**Universidad Nacional de Itapúa**

**Facultad de Ingeniería**

**Ingeniería en Informática**



**Trabajo Final de Grado**

**Infraestructuras SSI de Altas Prestaciones aplicadas a Banco de Datos**

**Autores**:

Hugo Armando Sendoa

David Ernesto Krüger

**Director:**

Prof. Ing. Amin Mansuri

**Asesores**:

Prof. Mgter. Lic. Horacio Kuna

Prof. Ing. Sergio Pohlmann

**Encarnación – Paraguay**

**2011**

Universidad Nacional de Itapúa

**Facultad de Ingeniería**

**HOJA DE EVALUACIÓN DE TFG**

**INTEGRANTES MESA EXAMINADORA:**

- ………………………………………

- ………………………………………

- ………………………………………

- ………………………………………

- ………………………………………

**CALIFICACIÓN FINAL:**

Hugo Sendoa: \_\_\_\_\_\_\_\_\_\_ ( )

David Krüger \_\_\_\_\_\_\_\_\_\_ ( )

**ACTA Nº:**

**FECHA:**

**…………………………. ……………………………..**

**Secretaria General Decano**

Dedicatoria

“Este trabajo está dedicado a mis padres, y a una persona muy especial. Porque gracias a ellos he podido completar una etapa de mi vida, poder terminar un ciclo de mi vida profesional, un anhelo imposible sin la inmensa confianza que en mi depositaron.”

*David Krüger*

“Dedico este anhelado trabajo a mis padres y hermanos por ser el origen de mis fuerzas, mis iconos de templanza y mi razón de existir.”

*Hugo Sendoa*

Agradecimientos

Nuestros agradecimientos a todos aquellos que de alguna u otra forma dieron un poco de sí para que la culminación de este trabajo se vuelva posible:

Primeramente queremos agradecer a Dios Todopoderoso por habernos iluminado y guiado siempre, habernos bendecido para llegar hasta donde hemos llegado, paso a paso.

A nuestros familiares, quienes han estado con nosotros siempre apoyando e inspirando.

A nuestro director del Trabajo Final de Grado, el profesor Ing. Amin Mansuri, por orientarnos en todo momento en la realización de este trabajo.

A nuestros asesores, el profesor Ing. Sergio Pohlmann, por su constante apoyo, brindando soluciones simples a problemas complejos. Al profesor Mgter. Lic. Horacio Kuna, quien nos guio con sus consejos y sugerencias desde un principio en este trabajo.

A las profesoras Dra. María Teresa Szostak y Dra. Elena Rosa Szostak, por brindarnos de su tiempo para ayudarnos en la elaboración de este libro.

A nuestros compañeros y amigos, por su apoyo constante y ayuda en todo momento.

Índice General

Dedicatoria 5

Agradecimientos 7

Índice General 9

Índice de Tablas 12

Índice de Figuras 13

Resumen 14

Introducción 15

El Problema 16

Justificacion 17

Objetivos 18

OBJETIVO GENERAL: 18

Objetivos Específicos 18

Resumen 19

1 Marco Teórico 20

1.1 COMPUTACIÓN DE ALTO RENDIMIENTO 20

1.1.1 Computación Paralela y Distribuida 21

1.1.2 Computo en sistemas Concurrentes 22

1.1.3 Arquitectura de procesamiento en paralelo 22

1.1.4 HPC basado en el modelo de Clúster 24

1.1.4.1 Características de un clúster. 25

1.2 Middleware SSI 25

1.2.1 Servicios y Beneficios de Middleware SSI 26

1.2.2 SSI al nivel del sistema operativo 27

1.2.2.1 SCO Unix Ware Non Stop Cluster 28

1.2.2.2 Sun Solares-MC 28

1.2.2.3 GLUnix 29

1.2.2.4 MOSIX 29

1.2.2.5 OpenMosix 30

1.2.2.6 Kerrighed 30

1.2.2.7 OpenSSI 31

1.3 Benchmarks 31

1.3.1 Tipos de Benchmarks 33

1.3.1.1 Micro benchmark: 33

1.3.1.2 Toy Benchmark: 33

1.3.1.3 Benchmark Sintético: 33

1.3.1.4 Benchmark de Aplicación: 34

1.3.2 Aplicación de Benchmarking a Base de datos 34

1.4 Resumen 36

2. MARCO METODOLÓGICO 37

2.1 DISEÑO METODOLÓGICO 37

2.2 DEFINICIÓN DEL ENTORNO TECNOLÓGICO APROPIADO PARA LA INFRAESTRUCTURA DE ALTA PRESTACIONES. 38

2.2.1 Optando por OpenSSI como un clúster de alto desempeño y balanceo de carga 38

2.2.1.1 Requerimientos para la implementación de OpenSSI 40

2.2.1.2 Organización de los nodos 43

2.3 CONSIDERACIONES DE BENCHMARK PARA EL PRESENTE TRABAJO. 43

2.3.1 Benchmarks OLTP 44

Metodología de Precio 49

2.3.2 BENCHMARKS OLAP 54

Metodología de Precio. 58

2.4 Caracterización de Pruebas. 58

2.4.1 Definición de Métricas para la cuantificación de Rendimiento. 58

2.4.1.1 Definición de valores para factor de Comparación. 59

2.4.1.2 Definición de las pruebas. 60

2.4.1.3 Aplicaciones a ser utilizadas 60

2.4.1.4 Descripción de SUT (Sistema bajo Pruebas del ingles System Under Test): 61

2.4.2 Metodología de evaluación con respecto a los componentes 64

2.4.2.1 Objetivo de Prueba 64

2.4.2.2 Tamaño del Banco de Datos 64

2.4.2.3 Periodicidad de las Pruebas 65

2.4.3 Utilización del Banco de Datos 66

2.4.3.1 Pruebas OLTP 66

2.4.3.2 Pruebas OLAP 67

Repetición de Pruebas 67

2.5 Resumen 67

3. RESULTADOS Y ANALISIS 68

3.1 IMPLEMENTACIÓN DEL CLÚSTER OPENSSI 68

3.1.1 Instalación del clúster OpenSSI 68

3.2 Obtención de Resultados y Caracterización de Pruebas 69

3.2.1 Pruebas OLTP: 69

3.2.2 Pruebas OLAP: 70

3.3 Análisis de resultados y gráficos 70

3.3.1 TRANSACCIONES OLTP 70

3.3.1.1 Pruebas realizadas en un periodo de 2 Minutos: 71

3.3.1.2 Pruebas realizadas en un periodo de 20 minutos: 73

3.3.1.3 Pruebas Realizadas durante un periodo de 120 minutos: 75

3.3.1.4 Análisis de los resultados obtenidos durante las pruebas OLTP 77

3.3.2 Transacciones OLAP 81

3.3.2.1 Análisis de los resultados obtenidos durante las pruebas OLAP 83

3.3.3 SÍNTESIS de análisis y resultados 90

3.4 Resumen 91

4 CONCLUSIONES Y RECOMENDACIONES 93

4.1 MEJORAS A LOGRAR 94

4.2 FUTURAS INVESTIGACIONES 94

Bibliografía 96

Anexos 104

Instalación del Sistema Operativo Debian 104

Instalación base de Debian 105

Configuraciones de Debian 106

Instalación del nodo init o master 108

Agregar nodos a la infraestructura. 116

Monitorización visual del clúster OpenSSI 120

Instalación de Postgres SQL 122

Instalación de la suite TPCC-UVA 124

Instalacion del Benchmark TPC-h 125

Script para instanciar el Postgres para pruebas OLTP y OLAP 126

Scripts utilitarios para pruebas OLAP 126

Índice de Tablas

Tabla 1 Tabla general de Resultados OLTP 70

Tabla 2 Pruebas de 2 minutos 71

Tabla 3 Porcentajes de mejora obtenida durante la adición de nodos. 72

Tabla 4 Pruebas realizadas en un periodo de 20 minutos 73

Tabla 5 Porcentaje de mejora obtenida en test de 20 minutos de transacciones OLTP 73

Tabla 6 Pruebas OLTP para periodos de 120 minutos 74

Tabla 7 Mejora Rendimiento OLTP para un periodo de 120 minutos 75

Índice de Figuras

Fig. 1: Diagrama de Esquematización de una Compañía considerada por el estándar TPC (fuente propia) 47

Fig. 2 Diagrama de relaciones entre tablas según el estándar TPC (fuente propia) 48

Fig. 3 Menu de opciones del Bench TPC-C UVA 52

Fig. 4 Diseño lógico de la base de datos 56

Fig. 5 Entidades y Relaciones para TPC-H 57

Fig. 6 Nodo Simple 61

Fig. 7 Cluster de 2 nodos 62

Fig. 8 Cluster de 3 nodos 62

Fig. 9 Cluster de 4 nodos 63

Fig. 10 Pantalla de instalacion 103

Fig. 11 Inicio de sesion 106

Gráfica 1 Rendimiento OLTP 1 Warehouse 71

Gráfica 2 Ganancia obtenida en Pruebas OLTP durante una ejecución de 2 minutos 72

Gráfica 3 Rendimiento de transacciones OLTP realizadas en un periodo de 120 minutos. 73

Gráfica 4 Ganancia obtenida 20 minutos 74

Gráfica 5 Rendimiento de transacciones OLTP para ejecuciones de 120 minutos 75

Gráfica 6 Ganancia obtenida 10 warehouse 76

Gráfica 7 Monitoreo vmstat 77

Gráfica 8 Carga de procesos en el cluster 78

Gráfica 9 Uso del procesador por nodo 79

Gráfica 10 Rendimiento Promedio OLAP 81

Gráfica 11 Ganancia por nodo agregado OLAP 82

Gráfica 12 Monitoreo con vmstat para Consultas OLAP de 1 nodo 83

Gráfica 13 Monitoreo con webView de consultas OLAP de 1 nodo 84

Gráfica 14 Monitoreo con vmstat para Consultas OLAP de 4 nodos 85

Gráfica 15 Monitoreo con webView del nodo 1 de consultas OLAP con 4 nodos 86

Gráfica 16 Monitoreo con webView del nodo 2 de consultas OLAP con 4 nodos 87

Gráfica 17 Monitoreo con webView del nodo 3 de consultas OLAP con 4 nodos 88

Gráfica 18 Monitoreo con webView del nodo 4 de consultas OLAP con 4 nodos 89

Resumen

Los sistemas de altas prestaciones brindan optimización de recursos para ser aprovechados en procesos de forma tal que esta se obtenga de forma eficiente, eficaz y confiable.

El Centro de Investigaciones de la Facultad de Ingeniería de la Universidad Nacional de Itapúa en sus diferentes líneas de investigación realiza estudios con el fin de ser ente generador de conocimiento en pos de la investigación científica. Impulsado por docentes y alumnos llevan adelante proyectos en donde los bancos de datos son algunas de las herramientas más importantes del uso cotidiano.

Consideramos que la necesidad de contar con los elementos que faciliten y promuevan a la investigación científica, además de ser teóricos, deben de ser aplicados y para ello contar con la presencia de una infraestructura que lo permita es gran utilidad.

Introducción

El cómputo de altas prestaciones se encuentra a disposición de toda institución o empresa. La optimización de los componentes internos que conforman estos sistemas permitió que puedan estar concentrados en un mismo espacio físico.

Actualmente el Centro de investigación en Computación de la Facultad de Ingeniería se encuentra ante la necesidad de satisfacer sus demandas computacionales aplicadas a los banco de datos.

El presente Trabajo Final de Grado presenta una propuesta para la satisfacción de las necesidades computacionales del CICFI mediante una metodología de implementación y certificación de una estructura basada en OpenSSI.

Se presenta la aplicación de la infraestructura SSI y la integración con la herramienta de monitoreo certificado por la elaboración de pruebas de rendimiento TPC-C y TPC-H para transacciones OLTP y OLAP respectivamente. Las mismas sirven como apoyo a la comunidad Open Source de OpenSSI debido a que es un ítem faltante para esta línea de investigación.

Se realizo la verificación del comportamiento de la suite TPCC-UVA para estructuras SSI, a su vez se detectaron las falencias detectadas y se proponen optimizaciones posibles de aplicación a la infraestructura para obtener mejores resultados.

De esta manera pretendemos que el presente material sirva como iniciativa a otros alumnos e inculque curiosidad sobre este campo ampliamente aceptado, carente de explotación y que nos apasiona conocido como Altas Prestaciones.

El Problema

Un banco de datos es conocido como una concentración de datos organizados de forma tal que la información se mantenga atómica, fiable y permita un acceso rápido a ella.

Los sistemas de altas prestaciones brindan optimización de recursos para ser aprovechados en procesos de manera a que sean procesados de forma eficiente, eficaz y confiable.

El Centro de Investigaciones de la Facultad de Ingeniería de la Universidad Nacional de Itapúa en sus diferentes líneas de investigación realiza estudios con el fin de ser ente generador de conocimiento en post de la investigación científica. Impulsado por docentes y alumnos llevan adelante proyectos en donde los bancos de datos son algunas de las herramientas más importantes del uso cotidiano.

Consideramos que la necesidad de contar con los elementos que faciliten y promuevan a la investigación científica, además de ser teóricos, deben de ser aplicados y para ello contar con la presencia de una infraestructura que lo permita es altamente necesario. Es por ello que la ejecución de aplicaciones de Banco de Datos en infraestructura de altas prestaciones es un factor ausente y necesario para la comunidad educativa.

En la actualidad el Centro de investigaciones de la Facultad de Ingeniería tiene líneas de altas prestaciones y minería de datos que no disponen de una estructura que brinde un rendimiento optimizando el tiempo de proceso de las operaciones realizadas sobre banco de datos. Si bien muchas de las investigaciones se basan en tendencias tecnológicas, hasta la fecha estas bases teóricas no pueden ser aplicadas sobre entornos distribuidos complementados con prestaciones que garanticen la ejecución de los mismos.

Justificación

La elaboración de alternativas de solución no solo comprende en obtener todas las formas de resolución de la problemática, se basa en obtener un abanico de posibilidades que cumplan con el objetivo y además de ser viables se encuentren al alcance de los facilitadores.

El avance de tecnologías en microprocesadores y la reducción del tamaño del micro conductor permitieron que las empresas logren ordenadores específicos que provean prestaciones cercanas al óptimo buscado a altamente superior a la media de los demás ordenadores. Estos súper ordenadores son conocidos como Mainframes o Servidores dedicados. Definitivamente la incrementación del poder computacional vino acompañado de un incremento en el costo de los materiales y la tecnología utilizada para la elaboración de los mismos, es por ello que el costo de adquirir un centro de cómputo de estas características es considerado como una inversión que debe ser altamente justificada.

En la actualidad el auge de los sistemas distribuidos con tecnologías en las cuales computadoras separadas físicamente, pero unidas por un medio en el cual forman una red, permitieron dividir un problema de tamaño mayor a muchos problemas independientes y de menor tamaño, resolviendo de forma paralela partes de un todo.

La adquisición de estas alternativas que permitirían combatir la problemática relacionada a este trabajo se ven enfrentadas a las limitaciones presupuestales a la cual todo ente estatal se encuentra apegado. De esta manera, la adquisición de un mainframe como ordenador de computo para aplicaciones de banco de datos en el Centro de Investigación en Computación de la Facultad de Ingeniería, se vería obligado a competir con otras insuficiencias que son afrontadas por el ente y sujeto a una presupuesto con un periodo de aceptación y aprobación mayor al necesario para satisfacer las necesidades actuales.

La adquisición de una infraestructura por parte del Centro disminuiría las prestaciones con respecto a un mainframe, aumentaría considerablemente las prestaciones con respecto a un ordenador de capacidades medias y por sobre todo no estaría sujeto a un presupuesto elevado.

Otro ítem a ser considerado durante la elección de un entorno de procesamiento para el Centro de Investigaciones, se debe al nivel de criticidad de los datos que son tratados en las diferentes líneas de investigación con las cuales el CICFI se encuentra trabajando. Esta última razón inclina la balanza por la optar por una infraestructura que provea altas prestaciones al centro, debido a que los estudios que serán la razón de uso del entorno no poseen datos críticos.

