UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE INGENIERÍA

APLICACIÓN DE LA TECNOLOGÍA GRID EN BASES DE DATOS ORACLE

Tesis

Que para obtener el grado de:

Maestro en Ingeniería

Presenta:

JOSE ELENO LOZANO RIZK

Ensenada, Baja California **Noviembre del 2007**

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE INGENIERÍA
UNIDAD ENSENADA

Aplicación de la Tecnología Grid en Bases de Datos Oracle

TESIS

Que para obtener el grado de maestría en ingeniería presenta:

José Eleno Lozano Rizk

Aprobada por:

MC. Raúl Hazas Izquierdo
Director de tesis

Dr. Juan Ivan Nieto Hipólito
Miembro del comité

MI. Víctor Velázquez Mejía
Miembro del comité

Ensenada Baja California, México. Noviembre 2007

DEDICATORIA

A mi esposa por todo su amor, cariño y apoyo que me ha brindado.

Y a mis padres por su gran apoyo en las decisiones que he tomado.

AGRADECIMIENTOS

Deseo agradecer al Ing. André Martínez y al Ing. Luis Hernández por haber facilitado el equipo necesario para la realización de este trabajo.

A mi director de tesis, M.C. Raúl Hazas Izquierdo, por su tiempo y dedicación así como sus grandes aportaciones y consejos en la realización de ésta.

A mis sinodales Dr. Juan Ivan Nieto Hipólito y M.I. Víctor Velázquez Mejía, por su tiempo y dedicación para la culminación de este trabajo.

CONTENIDO

I. INTRODUCCIÓN..... 3
I.1Antecedentes..... 3
I.2 Objetivos..... 5
I.2.1 Objetivo general..... 5
I.2.2 Objetivos específicos..... 5
I.3 Alcances y limitaciones..... 6
I.4 Organización de la tesis..... 7

II. ORACLE Y LA TECNOLOGÍA GRID..... 8
II.1 ¿Qué es la tecnología Grid?..... 8
II.2 Bases de datos Oracle..... 9
II.3 Tecnología Grid en bases de datos Oracle..... 12
II.3.1 Oracle Real Application Clusters (RAC) 13
II.3.2 Escalabilidad de Oracle Real Application Clusters..... 15

III. APLICACIÓN REAL Y EL MODELO EXPERIMENTAL DE ORACLE RAC... 17
III.1 Aplicación real..... 17
III.2 Modelo experimental de Oracle RAC..... 21
III.2.1 Software..... 25
III.2.2 Hardware..... 25
III.2.3 Proceso de instalación y configuración del cluster..... 27

IV. METODOLOGIA DE PRUEBAS PARA EL MODELO EXPERIMENTAL..... 29
IV.1 Descripción de la metodología..... 29
IV.2 Aplicación de la metodología de pruebas al modelo experimental..... 31
IV.2.1 Pruebas al modelo con un solo usuario..... 32
IV.2.2 Pruebas al modelo en un entorno multiusuario..... 35

V. CONCLUSIONES..... 38

GLOSARIO..... 40

REFERENCIAS..... 42

ANEXOS..... 44
1. Instalación y configuración de los sistemas operativos en los nodos..... 44
2. Instalación y configuración del software para Oracle RAC..... 49
3. Creación de una base de datos para Oracle RAC..... 56
4. Versión "Artículo" de la tesis..... 57

INDICE DE FIGURAS Y TABLAS

Figura II.1 Tecnología Grid de Oracle..... 12

Figura II.2 Izquierda: Arquitectura de cache compartido. Derecha: Arquitectura no compartida..... 13

Figura II.3 Oracle Real Applications Clusters..... 14

Figura II.4 Escalabilidad de Oracle RAC..... 16

Figura III.1 Entorno multiusuario para AplicacionX..... 18

Figura III.2 Uso de CPU en el servidor de terminales..... 19

Figura III.3 Porcentaje de CPU en el servidor de base de datos Oracle..... 20

Figura III.4 Modelo experimental de Oracle RAC utilizando un servidor NFS en Solaris..... 24

Figura IV.1 Implementación del modelo experimental de Oracle RAC para AplicacionX..... 32

Figura IV.2 Tiempos de ejecución de un usuario en Oracle RAC y un servidor sencillo de Oracle, en ambos se utilizó el servidor NFS como almacenamiento de la base de datos..... 34

Figura IV.3 Tiempos de ejecución promedio de tres usuarios simultáneos en Oracle RAC y un servidor sencillo de Oracle, en ambos se utilizó el servidor NFS como almacenamiento de la base de datos..... 36

Figura IV.4 Porcentaje de ganancia en tiempo de los procesos con Oracle RAC. La línea en rojo indica el promedio de ganancia de todos los procesos..... 37

Tabla II.1 Rango de usuarios por número de nodos..... 15

Tabla III.1 Descripción de la configuración del cluster experimental..... 26

I. INTRODUCCIÓN

I.1 Antecedentes.

La velocidad en los cambios tecnológicos lleva a las empresas a adaptarse lo más rápido posible para poder seguir compitiendo. Las organizaciones se convierten cada vez más adaptables a esta situación, pero a menudo, la capacidad de respuesta de sus sistemas de información no avanza con la misma rapidez. Al mismo tiempo, estas organizaciones desean obtener una mayor eficiencia de sus sistemas de información y reducir el costo de adquisición y/o actualización de equipo de cómputo.

Los grids computacionales son una nueva arquitectura en las tecnologías de la información que se adapta a las necesidades de cambio de cada organización. Se puede entender como grid computacional a la unificación de los recursos computacionales, es decir, que se puedan compartir los recursos sin importar que estén distribuidos geográficamente, y que desde la perspectiva del usuario, estos se vean como un solo recurso de cómputo virtual.

A principios del año 2000, surgieron los grids computacionales, los cuales ofrecen una solución a los problemas mencionados anteriormente, proporcionando una infraestructura de hardware y software que se adapte de manera fácil, rápida y económicamente accesible.

Algunas empresas han contribuido en la evolución de los grids computacionales, desarrollando software que permita el uso de esta tecnología, apoyando proyectos Open-Source y posteriormente ofreciendo sus propias versiones para incursionar en ámbitos diferentes del académico o científico. Tal es el caso de Sun Microsystems y Oracle Corporation, entre otras, que ya cuentan con soluciones de grid para un entorno empresarial. Oracle Corporation es una de las primeras compañías que ofreció una base de datos relacional diseñada para los grids computacionales.

El presente trabajo, está basado en un estudio de la tecnología grid y su aplicación en sistemas de información con bases de datos Oracle, con el fin de solucionar problemas de saturación de recursos y de inspección de datos en tiempo real, además de abrir el camino para futuras investigaciones sobre el tema.

I.2 Objetivos.

I.2.1 Objetivo general.

El objetivo principal de esta tesis es realizar un estudio sobre la Tecnología Grid y su aplicación en sistemas de información con bases de datos Oracle para solucionar los problemas de saturación de recursos y de inspección de datos en tiempo real que se pueden presentar al contar con uno o varios servidores de base de datos Oracle independientes, es decir, que sus recursos computacionales no se encuentren unificados.

I.2.2 Objetivos específicos:

- Realizar un estudio sobre la Tecnología Grid en bases de datos Oracle.
- Desarrollar un modelo experimental para la aplicación de esta tecnología.
- Desarrollar una metodología de pruebas para este modelo.
- Evaluar los resultados.

I.3 Alcances y limitaciones.

En el presente estudio, se da a conocer al lector, que hoy en día se cuenta con una amplia gama de opciones para la unificación de recursos computacionales en servidores de base de datos Oracle y de esa manera aprovechar al máximo los recursos con que se cuente.

Se da a conocer la implementación de un modelo experimental en un ambiente departamental, así como la metodología desarrollada para la fase de pruebas del modelo. Para la ejecución de las pruebas, se utiliza una aplicación real del tipo financiero, sin alteración alguna en su código.

Asimismo, se realiza un estudio sobre la aplicación de la tecnología grid en bases de datos Oracle y como ésta puede ser la solución para mejorar el desempeño de sistemas de información con bases de datos semejantes.

I.4 Organización de la tesis.

En el capítulo II se realiza un estudio sobre la aplicación de la tecnología grid en bases de datos Oracle.

En el capitulo III se desarrolla un modelo experimental para ejecutar una aplicación real en un ambiente de grid departamental para bases de datos Oracle.

En el capitulo IV se desarrolla una metodología para realizar pruebas al modelo experimental, se describe el proceso de ejecución de las mismas y se obtienen los resultados.

En el capitulo V se formulan las conclusiones referentes a esta tesis

II. ORACLE Y LA TECNOLOGÍA GRID

II.1 ¿Qué es la tecnología Grid?

La tecnología grid es un modelo emergente de cómputo que proporciona la habilidad de obtener un mayor rendimiento de cómputo aprovechando que computadoras interconectadas a través de una red, funcionen como una sola computadora virtual. El software que maneja a ese conjunto de computadoras se responsabiliza de distribuir la ejecución de procesos entre las mismas. Los grids se utilizan para resolver problemas que ocupen de gran poder de cómputo o una gran capacidad de datos, ya sea empleando redes locales o Internet.

Instituciones académicas y científicas han utilizado la tecnología grid desde hace varios años, explorando algunos de los problemas más complejos que enfrenta la humanidad. Tal es el caso del proyecto "The Smallpox Research Grid Project" [1], el cual hizo uso de la tecnología grid para obtener resultados en meses en lugar de años.

La tecnología grid permite la virtualización de recursos con el fin de crear una sola entidad a través de la cual los usuarios y las aplicaciones puedan acceder los recursos de cómputo y almacenamiento. Así como un usuario de Internet ve una instancia unificada del contenido Web, un usuario de grids, esencialmente ve una sola supercomputadora virtual.

Desde su núcleo, la tecnología grid esta basada en un conjunto de estándares abiertos, por ejemplo, Open Grid Services Architecture (OGSA) [2], lo que permite la comunicación entre entornos de cómputo heterogéneos e incluso, geográficamente dispersos.

II.2 Bases de datos Oracle.

En la actualidad existen varias empresas que ofrecen manejadores de base de datos relacionales, tal es el caso de Oracle Corporation.

Oracle esta diseñado para ser una base de datos trasladable; está disponible para las plataformas de mayor popularidad, desde Windows hasta las diferentes variedades de Unix. Por esta razón la arquitectura física de Oracle es diferente para cada sistema operativo.

Por ejemplo, en un sistema operativo Unix, Oracle está implementado como muchos diferentes procesos del sistema operativo, virtualmente, un proceso para cada función. En Unix ésta es la implementación correcta, debido al multiprocesamiento del mismo. Sin embargo, en Windows, ésta arquitectura no sería la adecuada y no trabajaría muy bien (seria lenta y no escalable). En Windows, Oracle está implementado como un proceso sencillo, con multihilos [3].

En Oracle existen dos términos que usualmente causan confusión, estos son: instancia y base de datos.

Una base de datos es un conjunto de archivos (archivos de datos, archivos temporales, archivos de bitácora y archivos de control).

Una instancia es un conjunto de procesos del sistema operativo, (o un solo proceso con muchos hilos), y el espacio de memoria asociado. Estos procesos pueden operar en una base de datos.

A fin de que una instancia pueda manipular los datos, la instancia debe "abrir" la base de datos. En la mayoría de los casos, una base de datos será abierta ("montada") por una instancia.

Sin embargo, en casos especiales, como en Oracle Real Application Clusters (Oracle RAC), se pueden tener muchas instancias simultáneamente abriendo una sola base de datos, la cual reside en un conjunto de almacenamiento compartido [3]. Esto da la ventaja de poder acceder a ésta sola base de datos desde muchas diferentes computadoras al mismo tiempo.

Este tipo de funcionalidad puede ser aprovechada por aplicaciones que requieran procesar un mayor volumen de transacciones, como los sistemas de compras en línea y los sistemas bancarios.

Archivos en Oracle.

En Oracle existen ocho tipos de archivos los cuales son utilizados por la base de datos y la instancia.

Los archivos asociados a una instancia son:

- *Archivos de parámetros*: Estos archivos contienen las rutas para que la instancia de Oracle pueda encontrar los archivos de control, también especifican ciertos parámetros de inicialización que definen algunas estructuras de memoria.
- *Archivos de rastreo*: Estos son archivos de diagnostico creados por un proceso servidor generalmente en respuesta a alguna condición de error.
- *Archivos de alerta*: Son similares a los archivos de rastreo, pero contienen información sobre eventos "esperados".

Los archivos asociados a una base de datos son:

- *Archivos de datos:* Estos archivos son para la base de datos, almacenan las tablas, los índices y otros segmentos.
- *Archivos temporales:* Estos archivos son utilizados para almacenamiento temporal.
- *Archivos de control:* Estos archivos contienen las rutas de los archivos de datos,

temporales, bitácoras, así como información relevante del estatus de los mismos.

- *Archivos de bitácora:* Almacenan las bitácoras de las transacciones.
- *Archivos de claves:* Estos archivos son utilizados para autenticar usuarios que realizan actividades administrativas a través de la red.

Memoria en Oracle.

Oracle maneja tres tipos principales de estructuras de memoria:

- *Área Global de Sistema (SGA):* Es un segmento de memoria el la cual todos los procesos de Oracle accederán a un punto o a otro. Su tamaño depende del espacio definido para la memoria compartida en el sistema operativo (SHMMAX).
- *Área Global de Proceso (PGA):* Esta es memoria que es privada para un solo proceso o hilo, y no es accesible para otros procesos.
- *Área Global de Usuario (UGA):* Esta es memoria asociada con la sesión activa del usuario.

Transacciones.

Las transacciones son una de las características principales de las base de datos. Su principal función es proporcionar integridad y consistencia a la base de datos.

Las transacciones en Oracle pueden proporcionar la consistencia de los datos todo el tiempo, no obstante que pueda existir un elevado número de accesos de datos concurrente. Estas transacciones exhiben todas las características ACID requeridas. ACID es un acrónimo de:

- *Atomicidad:* La secuencia íntegra de transacciones se completa o se aborta. No debe haber transacciones parciales.
- *Consistencia:* Una transacción toma la base de datos de un estado consistente al siguiente estado consistente.
- *Aislamiento:* Los efectos de una transacción no deben de ser visibles a otras

transacciones hasta que ésta ha sido terminada.

- *Durabilidad:* Una vez que la transacción termine, es permanente.

11.3 Tecnología Grid en bases de datos Oracle.

Oracle 10g es considerada como la primera infraestructura completa e integral que utiliza cabalmente el potencial de una grid computacional en bases de datos relacionales. Oracle 10g recoge los atributos fundamentales de las siguientes tecnologías computacionales:

Implementar uno a partir de muchos: Se trata de conseguir que un número indeterminado de máquinas funcionen como una sola. Para ello, se tiene que aplicar un proceso de virtualización en todas las capas del sistema y una zona común donde se almacenen los recursos (resource pooling).

Administrar muchos como si fueran uno: Para conseguirlo se debe contar con un software a la medida y de un sistema de administración centralizado.

Los dos puntos anteriores deben aplicarse a cada elemento del grid: sistemas de almacenamiento, bases de datos, servidores de aplicación y aplicaciones. Ver Figura II.1.

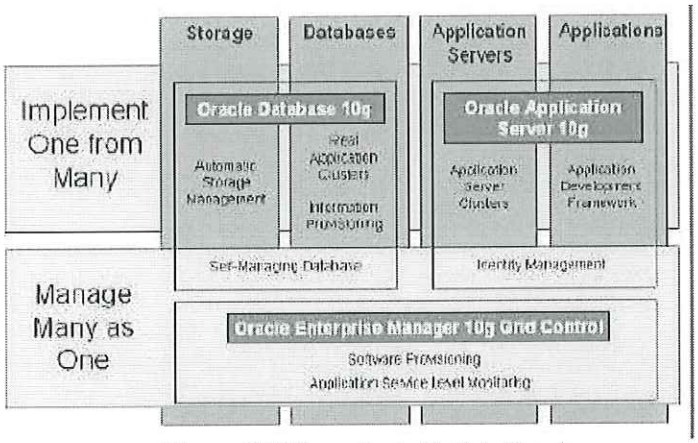


Figura II.1 Tecnología Grid de Oracle

La base de datos Oracle 10g añade muchas nuevas características enfocadas al cómputo

en grids. Mientras otros fabricantes implementan ciertas secciones de una tecnología grid, como por ejemplo una zona de almacenamiento común, Oracle es la primera compañía en ofrecer una base de datos relacional para grid real [5]. Oracle 10g se basa en Real Application Clusters, tecnología que es considerada por Oracle como el corazón de su cómputo en grids [6].

