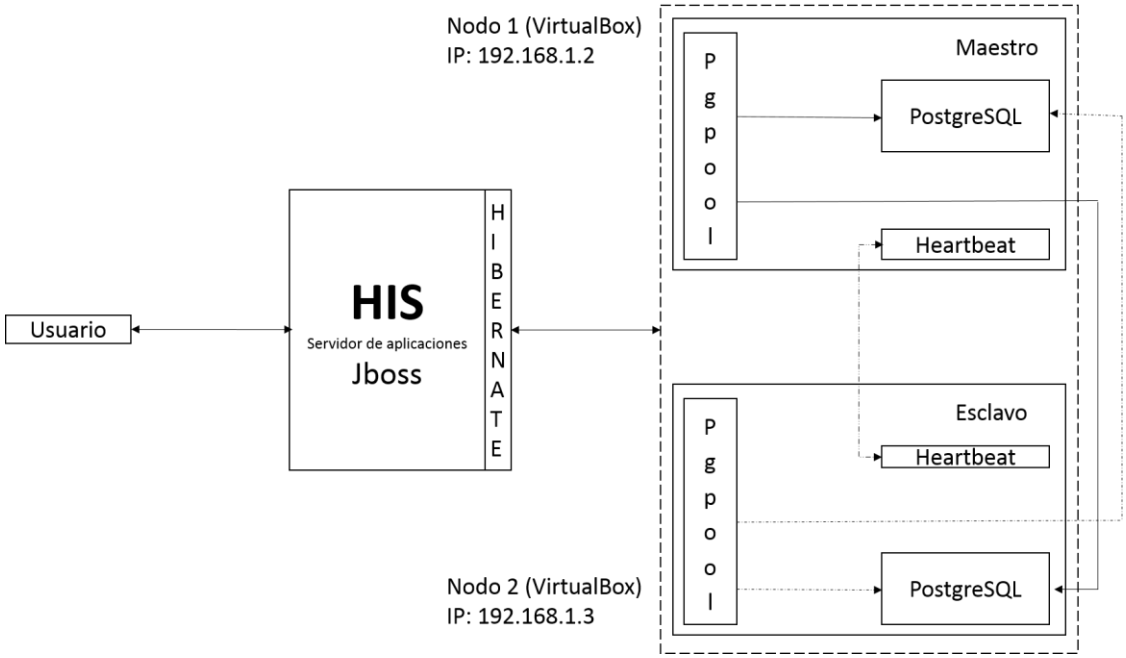


Figura 5: Modelo general de la solución propuesta

Fuente: Elaboración propia

En general el diseño de la estrategia de clúster persigue una arquitectura que no comparte los datos entre sus nodos, sino, que replica la base de datos del nodo maestro al esclavo. La solución contiene solo dos nodos en los que se ejecuta una instancia de Pgpool PostgreSQL y Heartbeat respectivamente. Para el despliegue de las herramientas se utilizó como sistema base Debian server

en su versión séptima. Cada nodo es el resultado de una instancia de una máquina virtual Oracle VM Virtual Box.

El flujo del sistema se inicia cuando la aplicación hace una petición SQL al clúster, esta petición es recogida por la instancia de Pgpool que se ejecuta en el nodo maestro. Luego si la sentencia es consulta, Pgpool la envía hacia cualquier servidor de base de datos, este la procesa y la envía de vuelta hacia la aplicación. Sin embargo si es una modificación, entonces Pgpool re-direcciona los comandos SQL hacia ambos servidores del clúster para mantener la consistencia en la base de datos.

Uno de los objetivos que persigue la presente investigación, es dotar al sistema de tolerancia a los fallos y así garantizar la disponibilidad de los servicios. Esto se logra con la herramienta Heartbeat, que es la encargada de monitorear ambos nodos a través de latidos, los cuales en caso de caída serian inexistentes. En este caso particular de la detección de caída de un nodo, Heartbeat ejecuta un script llamado “failover.sh” que se crea con pasos a seguir por parte del nodo que detecta la caída. Si cae el nodo maestro, se configura el esclavo para que deje de escuchar la réplica y asuma la responsabilidad del clúster *(maestro)*. Si cae el esclavo, entonces el nodo maestro deja de replicar la base de datos.

Nota: El sistema por sí mismo, detecta la caída del nodo, pero no es capaz de diagnosticar la caída. No sabría exactamente que fallo ocurrió más allá de si es de servicio o caída total del nodo.

2.4. Diseño de la propuesta de solución para el Sistema de Gestión Hospitalaria

En este epígrafe, se explica tanto el diseño lógico como físico alcanzado como propuesta para desarrollar la estrategia. Su objetivo principal es describir en detalle el comportamiento del sistema paso a paso desde diferentes vistas.

2.4.1. Diseño lógico

A continuación se muestra el diseño lógico del clúster y una explicación de cada una de sus funcionalidades.

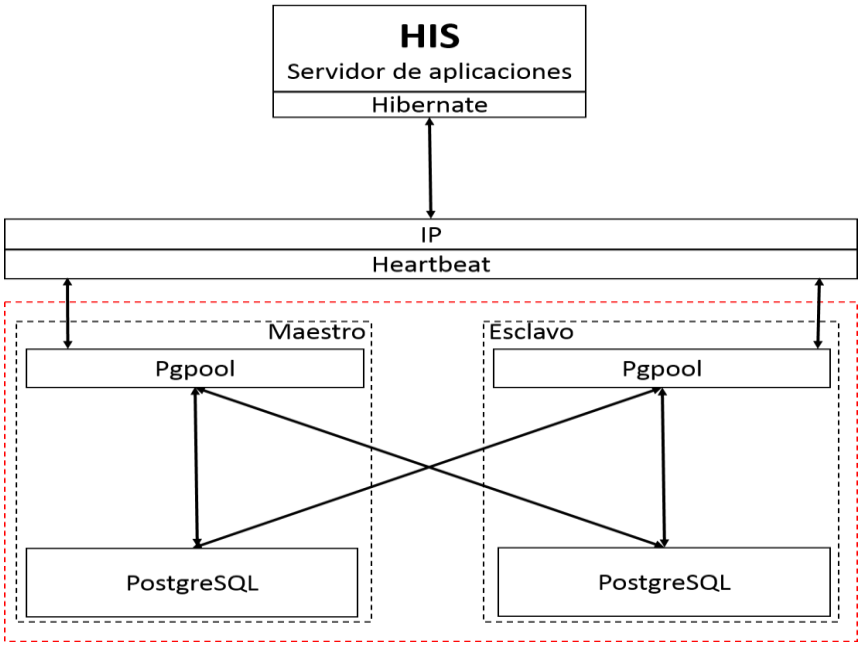


Figura 6: Diseño lógico.

Fuente: Elaboración propia.

La solución del clúster la integran dos balanceadores de carga, encargados de repartir las peticiones SQL entre las instancias de base de datos. Los balanceadores se alternan el tiempo de ejecución ya que uno de ellos es el respaldo para caso de fallas. Este balanceo se realiza ponderando cada nodo con un valor numérico donde el menor valor es el que menos consultas SQL procesa, en caso de ser igual el valor, tienen el mismo nivel de ocurrencia y las consultas se repartirían por igual entre ambos nodos.

Otra de las funcionalidades que el sistema experimenta es la de pool de conexiones que permite atender más usuarios de los que en realidad puede tener en ejecución el clúster. Para esto utiliza un algoritmo de cola donde las conexiones nuevas se ponen en espera y se van atendiendo en orden de llegada.

En cuanto a la réplica de los servidores, se utiliza la herramienta Pgpool que la realiza a nivel de SQL que replica en ambas direcciones la base de datos. En caso de fallas en el clúster el sistema es capaz de detectar su caída, sabría si es una caída de servicio o del nodo completo. A este sistema de fallas no le es posible automáticamente restablecer el nodo por la variabilidad de los errores. Para re-

establecer un nodo habría que repararlo *(la estrategia de la investigación no es capaz de realizar esta acción)* y entonces conectarlo al clúster.

2.4.2. Diseño físico

En aras de optimizar los recursos se ha trazado una estrategia de solución de dos nodos, en los cuales habrá una instancia de Pgpool, Heartbeat y PostgreSQL en cada uno. De esta forma ambos nodos serán esclavos y primarios al mismo tiempo. Una instancia de Pgpool balancea las cargas de su propio servidor y el del servidor adyacente.

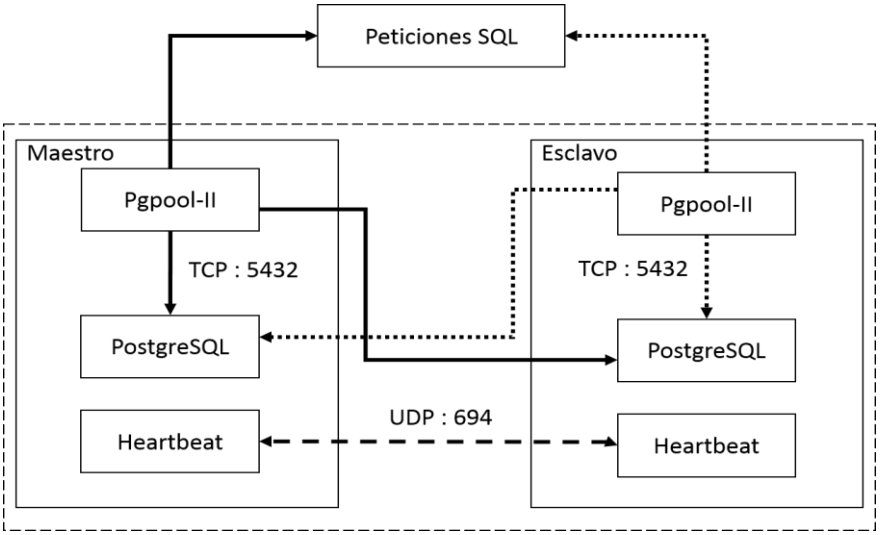


Figura 7: Diseño físico.

