Universidad de las Ciencias Informáticas

Facultad 3

Título: Paquete de instalación de réplicas en servidores *PostgreSQL.*

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

Autor(es):
Nodalis Ricelda Nodal González
Dunia Ortiz Nodal

Tutor(es):
Ing. Jorge Yuniel Jorrín Perdomo
Ing. Alejandro Casanova Mutis

Asesora:
Ing. Yudisney Vázquez Ortiz

Junio, 2009

DECLARACIÓN DE AUTORÍA

Declaramos ser autoras de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas, y al Centro de Tecnología de Almacenamiento y Análisis de Datos los derechos patrimoniales de la misma, con carácter exclusivo.

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

Nodalis Ricelda Nodal González

_____

Firma del Autor

Dunia Ortiz Nodal

_____

Firma del Autor

Jorge Yuniel Jorrin Perdomo

_____

Firma del Tutor

Alejandro Casanova Mutis

_____

Firma del Tutor

"Si una persona es perseverante, aunque sea dura de entendimiento, se hará inteligente; y aunque sea débil se transformará en fuerte."

Leonardo Da Vinci

AGRADECIMIENTOS

Quisiéramos agradecer a las personas que hicieron posible la realización de este trabajo…

A nuestros tutores Jorge Yuniel Jorrin y Alejandro Casanova por su apoyo y colaboración.

A Nuestra cotutora Yudisney Vázquez por ayudarnos y apoyarnos en todo momento.

A Pedro Y. Piñeiro, por darnos la oportunidad de desarrollar este trabajo tan importante para nosotras.

A Yorlandy Cabrera que sin su esfuerzo y dedicación no hubiera sido posible la realización de dicho trabajo.

Y a todas las personas que dieron su colaboración para alcanzar este sueño.

Nodalis y Dunia

Quisiera agradecer a todas las personas que de una forma u otra han colaborado con la elaboración de este trabajo, y con mi formación como profesional, en especial a mis padres Nitza y Samuel, por haber vivido cada momento de mi vida como estudiante a mi lado, por su apoyo, firmeza, dedicación, por incitarme con su ejemplo a ser una persona mejor, por darme las fuerza y el valor de llegar al final de esta travesía y sobre todo por no perder la fe en mi.

A mis hermanos por ser mis ejemplos, mi orgullo, mis ídolos y por quererme tanto.

A mi tía Mercedes y mi prima Ely que tanto me han querido y ayudado.

A mis abuelos, en especial a mi abuela Aleida que a pesar de estar lejos siempre me ha apoyado y ha sentido un gran orgullo de mí.

A mi buen amigo, y más que amigo a mi hermano Iván, que tanto quiero y aprecio, por ser tan sincero y verdadero conmigo, por estar siempre que lo necesito y por tener la fuerza y el cariño suficiente para soportarme y quererme tanto.

A mi gran amiga Niudis por haber estado los cinco años a mi lado, y ser una persona muy querida para mí, gracias por aguantar mis perretas, mis lágrimas, por ser parte de esas familia que hoy tengo, y sobre todo por ayudarme en mis partos con la tesis, te lo agradezco mucho,

A mi compañera de tesis Dunia por haber tenido la fuerza, el valor, la paciencia de ser mi amiga y soportar mis perretas mis crisis de perfección, todo este tiempo con una sonrisa, y sobre todo por ser tan buena persona. Te quiero mucho.

A Julio César con quien he compartido momentos maravillosos. Gracias por estar junto a mí, y brindarme la mano siempre que le he necesitado, por sacarme una sonrisa cuando es casi imposible sonreír, por darme la fuerza y el valor para seguir... por ser una persona tan maravillosa... gracias por estar.

A todos mis amigos y familiares, mis vecinos, en fin a todos los que me aprecian...

Nodalis

Quiero expresarles mis agradecimientos a todas las personas que han estado a mi lado en los momentos buenos y malos a lo largo de mi vida en la UCI:

En primer lugar a mis padres Olivia y Reinaldo por ser las personas más importantes en mi vida, que me ayudaron a ser fuerte cuando más lo he necesitado y que se han sacrificado tanto para que yo pudiera llegar hasta este momento… LOS AMO CON TODA MI ALMA…

A mi hermano Demis que también ha sido mi guía y mi ejemplo a seguir, que me mima tanto por ser su hermanita pequeña…quiero que sepa que es el mejor hermano del mundo…

A mi cuñada Dainerys que de una forma u otra me dio siempre su apoyo…gracias…

A mi mejor amiga Yaily, gracias por estar ahí conmigo en cada momento sin importar cuán difícil sea, eres lo máximo y le doy gracias a la vida por haberme puesto en mi camino una personita tan especial como tú… eres la hermana que nunca tuve…

A mi amiga y compañera de tesis Nodalis, que hemos transitado juntas este camino hasta llegar a la meta, Eres especial, me has dado fuerzas para seguir en los momentos que para mí todo estaba perdido…y sobre todo mucha confianza…

A mi amiga Niudis, aunque nos conocimos en tercer año me ha bastado para darme cuenta lo grande que eres, gracias por estar ahí en el momento preciso, te quiero mucho…eso lo sabes…

A Damián gracias por ser como eres… único…

En fin agradecerles a todos mis compañeros de los grupos por los que he transitado…nunca me voy a olvidar de ninguno, a mis amistades que no menciono pero que saben que los llevo en el corazón, a toda mi familia que es muy grande y no puedo nombrarlos uno a uno, a mis vecinos siempre tan preocupados…a todos gracias…

Dunia

DEDICATORIA

Quiero dedicar este trabajo a mi familia, en especial a mis padres Nitza y Samuel, que son mi razón de vivir, a mis hermanos y cuñadas, a mi sobrinito Leinier, por ser en mi vida como un rayito de luz… Los quiero a todos.

Nodalis R. Nodal González

En especial quiero dedicar este trabajo a mi mami Olivia por todo lo que hemos pasado juntas, aún estando lejos, la quiero mucho…

A mi papi Reinaldo que es mi ídolo, mi Dios, lo amo mucho también…

A mi hermanito querido que tanto cariño y amor me ha dado y que hoy se gradúa junto a mí…lo adoro…

Dunia Ortiz Nodal

RESUMEN

El avance de la informática en los últimos tiempos ha propiciado el surgimiento y desarrollo de las técnicas de replicación de datos. En el presente trabajo se propone una herramienta que facilita el proceso de instalación y configuración de soluciones de réplica de datos entre servidores, utilizadas por el gestor de base de datos *PostgreSQL,* acciones que actualmente se realizan de forma engorrosa, además concentra varias de estas soluciones, debido a que no se cuenta con una herramienta que realice esta función, y a su vez ofrezca la posibilidad de realizar el proceso de instalación y configuración de manera conjunta.

Por tales motivos se realiza un estudio sobre las soluciones de replicación de datos existentes, que propicia realizar la elección de las herramientas de réplica que más se utilizan en la Universidad de las Ciencias Informáticas (UCI); para la elaboración de la solución propuesta; además de un estudio de los *shells* o intérpretes de comandos existentes para definir el más apropiado a utilizar en el desarrollo de la herramienta.

Se efectúa la caracterización de las soluciones escogidas para ser empaquetadas, donde se detallan sus características y se fundamenta la elección realizada. Siendo posible la creación del paquete de instalación de las soluciones escogidas, así como la creación de las guías de instalación de las respectivas herramientas, el manual de usuario para la utilización de dicho paquete, además se documenta la investigación realizada con el fin de demostrar que dada la situación problémica actual se requiere la creación de un paquete de instalación que responda a las necesidades encontradas.

Palabras claves: replicación de datos, *PostgreSQL*, paquete, shell, instalación, configuración.

ÍNDICE

INTRODUCCIÓN

La influencia de la Informática en el mundo contemporáneo es muy amplia y multidisciplinaria. Existen muchas aplicaciones informáticas con acceso a través de la red. Es importante en algunos casos que estos servicios sean prestados de manera continua.

Para garantizar la disponibilidad de dichos servicios, se deben utilizar técnicas de replicación de datos entre servidores, garantizando en caso de que ocurriera un fallo en algunos de los componentes de las aplicaciones, que otras réplicas puedan servir las peticiones recibidas y consumarlas de manera totalmente trasparente al cliente.

Estas técnicas son manejadas por distintos sistemas gestores de base de datos, dígase *Oracle*, *MySQL*, *PostgreSQL*, siendo este último un potente gestor considerado como una de las alternativas de sistema de código abierto.

Muchos son los países que han realizado estudios sobre el tema de la replicación de datos en este gestor, como es el caso de Brasil, Argentina, Australia, Venezuela, entre otros.

Cuba no se ha quedado detrás; en el estudio y utilización de estas técnicas; a pesar de tener un desarrollo insipiente en la industria de software. Para llevar a cabo esta tarea se cuenta con varios centros de desarrollo, muchas personas capacitadas y sobre todo muchas estrategias para mejorar esa situación. Como parte de estas estrategias se encuentra la Universidad de las Ciencias Informáticas (UCI) primera universidad surgida sobre la base del nuevo concepto de universidad productiva, la cual incluye en sus objetivos la creación de productos, servicios y soluciones informáticas, para brindarle al país y al resto del mundo.

En aras de alcanzar este objetivo se han creado áreas dedicadas a la producción de software, en muchas de éstas, se desarrollan aplicaciones de gestión, los cuales necesitan mantener un constante flujo de intercambio de información siguiendo un modelo cliente servidor.

Los que se dedican a realizar dicho proceso de intercambio, se enfrentan a varios problemas fundamentales, como son: la poca experiencia a la hora de seleccionar la solución de réplica a utilizar según las necesidades de replicación que se tengan, la ausencia de una herramienta que agrupe varias soluciones de replicación de datos, la compleja instalación y configuración de las herramientas de

replicación, debido a que ninguna realiza estos procesos de forma conjunta. Sin dejar de resaltar que existe poco capital humano especializado en la replicación de datos en servidores *PostgreSQL*.

La presente investigación surge como necesidad de dar solución al **problema** que se deriva de la situación existente: la complejidad en la instalación y configuración de las soluciones de réplica dificulta la replicación de datos en servidores *PostgreSQL*.

Tomando como **objeto de estudio** el proceso de replicación de bases de datos y el **campo de acción** se enmarca en el proceso de replicación de bases datos en servidores *PostgreSQL*.

Para dar respuesta al problema planteado se define como **objetivo general**: Crear un paquete de instalación de soluciones existentes de réplicas en servidores *PostgreSQL,* que automatice la instalación y configuración de dichas soluciones.

Para llevar a cabo dicho objetivo se trazaron las **tareas de investigación**:

- Búsqueda detallada de las soluciones existentes de réplicas.
- Definición de los indicadores a tener en cuenta para desarrollar el proceso de replicación.
- Creación de un paquete de instalación que concentre las soluciones existentes de réplicas más utilizadas y las herramientas necesarias para realizar el proceso de replicación.
- Creación de un manual de usuario para la utilización del paquete.

Se parte de la **hipótesis** de que: si se crea un paquete de instalación de soluciones existentes de réplicas en servidores *PostgreSQL*, se automatizará la instalación y configuración de las mismas.

Para dar cumplimiento a las tareas propuestas anteriormente se van a utilizar los métodos científicos de la investigación: ***teórico y empírico*****.**

Entre los métodos ***teóricos*** utilizados para la investigación se emplearon:

- El método ***analítico-sintético****,* para la realización del estudio teórico de la investigación y en la investigación previa sobre el funcionamiento del proceso de replicación de datos que se lleva a cabo en la Universidad de Las Ciencias Informáticas, permitiendo extraer los elementos más importantes.

- El método **inductivo-deductivo**, para a través de un razonamiento llegar a un grupo de conocimientos particulares y generales.

De los métodos **empíricos** se utilizó:

- La **entrevista** para obtener información valiosa del proceso de replicación que se lleva a cabo en la Universidad de Las Ciencias Informáticas.

Con este Trabajo de Diploma se pretende obtener los siguientes **resultados:**

- Caracterización de la soluciones de réplica a empaquetar.
- Guía de instalación de la herramientas seleccionadas.
- Paquete de instalación de soluciones de réplicas.
- Manual de usuario de dicho paquete.

El presente trabajo de diploma ha sido organizado de la siguiente manera:

Capítulo I. Fundamentación teórica: En este capítulo se realiza un estudio del estado del arte del proceso de replicación de datos, las soluciones de réplicas existentes sustentadas en servidores *PostgreSQL*, los *shells* o intérpretes de comandos para definir el más apropiado para el desarrollo de la herramienta.

Capítulo II. Caracterización de las soluciones de réplica a empaquetar: En este capítulo se caracterizan las soluciones de réplicas seleccionadas desarrollándose un estudio detallado del funcionamiento de cada una, concluyendo con un resumen que define los indicadores más relevantes de las mismas.

Capítulo III. Presentación del paquete de Instalación: En este capítulo se realiza la presentación del paquete de instalación, donde se detalla cada uno de los pasos seguidos para su conformación, además del método utilizado para la validación de dicho producto.

Capítulo #1. Fundamentación teórica

Introducción

El proceso de replicación es un concepto muy importante en términos de bases de datos, este trae asociado varias definiciones, que especifican componentes que lo integran, entornos en los que se puede apreciar este proceso, los tipos réplica que existen, los modelos que se definen, entre otros. En este capítulo se ofrece una visión general de estos temas, facilitando de este modo un mejor entendimiento sobre el tema en cuestión, y a su vez se brinda una visión más amplia sobre las soluciones de réplicas existentes a nivel mundial y nacional; para servidores *PostgreSQL;* así como su importancia en el mundo de la informática, para lograr un mejor entendimiento de la solución que se confeccionará, además de las herramientas, técnicas y procesos empleados para su construcción.

1.1 Proceso de replicación de datos

El problema de la replicación de datos entre servidores ha sido un tema ampliamente estudiado por compañías desarrolladoras de gestores de base de datos a nivel mundial, debido a los conflictos de actualización, unicidad, supresión[1], entre otros, que trae este proceso, ya que en la actualidad tiene gran importancia el empleo de estas soluciones debido a que ofrecen muchas ventajas para cualquier proceso de intercambio de información, este proceso puede ser usado por varias razones, las cuales pueden ser categorizadas de las siguientes formas:

- Distribución de datos a otras ubicaciones.
- Consolidación de datos desde otras ubicaciones.
- Intercambio bidireccional de datos con otras ubicaciones.
- Algunas variantes o combinaciones de los anteriores.

Por los motivos antes mencionados se ha realizado un estudio de varias definiciones del proceso de replicación de datos como son:

“La replicación de datos: Consiste en asegurar la disponibilidad de los datos sincronizando copias de una fuente de datos en diferentes servidores” (Kioskea).

[1] *Supresión: Eliminación, desaparición.*

"La replicación es el proceso de crear y mantener versiones duplicadas de objetos de bases de datos, por ejemplo tablas, en un sistema de base de datos distribuida". (Steffen, Dr.Ing. Hermann , 2003)

"Técnica que permite copiar y distribuir idénticamente las tablas de una base de datos en múltiples bases de datos ubicadas en diferentes nodos de la red. La replicación asegura que los datos correctos estén siempre disponibles en el momento y en el lugar necesario". (Freedman, 1993)

Después de estudiar estas definiciones se decidió tomar para el desarrollo del presente trabajo la más concreta, de modo que la investigación está enfocada en una sola definición, la cual sería:
"La replicación de datos es el proceso de compartir objetos y datos de una base de datos a múltiples bases de datos, en diferentes localizaciones". (Piñeiro, et al., 2006)

Este proceso de intercambio de datos consiste en capturar la modificación que se desee hacer en un base de datos determinada, y trasmitir dicha modificación a todas las instancias de esta base de datos que se tengan, para luego aplicarla en cada una de ellas, tal y como se muestra a continuación.

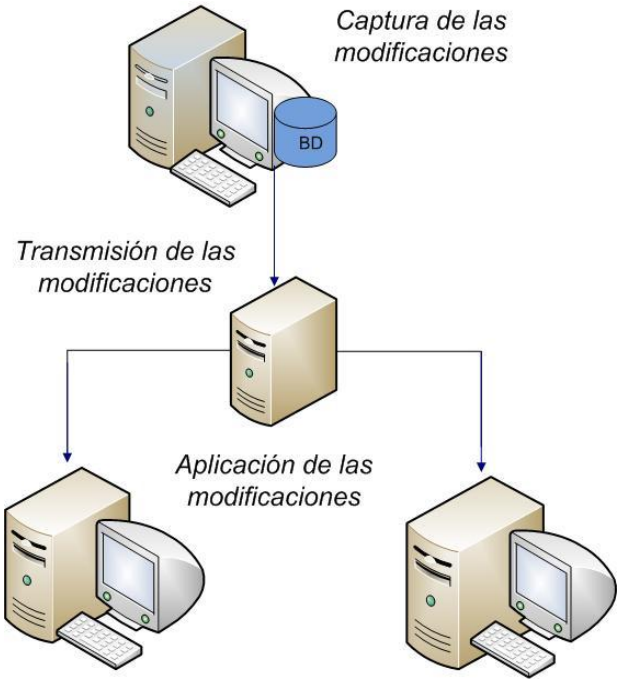


Fig. #1.1 Procesos de un mecanismo de réplica.

Este proceso de intercambio tiene una serie de características que lo distingue de cualquier otro proceso, dentro de las que se destacan:

Coordinación: Todas las partes que componen la base de datos, llegan a un consenso para la solicitud de los servicios a los objetos, para que se realice tal y como fue solicitado al final de la transacción, para lo que debe utilizar algún tipo de método de ordenamiento.

Alta disponibilidad: Es el tiempo estimado en el que un servicio puede ser accedido, siendo el 100% el mejor de los caso. Para prevenir los fallos que pueda presentar el servidor, debe existir un servidor adicional que tenga esta técnica de replicación que lo pueda sustituir en caso necesario.

Efectividad: Viene dada por la forma de distribuir y almacenar los datos. Concluyendo que a mayor efectividad, mayor será la disponibilidad de los datos para ejecutar procesos paralelos.

Tolerancia a fallos: Da la garantía de un comportamiento adecuado, donde pueden existir un número determinado de fallos y tipos de fallos.

1.1.1 Componentes de replicación

La replicación utiliza un lenguaje para representar los componentes de una topología[2] de replicación, que incluyen el publicador, el distribuidor, los suscriptores, las publicaciones, los artículos y las suscripciones, a continuación se explica brevemente en qué consiste cada uno de ellos (Corporation, 2009).

Publicador: Es una instancia de base de datos que permite que los datos estén disponibles para otras ubicaciones a través de la replicación. El publicador puede tener una o más publicaciones, cada una de las cuales representa un conjunto de objetos y datos relacionados lógicamente para replicar.

Distribuidor: Es una instancia de base de datos que funciona como almacén para datos específicos de replicación asociados con uno o más publicadores. Cada publicador está asociado con una sola base de datos (conocida como la base de datos de distribución) en el distribuidor. La base de datos de distribución almacena los datos de estado de la replicación, metadatos acerca de la publicación y, en algunos casos, funciona como cola para los datos que se transfieren del publicador a los suscriptores. En muchos casos,

[2] *Topología: Es la ruta por la que la replicación de datos viaja a través de una red.*

una sola instancia de servidor de bases de datos funciona como publicador y como distribuidor. Esto se conoce como un distribuidor local. Cuando el publicador y el distribuidor se configuran en instancias distintas del servidor de bases de datos, el distribuidor se denomina un distribuidor remoto.

Suscriptores: Es una instancia de base de datos que se encarga de recibir los datos replicados, de los publicadores y publicaciones. Además posibilita devolver los datos con modificaciones al publicador, o volver a publicarlos en otro suscriptor. (Microsoft Corporation, 2009)

Artículo: No es más que un objeto de base de datos que forma parte de una publicación, como es el caso de tablas, vistas, procedimientos almacenados, entre otros objetos. (Microsoft Corporation., 2009)

Publicación: Conjunto de unos o varios artículos de una base de datos, que permiten especificar los objetos o datos que se van a replicar.

Suscripción: Consiste en la solitud de una copia de publicación que obtiene el suscriptor, además precisa cual será la publicación recibida, el destino y el momento adecuado. Existen dos tipos de suscripciones: de inserción y de extracción.

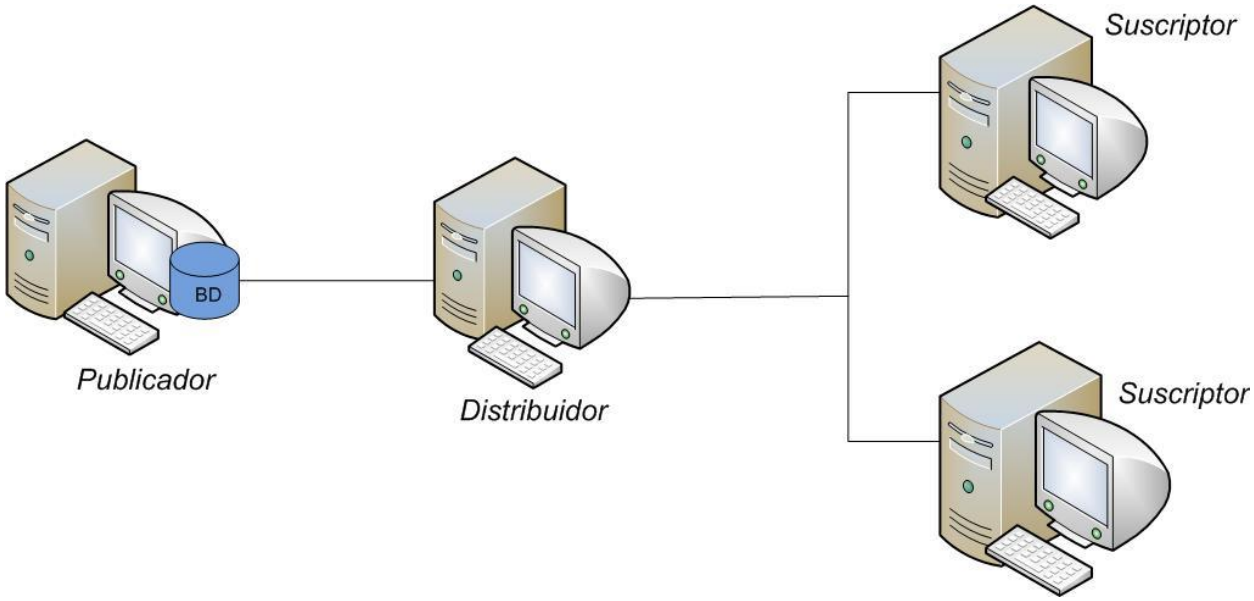


Fig. # 1.2 Componentes de replicación.

1.1.2 Entornos de replicación

La replicación de datos se puede apreciar en diversos entornos, dependiendo del gestor de base de datos y el sistema operativo que se utilice en cada uno de los servidores fuentes y destino involucrados, los cuales pueden ser entorno homogéneo y entorno heterogéneo. (Piñeiro, et al., 2006)

1.1.2.1 Entorno Homogéneo

En este entorno la réplica de datos se realiza entre servidores de datos con el mismo gestor y sobre el mismo sistema operativo, o entre servidores de datos con el mismo gestor pero donde los sistemas operativos son diferentes.

Ejemplo: *PostgreSQL* instalados en plataformas *Windows* y *Linux*.

1.1.2.2 Entorno Heterogéneo

La replicación en este entorno se ejecuta entre servidores de datos con diferentes gestores de datos y el mismo sistema operativo, o el caso de tener una réplica de datos entre servidores de datos con diferentes gestores de datos y diferentes sistemas operativos.

Ejemplo 1: *PostgreSQL* vs *Oracle* o *SQL Server* vs *PostgreSQL* ambos corriendo en *Windows*.

Ejemplo 2: *PostgreSQL* vs *Oracle* o *SQL Server* vs *PostgreSQL* uno en *Windows* y otro en *Linux*.

1.1.3 Tipos de replicación

La replicación de datos es un proceso complejo que tiene varias formas de realizarse, como es el caso de la replicación de instantáneas, la replicación transaccional y la replicación de mezcla.

1.1.3.1 Replicación de instantáneas

En este tipo de replicación los datos son copiados tal y como aparecen en un determinado momento. Trayendo consigo que no requiera de un control de cambios continúo. Las publicaciones de este tipo de replicación suelen realizarse con una menor frecuencia que las demás, debido a que consume más tiempo en propagar las modificaciones de los datos a los suscriptores. Esta replicación es sugerente cuando la mayoría de los datos reciben pocas modificaciones.

1.1.3.2 Replicación transaccional

Este tipo de replicación consiste en distribuir una instantánea inicial a los suscriptores, para cuando se realicen las modificaciones en el publicador estas se propaguen a los suscriptores. Al terminar dicha propagación tanto los suscriptores como los publicadores tendrán los mismos valores. Este tipo de replicación se utiliza cuando se quiere que las modificaciones de datos se distribuyan a los suscriptores. Regularmente pocos segundos después de producirse la modificación, es necesario que las transacciones sean lo más rápidas posible, para que se apliquen todas o ninguna al suscriptor, los cuales en su mayoría se conectan al publicador. La aplicación de estos cambios no puede permitir un período de latencia muy largo para los suscriptores que reciban cambios. (TechNet, 2009).