11.3.1 Oracle Real Application Clusters (RAC).

Oracle RAC es una base de datos en cluster con una arquitectura de cache compartido que sobrepasa las limitaciones de las bases de datos tradicionales (no compartidas) para proporcionar un mayor rendimiento a las aplicaciones que así lo demanden. [4] (ver Figura II.2)

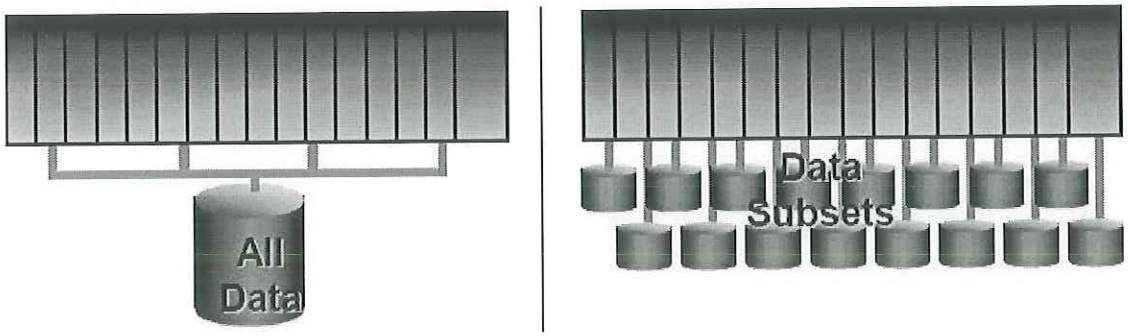


Figura II.2 Izquierda: Arquitectura de cache compartido. Varios servidores comparten el mismo cache así como el acceso a un sistema de almacenamiento donde está todo el conjunto de datos. Derecha: Arquitectura no compartida. Cada servidor tiene acceso solamente a un subconjunto de datos de manera local [5].

Un cluster es un grupo de servidores independientes que trabajan de forma conjunta para obtener un resultado. Los clusters proporcionan una mayor escalabilidad que los sistemas basados en SMP (single symmetric multiprocessor). En Oracle RAC si un servidor dentro del cluster falla, el acceso a los datos no se pierde y los usuarios pueden continuar trabajando. En caso de que se necesite un mayor poder de procesamiento, se le pueden agregar más nodos al cluster sin alterar la configuración actual [5].

Oracle RAC permite que varios nodos puedan acceder a la misma base de datos

simultáneamente. Esto proporciona tolerancia a fallas y balanceo de carga entre los servidores, en cierta manera, las consultas a la base de datos son distribuidas entre los nodos en el RAC para evitar que se sobrecargue alguno de ellos.

El corazón del Oracle RAC es el sistema de almacenamiento compartido (ver Figura II.3). Todos los nodos en el cluster deben tener acceso a los datos, bitácoras, archivos de control y archivos de parámetros.

Los discos de datos deben de estar disponibles para que todos los nodos tengan acceso a la base de datos. Cada nodo tiene sus propios archivos de bitácoras y archivos de control, sin embargo, los otros nodos deben de tener acceso a ellos para poderse recuperar en caso de una falla en el sistema [7].

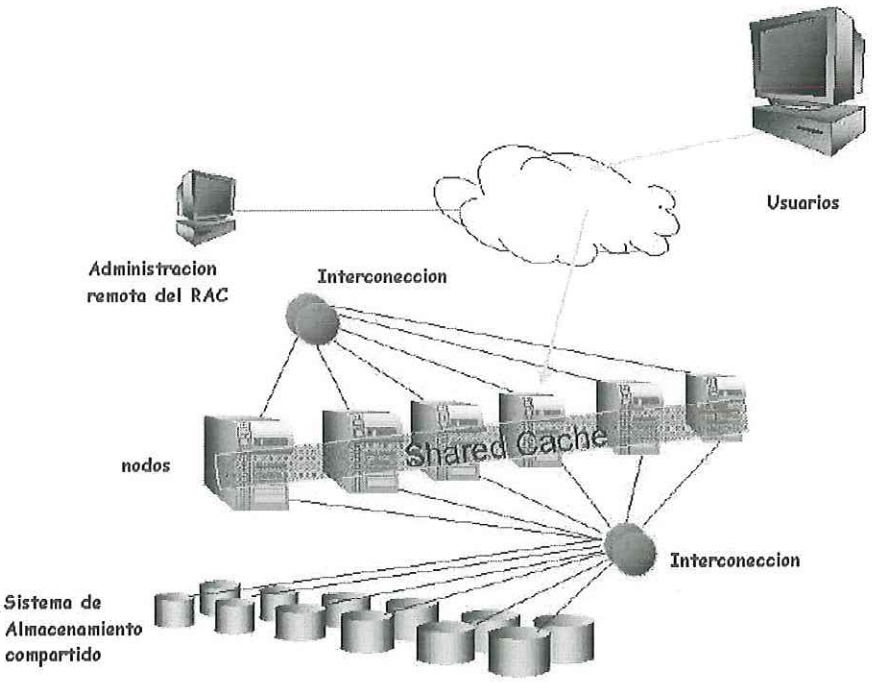


Figura II.3 Oracle Real Applications Clusters. [4]

Estos archivos deben estar dentro de un sistema de almacenamiento compartido el cual puede ser conformado por un NAS, un SAN, un ASM o un sistema de archivos para clusters.

No todas las soluciones de clusters para bases de datos utilizan un almacenamiento compartido. Algunas otras compañías usan una estrategia conocida como "Federated Cluster", en la cual los datos son distribuidos entre varias máquinas en lugar de que estén compartidos entre todos. Tal es el caso de Microsoft SQL Server [8].

Con Mysql, no es necesario contar con un sistema de almacenamiento compartido, funciona de manera similar a SQL Server, sin embargo, aun no es posible utilizar esta tecnología con bases de datos de varios gigabytes ya que utiliza principalmente la memoria RAM para realizar casi todas sus operaciones [10].

11.3.2 Escalabilidad de Oracle Real Application Clusters.

La compañía Dell Power Solutions realizó un estudio de escalabilidad para la versión 10g de Oracle RAC utilizando equipo propietario [13].

El principal objetivo de este estudio era obtener el número de transacciones por segundo conforme se iba aumentando el número de usuarios y número de nodos del cluster.

La aplicación que utilizaron para estresar a la base de datos del Oracle RAC, fue "Benchmark Factory for Databases" [14].

Una de las primeras pruebas que realizaron fue encontrar el numero "ideal" de usuarios por nodo, para este caso fue de 500 usuarios por nodo. La Tabla II.1 muestra el incremento de la carga hacia el cluster:

No. Nodos	No. Usuarios
1	100 a 500
2	100 a 1000
4	100 a 2000
6	100 a 3000
8	100 a 4000
10	100 a 5000

Tabla II.1 Rango de usuarios por número de nodos.

Durante toda la prueba se estuvieron monitoreando los nodos.

La Figura II.4 muestra los resultados de las pruebas para todos los nodos del cluster. Estos resultados indican que Oracle RAC escaló de forma casi lineal cuando se agregaban usuarios y nodos al RAC.

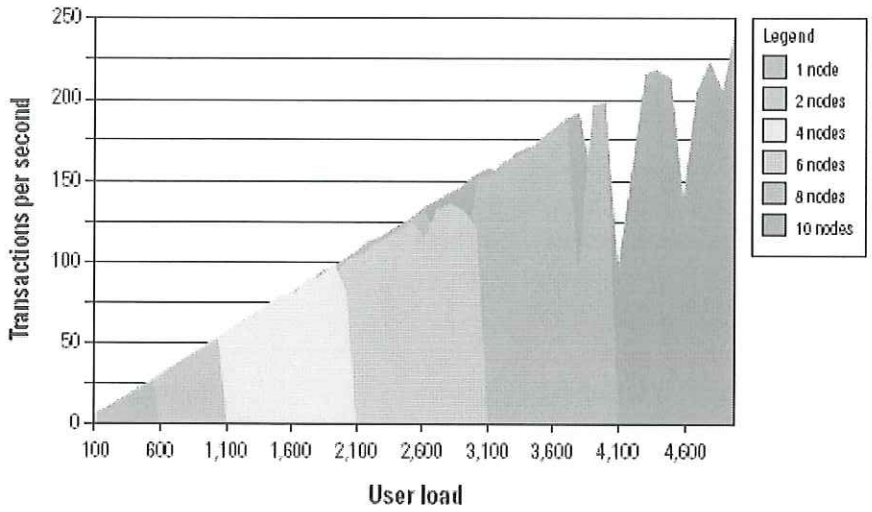


Figura II.4 Escalabilidad de Oracle RAC.

Como se observa en la Figura II.4 se muestran algunas caídas cuando se realizaron las pruebas con más de 4000 usuarios y más de 8 nodos, esto fue ocasionado (según autores del documento [13]) porque se acabó el espacio en disco asignado en una de las tablas de la base de datos, y se solucionó al incrementar el espacio a dicha tabla.

III. APLICACIÓN REAL Y EL MODELO EXPERIMENTAL DE ORACLE RAC

III.1 Aplicación real.

La aplicación a ejecutar en el modelo experimental de Oracle RAC, es un sistema de información de tipo financiero el cual utiliza a Oracle como su manejador de base de datos. Por cuestiones de confidencialidad el nombre de la aplicación fue sustituido por "AplicacionX". Este sistema fue desarrollado para la plataforma Win32 y cuenta con soporte para entornos multiusuarios. Actualmente, AplicacionX es utilizada por compañías de gran prestigio internacional.

Mediante tal aplicación se elaboran propuestas para la construcción de equipo militar y aeroespacial por parte de empresas privadas a gobiernos de países que así lo requieran, en su mayoría, al gobierno de EUA. Los cálculos para obtener los costos de las propuestas que realiza AplicaciónX requieren la mayor precisión ya que un centavo que se este perdiendo o ganando en una pieza, puede repercutir en cientos de miles de dólares tanto para ganancia como para perdida. Además, el tiempo de respuesta en la entrega de propuestas es vital para los clientes de esta aplicación ya que compiten entre ellos para entregar la mejor propuesta en el menor tiempo posible.

AplicacionX posee un alto grado de complejidad en cuestión de programación, y está compuesta por cinco módulos principales. Cada módulo tiene diversas operaciones como: manejo de reportes estándar y con formatos especiales; importar y exportar información en formatos propios así como en estándares abiertos (XML); realizar respaldos de toda la información del esquema de Oracle; restaurar esa información con manejo de duplicados; y el módulo principal y más importante realiza el cálculo para la obtención de costos. La base de datos puede llegar a consumir varios gigabytes de información, aunque varia dependiendo de cada cliente del producto, sin embargo, se tiene conocimiento que varios de ellos, sobre todo las compañías aeroespaciales, trabajan con varios millones de registros en el esquema de Oracle de esta aplicación.

El mercado de AplicacionX está enfocado a pequeñas y medianas empresas (pymes) y grandes corporaciones. Los especialistas de estas empresas trabajan de manera colaborativa para la elaboración de sus propuestas. Sin embargo se han presentado algunos problemas cuando ésta es utilizada en el entorno multiusuario con un escenario como el siguiente:

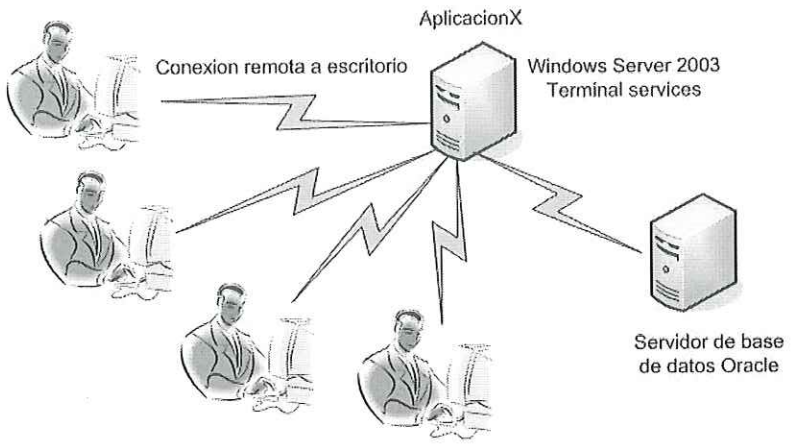


Figura III.1 Entorno multiusuario para AplicacionX.

La Figura III.1 muestra el escenario más utilizado por las empresas para ejecutar AplicacionX.

- Se instaló AplicacionX dentro de un Servidor de Terminales de Microsoft [11].
- Se creó la base de datos Oracle de AplicacionX en un servidor sencillo.
- 10 usuarios se conectaron desde su máquina a AplicacionX por medio del paquete "Conexión Remota a Escritorio" de Microsoft.
- Cada usuario ejecutó el mismo proceso* de AplicacionX de manera simultánea, utilizando los mismos datos de entrada.

*El proceso a ejecutar es un paquete [3] que esta dentro del esquema de AplicacionX en el servidor de Oracle, el cual realiza consultas y operaciones aritméticas.

Primeramente, se ejecutó ese proceso por un solo usuario con los mismos datos de

entrada, tardó 38 segundos aproximadamente en terminar.

Después, se ejecutó la misma prueba pero con 10 usuarios de manera simultánea. La Figura III.2 muestra el porcentaje de CPU utilizado por el servidor de terminales donde están conectados los usuarios. Este servidor tardó aproximadamente 7 minutos en que todos los usuarios terminaran de ejecutar el proceso de AplicacionX. El primer usuario que terminó el proceso tardó 50 segundos aprox.

Comparando los tiempos tanto del usuario independiente contra el del primer usuario que terminó en el entorno multiusuario, se tiene un incremento en el lapso de ejecución de los procesos de aproximadamente 12 segundos en el entorno multiusuario.

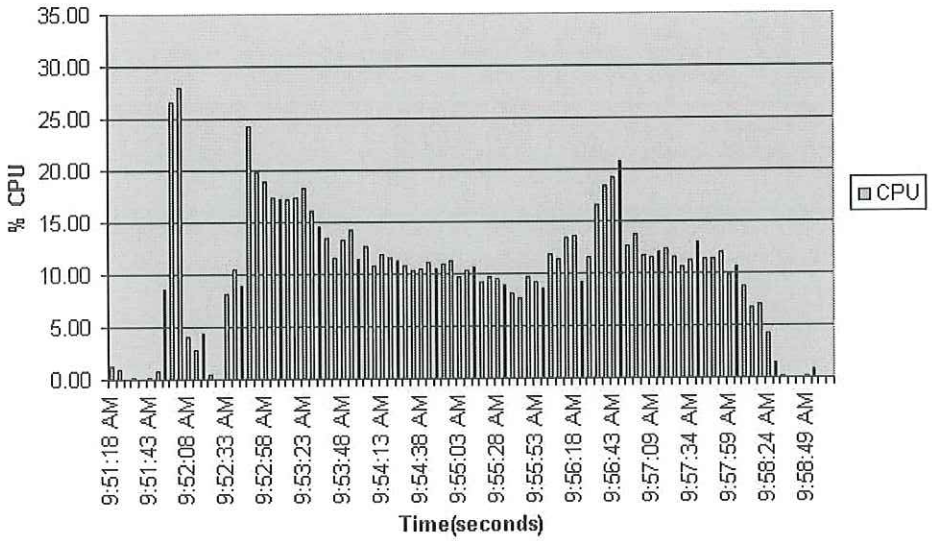


Figura III.2 Uso de CPU en el servidor de terminales.

Como se puede observar en la Figura III.2, el uso de CPU del servidor de terminales no llego ni al 30%, lo cual indica que no hubo saturación de recursos en ese servidor.

Sin embargo, la Figura III.3 muestra el porcentaje de CPU utilizado en el servidor de base de datos Oracle. Como se observa, el servidor presentó picos de más del 95% de uso de CPU durante los 7 minutos que duró la ejecución de los procesos.

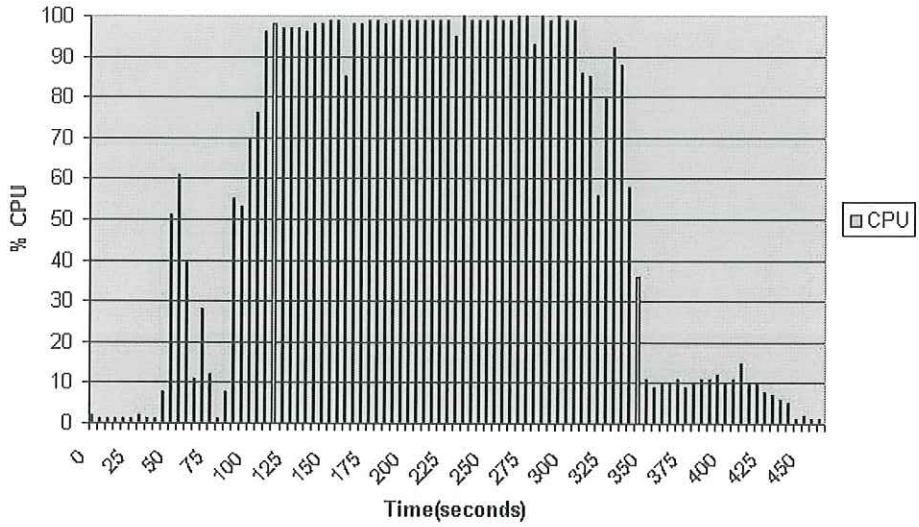


Figura III.3 Porcentaje de CPU en el servidor de base de datos Oracle.