Fuente: Elaboración propia.

2.5. Aspectos de configuración de la estrategia de clúster para el Sistema de Gestión Hospitalaria

A continuación se presentan los pasos necesarios para la configuración del clúster en dos máquinas virtuales creadas con la herramienta Virtual Box.

2.5.1. Instalación de paquetes y dependencias

Lo primero es instalar PostgreSQL 9.1 y sus dependencias. Para ello, se ejecutan los siguientes comandos, en ambos nodos:

```
# apt-get update
# apt-get install libpq-dev postgresql-server-dev-9.1 bison build-essential
# apt-get install postgresql-9.1 postgresql-contrib-9.1 postgresql-doc-9.1 uuid libdbd-pg-perl
pgpool2
```

2.5.2. Configuración de PostgreSQL

Esto dejará en cada nodo una instalación funcional de PostgreSQL a la que se le hará unas pequeñas modificaciones en la configuración. El resto de valores que se modifican serán comunes a todos los nodos. Solo se modifica de momento:

```
listen_addresses = ‘192.168.56.2 192.168.56.3’
```

Se crea el usuario pgpool2:

```
# su postgres
$ createuser –-superuser pgpool2
```

Luego se edita el fichero `/etc/postgresql/9.3/main/pg_hba.conf` y se añade el acceso para los IPs de los nodos con el usuario postgres:

```
Host     postgres          all               192.168.56.2      md5

host     postgres          all               192.168.56.3      md5
```

Seguido se configura write ahead log, editando, `/etc/postgresql/9.3/main/postgresql.conf` y cambiamos los siguientes parámetros:

```
archive_mode = on
archive_command = ‘exit 0’
wal_level = archive
```

Después se crea el siguiente directorio en ambos nodos:

```
mkdir –mode=700 /data/pg_xlog_archive
chown postgres: postgres /data/pg_xlog_archive
```

Se reinicia PostgreSQL para activar los cambios con cualquiera de los comandos:

```
# /etc/init.d/postgresql restart
```

```
# service postgresql restart
```

Una vez instalado y configurado PostgreSQL en ambos nodos. Se instala Pgpool solamente en el nodo maestro.

```
# apt-get install pgpool2
# su postgres

$ psql -f /usr/local/src/pgpool2/sql/pgpool-recovery/pgpool-recovery.sql template1
```

Aquí, se carga "template1" que permite a las tablas una serie de funciones para tratar la réplica de Pgpool y viene por defecto con la instalación.

Para configurar Pgpool en el nodo maestro, se edita el fichero `/etc/pgpool2/pcp.conf` y se genera una clave para el usuario postgres con el siguiente comando:

```
/usr/sbin/pg_md5 "password de postgres" >> /etc/pgpool2/pcp.conf
```

Luego se agrega el usuario postgres que faltó por poner en el comando anterior quedando:

```
postgres: "clave que genera tu pasword"
```

Se configura el `/etc/pgpool2/pgool.conf` *(Anexo 1)*. Se reinicia para comprobar que arranca correctamente y se lista el status del script para evidenciar que todo funciona correctamente. Si se hace correctamente deberá mostrar algo así:

```
root@master:~# /etc/init.d/pgpool2 status
Status of pgpool-II:[ok]
```

2.5.3. Configuración de la disponibilidad

Para la alta disponibilidad se necesita, instalar Pgpool en el segundo nodo. Se cambiaran algunos valores en el fichero `pgpool.conf` del nodo esclavo:

Heartbeat se sirve de una IP de servicio sobre la IP real de los nodos. De modo que para la aplicación, el clúster solamente tendrá que dirigirse a esta IP de servicio puesto que se enmascaran los nodos. Por tanto, en cada nodo, se crea la IP que Heartbeat gestionará.

Antes de instalar Heartbeat y configurarlo, se detiene la red:

```
# service networking stop
```

Luego se modifica el fichero `/etc/network/interfaces` *(Anexo 3)*. Añadiendo una interfaz eth0:0.

A continuación, se modifica la configuración de Pgpool para que en lugar de escuchar en todas las interfaces, solamente escuche en la IP de servicio:

```
# Pasará de ser así
listen_addresses = '*'
# A ser así
listen_addresses = '192.168.56.4'
```

2.5.4. Instalación de Heartbeat:

```
# apt-get install heartbeat
```

Una vez instalado encontraremos los ficheros de configuración de Heartbeat en `/etc/ha.d`.

```
ha.cf: fichero de configuración principal.

haresources: fichero de configuración de recursos.

authkeys: información de autenticación.
```

También se añade lo siguiente al fichero `/etc/hosts`**:**

```
192.168.1.2 master
192.168.1.3 esclavo
192.168.1.4 pgpool
```

Si todo ha ido bien se crea una base de datos desde Pgpool:

```
# psql –h 192.168.56.4 –p 5432 –U postgres

postgres=$ create database testing;
```

Se verifica en cada uno de los nodos:

Nodo maestro

```
# su postgres

$ psql -l
```

Nodo esclavo

```
# su postgres

$ psql –l
```

Ambos comandos debe producir una salida idéntica.

2.6. Seguridad en el diseño de la solución

En cualquier sistema, la seguridad, es lo fundamental para la garantía de la calidad y la sostenibilidad del servicio. Es por principio, un aspecto primario en todo sistema, garantizar la confidencialidad, disponibilidad e integridad, los tres pilares de la seguridad informática.

Para proveer seguridad en el diseño de la solución se tuvo en cuenta los siguientes aspectos:

- ✓ Como primera medida de seguridad se propone el cifrado MD5 *(no es más que un algoritmo de codificación que produce un resultado de 128 bits que después se cifra)* para la autenticación de los usuarios Postgresql y Pgpool.

- ✓ Otra medida de seguridad que se tuvo en cuenta, fue configurar los ficheros de configuración de acceso de las herramientas y limitarlos a lo mínimo, dejando solo conexiones desde el nodo maestro y locales para evitar intromisiones.

- ✓ La utilización de un firewall personal es una garantía extra de confidencialidad. Específicamente en la investigación se utiliza Iptables. Filtrando las conexiones de ambos nodos.

- ✓ La utilización de la técnica de replicado permite que la información no tenga pérdidas, por lo que se tuvo presente para la confección de la solución. Esta técnica de clúster resuelve el problema de integralidad de la información.

A continuación se muestra el diseño de seguridad e interconexión que demuestra los puntos superiores:

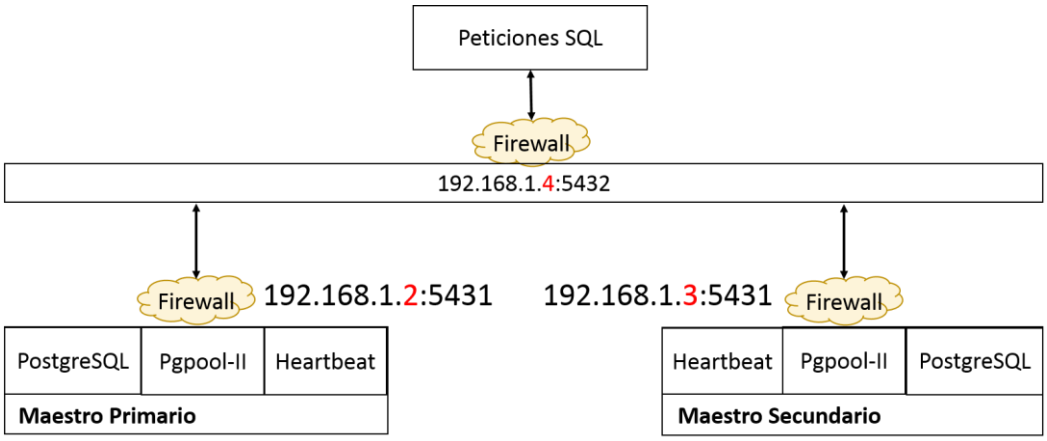


Figura 8: Diseño de seguridad e interconexión.

Fuente: Elaboración propia

Nota: La disponibilidad de la información que también es un pilar de la seguridad se tiene en cuenta desde el problema científico y se mitiga con la herramienta seleccionada en el capítulo uno de la investigación (epígrafe 1.11.2).

2.7. Conclusiones del capítulo

El estudio de la caracterización del entorno del Sistema de Gestión Hospitalaria posibilita la comprensión del funcionamiento de los módulos de este sistema, cómo se maneja el flujo de información entre ellos, y cómo está estructurada la arquitectura de los datos que en él se gestiona.

Con el análisis de las características de este sistema se da paso a la obtención de los requerimientos funcionales y no funcionales, sirviendo estos de apoyo para la elaboración del diseño físico y lógico, además de la comprensión de la necesidad de seguridad de la solución.

Capítulo 3

Validación de la propuesta de solución

3.1. Generalidades de las pruebas

Este capítulo tiene como propósito describir y analizar los resultados de las pruebas de rendimiento realizadas a la propuesta de solución. En el mismo se realiza un análisis de los resultados obtenidos en dichas pruebas mediante gráficas.

Para garantizar el correcto funcionamiento de cualquier sistema es necesario realizar un proceso de pruebas donde se demuestre que el mismo, está listo para su despliegue. Este proceso de pruebas, se realizó con el propósito de demostrar que la aplicación funciona correctamente, permitiendo su utilización, y por consecuencia una disminución en los tiempos de respuestas y disponibilidad del Sistema informático HIS.