1.1.3.3 Replicación de mezcla

Con esta replicación es posible que varios sitios funciones en línea o desconectados de manera independiente, par que más adelante se pueda mezclar las modificaciones de datos ejecutadas y obtener resultados únicos y uniformes. Inicialmente se aplica una instantánea a los suscriptores. Los datos son sincronizados entre los servidores a una hora determinada o puede ser a petición. Las actualizaciones son ejecutadas de forma independiente sin protocolos de confirmación en más de un servidor, de esta manera tanto el publicador como el suscriptor pueden actualizar los mismos datos, esto trae consigo que se puedan producir conflictos al mezclar las modificaciones de datos. Cuando ocurre un conflicto el agente de mezcla pide una resolución para determinar los datos que se aceptarán, para que luego se generalicen a otros sitios. Es útil cuando más de un suscriptor necesita actualizar datos y dar a conocer los cambios al publicador y a otros suscriptores, además cuando estos últimos necesitan recibir datos, realizar cambios, y sincronizar dichos cambios en el publicador y demás suscriptores.

1.1.4 Modelos de Replicación

Existen dos modelos de distribución de datos esencialmente aplicados a cada uno de los entornos antes vistos, como es el caso del modelo asíncrono y el modelo síncrono.

1.1.4.1 Modelo Asíncrono

Esta tecnología de replicación, es la más reciente para la tolerancia a fallos en servidores y almacenamiento en red. Posibilita la captura de los cambios locales efectuados, almacenándolos en una cola, para propagarlos en intervalos regulares en otros sitios. Con la utilización de este modelo existe un

período de tiempo antes de que los sitios alcancen la convergencia de datos[3]. El mismo opera en el nivel de las aplicaciones o en el del hardware, además realiza un volcado de datos el cual consiste en copiar las salvas para un dispositivo de almacenamiento para luego distribuirlas a los demás servidores. Este modelo tiene como agravante que los datos consultados estén desactualizados, además de que este proceso es realizado de forma manual. (IDG COMMUNICATIONS, 2009)

1.1.4.2 Modelo Síncrono

También conocida como la réplica en tiempo real, es una tecnología surgida a mediados de los años 80, aplica cualquier cambio o ejecuta cualquier procedimiento reproducido, en todos los sitios que participan en el ambiente de réplica como parte de una sola transacción. Si el procedimiento falla en cualquier sitio, entonces la transacción entera se anula. La réplica sincrónica asegura la consistencia de datos en todos los sitios en tiempo real, permitiendo además la sincronización de datos distribuidos.

1.1.5 Tipos de réplicas

Las réplicas de datos se dividen en dos amplias categorías: replicar datos en un servidor para un entorno de servidor (Multi-Maestro) y replicar datos entre un servidor y los clientes (Maestro-Esclavo).

1.1.5.1 Maestro-Esclavo (Master-Slave)

Es el caso en el que los datos se pueden modificar en múltiples ubicaciones, entonces la replicación debe procesar los cambios realizados en cada uno de los sitios de forma coordinada. Uno de los servidores, es visto como el servidor maestro, quien se encarga de distribuir los cambios a todos los sitios. Los cambios realizados en los destinos fluyen hacia los otros sitios a través del servidor maestro. Este modo de replicación puede ser usado para aplicaciones móviles donde los destinos pueden ser tanto una computadora de una oficina, como una laptop sin una ubicación específica. A menudo hay varios destinos que se conectan ocasionalmente al sistema fuente, esta conexión puede ser a través de líneas telefónicas, por lo qué la eficiencia es importante.

Los datos normalmente se replican entre servidores y clientes para admitir las siguientes aplicaciones:

- Intercambiar datos con usuarios en otras localizaciones.

[3] *Convergencia de datos: Concordancia, correlación entre los datos*

- Aplicaciones de punto de venta para el consumidor.
- Integrar datos de varios sitios.

1.1.5.2 Multi-Maestro (Multi-Master)

También llamada "*peer-to-peer*" o la réplica de camino de n. Esta réplica no tiene designada un servidor maestro. Cada ubicación copia los cambios desde todos los otros sitios directamente. Cada sitio en un ambiente de réplica de multi-maestro es un sitio de maestro, y cada sitio se comunica con otros sitios maestros. Puede ser usada para mantener sitios recuperables ante posibles desastres o caídas, así como para proveer sistemas con alta disponibilidad y para balancear la carga de consultas a través de las distintas ubicaciones.

Generalmente, los datos se replican entre servidores para proporcionar compatibilidad con las siguientes aplicaciones y requisitos:

- Mejorar la escalabilidad y la disponibilidad.
- Almacenamiento de datos e informes.
- Integrar datos de varios sitios.
- Integrar datos heterogéneos.

1.1.6 Ventajas del uso de replicación de datos

La réplica de datos es deseable por diversas razones: en primer lugar, puede producir un mejor desempeño (las aplicaciones pueden operar sobre copias locales en vez de tener que comunicarse con sitios remotos, lo que reduce la transferencia de datos); en segundo lugar, puede significar una mejor disponibilidad (un objeto estará disponible para su procesamiento en tanto esté disponible por lo menos una copia, al menos para propósitos de recuperación), además los datos replicados en dos servidores independientes, conectados a la red por vínculos independientes permiten seleccionar al otro servidor ante la falla de uno cualquiera de ellos o de su enlace.

También puede aumentar el paralelismo, ya que se podrían procesar las peticiones en varios nodos en paralelo, y por último se obtienen mejoras en la performance ya que múltiples clientes accediendo a un solo servidor lo transforman en un cuello de botella: por encima de un número máximo de requerimientos

por segundo, el tiempo medio de respuesta cae significativamente, mientras que con la replicación, múltiples servidores atendiendo a subconjuntos de clientes permiten mejorar el tiempo medio de respuesta.

En general cuando se habla del término replicación se refiere a protección y disponibilidad de los datos al permitir hacer una copia exacta de los datos de un sistema a otro, normalmente en tiempo real con la ventaja de que si el sistema primario falla, se puede reanudar el servicio de forma rápida desde el secundario. Con la replicación las versiones de los ficheros son idénticas.

1.1.7 Problemas frecuentes en la replicación de datos

Siempre que se enfrenta un proceso de replicación de datos se está expenso a problemas o conflictos que se pueden ocasionar en dicho proceso, como es el caso de:

Conflicto de actualización: Se produce cuando dos transacciones realizadas desde sitios diferentes actualizan el mismo registro, o sea cuando se produce la replicación de un *update* sobre un registro con otro *update* sobre el mismo registro.

Conflicto de unicidad: Sucede cuando una restricción de integridad es incumplida por la replicación de un registro, utilizando la clave primaria, es decir, cuando dos transacciones originadas de dos sitios totalmente diferentes, cada una inserta un registro en la tabla replicada que le corresponde utilizando el mismo valor de clave primaria.

Conflicto de supresión: Cuando se tienen dos transacciones que son originadas por sitios diferentes y una de ellas intenta eliminar un registro y la otra transacción intenta actualizar el mismo registro.

Conflicto de orden: Sucede cuando el orden de la propagación a los sitios se ve alterado. Es un conflicto que suele realizarse en ambientes de replicación con más de tres sitios maestros. Si por alguna razón la propagación a un sitio determinado se ve afectada o bloqueada, entonces la replicación de las modificaciones en los datos puede continuar su propagación a través de los sitios maestros, y al finalizar la propagación estas modificaciones debieron ser propagadas al determinado servidor en un orden diferente a como ocurrieron en los sitios maestros. (Steffen, Dr.Ing. Hermann , 2003)

1.2 Soluciones de replicación en PostgreSQL existentes a nivel mundial y nacional

La utilización de las herramientas de replicación de bases de datos, son indispensables para mantener un respaldo de los datos en otras localizaciones, por lo que son muy utilizadas en las aplicaciones de gestión. Actualmente existen en el mundo varias herramientas que permiten llevar a cabo el proceso de replicación de datos entre sistemas gestores de bases de datos. En este caso se refiere al gestor *PostgreSQL*. En esta sección se describirá brevemente algunas de las herramientas que utiliza *PostgreSQL* para el proceso de replicación de datos, y de esta manera poder fundamentar la elección que se realizará de las herramientas que formarán parte de la solución de este trabajo.

1.2.1 Cybercluster

Es un sistema de replicación multi-maestro, con técnicas de repetición síncrona. Es una herramienta basada en la versión *PostgreSQL* 8.3 que sincroniza la base de datos para asegurarse que todos los nodos dentro del clúster contienen los mismos datos. Posibilitando que todos los nodos puedan trabajar eficazmente al mismo tiempo. Entre sus principales funcionalidades se pueden encontrar, el equilibrio de carga, asegurando que todos los nodos puedan usarse durante el funcionamiento normal, además de contar con la posibilidad de manejo de errores, haciendo posible que no se interrumpa el proceso cuando algún nodo tenga alguna rotura, y por último la recuperación se realiza de forma automática.

Cybercluster como herramienta de replicación tiene ventajas que ofrecer, como es el caso de poseer el código abierto como solución heredera de software libre, puede transmitirse y usarse libremente según las condiciones de la licencia de BSD[4]. Otra ventaja significante es que *Cybercluster* contiene una disponibilidad de soporte las 24 horas del día, las aplicaciones existentes no necesitan ser cambiadas ya que esta herramienta soporta cualquier tipo de ellas. Es perfecta para aplicaciones de alto rendimiento. Además brinda la posibilidad de mantener consistencia y acceso a los datos en todo momento. Es compatible con sistemas operativos como: Windows (Desde Windows 2000), Linux (Debian, RedHat Enterprise Server, Suse Enterprise Server, etc.) FreeBSD, Mac OS X, Solaris.

[4] *Licencia BSD: licencia pertenece al grupo de licencias de software libre.*

1.2.2 Pgcluster

Es un sistema de replicación sincrónico, "multi-maestro" para *PostgreSQL*. Está constituido por tres tipos de servidores, en primer lugar se encuentra un servidor para balance de carga que recibe la consulta del cliente, y envía una consulta a la base de datos del clúster con el más bajo índice de carga. En segundo lugar está el clúster BD que no es más que el servidor que recibe la consulta a la base de datos de un usuario, y por último está el servidor de réplica encargado de enviar la consulta de la base de datos recibida a todos los clústeres y además guarda la orden recibida. (Company, 2009)

Pgcluster tiene dos funciones principales: la de compartir carga, dicha carga es distribuida según las demandas recibidas, además posee una alta disponibilidad, siendo evidente en el momento que ocurre un fallo en el clúster y continúa el servicio con los nodos restantes. Cuando es reparado puede restaurarse dinámicamente al sistema, sin detener el servicio.

Pgcluster está soportado bajo la licencia DEB[5], es considerada como una de las herramientas de replicación de datos más potentes ya que actúa como un parche adicional que permite manejar clúster de bases de datos, balanceo de carga y replicación de servidores con la combinación de cada uno. Tiene la capacidad de sincronizar el árbol de directorios. *Pgcluster* usa rsync[6] para la sincronización de datos con bases de datos maestro, en el momento de la adición del clúster BD o restauración.

1.2.3 Bucardo

Bucardo es un sistema de replicación asíncrono de *PostgreSQL*, permite tanto replicación multi-maestro como maestro-esclavo de fácil configuración. Este sistema no requiere modificación de la instalación de *PostgreSQL*, y se ejecuta como un *daemons*[7] en Perl [8]que se conecta a la base de datos de control y todas las bases de datos que se repita, las actualizaciones pueden suceder más rápido ya que los cambios de datos no son rastreados. Para instalar Bucardo, se necesita:

- Hacer e instalar los módulos Perl.
- Crear la base de datos.
- Crear el esquema de importación.

[5] *DEB: Extensión del formato de paquetes de software de Debian (libre).*
[6] *Rsync: Comando para copiar archivos entre servidores.*
[7] *Daemon: Es un proceso informático que se ejecuta en segundo plano.*
[8] *Perl: Es un lenguaje imperativo.*

Una de las ventajas de la herramienta de replicación de datos Bucardo consiste en permitir que una base de datos *PostgreSQL* sea replicada en otra base de datos *PostgreSQL*, agrupando tablas a la manera de transacción segura. Las versiones más recientes de esta herramienta emplean nuevas características, incluyendo un modelo robusto de daemon, configuración y logueo más flexibles, rutinas personalizadas de manejo de excepciones y conflictos, tiempos de replicación mucho más rápidos, y un alto nivel de auto-mantenimiento.

Muchos aseguran que replicar con Bucardo no es del todo satisfactorio, ya que esta herramienta presenta un número elevado de limitaciones entre las que se encuentran que requiere Postgres 8.1 o superior, con Pl/Perl y Pl/pgsql. Solo replica tablas, no la base de datos entera, no replica DDL[9] y requiere una llave primaria sobre cada tabla para ser replicada. (Bucardo)

1.2.4 Pgpool

Es un middleware[10] que funciona entre *PostgreSQL* y servidores de una base de datos *PostgreSQL* cliente, es una herramienta de fácil instalación y configuración, su código es estable, además de ser flexible, independientemente de que necesita de otras herramientas. Ofrece características como:

Agrupación de conexiones: Ya que guarda conexiones con el servidor *PostgreSQL*, y la reutiliza siempre que sea una nueva relación con las mismas propiedades (es decir, nombre de usuario, bases de datos, versión del protocolo). Siendo posible que se reduzca la conexión y el rendimiento en general es mucho mayor

Replicación: Utiliza la función de reproducción, esta función permite la creación de copias de seguridad en tiempo real de dos o más discos físicos, para evitar que se interrumpa el servicio en caso de un fallo de disco en un servidor.

Balance de carga: Es un servicio que actúa en el momento que hay varios usuarios haciendo muchas preguntas al mismo servidor, este reduce la carga compartiéndola entre los servidores, mejorando el rendimiento del sistema. Si se ejecuta una consulta SELECT en cualquier servidor devolverá el mismo resultado.

[9] *DDL: Comandos de lenguaje de definición de datos.*
[10] *Middleware: Software de conectividad para el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas.*

Limita las conexiones: Esta permite limitar el número de conexiones simultáneas con *PostgreSQL*, rechazándolas cuando excedan el número definido, aunque aumenta el consumo de recursos y afecta el rendimiento del sistema, este número tiene un límite máximo, poniendo las que lo superen en una cola en lugar de devolver un error.

Consultas paralelas: El uso de la función de búsqueda en paralelo hacen posible dividir los datos entre múltiples servidores de forma tal, que una consulta se puede ejecutar en todos los servidores simultáneamente para reducir el tiempo de ejecución, siendo la mejor obra de consulta en la búsqueda de datos a gran escala.

Pgpool tiene como ventaja que es libre, por lo que no se debe pagar por él, se puede configurar en un solo servidor, las conexiones son persistentes desde el punto de vista del servidor, por lo que no se tiene que preocupar por la persistencia desde el punto de vista del cliente, y conservar las ventajas de rendimiento.

Tiene como desventaja el hecho de ser multi-proceso haciéndolo gran consumidor de memoria RAM, una vez que se alcanza el número máximo de procesos, hace una cola de las conexiones entrantes en el sistema operativo, además de permitir solamente dos servidores corriendo (PostgreSQL, 2003-2009).

1.2.5 Slony

Es un sistema de replicación maestro-esclavo, soporta la réplica en cascada y permite a un esclavo ser a su vez maestro para otro servidor.

El panorama para el desarrollo de *Slony* es que se trata de un sistema de replicación maestro-esclavo que incluye todas las características y capacidades necesarias para replicar bases de datos grandes a un número razonablemente limitado de sistemas de esclavo. En este contexto razonable se refiere a un orden de una docena de servidores. Si el número de servidores crece más allá de ese, el coste de comunicaciones aumentará prohibitivamente, y las ventajas incrementales de tener servidores múltiples fallarán en ese punto. Además permite indicar qué cambios replicar de un servidor a otro. *Slony* implementa la réplica asincrónica, usando disparadores o *triggers* [11] para determinar las actualizaciones de las tablas, donde un solo origen (maestro) se puede replegar a los suscriptores múltiples (esclavos) incluyendo suscriptores conectados en cascada.

[11] *Triggers:* Es un procedimiento que se ejecuta cuando se cumple una condición establecida

La principal ventaja de este sistema de replicación es que el servidor primario manda las actualizaciones en tiempo real a la base de datos esclava y en caso de que falle el servidor primario, se continúa trabajando con la base de datos esclava.

Uno de los beneficios que proporciona la replicación de datos con *Slony*, es que el servidor primario manda las actualizaciones en tiempo real a la base de datos esclava, y en caso de que falle el servidor primario, continúa trabajando con la base de datos esclava. Esta solución es una excelente elección cuando se requiere realizar una replicación asíncrona de bases de datos de *PostgreSQL*. Es considerado como un sistema de replicación que soporta conexiones de tipo maestro y múltiples esclavos, ya sean con conexiones en cascada o *fileover*[12]. Hay que destacar su posibilidad de realizar una réplica de espejos, exactamente igual al origen de datos. *Slony* brinda gran flexibilidad, rendimiento y estabilidad.

Una de las desventajas que presenta la herramienta de replicación de datos *Slony* es la de ser considerablemente más lento y utiliza más recursos que otras soluciones, además está su replicación incorporada, o sea que utiliza SQL y *triggers* en lugar de un registro binario de envío para replicar los datos a través de los servidores. Por lo cual *Slony* puede ser menos adecuado para grandes instalaciones de clústeres. Además de que no es una solución nativa, es decir que no viene directamente embebido dentro de las opciones de PostgreSQL (Slony Develoment Group, 2007).

1.2.6 PyReplica

PyReplica es una herramienta de replicación asincrónica maestro-esclavo y multi-maestro, es decir replica en ambos sentidos (cada base de datos es un maestro y un esclavo al mismo tiempo), resulta simple para *PostgreSQL* basado en *Python*[13] para el que usa un disparador maestro, señales, secuencias, y un *script* cliente, tiene la influencia de *Slony* & *Londiste*, pero mucho más simple y fácil. Recientemente se ha agregado soporte para servicio básico en *Windows*, múltiples esclavos y replicación condicional mediante filtros escritos en *Python* (actualmente en fase de pruebas).

PyReplica al igual que las demás herramientas de replicación de datos presenta sus propias ventajas convirtiéndola en una solución fácil de administrar y eficiente. En primer lugar está su fácil instalación ya que simplemente ejecutar un *script SQL* en el servidor, y copiar un *script* daemon en el cliente, para lo que

[12] *Fileover: Bases de datos en el mismo nivel.*
[13] *Python: Es un lenguaje de programación interpretado.*

no se requiere compilación. Además tiene una fácil administración, en general no necesita administración para el uso normal, no es necesario aprender un nuevo conjunto de comandos o marco de trabajo. Hay que destacar su eficiencia pues tiene bajo impacto de uso de memoria y red sin *polling.*[14]Además es una herramienta de réplica multiplataforma, corre bajo *Linux* y W*indows*. Probado en *Debian* (*Triggers y Daemon*) y Windows XP (solo *Triggers*).

PyReplica presenta varias limitaciones como es el caso del control de fallo pues no es automático .La resolución de conflictos es otra de sus limitaciones ya que advierte al detectar conflictos de actualización/eliminación y falla en conflictos de inserción o errores de integridad de datos, pero no los corrige. Está además la replicación de cambios de esquema donde los comandos CREATE, ALTER, entre otros deben ejecutarse manualmente en todos los servidores y por último no hay soporte para objetos grandes (PostgreSQL, 2003-2009).

1.2.7 Réplica bidireccional basada en control de cambios

Esta réplica se basa en el control de cambios en las bases de datos relacionales y la sincronización de las mismas. Esta solución permite la implementación de sistemas de base de datos distribuidos, aislando la complejidad del sistema del mecanismo de replicación de los datos. Dos de los objetivos principales de esta solución son simplificar y optimizar el proceso de sincronización de la información entre servidores de base de datos (Landrian García).

Este mecanismo de réplica fue creado para gestores de base de datos relacionales, y está diseñado con el objetivo de ser un mecanismo simple y al mismo tiempo óptimo para la sincronización de datos.

[14] *polling.: Una operación de consulta constante, generalmente hacia un dispositivo de hardware.*

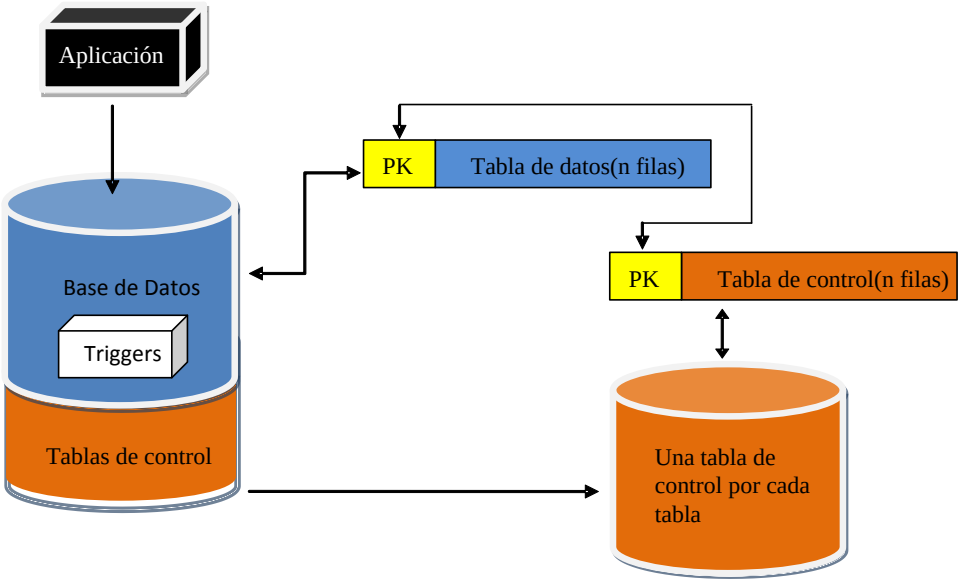


Fig. # 1.3 Esquema de funcionamiento del control de modificaciones.

Utiliza *triggers* para el mantenimiento de las tablas de control, sobre las tablas de la base de datos, estas tablas y *triggers* son generados automáticamente por una herramienta que utiliza el esquema de datos de las tablas que participan en la réplica.

El proceso de captura de modificaciones se divide en dos subprocesos en este mecanismo, y se realizan de forma independiente:

- El subproceso de control de modificaciones.
- El subproceso de descubrimiento de estas modificaciones.

Esta solución de réplica, disminuye la complejidad de un sistema distribuido, abstrayendo a los desarrolladores de la arquitectura de base de datos utilizada, además que posibilita la configuración de distintas arquitecturas de base de datos distribuidas sin tener que realizar cambio en el modelo de negocio. El producto *DBSynchronize* implementa este mecanismo de sincronización y permite diferentes configuraciones según el objetivo que se quiera cumplir. (Landrian García)

Luego de un estudio realizado a las diferentes soluciones de replicación de datos existentes, y una investigación en las distintas áreas donde se trabaja con la replicación de datos en la UCI, dígase el Centro de Tecnologías de Almacenamiento y Análisis de Datos, el proyecto ERP Cubano, el proyecto Identidad, entre otros, se llegó a la conclusión que las soluciones que se van a empaquetar serán *Slony*, *Pgcluster* y *Cybercluster* debido a que son herramienta muy potentes en el mundo de la replicación de datos específicamente para servidores *PostgreSQL,* además estas herramientas de replicación están siendo muy utilizadas no solo en la universidad con los proyectos antes mencionados sino en el país y el resto del mundo.

1.3 Herramientas y Tecnologías actuales

1.3.1 Tendencia al software libre

En el plano de las Tecnologías de la Informática y las Comunicaciones (TIC) el modelo de desarrollo seguido; basado en software propietario; ha garantizado durante muchos años la explotación del conocimiento y el aprovechamiento de esta ciencia con fines comerciales. Esto ha provocado que la evolución de la informática en el mercado de software esté condenada a una dependencia permanente al pago por los servicios y por el uso de herramientas.

Nuestro país con características de subdesarrollo ha ido aumentando su producción de software basado en estas líneas propietarias que impiden de cierta forma la comercialización de los productos por el pago de patentes y las prohibiciones que impone el bloqueo.

En estas circunstancias no es de extrañar que se creen estrategias de migración a software libre, y la utilización de herramientas de libre distribución y código abierto. Todo esto implica libertad y soberanía tecnológica, libertad para utilizar el software, modificarlo, distribuirlo, etc.

Se denomina software libre al que otorga a los usuarios las libertades de usarlo con cualquier propósito, estudiar su funcionamiento y adaptarlo a las necesidades, distribuir copias, y publicar sus mejoras en beneficio de la comunidad. La Fundación del Software Libre (FSF), creada en 1985, se dedica a dirigir, mantener e implantar licencias libres de restricciones para el usuario. Promociona el uso y desarrollo del software libre en todas las ramas de esta esfera. De modo más preciso el software libre se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito.