Si bien las limitaciones físicas con las cuales cuentan el tipo de infraestructuras distribuida (como trafico de red y uso de espacio físico entre otros) no alcanzan las prestaciones de un servidor dedicado, permite lograr altas prestaciones para problemas complejos en los cuales un ordenador de características medias no tendría capacidad, se necesitan disminuir costos y mantener el nivel de criticidad de datos establecidos.

Objetivos

Objetivo General:

Implementar en el Laboratorio del Centro de Investigación en Computación de la Facultad de Ingeniería una infraestructura de altas prestaciones del tipo *Single System Image* (SSI) para aplicaciones basadas en banco de datos.

Objetivos Específicos

* Plantear el entorno tecnológico sobre el cual se implementará la infraestructura de Alta Prestación.
* Aplicar la infraestructura del tipo SSI
* Integrar la herramienta de monitoreo visual para la infraestructura SSI.
* Evaluar el desempeño de la infraestructura mediante pruebas de rendimiento en aplicaciones de banco de datos.
* Elaborar una metodología de implementación de infraestructura de altas prestaciones SSI para aplicaciones de banco de datos.­­

Resumen

En este capitulo se hace una breve descripción del Centro de Investigación en Computación de la Facultad de Ingeniería y su misión, pues es donde se realizo la implementación del clúster. También se han descripto los problemas a resolver y los objetivos de este Trabajo Final de Grado.

En los capítulos siguientes se han explicado en mayor detalle el marco teorico, el marco metodológico, los resultados y las conclusiones resultantes de este Trabajo Final de Grado.

1 Marco Teórico

1.1 Computación de Alto Rendimiento

La computación de Alto Rendimiento (HPC por sus siglas en ingles High Performance Computing) es un concepto que abarca un amplio espectro de la aún más extensa área de las Ciencias de la Computación, específicamente enfocado a brindar un sistema solido, claro, flexible y estructurado para garantizar el funcionamiento óptimo en la distribución de los recursos.

HPC esta relacionada con la “supercomputación”, y surgió de los principios de este término y trabajo, en ocasiones estos dos términos son utilizados de manera indistinta. Sin embargo, el término supercomputación es usualmente aplicado a infraestructuras con propósitos específicos, un subconjunto especifico de HPC. Las supercomputadoras suelen ser equipos mas poderosos y con sistemas especializados, y por lo tanto muy caros. Un ejemplo podría ser la supercomputadora Cray [BEAS90] o el sistema Thinking Machine [HILLI93] de 1980. Sin embargo, estos sistemas de supercomputación especializados han sido casi completamente reemplazados por clústeres de computadoras básicas no especializadas.

Las computadoras de CPU (por sus siglas en ingles Central Processing Unit) estándar están basados en la arquitectura de Von Neumann [BACK77], y básicamente cuentan con la restricción de poder ejecutar únicamente una instrucción a la vez, una tras otra (ejemplo: procesamiento en serie). De esta manera, la única opción para poder realizar cálculos mas veloces en este tipo de arquitectura era incrementando la velocidad con la que los cálculos eran realizados. Aunque esta velocidad de cálculo ha aumentado de forma constante y rápida desde la invención del circuito integrado, el cual sigue la Ley de Moore [INTE05], muchos cálculos deseables son todavía tan complejos y extensos que incluso en el equipo serial mas rápido, estos requerirían semanas o inclusive años para finalizar el computo. Sin embargo, muchas aplicaciones pueden dividirse en segmentos y estos segmentos resolverse simultáneamente en varios equipos. Cuanto más se puede dividir la aplicación, mayor será la ejecución en paralelo. Todas las aplicaciones HPC involucran la paralelización y ejecución del computo en múltiples entornos distribuidos.

1.1.1 Computación Paralela y Distribuida

La computación paralela permite que varios cálculos sean ejecutados simultáneamente. Cuando los problemas complejos pueden ser segmentados en otros menos complejos que puedan ser resueltos concurrentemente (al mismo tiempo o en paralelo), en consecuencia es reducido el tiempo necesario para resolver dicho problema. Un punto importante en la paralización de aplicaciones es la frecuencia en que los segmentos de código necesitan comunicarse entre si. Algunos cálculos pueden ser divididos en segmentos que únicamente necesitan una comunicación parcial del resultado entre ellos e inclusive pueden no ser necesaria la comunicación [BARN11] .

Cuando las subtareas necesitan comunicarse entre sí para poder llevar a cabo cálculos en paralelo, se produce un nuevo conjunto de variables que interfieren en el procesamiento correcto de estos programas.Específicamente la necesidad de concurrencia y la sincronización introducen potenciales errores de programación, como podría ser la condición de interbloqueos, lo cual se produce cuando dos o más tareas se bloquean entre sí permanentemente teniendo cada una de estas tarea un bloqueo en un recurso que las demás tareas están tratando de acceder [COFF71]. Estos problemas adicionales son el mayor obstáculo para la implementación de computación paralela y obtener una buena prestación paralela de un programa. Otra consideración para la computación paralela es que la sobrecarga de la comunicación puede y hará que se pierda todo el rendimiento en la ejecución simultánea de tareas. El incremento de velocidad de un programa es gobernado por la Ley de Amdahl [AMDA67], que básicamente permite teóricamente calcular la cantidad de aceleración que se puede obtener basada en la cantidad de cálculo que se puede realizar en paralelo. En la actualidad, en donde los procesadores de múltiples núcleos ya son productos básicos, esta teoría requiere una mayor investigación basándose en el rendimiento del chip completo en lugar de centrarse en la eficiencia de cada núcleo [HILL08].

1.1.2 Computo en sistemas Concurrentes

El computo paralelo, para que sea considerado real, necesita de un sistema multiprocesador y de manera ideal debería existir un procesador por proceso a ser ejecutado. Como las necesidades computacionales de los sistema presentan más procesos que procesadores disponibles es necesario presentar una manera de gestionar los procesos tal que el acceso sea lo más optimo posible, a esta gestión de procesos suele denominarse multiproceso [MEND06]. Este ultimo permite la permite lo que conocemos como una ejecución concurrente del sistema.

La concurrencia en los sistemas presenta ventajas que pueden ser aprovechadas por el programa ejecutado como recursos compartidos, descomposición de tareas que agilizan el proceso, modularidad de programas y realizar varias resoluciones de problemas en simultáneo entre otros [CAND07]. Estas formas de aprovechamiento permite el paralelismo sea de acuerdo a la partición de datos conocido como paralelismo particionado o mediante la creación de un grafo que represente el flujo de datos llamado también paralelismo de secuencias [BARR95]. Ejemplo de este último es el modelado mediante redes de Petri [QUIL96]

1.1.3 Arquitectura de procesamiento en paralelo

Una taxonomía útil para organizar los sistemas de procesamiento en paralelo, con el principal motivo de estudiarlas [FLYN72], se basa en las instrucciones de control de secuencias de datos dispuestas en la arquitectura. Está dividida en:

* **SISD** (del inglés Single Instruction and Single Data): Tomando como ejemplo la Arquitectura de Von Neumann, corresponde a una única instrucción operando con una única secuencia de datos.
* **SIMD** (del inglés Single Instruction and Multiple Data): Una misma instrucción aplicada a múltiples secuencias de datos.
* **MISD** (del inglés Multiple Instruction and Single Data): Múltiples Instrucciones simultaneas aplicados a una única secuencia de datos.
* **MIMD** (del inglés Multiple Instruction and Multiple Data): trata sobre las arquitecturas que permiten un múltiple acceso a secuencias de datos por instrucciones múltiples. Tratando de organizar esta última a finales de la década del 90 se procede a subdividirla en dos secciones [DUNC90][HWAN94]:
  + **SMP** (del inglés Symetric MultiProcesing): Arquitecturas que comparten una memoria física única y cuya velocidad disminuye según se incrementan los procesadores.
  + **DMM** (del inglés Distibuited Memory Multiprocessing): Son sistemas en las cuales el procesador posee su propia memoria, evitan problemas de bloqueos y evita el acceso a la memoria física principal.

En el ámbito de HPC, en la mayoría de los casos se trata de sistemas MIMD, aunque los sistemas SIMD, en forma de clústeres de computadores basados en GPU [FAN04], se están convirtiendo en un subconjunto significativo y muy prometedor en la labor en la computación de alto rendimiento. Los sistemas MIMD pueden ser divididos en dos grupos principales, los que comparten una memoria principal (SMP), y aquellos que no comparten una memoria (DMM). Al momento de implementar una infraestructura de Clúster, usualmente se hace uso de ambos tipos de la arquitectura MIMD. Un procesador de memoria compartida, también llamado SMP (del ingles Symmetric multiprocessor), es aquel donde varios CPUs conviven en la misma máquina, un ejemplo de esto son los procesadores múltiples núcleos.

1.1.4 HPC basado en el modelo de Clúster

Existen varias formas de presentar un modelo correspondiente al cómputo en altas prestaciones, como por ejemplo una estructura basada en el Modelo de Clúster. En forma general se define la palabra clúster como un número de elementos del mismo tipo agrupados entre sí. Su definición proviene del idioma ingles [BUYY99] y es utilizado para definir un conjunto de elementos, de igual manera Tomas Sterling lo señalaba en su libro [STER02] en el cual realiza analogías con los sistemas biológicos, organizaciones humanas y estructuras de computadores.

Un clúster brinda flexibilidad de configuración. El número de nodos, la memoria disponible por nodo, el número de procesadores por nodo y la topología de red utilizada para la interconexión de las partes son los parámetros mas considerados a la hora de especificar el poder computacional del clúster. Otra característica muy bien vista por la comunidad tecnológica es la capacidad de adaptación a nuevas tecnologías de hardware y por tanto el incremento de la capacidad de cómputo no se vería ligado de forma unilateral a la compra de un nuevo equipo y de forma condescendiente el desecho del anterior como sucede con los mainframes.

La clave del comportamiento del clúster se basa principalmente en el manejo de los planificadores de plazo [TANE96] y en la capacidad de interconexión entre los componentes de la red conocidos también como nodos del clúster.

1.1.4.1 Características de un clúster.

Es un hecho que la mayoría de los más grandes problemas estratégicos en las cuales se aplica el cálculo numérico en relación con las ciencias de la computación, solo puede ser resueltas mediante una máquina cuya ingeniería permita realizar un computo complejo.

De manera desafortunada para usuarios o empresas el costo de estos ordenadores no compensa la necesidad de adquisición de un mainframe por un factor de costo o tiempo de uso. Es por ello que las alternativas viables para tener acceso a esta disponibilidad fueron siempre aceptadas y estudiadas.

El punto en el cual se unen las necesidades de disponibilidad, rendimiento y bajo costo de forma sinérgica dieron inicio a una tendencia que se expandió de acuerdo a la demanda de las necesidades. Este punto es conocido como Computación en Conjunto (del ingles Clúster Computing) cuya características principales se basan en la unión de una capa física de nodos, una abstracción del manejo del paso de mensajes y la aplicación de tecnologías Distribuidas.

1.2 Middleware SSI

El middleware puede ser considerado como una capa software que reside entre las aplicaciones y los niveles subyacentes como es el sistema operativo, las pilas de protocolo y el hardware [SCHA02]. Históricamente el concepto de middleware no proviene de la investigación académica, sino de la industria.

Una imagen de sistema único SSI (por sus siglas en ingles Single System Image) es una propiedad de un sistema que oculta la distribución y heterogeneidad de sus recursos, y los vuelve disponibles a los usuarios y aplicaciones como un recurso computacional unificado. SSI provee a los usuarios una vista globalizada de los recursos disponibles en el sistema, sin considerar únicamente al nodo al cual esté asociado físicamente dentro del entorno distribuido. Además, SSI puede asegurar la continua operatividad del sistema luego de una falla (alta disponibilidad) así también garantiza que el sistema se cargue uniformemente, provea multiprocesamiento y gestión de los recursos.

Los objetivos de un diseño SSI para un sistema basado en Clúster se enfocan principalmente en la transparencia completa de la gestión de recursos, el rendimiento escalable, y la disponibilidad del sistema para el soporte de las aplicaciones para usuarios [BUYY99]. SSI puede ser definido como una ilusión creada por hardware o software, el cual presenta una colección de recursos como un único recurso mucho más poderoso [PFIS98].

1.2.1 Servicios y Beneficios de Middleware SSI

Los principales servicios de una imagen de sistema único SSI incluyen los siguientes [BUYY99]:

* Proporciona una visión simple y directa de todos los recursos y actividades del sistema desde cualquier nodo del clúster.
* Libera al usuario final de tener que saber en que parte del clúster se ejecutará la aplicación.
* Permite el uso de los recursos de manera transparente con independencia de su ubicación física.
* Permite el trabajo del usuario por medio de una interfaz amigable y permite al administrador gestionar al clúster como una sola identidad.
* Ofrece la misma sintaxis de comandos que los de otros sistemas y por lo tanto reduce el riesgo de errores de operaciones y con este resultado los usuarios finales observan un mayor rendimiento, fiabilidad y mayor disponibilidad del sistema.
* Permite centralizar/descentralizar la gestión del sistema y el control para evitar la necesidad de administradores expertos para la gestión del sistema.
* Simplifica de gran manera la gestión del sistema y reduce así el costo de propiedad.
* Proporciona la comunicación de mensajes sin dependencia de localización
* Beneficia a los desarrolladores de sistemas en la reducción del tiempo, esfuerzo y conocimientos necesarios para la realización de tareas que permiten al personal actual manejar grandes o más complejos sistemas.
* Promueve el desarrollo de herramientas estándares y servicios.

Un buen SSI se obtiene usualmente mediante la cooperación entre todos estos niveles, un nivel más bajo pueden simplificar la implementación de un ser superior.

1.2.2 SSI al nivel del sistema operativo

Los sistemas operativos de los Clúster soportan una eficiente ejecución de aplicaciones paralelas en entorno distribuido con aplicaciones secuenciales. El objetivo es reunir los recursos en un clúster para proporcionar un mejor rendimiento tanto para las aplicaciones secuenciales y paralelas.

Para lograr este objetivo, el sistema operativo debe de ser compatible con la programación planificada en masa de algoritmos paralelos [SCHE98], identificación de los recursos ociosos en el sistema, tales como procesadores, memoria y redes; además ofrecen acceso globalizado a ellos. Debe soportar de manera optima el proceso de migración para proporcionar equilibrio de carga dinámica, así como la comunicación entre procesos rápidos, tanto par alas aplicaciones del sistema y de nivel de usuario. El sistema operativo debe asegurarse de que estas características estén disponibles para el usuario sin la necesidad de llamadas adicionales al sistema.

A continuación se listan los sistemas operativos más representativos que soportan SSI al nivel de kernel:

1.2.2.1 SCO Unix Ware Non Stop Cluster

Es el software de alta disponibilidad de SCO. Amplía significativamente el soporte por hardware, por lo que es más fácil y menos costoso de implementar el software de clustering mas avanzado para sistemas Intel. Es una extensión para el sistema operativo UnixWare en el que todas las aplicaciones se ejecutan de manera eficaz y fiable dentro de un entorno SSI, el cual elimina toda la gestión de carga. Cuenta con IP estándar como la interconexión, eliminando la necesidad de algún hardware propietario. La arquitectura clúster de Unix Ware Non Stop ofrece soporte integrado para aplicaciones de conmutación por error mediante un enfoque de *n + 1*. Con este enfoque, la copia de seguridad de la aplicación puede ser reiniciada en cualquiera de los varios nodos dentro del clúster. Esto permite que un nodo actúe como un nodo de copia de seguridad para los demás nodos dentro del clúster [WALK99].

1.2.2.2 Sun Solares-MC

Es una extensión prototipo de un kernel Solaris de nodo único. Proporciona imagen de sistema único y alta disponibilidad al nivel del kernel. Solaris MC esta implementado mediante técnicas de orientación a objetos. Utiliza de manera amplia el lenguaje de programación orientado a objetos de C++, el modelo estándar de objeto y el lenguaje de definición de interfaz de COBRA. Solaris MC utiliza un sistema de archivos global llamado Proxy del Sistema de Archivos PXFS (las siglas en ingles de Proxy FileSystem). Las características principales incluyen la imagen de sistema único, la semántica coherente y de alto rendimiento. El PXFS hace que los accesos a ficheros se vuelva transparente para los procesos. PXFS logra una imagen de sistema único por medio de la interceptación de las operaciones de accesos a archivos a nivel de la interfaz vnode/VFS (WritingFileSystem, instancias para la operatividad de los FileSystems). Solaris MC asegura que las aplicaciones de red no necesiten ser modificados y observen la misma conectividad de red, independiente de en que nodo se este ejecutando la aplicación [MATE95].