Estas pruebas se realizan fundamentalmente con el objetivo:

- ✓ Demostrar que la estrategia de clúster es una solución eficiente para el Sistema Gestión Hospitalaria del CESIM

Durante la realización de las pruebas se estudió el comportamiento de las funcionalidades de la solución, que definen el comportamiento del clúster y puede afectar el correcto funcionamiento de la aplicación. Hay que tener en cuenta que las pruebas de carga en esta estrategia, no suponen un marcador de mejora, porque no existe la ejecución de consultas en paralelo por lo que para medir los tiempos de respuesta del clúster y comprobar su correcto funcionamiento se realizaron pruebas de rendimiento.

Las pruebas de rendimiento realizadas para la validación de la propuesta de solución se realizaron en ordenadores virtuales con el software Virtual Box (epígrafe 1.11.5) dotadas de un procesador a 1 GHz, una memoria RAM asignada de 512 Mb y 8 GB de disco duro.

Para generar carga en el clúster se utilizó el software Jmeter a través del cual se realizó la emulación de un número concurrentemente de usuarios navegando por el clúster. Luego de recogido los datos, los valores obtenidos en cada ordenador se promedian para obtener el consumo de recursos promedio de manera general para cada prueba realizada.

3.2. Pruebas de configuración al clúster

Validación de la propuesta de solución

Prueba de balance de carga

En esta prueba, se espera como resultado que el balanceador de carga, sea capaz de distribuir las peticiones entre los servidores de base de datos que componen el clúster.

Para la comprobación de la realización efectiva del balanceo de carga según la herramienta Pgpool, fue necesario la utilización de Jmeter (Anexo 5 - 7) para generar 10 usuarios navegando por el sistema un total de 10 veces, cada uno ejecutando dos consultas simples:

```
# select id from bancosangre.entrevista_medicamento;

# select id from bancosangre.laboratorio;
```

La revisión de las conexiones a cada nodo se realizó mediante el comando:

```
root@master~# tail –f /var/log/postgresql/postgresql.main.log    →        nodo maestro

root@slave~# tail –f /var/log/postgresql/postgresql.main.log     →        nodo esclavo
```

Como resultado de esta prueba se realizaron 200 peticiones al balanceador de las cuales 96 peticiones fueron atendidas por el nodo maestro y 104 fueron atendidas por el nodo esclavo.

Pruebas de réplicas de datos

Los resultados esperados por esta prueba luego de las inserciones realizadas por el front-end de Pgpool, sean replicadas en ambos nodos, es decir, que las consultas de escritura se ejecuten en ambos nodos.

Para la correcta comprobación de esta funcionalidad, se creó una nueva tabla en la base de datos por el front-end de Pgpool y se le introdujo datos ficticios:

```
psql –h 192.168.56.4 –p 5432 –U postgres

postgres#= create table test(id varchar, nombre varchar);

postgres#= Insert into test values (‘00001’,’Felo’);

postgres#= Insert into test values (‘00002’,’America’);
```

```
postgres#= Insert into test values (‘00003’,’Fefa’);
```

Luego, se comprobó en cada nodo por separado que la tabla se ha creado, arrojando ambas tablas como resultado de la comprobación y el buen desempeño ante la prueba:

Nodo maestro

```
psql –h 192.168.56.2 –p 5431 –U postgres

postgres=# select * from test;

 id | nombre

-------+---------

 00001 | Felo

 00002 | America

 00003 | Fefa

(3 rows)
```

Nodo esclavo

```
psql –h 192.168.56.3 –p 5431 –U postgres

postgres=# select * from test;

 id | nombre

-------+---------

 00001 | Felo

 00002 | America

 00003 | Fefa

(3 rows)
```

Pruebas de pool de conexiones

Las pruebas de pool de conexiones, no son más que la forma de comprobar que el servidor está aceptando más conexiones de las que puede gestionar.

Para llevar a cabo esta prueba fue necesario la utilización de Jmeter y la simulación en él de una cantidad de usuarios mayor a 100 (Anexo 8), que es la configuración por defecto de PostgreSQL. Para la comprobación no se utilizan comandos ni se registran las trazas, solamente se espera que el clúster sea capaz de gestionar más de lo que ofrece la configuración por defecto. Cuando se simularon los usuarios el clúster fue capaz de soportar 150 usuarios activos sin fines de sobrecargar el clúster.

Recuperación ante fallas

Para la realización de la comprobación de esta funcionalidad es necesario eliminar un nodo de clúster cuando esté en funcionamiento. Para esto se conectó el servidor de aplicaciones del HIS (JBoss AS) (Anexo 9) y mientras se navegaba por la aplicación se puso un nodo fuera de línea.

La aplicación tardó aproximadamente 3 segundos en obtener una respuesta del servicio de base de datos pero mantuvo la conexión con el servidor de aplicaciones en todo momento.

Análisis de los resultados

Durante el proceso de pruebas de configuración realizadas al clúster fueron probadas las funcionalidades:

- ✓ Balanceo de carga.
- ✓ Réplica de datos.
- ✓ Pool de conexiones.
- ✓ Recuperación ante fallas

Como se pudo observar durante las pruebas, el comportamiento obtenido por parte del sistema, en cada una de las funcionalidades probadas fue satisfactorio, se puede llegar a la conclusión que el diseño del clúster se ha validado funcionalmente.

3.3. Pruebas de comparación del rendimiento con la propuesta anterior

Para llevar a cabo esta prueba de comparación se utilizó un plan iterativo donde se escoge la respuesta menos óptima para los tiempos de respuestas del clúster de la investigación, aunque por lo general rondan los mismos resultados. Se utiliza como referencia la propuesta actual que se encuentra

desplegada en el HIS como comparativa de la estrategia de clúster. A continuación se muestra la tabla de resultados recogidos:

Nota: en el epígrafe 3.1 se especifican los requerimientos de las máquinas virtuales.

Tabla 2: Resultado de las pruebas de rendimiento

Métrica	Propuesta anterior	Estrategia de clúster
Uso de CPU (%)	84	75
Uso de memoria RAM (%)	92	17.6
Uso de la red (%)	17.0	6.1
Tiempo de respuesta promedio con 200 usuarios concurrentes (segundos)	25.0	3.0

La estrategia de clúster, demuestra tener un desempeño mejor que el del despliegue actual. Como se puede apreciar anteriormente *(Tabla 2)*, se muestra la diferencia en cuanto a tiempos de respuestas, uso de CPU, uso de memoria RAM, y uso de la red entre la solución convencional y la propuesta por la investigación. La solución de la investigación, maneja con efectividad los recursos de los que dispone mejorando la propuesta anterior.

Nota: Hay que tener en cuenta que las pruebas se hacen en un ambiente virtual donde las prestaciones de los nodos, casi están en el margen de los requerimientos mínimos, afectando los resultados de las pruebas.

3.4. Conclusiones parciales

Se realizaron varios tipos de pruebas para validar la propuesta de solución de clúster de la presente investigación, entre las que se encuentran, las pruebas de configuración y pruebas de rendimiento.

Las pruebas de configuración se desarrollaron con el objetivo de comprobar que el sistema es capaz de realizar las funcionalidades correspondientes *(epígrafe 2.2)*. Arrojando elementos fehacientes que comprueban la veracidad e impacto de la investigación.

Validación de la propuesta de solución

Las pruebas de rendimiento, se realizaron para evidenciar que la solución de la investigación, brinda una capacidad de rendimiento y disponibilidad mayor que el despliegue convencional. Se confirma con estas pruebas, que el rendimiento y la disponibilidad son factores claves en la investigación.

Conclusiones generales

La realización de la presente investigación científica ha posibilitado cumplir con el objetivo definido en el diseño de la investigación. Por lo que se arriba a las siguientes conclusiones:

- ✓ El estudio de las soluciones existentes de la tecnología de clúster de base de datos sirvió de guía para obtener el diseño de una propuesta de solución que permitió aumentar la capacidad de respuesta y la disponibilidad del Sistema de Gestión Hospitalaria del CESIM.

- ✓ El análisis de las herramientas y tecnologías existentes garantizó los elementos necesarios para una adecuada construcción de la estrategia de clúster para el Sistema de Gestión Hospitalaria.

- ✓ La culminación de la validación y verificación de la estrategia de clúster, y desde el fundamento teórico metodológico, brindan elementos suficientes e irrefutables que permiten afirmar que una estrategia de clúster de base de datos para el Sistema de Gestión Hospitalaria del Centro de Informática Médica es una solución capaz de gestionar con eficacia, transparencia, disponibilidad y rendimiento los recursos de la base de datos, garantizando la integridad del sistema como un todo.

Una vez cumplido los objetivos del presente trabajo y en correspondencia con los resultados obtenidos, que quedaron expuestos en el presente documento, se recomienda como trabajos futuros:

- ✓ Actualizar la herramienta Pgpool a la próxima versión que contendrá un módulo llamado watchdog que gestiona la disponibilidad y permitirá desechar la herramienta Heartbeat.
- ✓ Utilizar la técnica de particionado en las tablas que en un futuro lo requieran con ánimo de aumentar el rendimiento de la estrategia en general.
- ✓ Continuar el estudio de las diversas técnicas de clúster que surjan en el futuro, para aplicarlas y proponer modificaciones a la estrategia propuesta, con el objetivo de mejorar los medidores de eficiencia alcanzados.