Esto presenta dos desventajas:

1) Un usuario tiene que esperar más tiempo para obtener sus resultados en un entorno multiusuario, del que toma cuando se trata de un solo usuario.
2) El servidor de base de datos Oracle se satura de peticiones al estar en este tipo de escenario multiusuarios, y trae como consecuencia un aumento en el tiempo de ejecución de los procesos de cada usuario. Además, el servidor de base de datos Oracle no solo es utilizado por AplicacionX sino por otras aplicaciones, por lo tanto, aquellas también se pudieran ver afectadas.

Una posible solución al problema de saturación del servidor, sería contar con otro servidor de base de datos Oracle y repartir los usuarios entre ambos servidores. Sin embargo, al hacerlo, se tienen que estar actualizando los datos en ambos servidores.

Esto genera más trabajo a los administradores (dba) ya que tendría que existir alguna forma de sincronización entre los servidores; también genera molestias al usuario ya que tendría que configurar su máquina para poderse conectar a un servidor y luego de nuevo configurarla para entrar al otro, además, si él llegara a necesitar cierto dato que está en

otro servidor y no se ha sincronizado, tendría que esperar o solicitarlo al DBA.

Un escenario real de esta aplicación es tener más de 1000 usuarios conectados de manera simultánea a AplicacionX, por lo tanto es necesario mejorar el rendimiento en su base de datos Oracle.

El estudio de escalabilidad realizado por Dell, muestra el comportamiento del Oracle RAC al forzar al límite la base de datos y al incrementar el número de nodos. Sin embargo, eso no asegura que AplicaciónX podría tener un comportamiento similar ya que la arquitectura de esta aplicación es muy diferente al software utilizado para poner a prueba el rendimiento de la base de datos.

En las pruebas de Dell, en su mayoría, el software realiza transacciones o "querys" a una tabla o un conjunto de tablas pero desde el código de esa aplicación, y en el caso de AplicacionX, los módulos principales realizan llamadas a paquetes que se encuentran incrustados dentro del servidor de Oracle. Los paquetes en Oracle [3] son elementos de programación en lenguaje PL/SQL que permiten combinar procedimientos, funciones, definiciones de tipos de datos y declaraciones de variables en una misma estructura. Se encuentran dentro del servidor de Oracle para que puedan ser utilizados por la aplicación.

Con el fin de lograr un mejor rendimiento en la base de datos de Oracle de AplicacionX, se propuso la implementación de un modelo experimental de Oracle Real Application Clusters.

III.2 Modelo Experimental de Oracle RAC.

Para el desarrollo de este modelo se tomó como guía una implementación de Oracle RAC [9] utilizando un cluster de PC's y un disco duro externo firewire como sistema de almacenamiento compartido.

Al tener el Oracle RAC funcionando, se procedió a ejecutar ciertos procesos de

AplicacionX (Ver Capítulo IV). Sin embargo, no fue posible llevarlos a cabo debido a una serie de problemas de comunicación entre los nodos y el disco duro firewire.

El problema se debió particularmente a que el controlador permite el acceso simultáneo al disco duro externo por dos instancias y, por razones que no pude determinar, observé que cuando un nodo realizaba una lectura al disco, este tardaba en responder hasta dos segundos en algunos casos. Esto trajo como consecuencia que al ejecutar el primer proceso de AplicacionX con el disco firewire, la base de datos y los registros del RAC se corrompieron y fue imposible continuar.

Aún más, desde el momento de la instalación de los nodos del cluster, al crear las particiones en el disco duro externo, se notó que tardaban unos segundos en lograr acceder a la información del disco duro externo. Este tipo de retardos no es tolerable en un ambiente de bases de datos relacionales y es por eso que Oracle recomienda el uso de sistemas de almacenamiento compartido que estén certificados por ellos, de preferencia del tipo SAN o NAS [7].

Debido a estos problemas, fue necesario buscar otra forma de implementar un Oracle RAC. Se tomó la opción de utilizar un servidor del tipo NAS como almacenamiento compartido. Sin embargo, este tipo de dispositivos certificados involucran una inversión de varios miles de dólares (algunas cotizaciones estuvieron en el rango entre los 20 mil y 40 mil dólares americanos por un sistema con la configuración mínima requerida por los proveedores –entre 5 y 15 TB-), y no se contaba con dicho presupuesto para efectos de la implementación de un modelo experimental.

El por qué de que estos dispositivos sean "certificados" es debido a que garantizan que este tipo de base de datos en cluster no se verá corrompida por algún evento inesperado de E/S, como sucedió con el disco duro firewire. Para el caso del NAS, garantizan que el protocolo NFS que ellos utilizan es menos susceptible a fallas de E/S.

Haciendo caso omiso de lo anterior, se tomó una opción relativamente económica y se

consiguió un disco duro externo (similar a un NAS con costo aproximado de 300 dólares americanos) al cual se le puede asignar una dirección IP y compartirlo por medio de una red, ya sea por el protocolo CIFS para Windows o NFS para Unix. Sin embargo, ni siquiera fue posible instalar el software para el manejo del cluster de Oracle, ya que en el proceso para configurar el "oracle cluster registry" (que es uno de los archivos compartidos por los nodos) el sistema indicó que las instancias del cluster no podían acceder a ese archivo.

Estudiando la situación, llegué a la conclusión de que el problema radicaba en el protocolo NFS, dado que tanto las opciones de exportación – montaje, como la versión de NFS en esos dispositivos no son las adecuadas para el Oracle RAC, por lo tanto, como NFS fue desarrollado por Sun Microsystems hace ya varios años, se decidió configurar un servidor con sistema operativo Solaris (propietario de Sun) y utilizar el protocolo NFS de ese sistema y así verificar el funcionamiento de Oracle RAC utilizando las opciones de exportación y montaje de NFS en Solaris.

Para que un sistema NFS pueda almacenar una base de datos de Oracle RAC debe considerar las siguientes opciones en el montaje:

- Opción "*hard*" ya que si llegara a "congelarse" el servidor NFS, esta opción no permitirá que se sigan escribiendo datos, de esa manera ayuda a prevenir la corrupción de los mismos.
- Opción "*nointr*" para evitar que se puedan enviar señales para matar procesos colgados de NFS debido a una falla en el servidor NFS. Es preferible que se quede colgado el proceso mientras se levanta de nuevo el servidor NFS, a que se corra el riesgo de pérdida o corrupción de datos por una interrupción.
- Opciones "*rsize y wsize*" que tengan un tamaño de 32768 bytes por cada lectura y escritura, y así mejorar el rendimiento de estos procesos.
- Opción "*noac*" para inhabilitar el cache y de esa manera siempre tener actualizados los archivos.
- Opciones "*tcp y vers*" el montaje de los directorios debe de ser por tcp, por la

confiabilidad de este protocolo, y NFS versión 3 o más reciente.

Finalmente, utilizando un servidor Solaris (versión 10) como almacenamiento compartido por NFS, fue posible lograr una instalación satisfactoria de Oracle RAC ya que esta versión si soporta las opciones de exportación y montaje recomendadas para dicha tecnología (Figura III.4). Estas opciones son relevantes para el funcionamiento de Oracle RAC ya que permiten que se pueda almacenar la base de datos y los archivos de control en el servidor Solaris y compartirlos por NFS a los demás nodos de Oracle RAC. Cabe mencionar que este modelo no está documentado por Oracle y solo lo recomiendo para pruebas y no para ambientes de producción.

La Figura III.4 muestra el modelo experimental de Oracle RAC. Se utilizó un servidor Solaris 10 con NFS [12] para almacenar la base de datos así como los registros del RAC. Según la documentación de Oracle solo es posible utilizar NFS en dispositivos certificados como los de la marca Network Appliance y lo ideal sería que se utilicen dispositivos SAN [7]. Sin embargo, para este modelo, si fue posible utilizar un servidor NFS en Solaris 10 en una PC de 32 bits.

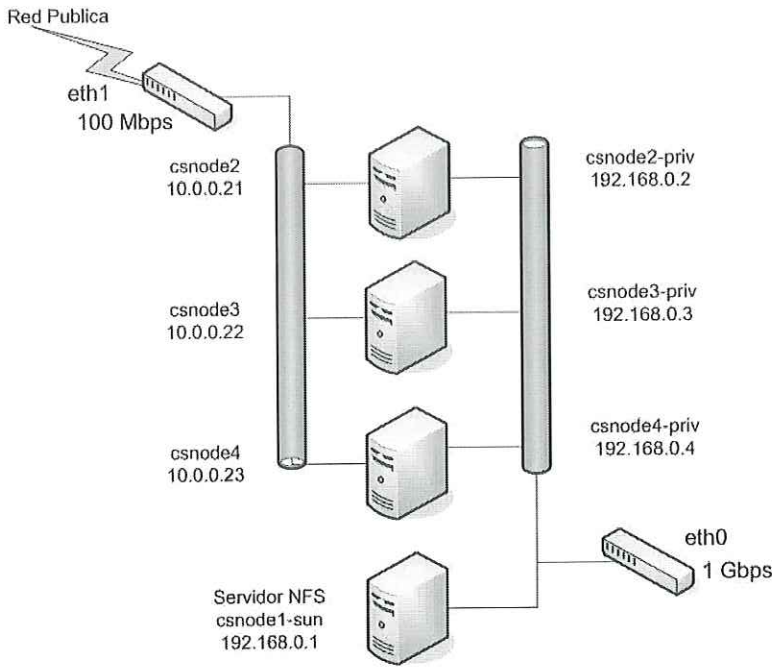


Figura III.4 Modelo experimental de Oracle RAC utilizando un servidor NFS en Solaris.

III.2.1 Software.

Se utilizó el siguiente software para el modelo experimental del RAC.

- CentOS Enterprise Linux 4.2 (kernel 2.6.9.22.EL)
- Solaris 10 OS, 32bits x86(PC)
- Oracle Database 10g Release 2 EE.
- Clusterware 10.2.0.1.0.

III.2.2 Hardware.

Para el desarrollo de este modelo, se utilizó el siguiente hardware:

Cluster de PCs con 3 nodos.

- Procesador Intel Pentium 4 a 3.0 GHz.
- Tarjeta madre Intel.
- 1GB DDR SDRAM (a 333MHz).
- 120GB 7200 RPM Disco duro interno (SATA).
- Tarjeta de video Integrada.
- Tarjeta de red integrada de 1 Gbps.
- Una tarjeta de red PCI 10/100.
- DVD ROM.

Sistema de almacenamiento.

- 1 PC con las mismas características que los nodos del cluster.

Red de Interconexión.

- 1 switch de 1 Gbps Dlink.
- 1 switch de 10/100 Linksys.

Los nodos csnode2, csnode3, csnode4 tienen 2 interfaces (eth0, eth1).

La tarjeta de red eth0 está conectada al switch de 1 gbps para la interconexión privada (requerimiento del RAC [7]). Cada nodo en el RAC estará utilizando dicha interfaz para acceder a la base de datos.

La tarjeta de red eth1 está conectada al switch de 100 mbps, y es utilizada para acceder a la red pública.

El servidor Solaris 10 solo tiene activa una interfaz de red (1 gbps), está conectada al switch de 1 gbps. No tiene acceso a la red pública.

La Tabla III.1 muestra la información sobre los directorios, sistema de archivos y nombre de las instancias usadas en este modelo.

Los nombres de las instancias de Oracle deben ser diferentes en cada nodo del RAC. El software Oracle Clusterware contiene los paquetes de Oracle RAC, estos se instalarán en el directorio */u01/app/oracle/product/10.2.0/crs* en cada nodo del RAC.

Archivos de Oracle				
Nodo	Instancia	Base de datos	$ORACLE_HOME	Sistema de archivos
csnode2	csrac21	csrac2	/u01/app/oracle	Ext3 (local)
csnode3	csrac22	csrac2	/u01/app/oracle	Ext3 (local)
csnode4	csrac23	csrac2	/u01/app/oracle	Ext3 (local)
Archivos compartidos del Oracle RAC				
Tipo de archivo	Nombre	Directorio	Punto de montaje	Sistema de archivos
Oracle cluster registry	cmregfile	/share/oradata/oracrs	/share	NFS
CRS voting disk	cmdiskfile	/share/oradata/oracrs	/share	NFS
Archivos de la base de datos de Oracle				
Nodo	Instancia	Directorio	Punto de montaje	Sistema de archivos BD
csnode2	csrac21	/share/oradata/oradb	/share	NFS
csnode3	csrac22	/share/oradata/oradb	/share	NFS
csnode4	csrac23	/share/oradata/oradb	/share	NFS

Tabla III.1 Descripción de la configuración del cluster experimental.

Dos archivos son requeridos por el Clusterware para que estén compartidos por todos los nodos en el cluster: el registro del cluster de Oracle (OCR) y el Voting Disk. Estos dos archivos fueron instalados dentro del directorio compartido que esta montado vía NFS.

El software de la base de datos de Oracle fue instalado en */u01/app/oracle/product/10.2.0/db_1* dentro de cada nodo en el cluster.

Los archivos de la base de datos (datos, bitácoras, archivos de control, etc) físicamente fueron almacenados en el directorio montado por NFS.

III.2.3 Proceso de instalación y configuración del software.

Los pasos para la instalación y configuración de los sistemas operativos así como el software del Oracle RAC se encuentran en los anexos 1, 2 y 3.

Un aspecto importante a considerar en ese proceso, es el parámetro del sistema operativo Linux para la memoria compartida (SHMMAX). El manejador de Oracle utiliza la memoria compartida del sistema operativo para permitir que los procesos puedan acceder a datos y estructuras de datos comunes. Estos datos y estructuras de datos son almacenados en un segmento de memoria compartida para permitir que los procesos puedan hacer uso de la forma más rápida disponible para intercomunicación entre procesos (IPC).

Oracle utiliza a la memoria compartida en Linux para almacenar su SGA. Esta es un área de memoria en la instancia de Oracle que es compartida para todos los procesos. Es importante que el tamaño de la SGA sea el adecuado para almacenar el cache de la base de datos, los archivos de bitácoras, así como otras estructuras compartidas de Oracle. Si el tamaño no es el adecuado, esto puede degradar considerablemente el rendimiento de la base de datos.

Para este modelo en un principio se había dejado un tamaño de 512MB. Conforme se ejecutaron las primeras pruebas, en modo monousuario, se notó que el Oracle RAC estaba casi un 50% más lento que el servidor sencillo de Oracle.

Al estudiar el comportamiento de los procesos de Oracle en los nodos, se observó que

varios de ellos ocupaban bastante memoria RAM (mas del 80%), esto ocasionado por los datos de entrada de AplicacionX. Un comportamiento similar se observa con aplicaciones paralelas con MPI, cuando éstas utilizan más memoria RAM debido también a sus datos de entrada, y se ejecutan en el mismo nodo (un nodo con dos o mas procesadores o cores), estas suelen abortarse. La solución a esas tareas paralelas es el incremento del parámetro para la memoria compartida (SHMMAX), ya que utilizan el mecanismo IPC para realizar el paso de mensajes.

Como se mencionó anteriormente, Oracle hace uso de IPC para lograr una mayor velocidad en la comunicación de sus procesos, por lo tanto, el parámetro para la memoria compartida se fue incrementando gradualmente. Una vez que se incrementaba se ejecutaban de nuevo las pruebas. Se realizó este procedimiento hasta lograr un tiempo de ejecución muy similar al obtenido en el servidor sencillo. Para este caso, el tamaño óptimo fue de 2GB en todos los nodos del Oracle RAC.

Con esto, al ejecutar una prueba con un solo usuario, se logró un rendimiento en tiempo muy similar al del servidor sencillo de Oracle.

IV. METODOLOGÍA DE PRUEBAS PARA EL MODELO EXPERIMENTAL

IV.1 Descripción de la metodología.

El propósito de esta metodología, es comparar tiempos de ejecución de una aplicación de escritorio multiusuario corriendo en un servidor sencillo de Oracle, con tiempos de esa misma aplicación corriendo en el modelo experimental de Oracle RAC, ambos utilizando los mismos datos de entrada.

Dicha metodología fue desarrollada con el propósito de que pueda ser utilizada para probar cualquier implementación de Oracle RAC, sin importar la arquitectura de hardware y de software que esté siendo usado en el cluster, sino enfatiza cuáles son los lapsos para la obtención de resultados de una aplicación de escritorio que utilice como manejador de base de datos a un Oracle RAC.