1.2.2.3 GLUnix

Otra alternativa disponible para que el sistema operativo soporte SSI es la implementación de una capa superior en el sistema operativo existente, la cual realiza las asignaciones globales de los recursos. Este es el enfoque seguido por GLUnix de Berkeley. Esta estrategia vuelve portable al sistema operativo y reduce el tiempo de desarrollo. GLUnix es una capa del sistema operativo diseñado para proveer soporte en la ejecución remota transparente, trabajos paralelos y secuenciales, balanceo de carga y compatibilidad con versiones anteriores de binarios de aplicaciones existentes. GLUnix esta completamente implementado en el nivel de usuario y no necesita modificación en el kernel, por lo que se vuelve mas fácil de implementar. Las principales características proporcionadas por GLUnix incluyen técnicas de planificación en paralelo de algoritmos paralelos; detección de recursos ociosos, proceso de migración, y balanceo de carga; comunicación rápida a nivel de usuario, y soporte de disponibilidad [GHOR98].

1.2.2.4 MOSIX

Mosix es una extensión del kernel de Linux que permite ejecutar aplicaciones no paralelizadas en un Clúster. Una de las posibilidades de MOSIX es la migración de procesos, que permite trasladar los procesos de nodo en nodo. Mosix opera de manera inadvertida y sus operaciones son transparentes para las aplicaciones. Esto significa que se pueden ejecutar aplicaciones secuenciales y paralelas al igual que lo haría un SMP (Sistema de Multiprocesamiento simétrico, Symmentric multiprocessing). No es necesario prestar atención en donde el proceso se está ejecutando o lo que otros usuarios podrían estar realizando. Poco después de que un nuevo proceso haya sido creado, Mosix intenta asignarlo al mejor nodo disponible en ese momento. Luego de esto, Mosix continua monitoreando el nuevo proceso, así como también los otros procesos existentes, evaluara los nodos para maximizar el rendimiento global. Todo esto es realizado sin necesidad de alterar la interfaz de Linux. Esto significa que todos los procesos pueden ser monitoreados y controlados como si estuviesen ejecutándose en un nodo en particular. La última versión de Mosix, llamada MOSIX2 es compatible con Linux 2.6. MOSIX2 es implementada como una capa virtualizada del sistema operativo, lo que provee al usuario y a las aplicaciones una imagen de sistema único SSI [BARA98].

1.2.2.5 OpenMosix

El sistema OpenMosix está basado en Mosix, la principal diferencia, se encuentra en su licencia GPL. OpenMosix es un conjunto de parches al kernel y unas utilidades y bibliotecas de área de usuario que permiten tener un sistema SSI completo para Linux. Al estar basado en el código de MOSIX, comparte algunas de sus características y limitaciones. OpenMosix hace uso del parche de Rik van Riel de mapeado inverso de memoria, que permite que el proceso que consume más recursos de OpenMosix pase de tener una complejidad computacional de *O(n)* a una complejidad *k*. En la práctica, elimina una de las partes del código de OpenMosix que pueden consumir una cantidad apreciable de procesador. OpenMosix fue lanzado como un parche para el kernel de Linux, pero también estuvo disponible en Live CDs especializados. El desarrollo de OpenMosix ha sido detenido por sus desarrolladores, pero el proyecto LinuxPMI continúa su desarrollo por medio del código OpenMosix [LOTTI05].

1.2.2.6 Kerrighed

Kerrighed es el resultado de un proyecto de investigación iniciado en 1999. Su objetivo es presentar un único SMP (Sistema de Multiprocesamiento simétrico, Symmentricmultiprocessing) por encima del clúster. Kerrighed se compone de un conjunto de servicios distribuidos del kernel a cargo de la gestión general de los recursos del cluster [LOTTI05][ MORI04].

1.2.2.7 OpenSSI

OpenSSI surge a inicios del año 2001, esta basado en los proyectos Non-Stop Cluster de UnixWare [WALK99]. Bruce Walker, director del proyecto y principal desarrollador de OpenSSI, demarco claramente que el objetivo era crear una plataforma para poder integrar tecnologías de clúster, con código abierto. La última versión de OpenSSI dispone de varios archivos del sistema y manejo de sistemas de disco en código abierto, como por ejemplo GFS, OpenGFS, Lustre, OCFS, DRBD, también integra un mecanismo de bloqueo distribuido (OpenDLM) y una política para obtener balanceo de carga, siendo esta una característica importante heredada de Mosix. OpenSSI permite el balanceo de carga de un clúster dinámicamente mediante el uso de migración de procesos, otra característica también heredada de Mosix. El modulo de migración de procesos de OpenSSI utiliza un mecanismo de acceso al recurso remoto manteniendo los recursos de acceso del sistema bloqueados para que estos no puedan ser migrados. Este mecanismo es usado principalmente en IPC (interfaz de la tarjeta de red), CFS (sistema de archivos del clúster) y también para algunas llamadas al sistema [BARA98].

1.3 Benchmarks

Toda infraestructura o sistema debe ser evaluado para conocer las ventajas y/o inconvenientes que puedan presentarse. Para ello se pretende cuantificar los resultados obtenidos de modo a comparar el rendimiento. Existen varias formas de elaborar una evaluación de rendimiento y la tendencia es evaluar los elementos bajo situaciones ideales. En este capítulo se presentan los conceptos básicos y los resultados obtenidos, así como la metodología aplicada para la obtención de estos.

Se denomina benchmark al o los procesos que son ejecutados en una máquina para proveer una carga de trabajo significativa con el fin de medir el rendimiento. Existen diferentes formas de realizar un benchmark, de los cuales podemos destacar a las generadas por una aplicación, función o procedimiento específico  [DONG83] , la ejecución de un test iterativo de una subrutina [BAIL86] o programas sintéticos que analizan el rendimiento individual de cada componente en un ambiente real, brindando una noción del comportamiento  [BOUT06].

Los benchmark miden el rendimiento y para ello deben satisfacer características consideradas como necesarias en el performance [BAIL86] como:

**Capacidad de cálculo del procesador:**

Independientemente, cada ordenador posee un procesador, cuyas características propias son incrementadas al formar parte de un conglomerado debido a la distribución de carga de trabajo, no obstante ante excesivas cargas el microprocesador debe responder de forma óptima. Este debe ser contrastado con las especificaciones técnicas del hardware.  En la actualidad existen muchas herramientas que nos ayudan a medir estas capacidades mediante operaciones matemáticas [DONG95][NUMR07]

Velocidad de I/O (del ingles Input/Output o entrada y salida):

En el común de las situaciones los sistemas que brindan rendimiento, trabajan con operaciones que requiere un espacio de memoria adicional debido a la complejidad de operaciones que dan como resultante la situación final, ejemplo de ello son las operaciones conocidas como Procesos de Transacción en Línea o OLTP (por sus siglas del Ingles OnLine Transaction Processing) ampliamente utilizadas en el comercio electrónico y transacciones bancarias [CHEN10].

La comparación de estas características medibles con respecto a otros valores brinda la percepción de mejor o peor  rendimiento respecto al punto comparado.

La aplicación de benchmark  se encuentra ligada a obtener métricas de software, infraestructura o un conjunto de ellas caracterizadas por las unidades que son consideradas para obtener un valor cuantitativo. Es por ello que dependiendo del uso dado, pueden variar las métricas o puntos de referencias establecidos por un estándar.

Cuando las pruebas de test son realizadas en su totalidad, provee información que debe ser estudiada. El análisis de esta permite conocer las características y el comportamiento de los componentes ante situaciones similares. Este proceso es conocido como caracterización de pruebas. [GARC08].

1.3.1 Tipos de Benchmarks

De manera a clasificar los benchmark se encuentran:

1.3.1.1 Micro benchmark:

Sentencia o conjunto de instrucción dedicada a la evaluación de un solo componente del sistema [LILJ05]. Generalmente, al ser tan pequeño, cabe en la memoria y presenta evaluaciones erróneas debido al cacheo.[ BERS92][ LIU04]

1.3.1.2 Toy Benchmark:

Son programas completos que realizan operaciones de manera dispersa, si este corresponde a una sección del núcleo del programa en un bucle se lo llama Program Kernels. Sus limitaciones de ejecución no son consideradas como test completos y deben ser complementados [LILJ05].

1.3.1.3 Benchmark Sintético:

Es un conjunto de aplicaciones en un solo programa, utilizado para medir partes de un sistema. Trabajan de manera directa con la sección a ser tratada y, contrariamente a los benchmark de aplicación, no reflejan un trabajo real debido a que trabaja de forma aislada, demostrando la situación ideal del objeto o sección del sistema en observación. [CHEN93][CURN76]

1.3.1.4 Benchmark de Aplicación:

Provee un conjunto de programas o instrucciones estándares que se encargan de similar una carga de trabajo completa, de manera tal como el sistema se comportaría ante una situación cercana a la realidad. Este tipo de benchmark provee una medida de variables durante su ejecución y por ende es importante aislar estos elementos que son considerados como objeto de estudio [LILJ05].

1.3.2 Aplicación de Benchmarking a Base de datos

La aplicación de Benchmark a Base de datos es un área en el cual se desarrollan varias investigaciones debido a su importancia en incrementar las prestaciones de las operaciones diarias de negocios (conocido como Business Operation), así como también en las proyecciones de mercado (o Business Intelligence).  La gestión de las múltiples transacciones pequeñas bajo el esquema de OLTP,  en contraste de las complejas consultas históricas utilizadas por el esquema OLAP (por sus siglas del ingles On Line Analytical Processing o Procesamiento analítico en línea) permiten tener un amplio panorama en todas las operaciones comúnmente utilizadas en la administración de datos.

Si la aplicación de un benchmark es dada a una base de datos se deben tener en cuenta los criterios de:

**Capacidad de cálculo:**

Específicamente las operaciones de punto flotante son las más resaltantes debido a su complejidad.

**Interbloqueo de transacciones:**

El tiempo de bloqueo entre operaciones que transaccionan en un mismo instante  influye considerablemente en cargas de trabajos de magnitud.

**Velocidad de Acceso a Disco (I/O):**

Debido a que se manejan datos históricos y por la magnitud de estos no pueden ser alojados en memoria dinámica y por ello deben ser alojados en medios de almacenamientos. Para datos del esquema OLTP suelen manejarse de manera interna, en cambio para el esquema OLAP los datos pueden ser alojados externamente a la infraestructura principal [SZEP11].

Se considera como objeto de validación la aplicación de un benchmark al banco de datos en diferentes situaciones de manera a verificar el comportamiento de la Base de Datos.

El resultado de estas pruebas define el comportamiento de la base de datos en diferentes escenarios contrastándolos con la infraestructura SSI.

Necesariamente la verificación de una base de datos en busca de rendimiento se debe realizar emulando el trabajo regular en situaciones que se presentan de manera frecuente, es por ello que todo benchmark aplicado debe realizar sentencias SQL sobre la estructura en cuestión.

Varias organizaciones realizaron una definición para ser considerada como estándar y proveen formas de medir y reportar los resultados arrojados. Entre ellas podemos encontrar los estándares como SPEC [SPEC11], TPC [TPCC11], Perfect Club [BERR89], The Open Source Database Benchmark [OSDB10]*,* The NAS Parallel Benchmark[NAPB10], HPL [HPLB08]  y otros.

1.4 Resumen

En este capitulo se describieron conceptos fundamentales detrás del computo con clústeres y se da una respuesta a la pregunta: ¿Por qué emplear Clústeres? También se cubrieron algunos conceptos básicos implicados en computo paralelo, optimización de programas para evaluar el rendimiento de la infraestructura, y de tecnologías SSI como alternativas a un clúster de altas prestaciones.

2. MARCO METODOLÓGICO

2.1 Diseño Metodológico

Basados en la estructura y procedimientos planteados según la finalidad de este trabajo final de grado se realizó una investigación aplicada tecnológica, a fin de generar nuevos conocimientos y brindar un producto a la institución, llevada a cabo mediante estrategias metodológicas fueron de tipo cuantitativas, orientando al resultado, buscando obtener datos sólidos y repetibles, generalizables, y particulares. Por su objetivo este trabajo final de grado fue descriptivo a fin de ir detallando los procesos en los cuales se fue trabajando.

Los procedimientos adoptados a fin de ir estructurando sistemáticamente nuestra estrategia de trabajo fueron de la siguiente manera:

Mediante actividades de refinamiento de la investigación documental orientada a la identificación de trabajos previos vinculados a la computación de altas prestaciones y su impacto actual en el entorno académico, fueron:

1. Estudio e implementación del entorno tecnológico apropiado para la infraestructura de Alta Prestación.
2. Estudio e identificación de técnicas de validación de los procedimientos desarrollados una vez integradas las herramientas a fin de determinar la capacidad máxima a soportar por parte de la infraestructura sin perder rendimiento.
3. Implementación y adaptación de una aplicación grafica, basada en View Web, que permita el monitoreo de recursos de la infraestructura.
4. Elaboración, implementación y resumen de pruebas de rendimiento a la Infraestructura.
5. Redacción del informe del trabajo final de grado realizada mediante los procesos de investigación expuestos anteriormente.

2.2 definición del entorno tecnológico apropiado para la infraestructura de Altas Prestaciones

De manera de afrontar esta problemática se busco un punto de inflexión en donde se encuentra la brecha de la relación costo – rendimiento, brindando servicios de manera transparente y cuyo resultado desemboca en las tecnologías distribuidas.

Por los antecedentes, la propuesta de proveer al Centro de Investigación en Computación de la Facultad de Ingeniería un entorno distribuido de bajo costo para procesos de Banco de Datos, se relaciona directamente con la unión de ordenadores físicos disponibles en el ente coordinados mediante una interconexión física y abstrayendo el paso de mensajes .

La utilización de entornos distribuidos de las características anteriormente mencionada requirió en la mayoría de los casos alterar la fuente de las operaciones, adaptándolos para el cálculo distribuido, es por ello que la propuesta de utilizar una capa que abstrae esa adaptación es incluida en esta propuesta.

2.2.1 Optando por OpenSSI como un clúster de alto desempeño y balanceo de carga

En los clústeres SSI de alto rendimiento los programas no tienen por qué notar que se están ejecutando en uno de ellos. El único requisito es que los programas desplieguen múltiples procesos, los cuales se asignan de forma transparente entre los ordenadores del clúster. Aunque los clústeres SSI no son tan escalables como otros tipos, ofrecen sin embargo la significativa ventaja de que los programas que se ejecutan en el sistema no tienen que conocer la existencia del clúster en el que se estén ejecutando. Otras plataformas de alto rendimiento basadas en clústeres requieren que el código fuente de los programas contengan código propio del clúster, o al menos que el programa se enlace con alguna librería de este. OpenSSI es una solución de clúster SSI extensa de código abierto para Linux basada en la plataforma clúster NonStop de HP. NonStop deriva de Locus, que se desarrollo en los ochenta.

OpenSSI puede repartir los procesos de forma transparente entre múltiples máquinas, característica conocida como nivelado de carga [WALK05]. Otras plataformas de clúster SSI para Linux capaces de realizar el nivelado de carga son OpenMosix y Kerrighed. OpenMosix es el clúster SSI mas popular de Linux. En julio del 2007 se anuncio que el proyecto OpenMosix terminaría en marzo de 2008. Kerrighed es relativamente nuevo, encontrándose actualmente en una fase de desarrollo rápido.

OpenSSI monitoriza constantemente la carga en los ordenadores del clúster y migra automáticamente los procesos entre los nodos. Este sistema es capaz de migrar una aplicación multihilo, pero no es capaz de migrar hilos individualmente. Un proceso migrado puede continuar con las mismas operaciones que estaba realizando en la máquina original, puede leer y escribir en los mismos ficheros o dispositivos e incluso puede continuar una comunicación interproceso (IPC) sobre un socket.

La mayoría de las aplicaciones se ejecutaron en OpenSSI sin ninguna modificación, con algunas excepciones, tal y como se indica en la página de ayuda del comando *migrate*. OpenSSI también proporciona una librería (API) que los programadores pueden utilizar para controlar el clúster.