- La libertad de estudiar cómo funciona el programa, y adaptarlo a las necesidades. El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias.
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. El acceso al código fuente es un requisito previo para esto.

1.3.2 GNU Linux

Linux es parte de la implementación de libre repartición de UNIX para laptop, servidores, y oficinas. Muchas personas colaboran con la mejora de este sistema operativo debido a la libertad que posee su código fuente, permitiendo cualquier modificación sobre él. Lo que da lugar a una fuerte y estable plataforma de usuarios y de desarrollo, siendo posible que actualmente millones de usuarios hayan optado por usar *Linux* como sistema operativo preferido. Independientemente de que versión de *Linux* se utilice, el fragmento de código común a todas es el núcleo de *Linux*. Si bien el núcleo se puede modificar para incluir el soporte para las utilidades que desee incorporar, cualquier núcleo de *Linux* ofrece las siguientes características:

Multiusuario: No solo puede tener muchas cuentas de usuario disponibles en un sistema *Linux*, sino que también puede tener múltiples usuarios conectados y trabajando en el sistema al mismo tiempo. Los usuarios pueden tener sus propios entornos organizados como ellos deseen: su propio directorio de inicio para almacenar archivos y su propia interfaz de escritorio.

Multitarea: En *Linux* es posible tener muchos programas ejecutados al mismo tiempo, lo que implica que se pueden ejecutar programas en segundo plano. Muchos de estos procesos del sistema hacen que *Linux* pueda funcionar como servidor.

Interfaz Gráfica de Usuario, GUI (Sistema X Window): El potente marco de trabajo con aplicaciones gráficas en *Linux* se denomina Sistema X *Window* (o simplemente X). X maneja las funciones de apertura de aplicaciones GUI basadas en X y de visualización de las mismas en un proceso de servidor X (el proceso que gestiona la pantalla, el ratón y el teclado).

Compatibilidad de hardware: Puede configurar compatibilidad para casi cualquier tipo de hardware que se puede conectar a un ordenador. Hay compatibilidad para disqueteras, CD-ROM, discos extraíbles (como, por ejemplo, unidades Zip), tarjetas de sonido, dispositivos de cinta, tarjetas de video entre otros.

Conectividad a la red: Para conectar un sistema *Linux* a una red*, Linux* ofrece soporte para una serie de tarjetas, módems y dispositivos seriales de Red de Área Local (LAN).

Servidores de red: Con *Linux* se puede tener acceso a una gran variedad de paquetes de software que le permiten utilizarlo como servidor de impresión, servidor de archivos, servidor FTP, servidor de correo, servidor Web, servidor de noticias o servidor de grupo de trabajo.

Al compararlo con otros sistemas operativos disponibles en el mercado, los principales valores de *Linux* son su precio y su fiabilidad. En términos de fiabilidad, el consenso general es que *Linux* es comparable a muchos sistemas comerciales UNIX, pero más fiables que *Windows NT* o *Windows 2000*. Otra ventaja de utilizar *Linux* es que siempre hay ayuda a la que recurrir en Internet brindando la posibilidad de que cada día se realicen aportes a este poderoso sistema operativo (Stallman, 2004).

Debido al proceso de migración hacia software libre en el que se encuantra inmerso el pais y las ventajas que ofrece la utilización del mismo se determinó elaborar el paquete de instalación para la distribucion de *Linux*, UBUNTU-8.10, de esta manera el trabajo será realizado en función de dicho sistema operativo.

1.3.3 Gestor de Base de datos PostgreSQL.

PostgreSQL es un sistema gestor de base de datos objeto-relacional. Es ampliamente considerado como una de las alternativas de sistema de bases de datos de código abierto. Soporta gran parte del SQL estándar y muchas funcionalidades modernas como son: consultas complejas, llaves foráneas, disparadores (*triggers*), vistas, integridad transaccional, así como el control de versionado concurrente (MVCC) que es una estrategia de almacenamiento que permite trabajar con grandes volúmenes de datos.

Cumple completamente con las características ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) para realizar transacciones seguras. Además, *PostgreSQL* puede ser personalizado por el usuario en muchas formas, según sus necesidades, por ejemplo: adicionando un nuevo tipo de datos, operadores, funciones, y lenguajes procedurales.

Utiliza licencia BSD, por lo que *PostgreSQL* puede ser utilizado, modificado y distribuido por cualquier usuario gratuitamente, para cualquier propósito ya sea con fines privados, comerciales o académicos.

Como características generales se pueden destacar:

Alta concurrencia: Mediante un sistema denominado MVCC (Acceso concurrente multiversión) *PostgreSQL* permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos.

Amplia variedad de tipos nativos: Provee nativamente soporte para números de precisión arbitraria, texto de largo ilimitado, figuras geométricas (con una variedad de funciones asociadas), direcciones IP (IPv4 e IPv6), bloques de direcciones estilo CIDR y direcciones MAC.

Soporte SQL92/SQL99: Implementa un subconjunto extendido de los estándares SQL92 y SQL99.
Transacciones: Permiten el paso entre dos estados consistentes manteniendo la integridad de los datos.

Integridad referencial: Soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.

Bloqueos de tabla y filas: Ofrece varios modos de bloqueo para controlar el acceso concurrente a los datos en tablas, algunos de estos modos de bloqueo los adquiere *PostgreSQL* automáticamente antes de la ejecución de una declaración, mientras que otros son proporcionados para ser usados por las aplicaciones.

Constraints y triggers: Posee *constraints*[15] y *triggers* que tienen la función de mantener la integridad y consistencia en la BD.

Múltiples tipos de datos predefinidos: Como todos los manejadores de bases de datos, *PostgreSQL* implementa los tipos de datos definidos para el estándar SQL3 y aumenta algunos otros.

Soporte de tipos y funciones de usuario: Soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.

Conectividad: Posee conectividad TCP/IP, JDBC y ODBC.

[15] *Constraints:* son los encargados de asegurar la integridad referencial en la base de datos.

Interfaz con diversos lenguajes: Contiene modelos de interfaz para C, C++, Java, Delphi, Python, Perl, PHP entre otros.

Instalación Ilimitada: Al no presentar costo asociado a la licencia del software *PostgreSQL* puede ser instalado siempre que sea necesario.

Ahorros considerables en costos de operación: *PostgreSQL* ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características de estabilidad y rendimiento.

1.3.4 Shells o intérpretes de comandos

Un *shell* o intérprete de comandos es la interfaz básica que permite la comunicación con el ordenador y arrancar los programas que se guarda en el disco duro. Internamente no es más que un programa que espera a que se le introduzca una orden o comando para ejecutarlo, tras lo cual vuelve a quedarse a la espera del siguiente comando. Esta funcionalidad básica es un punto común que tienen todos los intérpretes de comandos de cualquier sistema operativo. A partir de aquí, dependiendo de la *shell* y de los recursos que ponga a su disposición el sistema operativo, aparecen diversas extensiones que pueden ir desde una interfaz intuitiva y totalmente gráfica, hasta el autentico lenguaje de programación que se encuentra en casi todos los *shells* de los sistemas operativos del entorno Unix.

1.3.4.1 Bourne Shell (sh)

Bourne Shell intérprete de comandos desarrollado por Stephen Bourne, es considerada la *shell* original de UNIX. A pesar de pretender ser un intérprete de comandos interactivo, ganó popularidad como lenguaje de guiones (script). La principal dificultad radica en el diseño de un depósito completamente programable, que pude servir para que los usuarios escriban comandos en una terminal interactiva como interfaz. Es invocado por el comando sh. Actualmente este *shell* no es el más utilizado, ya que las versiones más recientes de intérpretes de comandos ofrecen recursos de los que carece *Bourne shell*, como es el caso de la edición de las líneas de órdenes, la recuperación de órdenes emitidas previamente, y los alias para las órdenes de uso común. Entre los recursos importantes que ofrece *Bourne shell* están los operadores para la ejecución en segundo plano, o ejecución condicional de órdenes, los enunciados para repetir la ejecución de órdenes, incluida la iteración a lo largo de una secuencia de valores que pueden asignarse a

una variable de iteración, las variables sustituibles, posee tres formas de entrecomillado, además de realizar la captura de señales, y ejecución de órdenes específicas cuando ocurre una señal determinada.

1.3.4.2 C Shell (csh)

C Shell, lenguaje de programación conocido como csh, fue desarrollado por Bill Joy y está incluida en todas las versiones de UNIX más recientes. Este lenguaje ha sido considerado más idóneo que Bourne Shell, a pesar de que actualmente no presenta un amplio uso ya que ha sido superado por otros *shell*. Recibe su nombre debido a la semejanza de sus comandos con el lenguaje de programación C, lo cual hace a la *shell* más fácil de aprender para los programadores. Al poseer una sintaxis muy parecida al lenguaje C, hace que muchas veces los archivos elaborados para csh no pueden ser ejecutados bajo otro *shell* dígase bash o ksh. Su indicador es el signo de porcentaje (%). Brinda varias capacidades básicas entre las que se puede encontrar la ejecución de varios trabajos simultáneamente, con o sin la intervención del usuario, puede parar la ejecución de un trabajo y empezar de nuevo, ejecuta comandos usados previamente, así como personalizar el ambiente a las necesidades personales de cada usuario.

1.3.4.3 Trustedi C Shell (tcsh)

Trustedi C Shell (tsch) *shell* o intérprete de comandos, el cual fue iniciado por ken Greer en los años 70, continuado por Paul Placeway en los años 80, luego Wilfredo Sánchez (ex líder de ingenieros de Mac OS X) a principio de los años 90 y desde entonces miles de personas le ofrecen mantenimiento alrededor del mundo. Esta *shell* es una versión mejorada del intérprete de comandos C shell y es ejecutada bajo la orden tsch. En 1984, tcsh reemplazó a csh como el intérprete de comandos por defecto en FreeBSD, y recientemente, en Mac OS X. Entre los recursos adicionales que ofrece se encuentra la capacidad para editar la línea de órdenes interactivamente, realiza llamadas sencillas de órdenes ejecutadas con anterioridad, las cuales se pueden editar, ofrece también la capacidad para programar la ejecución periódica de una orden, además posibilita la consulta de la documentación sobre una orden cuando es tecleada.

1.3.4.4 Korn Shell (ksh)

El Korn shell (ksh), intérprete de línea de órdenes para sistemas Linux y Unix. Fue desarrollado por David Korn, es un lenguaje totalmente compatible con Bourne Shell además de incluir elementos del intérprete de comandos C Shell.

Korn Shell es unos de los intérpretes de comandos más notables que han venido surgiendo en los últimos años. Incorpora las mejores cualidades de Bourne Shell y de C Shell. Añade posibilidades de programación avanzada, facilidades aritméticas y mayor rapidez de ejecución. Es invocado por la orden ksh. Una de las ventajas más significantes que ofrece este intérprete de comandos es su capacidad de ejecutarse en muchos sistemas diferentes, esto hace posible que programas escritos para un sistema sean ejecutados sin tener que ser cambiados en otros muchos sistemas , además que les hace posible a los programadores que estén familiarizados con ksh usar nuevos sistemas sin tener que pasar largos períodos de aprendizaje, y usar varios sistemas a la vez sin la confusión causada cuando cada sistema presenta una interfaz distinta. Comparado con otras *shell* ksh tiene beneficios para ser usado como lenguaje de comandos y beneficios para ser usado como lenguaje de programación.

Uno de los inconvenientes es que se distribuye de forma comercial y es por esta razón que otras *shells* han tenido un mayor auge. Entre las características más significativas de ksh se encuentra la edición interactiva de la línea de órdenes, ofrece mejores definiciones de funciones que brindan variables locales y permiten escribir funciones recursivas, además posee la capacidad para cambiar con facilidad de un directorio a otro.

1.3.4.5 Bourne Again Shell (Bash)

Bash es un intérprete de comandos del proyecto GNU. Este es una versión de *Bourne Shell* (sh) que incluye nuevas funcionalidades. Es un lenguaje interpretado de programación, que facilita el trabajo a los administradores, para realizar la mayoría de las actividades en el sistema, por lo que es más que una simple consola. Su mayor utilización se ve en los sistemas Unix. Este intérprete incorpora algunas características que lo convierte en uno de los intérpretes de comandos más utilizados a nivel mundial. Como lenguaje de alto nivel proporciona variables, control de flujo, funciones, control de procesos, redirección de entrada/salida y un lenguaje de orden para escribir programas por script.

Incluye características de otros intérpretes de comandos como es el caso de la sintaxis Bourne Shell, el redireccionamiento, comandos y variables de *Korn shell* y la edición de comandos de *C shell* los cuales forman parte de la familia de intérpretes de comandos de la Unix. Siendo considerado como el intérprete de comandos más extendido en *Linux,* posee una sintaxis familiar y asequible para programas externos,

propicio para sistemas de autocompilado[16]. El soporte del mismo para funciones permite al sistema acoger un diseño modular fácil de entender, sacando provechos del mismo permitiendo a los mantenedores de paquetes y a los desarrolladores reconfigurarlo inmediatamente.

Luego de realizar un estudio de los diferentes intérpretes de comandos y teniendo en cuenta las características y funcionalidades que brindan cada uno se escogió Bash para la elaboración del paquete de instalación, ya que es el intérprete de comandos más extendido y utilizado en Linux, posee una sintaxis familiar y asequible al usuario, además de ser un lenguaje apropiado para la realización de operaciones por consola.

Conclusiones del capítulo

Todos los conceptos y definiciones brindados anteriormente posibilitan una mejor comprensión del proceso de replicación de datos y de las soluciones que se han desarrollado para mejorar el mismo, y hacerlo más simple, para lograr este objetivo se llevó a cabo un estudio de todo el proceso de replicación de datos, así como las ventajas y desventajas del mismo. Se realizó además un estudio general de las soluciones de réplicas existentes para servidores *PostgreSQL* seleccionado *Cybercluster*, *Pgcluster* y *Slony* como herramientas de replicación a empaquetar. Se detalla algunas de las últimas tendencias y herramientas que a nivel mundial se vienen utilizando en el mundo del software, y además se realizo un estudio de los principales *shells* o intérpretes de comandos seleccionando *Bash* para la elaboración del paquete de instalación.

[16] *Sistema autocompilado: Código intermedio que automatiza la llamada a programas externos.*

Capítulo #2. Caracterización de las soluciones de réplicas a empaquetar

Introducción

A nivel mundial se han creado herramientas para la replicación de datos, cada una de ellas tiene características específicas que les posibilita diferenciarse una de otra, en el presente capítulo se brinda una caracterización de las herramientas seleccionadas para formar parte de la solución que brindará dicho trabajo. De manera que queden tratados los aspectos fundamentales de cada solución, para poder fundamentar la elección que se realizó.

2.1 Fundamentación de la elección de las soluciones de réplicas seleccionadas

Después de haber realizado un estudio sobre las soluciones de replicación de datos en servidores *PostgreSQL* se definieron como herramientas a empaquetar *Slony*, *Pgcluster*, *Cybercluster*. Esta elección estuvo basada fundamentalmente en varias entrevistas realizadas a diferentes áreas de la UCI donde se utilizan técnicas de replicación de datos entre servidores. Dichas entrevistas fueron realizadas de forma informal, debido a que no se tenían definidos objetivos específicos, sino que estaba encaminada a recopilar la mayor cantidad de información y documentación de las técnicas utilizadas en cada una de las áreas. Esta actividad se realizó de manera satisfactoria ya que los resultados alcanzados fueron lo suficientemente sólidos para realizar una caracterización general de las herramientas seleccionadas. Las áreas donde se realizaron dichas entrevistas fueron el Centro de Tecnologías de Almacenamiento y Análisis de Datos (CENTALAD), el proyecto ERP Cubano, el proyecto SINAPSIS y el centro UCI-FAR(UCID) .

2.2 Herramienta de replicación de datos Slony

Slony es una de las herramientas de replicación más utilizadas para entorno maestro-esclavo. Es un sistema de replicación maestro a múltiples esclavos, soportando la réplica en cascada. Por ejemplo, un nodo A puede alimentar otro nodo B que alimenta a su vez a otro C, por lo que permite a un esclavo ser a su vez maestro para otro servidor.

Incluye los tipos de funcionalidades necesarias para replicar bases de datos grandes a un número razonablemente limitado de sistemas esclavos (aproximadamente 12 esclavos inmediatos). Además permite indicar qué cambios replicar de un servidor a otro. Es un sistema diseñado para su uso en centros de datos y para hacer copias de seguridad, así como para ser usado en sitios auxiliares dónde el modo normal de funcionamiento es que todos los nodos estén disponibles a tiempo completo y donde todos puedan asegurarse. Es por este motivo que, aunque está demostrada su tolerancia a fallos, no se aconseja usarse cuando se tienen nodos que probablemente se puedan quedar fuera de la red, que estén incluidos dentro de una red inestable o en sitios donde la configuración cambia de manera casual.

Actualmente se está desarrollando la segunda versión de *Slony* denominada *Slony-II* que a diferencia de la versión que se utiliza será multi- maestro y presentará muchas características a considerar.

Slony implementa la réplica asincrónica, usando disparadores (*triggers*) para determinar las actualizaciones de las tablas, donde un solo origen (maestro) se puede replegar a los suscriptores múltiples (esclavos) incluyendo suscriptores conectados en cascada. (Slony Develoment Group, 2007)

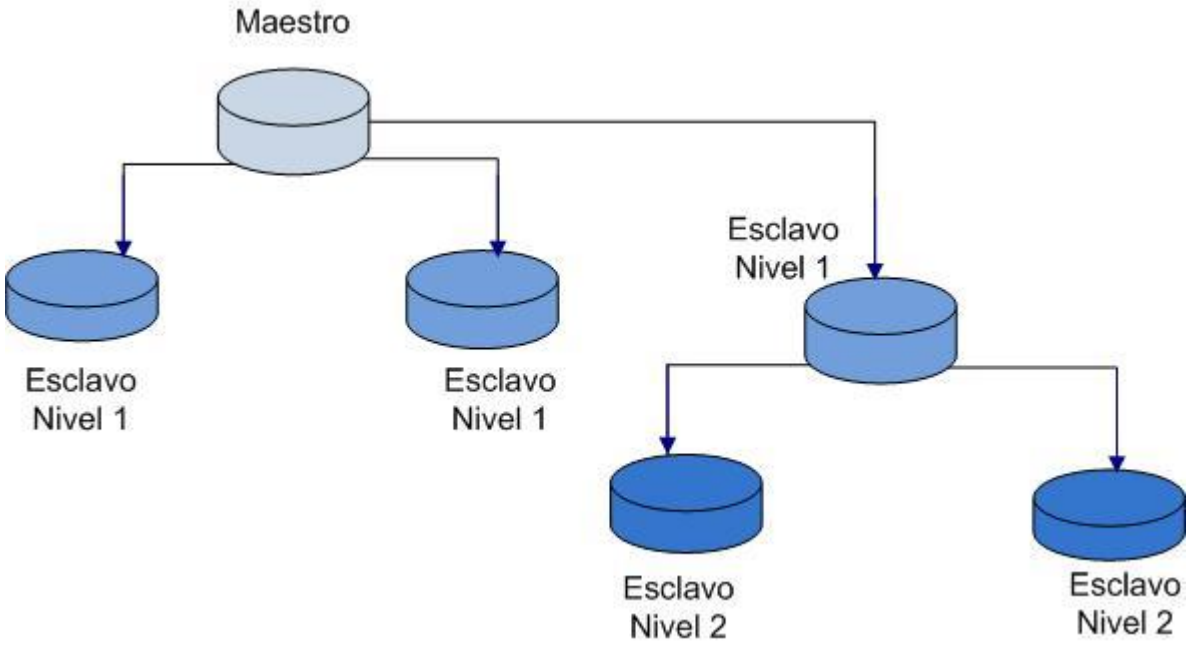


Fig. 2.1 Replicación con Slony.

Para lograr un trabajo eficiente al replicar utilizando Slony, se necesita tener un conocimiento preliminar de los conceptos asociados a la herramienta, como son:

Clúster: Es el conjunto de instancias de bases de datos *PostgreSQL* que están envueltos en la replicación.

Nodo: Se le llama así a cada una de las bases de datos envueltas en la replicación.

Replication set: Es el conjunto de tablas y/o secuencias a ser replicadas. En un mismo clúster puede haber varios *sets.*

Origin: Es el nodo principal (maestro), es el único en el que se puede escribir.

Subscribers: Son todos los demás nodos en el clúster (esclavos), los que reciben los datos en la réplica.

Providers: Es un nodo esclavo que sirve como proveedor para un subconjunto de nodos en el clúster (actúa como un nodo *origin* pero no se permite a ninguna aplicación escribir en él).

Switch-over: Acción de tomar un servidor esclavo el papel del maestro.

2.2.1 Características Generales

2.2.1.1 Requisitos

Para trabajar con *Slony* hay que tener en cuenta un conjunto de requisitos de configuración como:

Tener instalada versiones de *PostgreSQL* mayores o iguales a la 7.3.3 (7.4.8 o superior es recomendado) incluyendo la instalación de los paquetes postgresql-slony1 y slony1-bin que vienen asociados a la versión de *PostgreSQL* instalada. Verificar en los archivos de configuración de *PostgreSQL* que esté aceptando conexiones (listen_addresses='*') que se encuentra en **postgresql.conf** y configurar además el **pg_hba.conf** para indicar de que subred se pueden conectar. Otro requisito importante es definir la estructura del clúster (a cada nodo del clúster se le debe identificar con un número) y por último definir los *replication sets* (llaves para las tablas que no tienen un PK, tablas, secuencias).

2.2.1.2 Configuración

Todos los nodos involucrados en la replicación deben estar usando un *timezone*[17] reconocido por *PostgreSQL,* se debe setear[18] en el archivo **postgresql.conf**. También se deben crear los roles y espacios de tablas en los nodos, crear la base de datos y el esquema de la base de datos en los nodos. Además de que se usa un rol específico para la replicación.

2.2.1.3 Mantenimiento

En caso de añadir una nueva tabla es necesario crear un nuevo *replication set además a*ñadir la tabla y/o secuencia al nuevo *set y s*ubscribir los nodos al *replication set* con la ayuda de la sentencia *Merge set.* En caso de quitar una tabla se utilizan las sentencias *set drop table* y *set drop sequence*. Para añadir un nuevo *replication set se* utiliza *create set*, *set add table / set add sequences* y *subscribe set.* En caso de ser necesario quitar un *replication set* si es de un solo nodo, se utiliza la sentencia *unsubscribe set* y en caso de todos los nodo seria *drop set.*

2.3 Herramienta de replicación de datos Pgcluster

2.3.1 Características generales

Es un sistema de replicación para *PostgreSQL*. Una herramienta que actúa como un parche adicional que permite manejar clúster de bases de datos, balanceo de carga y replicación de servidores con la combinación de cada uno. Tiene la capacidad de sincronizar el árbol de directorios. (Pgcluster, 2005)

Funciona dentro de la misma base de datos, replicando los datos por consultas (funciones de BD que devuelven reportes condicionados).

Pgcluster puede construir dos tipos de sistemas:

- Sistema de Balance de carga (*load-balancing system*)
- Sistema muy disponible (*highly available system*)

2.3.1.1 Componentes

Pgcluster está constituido por un servidor de balance de carga, un servidor de clúster, un servidor de replicación.

[17] *Timezone: Zona* horaria
[18] Setear: establecer la configuración correcta de un programa o hardware

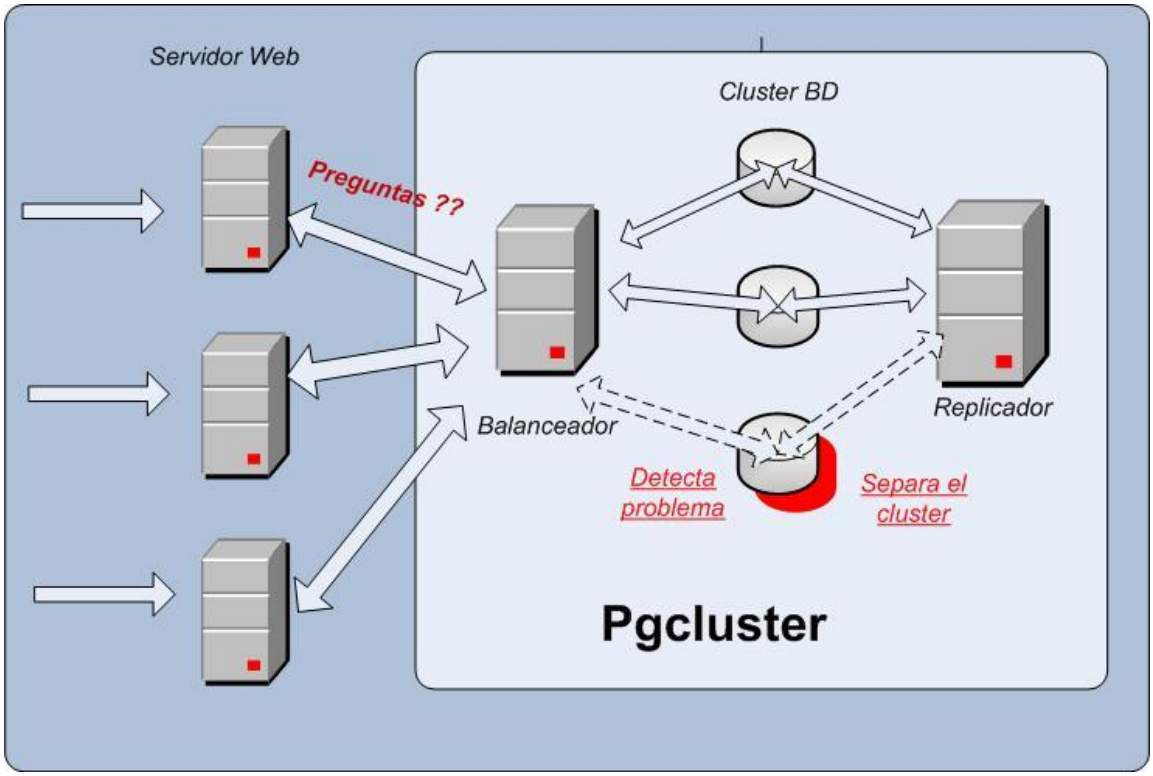


Fig. 2.2 Ejemplo de la composición de un sistema de alta disponibilidad Pgcluster.