Abarca, M. C. (24 de Noviembre de 2008). *wordpress*. Obtenido de http://xxito.files.wordpress.com/2008/11/trabajo-final-beowulf.pdf

access. (4 de Mayo de 2011). Obtenido de https://access.redhat.com/site/documentation/es-ES/Red_Hat_Enterprise_Linux/5/html/Cluster_Suite_Overview/ch.gfscs.cluster-overview-CSO.html

alegsa. (03 de Mayo de 2009). Obtenido de http://www.alegsa.com.ar/Dic/data%20center.php

Aliaga, A. I. (21 de Enero de 2008). *iessanvicente.* Obtenido de https://iessanvicente.com/colaboraciones/postgreSQL.pdf

apache. (2012). Obtenido de http://www.apache.org/licenses/

Aragón, J. A. (01 de Enero de 2014). *cala*. Obtenido de http://cala.unex.es/cala/epistemowikia/index.php?title=Arquitectura_alta_disponibilidad

Arboníes, A. L. (2000). *gobernabilidad.* Obtenido de http://www.gobernabilidad.cl/documentos/clustercono_desterri.pdf

Aycock, C. (14 de Julio de 2006). *insidehpc.* Obtenido de http://insidehpc.com/2006/07/14/what-is-a-clustered-database/

Barrios, J. D. (03 de Agosto de 2011). *alcancelibre*. Obtenido de http://www.alcancelibre.org/staticpages/index.php/introduccion-iptables

Beecher, V. V. (Julio de 2013). *mirrors.* Obtenido de http://mirrors.iyunwei.com/oracle/docs/11.2-E11882-01/server.112/e17157.pdf

bigfishgames. (s.f.). Obtenido de http://www.bigfishgames.es/

Bravo, J. A. (14 de Febrero de 2012). *scribd*. Obtenido de http://es.scribd.com/doc/81618208/Guia-para-la-Instalacion-de-un-cluster-de-bases-de-datos

Brien, D. (28 de Noviemre de 2007). *everac99*. Obtenido de http://everac99.wordpress.com/2007/11/28/el-oracle-rac-que-es-y-como-funciona/

Brien, D. (19 de Agosto de 2008). *everac99.* Obtenido de http://everac99.wordpress.com/2008/08/19/alta-disponibilidad-que-es-y-como-se-logra/

Calderón, E. A. (29 de Enero de 2011). *basesdedatosues*. Obtenido de http://basesdedatosues.blogspot.com/2010/06/replicacion-postgresql-slony-i.html

Caro, L. P. (2011). *tubasededatoslibre.* Obtenido de http://tubasededatoslibre.org/site/wp-content/uploads/2012/05/PonenciaLCaro.pdf

Castañeda, H. (2003). *edutec*, http://edutec.rediris.es/Revelec2/revelec21/jmontero.htm.

Castro, L. (2012). *aprenderinternet.* Obtenido de http://aprenderinternet.about.com/od/ConceptosBasico/g/Escalabilidad.htm

Chévez, A. (2012). *asteriscus.* Obtenido de http://asteriscus.com/presentacion/presentacion%20A71922.pdf

clusterinformatica.blogspot. (21 de Mayo de 2011). Obtenido de http://wwwclusterinformatica.blogspot.com/2011/05/cluster-informatica.html

configserver. (2014). Obtenido de http://www.configserver.com/

cujae. (Marzo de 2003). Obtenido de http://rii.cujae.edu.cu/index.php/revistaind/article/viewFile/204/188

cuore. (2010). Obtenido de http://www.cuore.es/download.php?p=descargas&f=Cinfa%2009.pdf

dbbalancer. (2011). Obtenido de http://dbbalancer.sourceforge.net/

debian. (08 de Diciembre de 2013). Obtenido de http://www.debian.org/

Delgado, A. J. (21 de Julio de 2013). *tubasededatoslibre*. Obtenido de http://tubasededatoslibre.org/site/bucardo-sincronizacion-asincrona/

Diaz, G. (2010). *webdelprofesor.* Obtenido de http://webdelprofesor.ula.ve/ingenieria/gilberto/paralela/05_LeyDeAmdahlYMoore.pdf

djangoproject. (s.f.). Obtenido de https://www.djangoproject.com/

Duran, C. (27 de Octubre de 2009). *slideshare.* Obtenido de http://www.slideshare.net/cduranmardones/trabajo-clusters

Durán, C. M. (15 de Octubre de 2009). *slideshare*. Obtenido de http://www.slideshare.net/cduranmardones/clusters-2293694

ehi. (2014). Obtenido de http://www.ehi.co.uk/news/primary-care/606%5D%5Bhttp://www.ehi.co.uk/news/primary-care/591

ehowenespanol. (s.f.). Obtenido de http://www.ehowenespanol.com/

eliax. (17 de Enero de 2012). Obtenido de http://www.eliax.com/index.cfm?post_id=9163

facebook. (s.f.). Obtenido de www.facebook.com/

Filippi, J. L. (2009). *sedici.* Obtenido de http://sedici.unlp.edu.ar/bitstream/handle/10915/4158/Documento_completo.pdf?sequence=1

Fleming. (12 de Octubre de 2004). *telegraph*. Obtenido de http://www.telegraph.co.uk/news/uknews/1473927/Bill-for-hi-tech-NHS-soars-to-20-billion.html

forat. (08 de Junio de 2010). Obtenido de http://www.forat.info/2010/06/08/balanceo-de-carga-entre-servidores-bajo-linux-debian-introduccion/

Germán, R. M. (2013). *Herramienta para la administración de cluster en servidores de aplicaciones JBoss.* Habana.

Gheorghiu, G. (25 de Febrero de 2005). *agiletesting.* Obtenido de http://agiletesting.blogspot.com/2005/02/performance-vs-load-vs-stress-testing.html

González, J. (Mayo de 2009). *eprints.* Obtenido de http://eprints.rclis.org/15422/1/Datacenters,%20Tendencias%20y%20Seguridad%20-%20Seccion%20Seguridad%20Publica.pdf

google. (s.f.). Obtenido de https://www.google.com.cu/?gws_rd=cr&ei=FYQ9U_OvJ9DIsATc_oGwBQ

hibernate. (2010). Obtenido de http://hibernate.org/orm/what-is-an-orm

hostingdiario. (20 de Febrero de 2013). Obtenido de http://hostingdiario.com/firewalls-para-servidores-linux-te-mostramos-las-mejores-opciones/

Ibañez, D. H. (2009). *unam.* Obtenido de http://www.ptolomeo.unam.mx:8080/xmlui/bitstream/handle/132.248.52.100/1182/Tesis.pdf?sequence=1

instagram. (s.f.). Obtenido de http://instagram.com

instagram-engineering. (s.f.). Obtenido de http://instagram-engineering.tumblr.com/

jakarta. (21 de Noviembre de 2011). Obtenido de https://jakarta.apache.org/

javaworld. (21 de Febrero de 2005). Obtenido de http://www.javaworld.com/article/2071844/java-web-development/on-the-road-to-simplicity.html

Javier. (2011). *in2test.* Obtenido de http://in2test.lsi.uniovi.es/gt26/presentations/ISO-29119-Javier-Tuya-AST-Seville-2011.pdf

jboss. (s.f.). Obtenido de http://www.jboss.org/jbossas/history

jbossas. (2012). Obtenido de http://jbossas.jboss.org/docs/

juntadeandalucia. (s.f.). Obtenido de http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/218

linalco. (2004). Obtenido de http://www.linalco.com/hpcc-cluster-de-calculo-alto-rendimiento-linux.html

lintips. (02 de Febrero de 2010). Obtenido de http://www.lintips.com/?q=node/119

linux-ha. (28 de Febrero de 2011). Obtenido de http://www.linux-ha.org/wiki/Main_Page

Lugo, L. I. (2011). *uprm.* Obtenido de http://www.uprm.edu/cti/docs/manuales/manuales-espanol/vax-vms/manuales/vms.pdf

lunenfeld. (1997). Obtenido de http://lunenfeld.ca/

Maribel. (21 de Marzo de 2012). *estudioteca*. Obtenido de http://www.estudioteca.net/universidad/telecomunicaciones/gestor-base-datos/

Martínez, I. F. (Agosto de 2004). *digeset.* Obtenido de http://digeset.ucol.mx/tesis_posgrado/Pdf/Felix_Ortigosa_Martinez.pdf

metanotion. (28 de Septiembre de 2006). Obtenido de http://www.metanotion.net/software/sandbox/block.html

Moreno, A. R. (Marzo de 2010). *upm.* Obtenido de http://oa.upm.es/4958/2/PFC_ALBERTO_MORENO_RAMIREZ.pdf

Moreno, J. (Febrero de 2013). *slideshare.* Obtenido de http://www.slideshare.net/jmorenol/clsters-alta-disponibilidad

Mustain, A. E. (18 de Noviembre de 2004). *onlamp*. Obtenido de http://www.onlamp.com/pub/a/onlamp/2004/11/18/slony.html

mysql. (2011). Obtenido de http://www.mysql.com/products/

Niclausse, N. (Febrero de 2013). *tsung.erlang-project*. Obtenido de http://tsung.erlang-projects.org/

Oberto, P. (24 de Marzo de 2012). *modelosbd2012t1*. Obtenido de http://modelosbd2012t1.wordpress.com/2012/03/24/base-de-datos-paralelas/

oracle. (Septiembre de 2009). Obtenido de http://www.oracle.com/technetwork/es/database/clustering/documentation/real-application-clusters-11gr2-1705079-esa.pdf

oracle. (2009). Obtenido de http://www.oracle.com/index.html

oracleclusters. (2014). Obtenido de http://pointer-oracleclusters.wikispaces.com/

Paulo. (02 de Octubre de 2010). *lintips*. Obtenido de http://www.lintips.com/?q=node/117

pgadmin. (2010). Obtenido de http://www.pgadmin.org/index.php

pgfoundry. (28 de Noviembre de 2012). Obtenido de http://pgfoundry.org/projects/pgbouncer

pgpool. (2010). Obtenido de http://www.pgpool.net/mediawiki/index.php/Main_Page

pgpool. (19 de Diciembre de 2013). Obtenido de http://www.pgpool.net/mediawiki/index.php/Main_Page

plproxy. (13 de Noviembre de 2011). Obtenido de http://plproxy.projects.pgfoundry.org/doc/tutorial.html

postgresql. (2009). Obtenido de http://www.postgresql.org/docs/devel/static/pgbench.html

postgresql. (12 de Septiembre de 2011). Obtenido de http://www.postgresql.org/about/press/presskit91/es/

postgresql. (2012). Obtenido de http://www.postgresql.org/docs/devel/static/pgbench.html

postgresql. (11 de Julio de 2013). Obtenido de http://wiki.postgresql.org/wiki/Tungsten

Pressman, R. (1993). *La ingeniería de software.*

rafaelma. (17 de Julio de 2009). *postgresql*. Obtenido de http://www.postgresql.org.es/node/313?page=1

Ramírez, A. M. (Marzo de 2010). *upm.* Obtenido de http://oa.upm.es/4958/2/PFC_ALBERTO_MORENO_RAMIREZ.pdf

Ramírez, R. Z. (2008). *Sistemas Gestore de Base de Datos.*

redhat. (2014). Obtenido de http://www.redhat.com/products/jbossenterprisemiddleware/

Reynier Manuel Germán Agüero, J. M. (2013). *Herramienta para la administración de cluster en servidores de aplicaciones JBoss.* Habana.