Se sabe que OpenMosix es el sistema de clúster más conocido y usado, sin embargo, aunque es difícil luchar contra un sistema plenamente acogido, desarrollado y mejorado por muchísima gente de diferente índole, OpenSSI, con cada actualización de Kernel nueva va ganando más adeptos y haciéndose un hueco entre el mundo de la supercomputación.

2.2.1.1 Requerimientos para la implementación de OpenSSI

***Requerimientos de Hardware***

La elección del hardware a utilizar en un clúster puede definirse primeramente en base a conocer la magnitud del problema o problemas que se quieren resolver, el hardware que puede adquirirse en el mercado, los recursos económicos y humanos con los que se cuentan, tanto para administrar el clúster cómo para programar y ejecutar las aplicaciones. Cuando se planea la adquisición, se propone la compra del “mejor” equipo de computo.

La descripción de estos nodos es una propuesta ideal para ensamblar el clúster, en ésta también se describe el tipo de red o canal de comunicación que se recomienda, aunque por supuesto es hardware que cumple con los requerimientos básicos. Cabe aclarar que se propone la construcción de un clúster especifico, pues su propósito, problema o proyecto en particular, que se ha propuesto es para la ejecución de banco de datos.

En el capítulo 4. *Conclusiones y Recomendaciones*, en las pruebas de rendimiento se mostrará la utilidad de OpenSSI en relación a motores de banco de datos, por tanto se propondrá el diseño de un clúster que pueda en el mayor de los casos responder a las necesidades generales.

Se identifican como nodos aquellas unidades individuales de procesamiento en el clúster. Para la implementación de un clúster OpenSSI, el hardware soportado es para las arquitecturas IA32 y compatibles, es decir, en términos del hardware disponible en el mercado se refiere a los procesadores Intel y AMD con tecnología para procesamiento de palabras de 32 bits. OpenSSI ha sido desarrollado y probado para funcionar en estos dos tipos de procesadores y no es ideal ensamblar clústeres con ambos tipos de procesadores.

Los factores en la elección de uno u otro tipo de procesador son principalmente su costo y también por supuesto el software que se podrá aprovechar al máximo. Como ejemplo, se puede mencionar que por cuestiones de mercado, los procesadores AMD han sido más económicos sin querer decir con esto que tengan un menor rendimiento o tecnologías más obsoleta o atrasada, pero en cuanto al desarrollo de aplicaciones de software comerciales y no comerciales, como por ejemplos los compiladores, estas han sido desarrolladas y optimizadas para los procesadores Intel. Esto no quiere decir que sea imposible de implementar un clúster de bajo costo con hardware de bajo costo, simplemente se requiere en la mayoría de los casos un estudio de las herramientas disponibles para optimizar las aplicaciones de los usuarios.

Para este trabajo, el hardware requerido se encuentra disponible en el Centro de Investigación de Computo de la Facultad de Ingeniería, C.I.C.F.I.; allí se implementó la infraestructura de un clúster conformado por 3 nodos. A continuación se listara el hardware disponible de manera ilustrativa, no se abarcara en detalles los dispositivos periféricos (teclado, mouse, etc.), estas quedan a criterio de las necesidades reales que se pueda dar a estos dispositivos en el clúster

|  |  |
| --- | --- |
| Procesador | AMD Athlon(tm) Processor LE-1640  Processor Speed: 2.6 GHz.  L2: 1 MB. |
| Memoria (RAM): | 3 GB DDR-2 800 MHz. |
| Disco Duro (Hard Disk): | 250 GB, SCSI de 15,000 rpm. |
| Placa Madre (Motherboard): | ASUSTeK Computer INC. M2N-MX SE Plus  System Bus Speed: 800 MHz.  System Memory Speed: 667 MHz. |
| Tarjeta de Red | Intel (R) Pro/100 Network Conecction  Fast Ethernet |

***Requerimientos de Software***

**Sistema Operativo**

Debian GNU/Linux es un sistema operativo libre que soporta un total de doce arquitecturas de procesador e incluye los entornos de escritorio KDE, GNOME, Xfce y LXDE. La versión 5.0.8 Lenny con kernel 2.6 es compatible con las versiones openssi 1.9.6 y la versión alpha de openssi 2.0.

**Sistema de Archivos**

***Ext3***: Sistema de archivo utilizado para las particiones Linux, incluye la característica de *journaling* que previene el riesgo de corrupciones del sistema de archivos y es de los más utilizados por presentar mejor desempeño en el manejo de archivos.

***Cluster File System (CFS)***: Sistema de archivos distribuidos pertenecientes a un grupo de servidores que trabajan en conjunto para proporcionar un servicio de alto rendimiento para sus usuarios en vez de un único servidor con un conjunto de clientes. Para los usuarios del clúster el sistema de archivos es transparente, simplemente un sistema de archivos. El programa de manejo del sistema de archivos se encarga de distribuir las solicitudes a través de los elementos del almacenamiento del clúster. Los sistemas de archivos en clúster (CFS) permiten que múltiples servidores puedan acceder al mismo sistema de archivos. CFS resuelve las desventajas y complejidades de los discos duros proveyendo una solución más simple para la administración del almacenamiento

**Herramienta de Monitoreo**

***OpenSSI webView***: Aplicación grafica de monitoreo de OpenSSI, con ella puede realizarse parte de las tareas comunes de monitoreo que se harían con los comandos. El propósito de esta herramienta es la de proveer una visión general del estado del clúster, graficar las funciones claves del sistema. Permite al administrador del clúster controlar el estado de los recursos de cada nodo y de la infraestructura en general.

2.2.1.2 Organización de los nodos

En esta sección se da una descripción de la topología de la red y de los componentes de la conexión entre nodos del clúster, como son el switch y el cable de red utilizados. Cabe mencionar que esta configuración es parte de la red actual del Centro de Investigación en Computación de la Facultad de Ingeniería (CICFI).

La topología de red utilizada es la estrella, se caracteriza por tener todos sus nodos conectados a un controlador central, en este caso, un switch de 8 nodos de los cuales 4 nodos pertenecen al clúster. Todas las comunicaciones pasan a través del switch, siendo éste el encargado de controlarlas. Por este motivo, el fallo de un nodo en particular es fácil de detectar y no daña el resto de la red, pero un fallo en el controlador central desactiva la red completa.

El cable con el que están conectados los nodos del clúster al switch que los une, es estructurado de la marca Kron, para velocidades de transmisión de datos Ethernet 10/100/1000.

Los equipos conectados a este switch se conectan a una velocidad auto negociable de 10Base-T/100Base-TX, según la tarjeta de red.

2.3 Consideraciones de benchmark para el presente trabajo.

Se definió como objetivo el análisis del comportamiento de un banco de datos sobre una infraestructura SSI.

Debido a falta de un  benchmark que realice todas las validaciones necesarias para una infraestructura SSI se requirió del uso de benchmark sintéticos para evaluar los componentes. Con el montado del Banco de Datos se presentó la posibilidad de utilizar un benchmark de Aplicación que simule el trabajo real en una misma operación de banco de datos sobre un sistema SSI.

Considerando el benchmark a ser utilizado, se optó por un aplicación basada en uno de los estándares propuesto por entes reconocidos, logrando de esta manera tener referencia de las medidas ya publicadas por estos entes.

**Caracterización de la Carga.**

Las prestaciones de todo sistema se orientan según el tipo de trabajo o acciones para el cual se destinan, de esta manera, un sistema utilizado para simulaciones no posee las mismas características que un sistema transaccional de Débito - Crédito. Al trabajo de realizar el modelo representativo que simula una carga futura se le llama Caracterización de la Carga

Con el objetivo de caracterizar el uso de la infraestructura SSI, esta caracterización de carga se dividió en dos tipos según la manera de interactuar con el banco de datos: Transacciones OLTP y Consultas OLAP.

2.3.1 Benchmarks OLTP

Debido a las limitaciones económicas y del propio trabajo de tesis, el cual requiere que sean Benchmark de Aplicación Libre para transacciones OLTP se optó por la utilización del estándar TPC debido a su continuidad de los proyectos y a la utilización del mismo por entes de trayectoria [DONG97].

Las Implementaciones TPC son de dominio público, pero no existen muchas aplicaciones de uso libre que ayuden a utilizar de manera rápida este estándar, ya que mayormente son iniciativas de empresas privadas que utilizan para fines particulares.

2.3.1.1 TPC: The Transaction Processing Performance Council

El Transaction Processing Performance Council (TPC según sus siglas en ingles) es una organización sin fines de lucro destinada a definir procesos transaccionales y Benchmark de Banco de Datos para determinar de manera confiable el manejo de datos óptimo en la industria para condiciones de rendimiento.[TPCC11]

No existe una aplicación estándar provista por el consorcio de TPC, pero permiten que las empresas realicen implementaciones siguiendo el estándar para la realización de pruebas de rendimiento. Esta característica fue adoptada a finales de los años 80 cuando existían benchmark como TP1 [GRAY05] y Débito - Crédito [ZHOU99], que no era regulados por un ente y por ende emitían información con falta de veracidad, debido a publicaciones de empresas productoras de servidores. De esta forma se genero el primer benchmark basado en el consorcio TPC conocido como TPC-A, cuyas características reforzaban las reglas Atomicidad, Consistencia, Aislamiento y Durabilidad de transacciones al igual que la posibilidad de ejecutar en redes locales así como también en otras estructuras de red. Posteriormente se genero el estándar TPC-B, que fortaleció las características iniciales del TPC-A y agrego el uso de terminales simulando una situación aun más real del uso de un servidor en un ambiente cliente/servidor.

El propósito del Consorcio se definió de esta manera en los años 90 generando marcas de rendimiento de transacciones de base de datos, consumo de energía y otros. [SHAN98]

El consorcio TPC provee una expectativa de Precio/Rendimiento. Además, tiene extensiones encargadas de proveer información acerca del Consumo de energía [TPCE10]

Estándar TPC-C

El benchmark basado en el estándar TPC-C es una carga de trabajo de Procesos de Transacciones En Línea (OLTP, por sus siglas en Ingles Online Transaction Process) creado por el consorcio TPC. Es un conjunto de transacciones de lectura, inserción y actualización que simulan actividades OLTP complejas.

En TPC-C se utiliza al Rendimiento de trabajo para medir el número de órdenes o solicitudes de trabajo procesadas por minuto. La métrica utilizada es la de transacciones por minuto (tpmC).[TPCC11]

Para un mejor estudio el consorcio TPC dividió en clausulas que son expuestas como sigue:

Diseño Lógico de la Base de Datos:

Sección que define el entorno de aplicación y Negocio, la especificación de entidad, relación y característica de la Base de datos, junto con la disposición de tablas. Además cuenta con reglas de implementación, integridad y el manejo del acceso transparente de datos.

El estándar TPC-C establece una serie de transacciones que son pensadas para suponer que se trata de una empresa en la cual se desempeña la adquisición de productos. [LIMA09]

El modelo que es inicialmente representado en el benchmark presenta a una empresa de ventas con almacenes (del Ingles Warehouses) geográficamente distribuidos manejando el concepto de Departamentos o Distritos (del Ingles District). De esta manera cada Warehouse puede tener como máximo 10 District y este a su vez solo puede dar atención a 3000 Clientes por District.

Los almacenes tienen un Stock de 100000 productos que son vendidos por la compañía. [TPCC11].

En la Fig. 1 se presenta la esquematización provista por TPC-C de manera a visualizar el negocio.

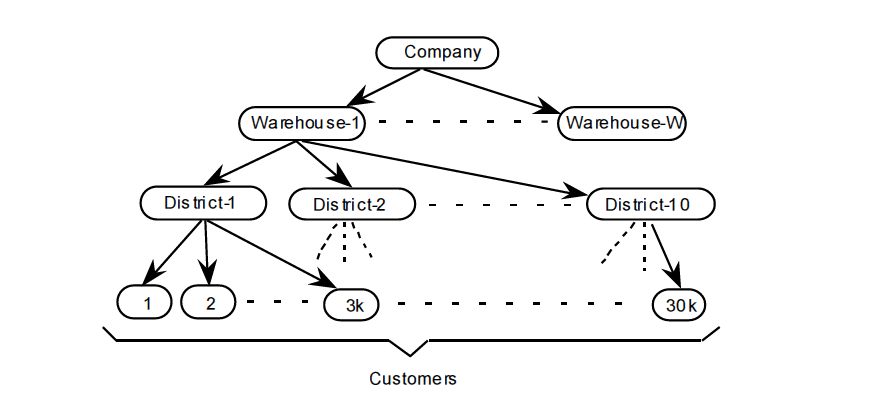


Fig. 1: Diagrama de Esquematización de una Compañía considerada por el estándar TPC (fuente propia)

A continuación se encuentra la Fig. 2 que presenta las relaciones entre tablas considerada para la aplicación del estándar TPC-C.

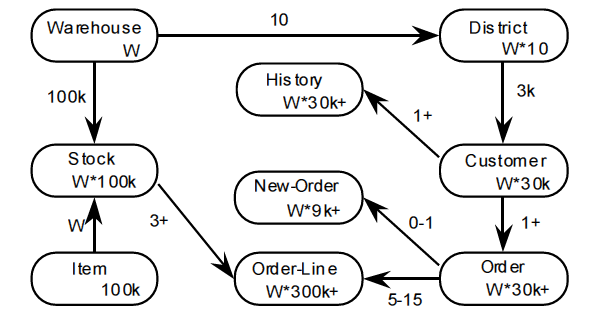


Fig. 2 Diagrama de relaciones entre tablas según el estándar TPC (fuente propia)

Perfil de Terminal y Transacciones

Sección que define los requisitos de Entrada y Salida de terminal. Los requisitos generales para perfiles de transacciones son sub divididas en:

Transacciones de Solicitud de Orden o New Order Transaction

Consiste en una transacción en la cual se ingresa una orden al esquema, es una transacción de alta frecuencia y de coste medio para la estructura debido a su lectura-escritura de datos.

Transacciones de Pago o Payment Transaction

Consta de una transacción que interactúa con el estado de cuenta del cliente, refleja el pago de una solicitud y representa un costo bajo para la estructura debido a las entidades que afecta. Esta transacción tiene una frecuencia alta de ejecución.

Transacciones de Estado de Orden o Order Status Transaction

Consiste en una consulta de la situación actual del cliente, representa una exigencia de nivel medio a la estructura debido a que es de solo lectura y mantiene una frecuencia baja de ejecución.

Transacciones de Envío o Delivery Transaction

Consta en la agrupación del envío de los productos solicitados. Esta agrupación se realiza cada 10 ítems y son enviadas de forma batch. Corresponde a un proceso de baja frecuencia y manteniendo una lectura-escritura de datos.

Transacciones de manejo de Stock o Stock-Level Transaction

Consta de una operación de variación de ítems en los almacenes; si bien es de baja ejecución requiere un alto costo computacional.

Propiedades del Sistema y Transacciones

Sección que define de manera generalizada las consideraciones ACID (por sus siglas del Ingles Atomicity, Consistency, Isolation and Durability) para el manejo de transacciones.

Escalado y Población de Base de Datos

Sección que define las reglas de ampliación de la base de datos y su vez la población de datos requeridos como mínimo para ser considerados.

Métricas de Rendimiento y Tiempo de Respuesta

Sección en la cual se define las métricas a ser consideradas por el consorcio TPC para considerar un rendimiento y un tiempo de respuesta optimo

SUT, Drivers y definiciones de Confirmación

En esta sección se define acerca de del Sistema bajo Prueba SUT (por sus siglas en Ingles System under Test), Drives Externos que son Provistos por emuladores de terminales externas conocidas como RTE (del Ingles Remote Terminal Emulator) y son usados para emular la carga de trabajo que pueden realizar los usuarios.

**Metodología de Precio**

Inicialmente el estándar TPC trata de realizar un cálculo precio/rendimiento de manera a demostrar el rendimiento según los costos del sistema a adquirir. En esta sección define como reportar y realizar este cálculo para publicaciones en el sitio de TPC.