Servidor de Balance de Carga: Recibe la consulta del cliente, y envía una consulta a la base de datos del clúster con el más bajo índice de carga. Con la combinación del clúster de BD y un replicador, *Pgcluster* puede distribuir la carga de acceso. Además el balanceador de carga verifica que no haya problema en el momento de comunicación con el clúster BD. Cuando el problema del clúster BD se descubre, el balanceador de carga lo separa del sistema. Aunque no exista ninguna necesidad de un servidor de balance de carga la construcción de un sistema *load-balancing*, es requerido para la construcción de un sistema de alta disponibilidad.

Servidor de Clúster: Conocido también como Clúster PP el servidor que recibe la consulta a la base de datos de un usuario. Un acceso de carga se puede distribuir por el aumento del número de agrupaciones PP. Clúster PP será ejecutado de inmediato, cuando la consulta de referencia a los datos se reciba. La otra consulta se pasa al servidor de replicación y requiere un duplicado.

Servidor de Replicación: El servidor de replicación envía la consulta de la base de datos de clúster para cada grupo de PP. La replicación del servidor guarda la orden recibida y envía una consulta a cada uno de los grupos PP. El control de la replicación del servidor es el problema de la categoría PP en el momento de la comunicación con la base de datos de clúster. Cuando se detecta un problema, la replicación del servidor separa de la categoría PP posteriores repeticiones.

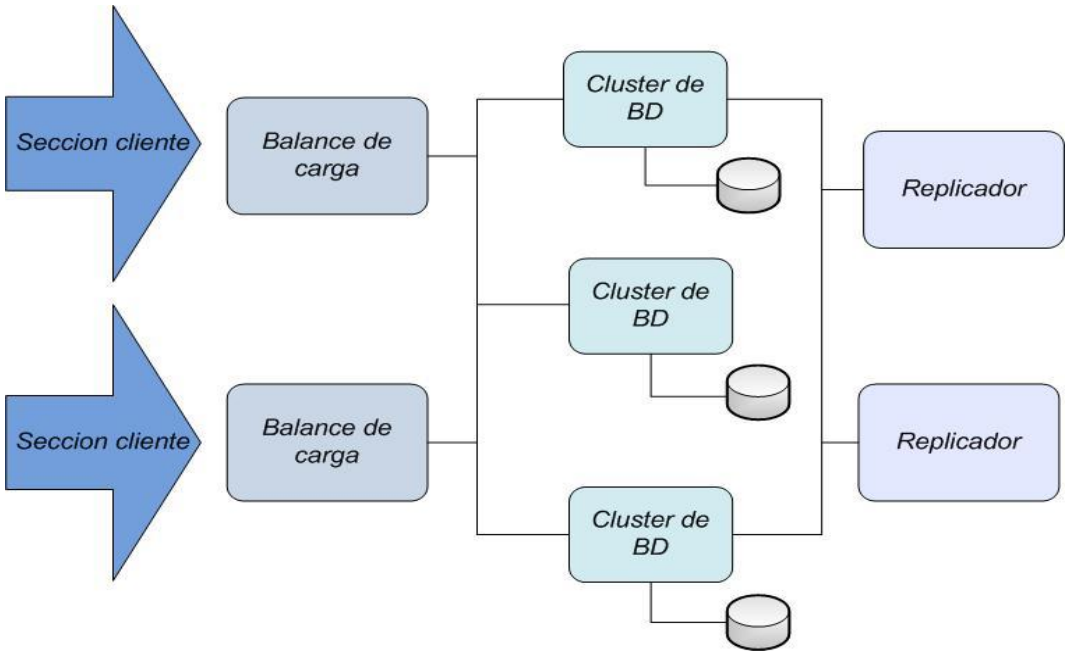


Fig. 2.3. Representación de la interrelación del balanceador de carga, clúster BD y servidor de replicación.

2.3.1.2 Funcionalidad

Pgcluster tiene dos funciones principales:

Compartir carga: La carga de la sesión de las demandas es distribuida. Es efectivo en aplicaciones Web donde existe gran demanda por el número de peticiones.

Alta disponibilidad: Cuando ocurre un fallo en el clúster BD, el servidor de balance de carga y el de replicación separan el fallo del sistema, y continúa el servicio usando los clústeres BD restantes. El clúster BD cuando es reparado puede restaurarse dinámicamente a un sistema, sin detener el servicio. Posteriormente los datos se copian automáticamente en la BD restaurada o añadidos desde otra BD.

2.3.1.3 Arquitectura

Esta herramienta está conformada por un balaceador de carga, el cual hay que configurar previamente, poniendole el nombre del host, el IP que posee, y los puertos por donde va recibir y enviar datos, se tiene además el (los) servidores de réplica, que también se le configura los mismos parámetros que al balanceador de carga, y la misma operación para los clústeres BD.

La arquitectura de esta herramienta quedaría de la siguiente forma:

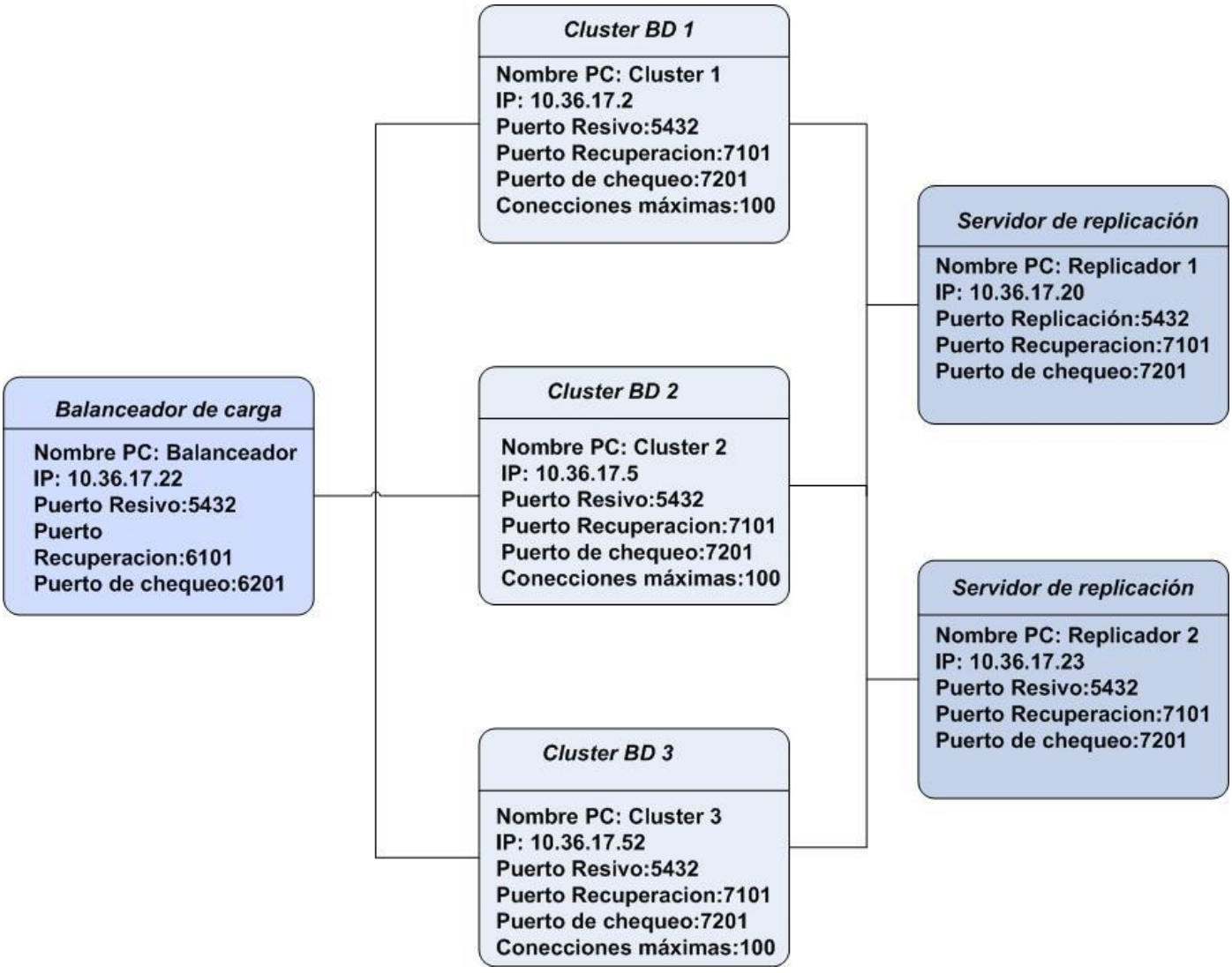


Fig. 2.4. Arquitectura de Pgcluster.

A continuación se muestra la configuración detallada del balanceador de carga y su relación con los clúster de BD, ya que estos elementos tienen que ser configurados previamente.

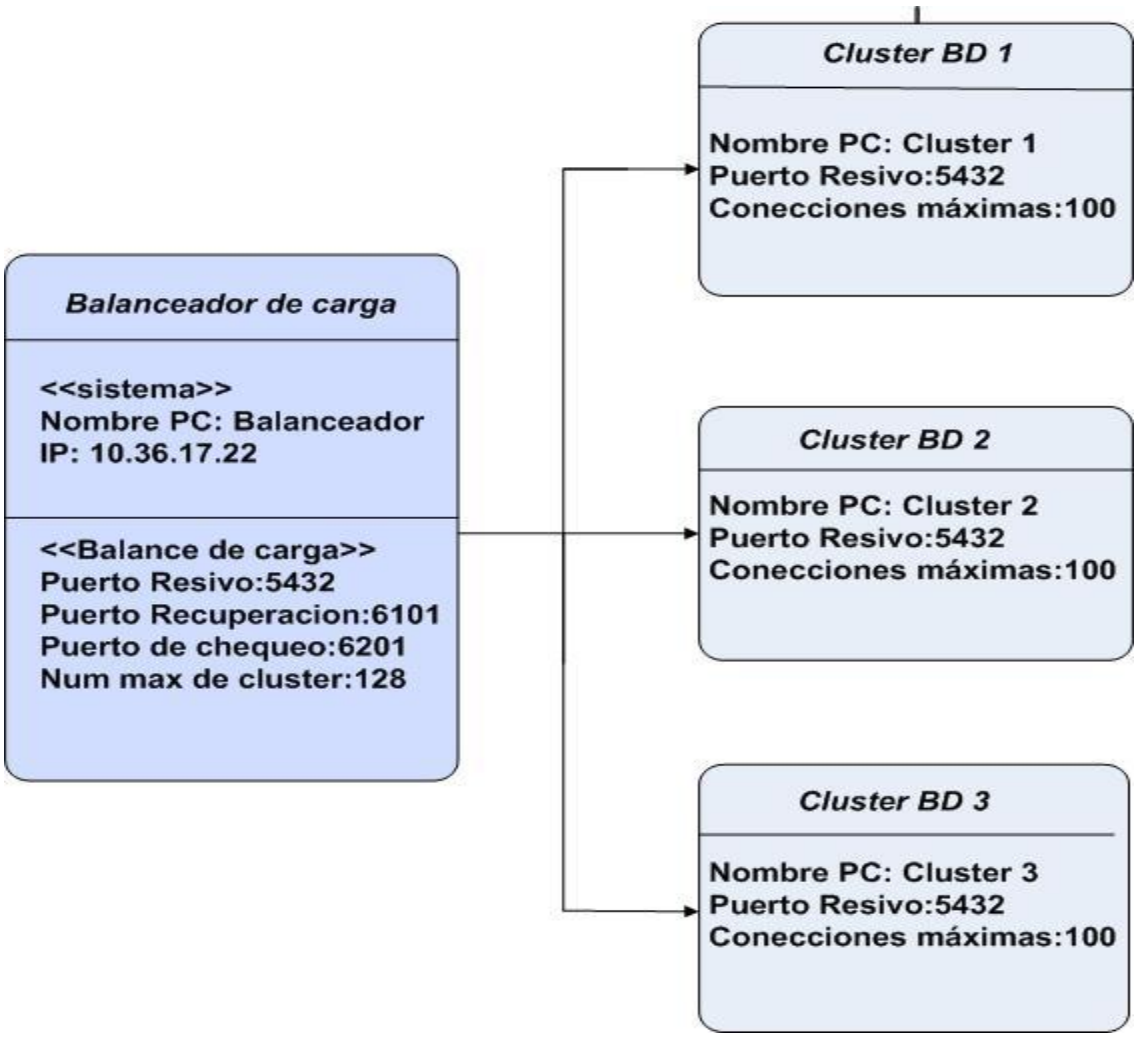


Fig. 2.5. Arquitectura del servidor balance de carga.

2.3.1.4 Diseño lógico

El diseño lógico ideal para el correcto funcionamiento de esta herramienta, consiste en un balanceador de carga conectado a dos clústeres de BD, además de dos servidores de réplica que también van conectados ambos a los clústeres. El balanceador realizaría la acción de compartir las cargar para que no exista un clúster sobrecargado mientras el otro está libre. Y los servidores de replicación harían posible mantener la información entre ambos clústeres actualizada.

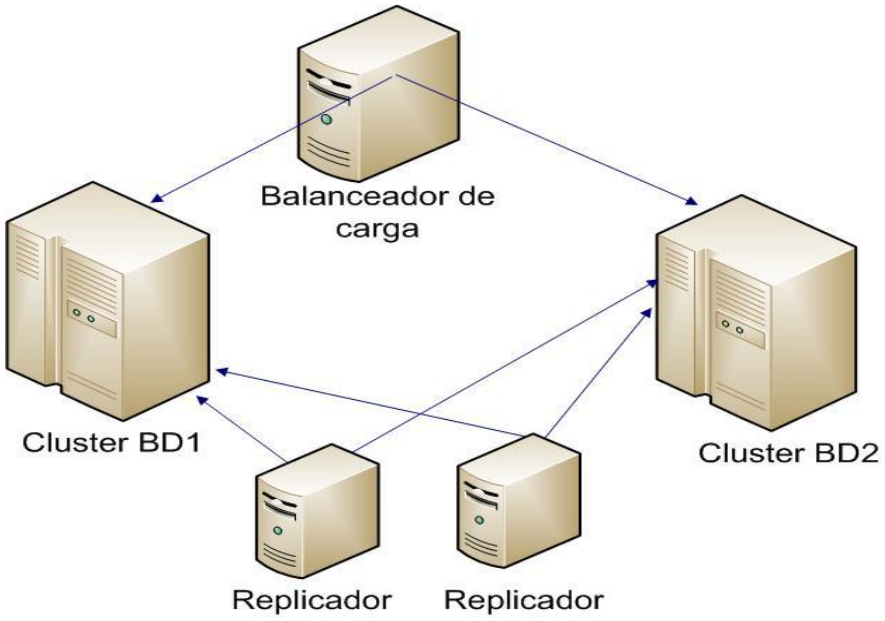


Fig. 2.6. Funcionamiento lógico de Pgcluster.

2.3.1.5 Sincronización de los datos

Pgcluster usa rsync para la sincronización de datos con bases de datos maestro, en el momento de adición del clúster BD o la restauración. Por lo tanto, rsync es un requerimiento a la hora de usar *Pgcluster*. El rsync es un comando para copiar archivos entre servidores así como el comando rcp. Lo bueno de rsync es la habilidad de poder elegir el archivo para una copia por la diferencia que haya en un archivo o disminuir el tiempo de transferencia por compresión de los datos que se transmiten, puede prevenir una fuga de información por la encriptación de datos, o puede acreditar a un usuario con seguridad usando autenticación RSA.

2.3.1.6 Estructura y sucesión de procesos

El servidor de balance de carga recibe la pregunta del cliente, y envía una pregunta al clúster BD con la proporción de carga más baja, verifica que no haya problema en el momento de comunicación con el clúster BD. Cuando el problema del clúster BD se descubre, el balanceador de carga separa el clúster BD del sistema.

Aunque no exista ninguna necesidad de un servidor de balance de carga la construcción de un sistema *load-balancing*, es requerido para la construcción de un sistema de alta disponibilidad.

2.3.1.7 Parámetros de Configuración

Con el fin de resolver la dirección de cada servidor, los nombres de servidores y direcciones IP se describen en el archivo "/etc/hosts" .Posteriormente cuando se inicializa la base de datos se crea la plantilla de los archivos de configuración. Para permitir la conexión a una red el clúster BD se conecta a un servidor de balance de carga o a un servidor de replicación a través de la red. Luego se configura un puerto de recepción, y el número máximo de conexiones .Cuando esté configurado se tiene en cuenta el entorno de máquina y el tamaño de la base de datos.

El archivo **cluster.conf** es el archivo de configuración de clúster BD. Después del proceso de inicialización de la BD, un ejemplo de dicho archivo es creado automáticamente en el directorio de almacenamiento de la BD y se edita el archivo de ejemplo acorde con el entorno de cada servidor.

Para la configuración del servidor de replicación se declara el valor de configuración del servidor de replicación (nombre del host, el número de puerto para replicación, número de puerto para recuperación). Un servidor de replicación puede ser configurado más de una vez.

Para la configuración del Clúster BD se describe el número de puerto que recibe la demanda de recuperación de un servidor de replicación cuando se convierte en base de datos maestros. Posteriormente se describe el camino del comando *rsync* usado en el momento de recuperación.

Cuando todos los servidores de replicaciones se detienen y el servidor del clúster BD cambia el estado a *stand-alone*, aquí se puede especificar el modo de operación, este puede ser *real_only* (solo lectura) solamente acepta consultas de búsqueda y *read_write* (lectura y escritura) para aceptar también consultas de actualizaciones. Luego se describe el nombre del archivo de LOG que guarda el estado del servidor de replicación al cual este clúster BD está conectado y el nombre del archivo de LOG que guarda los

mensajes de depuración y los mensajes de errores de este clúster BD.

2.4 Herramienta de replicación de datos Cybercluster

Cybercluster es una herramienta basada en la versión *PostgreSQL* 8.2 que sincroniza la base de datos para asegurarse que todos los nodos dentro del clúster contienen los mismos datos. Posibilitando que todos los nodos puedan trabajar eficazmente al mismo tiempo. Entre sus principales funcionalidades se pueden encontrar, el equilibrio de carga para asegurar que todos los nodos puedan usarse durante el funcionamiento normal, además de contar con la posibilidad de manejo de errores, haciendo posible que no interrumpa el proceso cuando un nodo tenga alguna rotura, y por último la recuperación se realiza de forma automática(Company, 2009).

2.4.1 Características generales

2.4.1.1 Composición

Este proyecto cuyo código fue recientemente abierto integra y mejora varias herramientas existentes de manejo de clúster, como *Pgcluster* y *Pgpool*. Permitiendo procesar los cambios realizados en cada uno de los sitios de forma coordinada, además de ser el primer software en proporcionar la tecnología de repetición de inicio-fin. Proporciona una solución de replicación que asegura que su clúster de bases de datos es consistente en cada punto del tiempo, basándose en una arquitectura la cual consiste en una arquitectura distribuida en la que cada nodo es independiente y autosuficiente.

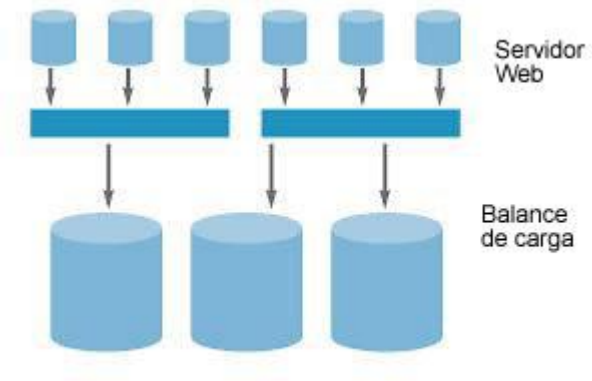


Fig.2.7. Modo lectura de la herramienta Cybercluster.

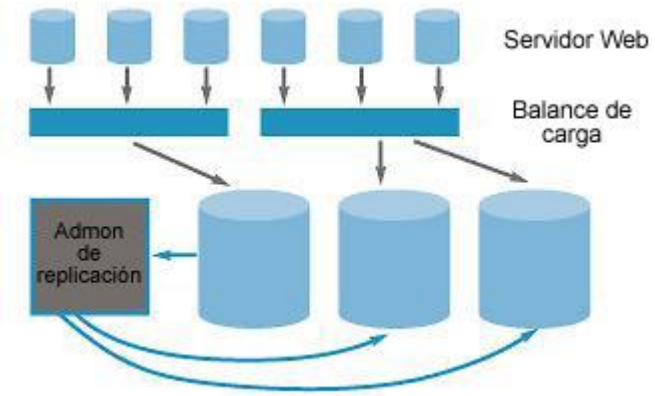


Fig. 2.8. Modo escritura de la herramienta Cybercluster.

Esta herramienta es una versión ampliada de *Pgcluster* por lo que también está formada por nodos clúster de BD, servidores de replicación y servidores de balance de carga componentes de alta importancia.

Clúster BD

Los nodos del clúster atienden las demandas reales que entran. Siempre que una demanda se reciba del cliente un clúster de BD tiene que determinar si está enfrentando una lectura o una escritura.

Leer se ejecuta directamente por la base de datos y el resultado se envía al cliente. En caso de que una operación de escritura se detecta, el clúster de BD envía un mensaje al administrador de la replicación, para asegurarse que se envía a los nodos restantes de la base de datos dentro del clúster.

Servidor de replicación

El servidor de la replicación es el componente central del sistema. Toma las demandas entrantes solo por los nodos de la base de datos y copia esos cambios a todos los nodos de la base de datos en el sistema. Si el administrador de la replicación descubre un problema en un nodo de la base de datos, quitará la máquina de la lista de nodos de la base de datos activos, revisa las líneas de código escritas en el disco que contienen una descripción detallada del error. Esto debe ser hecho para asegurarse que esos nodos de la base de datos pueden estar fuera de sincronización en caso del fracaso. Basándose en las líneas de código el administrador del sistema puede decidir entonces si es posible agregar la máquina de nuevo al clúster o no.

Servidor de balance de carga

Cybercluster contiene su propio balance de carga que puede usarse para distribuir la carga dentro del clúster. La carga en el sistema es determinada por el número de preguntas activas. La máquina con el número más bajo de preguntas activas se escogerá para realizar una nueva demanda.

Si el balance de carga descubre un problema en un nodo de la base de datos, automáticamente destaca el nodo de la base de datos del sistema activo, para asegurarse que ninguna demanda se enviará a esa máquina rota. El balance de carga usando *Cybercluster* es optativo. Sin balance de carga las aplicaciones disponibles pueden conectar a cualquier nodo de la base de datos dentro del sistema. Este balance no está de ninguna manera involucrado con la replicación, solo distribuye carga entre los nodos.

2.4.1.2 Sistemas Operativos que soporta

Cybercluster es compatible con varios sistemas operativos, como es el caso de Windows (Desde Windows 2000), Linux (Debian, RedHat Enterprise Server, Suse Enterprise Server, etc.), FreeBSD, Mac OS X, Solaris (x86, Sparc).

2.4.2 Funcionalidades

Cybercluster proporciona un juego rico de rasgos, por lo que posee disímiles funcionalidades importantes

2.4.2.1 Balanceador de carga (Load Balancer)

Cybercluster proporciona algoritmos de balanceador de carga para asegurarse que todos los nodos del banco de datos pueden usarse durante el funcionamiento normal. Esto es sobre todo importante cuando se tiene lectura de alto rendimiento.

2.4.2.2 Equilibrio de carga (Error Handling)

Cybercluster proporciona los algoritmos vacilantes, para asegurarse que todos los nodos de la base de datos pueden usarse durante el funcionamiento normal. Esto es especialmente importante cuando viene a la lectura alto rendimiento.

2.4.2.3 Manejo del error (Error Handling)

Cuando el software descubre un error del clúster automáticamente destaca los nodos rotos del sistema activo y continúa su funcionamiento normal en las base de datos de las máquinas restantes. En cuanto el nodo roto se repare, puede agregarse seguramente de nuevo al sistema.