Rodriguez, L. (2001). *docencia.* Obtenido de http://docencia.lbd.udc.es/bdd-grao/teoria/tema1/1.3-IntroduccionALasBDsDocumentales.pdf

Schumacher, A. &. (2008). *oooes.* Obtenido de http://oooes.org/archivos/Bases_de_datos_libres.pdf

seamframework. (s.f.). Obtenido de http://www.seamframework.org/Documentation

sergioalmendros. (s.f.). Obtenido de http://sergioalmendros.host56.com/CLASIFICACION%20CLUSTERS.html

severalnines. (01 de Mayo de 2013). Obtenido de http://www.severalnines.com/news/article/database-clustering/gaming-company-turns-to-mysql-cluster-for-database-solutions/801580022

Shrivastava, T. (16 de Diciembre de 2013). *tecmint*. Obtenido de http://www.tecmint.com/open-source-security-firewalls-for-linux-systems/

slony. (2010). Obtenido de http://slony.info/

Sommervile, I. (2005). *Ingerniería del Softeare.*

sqlmanager. (1999). Obtenido de http://www.sqlmanager.net/

sqlmanager. (1999). Obtenido de http://www.sqlmanager.net/en/support/faq/licensing

suprabt. (2011). Obtenido de http://www.suprabt.com/proyectos_a.html

technet. (2014). Obtenido de http://technet.microsoft.com/en-us/library/hh270278.aspx

thewire. (17 de Enero de 2013). Obtenido de http://www.thewire.com/technology/2013/01/how-many-users-does-instagram-have/61139/

upct. (Octubre de 2011). Obtenido de http://ocw.bib.upct.es/mod/resource/view.php?id=9853&redirect=1

usenix. (2014). Obtenido de https://www.usenix.org/conference/als-2000/sequence-analysis-216-processor-beowulf-cluster

windows. (2012). Obtenido de http://windows.microsoft.com/es-es/windows/home

wordpress. (27 de Julio de 2007). Obtenido de http://kaiv.wordpress.com/2007/07/27/postgresql-cluster-partitioning-with-plproxy-part-i/

Yáñez, R. M. (2011). *Introducción a las tecnologías de clustering en GNU/Linux.*

Abarca, M. C. (24 de Noviembre de 2008). *wordpress*. Obtenido de http://xxito.files.wordpress.com/2008/11/trabajo-final-beowulf.pdf

access. (4 de Mayo de 2011). Obtenido de https://access.redhat.com/site/documentation/es-ES/Red_Hat_Enterprise_Linux/5/html/Cluster_Suite_Overview/ch.gfscs.cluster-overview-CSO.html

alegsa. (03 de Mayo de 2009). Obtenido de http://www.alegsa.com.ar/Dic/data%20center.php

Aliaga, A. I. (21 de Enero de 2008). *iessanvicente.* Obtenido de https://iessanvicente.com/colaboraciones/postgreSQL.pdf

apache. (2012). Obtenido de http://www.apache.org/licenses/

Aragón, J. A. (01 de Enero de 2014). *cala*. Obtenido de http://cala.unex.es/cala/epistemowikia/index.php?title=Arquitectura_alta_disponibilidad

Arboníes, A. L. (2000). *gobernabilidad.* Obtenido de http://www.gobernabilidad.cl/documentos/clustercono_desterri.pdf

Aycock, C. (14 de Julio de 2006). *insidehpc.* Obtenido de http://insidehpc.com/2006/07/14/what-is-a-clustered-database/

Barrios, J. D. (03 de Agosto de 2011). *alcancelibre*. Obtenido de http://www.alcancelibre.org/staticpages/index.php/introduccion-iptables

Beecher, V. V. (Julio de 2013). *mirrors.* Obtenido de http://mirrors.iyunwei.com/oracle/docs/11.2-E11882-01/server.112/e17157.pdf

bigfishgames. (s.f.). Obtenido de http://www.bigfishgames.es/

Bravo, J. A. (14 de Febrero de 2012). *scribd*. Obtenido de http://es.scribd.com/doc/81618208/Guia-para-la-Instalacion-de-un-cluster-de-bases-de-datos

Brien, D. (28 de Noviemre de 2007). *everac99*. Obtenido de http://everac99.wordpress.com/2007/11/28/el-oracle-rac-que-es-y-como-funciona/

Brien, D. (19 de Agosto de 2008). *everac99.* Obtenido de http://everac99.wordpress.com/2008/08/19/alta-disponibilidad-que-es-y-como-se-logra/

Calderón, E. A. (29 de Enero de 2011). *basesdedatosues*. Obtenido de http://basesdedatosues.blogspot.com/2010/06/replicacion-postgresql-slony-i.html

Caro, L. P. (2011). *tubasededatoslibre.* Obtenido de http://tubasededatoslibre.org/site/wp-content/uploads/2012/05/PonenciaLCaro.pdf

Castañeda, H. (2003). *edutec*, http://edutec.rediris.es/Revelec2/revelec21/jmontero.htm.

Castro, L. (2012). *aprenderinternet.* Obtenido de http://aprenderinternet.about.com/od/ConceptosBasico/g/Escalabilidad.htm

Chévez, A. (2012). *asteriscus.* Obtenido de http://asteriscus.com/presentacion/presentacion%20A71922.pdf

clusterinformatica.blogspot. (21 de Mayo de 2011). Obtenido de http://wwwclusterinformatica.blogspot.com/2011/05/cluster-informatica.html

configserver. (2014). Obtenido de http://www.configserver.com/

cujae. (Marzo de 2003). Obtenido de http://rii.cujae.edu.cu/index.php/revistaind/article/viewFile/204/188

cuore. (2010). Obtenido de http://www.cuore.es/download.php?p=descargas&f=Cinfa%2009.pdf

dbbalancer. (2011). Obtenido de http://dbbalancer.sourceforge.net/

debian. (08 de Diciembre de 2013). Obtenido de http://www.debian.org/

Delgado, A. J. (21 de Julio de 2013). *tubasededatoslibre*. Obtenido de http://tubasededatoslibre.org/site/bucardo-sincronizacion-asincrona/

Diaz, G. (2010). *webdelprofesor.* Obtenido de http://webdelprofesor.ula.ve/ingenieria/gilberto/paralela/05_LeyDeAmdahlYMoore.pdf

djangoproject. (s.f.). Obtenido de https://www.djangoproject.com/

Duran, C. (27 de Octubre de 2009). *slideshare.* Obtenido de http://www.slideshare.net/cduranmardones/trabajo-clusters

Durán, C. M. (15 de Octubre de 2009). *slideshare*. Obtenido de http://www.slideshare.net/cduranmardones/clusters-2293694

ehi. (2014). Obtenido de http://www.ehi.co.uk/news/primary-care/606%5D%5Bhttp://www.ehi.co.uk/news/primary-care/591

ehowenespanol. (s.f.). Obtenido de http://www.ehowenespanol.com/

eliax. (17 de Enero de 2012). Obtenido de http://www.eliax.com/index.cfm?post_id=9163

facebook. (s.f.). Obtenido de www.facebook.com/

Filippi, J. L. (2009). *sedici.* Obtenido de http://sedici.unlp.edu.ar/bitstream/handle/10915/4158/Documento_completo.pdf?sequence=1

Fleming. (12 de Octubre de 2004). *telegraph*. Obtenido de http://www.telegraph.co.uk/news/uknews/1473927/Bill-for-hi-tech-NHS-soars-to-20-billion.html

forat. (08 de Junio de 2010). Obtenido de http://www.forat.info/2010/06/08/balanceo-de-carga-entre-servidores-bajo-linux-debian-introduccion/

Germán, R. M. (2013). *Herramienta para la administración de cluster en servidores de aplicaciones JBoss.* Habana.