Entre los aplicativos de uso público se encuentra una interesante opción desarrollada por catedráticos de la Universidad de Valladolid [TUVA02] conocido como TPCC-UVA que maneja los cálculos de Transacciones por minuto (tpmC) sin soporte a cálculos de costo - rendimiento (price/tpmC)  [TUVA02]

2.3.1.2 TPC-C UVA

El Benchmark TPC-C UVA es una implementación libre, bajo licencia GPL [GNPL07], del estándar TPC-C para medir sistemas OLTP. Fue concebido en el seno de la Universidad de Valladolid y es utilizado en la actualidad para medir ambientes de alto rendimiento, pero con la característica que también funciona de manera optima para ordenadores monoprocesadores.[TUVA02][XIAO06]

El esquema de TPC-CUVA fue desarrollado ampliamente en C considerando como banco de datos a Postgres SQL[HILL06] y el uso de un monitor de transacciones (TM por sus siglas en ingles de Transaction Monitor) libre para resguardarse bajo la licencia GPL.

Entre las características principales del TPC-C UVA se encuentra el estricto control de los requisitos del estándar TPC y el desarrollo en C del TM que permite que este benchmark sea ejecutado prácticamente en cualquier sistema operativo basado en Unix. Aun considerando la restricción que fue desarrollado para Banco de Datos en Postgres, es una herramienta potente debido a que la aplicación simula un comportamiento deseando al momento de ejecutar las pruebas. [LIMA09]

Opciones de la Suite TPC-C UVA

La aplicación generada por los catedráticos de la universidad de Valladolid consta de un menú textual que inicialmente provee los ítems necesarios para la elaboración del entorno de pruebas.

Respetando el esquema de la compañía propuesto por TPC el benchmark TPC-C UVA en su primera Opción genera una base de datos completa con un máximo de hasta 100 Warehouse con aproximadamente 138 megabytes de espacio por cada uno. Este proceso genera todo el esquema necesario para la elaboración de las pruebas. [CARRA08]

Luego de Generar el banco de datos con la cantidad de Warehouses deseado la aplicación de TPC-C UVA presenta un menú con las próximas posibles acciones a ser tomadas (Fig 3).

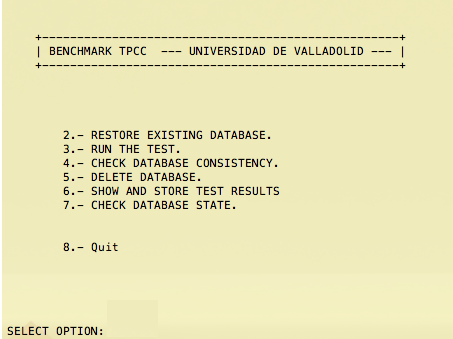


Fig. 3 Menu de opciones del Bench TPC-C UVA

Acerca del Uso de TPC-C UVA

**Opción 1** - Creación de una Nueva Base de Datos: Esta opción genera una nueva base de datos acorde al estándar TPC. El numero de Warehouse puede variar entre uno a cien y utiliza el directorio especificado en la instalación como directorio de creación. Se debe tener en cuenta que cada almacén de datos posee aproximadamente 137 megabytes de espacio necesario.

**Opción 2** – Restauración de la Base de Datos Existente: Esta opción deshace todos los cambios realizados al banco de datos. Es considerado como el nivel inicial de cada ejecución del benchmark.

**Opción 3** – Ejecución del Test: Es la opción principal de la Aplicación. Esta ejecuta el test en la base de datos creada o restaurada. Para la ejecución del test se establecieron parámetros que son solicitados en cada ejecución del test y son detallados a continuación:

*Numero de Warehouse*: Se indica cual es el número de Warehouse a utilizar, Si bien tenemos N almacenes es posible utilizar x Almacenes (donde 0 < x <= N)

*Número de Terminales por Warehouse*: Si bien el estándar TPC establece de manera inicial 10 terminales, mediante este valor se puede personalizar la cantidad de terminales que realizaran el test. Cada Terminal es un proceso que de manera independiente realiza la ejecución del test sobre cada Warehouse.

*Periodo de Ramp-Up*: es considerado el tiempo promedio en el cual el sistema se mantiene estable. Según la definición inicial el periodo de estabilidad del benchmark se encuentra aproximadamente en 20 minutos.

*Periodo de Medida*: Es el parámetro que configura el tiempo en el cual se ejecutara el benchmark. Según el estándar TPC este se encuentra entre 120 y 480 minutos

Luego de detallar los parámetros iniciales la aplicación consulta si se desean realizar mantenimientos periódicos de los recursos a ser utilizados. Estas opciones permiten configurar el periodo de realización de mantenimientos así como el número máximo de mantenimientos a ser realizados. Para la prueba máxima de 8 horas se recomienda limpiezas cada 60 minutos y como máximo 6 en la ejecución.

**Opción 4** – Constatar la consistencia de la Base de Datos: Esta opción realiza verificaciones constando que se realizan las 12 condiciones necesarias de consistencia de base de datos definido en el estándar TPC en su Capitulo 3.

**Opción 5** – Eliminación del Banco de Datos de Prueba: Permite indicar al usuario que se desea eliminar el banco de datos.

**Opción 6** – Verificación y almacenamiento de Resultados: Esta opción verifica los estados temporales del test realizado. Define si el banco de datos aprobó los valores mínimos de estándar TPC para banco de datos.

**Opción 7** – Verificación del estado de la Base de Datos: Esta opción presenta al usuario de la aplicación constatar cuantos registros existen en el banco de datos de manera verificar el funcionamiento de la aplicación.

**Opción 8** – Salir: Opción que finaliza la aplicación TPC-C UVA

2.3.2 Benchmarks OLAP

Con el fin de abarcar todas las aristas del análisis de rendimiento de base de datos fue necesario optar por una opción que cubra las expectativas OLAP. Así como la implementación TPC-C UVA se encargó de medir de forma explicita el comportamiento OLTP, el consorcio TPC provee un estándar para OLAP llamado TPC-H.

2.3.2.1 TPC-H

El TPC-H es un benchmark de aplicación que nació como sucesor del TPC-D, el cual en 1999 fue dividido para dar paso a TPC-R para verificaciones de reportes y TPC-H.

El Benchmark TPC-H trabaja sobre consultas realizadas de manera directa al Banco de datos, de forma tal que no puedan ser optimizadas por el Administrador de Base de Datos o DBA (por sus siglas en ingles de Data Base Administrator) y cuyo esquema de base de datos se encuentre normalizado en la 3ra forma normal. [POESS00]

Las consultas seleccionadas para la ejecución del Benchmark TPCH tienen las siguientes características:

* Alto grado de complejidad.
* Variedad de accesos utilizados.
* Son de carácter ad hoc.
* Abarcan un amplio porcentaje de datos disponibles.
* Todas las consultas difieren entre sí.
* Contienen consultas parametrizadas que varían durante la ejecución.

Para un mejor estudio el consorcio TPC dividió en clausulas que son expuestas como sigue:

**Diseño Lógico de la Base de Datos.**

Al igual que en el TPC-C, esta es una sección que define el entorno de aplicación y negocio, la especificación de entidad, relación y característica de la Base de datos.

El estándar TPC H establece una serie de consultas que son pensadas para suponer que se trata del estudio posible para cualquier modelo de empresa.

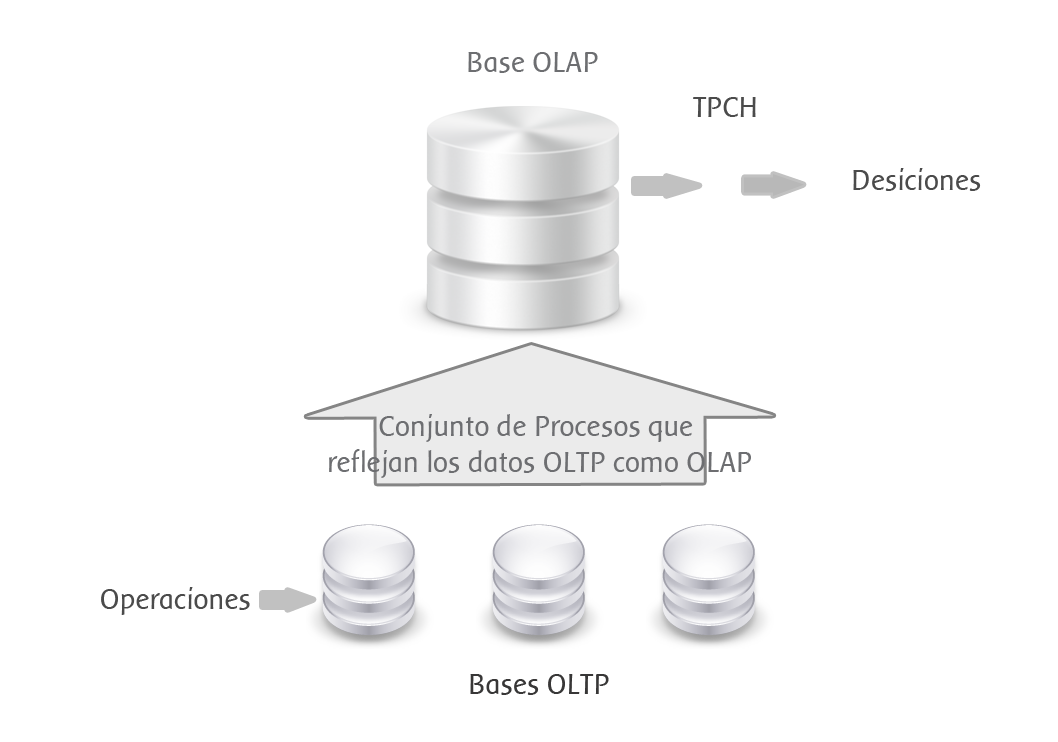


Fig. 4 Diseño lógico de la base de datos

El modelo presentado inicialmente por el estándar TPCH muestra como las operaciones realizadas de manera periódica en las bases OLTP, luego de un conjunto de procesos pasan a ser objeto de estudio en la base de datos OLAP para la toma de decisiones.

Análogamente al sistema TPCC, el sistema requerido por el estándar TPCH genera entidades y relaciones basadas en la tercera forma normal y esquematizada como se muestra en la fig. 5 [POESS00].

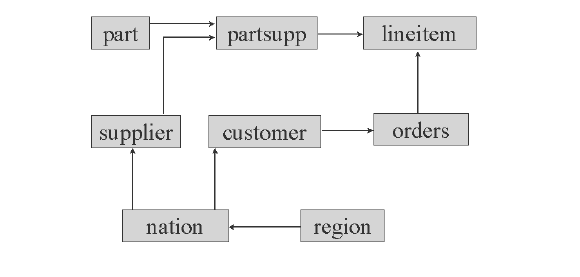


Fig. 5 Entidades y Relaciones para TPC-H

**Consultas y funciones de actualización.**

En esta sección se definen 22 consultas y entre ellos 2 funciones de actualización que pueden ser ejecutadas por parte en las pruebas TPCH.

Cada una de las consultas están caracterizadas por:

* Ilustrar las preguntas que normalmente se realizan en el transcurso de las toma de decisión.
* Estar definidas según el estándar SQL-92 para poder ser medida por consulta en caso de ser necesario.
* Tener parámetros de sustitución que son descriptos para obtener los resultados
* Validar las consultas en bases aptas para este tipo de operaciones

**Propiedades ACID (Atomicity, Consistency, Isolation, and Durability):**

En esta sección se define el uso de las propiedades ACID para mantener la consistencia del banco de datos a lo largo del periodo de test.

**Escalado y población de la base datos.**

En esta sección se definen las clausulas de escalado del banco de datos, así como también cuales son las medidas de población aceptadas por el estándar

**Métricas de performance y reglas de ejecución.**

En esta sección se definen los componentes del benchmark, las reglas a ser consideradas para la ejecución y configuración así como las métricas definidas por el estándar en sus clausulas.

**Sistema bajo Test (SUT) e implementación de drivers.**

Sección que define el manejo de los controladores en los posibles escenarios en los cuales el benchmark puede actuar.

**Metodología de Precio.**

Como el estándar TPCH intenta medir el precio de Transacciones por minutos realizadas en el control del soporte de decisiones (TpmH) define en esta sección políticas para la toma de resultados

2.4 Caracterización de Pruebas.

Se conoce como Caracterización de Pruebas a la definición de variables y métricas a ser considerados en la ejecución de los test de Carga. De manera a facilitar su comprensión se encuentra organizada como sigue:

2.4.1 Definición de Métricas para la cuantificación de Rendimiento.

La noción de rendimiento engloba muchos conceptos y para ello es necesario definirla. De manera a orientar Rendimiento junto con carga de trabajo damos énfasis a las siguientes definiciones:

2.4.1.1 Definición de valores para factor de Comparación.

Para cada métrica considerada como medida de rendimiento debe necesariamente existir un valor comparativo, que denote el éxito o fracaso de la evaluación mediante Benchmarks. Para ello existen dos posibilidades de evaluar mediante:

**La aplicación de Benchmark a Mainframes**

Actualmente en la página oficial del Benchmark existe publicaciones de pruebas realizadas a mainframes con distintas tecnologías y motores de base de datos publicadas por empresas privadas. [DONG97]. Este listado agrupa año tras año desde 1986 a las arquitecturas más resaltantes en cuanto a desempeño, englobando la disponibilidad adquirida mediante los sistemas de Procesamiento Paralelo Masivo (MPP por su siglas del ingles Masive Parallel Processing) y la estabilidad obtenida con los sistemas de Multi Procesamiento Simétrico (SMP por su siglas del ingles Symetric Multi Processing). Como criterio para medir el rendimiento se utiliza el mejor rendimiento obtenido por el Benchmark LINPACK [LINP08] que es una colección de subrutinas de Fortran que analiza y resuelve ecuaciones lineales para obtener el tiempo de proceso.

El equivalente para la utilización de las librerías de Linpack es una interface en C llamada Lapack [LAPA11] que provee los medios necesarios para utilizar ampliamente estas librerías.

**La comparación contra valores estándares del Benchmark:**

Para tener un punto de comparación de valores base se realizó una prueba en un ordenador no perteneciente a un entorno de clúster con aplicativos mínimos que plasmen el funcionamiento del benchmark para ordenador como punto de comparación.

El factor de comparación fue de Transacciones por minutos (tmpC) considerando esta medida como la utilizada por el Consorcio de TPC y corresponde a la medida para medir el Troughtput.

2.4.1.2 Definición de las pruebas.

De manera a definir las pruebas realizadas se separan los conceptos y consideraciones a tener en cuenta.

**Consideraciones de limitaciones de eficiencia.**

***Entrada y Salida*** (I/O por sus siglas del ingles Input Output): es considerado como la mayor limitación de eficiencia en los banco de datos debido a que el acceso y lectura a discos de almacenamiento es más lento que el acceso a memoria dinámica.

***Utilización de Consultas Ad hoc***: Son consultas no repetitivas utilizadas para las pruebas OLAP, al no ser reutilizadas el cache debe ser siempre vaciado para volver a consultar.

***Normalización de Datos para consultas OLAP*:** un requisito de la ejecución de benchmark OLAP es que las tuplas sean des normalizadas ocupando un mayor espacio en disco.

2.4.1.3 Aplicaciones a ser utilizadas

En el marco del desarrollo de las pruebas se definen las herramientas según el tipo de análisis a ser realizado:

***Transacciones OLTP*:** Se realizan transacciones OLTP según el estándar TPCC con la utilización del TPCC-UVA. Los resultados son generados por la aplicación. Los gráficos según los resultados obtenidos se generan con el programa GNUplot [BRAG03] una herramienta bajo licencia GPL.

***Consultas OLAP*:** Se genera los datos utilizando el programa provisto por el consorcio TPC llamado DBGEN [DBGE00]

2.4.1.4 Descripción de SUT (Sistema bajo Pruebas del ingles System Under Test):

Para la ejecución de los test se definen los siguientes objetos de estudio:

***Nodo simple*:** Ordenador estándar con características mínimas similares a los ordenadores que pueden obtenerse en el mercado.



Fig. 6 Nodo Simple

***Clúster de dos Nodos*:** Consta con la mínima cantidad de nodos para ser considerado un clúster. Se considera como un clúster homogéneo debido a que tanto el nodo maestro como el nodo esclavo poseen las mismas características entre sí.