2.4.2.4 Recuperación (Recovery)

Cuando un clúster de BD empieza en modo de recuperación automáticamente manda a *rsync* usando un amo activo.

2.4.2.5 Restricciones

Al reproducir un objeto grande, necesita ser puesto en un directorio que pueda ser leído por todos los nodos del banco de datos.

2.4.3 Instalación de Cybercluster

Para realizar la instalación de la solución de réplica *Cybercluster* en Linux (cualquiera de sus versiones) es necesario tener inicialmente instaladas las herramientas para descompactar **tar, bzip2** además de algunas librerías como **zlib, g++, zlib1g-dev, libreadline-dev, libreadline-dbg, make, flex, bison.**

2.4.3.1 Instalación de la fuente

El paquete fuente puede ser compilado e instalado de la misma manera como *PostgreSQL*, se descompacta y se ejecuta el:

. / configure [opciones...]

make

make install

2.4.3.2 Instalación de binarios

Los paquetes binarios (por ejemplo, en Debian), puede ser instalado con la distribución de Linux habituales de gestión de paquetes de software.

2.4.4 Configuración

Para replicar con Cybercluster es necesario configurar varios archivos .En el archivo /bin se encuentran los scripts necesarios para levantar los clústeres, balanceador y replicador. Además de configurar cuatro servidores diferentes, para el caso de los clúster se configuran la cantidad que se tengan, el servidor de replicación y el servidor de balance de carga en caso de que sea necesario (Para mayor información sobre la configuración ver Anexo #1).

2.4.5 Iniciar Cybercluster:

Una vez el clúster se ha configurado pueden iniciarse los nodos de la base de datos usando el orden siguiente:

pg_ctl - D start < datadir> start.

El servidor de la replicación puede empezarse antes o después de que todos los nodos se pongan en marcha. El comando siguiente puede usarse:

pgreplicate - D <el datadir>

El balance de carga que usa el servidor tiene el orden siguiente puede empezarse:

pglb - D < datadir>

2.4.5.1 Recuperación de un nodo de la base de datos

Si un nodo de la base de datos se ha caído fuera del clúster por el fracaso del hardware, problemas relacionados con el SO, debe re-sincronizarse con el clúster que opera, antes de que sea de nuevo utilizable. Los comandos siguientes ofrecen esa posibilidad.

Cualquiera de estos tres comandos puede usarse:

- "- **R**" es contundente para usar la recuperación *rsync-based.* Este método también se llama recuperación fría porque el nodo de la base de datos que mantiene los datos de la recuperación se detendrá durante dicho proceso. La ventaja de este método es que no verán datos sucios que son modificados por la base de datos de la fuente.
- "- **u**" las fuerzas de la recuperación *rsync-basad* sin apoyar los directorios de los datos originales, acelera la recuperación considerablemente pero sólo debe usarse cuando el *rsync* ya se configura

propiamente (por ejemplo el usuario que accede con la contraseña ssh). Ésta también es una recuperación fría.

- "- **U**" proporciona la opción para usar una recuperación pg_dump-basad. Dependiendo del tamaño de su base de datos estos pueden tomar mucho tiempo. Sin embargo, a diferencia de los otros métodos de recuperación, ésta es una recuperación caliente, significando que el nodo de base de datos de fuente no se detendrá durante la recuperación como el pg_dump simplemente se comporta como un cliente de la base de datos normal. Éste es el único método que puede usarse si *Cybercluster* se usa para reproducir a una arquitectura del sistema diferente (por ejemplo alguna otra arquitectura de CPU).

2.4.5.2 Agregar los nodos de la base de datos

Pueden agregarse los nodos de la base de datos fácilmente a un clúster activo. Los comandos usados para agregar un nodo son iguales que los comandos de la recuperación.

2.4.5.3 Supervisando Cybercluster

Después de inicializar el servidor de la replicación (por defecto escucha en puerto 8401) es posible supervisar el clúster. El puerto puede cambiarse modificando el director en pgreplicate.conf.

2.5 Comparación entre herramientas propuestas a empaquetar

Cada solución de réplica posee sus propias características o indicadores que la diferencian unas de otras, como pueden ser el tipo de réplica, el modelo, el sistema operativo en la que está soportada entre otras. Cuando se realiza el proceso de replicación de datos se utilizan estos indicadores para escoger el tipo de réplica adecuada a las necesidades del usuario. Para facilitar el proceso de elección de la herramienta de réplica a utilizar, su muestra el siguiente cuadro resumen donde se recogen los principales indicadores a tener en cuenta.

Herramientas de replicación	**Indicadores**			
	Licencia	Tipo de réplica	Modelo de replicación	Sistema Operativo

Slony	Licencia de PostgreSQL (BSD)	Maestro-esclavo	Asíncrono	Fedora Core, Red Hat Linux, Debian GNU/Linux, Ubuntu/ Linux, FreeBSD.
Pgcluster	DEB	Multi-Maestro	Síncrono	FreeBSD, Linux, SunOS/Solaris.
Cybercluster	BSD	Multi-Maestro	Síncrono	Windows, Linux, FreeBSD, Mac OS X, Solaris.

Tabla. 2.1 Comparación entre las herramientas a empaquetar.

Conclusiones del capítulo

Con la caracterización realizada a las herramientas seleccionadas, se pueden apreciar las peculiaridades de cada una de ellas, aportándose información detallada de su instalación y configuración. Con este estudio se logra un mejor entendimiento de dichas herramientas y facilita su elección según las necesidades de los usuarios, donde se pueden apoyar en un cuadro resumen que realiza una comparación entre las herramientas a empaquetar de acuerdo a varios indicadores definidos para la caracterización, además en los anexos se encuentra la guía de instalación de las tres herramientas, donde se detalla el proceso de instalación y configuración de cada una de estas.

Capítulo #3 Presentación y validación del paquete de instalación

Introducción

El presente capítulo se enfoca en la presentación de la herramienta elaborada, donde se ofrece una descripción de la misma y su representación gráfica, brindando además una descripción de cada una de las funcionalidades que ofrece al usuario, su modo de empleo, lo cual viene acompañado de las respectivas guías de instalación de cada una de las soluciones de réplicas empaquetadas. Además se demostrará la validez y el funcionamiento de la herramienta elaborada mediante el método de validación Estudio de Casos.

3.1 Descripción general de la herramienta

El paquete de instalación que aquí se presenta es una ayuda para la instalación y configuración de las soluciones de réplicas que utiliza *PostgreSQL* específicamente, para *Slony*-|, *Pgcluster-1.7.0* y *Cybercluster-1.2.1,* soluciones que forman parte de la familia de software libre. El mismo contiene tres archivos nombrados Cybercluster, Pgcluster y Slony, además del fichero Selección.sh. Cada uno de estos archivos contiene un compactado con el instalador de las respectivas herramientas de replicación, así como el fichero Instalar.sh que permite realizar el proceso de instalación y configuración de dichas herramientas, además el fichero Desinstalar.sh para el caso de necesitar la desinstalación de las mismas. La idea general de la herramienta es que sea sencilla, práctica y fácil de usar en todos aquellos centros que utilicen la replicación de datos.

3.2 Características específicas

3.2.1 Composición

El paquete de instalación está conformado por los siguientes elementos:

- **Cybercluster:** archivo que contiene el instalador de la herramienta de replicación *Cybercluster*, un fichero para la instalación y configuración, y otro para realizar la desinstalación.
- **Pgcluster:** archivo que está formado por el instalador de la herramienta de replicación *Pgcluster*, un fichero para la instalación y configuración, y otro para realizar la desinstalación.

- **Slony:** contiene un archivo con los ficheros de configuración de la herramienta de replicación *Slony*, además de un fichero para la instalación y configuración, y otro para realizar la desinstalación.
- **Selección.sh:** este es el fichero que permite al usuario obtener una recomendación de la herramienta de replicación a usar según sus necesidades.

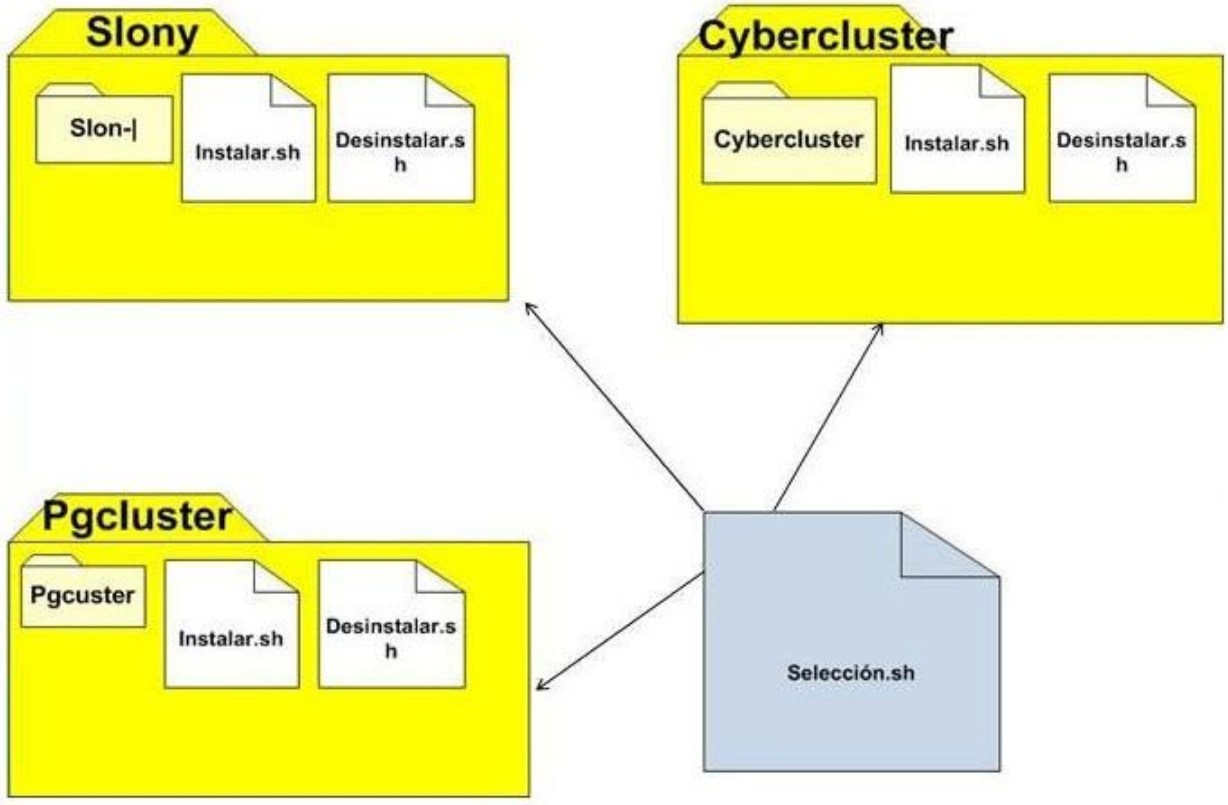


Fig. 3.1 Estructura del paquete de instalación de soluciones de réplicas.

3.2.1.1 Script de selección

Este fichero nombrado como **selección.sh** le brinda al usuario la posibilidad de seleccionar la solución de réplica adecuada, según las características del entorno en el que se vaya a realizar el proceso de replicación, el tipo de réplica que se requiera, así como el modelo de replicación a utilizar. Esto se logra de forma predeterminada, es decir, se brindan las combinaciones posibles de estos indicadores de modo que

el usuario escoja la combinación que cumpla con sus necesidades, y según la selección, obtenga el nombre de la solución de réplica que debe instalar así como la posibilidad de iniciar dicha instalación y configuración.

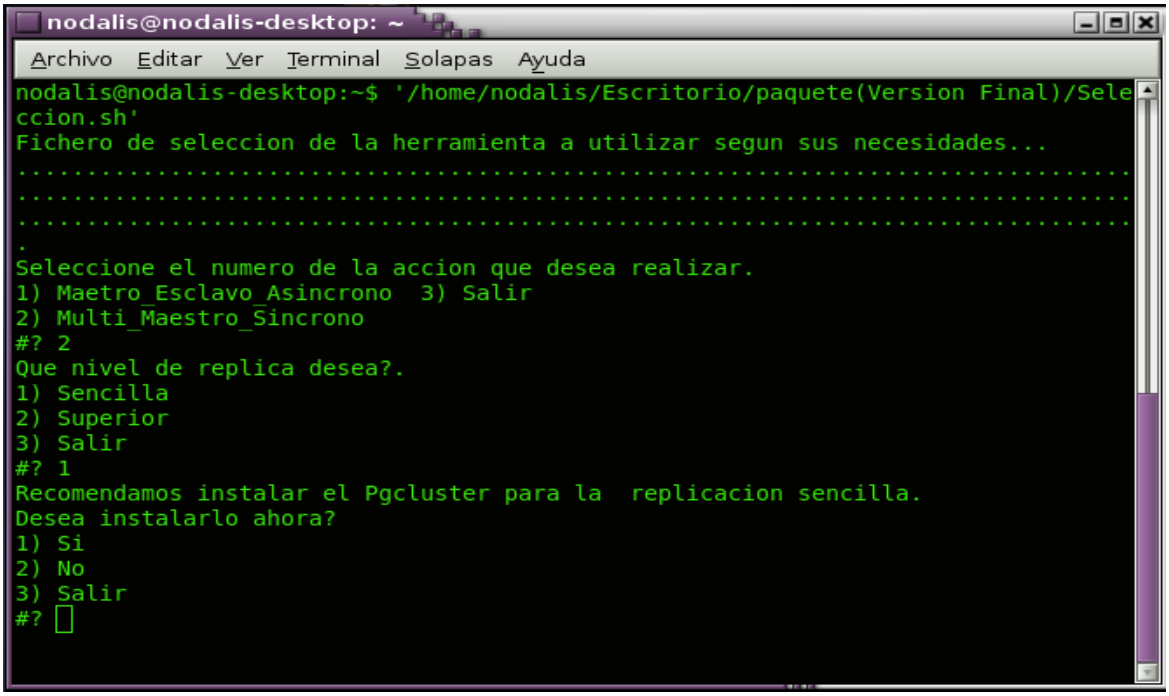


Fig. 3.2 Pantalla inicial para la selección de la solución.

Como se puede apreciar en la figura anterior se ofrecen dos posibilidades de replicación según las herramientas empaquetadas, de las cuales el usuario debe escoger la que necesite, para que la herramienta le devuelva la solución de réplica indicada a instalar.

3.2.1.2 Script de instalación y configuración de Cybercluster

Para la instalación del *Cybercluster* se cuenta con el fichero **instalar.sh**. Dicho fichero contiene una secuencia de comandos que permiten la instalación y configuración de la herramienta, independientemente de su comportamiento como clúster, servidor de replicación o balanceador de carga. El proceso de instalación y configuración actualmente se hace de forma separada, es decir, se compilaba el código fuente para la instalación y luego se accedía a los ficheros de configuración por separado. Con

esta herramienta se brinda la posibilidad de unir ambos procesos, de manera que la compilación y configuración de la misma se realice de forma conjunta, haciendo este proceso más sencillo al usuario.

Para iniciar este proceso se necesita instalar previamente algunas librerías que son necesarias, como es el caso de **tar, zlib1g-dev, zlib, bzip2, bzip3, gcc, g++, make, libreadline-dev, libreadline-dbg**, ya que van a ser usadas más tarde. Seguidamente se accede a la carpeta que contiene el instalador de la herramienta para descompactarla y poder realizar la instalación. Ya con las dependencias instaladas, y dentro del archivo descompactado se procede a realizar la instalación de la herramienta mediante los siguientes comandos:

```
. /configure
make
make install
```

Seguidamente se crea el usuario *Postgresql* y la carpeta **data,** esta última es donde una vez instalado *Postgresql* se almacenan los ficheros de configuración, además se dan los permisos para reconocerla como parte de la carpeta **pgsql** que es donde van a estar todos los archivos y ficheros de *PostgreSQL*.

Hasta aquí es la instalación de la herramienta *Cybercluster* con el *Posgresql-8.3* integrado (estos pasos son especificados en la guía de instalación que se encuentra en el Anexo #1).
Seguidamente de la instalación se ofrece la oportunidad de escoger el modo en que se va a configurar la herramienta.

En caso de la configuración de un servidor de replicación:

Cuando se quiere configurar el *Cybercluster* para que se comporte como servidor de replicación se hace necesario configurar el fichero **pgreplicate.conf** que se encuentra en el archivo share ubicado en:

```
cd /usr/local/pgsql/share
```

Luego el ejecutable muestra dicho fichero para; apoyándose con la guía; poder hacer las modificaciones pertinentes (ver las modificaciones en el Anexo #1). Y por último se inicia dicho servidor para ser utilizado.

En caso de la configuración de un servidor de clúster BD:

Para este servicio se requiere configurar tres ficheros ubicados en la carpeta **data** de postgres (/usr/local/pgsql/data/), los cuales son **postgresql.conf, cluster.conf, pg_hba.conf.** En el primero se

configura el puerto y se le da los permisos para escuchar desde toda la red, en el segundo se definen los ip del clúster, el replicador y del balanceador, ya en el último fichero se modifica la subred de donde se pueden conectar.

Y por último se inicia el clúster, para poder ser usado. Hasta este punto queda configurado el clúster de base de datos.

En caso de la configuración de un balanceador:

Para realizar esta configuración es necesario editar el fichero **pglb.conf** el cual copia la información que posee el fichero **pglb.conf.sample,** luego de ejecutar los cambios en el fichero, solo quedaría iniciar el balanceador.

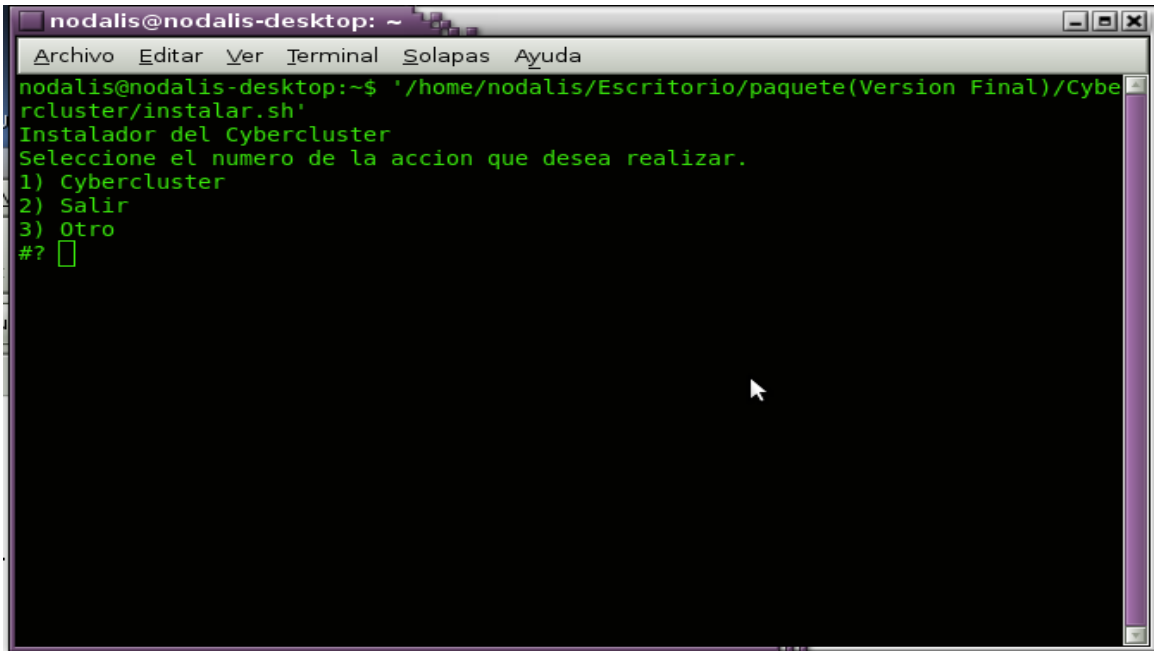


Fig. 3.3 Pantalla principal del Cybercluster.

3.2.1.5 Script de instalación y configuración de Pgcluster

Para la instalación de *Pgcluster* se cuenta con el script **instalar.sh** el cual tiene rasgos similares al elaborado para el *Cybercluster* en lo que respecta a instalación y configuración, por lo que se puede decir que ambas instalación se realizan de igual forma, solo se diferencian en el momento de la configuración por los ficheros que se editan, ya que el servidor de replicación y el balanceador de caga realizan la

configuración de igual modo, diferenciándose en la configuración de los clúster de BD, porque en este caso se incluye la configuración del fichero **postgresql.conf,** a este fichero solo hay que hacerle dos modificaciones, una es habilitar el puerto de escucha 5432 y la otra ponerlo a escuchar desde todos los ip (para una explicación más detallada ver Anexo #1).

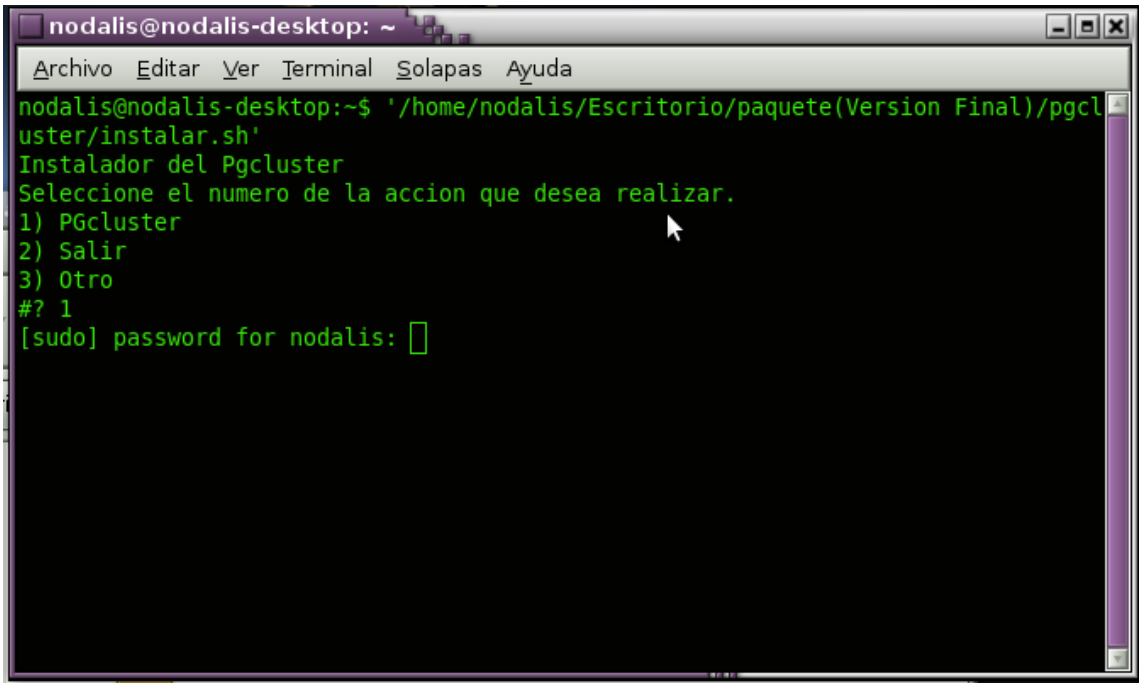


Fig. 3.4 Pantalla principal del Pgcluster.

3.2.1.6 Script de instalación y configuración de Slony

Para la instalación y configuración de la herramienta *Slony* se cuenta con el fichero **instalar.sh** el cual realiza una serie de pasos que permite llevar a cabo este proceso.

Inicialmente se instalan los tres paquetes necesarios para replicar con *Slony,* en caso de que estos estén ya funcionando lo reinstala con la versión más actualizada. Luego se procede a la configuración de tres scripts.

El primer script a ejecutar es **01.nodes_and_path.sh** el cual establece los nodos de replicación, de este solo se le debe cambiar el nombre del clúster si se desea y el conninfo el cual contiene la cadena de conexión a la base de datos (ver Anexo #3).

Luego de haber ejecutado este script quedarían establecidos los nodos de replicación, ahora sería necesario poner a correr el proceso slon.[19]

Con el slon corriendo se procede a configurar los set de replicación para ello se ejecuta el script **04.create_set.sh**, (ver modificaciones en el Anexo #3).

Al crear los set de replicación solo se debe especificar cuál es el clúster que va a recibir los datos, y cuál los va a enviar, para eso se ejecuta el script **05.subscribe_3.sh**.

De esta manera quedaría concluida la configuración de la herramienta.