Tanto en el servidor sencillo como en el Oracle RAC, es necesario utilizar los mismos elementos, tales como: datos de entrada, versión de la aplicación, número de usuarios simultáneos, mismas características de hardware y software en el equipo, etc.

A continuación se describen los pasos para establecer los elementos mínimos:

1.- Instalar el sistema operativo y la base de datos Oracle, la configuración debe de ser la misma para todos los nodos dentro del RAC así como para el servidor sencillo.

2.- Optimizar los parámetros del sistema operativo: Es necesario optimizar la configuración típica del sistema operativo para que mejor se adapte a Oracle. El anexo 2 describe la configuración documentada por Oracle.

3.- Obtener el rendimiento de la red en el cluster: Ejecutar aplicaciones como Iperf para verificar la velocidad de la red privada del Oracle RAC así como del servidor sencillo.

4.- Establecer puntos de referencia para la carga de los nodos: Entrar a los nodos del RAC y documentar la carga de CPU, memoria RAM utilizada y disponible, memoria virtual utilizada y disponible, espacio en disco. Hacer la misma documentación para el servidor sencillo.

Después de realizar los pasos anteriores, los pasos siguientes, definidos conforme a la manera en como las empresas utilizan AplicacionX en modo multiusuario, son usados para la ejecución de las pruebas al modelo experimental del Oracle RAC.

1.- En una máquina con hardware similar al de los nodos del RAC, instalar Windows Server 2003 y habilitar el servidor de Terminales.

2.- Instalar AplicacionX en el servidor Windows 2003 y configurar el cliente de Oracle para que se pueda comunicar con el Oracle RAC, de igual forma para el servidor sencillo.

3.- Crear la base de datos de prueba en el Oracle RAC y en el servidor sencillo.

4.- Desde el servidor Windows 2003, crear el esquema de AplicaciónX en la base de datos de Oracle RAC y en la base de datos del servidor sencillo. Si ya existe el esquema de aplicacionX, eliminarlo y crearlo de nuevo.

5.- Importar los datos de entrada al esquema de AplicacionX. Los mismos datos deben ser importados a la base de datos del Oracle RAC y a la base de datos del servidor sencillo.

6.- Desde una máquina "cliente" conectarse remotamente al servidor Windows 2003, abrir AplicacionX y ejecutar un proceso. Tomar los tiempos de inicio y fin, así como el porcentaje de uso de CPU en el Oracle RAC y en el servidor sencillo.

7.- Ejecutar el paso 6 con un usuario y luego con varios usuarios simultáneos.

8.- Documentar los resultados.

VI.2 Aplicación de la metodología de pruebas al modelo experimental.

Se configuró un servidor Windows 2003 con el Servidor de Terminales habilitado (se le llamará "codetest" a este servidor). Además, se le instaló la versión cliente de "Oracle 10g Release 2". Esto con el fin de que los usuarios accedieran remotamente a AplicacionX.

Hostname: codetest
IP: 10.0.0.200
Ejemplo de archivo *tnsnames.ora* para codetest:

C:\oracle\ora10gr2\network\admin\tnsnames.ora

```
csrac2 =
 (DESCRIPTION =
   (ADDRESS = (PROTOCOL = TCP)(HOST = csnode2-vip)(PORT = 1521))
   (ADDRESS = (PROTOCOL = TCP)(HOST = csnode3-vip)(PORT = 1521))
   (ADDRESS = (PROTOCOL = TCP)(HOST = csnode4-vip)(PORT = 1521))
   (LOAD_BALANCE = yes)
   (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = csractest2)
    )
   )
 )
```

NOTA: Los nombres csnode2-vip, csnode3-vip y csnode4-vip deben existir en el servidor DNS o dentro del archivo "host" de codetest.

Se crearon varias cuentas de usuario en el servidor codetest: user1, user2 y user3. Estos usuarios contaron con derechos como administrador (en los derechos de acceso remoto) ya que el software cliente de Oracle fue instalado en la unidad C, que es la misma donde fue instalado el sistema operativo, por lo tanto, fue necesario que los usuarios tuvieran acceso a esa unidad.

Como "Administrador" en codetest, se instaló AplicacionX y se creó la base de datos en el Oracle RAC y en el servidor sencillo. El procedimiento para la creación de esta base de datos no cambió para el Oracle RAC.

Se configuró AplicacionX para que pudiera hacer uso de esta base de datos en cluster tal y como si fuera para una base de datos sencilla. No fue necesaria una configuración especial.

La Figura IV.1 muestra la implementación del modelo experimental de Oracle RAC que fue utilizado por AplicacionX.

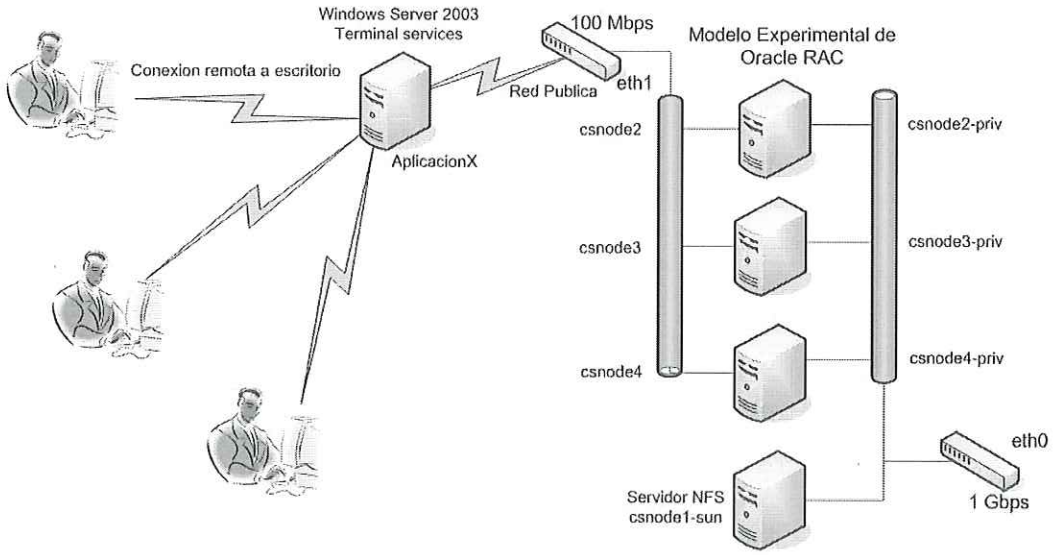


Figura IV.1 Implementación del modelo experimental de Oracle RAC para AplicacionX.

VI.2.1 Pruebas al modelo con un solo usuario.

Las pruebas se realizaron con un solo usuario.

Usuario: Administrator
Datos: Propuesta JLR Version 1.

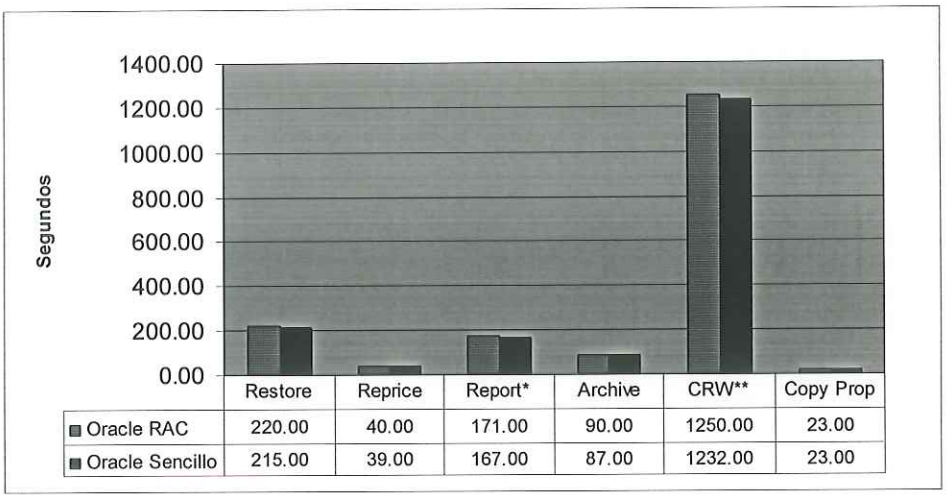
El usuario "Administrador" se conectó desde una máquina cliente a codetest por medio de Remote Desktop Connection (RDC). El Archivo de conexión utilizado para RDC, al conectarse, abrió el programa AplicacionX de manera automática. A continuación se muestra la configuración para este archivo (administrator.rdp):

```
screen mode id:i:1
desktopwidth:i:800
desktopheight:i:600
session bpp:i:16
winposstr:s:0,1,0,0,1246,933
full address:s:codetest
compression:i:1
keyboardhook:i:2
audiomode:i:0
redirectdrives:i:0
redirectprinters:i:1
redirectcomports:i:0
redirectsmartcards:i:1
displayconnectionbar:i:1
autoreconnection enabled:i:1
username:s:administrator
domain:s:CODETEST
alternate shell:s:c:\appx\winappx.exe
shell working directory:s:
password 51:b: xxxxx????
disable wallpaper:i:1
disable full window drag:i:1
disable menu anims:i:1
disable themes:i:0
disable cursor setting:i:0
bitmapcachepersistenable:i:1
```

1) Dentro de AplicaciónX, se ejecutó el proceso "Restore".
2) Se tomó el tiempo de inicio y finalización del proceso. Para este caso, AplicacionX escribió dentro de un archivo de texto en el directorio del usuario, una estampa de tiempo cuando inició y finalizó el proceso.
3) Se documentaron los resultados en una hoja de Excel.
4) Se repitieron los pasos 2-4 para el proceso "Reprice".
5) Se repitieron los pasos 2-4 para el proceso "Report".
6) Se repitieron los pasos 2-4 para el proceso "Archive".
7) Se repitieron los pasos 2-4 para el proceso "CRW".
8) Se repitieron los pasos 2-4 para el proceso "Copy Proposal".

Los resultados de las pruebas anteriores fueron comparados con los resultados de esas mismas pruebas ejecutadas en un servidor sencillo de base de datos Oracle (ver Figura IV.1). Dicho servidor fue de las mismas características tanto de hardware como de software que los servidores del Oracle RAC. Los datos se almacenaron dentro de un directorio del servidor Solaris, compartido a través de NFS, de igual forma que en Oracle RAC.

Un solo usuario se conectó a AplicacionX por medio del servidor de terminales. Como se muestra en la Figura IV.2, la base de datos del RAC en la mayoría de los casos fue un poco más lenta que el servidor sencillo de Oracle.



	Restore	Reprice	Report*	Archive	CRW**	Copy Prop
Oracle RAC	220.00	40.00	171.00	90.00	1250.00	23.00
Oracle Sencillo	215.00	39.00	167.00	87.00	1232.00	23.00

Figura IV.2 Tiempos de ejecución de un usuario en Oracle RAC y un servidor sencillo de Oracle, en ambos se utilizó el servidor NFS como almacenamiento de la base de datos.

La razón de esto fue la carga extra que se originó al contar con una configuración de base de datos en RAC debido a todos los paquetes de monitoreo y administración propios de esta arquitectura.

También se observó que, como los procesos que se ejecutaron en AplicacionX son paquetes de Oracle, el Oracle RAC los ejecutó de inicio a fin dentro del mismo nodo, es decir, no distribuyó las sentencias SQL de los paquetes entre los nodos.

Como consecuencia de esto, un solo usuario que ejecutó algunos procesos de AplicaciónX en el modelo experimental de Oracle RAC no pudo obtener una ganancia en tiempo.

VI.2.2 Pruebas al modelo en un entorno multiusuario.

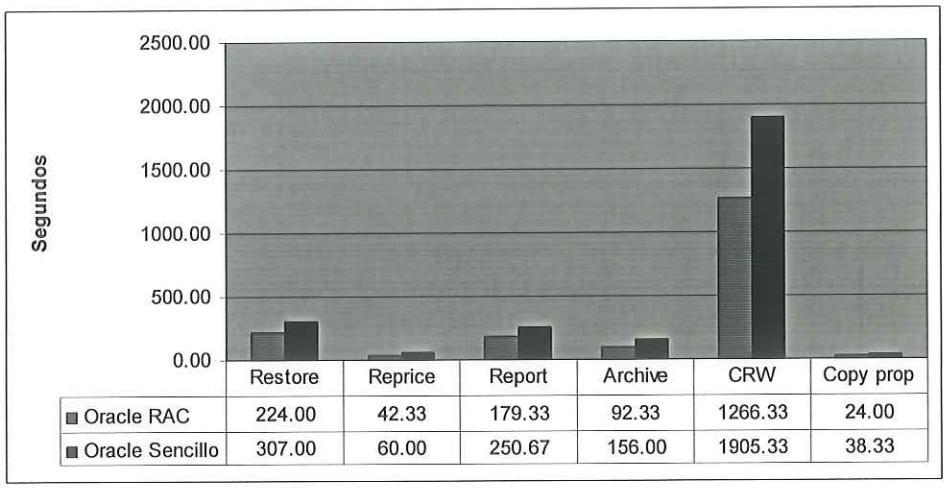
Las pruebas se realizaron con tres usuarios simultáneos: user1, user2 y user3.

Datos:
Propuesta JLR Version 1.
Propuesta JLR Version 2.
Propuesta JLR Version 3.

Cada usuario se conectó desde una máquina cliente a codetest por medio de Remote Desktop Connection (RDC). El Archivo de conexión utilizado por RDC para conectarse fue similar al utilizado en las pruebas con un solo usuario.

1. Dentro de AplicaciónX, cada usuario seleccionó la versión de la propuesta que le correspondió, en este caso fue Versión 1 para el user1, y así sucesivamente. Se ejecutó el proceso "Restore".
2. Se tomó el tiempo de inicio y finalización del proceso. Para este caso, la aplicación escribió, dentro de un archivo de texto en el directorio del usuario, una estampa de tiempo cuando inició y finalizó el proceso.
3. Se documentaron los resultados en una hoja de Excel.
4. Se repitieron los pasos 2-4 para el proceso "Reprice".
5. Se repitieron los pasos 2-4 para el proceso "Report".
6. Se repitieron los pasos 2-4 para el proceso "Archive".
7. Se repitieron los pasos 2-4 para el proceso "CRW".
8. Se repitieron los pasos 2-4 para el proceso "Copy Proposal".

Los resultados de las pruebas fueron comparados con los resultados de esas mismas pruebas ejecutadas en un servidor sencillo de Oracle con los tres usuarios de manera simultánea. Dicho servidor fue de las mismas características tanto de hardware como de software que los servidores del Oracle RAC. Los datos se almacenaron dentro de un directorio del servidor Solaris, compartido a través de NFS, de igual forma que el Oracle RAC. Los resultados se muestran en la Figura IV.3.



	Restore	Reprice	Report	Archive	CRW	Copy prop
Oracle RAC	224.00	42.33	179.33	92.33	1266.33	24.00
Oracle Sencillo	307.00	60.00	250.67	156.00	1905.33	38.33

Figura IV.3. Tiempos de ejecución promedio de tres usuarios simultáneos en Oracle RAC y un servidor sencillo de Oracle, en ambos se utilizó el servidor NFS como almacenamiento de la base de datos.

La Figura IV.3 muestra el promedio del tiempo de ejecución de cada proceso y se observa que Oracle RAC concluye más rápidamente que el servidor sencillo. La razón de esto es que, cuando cada usuario ejecutó un proceso al mismo tiempo, el Oracle RAC distribuyó los procesos entre los nodos que contaban con menos carga de trabajo.

Como estos procesos son paquetes de Oracle, el RAC antes de ejecutar los paquetes, los distribuyó a los nodos que contaban con menos carga de trabajo, y una vez distribuidos, cada nodo inició con la ejecución del paquete. Por lo tanto, cuando se ejecutó un proceso de manera simultanea por los tres usuarios, este proceso se ejecutó en los tres nodos del cluster (un proceso en cada nodo), y no en un solo nodo (los tres procesos en un nodo) como fue en el caso del servidor sencillo de Oracle.

De esta manera, se logró un beneficio en el tiempo de ejecución de los procesos, en algunos casos, de más de 600 segundos de diferencia (Figura IV.3).

La Figura IV.4 muestra el porcentaje de ganancia en tiempo de cada proceso, como se puede apreciar, en algunos casos la ganancia fue de un 40% aproximadamente.