Fig. 7 Cluster de 2 nodos

***Clúster de Tres Nodos*:** Consta de tres nodos homogéneos similares al nodo simple.

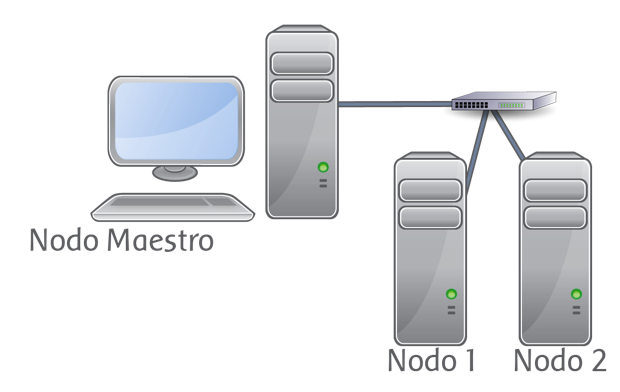


Fig. 8 Cluster de 3 nodos

***Clúster de Cuatro Nodos*:** Se incrementa la capacidad del clúster de tres nodos mediante la adición de un nodo mas, similar a los anteriores para mantener la homogeneidad del clúster de forma tal de proveer tolerancia a fallos.

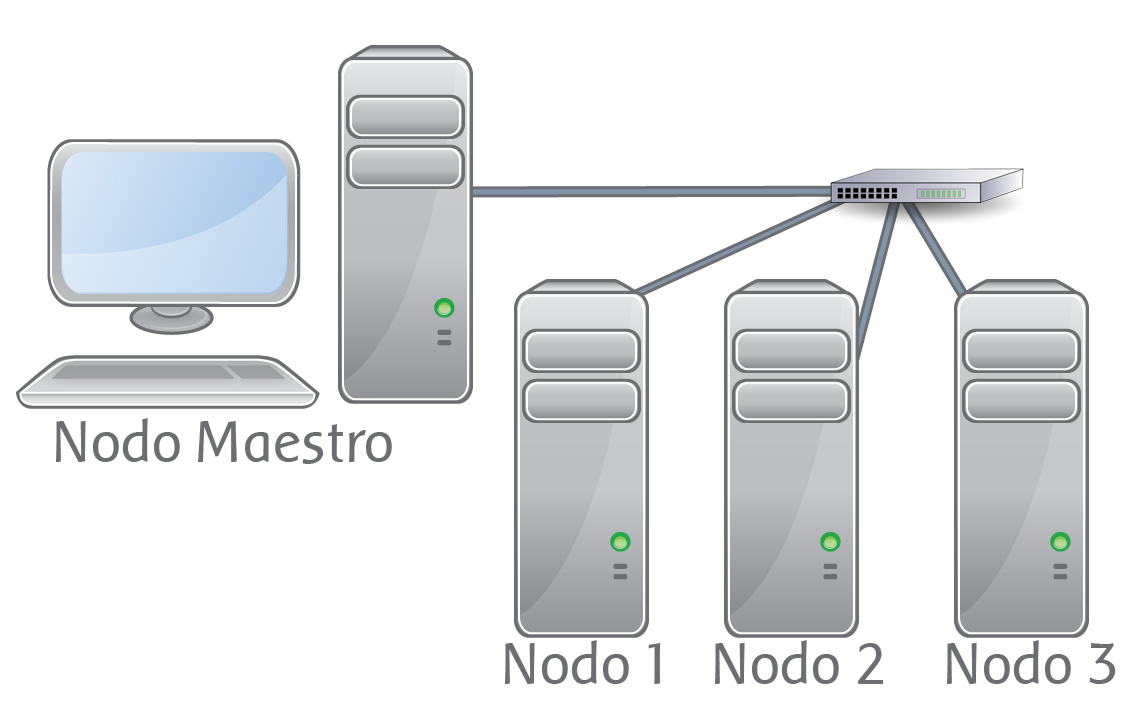


Fig. 9 Cluster de 4 nodos

Debido a que tanto el nodo simple, como los nodos componentes de cada uno de las estructuras poseen las mismas características, se engloban las características de en este apartado.

2.4.2 Metodología de evaluación con respecto a los componentes

La obtención de resultados fiables que requieran la aplicación de una cantidad de pasos dispuestos de forma tal, que los resultados obtenidos sean siempre comparable entre ellos.

El objetivo de esta sección es presentar la metodología utilizada en los análisis de los Sistemas Bajo Prueba (SUT).

2.4.2.1 Objetivo de Prueba

Las pruebas consistieron en ejecutar los diferentes test de pruebas en diferentes SUT, es decir, la misma prueba fue ejecutada en cada uno de los entornos.

2.4.2.2 Tamaño del Banco de Datos

De manera definir un tamaño a ser utilizado se contempló para cada uno de los tipos de pruebas valores diferentes definidos a continuación;

**Pruebas OLTP**

Se utilizó un máximo de 10 Warehouse el cual en promedio posee 137 megabytes por almacén, es decir, que en su totalidad conforman 1,33 Gigabytes.

**Pruebas OLAP**

Se utilizó una base promedio con 100 gigabytes de Datos, considerado así por el estándar TPCH.

2.4.2.3 Periodicidad de las Pruebas

Como los objetos de Prueba se basaron en dos formas diferentes de interactuar con la información se definió la periodicidad en dos tipos de pruebas:

**Pruebas OLTP**

Con el fin de establecer un tiempo estándar se tomaron 3 tiempos diferentes de pruebas con una unidad de medida en minutos, definidos como siguen:

1. **Prueba de Dos minutos:** esta prueba consistió en que el periodo de medida sea de dos minutos (2 min.) y fue el equivalente a una prueba de valores de tiempo mínimo. Para ello se utilizó un tiempo de Rampa de Un minuto debido a que es el mínimo valor permitido por los benchmark a ser utilizados.
2. **Prueba de Veinte Minutos:** Consistió en un valor aleatorio obtenido mediante un consenso. Para ello se utiliza el valor de Diecinueve minutos (19 min.) de periodo de rampa debido a que es un tiempo cercano al óptimo y en donde la cantidad de Transacciones procesadas por minuto empezó a llegar al máximo presentado.
3. **Prueba de Ciento veinte minutos:** Consistió en un valor considerado como mínimo optimo por los creadores del TPCC-UVA. Para ello se utilizó un periodo de rampa de Veinte minutos (20 min.) considerado como optimo mínimo según el manual del Benchmark aplicado.

Pruebas OLAP

Con respecto a la ejecución de Pruebas OLAP no se establecieron periodos de pruebas debido a que las pruebas OLAP se basaron en consultas al Banco de Datos y no en Operaciones por minutos.

2.4.3 Utilización del Banco de Datos

Debido a la limitación en cuanto al tamaño del disco duro, del nodo simple al ejecutarse las pruebas se manejó la variable de utilización del Banco de Datos, dividido según el tipo de prueba:

2.4.3.1 Pruebas OLTP

En la estructura de Banco de datos se puede limitar la ejecución de pruebas utilizando Almacenes o Warehouse para procesar las transacciones OLTP. Esta fue una característica que no permitió trabajar con todo el banco de datos y no contempló las terminales que se conectaron a esos almacenes. Para ello la división se realizó como sigue:

1. **Utilización de Un (1) Warehouse:** Fue considerado como el valor mínimo soportado por los SUT.
2. **Utilización de Seis (6) Warehouse:** Fue la máxima cantidad de almacenes soportada por un Nodo simple en el cual las pruebas resultaron exitosas. Excediendo este Valor las pruebas no concluyen según verificaciones preliminares realizadas.
3. **Utilización de Diez (10) Warehouse:** fue el estándar considerado por el estándar TPCC como medida optima de TPmC

2.4.3.2 Pruebas OLAP

No existe una limitación con respecto al Nodo simple, por la cual se utiliza la totalidad del Banco de Datos estipulado para este tipo de pruebas (100 Gb).

Repetición de Pruebas

De manera a reflejar resultados obtenidos según el promedio resultante de una serie de ejecución se estableció un valor de ejecución de 5 veces para cada prueba realizada (OLTP y OLAP).

2.5 Resumen

En este capitulo se han detallado las formas y procedimientos seguidos para la elaboración de este trabajo, también se han explicado los elementos involucrados en la realización de las pruebas de rendimiento de la infraestructura y la justificación del uso de las mismas. Se ha explicado con ayuda de gráficos la composición de los nodos de la infraestructura para las diferentes pruebas que se llevaron a cabo.

En conclusión se ha podido determinar el entorno tecnológico sobre el cual la infraestructura fue implementada, los procedimientos para que dicha implementación se haya llevado a cabo. Y por ultimo las metodologías escogidas para las pruebas de rendimiento que se llevaron a cabo

3. RESULTADOS Y ANALISIS

3.1 Implementación del clúster OpenSSI

Como se habló en la sección **2.2.2 *Optando por OpenSSI como un clúster de alto desempeño y balanceo de carga***, la distribución de procesos en los nodos del clúster es la principal característica de funcionalidad de OpenSSI. De una forma ideal, los procesos buscaran ejecutarse en un ambiente igual o muy similar del nodo donde provienen, con mejores recursos de memoria y procesador. Hablando en términos del programa, éste estaría ejecutándose en nodos externos, por lo que es indispensable que el procesador externo tenga el conjunto de instrucciones necesarias para poder procesar el código compilado en el nodo anfitrión, de otra forma el programa no podría ejecutarse.

Con base en estas observaciones, se concluye que la construcción de un clúster OpenSSI como en otros tipos de clústeres es ideal que se haga con hardware homogéneo, aunque muchas de las veces debido a los recursos con los que se pueden disponer en nuestro centros de investigación no es posible cumplir con este requerimiento, no indispensable pero si recomendado, pues además de proveer un ambiente de ejecución adecuado para los procesos, facilita la administración del software instalado en los nodos del clúster.

3.1.1 Instalación del clúster OpenSSI

El proceso de instalación fue dificultoso al inicio, principalmente por lo incompleto y esparzo que resultó ser la documentación oficial [SSID11]. Es bastante aconsejable estar muy bien familiarizado con la distribución que se estará utilizando, en este caso en particular Debian Lenny, de modo que se puedan investigar y solventar los posibles problemas.

En un entorno OpenSSI debe existir un nodo especial denominado nodo init o master. Éste arranca directamente desde el sistema de archivos raíz. El resto de los nodos arrancan desde la red. Para arrancar los nodos non-init o esclavos por red es necesario Etherboot (definiciones) o PXE (definiciones). Etherboot es software libre y se instala en un medio que permita el arranque, como un CD o un disco duro, o bien se graba en la memoria ROM de la tarjeta de red. La mayoría de las tarjetas de red son compatibles con Etherboot. PXE, por otro lado, se encuentra integrada en algunas tarjetas de red y normalmente se habilita desde el menú de la BIOS.

Debido a que el propósito principal de este trabajo de tesis, es proveer una guía metodología completa de la instalación del nodo master, agregar nodos esclavos y la instalación del una herramienta web para el monitorización visual del clúster OpenSSI. Los pasos detallados de los procesos mencionados previamente se pueden encontrar en la sección de anexos respectivos.

3.2 Obtención de Resultados y Caracterización de Pruebas

Con el fin de realizar las estadísticas con los resultados obtenidos luego de las pruebas se estableció una metodología de registro de Resultados según el tipo de prueba y definidos a continuación:

3.2.1 Pruebas OLTP:

Para el registro de cada prueba se tuvo presente el formulario establecido como estándar por el consorcio TPCC, ver anexo Formulario OLTP de Pruebas, y a su vez fueron registrados los resultados generales de cantidad de transacciones por minutos (TpmC) en la Planilla General de Pruebas, la cual se halla disponible como uno de los anexos a este trabajo.

3.2.2 Pruebas OLAP:

Los resultados de los tiempos registrados se encuentran en el anexo de Planilla general de pruebas

De manera a obtener un solo resultado de las pruebas realizadas, se presentó como resultado final el promedio de los valores obtenidos del total de ejecuciones realizadas por cada tipo de prueba

3.3 Análisis de resultados y gráficos

En este apartado se exponen los gráficos generados a partir de los resultados obtenidos durante las pruebas. Para una mejor comprensión se decidió separar el análisis, según el tipo de prueba ejecutada.

3.3.1 Transacciones OLTP

Las pruebas se realizaron en 5 ejecuciones y los resultados tomados fueron los promedios de estas.

De manera a contrastar el comportamiento de las transacciones OLTP se presenta un resumen general en la Tabla 1. En ella se despliegan las transacciones por minutos arrojadas por el benchmark tpcc-uva (TpmC-uva) reflejando su comportamiento ante diferentes situaciones.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tipos\nodos | 1 (seg.) | 2 (seg.) | 3 (seg.) | 4 (seg.) |
| 1-2 | 16,995 | 16,499 | 17,000 | 16,999 |
| 1-20 | 13,650 | 13,600 | 13,650 | 13,649 |
| 1-120 | 12,500 | 12,580 | 12,617 | 12,292 |
| 6-2 | 64,998 | 104,443 | 101,498 | 101,496 |
| 6-20 | 77,250 | 78,900 | 78,850 | 77,950 |
| 6-120 | 65,850 | 68,958 | 75,875 | 74,292 |
| 10-2 | 78,540 | 163,540 | 165,497 | 160,993 |
| 10-20 | 76,850 | 128,850 | 128,300 | 119,149 |
| 10-120 | 69,042 | 103,042 | 108,903 | 116,842 |

Tabla 1 Tabla general de Resultados OLTP

A modo de permitir una mejor visualización del comportamiento de las pruebas, se aislaron estas según el tiempo de ejecución del benchmark y están definidas como sigue:

3.3.1.1 Pruebas realizadas en un periodo de 2 Minutos:

La agrupación de pruebas realizadas con un periodo de 2 minutos permite notar que se presenta un comportamiento optimizado para un mayor número de warehouse. Estas afirmaciones son las reflejadas según los datos de la Tabla 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Warehouse\nodos | 1 (seg.) | 2 (seg.) | 3 (seg.) | 4 (seg.) |
| 1 warehouse | 16,995 | 16,499 | 17,000 | 16,999 |
| 6 warehouse | 64,998 | 104,443 | 101,498 | 101,496 |
| 10 warehouse | 78,540 | 163,540 | 165,497 | 160,993 |

Tabla 2 Pruebas de 2 minutos

La tabla demuestra que para el valor mínimo de warehouse () no existe una variación (∂) significante de la línea formada por las intersecciones de TpmC-uva y el numero de nodos. Estos valores fueron obtenidos teniendo como variable el tiempo de ejecución del benchmark y se expresa en la Gráfica 1.

Gráfica 1 Rendimiento OLTP 1 Warehouse

Se ha realizado un análisis de la eficiencia con respecto al nodo principal cuyo resultado se reflejó en la Tabla 3, que expresa los porcentajes de mejora según se agregan nuevos nodos.

|  |  |  |  |
| --- | --- | --- | --- |
| Warehouse\nodo agregado | 2do (% mejora) | 3ro (% mejora) | 4to (% mejora) |
| 1 warehouse | -2,919 | 0,029 | 0,024 |
| 6 warehouse | 60,686 | 56,156 | 56,152 |
| 10 warehouse | 108,225 | 110,717 | 104,982 |

Tabla 3 Porcentajes de mejora obtenida durante la adición de nodos.

Se puede observar que para un warehouse existe un porcentaje de ganancia que oscila entre el -2,919% y 0,029% ampliamente mejorado al ejecutar la evaluación con seis warehouse, cuya mejora se extiende a un rango del 56,156% y 60,668% para el peor y mejor caso respectivamente. Esta mejora obtenida con el incremento de la cantidad de warehouse se afirma al aumentar nuevamente a diez warehouse, obteniendo porcentajes de ganancia que varían desde el 104,982% al 108,225%. Las ganancias expuestas quedan reflejadas en la Gráfica 2.