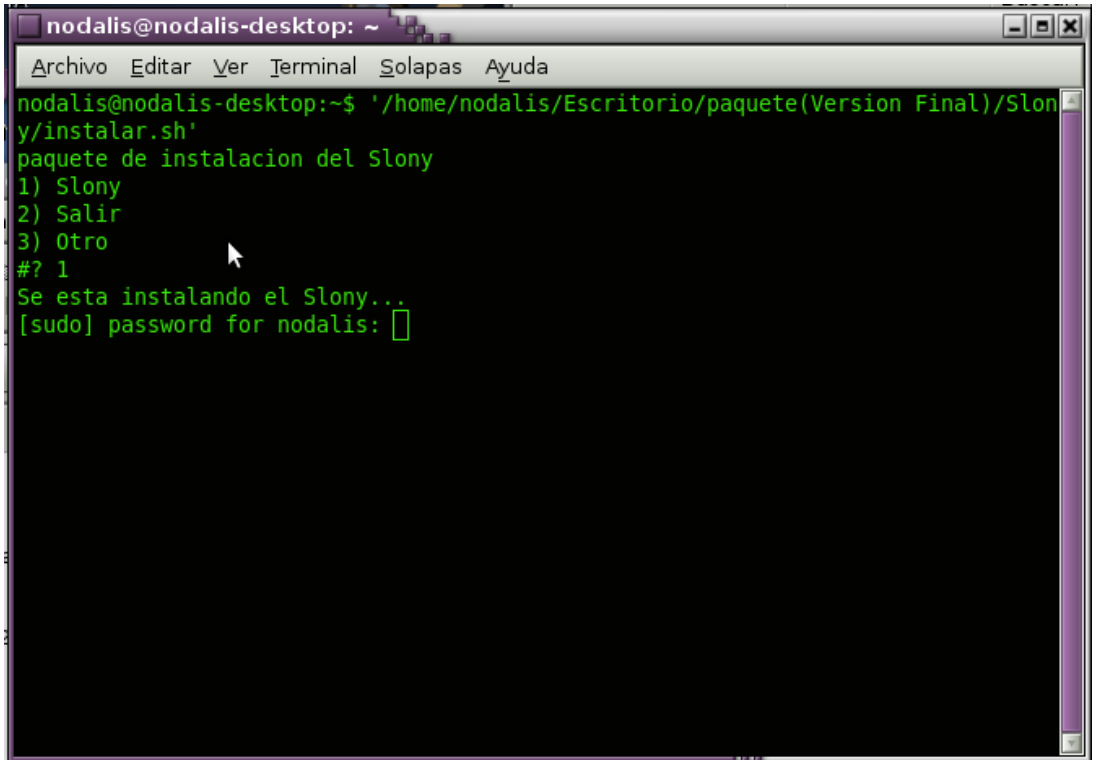


Fig. 3.5 Pantalla principal de Slony.

[19] *Slon: Es el daemon de aplicación que "corre" Slony-I.*

3.2.2 Funcionalidades que brinda al usuario

Este producto le brinda al usuario la solución de réplica más sugerente de acuerdo a las necesidades que este tenga, además le brinda la posibilidad de mejorar el proceso de instalación y configuración de las soluciones de réplica *Cybercluster*, *Pgcluster*, o *Slony*, ya que centraliza ambos procesos, de manera tal que resulten más amigable al usuario, debido a que luego de haber terminado la instalación, se inicia la configuración de los ficheros necesarios automáticamente, mostrándose estos en la pantalla para realizárseles las modificaciones pertinentes.

3.3 Validación de la herramienta

Uno de los últimos períodos del ciclo de vida antes de entregar un software para su explotación es el análisis de los resultados obtenidos con el desarrollo de este trabajo. Para esto se utilizará el método de Estudio de Casos, para analizar el proceso de replicación de datos antes y después de aplicada la solución. Se presentará un análisis comparativo basado en pruebas realizadas al paquete de instalación en el CENTALAD (ver resultados en el Anexo #7), que validan el trabajo realizado y la calidad del mismo.

3.3.1 Método estudio de Casos

Según Chetty, estudioso de temas relacionados con la metodología de la investigación científica indica que el método de estudio de caso es: “una metodología rigurosa que es adecuada para investigar fenómenos en los que se busca dar respuesta a cómo y por qué ocurren, permite estudiar un tema determinado, es ideal para el estudio de temas de investigación en los que las teorías existentes son inadecuadas, permite estudiar los fenómenos desde múltiples perspectivas y no desde la influencia de una sola variable, y además, permite explorar en forma más profunda y obtener un conocimiento más amplio sobre cada fenómeno, lo cual permite la aparición de nuevas señales sobre los temas que emergen”. (Universidad del Norte, 2004)

Pasos de un Estudio de Casos

- Diseño del estudio.
- Realización del estudio.
- Análisis y conclusiones.

Inicialmente se establecen los objetivos del estudio, se elabora el diseño, la estructura de la solución y se determina si el trabajo tiene como objetivo la interpretación de significados, o una guía para la acción. Seguidamente se realiza la recopilación de datos y se agrupa la realidad en todas las fuentes del caso. Por último se analiza la evidencia.

La forma de vincular los datos con las proposiciones es variada y los criterios para interpretar los hallazgos de un estudio no son únicos.

Luego de un análisis de la definición y las principales características del método de Estudio de Casos se puede apreciar que está acorde con la solución y por tanto sirve para validar la misma, pues permite tener una idea completa del objeto de estudio que en este caso es el proceso de replicación de datos.

3.3.1.1 Diseño del estudio

El objetivo principal de aplicar el método de Estudio de Casos, es demostrar la importancia de aplicar el paquete de instalación para resolver la problemática existente en cuanto a la instalación y configuración de las soluciones de réplicas en la UCI.

El estudio se comienza a partir de la instalación y configuración de las herramientas de replicación, describiendo una serie de problemas vinculados a estas, para luego aplicar la solución que se creó y arrojar un resultado que valide que el paquete de instalación es capaz de propiciar una adecuada instalación y configuración.

El principal problema que presentan las herramientas de réplicas es la compleja instalación y configuración, de ahí que se realizara un paquete de instalación que posibilitara agilizar ambos procesos.

Seguidamente se precisan las variables que contribuirán a obtener el resultado que se espera de aplicar el método de Estudio de Casos.

Variable independiente: Paquete de instalación de soluciones de réplicas para servidores *PostgreSQL.*

Variable Dependiente: Amigable instalación y configuración de las soluciones de réplicas para servidores *PostgreSQL.*

Para darle valor a cada variable, se establecen unidades de medidas a las cuales se le asignaron valores numéricos, lo que permitirá una mejor interpretación de la representación gráfica que se realice de las variables.

Unidades de medida:

- Bajo (1)
- Parcialmente bajo (2)
- Medio (3)
- Alto (4)
- Muy alto (5)

Variable Independiente	Indicadores	Sub-Indicadores	Sub-Indicadores	Unidades de medida
Paquete de instalación de soluciones de réplicas para servidores *PostgreSQL.*	Estructura	Instaladores de las herramientas de réplicas	Instalador del *Cybercluster*	Bajo Parcialmente bajo Medio Alto Muy alto
			Instalador del *Pgcluster*	Bajo Parcialmente bajo Medio Alto Muy alto
			Instalador del *Slony*	Bajo Parcialmente bajo Medio Alto Muy alto

		Scripts del *Cybercluster*	Script de instalación del *Cybercluster*	Bajo Parcialmente bajo Medio Alto Muy alto
			Script de desinstalación del *Cybercluster*	Bajo Parcialmente bajo Medio Alto Muy alto
		Scripts del *Pgcluster*	Script de instalación del *Pgcluster*	Bajo Parcialmente bajo Medio Alto Muy alto
			Script de desinstalación del *Pgcluster*	Bajo Parcialmente bajo Medio Alto Muy alto
		Scripts del *Slony*	Script de instalación *Slony*	Bajo Parcialmente bajo Medio Alto Muy alto
			Script de desinstalación	Bajo

			Slony	Parcialmente bajo Medio Alto Muy alto
		Scripts de selección		Bajo Parcialmente bajo Medio Alto Muy alto
		Manual de usuario	Introducción al paquete	Bajo Parcialmente bajo Medio Alto Muy alto
			Opciones generales	Bajo Parcialmente bajo Medio Alto Muy alto
			Acciones a realizar	Bajo Parcialmente bajo Medio Alto Muy alto

Tabla 3.1. Operacionalización de la variable independiente.

Variable dependiente	Indicadores	Sub-Indicadores	Unidades de medida

Amigable instalación y configuración de las soluciones de réplicas para servidores *PostgreSQL.*	Instalación y configuración	Tiempo empleado	Bajo Parcialmente bajo Medio Alto Muy alto
		Esfuerzo realizado	Bajo Parcialmente bajo Medio Alto Muy alto
		Calidad del proceso	Bajo Parcialmente bajo Medio Alto Muy alto
		Repetitivo con igual calidad	Bajo Parcialmente bajo Medio Alto Muy alto
		Independiente del conocimiento de usuario	Bajo Parcialmente bajo Medio Alto Muy alto

Tabla3.2 Operacionalización de la variable dependiente.

Existe una estrecha relación entre la variable independiente y la dependiente. La tabla que se muestra a continuación muestra la relación entre los indicadores de las variables, evidenciándose así que todo el contenido tiene una estrecha relación y que el funcionamiento de los indicadores representa un cambio funcional en cada una de las variables. Reflejándose de este modo el impacto que produce una variable sobre otra.

	Tiempo	Esfuerzo	Calidad	Repetitivo	Independiente
Instaladores de las herramientas de réplicas				x	x
Scripts del *Cybercluster*	x	x	x	x	x
Scripts del *Pgcluster*	x	x	x	x	x
Scripts del *Slony*	x	x	x	x	x
Scripts de selección	x		x		x
Manual de usuario	x	x	x	x	x

Tabla.3.3 Impacto que causa la variable independiente sobre la dependiente.

Con el propósito de saber cómo era la situación del proceso de instalación y configuración de las herramientas de replicación antes y después de aplicada la solución propuesta, a continuación se establece una comparación entre estas dos etapas, a partir del comportamiento de los indicadores definidos anteriormente.

Una vez definido el diseño del caso de estudio a realizar, se procede a la aplicación o realización del mismo, con el objetivo de recolectar los datos necesarios para su posterior análisis.

3.3.1.2 Realización del estudio

Para llevar a cabo la realización del estudio se tuvieron en cuenta los sub indicadores de las variables dependientes definidas con anterioridad, las evidencias obtenidas se describen a continuación, teniendo en cuenta en primer lugar la situación existente antes de aplicar la solución y luego de aplicada la misma.

Estudio realizado antes de aplicar la solución

Mientras se realizaba el proceso de instalación y configuración de las herramientas de replicación (*Cybercluster*, *Pgcluster*, *Slony*) de forma separada y manual, se tornaba engorroso y poco amigable para el usuario que muchas veces no contaba con el conocimiento necesario de cómo emplear dichas herramientas.

Instalación y configuración

Según se ha ido utilizando las soluciones de replicación en los proyectos de gestión de la UCI se han detectado inconvenientes en cuanto a la instalación y configuración de las mismas, debido a que no existe una herramienta que permita realizar estos procesos de forma conjunta. Otro inconveniente es la falta de una herramienta que agrupara varias soluciones de réplica además de la documentación necesaria para su uso.

Estudio realizado después de aplicar la solución

Con la aplicación de la herramienta creada, se puede apreciar una mejora en cuanto a la selección de la solución de réplica a utilizar según las necesidades del usuario, debido a que esta agrupa varias soluciones de réplica con su respectiva documentación, ofrece una sugerencia de la solución de réplica más ventajosa a utilizar, además de que une los procesos de instalación y configuración.

Instalación y configuración

Las evidencias obtenidas en cuanto a este indicador demuestran como el hecho de contar con una herramienta que agrupa varia soluciones de réplica, y que además une el proceso de instalación y configuración de las mismas, permite que la selección de la solución a utilizar sea la más acertada, así como que el proceso de instalación y configuración se agrupe en un único proceso, siendo de esta manera más fáciles y amigables para el usuario.

Análisis de los datos

En este espacio se pretende demostrar la importancia y beneficios que posibilita la utilización del paquete de instalación, para el proceso de replicación de datos en servidores *PostgreSQL* en la universidad, así como la contribución de esta solución al éxito del proceso.

Para el análisis de los datos obtenidos en la sección anterior se utilizaron gráficas que permitieran mostrar los resultados alcanzados antes y después de aplicada la solución.

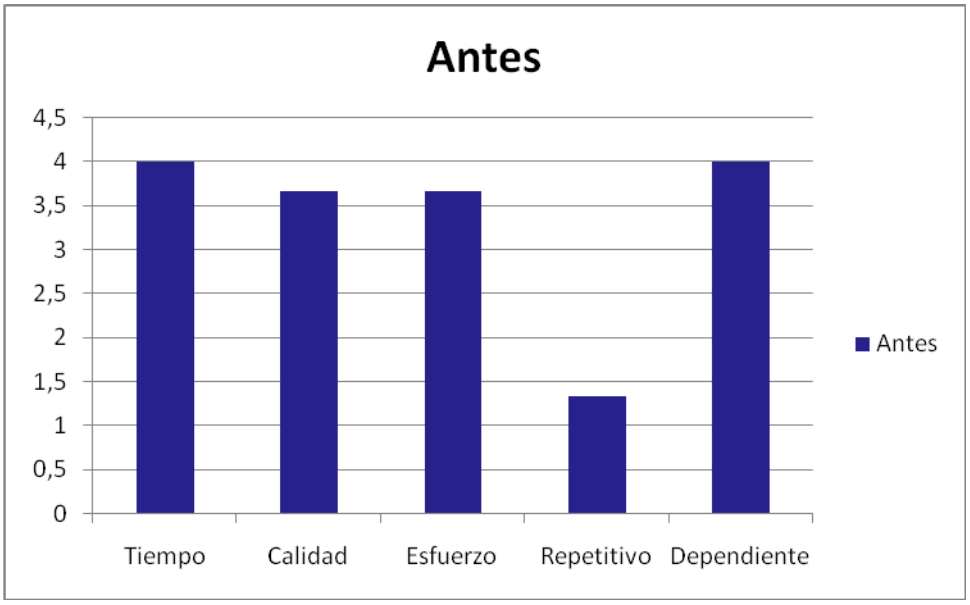


Fig. 3.6 Instalación y configuración antes de utilizar el paquete de instalación.

Lo mostrado en la figura anterior refleja como la inexistencia de un paquete de instalación de soluciones de réplicas, propicia un alto empleo de tiempo, de esfuerzo, una alta dependencia del conocimiento que tenga el usuario, además de una baja eficiencia en el proceso repetitivo de instalación y configuración, por tanto dichos procesos son engorrosos para el usuario.

A continuación se reflejan los resultados del estudio realizado a este indicador una vez aplicada la solución siendo totalmente diferentes, debido a los elementos que fueron definidos en la solución cómo, instaladores de las herramientas, scripts de instalación y configuración, script de selección, manual de

usuario, aspectos que posibilitaron un cambio sustancial en los sub indicadores mencionados con anterioridad siendo muy alto el tiempo y esfuerzo empleado.

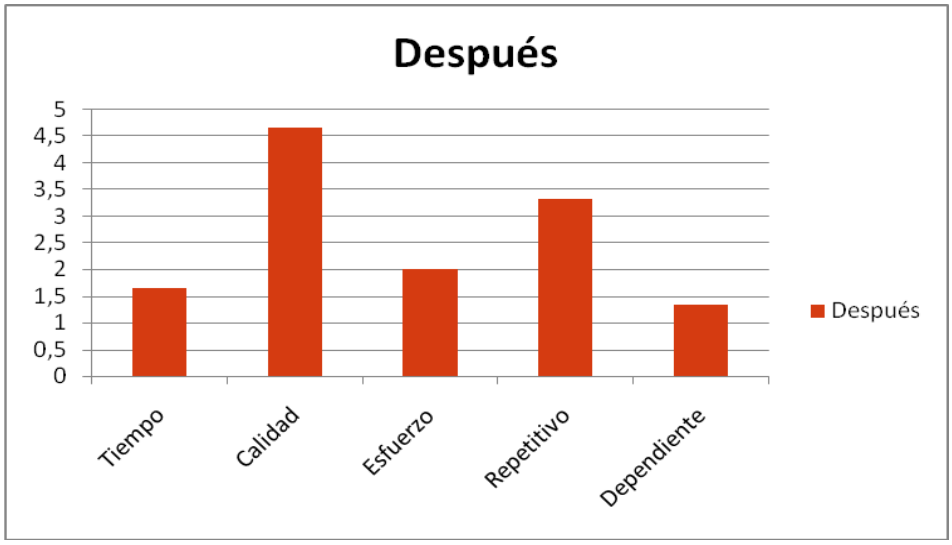


Fig.3.7 Control de la información después de la aplicación del mecanismo.

Lo expuesto con anterioridad, evidencia el impacto positivo que causa la solución en el proceso de instalación y configuración de las herramientas de réplicas, evidenciándose esto en las gráficas representadas, además se demuestra cómo el paquete de instalación contribuye al logro de una buena instalación y configuración de soluciones de réplicas en servidores *PostgreSQL.*

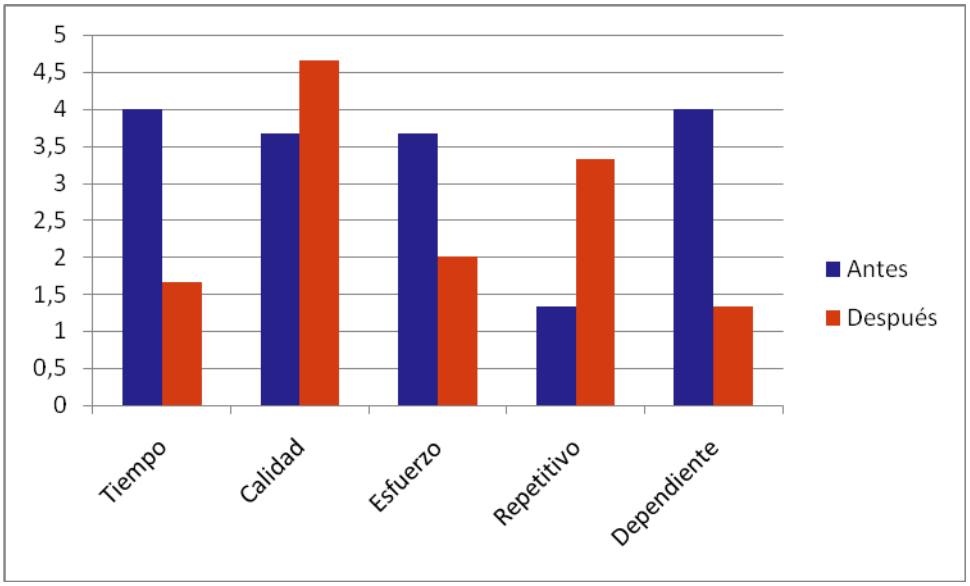


Fig. 3.8 Comparación del control de la información antes y después de la aplicación del mecanismo.

El estudio de los sub indicadores mencionados, permite llegar a la conclusión de que la aplicación del paquete de instalación facilita el proceso de instalación y configuración, mostrándose una disminución de tiempo y esfuerzo empleado por el usuario y una mayor calidad.

3.3.1.3 Cumplimiento de los Requisitos de la solución

Para el desarrollo de la solución propuesta, se definieron necesidades fundamentales con los cuales debía cumplir el paquete de instalación, a continuación se verifica su cumplimiento, teniendo en cuenta los resultados obtenidos una vez validada la solución.

Facilitar la instalación y configuración de las soluciones de réplica de datos, constituye la principal necesidad con lo cual debía cumplir la solución elaborada, el análisis elaborado en la sección anterior permitió demostrar el cumplimiento del mismo con la disminución de sub-indicadores como tiempo, esfuerzo, la dependencia del conocimiento, además de notarse un aumento en la calidad del proceso de instalación y configuración.

Lo expuesto con anterioridad se muestra a través de la siguiente grafica:

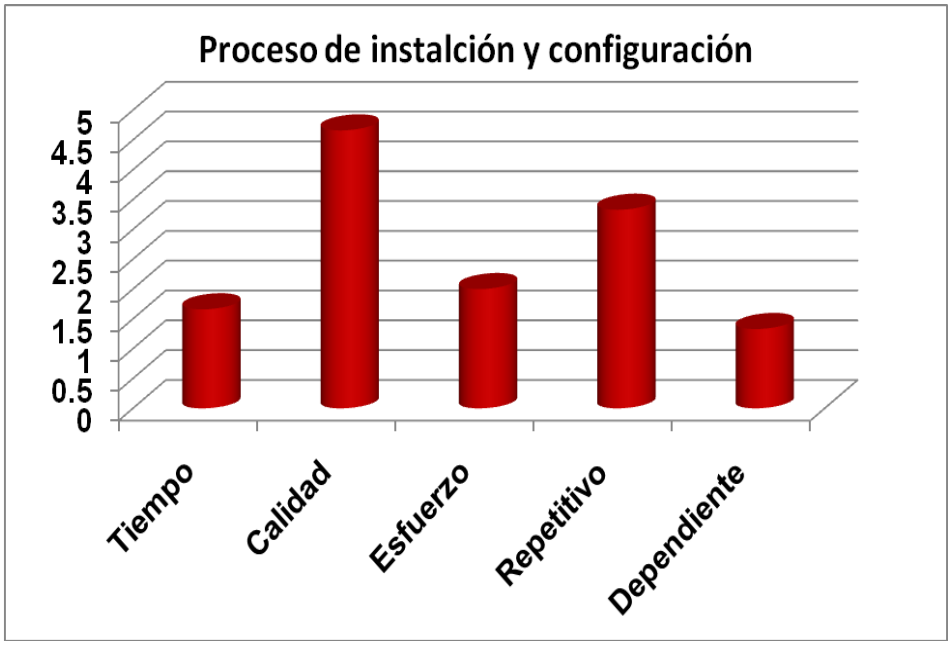


Fig. 3.9 Resultados obtenidos en cuanto a la instalación y configuración de las soluciones de réplica.

3.3.1.4 Cumplimiento del método

Para llevar a cabo el método de validación de Estudio de Casos, se realizaron pruebas experimentales(ver Anexo #7) que propiciaron una estadística del comportamiento de los indicadores definidos antes y después de emplear el paquete de instalación elaborado, par esto se repitió el proceso de instalación y configuración de las herramientas empaquetadas haciendo uso del paquete creado, este proceso se realizó en varias computadoras, tomando muestra del tiempo empleado, el esfuerzo realizado, la calidad en cada repetición entre otros, haciendo posible mostrar los datos obtenidos mediante las graficas anteriores.

Conclusiones del capítulo

De este modo, en el presente capítulo ha quedado especificada la estructura de la herramienta elaborada, así como la forma en que puede intercambiar la información con los usuarios, brindando una amplia explicación de la utilización del paquete de instalación con sus respectivas características. Por último se detalla todo el proceso de validación de la herramienta construida, con la utilización del método de prueba Estudio de Casos, con el que se emitió la calidad del producto confeccionado.

CONCLUSIONES GENERALES

Para el desarrollo del presente trabajo de diploma se realizó un estudio detallado sobre el proceso de replicación de datos, que aportó una serie de definiciones y conceptos importantes a tener en cuenta a la hora de llevar a cabo este proceso tan utilizado en las aplicaciones de gestión. Se realizó además una búsqueda exhaustiva de las soluciones de réplicas existentes a nivel mundial para fundamentar la elección de las herramientas que quedarían concentradas en el paquete de instalación. Resultaron definidos los indicadores a tener en cuenta para desarrollar el proceso de replicación de datos entre los que se encuentran el tipo de réplica, el modelo, el entorno de replicación, por solo mencionar algunos. Luego se confeccionó un paquete de instalación que concentra las soluciones de réplicas *Cybercluster*, *Pgcluster* y *Slony*. Por último se elaboró un manual donde permanece recogida toda la información necesaria para la utilización del paquete de instalación. Por tanto, se consideran cumplidas las tareas trazadas en el inicio de este trabajo, donde se materializó la hipótesis planteada demostrándose que con la elaboración de un paquete de instalación de réplicas, se facilita y automatiza el proceso de instalación y configuración de las mismas.

RECOMENDACIONES

Este trabajo proporciona un paquete de instalación de réplicas para servidores que utilizan el gestor de BD *PostgreSQL* sobre la plataforma GNU/Linux, por lo que se recomienda la implementación de una versión para servidores cuyo sistema operativo sea *Windows,* siempre manteniendo el SGBD *PostgreSQL.*

Se recomienda además enriquecer la herramienta elaborada con nuevas funcionalidades, como por ejemplo la integración de otras soluciones de réplicas utilizadas a nivel mundial.

Además se considera que debido a que la herramienta elaborada se ejecuta bajo la utilización de comandos *Bash*, sería interesante elaborar una versión de este producto con interfaz gráfica para todos aquellos usuarios que prefieran este método.

Por otra parte, se considera que una aplicación de este tipo podría ser muy útil para cualquier institución interesada en respaldar sus datos o satisfacer otra necesidad a través de la réplica, por lo que se recomienda sea generalizada.

Bibliografía

Amazon.com Company. ALEXA. [En línea] The Web Infromation Company. [Citado el: 2 de marzo de 2009.] http://www.alexa.com/data/details/main/skytool.com.

Biazus, Diogo. 2008. http://www.arpug.com.ar. [En línea] 2008. [Citado el: 25 de febrero de 2009.] http://www.arpug.com.ar/pgday2008/slides/postgresql-ha-es.pdf.