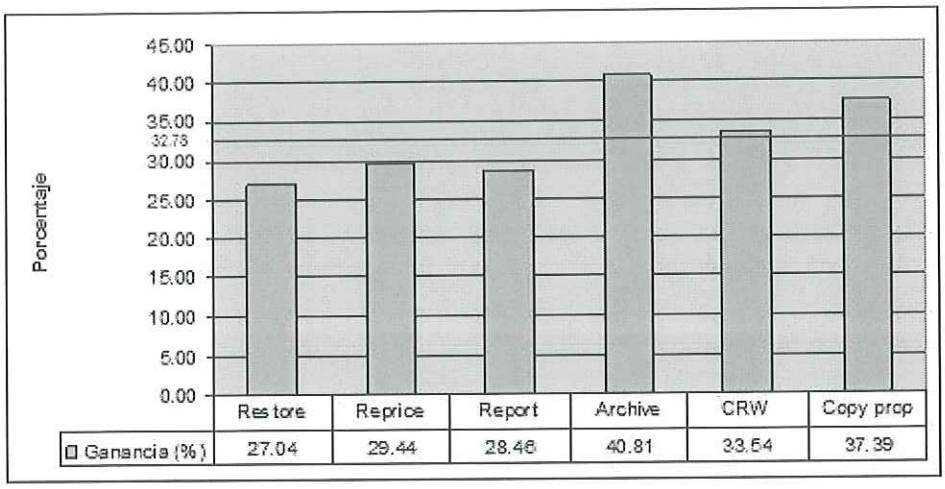


Figura IV.4 Porcentaje de ganancia en tiempo de los procesos con Oracle RAC. La línea en rojo indica el promedio de ganancia de todos los procesos.

En promedio, se obtuvo una ganancia en tiempo con el modelo experimental del Oracle RAC de 32.78% con respecto a un servidor sencillo de Oracle.

Otro de los aspectos interesantes de esta tecnología es su tolerancia a fallas. Cuando se ejecutaron más pruebas en modo multiusuario al cluster, uno de los nodos presentó una falla de energía y se apagó. Esto no ocasionó que se tuviera que para la prueba, sino que los otros nodos continuaron atendiendo las peticiones de otros usuarios a la base de datos. Sin embargo, para que éste usuario volviera a ejecutar esas peticiones, se tuvo que hacer de forma manual, o sea, que el usuario ejecutara de nuevo el proceso donde se quedó.

V. CONCLUSIONES

La tecnología grid de Oracle permite crear un conjunto común de capacidad de procesamiento enlazando varios servidores. De esta manera, los servidores trabajan de forma conjunta para procesar las consultas a la base de datos, evitando así, la saturación de recursos de un solo servidor. Además, aunque se estén utilizando varios servidores, no es necesario realizar algún proceso para la sincronización de datos entre ellos ya que ésta tecnología utiliza un sistema de almacenamiento compartido para hospedar la base de datos, y por lo tanto, los servidores tienen acceso a datos actuales.

Al iniciar las pruebas de AplicacionX en modo multiusuario, se observó que la ejecución de los procesos de cada usuario tardaba más tiempo en finalizar. Un análisis de la utilización del servidor de terminales contra la carga en el servidor de base de datos, nos llevó a conjeturar que los largos tiempos de ejecución eran ocasionados por la saturación de recursos (procesador) del servidor de base de datos Oracle.

Pruebas posteriores mostraron que para lograr una reducción en los tiempos de ejecución de los procesos de AplicacionX en modo multiusuario, era necesario mejorar el rendimiento de la base de datos Oracle.

Esto motivó la implementación de un modelo experimental de Oracle RAC, donde se observó que las consultas de cada usuario eran repartidas entre los nodos del cluster, es decir, las diferentes instancias de Oracle que se encontraban direccionadas a la base de datos de AplicacionX, eran utilizadas por cada nodo del cluster para procesar las consultas a los datos. En éste modelo, se notó que todos los nodos del cluster participaron por igual en el procesamiento de las consultas de los usuarios.

Todo parece indicar que Oracle RAC reparte los procesos de los usuarios entre los nodos que en ese preciso instante tengan menor carga en el procesador. También se observó que, cuando más usuarios ejecutaban sus procesos, el cluster llegó a tener sus nodos

trabajando a su máxima capacidad de procesamiento, registrando picos de más del 95%. Sin embargo, se pudo apreciar que dicha utilización en el procesador de cada nodo era de menor duración que la que se presentó en el servidor sencillo.

Las políticas de calendarización, transparencia y balanceo de carga de trabajo del Oracle RAC, propios de un sistema de grid computacional a nivel departamental, arrojaron resultados que muestran que la implementación de un modelo experimental para manejar la base de datos de la AplicacionX, logró obtener una reducción en tiempo de ejecución con respecto a un servidor sencillo de Oracle, de 32.78%, en promedio.

Sin embargo, también se observó que éste beneficio solo es posible cuando la aplicación es utilizada en modo multiusuario. Cuando las pruebas se realizaron con un solo usuario, el tiempo de respuesta fue ligeramente mayor con respecto al obtenido en un servidor sencillo de Oracle. Suponemos que dicho incremento es ocasionado por la generación e intercambio de los paquetes de administración y monitoreo del Oracle RAC.

Los resultados de las pruebas realizadas, permiten sugerir al ambiente Oracle RAC, como una buena opción para mejorar el rendimiento de bases de datos Oracle que utilicen un esquema multiusuario, sin necesidad de alterar el código de aplicaciones existentes.

GLOSARIO

Benchmarks.
Son aplicaciones o programas los cuales sirven para obtener el rendimiento de los equipos (velocidad de procesamiento).

Calendarizacion.
Referente a la tecnología Grid, es un proceso el cual, elige que host es el más adecuado para ejecutar alguna tarea dependiendo de los requerimientos de la misma.

CPU.
Abreviatura de Central Processing Unit. Unidad central de procesamiento. Es el cerebro de la computadora.

Ducto.
Un ducto o bus en ingles, es un conjunto de cables por lo cuales pasan datos de un lugar de la computadora a otro.

Gigaflop.
Un billón de operaciones de punto faltante por segundo.

Grid.
Malla por su significado en español.

Hardware.
Es todo aquello que podemos tocar, el monitor, el teclado, la computadora en si, (lo que alberga las tarjetas, el disco duro, la unidad de disquete, el procesador, memoria, etc.), la impresora, el ratón (mouse), los cables, conexiones, etc.

Idle.

Estado en el cual el procesador se encuentra activo, pero sin ejecutar alguna operación, por lo tanto se dice que el procesador esta siendo desperdiciado o que esta inactivo.

IPC
Abreviatura de Interprocess Communications. Comunicación entre procesos.

Memoria caché.
Es un tipo de memoria de alta velocidad que es usada para almacenar datos que se utilizan con más frecuencia por algunos programas y de esta forma ganan más velocidad de procesamiento.

Middleware.

Software que conecta dos o más aplicaciones independientes.

MPI

Abreviatura de Mesagge Passing Interface. Interface de paso de mensajes. Comúnmente usado para la parelelización de aplicaciones.

NFS.

Abreviatura de Network File System. Sistema de archivos de red.

RAC

Abreviatura de Real Applications Cluster.

Software.

Todo el hardware que existe no puede funcionar si no hay un programa o programas que hacen que funcione de manera adecuada. Estos programas hacen que los usuarios puedan interactuar con la computadora.

REFERENCIAS

[1] "The Smallpox Research Grid Project", IBM LifeScience Solutions.

[2] "The Physiology of the Grid. An Open Grid Services Architecture for Distributed Systems Integration". Autores: Ian Foster, Carl Kesselman, Jeffrey M. Nick, Steven Tuecke.

[3] "Expert Oracle Database Architecture" Autor: Thomas Kite. Editorial Apress, 2005.

[4] "Oracle Real Application Clusters 10g, the foundation for enterprise grid computing". Oracle Whitepaper. Autor: R. Hietter. Septiembre 2005.

[5] "Oracle and the Grid". Oracle Whitepaper. Autor: Brajesh Goyal. Noviembre 2002. Oracle Corporation.

[6] "Oracle RAC & Grid Tuning with Solid State Disk". Autores: Mike Ault, Donald K. Burleson. Editorial Rampant Books. Enero 2006.

[7] "Oracle 10g Grid & Real Application Clusters". Autores: Mike Ault, Madhu Tumma. Editorial Rampant Books. Mayo 2004.

[8] Microsoft SQL Server Clustering WhitePaper.
http://research.microsoft.com/~gray/SQL_Server_Clustering_Whitepaper.doc

[9] "Oracle RAC". Implementacion de Oracle RAC usando un disco duro externo firewire.
http://www.oracle.com/technology/pub/articles/hunter_rac10gr2.html

[10] "MySQL Cluster, very big and fast databases on commodity hardware". Autor: Alex Aulbach. PHP Conference, Noviembre 2005.

[11] Microsoft Terminal Services. http://www.microsoft.com/windowsserver2003/technologies/terminalservices/default.mspx.

[12] NFS, Network File System. http://www.cs.sfu.ca/CC/401/tiko/lecnotes/ch4.3.pdf

[13] "Testing Oracle 10g Scalability" Autores: Zafar Mahmood, Anthony Fernandez, Bert Scalzo, y Murali Vallath. Dell Power Solutions. Febrero 2006.

[14] "Benchmark Factory for Databases". Página de la compañía. http://www.quest.com/benchmark-factory/

ANEXOS

Anexo 1. Instalación y configuración de los sistemas operativos en los nodos.

Nodo para almacenamiento compartido.

Insertar en el DVD ROM el disco de Solaris 10.

Con el fin de agilizar la instalación:

- Seleccionar que instale todos los paquetes.
- Seleccionar las opciones por default.

En la configuración de red:

Hostname: csnode1-sun

eth0:
- Apagar la opcion [Configure using DHCP]
- Activar la opcion [Activate on boot]
- Direccion IP: 192.168.0.1
- Mascara: 255.255.255.0
- Gateway: Dejarlo en blanco.

Como root, modificar el archivo */etc/hosts*:

```
#
# Internet host table
#
127.0.0.1      localhost
192.168.0.1    csnode1-sun    loghost
192.168.0.2    csnode2-priv
192.168.0.3    csnode3-priv
192.168.0.4    csnode4-priv
```

IMPORTANTE: Crear un usuario "oracle" con el mismo id de usuario y grupo que en los clientes Linux.

```
# mkdir -p /u01/app
# groupadd -g 115 dba
# useradd -u 175 -g 115 -d /u01/app/oracle -s /bin/bash -c "Oracle Software Owner" oracle
# chown -R oracle:dba /u01
# passwd oracle
```

Crear el directorio donde se compartiran los archivos, debe pertenecer al usuario "oracle".

```
# cd /export/home
# mkdir oradata
# chown –R oracle:dba oradata
# chmod –R 755 oradata
```

Exportar el directorio a compartir. Agregar la siguiente linea al archivo */etc/dfs/dfstab*. Dar permisos para que el usuario "root" pueda modificar los archivos.

```
# vi dfstab

#--   Place share(1M) commands here for automatic execution
#     on entering init state 3.
#
#     Issue the command 'svcadm enable network/nfs/server' to
#     run the NFS daemon processes and the share commands, after adding
#     the very first entry to this file.
#
#     share [-F fstype] [ -o options] [-d "<text>"] <pathname> [resource]
#     .e.g,
#     share  -F nfs  -o rw=engineering  -d "home dirs"  /export/home2
share -F nfs -o rw=csnode2-priv:csnode3-priv:csnode4-priv,root=csnode2-priv:csnode3-priv:csnode4-priv -d "export"
/export/home

# exportfs –a
```

El directorio /export/home y todo su contenido será compartido en todos los nodos del cluster.

Instalando Linux en los otros nodos.

Insertar el disco de instalación de CentOS 4.2 Linux.

Con el fin de agilizar la instalación:

- Seleccionar que instale todos los paquetes.
- Deshabilitar el Firewall y SE Linux
- A la hora de particionar el disco, dejar 2GB para el swap.

Primer nodo. Hostname: csnode2

eth0:
- Apagar la opcion [Configure using DHCP]
- Activar la opcion [Activate on boot]
- Direccion IP: 192.168.0.2
- Mascara: 255.255.255.0

eth1:
- Apagar la opcion [Configure using DHCP]
- Activar la opcion [Activate on boot]
- Direccion IP: 10.0.0.21
- Mascara: 255.0.0.0

Segundo nodo. Hostname: csnode3

eth0:
- Apagar la opcion [Configure using DHCP]
- Activar la opcion [Activate on boot]
- Direccion IP: 192.168.0.3
- Mascara:: 255.255.255.0

eth1:
- Apagar la opcion [Configure using DHCP]

- Activar la opcion [Activate on boot]
- Direccion IP: 10.0.0.22
- MAscara: 255.0.0.0

Tercer nodo. Hostname: csnode4

eth0:
- Apagar la opcion [Configure using DHCP]
- Activar la opcion [Activate on boot]
- IP Address: 192.168.0.4
- Mascara: 255.255.255.0

eth1:
- Apagar la opcion [Configure using DHCP]
- Activar la opcion [Activate on boot]
- Direccion IP: 10.0.0.23
- Mascara: 255.0.0.0

Como root, modificar el archivo */etc/hosts*:

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1            localhost.localdomain      localhost

# Servidor NFS  (Solaris 10)
192.168.0.1          csnode1-sun

# Red Publica (eth1)
10.0.0.21      csnode2
10.0.0.22      csnode3
10.0.0.23      csnode4

# Red Privada (eth0) RAC
192.168.0.2    csnode2-priv
192.168.0.3    csnode3-priv
192.168.0.4    csnode4-priv

# IP Virtual Publico -Requerimiento de Oracle RC-
10.0.1.211     csnode2-vip
10.0.1.212     csnode3-vip
10.0.1.213     csnode4-vip
```

Ajuste de parámetros de configuración de red (nodos Linux).

Oracle sugiere que se ajuste el tamaño default y máximo del buffer de envío (SO_SNDBUF) y del buffer de recepción (SO_RCVBUF) a 256KB. Para realizar esto, ejecutar:

```
# su - root

# sysctl -w net.core.rmem_default=262144
net.core.rmem_default = 262144

# sysctl -w net.core.wmem_default=262144
net.core.wmem_default = 262144

# sysctl -w net.core.rmem_max=262144
net.core.rmem_max = 262144

# sysctl -w net.core.wmem_max=262144
net.core.wmem_max = 262144
```

Estos comandos realizan cambios "al vuelo" al sistema. Para que estos cambios sean permanentes (en caso de que se reinicien los nodos), agregar las siguientes líneas al archivo */etc/sysctl.conf* para cada nodo en el cluster:

```
# Default setting in bytes of the socket receive buffer
net.core.rmem_default=262144

# Default setting in bytes of the socket send buffer
net.core.wmem_default=262144

# Maximum socket receive buffer size which may be set by using
# the SO_RCVBUF socket option
net.core.rmem_max=262144

# Maximum socket send buffer size which may be set by using
# the SO_SNDBUF socket option
net.core.wmem_max=262144
```

Linux esta configurado para ejecutar los servicios de Telnet y FTP, pero por default, estos están deshabilitados. Para habilitar estos servicios, entrar como root a cada servidor Linux y ejecutar:

```
# chkconfig telnet on
# service xinetd reload
Reloading configuration: [  OK  ]
```

Para correr el demonio vsftpd, y que pueda ser iniciado desde */etc/init.d/vsftpd*, realizar:

```
# /etc/init.d/vsftpd start
Starting vsftpd for vsftpd: [ OK ]
```

Para hacer que este servicio inicialice o termine al reiniciar el sistema, ejecutar:

```
# ln -s /etc/init.d/vsftpd /etc/rc3.d/S56vsftpd
# ln -s /etc/init.d/vsftpd /etc/rc4.d/S56vsftpd
# ln -s /etc/init.d/vsftpd /etc/rc5.d/S56vsftpd
```

Crear el usuario oracle y sus directorios en todos los nodos del cluster, como root:

```
# mkdir -p /u01/app
# groupadd -g 115 dba
# useradd -u 175 -g 115 -d /u01/app/oracle -s /bin/bash -c "Oracle Software Owner" -p oracle oracle
# chown -R oracle:dba /u01
# passwd oracle
```

Como se puede observar, se le dio el mismo id de usuario y el mismo id de grupo que en el servidor Solaris.

Montar el directorio que exporta el servidor Solaris, en todos los nodos ejecutar como root:

```
# cd /
# mkdir share
# vi /etc/fstab

# This file is edited by fstab-sync - see 'man fstab-sync' for details
/dev/VolGroup00/LogVol00 /                       ext3    defaults        1 1
LABEL=/boot             /boot                   ext3    defaults        1 2
none                    /dev/pts                devpts  gid=5,mode=620  0 0
```

```
none                    /dev/shm                tmpfs   defaults        0 0
none                    /proc                   proc    defaults        0 0
none                    /sys                    sysfs   defaults        0 0
/dev/VolGroup00/LogVol01 swap                   swap    defaults        0 0
# NFS mount line
csnode1-sun:/export/home        /share  nfs     bg,hard,nointr,rsize=32768,wsize=32768,tcp,noac,vers=3,timeo=600        0 0
/dev/hdc                /media/cdrecorder       auto    pamconsole,exec,noauto,managed 0 0