Gráfica 2 Ganancia obtenida en Pruebas OLTP durante una ejecución de 2 minutos

3.3.1.2 Pruebas realizadas en un periodo de 20 minutos:

Se realizaron experiencias durante un periodo de 20 minutos en el cual se presentó una mayor cantidad de transacciones por minuto conforme se aumentó la cantidad de warehouse. Esta aseveración se observa en los datos plasmados en la Tabla 4.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Warehouse\nodos | 1 (seg.) | 2 (seg.) | 3 (seg.) | 4 (seg.) |
| 1 warehouse | 13,650 | 13,600 | 13,650 | 13,649 |
| 6 warehouse | 77,250 | 78,900 | 78,850 | 77,950 |
| 10 warehouse | 76,850 | 128,850 | 128,300 | 119,149 |

Tabla 4 Pruebas realizadas en un periodo de 20 minutos

La tabla despliega un aumento de rendimiento durante el incremento de la cantidad de warehouse para todas las situaciones, no variando de manera considerable durante la adición del 3er y 4to nodo. Por otro lado notamos un incremento considerable únicamente en el segmento denotado por el TpmC obtenido entre las ejecuciones con 1 y 2 nodos. Estas afirmaciones son perfiladas en la Gráfica 3.

Gráfica 3 Rendimiento de transacciones OLTP realizadas en un periodo de 120 minutos.

Se elaboro una tabla de porcentaje de mejoras obtenidas durante el incremento de nodos, representados en la Tabla 5, de manera a contrastar el comportamiento general de las pruebas realizadas en este periodo de tiempo.

|  |  |  |  |
| --- | --- | --- | --- |
| Warehouse\nodos | 2do(% mejora) | 3ro(% mejora) | 4to(% mejora) |
| 1 warehouse | -0,366 | 0,000 | -0,007 |
| 6 warehouse | 2,136 | 2,071 | 0,906 |
| 10 warehouse | 67,664 | 66,949 | 55,041 |

Tabla 5 Porcentaje de mejora obtenida en test de 20 minutos de transacciones OLTP

De esta última deducimos que el mayor margen de ganancia se obtiene durante la ejecución de las pruebas en 10 warehouse de datos y contrariamente a las pruebas realizadas con un warehouse que no presentaron ganancia significativa con respecto al 1er nodo tal como lo refleja la Gráfica 4.

Gráfica 4 Ganancia obtenida 20 minutos

3.3.1.3 Pruebas Realizadas durante un periodo de 120 minutos:

El tercer grupo de pruebas fue utilizando un periodo de 120 minutos presentado la menor cantidad de transacciones con respecto a los experimentos anteriores, tal como lo expresa la Tabla 6.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Warehouse \nodos | 1 (seg.) | 2 (seg.) | 3 (seg.) | 4 (seg.) |
| 1 warehouse | 12,500 | 12,580 | 12,617 | 12,292 |
| 6 warehouse | 65,850 | 68,958 | 75,875 | 74,292 |
| 10 warehouse | 69,042 | 103,042 | 108,903 | 116,842 |

Tabla 6 Pruebas OLTP para periodos de 120 minutos

Esta última evaluación obtuvo un incremento en el rendimiento durante la adición de warehouse a las pruebas y presentó una pendiente considerable durante la adición de nodos únicamente para 10 warehouse de datos. La afirmación es contrastada en la Gráfica 5.

Gráfica 5 Rendimiento de transacciones OLTP para ejecuciones de 120 minutos

Se contrastó los beneficios obtenidos en las pruebas para un periodo de 120 minutos y para ello se elaboró la Tabla 7.

|  |  |  |  |
| --- | --- | --- | --- |
| Warehouse\nodos | % mejora | % mejora | % mejora |
| 1 warehouse | 0,640 | 0,936 | -1,664 |
| 6 warehouse | 4,720 | 15,224 | 12,820 |
| 10 warehouse | 49,245 | 57,734 | 69,233 |

Tabla 7 Mejora Rendimiento OLTP para un periodo de 120 minutos

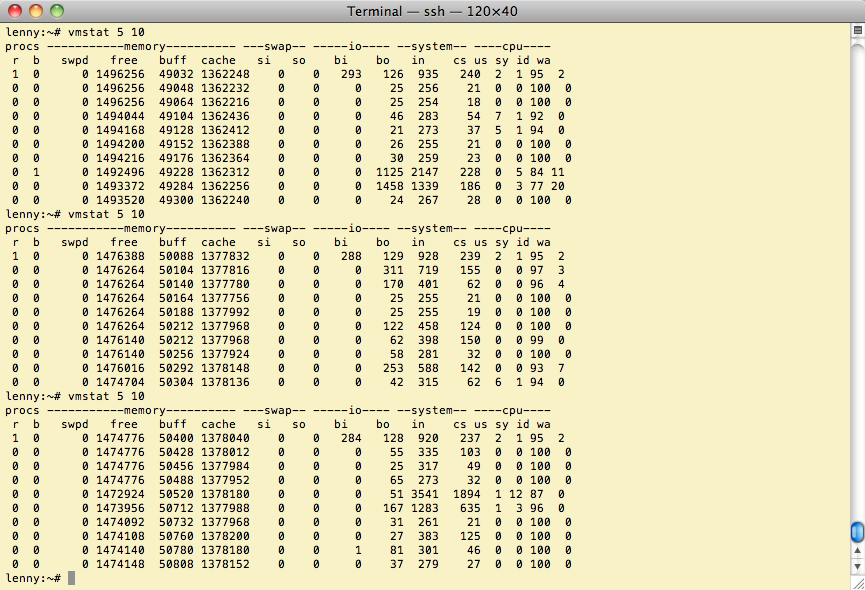
Esta tabla mantiene el comportamiento reflejado a lo largo de las pruebas que presenta mejores márgenes de ganancia conforme se incrementan los warehouse y presentando un crecimiento considerable para warehouse mayores a 1.

Gráfica 6 Ganancia obtenida 10 warehouse

3.3.1.4 Análisis de los resultados obtenidos durante las pruebas OLTP

Para las pruebas de 2 y de 20 minutos es posible apreciar que no se existe una escalabilidad marcada en cuanto al rendimiento con relación al incremento de nodos.

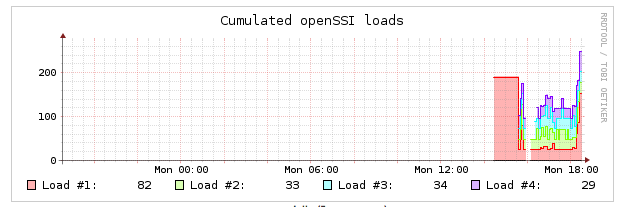
Se utilizaron herramientas de control como *vmstat* y *webView* para la verificación de la existencia de algún cuello de botella en la red o en el proceso de Escritura/Lectura del disco. Además mediante el uso de la herramienta *vmstat* se pudo observar en la Gráfica 7, los valores de las columnas SWAP e IO son o nulos o ínfimos, indicando así que no existen procesos en espera para escritura o lectura y descartando un posible cuello de botella.



Gráfica 7 Monitoreo vmstat

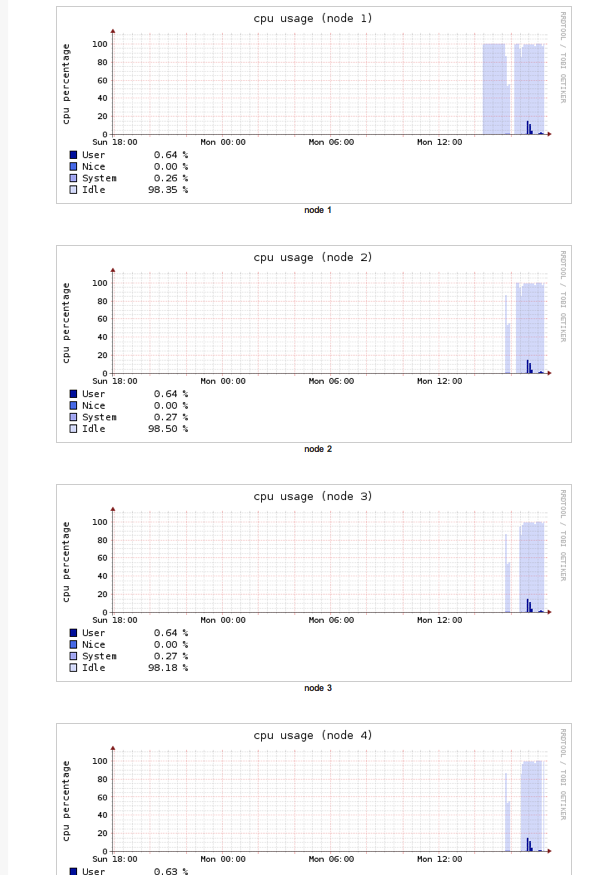
Con webView se ha controlado el tráfico de red, el cual para los casos de pruebas de 2 y 20 minutos no presenta uso considerable alguno, evidenciando que este tampoco ha estado afectando al rendimiento de la infraestructura. Sin embargo se puede apreciar en las graficas proporcionadas por el webView que la distribución de carga entre nodos y el uso de procesador de cada nodo alcanzan picos muy elevados. Esta última permite apreciar que la migración de los procesos a los demás nodos ha estado funcionando correctamente, balanceando la carga de trabajo a través de todos los nodos de la infraestructura.

En cuanto a las pruebas realizadas durante un periodo de 120 minutos, la variable W presenta un incremento de rendimiento más notable, debido a que el periodo de prueba es el mínimo óptimo establecido por las recomendaciones del TPCC-UVA.



Gráfica 8 Carga de procesos en el cluster

Se observa en la Gráfica 5 que la pendiente no es muy pronunciada para el periodo de prueba más alto. Pero al igual que en los casos de prueba de menor tiempo, la distribución de carga, el uso de los procesadores y memoria, indican que la infraestructura estuvo trabajando con su máxima capacidad. Estos datos han sido compartidos con la comunidad de OpenSSI para poder ayudar a la continuación de la evaluación de las pruebas de rendimiento de E/S (Entrada/Salida) de datos. Actualmente todavía no existe una configuración ideal de la infraestructura SSI para este tipo de pruebas, por ello se espera que el presente trabajo sirva como punto de ayuda en cuanto a ajustes del middleware y del motor de base de datos se refiera.



Gráfica 9 Uso del procesador por nodo

3.3.2 Transacciones OLAP

Por otra parte se realizaron pruebas que cubren el procesamiento analítico en línea (OLAP por sus siglas en ingles) que al igual que las transacciones OLTP se ejecutaron 5 veces tomando como resultado el mejor caso obtenido.

El objetivo de las pruebas OLAP se basaron en verificar el comportamiento de la ejecución de consultas utilizadas en los sistemas de información gerencial en SUT diferentes, conforme se incrementan los nodos.

A modo de resumen general se elaboro el promedio de todas las consultas OLAP realizadas en los distintos entornos y cuyos datos son expresados en la Tabla 8.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tamaño\nodos | 1 (tpmh.) | 2(tpmh.) | 3 (tpmh.) | 4 (tpmh.) |
| 1 GB | 26,096 | 42,066 | 66,282 | 60,676 |
| 10 GB | 26,106 | 98,304 | 123,991 | 204,090 |
| 100 GB | 21,817 | 29,966 | 108,927 | 84,692 |

Tabla 8 Resultados Generales OLAP

De manera a realizar un análisis acerca de este resultado, la Gráfica 10 muestra una notable optimización de rendimiento conforme se aumentan los nodos. Este comportamiento regular de los tiempos obtenidos con el aumento de nodos en el sistema bajo prueba, plasma las mejoras obtenidas principalmente en el segmento denotado entre el 2do y el 3er nodo en donde la pendiente es más pronunciada. Además de ello se verifica un leve decremento de rendimiento al agregar el 4to. nodo para los valores máximos considerados.

Gráfica 10 Rendimiento Promedio OLAP

La optimización del sistema se plasmo en la Tabla 9, en la cual se calculan los porcentajes de ganancia obtenida durante el incremento de nodos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tamaño\nodos | 1 (%.) | 2(%.) | 3 (%.) | 4 (%.) |
| 1 GB | 0,000 | 37,963 | 60,628 | 56,990 |
| 10 GB | 0,000 | 73,443 | 78,945 | 87,209 |
| 100 GB | 0,000 | 27,193 | 79,971 | 74,239 |

Tabla 9 Porcentaje Mejora para consultas OLAP con respecto rendimiento del primer nodo.

Se elaboró la Gráfica 11 que traza los datos de la tabla anterior permitiendo que los rangos de porcentaje de optimización se reflejen una mejor manera.

Gráfica 11 Ganancia por nodo agregado OLAP

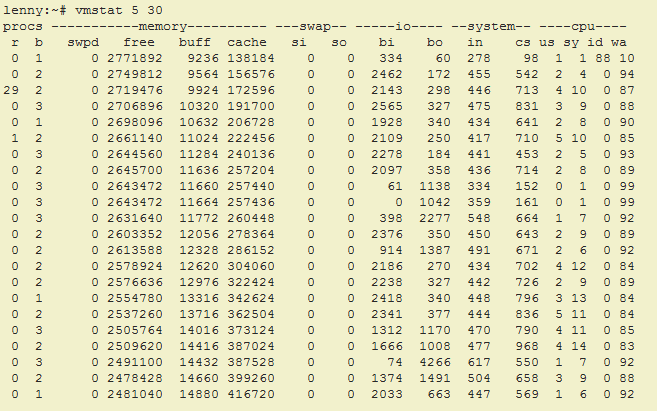
En la Gráfica 11 en el cual se aprecia que, durante la adición del 2do nodo, existe un margen de ganancia del 37,963% al 60,628% para banco de datos de 1 GB y alcanzando una optimización, con 10 GB de datos, de ganancias que van desde el 73,443% al 87,209%. Durante la ejecución en bancos de datos de 100 GB se reflejo un margen de optimización entre el 27,193% y el 79,971%.

3.3.2.1 Análisis de los resultados obtenidos durante las pruebas OLAP

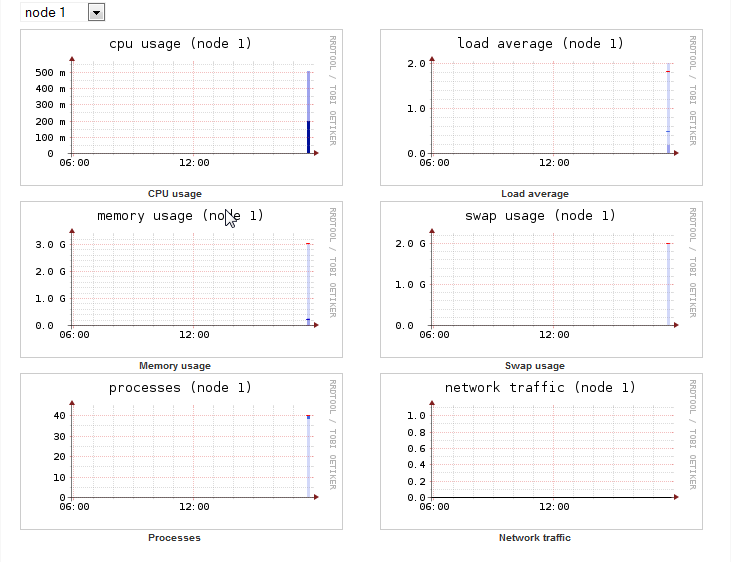
Las pruebas se comportaron de manera regular según el promedio general obtenido, es por ello que se omite el análisis según el tamaño del banco de datos.

Consideramos como hipótesis que la razón del incremento del margen de ganancia, se debió a la optimización de carga y memoria brindada al agregar un nodo más a la estructura.

Para corroborar estas observaciones se obtuvo un estado del la estructura capturando en la Gráfica 12, mediante el uso de vmstat en el cual se puede notar el uso de carga, swap cpu y memoria para las pruebas en las que se utilizo solo el nodo 1. Estas medidas se encuentraron soportas con el monitoreo visualizado en la Gráfica 13 obtenida mediante la herramienta de monitoreo webView, en la cual se puede apreciar la utilización de cpu, carga, memoria y swap expuestas por el *vmstat* de manera grafica.