Bucardo. Bucardo. [En línea] http://bucardo.org/bucardo.html.

2001. College of Engineering University of Hawai. [En línea] University of Hawai, 2001. http://www.eng.hawaii.edu/Tutor/csh.html.

Company, PostgreSQL Database. 2009. Cybertec. [En línea] 2009. http://www.postgresql.at/spanish/support_cybercluster_s.html.

Corgatelli, Ariel. 2004. [En línea] novienmbre de 2004. http://www.infosertec.com.ar.

Corporation, Microsoft. 2009. Microsoft TechNet. [En línea] 2009. https://technet.microsoft.com/es-es/library/ms152567.aspx.

Cycma S.L. CYCMA. [En línea] [Citado el: 2 de Marzo de 2009.] http://www.cycma.es/replicacion.asp.

Free Software Fundation. 1996-2009. The GNU Operating System. [En línea] 1996-2009. [Citado el: 2 de marzo de 2009.] http://www.gnu.org/software/bash/manual/bashref.html#What-is-Bash_003f.

Freedman, A. 1993. www.agapea.com. [En línea] 1993. [Citado el: 10 de febrero de 2009.] http://www.agapea.com/Diccionario-bilingue-de-Computacion-n9673i.htm.

2004. GMANE. [En línea] Abril de 2004. http://news.gmane.org/group/gmane.comp.db.postgresql.pgcluster/last=/force_load=t.

Integrated SCM & Project Management. www.nsis.com.ar. [En línea] [Citado el: 28 de febrereo de 2009.] http://www.nsis.com.ar/public/browser/pyreplica/LEEME.txt.

Kioskea. 2009. Kioskea.net. [En línea] 2009. [Citado el: 20 de enero de 2009.] http://es.kioskea.net/contents/surete-fonctionnement/replication-bases-donnees.php3.

Landrian García, MsC.Jorge. *Réplica bidireccional basada en control de cambios.* Ciudad de La Habana : s.n.

MARLLYVI. 2008. MARLLYVI. [En línea] 2008. http://marllyvivi.blogspot.com/.

Microsoft Corporation. 2008. Microsoft Developer Network. [En línea] agosto de 2008. [Citado el: 23 de enero de 2009.] http://msdn.microsoft.com/es-es/library/bb500351.aspx.

O´REILLY. 2009. www.onlamp.com. [En línea] 2009. [Citado el: 27 de Febrero de 2009.] http://www.onlamp.com/pub/a/onlamp/2004/11/18/slony.html.

O'REILLY. 2009. Mustain,E "Introducing Slony". [En línea] 2009. http://www.onlamp.com/pub/a/onlamp/2004/11/18/slony.html .

Pgcluster. 2005. Sitio oficial PgCluster. “PGCluster is the synchronous replication system of the multi-master composition for PostgreSQL. [En línea] abril de 2005. http://pgcluster.projects.postgresql.org.

Piñeiro, DrC. Pedro Yobanis, y otros. 2006. *Réplica entre servidores Oracle, alternativas de solución.* Ciudad de la Habana : s.n., 2006.

PostgreSQL.org. 2009. PostgreSQL. [En línea] 2009. http://postgresql.org.pe/articles/pgsql_alta_disponibilidad.pdf.

—. 2009. www.postgresql.org. [En línea] 2009. [Citado el: 15 de febrero de 2009.] http://www.postgresql.org/docs/8.2/interactive/intro-whatis.html.

Potgres.org. 2006. www.progress.com. [En línea] O´REILLY, 2006. [Citado el: 25 de febrero de 2009.] http://www.progress.com/realtime/techsupport/documentation.

Reingart, Mariano. www.arpug.com.ar. [En línea] [Citado el: 28 de febrero de 2009.] http://www.arpug.com.ar/pgday2008/slides/pyreplica.pdf.

Slony Develoment Group. 2007. Slony-| (Enterpise-Level Replication System). [En línea] 2007. http://www.slony.info/documentation/.

Stallman, Richard. 2004. Stallman, Richard "Software libre para una sociedad libre". [En línea] diciembre de 2004. http://biblioweb.sindominio.net/pensamiento/softlibre/.

Steffen, Dr.Ing. Hermann . 2003. http://www.fing.edu.uy. [En línea] 2003. [Citado el: 10 de febrero de 2009.] http://www.fing.edu.uy/inco/cursos/tagsi/Trabajos/2003/TAGSI03-TareaTecnSI-grupo4.pdf.

TechNet, Microsoft. 2009. Microsoft TechNet. [En línea] 2009. https://technet.microsoft.com/es-es/library/ms152531.aspx.

TechTarget. 2009. Search EnterpriseLinux.com. [En línea] 2009. http://searchenterpriselinux.techtarget.com/sDefinition/0,,sid39_gci214635,00.html.

Universidad del Norte. 2004. Sistema de Informacion Bibliográfica. [En línea] 2004. http://ciruelo.uninorte.edu.co/pdf/pensamiento_gestion/20/5_El_metodo_de_estudio_de_caso.pdf .

UseModwiki. 2001. Welcome to the tcsh site. [En línea] julio de 2001. http://www.tcsh.org/Welcome.

GLOSARIO DE TERMINOS

Archivo: Lugar donde se guardan documentos. Conjunto de documentos.

Commit: Una sentencia COMMIT en SQL finaliza una transacción de base de datos dentro de un sistema gestor de base de datos relacional (RDBMS) y pone visibles todos los cambios a otros usuarios.

Constraints: Son las encargadas de asegurar la integridad referencial en la base de datos.

DDL: comandos de lenguaje de definición de datos (DDL) la ejecución de estos comandos son usados para la eliminación de una columna de un artículo de una tabla replicada. Se replica automáticamente estos comandos DDL siempre que se haya habilitado la réplica DDL.

Daemon: Es un tipo especial de proceso informático que se ejecuta en segundo plano en vez de ser controlado directamente por el usuario (es un proceso no interactivo). Este tipo de programas se ejecutan de forma continua (infinita), vale decir, que aunque se intente cerrar o matar el proceso, éste continuará en ejecución o se reiniciará automáticamente. Todo esto sin intervención de terceros y sin dependencia de consola alguna.

Fichero: Agrupación de información que puede ser manipulada de forma unitaria por el sistema operativo de un ordenador. Un fichero puede tener cualquier tipo de contenido (texto, ejecutables, gráficos, etc.)

Licencia BSD: Otorgada principalmente para los sistemas BSD (Berkeley Software Distribution, Distribución de Software Berkeley, que identifica un sistema operativo derivado del Unix y nacido a partir de las aportaciones realizadas a ese sistema por la Universidad de California en Berkeley). Esta licencia pertenece al grupo de licencias de software libre y tiene menos restricciones en comparación con otras, estando muy cercana al dominio público.

Middleware es un software de conectividad que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas.

Primary Key: En español llave primaria Una llave primaria es un campo único, irrepetible que sirve para identificar los registros de una tabla como distintos, además de que sirve para hacer relaciones

Python: es un lenguaje de programación interpretado creado por Guido Van Rossum en el año 1991. Python permite dividir el programa en módulos reutilizables desde otros programas Python. Viene con

una gran colección de módulos estándar que se pueden utilizar como base de los programas (o como ejemplos para empezar a aprender Python). Python como lenguaje de programación interpretado ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar.

Réplica de espejos: Réplica exactamente igual al origen de los datos.

Setear: Significa establecer la configuración correcta de un programa o hardware.

Shell: interfaz usada para interactuar con el núcleo de un sistema operativo. Cualquier programa que los usuarios usen para tripear comandos. Dado que en los sistemas operativos Unix los usuarios pueden seleccionar cuál shell quieren usar (programa que se debería ejecutar cuando se loguean.

Slon: Es el daemon de aplicación que "corre" Slony-I replicación. Slon debe ejecutarse para cada nodo en un clúster Slony-I.

Supresión: Mecanismo de defensa en que el individuo se enfrenta a conflictos emocionales y amenazas de origen interno o externo evitando intencionadamente pensar en problemas, deseos, sentimientos o experiencias que le producen malestar.

Topología de replicación es la ruta por la que la replicación de datos viaja a través de una red. Se produce la replicación entre dos controladores de dominio a la vez.

Triggers o disparador: Se defina así a una subrutina que es ejecutada de manera automática cuando se produce algún tipo de transacción (inserción, borrado o actualización) en la tabla de una base de datos.

Triggers (o disparador) en una Base de datos, es un procedimiento que se ejecuta cuando se cumple una condición establecida al realizar una operación de inserción (INSERT), actualización (UPDATE) o borrado (DELETE). Son objetos relacionados con tablas y almacenados en la base de datos que se ejecutan o se muestran cuando sucede algún evento sobre sus tablas asociadas.

Time Zone: Zona Horaria en el caso de P*ostgreSQL* la zona horaria reconocida por él.

ANEXOS

Anexo # 1: Guía de instalación de Cybercluster*.*

Para realizar la instalación del *Cybercluster* se necesita tener tantas maquinas (reales o virtuales) para cada *clúster*, una para el servidor de réplica y otra para el balance da carga en caso de que se vaya a utilizar.

Composición: 5 servers (4 en el ejemplo)

1. Balanceador
2. Clúster de BD (3), el ejemplo se va mostrar para 2 clústeres.
3. Replicador (este es el elemento fundamental del C*ybercluster*).

Antes de empezar lo primero es copiar la herramienta C*ybercluster* 1.2.1.tar.bz2 a home/install en todos los servidores.

Seguidamente se realizan los pasos siguientes:

1. Instalar los descompactadores (porque el instalador está compactado) el proceso se realiza de siguiente forma.

```
#apt-get install tar
```

```
#apt-get install bzip2
```

2. Descompactar el *Cybercluster*

```
#cd /home/install
#tar -xjvf Cybercluster-1.2.0.tar.bz2
```

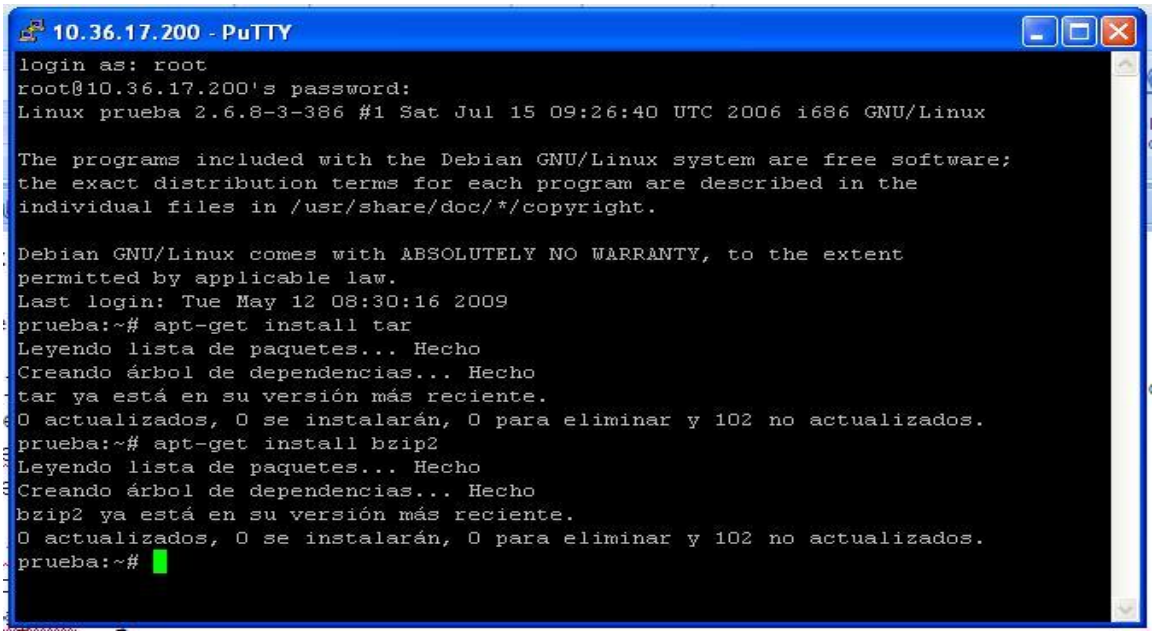
3. Instalar las dependencias necesarias

```
# apt-get install
# apt-get install g++
# apt-get install zlib1g-dev
# apt-get install libreadline-dev
# apt-get install libreadline-dbg
```

apt-get install make

apt-get install flex

apt-get install bison



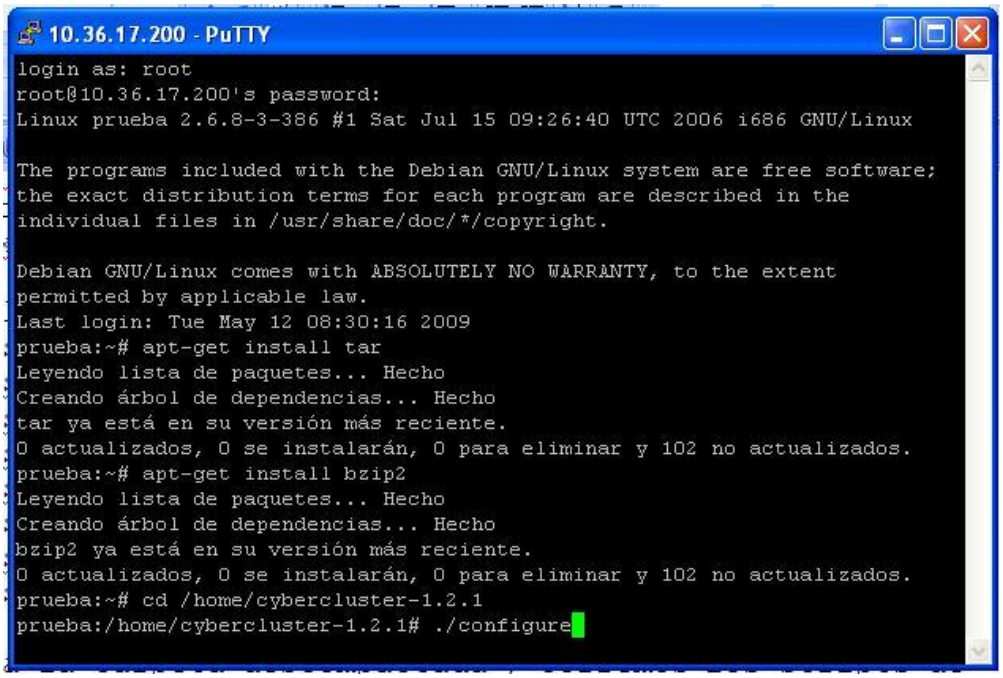
4. Entra a la carpeta descompactada y se corren los scripts de configuración e instalación

cd /home/install/cybercluster-1.2.1

#./configure

#make

#make install



5. Se crea el usuario *postgres*

#adduser Postgres

Siendo necesario la definición de una contraseña.

6. Se crea la carpeta data y se pone a *postgres* como dueño de toda esa herramienta

#mkdir /usr/local/pgsql/data

#chown -R postgres /usr/local/pgsql

7. Luego se loguea como postgres

su - postgres

8. Se crea el home del Cybercluster en /usr/local/pgsql/data

$ /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data

Hasta aquí tienes instalada la herramienta que te instala el postgresql8.3; ahora todo se centra en la carpeta esa que se crea para realizar la configuración.

/usr/local/pgsql

* En /data: se encuentran los archivos de configuración del *postgres* y de los *clúster*

* En /bin: los scripts necesarios para levantar los *clústeres*, balanceador y replicador

Además se tiene que configurar 4 servers diferentes
*clúster (2)
*Balanceador (1)
*replicador (1)
Para la configuración de un clúster:
9. Ir a la carpeta /usr/local/pgsql/data
Ahí hay 4 archivos

9.1 postgres.conf en este fichero lo único que hay que hacerle es quitar el comentario al puerto de escucha 3452 y ponerlo a escuchar desde todos los ip de la siguiente forma:

Listen_addresses = "*"
port = 5432

9.2 cluster.conf se define el ip del clúster, el ip del replicador y balanceador.

Replicador:
<Replicate_server_Info> Ip_del_replicador </Replicate_server_Info>

Clúster:
<Host_Name> Ip_del_cluster </Host_Name>
<when_stand_alone>read_write </when_stand_alone>

El caso del **cluster.conf** dejas el ip del replicador sin ponerle nada todavía

9.3 pg_hba.conf se le especifica desde que subred se pueden conectar.

#ipv4 local conections:

host all all 127.0.0.1/32 trust

host all all 10.0.0.0/8 (Desde toda la uci) trust

Solo es necesario que cambiar esas.

10. lo que queda es inicializar el clúster

$/usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data/ start

Hasta aquí están configurados los 2 clústeres de BD. Quedaría el balanceador y replicador.

Configuración del Balanceador:

Serian los mismos pasos anteriores, menos el 9.2 ya que ahora se va a configurar el balanceador no un clúster.

Ir a la carpeta share donde están las configuraciones de ejemplo y se copia el fichero **pglb.conf.sample** a **pglb.conf**

#cd /usr/local/pgsql/share

#cp pglb.conf.sample pglb.conf

Lo que quedaría es configurar el fichero **pglb.conf** e iniciar el balanceador.

Este fichero va a tener los IP de los clúster y el del mismo como balanceador.

Solo se tendría que cambiar la información siguiente en el fichero.

<Cluster_Server_Info>

<Host_Name> 10.7.12.5 (Cluster1) </Host_Name>

<Port> 5432 </Port>

<Max_Connect> 32 </Max_Connect>

</Cluster_Server_Info>

<Cluster_Server_Info>

<Host_Name> 10.7.12.6 (Cluster2) </Host_Name>

<Port> 5432 </Port>

<Max_Connect> 32 </Max_Connect>

</Cluster_Server_Info>

Y los datos del balanceador:

<Host_Name> 10.7.12.8 </Host_Name>
<Backend_Socket_Dir> /tmp </Backend_Socket_Dir>
<Receive_Port> 5432 </Receive_Port>
<Recovery_Port> 6001 </Recovery_Port>
<Max_Cluster_Num> 128 </Max_Cluster_Num>
<Use_Connection_Pooling> no </Use_Connection_Pooling>
<LifeCheck_Timeout> 3s </LifeCheck_Timeout>
<LifeCheck_Interval> 15s </LifeCheck_Interval>
<Connection_Life_Time> 0s </Connection_Life_Time>

PARA ARRANCAR, PARAR EL BALANCEADOR

A. Start cluster DB server

$ /usr/local/pgsql/bin/pg_ctl start -D /usr/local/pgsql/data

B. Stop cluster DB server

$ /usr/local/pgsql/bin/pg_ctl stop -D /usr/local/pgsql/data

Configuración del Replicador:

Serian los mismos pasos anteriores para configurar el clúster, menos el 9.2 ya que ahora se va a configurar el replicador.

Ir a la carpeta share donde están las configuraciones de ejemplo y se copia el fichero **pgreplicate.conf.sample** a **pgreplicate.conf**

#cd /usr/local/pgsql/share

#cp pgreplicate.conf.sample pgreplicate.conf

Editar el fichero **pgreplicate.conf**:

Poniendo la información de los ip de los clústeres y el ip de el mismo al igual que el balanceador.

Y se levanta el replicador:

A. Start replication server

$ /usr/local/pgsql/bin/pgreplicate -D /usr/local/pgsql/share

B. Stop replication server

$ /usr/local/pgsql/bin/pgreplicate -D /usr/local/pgsql/share stop

---

Ahora solo quedaría probar si funciona el sistema, se va a hacer la prueba y crear un BD nueva en un clúster, esta tiene que crearse automáticamente en el otro clúster.

Para luego conectarse desde el pgadmin u otro cliente que se tenga.

Anexo # 2: Guía de instalación de SLONY.

-Primeramente para realizar la replicación se debe tener instalado los siguientes paquetes:

*postgresql-8.1

*postgresql-8.1-slony1

*slony-1-bin

Teniendo esto instalado se puede realizar configuración de la base de datos para replicar. Para ello se va a ejecuta 3 scripts que se encuentran junto con esta documentación. Seguidamente se muestran los pasos a seguir:

1. El primer script a ejecutar es **01.nodes_and_path.sh** el cual establece los nodos de replicación, de este solo se debe cambiar el nombre del clúster si lo se desea y el conninfo el cual contiene la cadena de conexión a nuestra base de datos, ver ejemplo:

```
#!/bin/sh

slonik <<_EOF_

      cluster name = conector;
     node 1 admin conninfo = 'dbname=convenio user=postgres host=10.36.17.1
password=postgres';
     node 2 admin conninfo = 'dbname=convenio user=postgres host=10.36.17.1
password=postgres';

     init cluster (id=1, comment='Nodo 1');
     echo 'Nodo 1 creado';

     store node (id=2, comment='Nodo 2');
     echo 'Nodo 2 creado';

     store path (server=1, client=2, conninfo='dbname=convenio user=postgres
host=10.7.12.1 password=postgres');
     echo 'Almacenadas las rutas para el nodo 1';

     store path (server=2, client=1, conninfo='dbname=convenio user=postgres
host=10.7.13.1 password=postgres');
     echo 'Almacenadas las rutas para el nodo 2';

_EOF_
```

2. Luego de ejecutar este script y establecidos los nodos de replicación, ahora se pone a correr el slon, pero para ello se debe entrar como postgres de la siguiente forma:

su postgres

-Estando como postgres se ejecuta el siguiente comando:

$ slon clustername 'dbname=nombre'

***clustername:** Es el nombre del clúster creado en el script anterior

***nombre:** Nombre de la base de datos.

3. Con esto ya está corriendo el Slon solo se debe configurar los set de replicación para ello se ejecuta el **script 04.create_set.sh**, el cual crea nuestros sets de replicación.

```
#!/bin/sh
slonik <<_EOF_
    cluster name = conector;
     node 1 admin conninfo = 'dbname=nombrebd user=usuariobd host=10.7.12.1 password=contraseña';
     node 2 admin conninfo = 'dbname=nombrebd user=usuariobd host=10.7.13.1 password=contraseña';

     create set (id=1, origin=1, comment='tablas del conector');
     set add table (set id=1, origin=1,id=1, full qualified name='public.nombretabla');
     echo 'Se ha creado el set 1 con la tabla de convenio';
_EOF_
```

***nombretabla:** Es nombre de la tabla a replicar.

4. Al crear los set de replicación solo se debe decir quién va a recibir los datos, y quién los va a enviar, para ello se ejecuta el script **05.subscribe_3.sh**, el cual ya concluiría nuestra configuración.

```
#!/bin/sh
slonik <<_EOF_
    cluster name = conector;
     node 1 admin conninfo = 'dbname=nombrebd user=usuariobd host=10.7.12.1 password=contraseña';
     node 2 admin conninfo = 'dbname=nombrebd user=usuariobd host=10.7.13.1 password=contraseña';
     subscribe set (id=1, provider=1, receiver=2, forward=yes);
```

```
     echo 'El nodo 2 está suscrito al set 1, proveedor es nodo 1';
_EOF_
```

5. Al terminar de ejecutar este script inmediatamente su base de datos comienza a replicarse.

Anexo # 3: Guía de instalación de Pgcluster.

Para instalar dicha herramienta se necesita contar con tantas maquinas (reales o virtuales) una para cada *clúster*, una para el servidor de réplica y otra para el balance da carga en caso de que se vaya a utilizar.

Composición: 4 servers

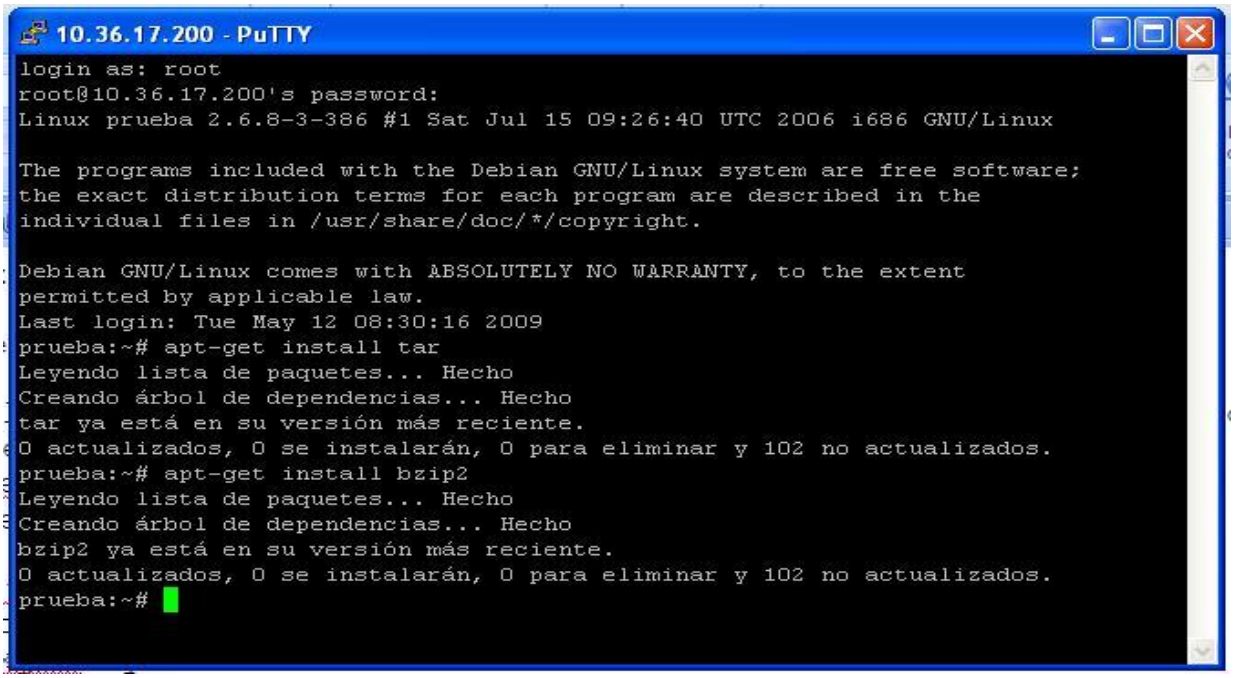
1. Balanceador
2. Clúster de BD (2)
3. Replicador.