# mount /share
#ls –la /share
drwxr-xr-x  4 root   root  512 Apr 21 17:03 ./
drwxr-xr-x 27 root   root 4096 Apr 21 15:35 ../
drwxr-xr-x  2 root   root 8192 Apr 19 15:01 lost+found/
drwxr-xr-x  5 oracle dba   512 Apr 26 10:54 oradata/
```

NOTA: Las opciones de montaje que son usadas en esta implementación, son las que estan documentadas que trabajan en los dispositivos NAS certificados [7].

IMPORTANTE: Los usuarios root y oracle de cada nodo en el cluster debe poder leer y escribir en el directorio /share/oradata.

Anexo 2. Instalación y configuración del software para Oracle RAC.

Configuración de la cuenta "oracle" en los clientes linux.

Como usuario oracle en todos los nodos del cluster.

NOTA: Es necesario asegurarse de asignar a cada nodo dentro del RAC un ORACLE_SID unico.

csnode2 : ORACLE_SID=csrac21
csnode3: ORACLE_SID=csrac22
csnode4: ORACLE_SID=csrac23

Usar el siguiente archivo *.bash_profile* para la cuenta "oracle":

```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

alias ls="ls -FA"

# User specific environment and startup programs
export ORACLE_BASE=/u01/app/oracle
export ORACLE_HOME=$ORACLE_BASE/product/10.2.0/db_1
export ORA_CRS_HOME=$ORACLE_BASE/product/10.2.0/crs
export ORACLE_PATH=$ORACLE_BASE/common/oracle/sql:.:$ORACLE_HOME/rdbms/admin

# Each RAC node must have a unique ORACLE_SID. (i.e. csrac21, csrac22, csrac23...)
export ORACLE_SID=csrac21

export PATH=.:${PATH}:$HOME/bin:$ORACLE_HOME/bin
export PATH=${PATH}:/usr/bin:/bin:/usr/bin/X11:/usr/local/bin
export PATH=${PATH}:$ORACLE_BASE/common/oracle/bin
export ORACLE_TERM=xterm
export TNS_ADMIN=$ORACLE_HOME/network/admin
export ORA_NLS10=$ORACLE_HOME/nls/data
export LD_LIBRARY_PATH=$ORACLE_HOME/lib
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:$ORACLE_HOME/oracm/lib
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/lib:/usr/lib:/usr/local/lib
export CLASSPATH=$ORACLE_HOME/JRE
export CLASSPATH=${CLASSPATH}:$ORACLE_HOME/jlib
export CLASSPATH=${CLASSPATH}:$ORACLE_HOME/rdbms/jlib
export CLASSPATH=${CLASSPATH}:$ORACLE_HOME/network/jlib
export THREADS_FLAG=native
export TEMP=/tmp
export TMPDIR=/tmp
```

Crear los directorios que seran utilizados para almacenar los dos archivos compartidos por el Oracle Clutsterware y por la base de datos, como root en todos los nodos:

```
$ su -
# mkdir -p /share/oradata/oracrs
# mkdir –p /share/oradata/oradb
# chown -R oracle:dba /share/oradata
```

Ajuste de parámetros del sistema operativo para oracle.

- Determinar los límites para memoria compartida:

```
# ipcs -lm

------ Shared Memory Limits --------
max number of segments = 4096
max seg size (kbytes) = 32768
max total shared memory (kbytes) = 8388608
min seg size (bytes) = 1
```

- Cambiar SHMMAX a 2Gb.

Para determinar el valor de SHMMAX, realizar lo siguiente:

```
# cat /proc/sys/kernel/shmmax
536870912
```

Para cambiar el valor:

```
# sysctl -w kernel.shmmax=2147483648
```

Para hacerlo permanente:

```
# echo "kernel.shmmax=2147483648" >> /etc/sysctl.conf
```

Verificar el valor de SHMMNI:

```
 # cat /proc/sys/kernel/shmmni
4096
```

Verificar SHMALL:

```
# cat /proc/sys/kernel/shmall
2097152
```

- Configurar los semáforos.

Para determinar los límites de los semáforos:

```
# ipcs -ls

------ Semaphore Limits --------
max number of arrays = 128
max semaphores per array = 250
max semaphores system wide = 32000
max ops per semop call = 32
semaphore max value = 32767
```

El siguiente comando también puede ser utilizado:

```
# cat /proc/sys/kernel/sem
250     32000  32      128
```

Para cambiar los límites de los semáforos:

```
# sysctl -w kernel.sem="250 32000 100 128"
```

Para hacerlo permanente:

```
# echo "kernel.sem=250 32000 100 128" >> /etc/sysctl.conf
```

- Configurar el manejo de archivos

Para determinar el número máximo de manejo de archivos para el sistema, ejecutar:

```
# cat /proc/sys/fs/file-max
102563
```

Configurar el manejo de archivos a 65536

```
# sysctl -w fs.file-max=65536
```

Hacerlo permanente:

```
# echo "fs.file-max=65536" >> /etc/sysctl.conf
```

Para consultar el uso actual del manejo de archivos:

```
# cat /proc/sys/fs/file-nr
825     0       65536
```

Verificar la configuracion de ulimit:

```
# ulimit
unlimited
```

- Configurar el módulo del kernel "hangcheck-timer"

Verificar si el módulo está incluido:

```
# find /lib/modules -name "hangcheck-timer.ko"
/lib/modules/2.6.9-22.EL/kernel/drivers/char/hangcheck-timer.ko
```

Para configurar y cargar el módulo:

```
# su -
# echo "options hangcheck-timer hangcheck_tick=30 hangcheck_margin=180" >> /etc/modprobe.conf
```

Cargar el módulo al iniciar el sistema:

```
# echo "/sbin/modprobe hangcheck-timer" >> /etc/rc.local
```

- Configurar los nodos del RAC para el acceso remoto.

Verificar si el paquete rsh esta instalado en todos los nodos.

```
# rpm -q rsh rsh-server
rsh-0.17-25.3
rsh-server-0.17-25.3
```

Para habilitar los servicios “rsh” y “rlogin” en todos los nodos:

```
 # su -
# chkconfig rsh on
# chkconfig rlogin on
# service xinetd reload
Reloading configuration: [  OK  ]
```

Para permitir que el usuario "oracle" sea confiable entre los nodos del RAC:

```
# su -
# touch /etc/hosts.equiv
# chmod 600 /etc/hosts.equiv
# chown root.root /etc/hosts.equiv
```

Agregar todos los nodos del RAC en el archivo */etc/hosts.equiv*:

```
# cat /etc/hosts.equiv
+csnode2 oracle
+csnode3 oracle
+csnode4 oracle
+csnode2-priv oracle
+csnode3-priv oracle
+csnode4-priv oracle
```

Renombrar la versión de rsh de Kerberos para que la versión estándar de rsh sea utilizada.

```
# su -
# which rsh
/usr/kerberos/bin/rsh

# mv /usr/kerberos/bin/rsh /usr/kerberos/bin/rsh.original
# mv /usr/kerberos/bin/rcp /usr/kerberos/bin/rcp.original
# mv /usr/kerberos/bin/rlogin /usr/kerberos/bin/rlogin.original
# which rsh
/usr/bin/rsh
```

Proceso de instalación del Clusterware software.

Entrar como oracle en un nodo del cluster:

```
$ cd ~oracle
$ /u01/app/oracle/orainstall/clusterware/runInstaller
```

<table>
<tr><td>Nombre Pantalla</td><td>Respuesta</td></tr>
<tr><td>Welcome Screen</td><td>Click Next</td></tr>
<tr><td>Specify Inventory directory and credentials</td><td>Aceptar los valores por default:
Inventory directory: /u01/app/oracle/oraInventory
Operating System group name: dba</td></tr>
<tr><td>Specify Home Details</td><td>Dejar los valores por default para el directorio fuente. Cambiar el directorio destino para ORACLE_HOME:
Name: OraCrs10g_home
Location: /u01/app/oracle/product/10.2.0/crs</td></tr>
<tr><td>Product-Specific Prerequisite Checks</td><td>El instalador realizará una serie de pruebas para verificar el sistema.
Click Next para continuar.</td></tr>
<tr><td>Specify Cluster Configuration</td><td>Cluster Name: cscrs2
<table>
<tr><th>Public Node Name</th><th>Private Node Name</th><th>Virtual Node Name</th></tr>
<tr><td>csnode2</td><td>csnode2-priv</td><td>csnode2-vip</td></tr>
<tr><td>csnode3</td><td>csnode3-priv</td><td>csnode3-vip</td></tr>
<tr><td>csnode4</td><td>csnode4-priv</td><td>csnode4-vip</td></tr>
</table></td></tr>
<tr><td>Specify Network Interface Usage</td><td>
<table>
<tr><th>Interface Name</th><th>Subnet</th><th>Interface Type</th></tr>
<tr><td>eth0</td><td>192.168.0.0</td><td>Private</td></tr>
<tr><td>eth1</td><td>10.0.0.0</td><td>Public</td></tr>
</table></td></tr>
<tr><td>Specify OCR Location</td><td>Seleccionar External Redundancy":
Especificar OCR: /share/oradata/oracrs/cmregfile</td></tr>
<tr><td>Specify Voting Disk Location</td><td>Selectcionar"External Redundancy"
Spcificar Voting Disk: /share/oradata/oracrs/cmdiskfile</td></tr>
<tr><td>Summary</td><td>Click Install para iniciar la instalación.</td></tr>
<tr><td>Execute Configuration Scripts</td><td>Después de que la instalación termine, se abrirá una pantalla que indicara que se ejecuten los scripts orainstRoot.sh y root.sh . Abre una nueva consola como root en cada nodo dentro del RAC.
Dentro del directorio /u01/app/oracle/oraInventory ejecuta orainstRoot.sh EN TODOS LOS NODOS del RAC cluster:
Dentro de la nueva consola en cada nodo, modificar el valor de entrada para el CSS miscount de 60 a 360 en el archivo $ORA_CRS_HOME/install/rootconfig como se ve a continuación:
CLSCFG_MISCNT="-misscount 60" to CLSCFG_MISCNT="-misscount 360"
Ahora, dentro del directorio /u01/app/oracle/product/10.2.0/crs ejecutar el archivo root.sh en todos los nodos uno a la vez.
Varios mensajes de precaución pueden salir mientras se ejecuta el script. Estos mensajes se pueden ignorar, sin embargo si sale el mensaje de error The given interface(s), "eth0" is not public. Public interfaces should be used to configure virtual IPs, realizar lo siguiente:

Ejecutar vipca manualmente en otra consola.
$ORA_CRS_HOME/bin/vipca
Cuando el asistente de configuración de VIP (Virtual IP) aparezca, realizar la siguiente configuración:
Welcome: Click Next
Network interfaces: Seleccionar ambas interfaces - eth0 y eth1
Virtual IPs for cluster notes:
Node Name: csnode2</td></tr>
</table>

	IP Alias Name: csnode2-vip **IP Address:** 10.0.1.211 **Subnet Mask:** 255.0.0.0 **Node Name:** csnode3 **IP Alias Name:** csnode3-vip **IP Address:** 10.0.1.212 **Subnet Mask:** 255.0.0.0 **Node Name:** csnode4 **IP Alias Name:** csnode4-vip **IP Address:** 10.0.1.213 **Subnet Mask:** 255.0.0.0 **Summary:** Click Finish **Configuration Assistant Progress Dialog:** Click OK para completar la configuración. **Configuration Results:** Click Exit Regresa al proceso de instalación donde indicaba que se ejecutaran los scripts.
End of installation	Salir del instalador.

Después de la instalación de Oracle Clusterware, algunas pruebas se pueden ejecutar para verificar si la instalación fue exitosa.

Ejecutar los siguientes comandos en todos los nodos del RAC.

Para verificar los nodos del RAC que están disponibles:

```
$ /u01/app/oracle/product/10.2.0/crs/bin/olsnodes -n
csnode2  1
csnode3  2
csnode4  3
```

Para verificar los scripts de arranque de Oracle Clusterware:

```
[root@csnode2 ~]# ls -l /etc/init.d/init.*
-r-xr-xr-x  1 root root  1951 Mar 13 16:01 /etc/init.d/init.crs
-r-xr-xr-x  1 root root  4721 Mar 13 16:01 /etc/init.d/init.crsd
-r-xr-xr-x  1 root root 35401 Mar 13 16:01 /etc/init.d/init.cssd
-r-xr-xr-x  1 root root  3197 Mar 13 16:01 /etc/init.d/init.evmd
```

Proceso de instalación del software para la base de datos Oracle.

Entrar como usuario "oracle" en un nodo:

```
$ cd ~oracle
$ /u01/app/oracle/orainstall/database/runInstaller
```

Nombre de la pantalla	**Respuesta**
Welcome Screen	Click Next
Select Installation Type	En este caso, se seleccionó la opción Enterprise.
Specify Home Details	Seleccionar el directorio para ORACLE_HOME: **Name:** OraDb10g_home1 **Location:** /u01/app/oracle/product/10.2.0/db_1

Specify Hardware Cluster Installation Mode	Seleccionar la opción para instalarse en un cluster y seleccionar todos los nodos disponibles.
Product-Specific Prerequisite Checks	El instalador realizará una serie de pruebas para verificar el sistema. En esta instalación solo falló lo siguiente: Checking for ip_local_port_range=1024 - 65000; found ip_local_port_range=32768 - 61000. Failed Para continuar solo apagar el checkbox "Checking kernel parameters".
Select Database Configuration	Seleccionar la opción: "Install database software only." La base de datos en cluster va a hacer creada mas adelante.
Summary	Click en **Install** para iniciar la instalación.
Root Script Window - Run root.sh	Cuando la instalación finalice, una ventana indicará que se ejecute el script *root.sh*. Este script es necesario ejecutarlo como root en todos los nodos del RAC iniciando en el nodo donde esta el proceso de instalación. Primeramente, abrir una nueva consola como usuario root. Dentro del directorio /u01/app/oracle/product/10.2.0/db_1 ejecutar *root.sh*. Regresar al proceso de instalación donde indicaba que se ejecutara el script.
End of installation	Salir del instalador.

- *Crear el proceso TNS Listener*

El Asistente para la Configuración de Red (NETCA) configurará el TNS listener en un ambiente de cluster en todos los nodos en el cluster.

Desde un nodo, iniciar NETCA como el usuario "oracle", ejecutando:

```
# su - oracle
$ netca &
```

Nombre de la pantalla	**Respuesta**
Select the Type of Oracle Net Services Configuration	Seleccionar Configuración en Cluster
Select the nodes to configure	Seleccionar todos los nodos: csnode2, csnode3, csnode4
Type of Configuration	Seleccionar Listener configuration.
Listener Configuration - Next 6 Screens	Configurar el listener: **What do you want to do:** Add **Listener name:** CSLISTENER2 **Selected protocols:** TCP **Port number:** 1521 **Configure another listener:** No **Listener configuration complete** [Next]
Type of Configuration	Seleccionar configuración de Naming Methods
Naming Methods Configuration	Configuración: **Selected Naming Methods:** Local Naming **Naming Methods configuration complete!** [Next]
Type of Configuration	Salir del NETCA.

Esto va a crear tres escuchadores (uno en cada nodo) CSLISTENER2_CSNODE2, CSLISTENER2_CSNODE3 y CSLISTENER2_CSNODE4.

Anexo 3. Creación de una base de datos Oracle para el RAC.

Para iniciar proceso para crear la base de datos, ejecutar desde un nodo como usuario "oracle":