Gheorghiu, G. (25 de Febrero de 2005). *agiletesting.* Obtenido de http://agiletesting.blogspot.com/2005/02/performance-vs-load-vs-stress-testing.html

González, J. (Mayo de 2009). *eprints.* Obtenido de http://eprints.rclis.org/15422/1/Datacenters,%20Tendencias%20y%20Seguridad%20-%20Seccion%20Seguridad%20Publica.pdf

google. (s.f.). Obtenido de https://www.google.com.cu/?gws_rd=cr&ei=FYQ9U_OvJ9DIsATc_oGwBQ

hibernate. (2010). Obtenido de http://hibernate.org/orm/what-is-an-orm

hostingdiario. (20 de Febrero de 2013). Obtenido de http://hostingdiario.com/firewalls-para-servidores-linux-te-mostramos-las-mejores-opciones/

Ibañez, D. H. (2009). *unam.* Obtenido de http://www.ptolomeo.unam.mx:8080/xmlui/bitstream/handle/132.248.52.100/1182/Tesis.pdf?sequence=1

instagram. (s.f.). Obtenido de http://instagram.com

instagram-engineering. (s.f.). Obtenido de http://instagram-engineering.tumblr.com/

jakarta. (21 de Noviembre de 2011). Obtenido de https://jakarta.apache.org/

javaworld. (21 de Febrero de 2005). Obtenido de http://www.javaworld.com/article/2071844/java-web-development/on-the-road-to-simplicity.html

Javier. (2011). *in2test.* Obtenido de http://in2test.lsi.uniovi.es/gt26/presentations/ISO-29119-Javier-Tuya-AST-Seville-2011.pdf

jboss. (s.f.). Obtenido de http://www.jboss.org/jbossas/history

jbossas. (2012). Obtenido de http://jbossas.jboss.org/docs/

juntadeandalucia. (s.f.). Obtenido de http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/218

linalco. (2004). Obtenido de http://www.linalco.com/hpcc-cluster-de-calculo-alto-rendimiento-linux.html

lintips. (02 de Febrero de 2010). Obtenido de http://www.lintips.com/?q=node/119

linux-ha. (28 de Febrero de 2011). Obtenido de http://www.linux-ha.org/wiki/Main_Page

Lugo, L. I. (2011). *uprm.* Obtenido de http://www.uprm.edu/cti/docs/manuales/manuales-espanol/vax-vms/manuales/vms.pdf

lunenfeld. (1997). Obtenido de http://lunenfeld.ca/

Maribel. (21 de Marzo de 2012). *estudioteca*. Obtenido de http://www.estudioteca.net/universidad/telecomunicaciones/gestor-base-datos/

Martínez, I. F. (Agosto de 2004). *digeset.* Obtenido de http://digeset.ucol.mx/tesis_posgrado/Pdf/Felix_Ortigosa_Martinez.pdf

metanotion. (28 de Septiembre de 2006). Obtenido de http://www.metanotion.net/software/sandbox/block.html

Moreno, A. R. (Marzo de 2010). *upm.* Obtenido de http://oa.upm.es/4958/2/PFC_ALBERTO_MORENO_RAMIREZ.pdf

Moreno, J. (Febrero de 2013). *slideshare.* Obtenido de http://www.slideshare.net/jmorenol/clsters-alta-disponibilidad

Mustain, A. E. (18 de Noviembre de 2004). *onlamp*. Obtenido de http://www.onlamp.com/pub/a/onlamp/2004/11/18/slony.html

mysql. (2011). Obtenido de http://www.mysql.com/products/

Niclausse, N. (Febrero de 2013). *tsung.erlang-project*. Obtenido de http://tsung.erlang-projects.org/

Oberto, P. (24 de Marzo de 2012). *modelosbd2012t1*. Obtenido de http://modelosbd2012t1.wordpress.com/2012/03/24/base-de-datos-paralelas/

oracle. (Septiembre de 2009). Obtenido de http://www.oracle.com/technetwork/es/database/clustering/documentation/real-application-clusters-11gr2-1705079-esa.pdf

oracle. (2009). Obtenido de http://www.oracle.com/index.html

oracleclusters. (2014). Obtenido de http://pointer-oracleclusters.wikispaces.com/

Paulo. (02 de Octubre de 2010). *lintips*. Obtenido de http://www.lintips.com/?q=node/117

pgadmin. (2010). Obtenido de http://www.pgadmin.org/index.php

pgfoundry. (28 de Noviembre de 2012). Obtenido de http://pgfoundry.org/projects/pgbouncer

pgpool. (2010). Obtenido de http://www.pgpool.net/mediawiki/index.php/Main_Page

pgpool. (19 de Diciembre de 2013). Obtenido de http://www.pgpool.net/mediawiki/index.php/Main_Page

plproxy. (13 de Noviembre de 2011). Obtenido de http://plproxy.projects.pgfoundry.org/doc/tutorial.html

postgresql. (2009). Obtenido de http://www.postgresql.org/docs/devel/static/pgbench.html

postgresql. (12 de Septiembre de 2011). Obtenido de http://www.postgresql.org/about/press/presskit91/es/

postgresql. (2012). Obtenido de http://www.postgresql.org/docs/devel/static/pgbench.html

postgresql. (11 de Julio de 2013). Obtenido de http://wiki.postgresql.org/wiki/Tungsten

Pressman, R. (1993). *La ingeniería de software.*

rafaelma. (17 de Julio de 2009). *postgresql*. Obtenido de http://www.postgresql.org.es/node/313?page=1

Ramírez, A. M. (Marzo de 2010). *upm.* Obtenido de http://oa.upm.es/4958/2/PFC_ALBERTO_MORENO_RAMIREZ.pdf

Ramírez, R. Z. (2008). *Sistemas Gestore de Base de Datos.*

redhat. (2014). Obtenido de http://www.redhat.com/products/jbossenterprisemiddleware/

Reynier Manuel Germán Agüero, J. M. (2013). *Herramienta para la administración de cluster en servidores de aplicaciones JBoss.* Habana.

Rodriguez, L. (2001). *docencia.* Obtenido de http://docencia.lbd.udc.es/bdd-grao/teoria/tema1/1.3-IntroduccionALasBDsDocumentales.pdf

Schumacher, A. &. (2008). *oooes.* Obtenido de http://oooes.org/archivos/Bases_de_datos_libres.pdf

seamframework. (s.f.). Obtenido de http://www.seamframework.org/Documentation

sergioalmendros. (s.f.). Obtenido de http://sergioalmendros.host56.com/CLASIFICACION%20CLUSTERS.html

severalnines. (01 de Mayo de 2013). Obtenido de http://www.severalnines.com/news/article/database-clustering/gaming-company-turns-to-mysql-cluster-for-database-solutions/801580022

Shrivastava, T. (16 de Diciembre de 2013). *tecmint*. Obtenido de http://www.tecmint.com/open-source-security-firewalls-for-linux-systems/

slony. (2010). Obtenido de http://slony.info/

Sommervile, I. (2005). *Ingerniería del Softeare.*

sqlmanager. (1999). Obtenido de http://www.sqlmanager.net/

sqlmanager. (1999). Obtenido de http://www.sqlmanager.net/en/support/faq/licensing

suprabt. (2011). Obtenido de http://www.suprabt.com/proyectos_a.html

technet. (2014). Obtenido de http://technet.microsoft.com/en-us/library/hh270278.aspx

thewire. (17 de Enero de 2013). Obtenido de http://www.thewire.com/technology/2013/01/how-many-users-does-instagram-have/61139/

upct. (Octubre de 2011). Obtenido de http://ocw.bib.upct.es/mod/resource/view.php?id=9853&redirect=1

usenix. (2014). Obtenido de https://www.usenix.org/conference/als-2000/sequence-analysis-216-processor-beowulf-cluster

windows. (2012). Obtenido de http://windows.microsoft.com/es-es/windows/home

wordpress. (27 de Julio de 2007). Obtenido de http://kaiv.wordpress.com/2007/07/27/postgresql-cluster-partitioning-with-plproxy-part-i/

Yáñez, R. M. (2011). *Introducción a las tecnologías de clustering en GNU/Linux.*

Anexo 1

```
#------------#

# CONNECTIONS #

#------------#

listen_addresses = 'localhost'

port = 5432

socket_dir = '/var/run/postgresql'

pcp_port = 9898

pcp_socket_dir = '/var/run/postgresql'

#---------#

# BACKEND #

#---------#

backend_hostname0 = '192.168.56.2'

backend_port0 = 5431

backend_weight0 = 0

backend_data_directory0 = '/var/lib/postgresql/9.1/main'

backend_flag0 = 'ALLOW_TO_FAILOVER'

backend_hostname1 = '192.168.56.3'

backend_port1 = 5431

backend_weight1 = 1

backend_data_directory1 = '/var/lib/postgresql/9.1/main'

backend_flag1 = 'ALLOW_TO_FAILOVER'

#------#

# AUTH #

#------#
```

```
enable_pool_hba = off

pool_passwd = ''

authentication_timeout = 60

#-----#

# SSL #

#-----#

ssl = off

#-------#

# POOLS #

#-------#

num_init_children = 32

max_pool = 4

child_life_time = 300

child_max_connections = 0

connection_life_time = 0

client_idle_limit = 0

#------#

# LOGS #

#------#

log_destination = 'stderr'

print_timestamp = on

log_connections = on

log_hostname = off

log_statement = on

log_per_node_statement = on

log_standby_delay = 'none'
```

```
syslog_facility = 'LOCAL0'

syslog_ident = 'pgpool'

debug_level = 0

#----------------#

# FILE LOCATIONS #

#----------------#

pid_file_name = '/var/run/postgresql/pgpool.pid'

logdir = '/var/log/postgresql'

#--------------------#

# CONNECTION POOLING #

#--------------------#

connection_cache = on

reset_query_list = 'ABORT; DISCARD ALL'

#------------------#

# REPLICATION MODE #

#------------------#

replication_mode = on

replicate_select = off

insert_lock = on

lobj_lock_table = ''

replication_stop_on_mismatch = off

failover_if_affected_tuples_mismatch = off

#---------------------#

# LOAD BALANCING MODE #

#---------------------#

load_balance_mode = on
```

```
ignore_leading_white_space = on

white_function_list = ''

black_function_list = 'nextval,setval'

#------------------#

# MASTER/SLAVE MODE #

#------------------#

master_slave_mode = on

master_slave_sub_mode = 'slony'

sr_check_period = 0

sr_check_user = 'pgpool'

sr_check_password = 'pgpool'

delay_threshold = 0

follow_master_command = ''

#------------------------------#

# PARALLEL MODE AND QUERY CACHE #

#------------------------------#

parallel_mode = off

enable_query_cache = off

pgpool2_hostname = ''

system_db_hostname = 'localhost'

system_db_port = 5432

system_db_dbname = 'pgpool'

system_db_schema = 'pgpool_catalog'

system_db_user = 'pgpool'

system_db_password = ''

#-------------#
```

```
# HEALTH CHECK #

#--------------#

health_check_period = 0

health_check_timeout = 20

health_check_user = 'pgpool'

health_check_password = 'pgpool'

#----------------------#

# FAILOVER AND FAILBACK #

#----------------------#

failover_command = '/var/lib/postgresql/bin/failover.sh %d %M %m'

failback_command = ''

fail_over_on_backend_error = on

#-----------------#

# ONLINE RECOVERY #

#-----------------#

recovery_user = 'nobody'

recovery_password = ''

recovery_1st_stage_command = ''

recovery_2nd_stage_command = ''

recovery_timeout = 90

client_idle_limit_in_recovery = 0
```