Antes de empezar lo primero es copiar la herramienta *pgcluster-1.7.0rc12.tar* a home/install en todos los servidores.

Seguidamente se realizan los pasos siguientes:

1. Instalar los descompactadores (porque el instalador está compactado) el proceso se realiza de siguiente forma.

```
#apt-get install tar
#apt-get install bzip2
```



2. Descompactar el *pgcluster-1.7.0rc12.tar*

#cd /home/install

#tar -xjvf pgcluster-1.7.0rc12.tar

3. Instalar las dependencias necesarias

apt-get install

apt-get install g++

apt-get install zlib1g-dev

apt-get install libreadline-dev

apt-get install libreadline-dbg

apt-get install make

apt-get install flex

apt-get install bison

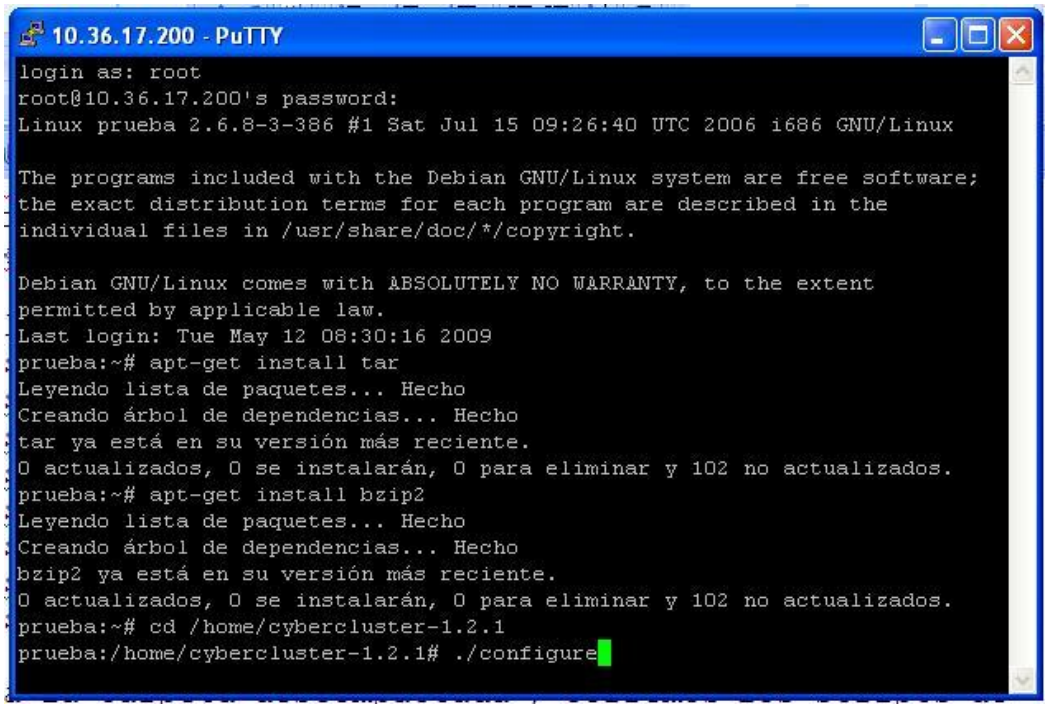
4. Se entra a la carpeta descompactada y se corren los scripts de configuración e instalación

cd /home/install/ pgcluster-1.7.0rc12

#./configure

#make

#make install



5. Se crea el usuario *postgres*

#adduser postgres

6. Se crea la carpeta data y se pone a *postgres* como dueño de toda esa herramienta

#mkdir /usr/local/pgsql/data

#chown -R postgres /usr/local/pgsql

7. Luego se loguea como postgres

su - postgres

8. Se crea el home del Cybercluster en /usr/local/pgsql/data

$ /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data

Hasta aquí está instalada la herramienta con el postgre8.3. Para la configuración se realiza en dependencia que se vaya a configurar.

Configuración del balance de carga:

Quedaría el balanceador y replicador.

Configuración del Balanceador:

Serian los mismos pasos anteriores, menos el 9.2 ya que ahora se configura el balanceador no un clúster.

Ir a la carpeta share donde están las configuraciones de ejemplo y se copia el fichero **pglb.conf.sample** a **pglb.conf**.

#cd /usr/local/pgsql/share

#cp pglb.conf.sample pglb.conf

Lo que quedaría por configurar seria el fichero **pglb.conf** e iniciar el balanceador.

Este fichero va a tener los IP de los clúster y el del mismo como balanceador.

Solo se tendría que cambiar la información siguiente en el fichero.

```
<Cluster_Server_Info>
   <Host_Name>          10.7.12.5 (Cluster1)     </Host_Name>
   <Port>                  5432                  </Port>
   <Max_Connect>           32                    </Max_Connect>


</Cluster_Server_Info>
<Cluster_Server_Info>
   <Host_Name>          10.7.12.6 (Cluster2)        </Host_Name>
   <Port>               5432                        </Port>
   <Max_Connect>        32                          </Max_Connect>
</Cluster_Server_Info>
```

Y los datos del balanceador:

```
<Host_Name>                10.7.12.8        </Host_Name>
<Backend_Socket_Dir>       /tmp             </Backend_Socket_Dir>
<Receive_Port>             5432             </Receive_Port>
```

<Recovery_Port> 6001 </Recovery_Port>
<Max_Cluster_Num> 128 </Max_Cluster_Num>
<Use_Connection_Pooling> no </Use_Connection_Pooling>
<LifeCheck_Timeout> 3s </LifeCheck_Timeout>
<LifeCheck_Interval> 15s </LifeCheck_Interval>
<Connection_Life_Time> 0s </Connection_Life_Time>

<Host_Name> lb.pgcluster.org </Host_Name>
<Backend_Socket_Dir> /tmp </Backend_Socket_Dir>
<Receive_Port> 5432 </Receive_Port>
<Recovery_Port> 6001 </Recovery_Port>
<Max_Cluster_Num> 128 </Max_Cluster_Num>
<Use_Connection_Pooling> no </Use_Connection_Pooling>
<LifeCheck_Timeout> 3s </LifeCheck_Timeout>
<LifeCheck_Interval> 15s </LifeCheck_Interval>

PARA ARRANCAR, PARAR EL BALANCEADOR

A. Start clúster DB server

$ /usr/local/pgsql/bin/pg_ctl start -D /usr/local/pgsql/data

B. Stop cluster DB server

$ /usr/local/pgsql/bin/pg_ctl stop -D /usr/local/pgsql/data

Para configurar un clúster

Se necesita editar solamente los ficheros **pg_hba.conf'** y **'cluster.conf.**

En **pg_hba.conf**

Solo hay que dar los permisos para los ip que deben acceder a la BD.

En **'cluster.conf** se configura de la siguiente forma:

```
set cluster DB server information
#               o Host_Name :               nombre del cluster
#               o Port :                    Connection port for postmaster
#               o Recovery_Port :           Connection for recovery process
#-------------------------------------------------------------
<Replicate_Server_Info>
        <Host_Name>             pgr.pgcluster.org       </Host_Name>
        <Port>                  8001                    </Port>
        <Recovery_Port>         8101                    </Recovery_Port>
</Replicate_Server_Info>
#-------------------------------------------------------------
# set Cluster DB Server information
#               o Host_Name :       Host name which connect with replication server
#               o Recovery_Port :   Connection port for recovery
#               o Rsync_Path :      Path of rsync command
#               o Rsync_Option:     File transfer option for rsync
#               o Rsync_Compress :  Use compression option for rsync
#               [yes/no]. default: yes
```

```
#               o Pg_Dump_Path:     path of pg_dump
#               o When_Stand_Alone :            When all replication servers fell,
#                                       you can set up two kinds of permittion,
#                                       "real_only" or "read_write".
#               o Replication_Timeout : Timeout of each replication request
#               o Lifecheck_Timeout :       Timeout of the lifecheck response
#               o Lifecheck_Interval :      Interval time of the lifecheck
#                                   (range 1s - 1h)
#                                   10s   -- 10 seconds
#                                   10min -- 10 minutes
#                                   1h    -- 1 hours
#-------------------------------------------------------------
<Host_Name>                     c1.pgcluster.org                </Host_Name>
<Recovery_Port>                 7001                            </Recovery_Port>
<Rsync_Path>                    /usr/bin/rsync                  </Rsync_Path>
<Rsync_Option>                  ssh -1                          </Rsync_Option>
<Rsync_Compress>                yes                             </Rsync_Compress>
<Pg_Dump_Path>                  /usr/local/pgsql/bin/pg_dump    </Pg_Dump_Path>
<When_Stand_Alone>              read_only                       </When_Stand_Alone>
<Replication_Timeout>           1min                            </Replication_Timeout>
<LifeCheck_Timeout>             3s                              </LifeCheck_Timeout>
<LifeCheck_Interval>            11s                             </LifeCheck_Interval>
#-------------------------------------------------------------
```

```
# set partitional replicate control information
#          set DB name and Table name to stop reprication
#            o DB_Name :                DB name
#            o Table_Name :      Table name
#------------------------------------------------------------
#<Not_Replicate_Info>
#        <DB_Name>                test_db        </DB_Name>
#        <Table_Name>             log_table      </Table_Name>
#</Not_Replicate_Info>
```

Para el servidor de replicación.

Ir a la carpeta share donde están las configuraciones de ejemplo y se copia el fichero **pgreplicate.conf.sample** a **pgreplicate.conf**

```
#cd /usr/local/pgsql/share
#cp  pgreplicate.conf.sample    pgreplicate.conf
```

Editar el fichero **pgreplicate.conf**:

Poniendo la información de los ip de los clústeres y el ip de el mismo al igual que el balanceador.

```
<Cluster_Server_Info>
        <Host_Name>              c1.pgcluster.org      </Host_Name>
        <Port>                   5432                  </Port>
        <Recovery_Port>          7001                  </Recovery_Port>
```

```
</Cluster_Server_Info>
<Cluster_Server_Info>
        <Host_Name>             c2.pgcluster.org    </Host_Name>
        <Port>                  5432                </Port>
        <Recovery_Port>         7001                </Recovery_Port>
</Cluster_Server_Info>



<LoadBalance_Server_Info>
        <Host_Name>             lb.pgcluster.org    </Host_Name>
        <Recovery_Port>         6001                </Recovery_Port>
</LoadBalance_Server_Info>



<Host_Name>                 pgr.pgcluster.org   </Host_Name>
<Replication_Port>          8001                </Replication_Port>
<Recovery_Port>             8101                </Recovery_Port>
<RLOG_Port>                 8301                </RLOG_Port>
<Response_Mode>             normal              </Response_Mode>
<Use_Replication_Log>       no                  </Use_Replication_Log>
<Replication_Timeout>       1min                </Replication_Timeout>
<LifeCheck_Timeout>         3s                  </LifeCheck_Timeout>
```

<LifeCheck_Interval> 15s </LifeCheck_Interval>

Y se levanta el replicador:

4-1. replication server

A. Start replication server

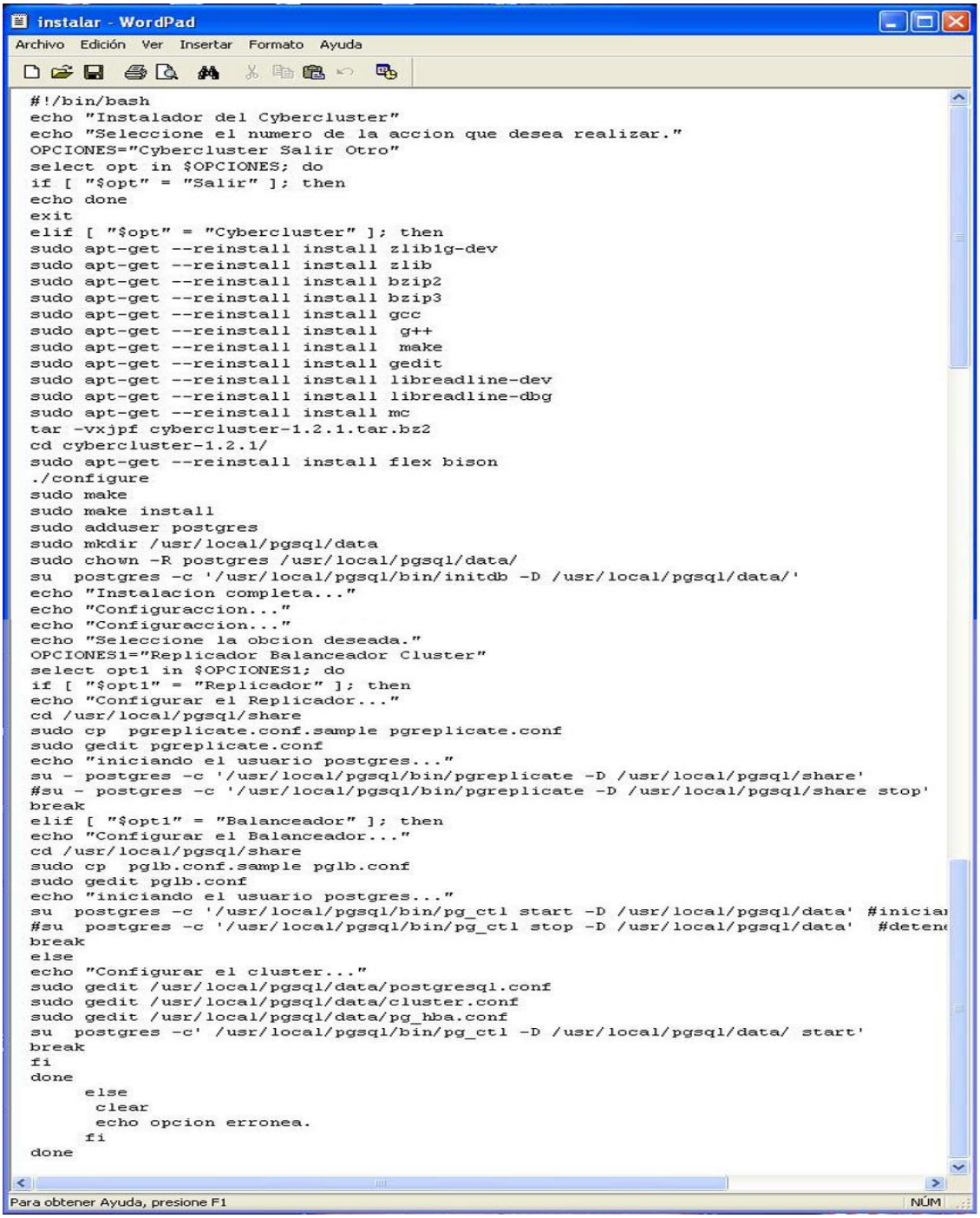
$ **/usr/local/pgsql/bin/pgreplicate -D /usr/local/pgsql/etc**

B. Stop replication server

$/usr/local/pgsql/bin/pgreplicate -D /usr/local/pgsql/share stop

---

Anexo # 4: Script de instalación del Cybercluster.



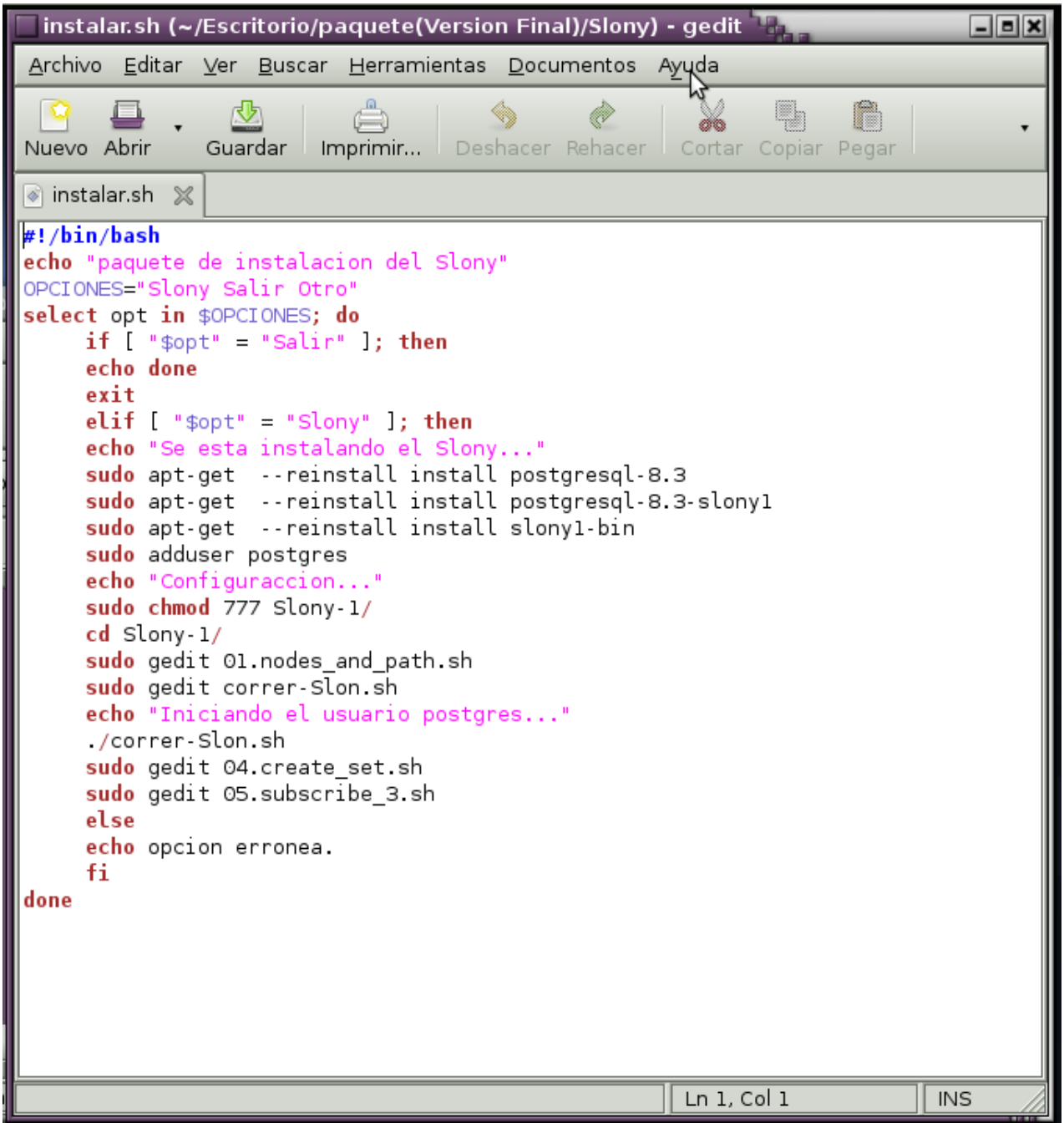
instalar - WordPad

Archivo Edición Ver Insertar Formato Ayuda

```
#!/bin/bash
echo "Instalador del Cybercluster"
echo "Seleccione el numero de la accion que desea realizar."
OPCIONES="Cybercluster Salir Otro"
select opt in $OPCIONES; do
if [ "$opt" = "Salir" ]; then
echo done
exit
elif [ "$opt" = "Cybercluster" ]; then
sudo apt-get --reinstall install zlib1g-dev
sudo apt-get --reinstall install zlib
sudo apt-get --reinstall install bzip2
sudo apt-get --reinstall install bzip3
sudo apt-get --reinstall install gcc
sudo apt-get --reinstall install  g++
sudo apt-get --reinstall install  make
sudo apt-get --reinstall install gedit
sudo apt-get --reinstall install libreadline-dev
sudo apt-get --reinstall install libreadline-dbg
sudo apt-get --reinstall install mc
tar -vxjpf cybercluster-1.2.1.tar.bz2
cd cybercluster-1.2.1/
sudo apt-get --reinstall install flex bison
./configure
sudo make
sudo make install
sudo adduser postgres
sudo mkdir /usr/local/pgsql/data
sudo chown -R postgres /usr/local/pgsql/data/
su  postgres -c '/usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data/'
echo "Instalacion completa..."
echo "Configuraccion..."
echo "Configuraccion..."
echo "Seleccione la obcion deseada."
OPCIONES1="Replicador Balanceador Cluster"
select opt1 in $OPCIONES1; do
if [ "$opt1" = "Replicador" ]; then
echo "Configurar el Replicador..."
cd /usr/local/pgsql/share
sudo cp  pgreplicate.conf.sample pgreplicate.conf
sudo gedit pgreplicate.conf
echo "iniciando el usuario postgres..."
su - postgres -c '/usr/local/pgsql/bin/pgreplicate -D /usr/local/pgsql/share'
#su - postgres -c '/usr/local/pgsql/bin/pgreplicate -D /usr/local/pgsql/share stop'
break
elif [ "$opt1" = "Balanceador" ]; then
echo "Configurar el Balanceador..."
cd /usr/local/pgsql/share
sudo cp  pglb.conf.sample pglb.conf
sudo gedit pglb.conf
echo "iniciando el usuario postgres..."
su  postgres -c '/usr/local/pgsql/bin/pg_ctl start -D /usr/local/pgsql/data' #inicia
#su  postgres -c '/usr/local/pgsql/bin/pg_ctl stop -D /usr/local/pgsql/data'  #detene
break
else
echo "Configurar el cluster..."
sudo gedit /usr/local/pgsql/data/postgresql.conf
sudo gedit /usr/local/pgsql/data/cluster.conf
sudo gedit /usr/local/pgsql/data/pg_hba.conf
su  postgres -c' /usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data/ start'
break
fi
done
      else
       clear
       echo opcion erronea.
      fi
done
```

Para obtener Ayuda, presione F1 NÚM

Anexo # 5: Script de instalación del Slony.



Anexo # 6: Script de instalación del Pgcluster.

Anexo # 7:Resultados de las pruebas de validación efectuadas.

UCi
Universidad de las Ciencias Informáticas

Centro de Tecnologías de Almacenamiento y Análisis de Datos

La Habana, 29 de mayo de 2009

Aniversarios 50 de la Revolución

Por este medio quedan registradas los resultados de las pruebas efectuadas al producto: **Paquete de Instalación de Réplica en Servidores PostgreSQL**.

Se realizaron varias pruebas experimentales para demostrar que cumplía con los requerimientos funcionales trazados. Dichos experimentos se llevaron a cabo con la utilización de varias PC donde se realizó el proceso de instalación y configuración de la soluciones de réplicas *Cybercluster*, *Pgcluster*, y *Slony*.

Este proceso de hizo de manera repetitiva, es decir se probó el funcionamiento del producto en más de una ocasión, arrojando en cada prueba resultados satisfactorios en comparación con la realización de este proceso antes de aplicar la solución elaborada. Los resultados obtenidos son los siguientes:

- ✓ **Disminución del tiempo:** El tiempo empleado por el usuario para realizar el proceso de instalación y configuración de las herramientas de réplicas empaquetadas disminuyo en gran medida, pues ya se pueden realizar la instalación y la configuración de manera conjunta.
- ✓ **Disminución en el esfuerzo:** El esfuerzo empleado por el usuario para realizar el proceso de instalación y configuración de las herramientas empaquetadas también disminuyo considerablemente , en este caso una vez comenzado el proceso de configuración la herramienta muestra los ficheros de configuración de cada herramienta automáticamente.
- ✓ **Disminución de la dependencia del conocimiento:** Anteriormente el usuario debía poseer un amplio conocimiento para realizar estos procesos, con esta herramienta se le brinda toda la documentación necesaria, por lo que puede ser utilizada por usuarios principiantes en estos temas.
- ✓ **Aumento en la calidad:** Se evidencia un aumento en la calidad del proceso de instalación y configuración de estas herramientas, pues el producto en general cuenta con todos los requisitos necesarios para esto. Provee los instaladores, los archivos de configuración, las guías de instalación y configuración de cada herramienta empaquetada así como el manual de usuario para el buen eso del producto en general.

De esta manera queda demostrado el buen funcionamiento y la validez del producto elaborado.

Dr.C. Pedro Yobanis Piñero Pérez
Jefe General
Centro de Tecnología de Almacenamiento y Análisis de Datos.