```
# su - oracle
$ dbca &
```

Nombre Pantalla	Respuesta
Welcome Screen	Selectcionar "Oracle Real Application Clusters database."
Operations	Selectcionar **Create a Database**.
Node Selection	Seleccionar todos los nodos: csnode2, csnode3 y csnode4.
Database Templates	Selectcionar **Custom Database**.
Database Identification	Configurar: **Global Database Name:** csrac2 **SID Prefix:** csrac2
Management Option	Dejar las opciones por default: *"Configure the Database with Enterprise Manager / Use Database Control for Database Management."*
Database Credentials	Claves de las cuentas. SYS -> sys SYSTEM -> appx Todas las otras cuentas -> sys
Storage Options	Seleccionar **use Clustered File System**.
Database File Locations	Seleccionar **All databases files use the same path** Archivos de la base de datos: **/share/oradata/oradb**
Recovery Configuration	Apagar la opción "Specify Flash Recovery Area".
Database Content	Dar valores por default
Database Services	Nombre de la configuración para el servicio de la base de datos: *csractest*.
Initialization Parameters	Dejar valores por default.
Database Storage	Dejar configuración default.
Creation Options	Valores por default.
End of Database Creation	Al final de la instalación, el sistema iniciara todas las instancias de la base de datos en los nodos del RAC. Puede tardar varios minutos.

Cuando el DBCA termine, la base de datos ya esta lista para ser utilizada por todos los nodos dentro del RAC.

Anexo 4

Versión

"Artículo"

de la tesis

Aplicación de la Tecnología Grid en Bases de Datos Oracle

El software de Oracle Real Application Clusters proporciona una plataforma fiable para un grid de base de datos relacionales. Se presenta la implementación de un modelo experimental de ésta tecnología para solucionar los problemas de saturación de recursos en la base de datos Oracle de una aplicación real del tipo financiera sin alteración alguna en su código.

POR JOSÉ LOZANO RIZK

La velocidad en los cambios tecnológicos lleva a las empresas a adaptarse lo más rápido posible para poder seguir compitiendo. Las organizaciones se convierten cada vez más adaptables a esta situación, pero a menudo, la capacidad de respuesta de sus sistemas de información no avanza con la misma rapidez. Al mismo tiempo, estas organizaciones desean obtener una mayor eficiencia de sus sistemas de información y reducir el costo de adquisición y/o actualización de equipo de cómputo.

Los grids computacionales son una nueva arquitectura en las tecnologías de la información que se adapta a las necesidades de cambio de cada organización.

A principios del año 2000, surgieron los grids computacionales, los cuales ofrecen una solución a los problemas mencionados anteriormente, proporcionando una infraestructura de hardware y software que se adapte de manera fácil, rápida y económicamente accesible.

Algunas empresas han contribuido en la evolución de los grids computacionales, desarrollando software que permita el uso de esta tecnología, apoyando proyectos Open-Source y posteriormente ofreciendo sus propias versiones para incursionar en ámbitos diferentes del académico o científico. Tal es el caso de Sun Microsystems y Oracle Corporation, entre otras.

Oracle Corporation es una de las primeras compañías que ofreció una base de datos relacional diseñada para los grids computacionales.

Tecnología Grid en bases de datos Oracle.

Oracle 10g es considerada como la primera infraestructura completa e integral que utiliza cabalmente el potencial de una grid computacional en bases de datos relacionales. Oracle 10g recoge los atributos fundamentales de las siguientes tecnologías computacionales:

Implementar uno a partir de muchos: Se trata de conseguir que un número indeterminado de máquinas funcionen como una sola. Para ello, se

tiene que aplicar un proceso de virtualización en todas las capas del sistema y una zona común donde se almacenen los recursos (resource pooling).

Administrar muchos como si fueran uno: Para conseguirlo se debe contar con un software a la medida y de un sistema de administración centralizado.

Los dos puntos anteriores deben aplicarse a cada elemento del grid: sistemas de almacenamiento, bases de datos, servidores de aplicación y aplicaciones.

La Figura 1 muestra la tecnología grid de Oracle.

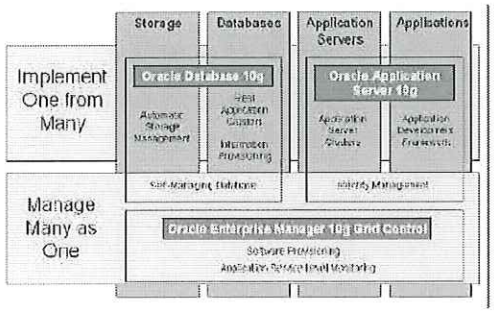


Figura 1. Tecnología grid de Oracle.

La base de datos Oracle 10g añade muchas nuevas características enfocadas al Grid Computing. Mientras otros fabricantes implementan ciertas secciones de una tecnología grid, como por ejemplo una zona de almacenamiento común, Oracle es la primera compañía en ofrecer una base de datos relacional para grid real. Oracle 10g se basa en Real Application Clusters, tecnología que es considerada por Oracle como el corazón de su cómputo en grids.

Oracle Real Application Clusters (RAC).

Oracle RAC es una base de datos en cluster con una arquitectura de cache compartido que sobrepasa las limitaciones de las bases de datos tradicionales (no compartidas) para proporcionar un mayor rendimiento a las aplicaciones que así lo demanden (Figura 2).

Un cluster es un grupo de servidores independientes que trabajan de forma conjunta para obtener un resultado. Los clusters proporcionan una mayor escalabilidad que los sistemas basados en SMP (single symmetric multiprocessor).

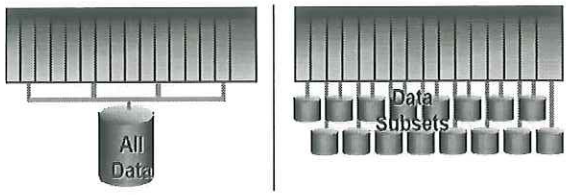


Figura 2. Izquierda: Arquitectura de cache compartido. Varios servidores comparten el mismo cache así como el acceso a un sistema de almacenamiento donde está todo el conjunto de datos. Derecha: Arquitectura no compartida. Cada servidor tiene acceso solamente a un subconjunto de datos de manera local.

En Oracle RAC si un servidor dentro del cluster falla, el acceso a los datos no se pierde y los usuarios pueden continuar trabajando. En caso de que se necesite un mayor poder de procesamiento, se le pueden agregar más nodos al cluster sin alterar la configuración actual.

Oracle RAC permite que varios nodos puedan acceder a la misma base de datos simultáneamente. Esto proporciona tolerancia a fallas y balanceo de carga entre los servidores, en cierta manera, las consultas a la base de datos son distribuidas entre los nodos en el RAC para evitar que se sobrecargue alguno de ellos.

El corazón del Oracle RAC es el sistema de almacenamiento compartido (ver Figura 3). Todos los nodos en el cluster deben tener acceso a los datos, bitácoras, archivos de control y archivos de parámetros.

Los discos de datos deben de estar disponibles para que todos los nodos tengan acceso a la base de datos. Cada nodo tiene sus propios archivos de bitácoras y archivos de control, sin embargo, los otros nodos deben de tener acceso a ellos para poderse recuperar en caso de una falla en el sistema.

Estos archivos deben estar dentro de un sistema de almacenamiento compartido el cual puede ser conformado por un NAS, un SAN, un ASM o un sistema de archivos para clusters.

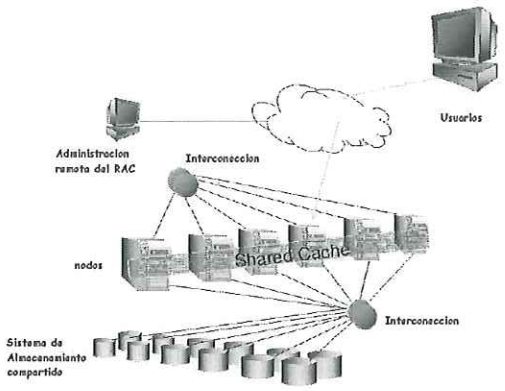


Figura 3. Oracle Real Applications Clusters.

Aplicación real.

La aplicación a ejecutar en el modelo experimental de Oracle RAC, es un sistema de información de tipo financiero el cual utiliza a Oracle como su manejador de base de datos.

Por cuestiones de confidencialidad el nombre de la aplicación fue sustituido por "AplicacionX". Este sistema fue desarrollado para la plataforma Win32 y cuenta con soporte para entornos multiusuarios. Actualmente, AplicacionX es utilizada por compañías de gran prestigio internacional.

Mediante tal aplicación se elaboran propuestas para la construcción de equipo militar y aeroespacial por parte de empresas privadas a gobiernos de países que así lo requieran, en su mayoría, al gobierno de EUA. Los cálculos para obtener los costos de las propuestas que realiza AplicaciónX requieren la mayor precisión ya que un centavo que se este perdiendo o ganando en una pieza, puede repercutir en cientos de miles de dólares tanto para ganancia como para perdida. Además, el tiempo de respuesta en la entrega de propuestas es vital para los clientes de esta aplicación ya que compiten entre ellos para entregar la mejor propuesta en el menor tiempo posible.

AplicacionX posee un alto grado de complejidad en cuestión de programación, y está compuesta por cinco módulos principales. Cada módulo tiene diversas operaciones como: manejo de reportes estándar y con formatos especiales; importar y exportar información en formatos propios así como en estándares abiertos (XML); realizar respaldos de toda la información del esquema de Oracle; restaurar esa información con manejo de duplicados; y el módulo principal y más importante realiza el cálculo para la obtención de costos.

La base de datos puede llegar a consumir varios gigabytes de información, aunque varia dependiendo de cada cliente del producto.

El mercado de AplicacionX está enfocado a pymes y empresas grandes. Los especialistas de estas empresas trabajan de manera colaborativa para la elaboración de sus propuestas. Sin embargo se han presentado algunos problemas cuando ésta es utilizada en el entorno multiusuario con un escenario como el siguiente:

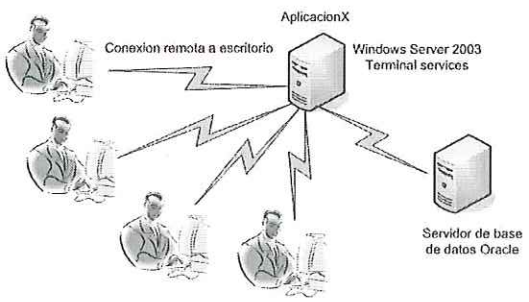


Figura 4. Entorno multiusuario para AplicacionX.

La Figura 4 muestra el escenario más utilizado por las empresas para ejecutar AplicacionX.

- Se instaló AplicacionX dentro de un Servidor de Terminales de Microsoft.
- Se creó el esquema de AplicacionX en un servidor de base de datos Oracle.
- 10 usuarios se conectaron desde su máquina a AplicacionX por medio del paquete "Conexión Remota a Escritorio" de Microsoft.
- Cada usuario ejecutó el mismo proceso* de AplicacionX de manera simultánea, utilizando los mismos datos de entrada.

*El proceso a ejecutar es un paquete que está dentro del esquema de AplicacionX en el servidor de Oracle, el

cual realiza consultas y operaciones aritméticas.

Primeramente, se ejecutó ese proceso por un solo usuario con los mismos datos de entrada, tardó 38 segundos aproximadamente en terminar.

Después, se ejecutó la misma prueba pero con 10 usuarios de manera simultánea. La Figura 5 muestra el porcentaje de CPU utilizado por el servidor de terminales donde están conectados los usuarios. Éste servidor tardó aproximadamente 7 minutos en que todos los usuarios terminaran de ejecutar el proceso de AplicacionX. El primer usuario que terminó el proceso tardó 50 segundos aproximadamente.

Comparando los tiempos tanto del usuario independiente contra el del primer usuario que terminó en el entorno multiusuario, se tiene un incremento en el lapso de ejecución de los procesos de aproximadamente 12 segundos en el entorno multiusuario.

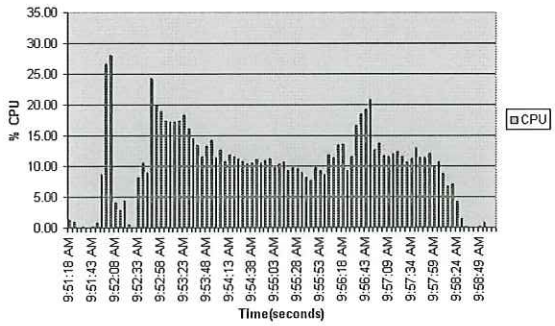


Figura 5. Uso de CPU en el servidor de terminales.

Como se puede observar en la Figura 5, el uso de CPU del servidor de terminales no llego ni al 30%, lo cual indica que no hubo saturación de recursos en ese servidor.

Sin embargo, la Figura 6 muestra el porcentaje de CPU utilizado en el servidor de base de datos Oracle. Como se observa, el servidor presentó picos de más del 95% de uso de CPU durante los 7 minutos que duró la ejecución de los procesos.

Esto presenta dos desventajas:

1) Un usuario tiene que esperar más de tiempo para obtener sus resultados en un entorno multiusuario, del que toma cuando se trata de un solo usuario.

2) El servidor de base de datos Oracle se satura de peticiones al estar en este tipo de escenario multiusuarios, y trae como consecuencia un aumento en el tiempo de ejecución de los procesos de cada usuario.

Además, el servidor de base de datos Oracle no solo es utilizado por AplicacionX sino por otras aplicaciones, por lo tanto, aquellas también se pudieran ver afectadas.

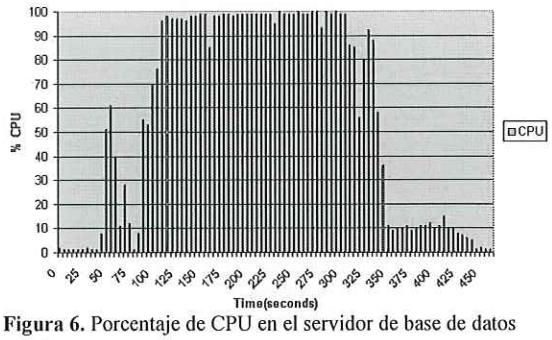


Figura 6. Porcentaje de CPU en el servidor de base de datos Oracle.

Una posible solución al problema de saturación del servidor, sería contar con otro servidor de base de datos Oracle y repartir los usuarios entre ambos servidores.

Sin embargo, al hacerlo, se tienen que estar actualizando los datos en ambos servidores, esto genera más trabajo a los administradores (dba) ya que tendría que existir alguna forma de sincronización entre los servidores; también genera molestias al usuario ya que tendría que configurar su máquina para poderse conectar a un servidor y luego de nuevo configurarla para entrar al otro, además, si él llegara a necesitar cierto dato que esta en otro servidor y no se ha sincronizado, tendría que esperar o solicitarlo al dba.

Un escenario real de esta aplicación es tener más de 1000 usuarios conectados de manera simultánea a AplicacionX, por lo tanto es necesario mejorar el rendimiento en su base de datos Oracle.

Un estudio de escalabilidad realizado por Dell,

muestra el comportamiento del Oracle RAC al forzar al límite la base de datos y al incrementar el número de nodos. Sin embargo, eso no asegura que AplicaciónX podría tener un comportamiento similar ya que la arquitectura de esta aplicación es muy diferente al software utilizado para poner a prueba el rendimiento de la base de datos.

En las pruebas de Dell, en su mayoría, el software realiza transacciones o "querys" a una tabla o un conjunto de tablas pero desde el código de esa aplicación, y en el caso de AplicacionX, los módulos principales realizan llamadas a paquetes que se encuentran incrustados dentro del servidor de Oracle. Los paquetes en Oracle [3] son elementos de programación en lenguaje PL/SQL que permiten combinar procedimientos, funciones, definiciones de tipos de datos y declaraciones de variables en una misma estructura. Se encuentran dentro del servidor de Oracle para que puedan ser utilizados por la aplicación.

Con el fin de lograr un mejor rendimiento en la base de datos de Oracle de AplicacionX, se propuso la implementación de un modelo experimental de Oracle Real Application Clusters.

Modelo Experimental de Oracle RAC.

Para el desarrollo de este modelo se tomó como guía una implementación de Oracle RAC utilizando un cluster de PC's y un disco duro externo firewire como sistema de almacenamiento compartido.

Al tener el Oracle RAC funcionando, se procedió a ejecutar ciertos procesos de AplicacionX. Sin embargo, no fue posible llevarlos a cabo debido a una serie de problemas de comunicación entre los nodos y el disco duro firewire.

El problema se debió particularmente a que el controlador permite el acceso simultáneo al disco duro externo por dos instancias y, razones que no pude determinar, cuando un nodo realizaba una lectura al disco, este tardaba en responder hasta dos segundos en algunos casos.

Esto trajo como consecuencia que al ejecutar el primer proceso de AplicacionX con el disco firewire, la base de datos y los registros del RAC se corrompieron y fue imposible continuar.

Aún más, desde el momento de la instalación de los nodos del cluster, al crear las particiones en el disco duro externo, se notó que tardaban unos segundos en lograr acceder a la información del disco duro externo. Este tipo de retardos no es tolerable en un ambiente de bases de datos relacionales y es por eso que Oracle recomienda el uso de sistemas de almacenamiento compartido que estén certificados por ellos, de preferencia del tipo SAN o NAS.