Anexo 2

```
# – Backend Connection Settings -
backend_hostname0 = ’192.168.56.3’
```

```
backend_port0 = 5431
backend_weight0 = 1
backend_data_directory0 = ‘/var/lib/postgresql/9.1/main’
#backend_flag0 = ‘ALLOW_TO_FAILOVER’
backend_hostname1 = ’192.168.56.2’
backend_port1 = 5431
backend_weight1 = 1
backend_data_directory1 = ‘/var/lib/postgresql/9.1/main’
#backend_flag1 = ‘ALLOW_TO_FAILOVER’
# …
pgpool2_hostname = ‘slave’
```

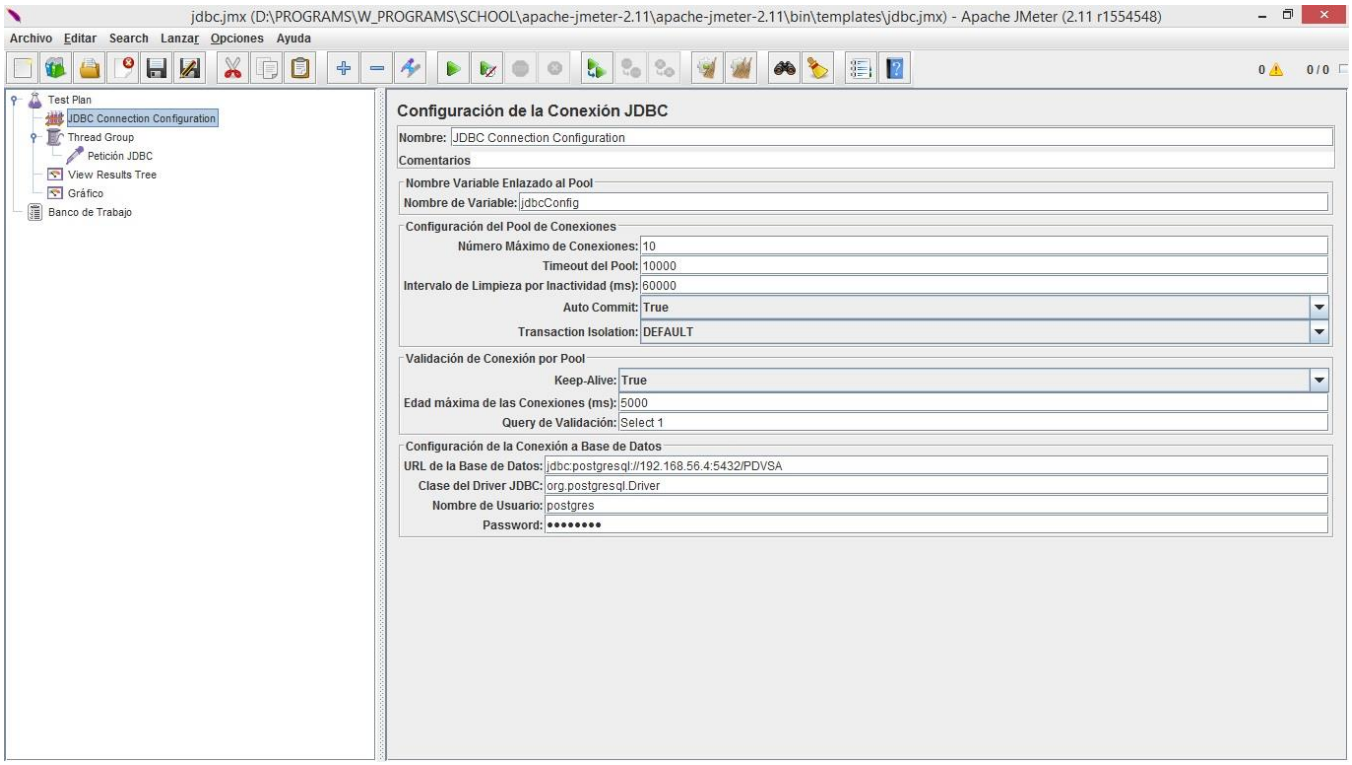
Anexo 3

```
# The loopback network interface
auto lo bond0
iface lo inet loopback
# The primary network interface
allow-hotplug eth0
allow-hotplug eth1
iface bond0 inet static
 slaves eth0 eth1
 address 192.168.56.2
 netmask 255.255.255.0
 gateway 192.168.56.1
 network 192.168.56.0
 broadcast 192.168.56.255
 bond_mode active-backup
 bond_miimon 100
 bond_downdelay 150
iface bond0:0 inet static
 address 192.168.56.4
 netmask 255.255.255.0
 network 192.168.56.0
 broadcast 192.168.56.255
```

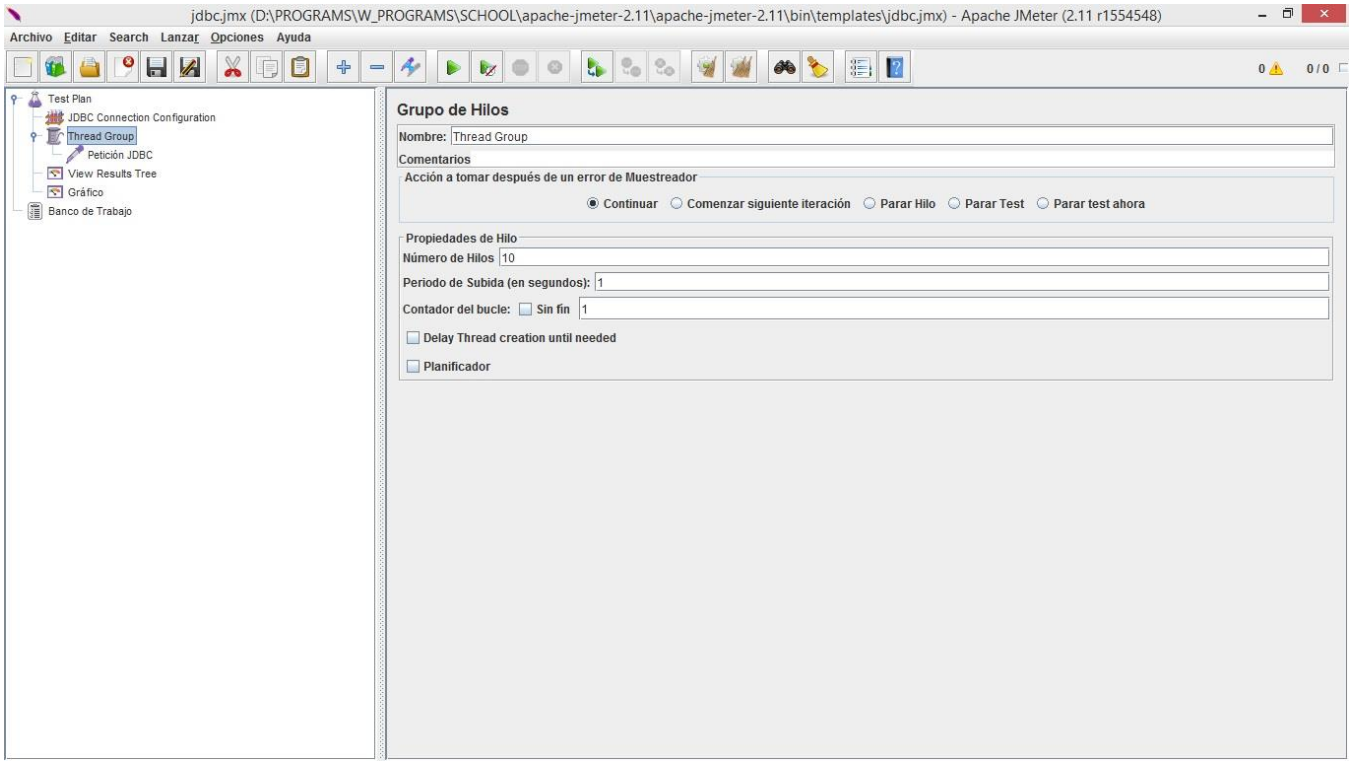
Anexo 4

```
# The loopback network interface
auto lo
iface lo inet loopback
# The primary network interface
auto eth0
iface eth0 inet static
 address 192.168.56.2
 network 192.168.56.0
 netmask 255.255.255.0
 gateway 192.168.56.1
iface eth0:0 inet static
 address 192.168.56.4
 network 192.168.56.0
 netmask 255.255.255.0
 broadcast 192.168.56.255
```