Debido a estos problemas, fue necesario buscar otra forma de implementar un Oracle RAC. Se tomó la opción de utilizar un servidor del tipo NAS como almacenamiento compartido.

Sin embargo, este tipo de dispositivos certificados involucran una inversión de varios miles de dólares (algunas cotizaciones estuvieron en el rango entre los 20 mil y 40 mil dólares americanos por un sistema con la configuración mínima requerida por los proveedores –entre 5 y 15 TB-) y no se contaba con dicho presupuesto para efectos de la implementación de un modelo experimental.

El porqué de que estos dispositivos sean "certificados" es debido a que garantizan que este tipo de base de datos en cluster no se verá corrompida por algún evento inesperado de E/S, como sucedió con el disco duro firewire. Para el caso del NAS, garantizan que el protocolo NFS que ellos utilizan es menos susceptible a fallas de E/S.

Haciendo caso omiso de lo anterior, se tomó una opción relativamente económica y se consiguió un disco duro externo (similar a un NAS con costo aproximado de 300 dólares americanos) al cual se le puede asignar una dirección IP y compartirlo por medio de la red, ya sea por el protocolo CIFS para Windows o NFS para Unix.

Sin embargo, ni siquiera fue posible instalar el software para el manejo del cluster de Oracle, ya que en el proceso para configurar el "oracle cluster registry" (que es uno de los archivos

compartidos por los nodos) el sistema indicó que las instancias del cluster no podían acceder a ese archivo.

Estudiando la situación, llegué a la conclusión de que el problema radicaba en el protocolo NFS, dado que tanto las opciones de exportación – montaje, como la versión de NFS en esos dispositivos no son las adecuadas para el Oracle RAC, por lo tanto, como NFS fue desarrollado por Sun Microsystems hace ya varios años, se decidió configurar un servidor con sistema operativo Solaris (propietario de Sun) y utilizar el protocolo NFS de ese sistema y así verificar el funcionamiento de Oracle RAC utilizando las opciones de exportación y montaje de NFS en Solaris.

Finalmente, utilizando un servidor Solaris (versión 10) como almacenamiento compartido por NFS, fue posible lograr una instalación satisfactoria de Oracle RAC ya que esta versión si soporta las opciones de exportación y montaje recomendadas para dicha tecnología (Figura 7). Estas opciones son relevantes para el funcionamiento de Oracle RAC ya que permiten que se pueda almacenar la base de datos y los archivos de control en el servidor Solaris y compartirlos por NFS a los demás nodos de Oracle RAC. Cabe mencionar que este modelo no está documentado por Oracle y solo lo recomiendo para pruebas y no para ambientes de producción.

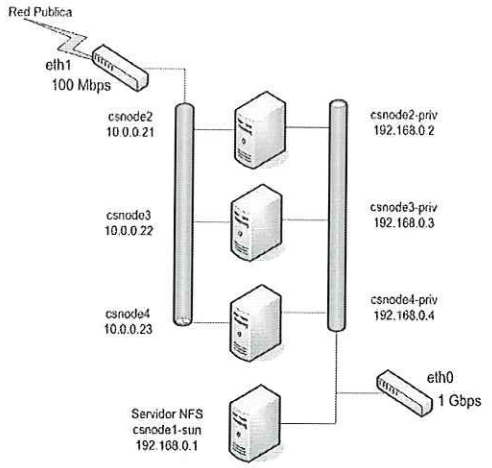


Figura 7. Modelo experimental del Oracle RAC utilizando un servidor NFS en Solaris.

La Figura 7 muestra el modelo experimental del Oracle RAC. Se utilizó un servidor Solaris 10 con NFS para almacenar la base de datos así como los registros del RAC. Según la documentación de Oracle solo es posible utilizar NFS en dispositivos certificados como los de la marca Network Appliance y lo ideal sería que se utilicen dispositivos SAN. Sin embargo, para este modelo, si fue posible utilizar un servidor NFS en Solaris 10 en una PC de 32 bits.

Software.

Se utilizó el siguiente software para el modelo experimental del RAC.

- CentOS Enterprise Linux 4.2 (kernel 2.6.9.22.EL)
- Solaris 10 OS, 32bits x86(PC)
- Oracle Database 10g Release 2 EE.
- Clusterware 10.2.0.1.0.

Hardware.

Para el desarrollo de este modelo, se utilizó el siguiente hardware:

Cluster de PCs con 3 nodos.

- Procesador Intel Pentium 4 a 3.0 GHz.
- Tarjeta madre Intel.
- 1GB DDR SDRAM (a 333MHz).
- 120GB 7200 RPM Disco duro interno (SATA).
- Tarjeta de video Integrada.
- Tarjeta de red integrada de 1 Gbps.
- Una tarjeta de red PCI 10/100.
- DVD ROM.

Sistema de almacenamiento.

- 1 PC con las mismas características que los nodos del cluster.

Red de Interconexión.

- 1 switch de 1 Gbps Dlink.
- 1 switch de 10/100 Linksys.

Los nodos csnode2, csnode3, csnode4 tienen 2 interfaces (eth0, eth1).

La tarjeta de red eth0 está conectada al switch de 1 gbps para la interconexión privada (requerimiento del RAC). Cada nodo en el RAC estará utilizando dicha interfaz para acceder a la base de datos. La tarjeta de red eth1 está conectada al switch de 100 mbps, y es utilizada para acceder a la red pública.
El servidor Solaris 10 solo tiene activa una interfaz de red (1 gbps), está conectada al switch de 1 gbps. No tiene acceso a la red pública.

NOTA: Para mayor información sobre las opciones de montaje de NFS, así como el procedimiento de instalación y configuración del sistema operativo y del software para el Oracle RAC, consultar el capítulo III.2 de la tesis "Aplicación de la Tecnología Grid en Bases de Datos Oracle".

Descripción de la metodología de pruebas para el modelo experimental.

El propósito de esta metodología, es comparar tiempos de ejecución de una aplicación de escritorio multiusuario corriendo en un servidor sencillo de Oracle, con tiempos de esa misma aplicación corriendo en el modelo experimental de Oracle RAC, ambos utilizando los mismos datos de entrada.

Dicha metodología fue desarrollada con el propósito de que pueda ser utilizada para probar cualquier implementación de Oracle RAC, sin importar la arquitectura de hardware y de software que esté siendo usado en el cluster, sino que enfatiza cuáles son los lapsos para la obtención de resultados de una aplicación de escritorio que utilice como manejador de base de datos a un Oracle RAC.

Tanto en el servidor sencillo como en el Oracle RAC, es necesario utilizar los mismos requerimientos, tales como: datos de entrada, versión de la aplicación, número de usuarios simultáneos, mismas características de hardware y software en el equipo, etc.

A continuación se describen los pasos para establecer los requerimientos mínimos:

1.- Instalar el sistema operativo y la base de datos Oracle, la configuración debe de ser la misma para todos los nodos dentro del RAC así como para el servidor sencillo.

2.- Optimizar los parámetros del sistema operativo: Es necesario optimizar, la configuración típica del sistema operativo para que mejor de adapte a Oracle (consultar recomendaciones de Oracle).

3.- Obtener el rendimiento de la red en el cluster: Ejecutar aplicaciones como Iperf para verificar la velocidad de la red privada del Oracle RAC así como del servidor sencillo.

4.- Establecer puntos de referencia para la carga de los nodos: Entrar a los nodos del RAC y documentar la carga de CPU, memoria RAM utilizada y disponible, memoria virtual utilizada y disponible, espacio en disco. Hacer la misma documentación para el servidor sencillo.

A continuación se muestran los pasos para la ejecución de las pruebas:

1.- En una máquina con hardware similar al de los nodos del RAC, instalar Windows Server 2003 y habilitar el servidor de Terminales.

2.- Instalar AplicacionX en el servidor Windows 2003 y configurar el cliente de Oracle para que se pueda comunicar con el Oracle RAC, de igual forma para el servidor sencillo.

3.- Crear la base de datos de prueba en el Oracle RAC y en el servidor sencillo.

4.- Desde el servidor Windows 2003, crear el esquema de AplicaciónX en la base de datos de Oracle RAC y en la base de datos del servidor sencillo. Si ya existe el esquema de aplicacionX, eliminarlo y crearlo de nuevo.

5.- Importar los datos de entrada al esquema de AplicacionX. Los mismos datos deben ser importados al esquema del Oracle RAC y al esquema del servidor sencillo.

6.- Desde una máquina "cliente" conectarse remotamente al servidor Windows 2003, abrir AplicacionX y ejecutar un proceso. Tomar los tiempos de inicio y fin, así como el porcentaje de

uso de CPU en el Oracle RAC y en el servidor sencillo.

7.- Ejecutar el paso 6 con un usuario y luego con varios usuarios simultáneos.

8.- Documentar los resultados.
La Figura 8 muestra la implementación del modelo experimental de Oracle RAC que fue utilizado por AplicacionX.

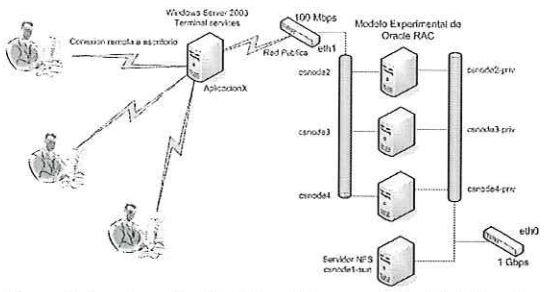


Figura 8. Implementación del modelo experimental de Oracle RAC para AplicacionX.

Pruebas al modelo con un solo usuario.

Un solo usuario ejecutó varios procesos de AplicacionX (uno a la vez) usando el Oracle RAC.

Los resultados de las pruebas anteriores fueron comparados con los resultados de estas mismas pruebas pero ejecutadas en un servidor sencillo de base de datos Oracle.

Éste servidor fue de las mismas características tanto de hardware como de software que los servidores del Oracle RAC. Los datos se almacenaron dentro de un directorio del servidor Solaris, compartido a través de NFS, de igual forma que en Oracle RAC.

Un solo usuario se conectó a AplicacionX por medio del servidor de terminales. Como se muestra en la Figura 9, la base de datos del RAC, en la mayoría de los casos, fue un poco más lenta que el servidor sencillo de Oracle.

La razón de esto fue la carga extra que se originó al contar con una configuración de base de datos en RAC debido a todos los paquetes de monitoreo y administración propios de esta arquitectura.

También se observó que, como los procesos que se ejecutaron en AplicacionX, son paquetes de Oracle, por lo tanto, el Oracle RAC los ejecutó de inicio a fin dentro del mismo nodo, es decir, no distribuyó las sentencias SQL de los paquetes entre los nodos.

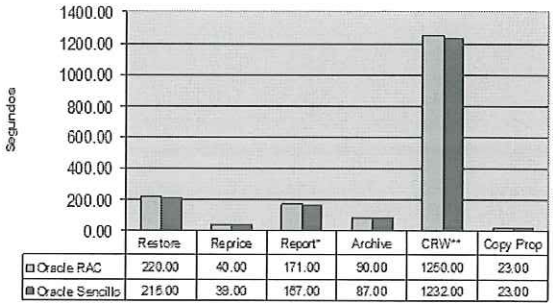


Figura 9. Tiempos de ejecución de un usuario en Oracle RAC y un servidor sencillo de Oracle, en ambos se utilizó el servidor NFS como almacenamiento de la base de datos.

Como consecuencia de esto, un solo usuario que ejecutó algunos procesos de AplicaciónX en el modelo experimental de Oracle RAC no pudo obtener una ganancia en tiempo.

Pruebas al modelo en un entorno multiusuario.

Tres usuarios ejecutaron de manera simultánea ciertos procesos de AplicacionX. Los resultados de las pruebas fueron comparados con los resultados de esas mismas pruebas pero ejecutadas en un servidor de base de datos Oracle sencillo y los tres usuarios de manera simultánea. Dicho servidor fue de las mismas características tanto de hardware como de software que los servidores del Oracle RAC. Los datos se almacenaron dentro de un directorio del servidor Solaris, compartido a través de NFS, de igual forma que el Oracle RAC. Los resultados se muestran en la Figura 10.

La Figura 10 muestra el promedio del tiempo de ejecución de cada proceso y se observa que Oracle RAC concluye más rápidamente que el servidor sencillo. La razón de esto fue que cuando cada usuario ejecutó un proceso al mismo tiempo, el Oracle RAC distribuyó los procesos entre los nodos que contaban con menos carga de trabajo.

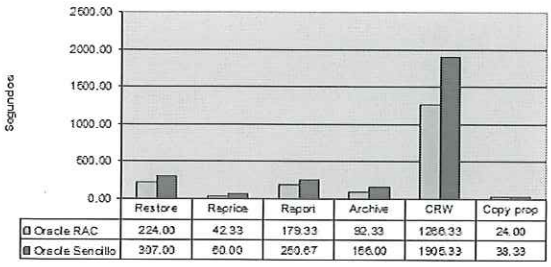


Figura 10. Tiempos de ejecución promedio de tres usuarios simultáneos en Oracle RAC y un servidor sencillo de Oracle, en ambos se utilizó el servidor NFS como almacenamiento de la base de datos.

Como estos procesos son paquetes de Oracle, el RAC antes de ejecutar los paquetes, los distribuyó a los nodos que contaban con menos carga de trabajo, y una vez distribuidos, cada nodo inició con la ejecución del paquete. Por lo tanto, cuando se ejecutó un proceso de manera simultánea por los tres usuarios, éste proceso se ejecutó en los tres nodos del cluster (un proceso en cada nodo), y no en un solo nodo (los tres procesos en un nodo) como fue en el caso del servidor sencillo de Oracle. De esta manera, se logró un beneficio en el tiempo de ejecución de los procesos, en algunos casos, de más de 600 segundos de diferencia (Figura 10).

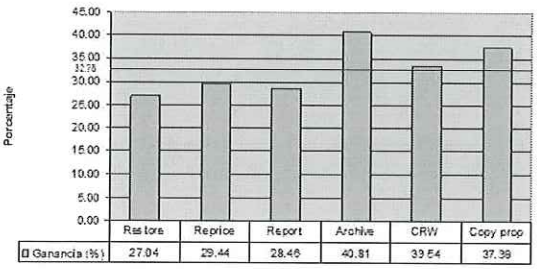


Figura 11. Porcentaje de ganancia en tiempo de los procesos con Oracle RAC. La línea en rojo indica el promedio de ganancia de todos los procesos.

La Figura 11 muestra el porcentaje de ganancia en tiempo de cada proceso, como se puede apreciar, en algunos casos la ganancia fue de un 40% aproximadamente.

En promedio, se obtuvo una ganancia en tiempo con Oracle RAC de 32.78% con respecto a un servidor sencillo de Oracle.

Conclusiones.

La tecnología grid de Oracle permite crear un conjunto común de capacidad de procesamiento enlazando varios servidores. De esta manera, los servidores trabajan de forma conjunta para procesar las consultas a la base de datos, evitando así, la saturación de recursos de un solo servidor. Además, aunque se estén utilizando varios servidores, no es necesario realizar algún proceso para la sincronización de datos entre ellos ya que ésta tecnología utiliza un sistema de almacenamiento compartido para hospedar la base de datos, y por lo tanto, los servidores acceden a datos actuales.

Al iniciar las pruebas de AplicacionX en modo multiusuario, se observó que la ejecución de los procesos de cada usuario tardaba más tiempo en finalizar. Un análisis de la utilización del servidor de terminales contra la carga en el servidor de base de datos, nos llevó a conjeturar que los largos tiempos de ejecución eran ocasionados por la saturación de recursos (procesador) del servidor de base de datos Oracle.

Pruebas posteriores mostraron que para lograr una reducción en los tiempos de ejecución de los procesos de AplicacionX en modo multiusuario, era necesario mejorar el rendimiento de la base de datos Oracle.

Esto motivó la implementación de un modelo experimental del Oracle RAC, donde se observó que las consultas de cada usuario eran repartidas entre los nodos del cluster, es decir, las diferentes instancias de Oracle que se encontraban direccionadas a la base de datos de AplicacionX, eran utilizadas por cada nodo del cluster para procesar las consultas a los datos. En éste modelo, se notó que todos los nodos del cluster participaron por igual en el procesamiento de las consultas de los usuarios.

Todo parece indicar que Oracle RAC reparte los procesos de los usuarios entre los nodos que en ese preciso instante tengan menor carga en el procesador. También se observó que, cuando más usuarios ejecutaban sus procesos, el cluster llegó a tener sus nodos trabajando a su máxima capacidad de procesamiento, registrando picos de más del

95%. Sin embargo, se pudo apreciar que dicha utilización en el procesador de cada nodo era de menor duración que la que se presentó en el servidor sencillo.

Las políticas de calendarización, transparencia y balanceo de carga de trabajo del Oracle RAC, propios de un sistema de grid computacional a nivel departamental, arrojaron resultados que muestran que la implementación de un modelo experimental para manejar la base de datos de la AplicacionX, logró obtener una reducción en tiempo de ejecución con respecto a un servidor sencillo de Oracle, de 32.78%, en promedio.

Sin embargo, también se observó que este beneficio solo es posible cuando la aplicación es utilizada en modo multiusuario. Cuando las pruebas se realizaron con un solo usuario, el tiempo de respuesta fue ligeramente mayor con respecto al obtenido en un servidor sencillo de Oracle. Suponemos que dicho incremento es ocasionado por la generación e intercambio de los paquetes de administración y monitoreo del Oracle RAC.

Los resultados de las pruebas realizadas, permiten sugerir al ambiente Oracle RAC, como una buena opción para mejorar el rendimiento de bases de datos Oracle que utilicen un esquema multiusuario, sin necesidad de alterar el código de aplicaciones existentes.