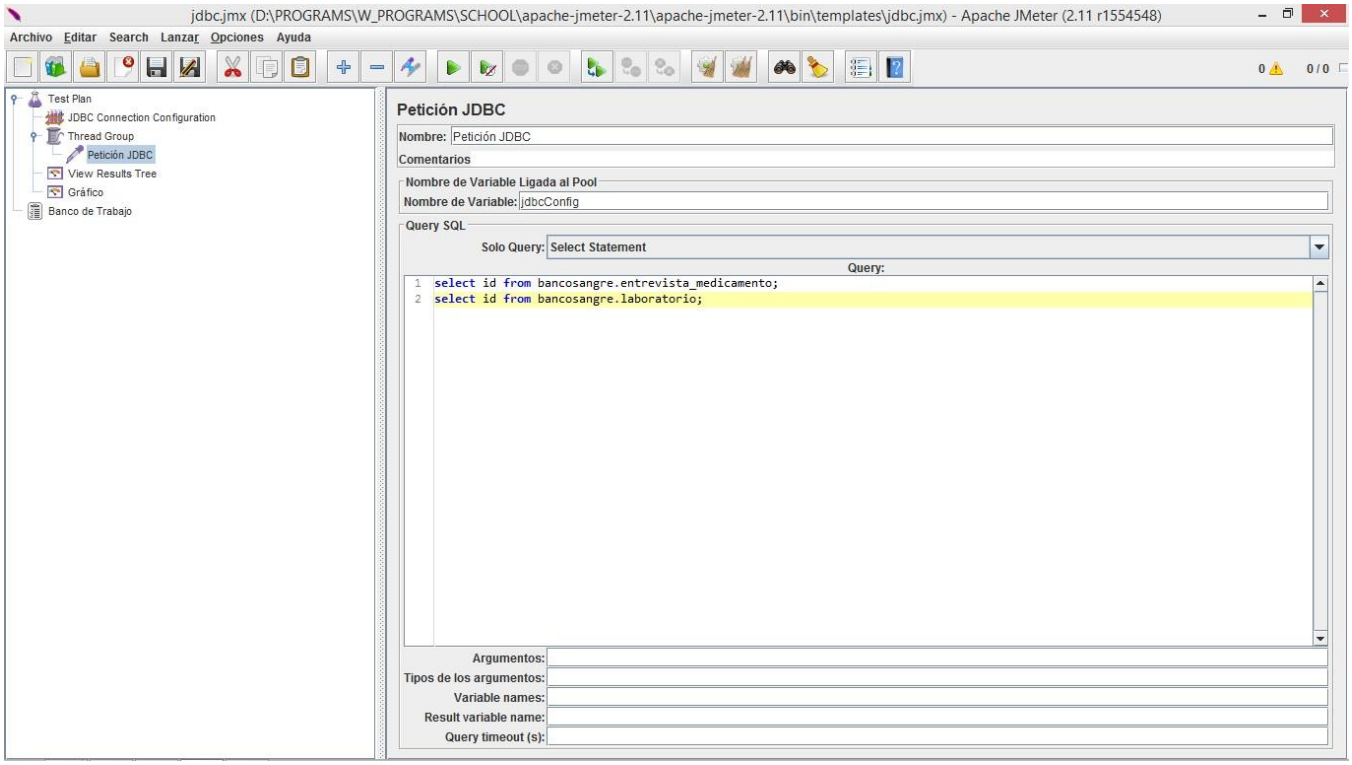
Anexo 5



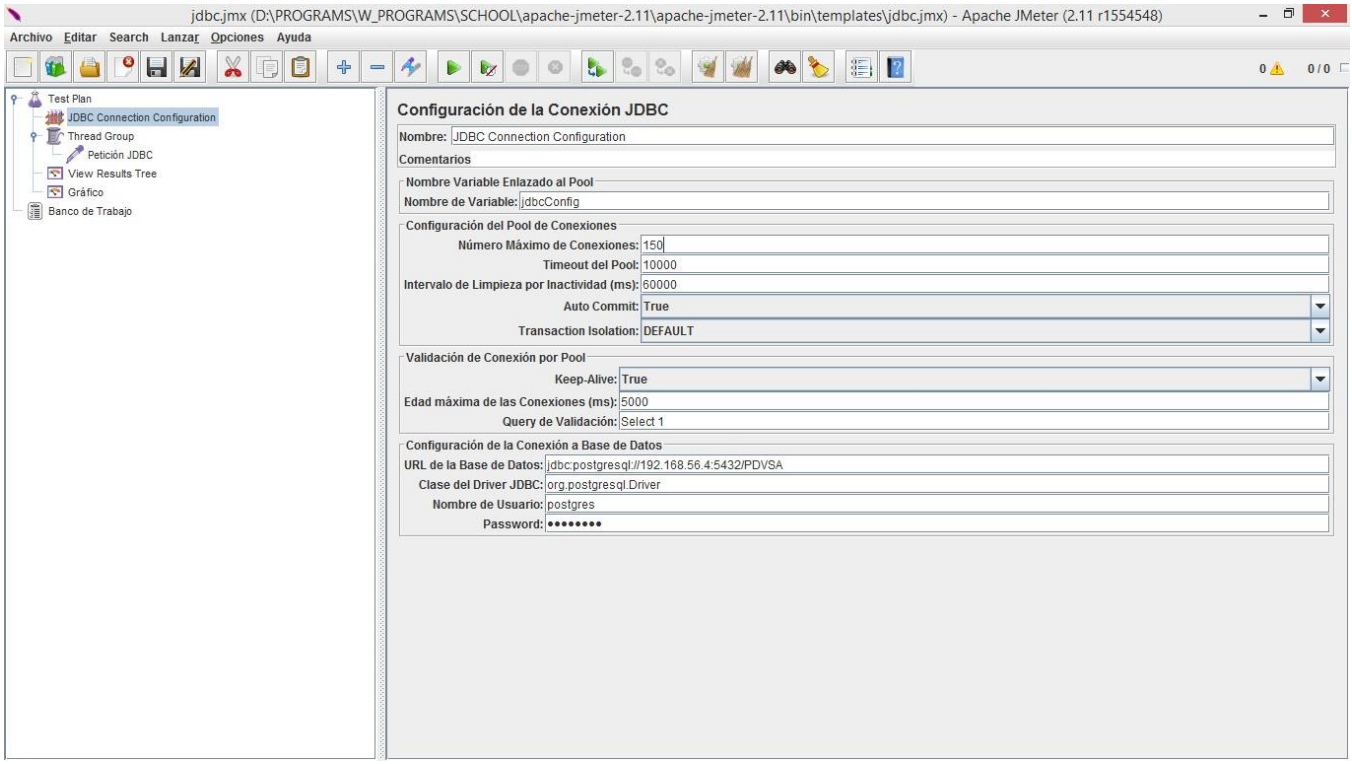
Anexo 6


jdbc.jmx (D:\PROGRAMS\W_PROGRAMS\SCHOOL\apache-jmeter-2.11\apache-jmeter-2.11\bin\templates\jdbc.jmx) - Apache JMeter (2.11 r1554548)
Archivo Editar Search Lanzar Opciones Ayuda
Test Plan
JDBC Connection Configuration
Thread Group
Petición JDBC
View Results Tree
Gráfico
Banco de Trabajo
Grupo de Hilos
Nombre: Thread Group
Comentarios
Acción a tomar después de un error de Muestreador
Continuar
Comenzar siguiente iteración
Parar Hilo
Parar Test
Parar test ahora
Propiedades de Hilo
Número de Hilos 10
Periodo de Subida (en segundos): 1
Contador del bucle: Sin fin 1
Delay Thread creation until needed
Planificador

Anexo 7


jdbc.jmx (D:\PROGRAMS\W_PROGRAMS\SCHOOL\apache-jmeter-2.11\apache-jmeter-2.11\bin\templates\jdbc.jmx) - Apache JMeter (2.11 r1554548)
Archivo Editar Search Lanzar Opciones Ayuda
Test Plan
JDBC Connection Configuration
Thread Group
Petición JDBC
View Results Tree
Gráfico
Banco de Trabajo
Petición JDBC
Nombre: Petición JDBC
Comentarios
Nombre de Variable Ligada al Pool
Nombre de Variable: jdbcConfig
Query SQL
Solo Query: Select Statement
Query:
1 select id from bancosangre.entrevista_medicamento;
2 select id from bancosangre.laboratorio;
Argumentos:
Tipos de los argumentos:
Variable names:
Result variable name:
Query timeout (s):

Anexo 8


jdbc.jmx (D:\PROGRAMS\W_PROGRAMS\SCHOOL\apache-jmeter-2.11\apache-jmeter-2.11\bin\templates\jdbc.jmx) - Apache JMeter (2.11 r1554548)
Archivo Editar Search Lanzar Opciones Ayuda
Test Plan
JDBC Connection Configuration
Thread Group
Petición JDBC
View Results Tree
Gráfico
Banco de Trabajo
Configuración de la Conexión JDBC
Nombre: JDBC Connection Configuration
Comentarios
Nombre Variable Enlazado al Pool
Nombre de Variable: jdbcConfig
Configuración del Pool de Conexiones
Número Máximo de Conexiones: 150
Timeout del Pool: 10000
Intervalo de Limpieza por Inactividad (ms): 60000
Auto Commit: True
Transaction Isolation: DEFAULT
Validación de Conexión por Pool
Keep-Alive: True
Edad máxima de las Conexiones (ms): 5000
Query de Validación: Select 1
Configuración de la Conexión a Base de Datos
URL de la Base de Datos: jdbc:postgresql://192.168.56.4:5432/PDVSA
Clase del Driver JDBC: org.postgresql.Driver
Nombre de Usuario: postgres
Password: ••••••••

Anexo 9