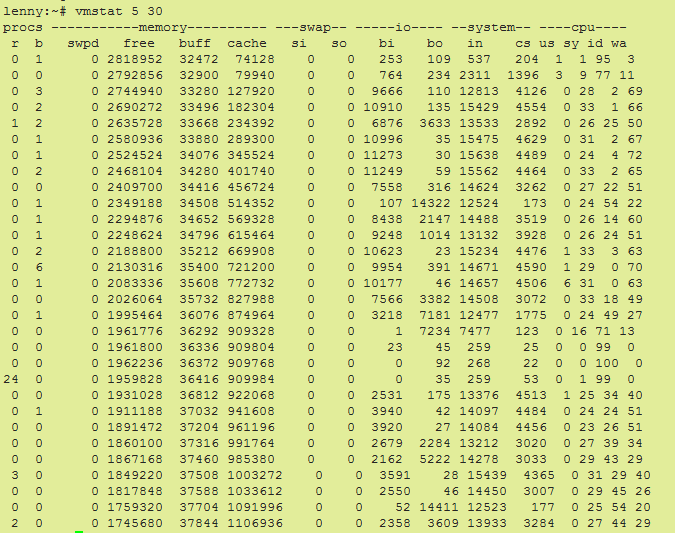


Gráfica 12 Monitoreo con vmstat para Consultas OLAP de 1 nodo



Gráfica 13 Monitoreo con webView de consultas OLAP de 1 nodo

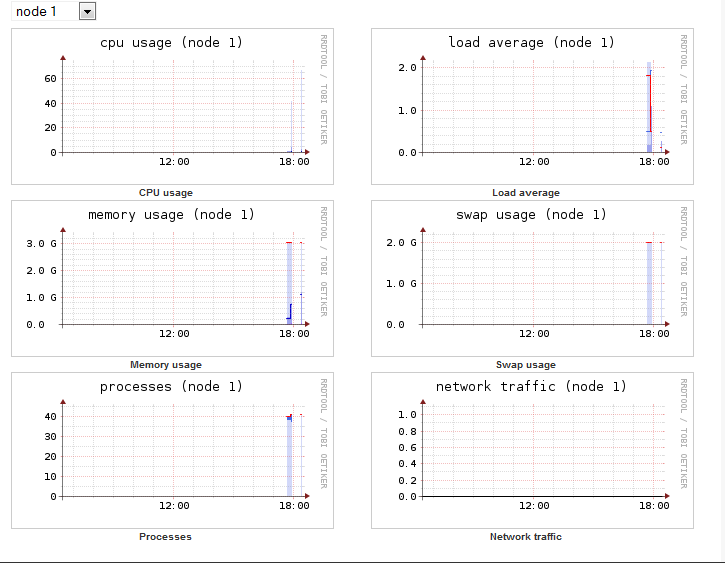
Con el fin de obtener un contraste para evaluar el comportamiento se obtuvo la misma instrumentación durante el trabajo similar con 4 nodos el cual genero el resultado expuesto en la Gráfica 14 durante el uso del utilitario *vmstat*.



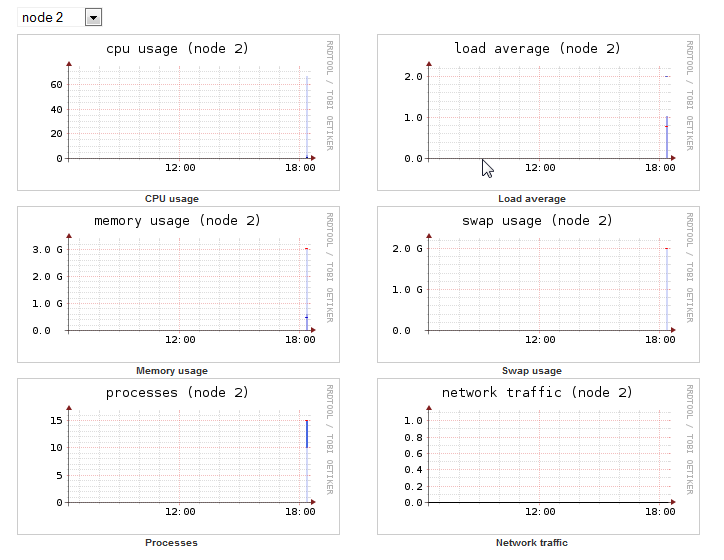
Gráfica 14 Monitoreo con vmstat para Consultas OLAP de 4 nodos

Un trabajo similar se realizo con la herramienta webView quedando definido como sigue:

1. Gráfica 15: trabajo del nodo 1 durante la realización de una consulta OLAP utilizando 4 nodos.
2. Gráfica 16: trabajo del nodo 2 durante la realización de una consulta OLAP utilizando 4 nodos
3. Gráfica 17: trabajo del nodo 3 durante la realización de una consulta OLAP utilizando 4 nodos
4. Gráfica 18: trabajo del nodo 4 durante la realización de una consulta OLAP utilizando 4 nodos.



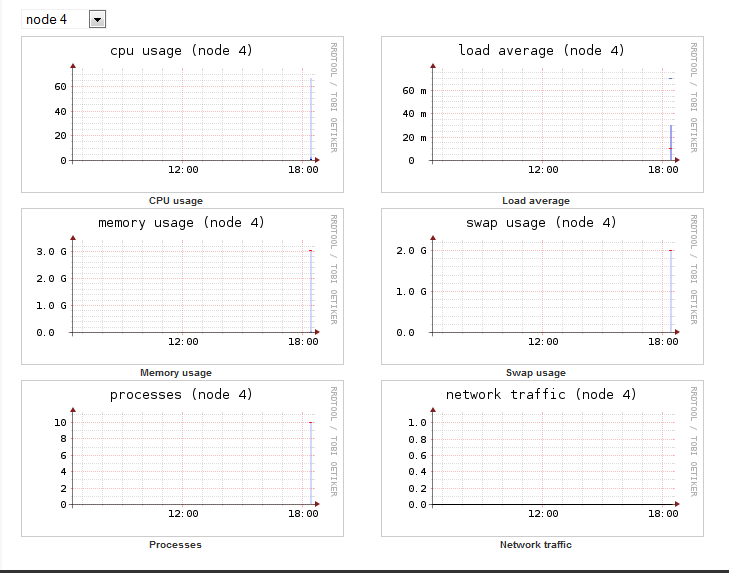
Gráfica 15 Monitoreo con webView del nodo 1 de consultas OLAP con 4 nodos



Gráfica 16 Monitoreo con webView del nodo 2 de consultas OLAP con 4 nodos



Gráfica 17 Monitoreo con webView del nodo 3 de consultas OLAP con 4 nodos



Gráfica 18 Monitoreo con webView del nodo 4 de consultas OLAP con 4 nodos

Luego de contrastar el trabajo con 4 nodos con respecto a la realizada con un nodo, hemos notado que la carga y el uso de memoria junto a otros recursos, también son migrada a los nuevos nodos; aumentando el rendimiento de la infraestructura como se visualiza en las graficas y evidenciando de esta manera el correcto funcionamiento de la infraestructura para consultas OLAP.

3.3.3 Síntesis de análisis y resultados

A modo de globalizar las deducciones realizadas se obtuvieron dos panoramas bien marcados. Uno de ellos es el referente al comportamiento de la Transacciones OLTP según la cantidad de warehouse utilizados, en el cual es posible apreciar que dado un warehouse () igual a 1 existe un rango de ganancia () que oscila entre el -3% y el +3% denotado por la expresión:

Además según el comportamiento de la ganancia (), para transacciones de un periodo (*t*) de 120 minutos a partir del 2do nodo, asciende en segmentos de aproximadamente igual al 10% quedando expresado como sigue:

El otro panorama es el resultado de las operaciones OLAP realizadas, que para todos los tamaños de banco de datos (*T*) incluidos en el test durante:

1. la incorporación del 2do nodo se nota un incremento que oscila entre el 3% y el 10% aproximadamente tal como muestra la siguiente expresión:
2. La incorporación del 3er nodo denota un incremento entre el 35% y el 71% aproximadamente:
3. La incorporación del 4to nodo denota un incremento entre el 20% y el 82% aproximadamente:

3.4 Resumen

En este capitulo se ha descripto la metodología de implementación ideal para el clúster OpenSSI de acuerdo al equipo de computo disponible en el CICFI para la construcción del mismo. La instalación de nodos y su configuración, y la implementación de la herramienta de monitoreo de la infraestructura.

También se han presentado los resultados de pruebas de rendimiento del tipo OLTP y OLAP para aplicaciones de banco de datos. En conclusión se ha comprobado que la infraestructura presenta una escalabilidad de rendimiento durante las pruebas OLTP de prolongado tiempo de ejecución. Así también durante las ejecuciones de las pruebas OLAP fue posible demostrar su escalabilidad al disminuir el tiempo necesario de procesamiento en relación al incremento de nodos a la infraestructura.

4 CONCLUSIONES Y RECOMENDACIONES

Como culminación del presente trabajo, se presentan en este capítulo las conclusiones finales y son propuestos algunos tópicos como futuras líneas de investigaciones.

En este trabajo se ha implementado en el Laboratorio del Centro de Investigación en Computación de la Facultad de Ingeniería (CICFI) una infraestructura de altas prestaciones SSI [MORI04] para aplicaciones de banco de datos, como alternativa a la adquisición de costosos Mainframes.

Se ha evaluado el desempeño de la infraestructura mediante pruebas de rendimiento basadas en el estándar TPC [TPCC11]. Estos resultados además sirven como aporte a la comunidad de código abierto del proyecto OpenSSI [WALK05] sobre el comportamiento de la infraestructura ejecutando pruebas de rendimiento TPC-C y TPC-H.

Con los resultados obtenidos se demostró que una infraestructura de altas prestaciones brinda mayor eficiencia que un ordenador de características medias. Si bien las limitaciones con las que se cuenta en el CICFI como tráfico de red y uso del espacio físico entre otros, son incidentes directos en la eficiencia de la infraestructura, se ha podido demostrar que con los recursos presentes se ha podido adquirir una infraestructura que provea altas prestaciones a menor coste y manteniendo el nivel de criticidad de los datos establecidos.

Se ha optimizado la infraestructura para la ejecución de Banco de datos mediante ajustes en las configuraciones del middleware SSI. Además el presente trabajo de tesis sirve como guía metodológica de implementación de una infraestructura de altas prestaciones basado en el middleware OpenSSI [WALK05] y optimizado para la ejecución de aplicaciones de banco de datos.

Hemos ejecutado las pruebas con un clúster OpenSSI de hasta 4 maquinas y obtuvimos buena eficiencia. Estamos conscientes que la infraestructura distribuida al igual que cualquier otra que utilice el trafico de red, presentara ciertas limitaciones al ejecutarse con una gran cantidad de maquinas por el exceso de tráfico. Esto se podría mejorar recompilando el propio kernel de OpenSSI para soporte de una mayor cantidad de nodos. Con estas mejoras pensamos que podría mejorar el rendimiento.

4.1 Mejoras a lograr

* Optimizar el estándar TPC-C UVA para entornos distribuidos de alto rendimiento.
* Extender la herramienta de monitoreo WebView, con el modulo de administración para agregar y remover nodos por medio de su interfaz graficas.
* Implementar un sistema de almacenamiento que use múltiples discos, como alternativa al sistema de tolerancia de fallos SSIfailover del middleware.
* La mejora de los componentes físicos que participan durante el trafico de red, como ser placas de red mas veloces, un enrutador de mayor velocidad.

4.2 Futuras Investigaciones

De forma a continuar con el trabajo iniciado en esta tesis, los siguientes tópicos son propuestos para trabajos futuros:

* Evaluar las nuevas versiones de OpenSSI que ya aprovechan algunas arquitecturas de 64 bits;
* Realizar la implementación de la infraestructura SSI con instancias virtuales;
* Realizar pruebas con una gran cantidad de clientes;
* Evaluar la implementación de entornos elásticos sobre la infraestructura SSI.

Bibliografía

[BEAS90] BEASLEY J. E. Linear programming on Cray supercomputers. The Journal of the Operational Research Society. Vol. 41, No. 2, Feb., 1990

[HILLI93] HILLIS W. Daniel and Tucker Lewis W. The CM-5 Connection Machine: A Scalable Supercomputer. Communications of the ACM, November 1993, Vol. 36, No. 11.

[BACK77] BACKUS John. Can programming be liberated from the von Neumann style? Communications of the ACM, 21(8), 1977

[INTE05] Excerpts from A Conversation with Gordon Moore: Moore's Law. G Moore - Interview by Intel Corporation. 2005

[BARN11] BARNEY Blaise. Introduction to Parallel Computing. 2011 [En Línea] https://computing.llnl.gov/tutorials/parallel\_comp/

[COFF71] COFFMAN E. G., Elphick M, Shoshani A. System Deadlocks. Journal ACM Computing Surveys (CSUR) Volume 3 Issue 2, June 1971

[AMDA67] HL G. M, Validity of the Single-Processor Approach to Achieving Large Scale Computing Capabilities. In AFIPS Conference Proceedings, pages 483–485, April 1967

[HILL08] HILL Mark D. and Marty Michael R. Amdahl’s Law in the Multicore Era. IEEE Computer 2008

[FLYN72] FLYNN. M. Computer organizations and their effectiveness. IEEE Trans-actions on Computers, September 1972

[DUNC90] DUNCAN Ralph. A Survey of Parallel Computer Architectures. IEEE Computer Society Press. Vol 23, Issue 2, February 1990

[HWAN94] HWANG K., F. Briggs. Computer Architecture and Parallel Processing. McGraw-Hill. 1994

[FAN04] FAN Zhe, Qiu Feng, Kaufman Arie, and Yoakum-Stover Suzanne. GPU cluster for high performance computing. Proceedings of the 2004 ACM/IEEE Supercomputing Conference (SC'04), page 47059, November 2004

[MEND06] MÉNDEZ PALMA, T. José et all. Programación Concurrente. Thomson Editories. España. 2006

[CAND07] CANDELA Santiago. Fundamentos de los Sistemas Operativos. Thomson Editories. España. 2007

[BARR95] BARRENA GARCIA, Manuel. Técnicas de Particionamiento Multidimensional basadas en índices de múltiples atributos en base de datos paralelas. Tesis Doctoral. España 1995

[QUIL96] QUILES, Francisco J. Servicio de Publicaciones de La Universidad de Castilla – La Mancha. España. 1996

[BUYY99] BUYYA Rajkumar. High Performance Cluster Computing: Architectures and Systems, Volume 1. School of Computer Science and Software Engineering Monash University Melbourne, Australia. February 1999

[STER02] STERLING T. Beowolf Cluster Computing. The MIT Press. Cambrige, Massachusetts – London,England. 2002.

[TANE96] TANENBAUM, Andrew S. Sistemas operativos distribuidos. Prentice Hall Hispanoamericana. México. MX. 1996.

[SCHA02] SCHANTZ R. and Schmidt D. Research Advances in Middleware for Distributed Systems: State of the Art. In IFIP World Computer Congress, Kluwer Academic Publishers, pages 1-36, Canadá, August 2002.

[PFIS98] Greg PFISTER, In Search of Clusters, 2nd Edition, Prentice Hall 1998.

[SCHE98] Gang SCHEDULING. Timesharing on Parallel Computers, SC98, November 1998

[WALK99] WALKER, B., and Steel, D. Implementing a full single system image UnixWare cluster: Middleware vs. underware. Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications. 1999.

[MATE95] MATENA Vlada, Thadani Moti N, Shirriff Ken, Khalidi Yousef A. and Bernabeu Jose M.. Solaris MC: A Multi-Computer OS. November 1995

[GHOR98] GHORMLEY, D., Petrou, D., Rodrigues, S.,Vahdat, A., and Anderson, T. GLUnix: A global layer Unix for a network of workstations. Journal of Software Practice and Experience. 1998 <http://now.cs.berkeley.edu/Glunix/glunix.html>

[BARA98] BARAK, A., and La’adan, O. The MOSIX multicomputer operating system for high performance cluster computing. Journal of Future Generation Computer Systems. 1998 <http://www.mosix.cs.huji.ac.il/>.

[LOTTI05] LOTTIAUX, R.; Gallard, P.; Vallee, G.; Morin, C.; Boissinot, B.; IRISA, Rennes. OpenMosix, OpenSSI and Kerrighed: a comparative study. France. May 2005

[MORI04] MORIN Christine, Lottiaux Renaud, Valle Geoffroy, Gallard Pascal, Margery David, Berthou Jean-Yves, and Scherson Isaac. Kerrighed and data parallelism: Cluster computing on single system image operating systems. In Proceedings of Cluster 2004. IEEE, September 2004

[DONG83] DONGARRA J. Performance of various computers using standard linear equations software in a Fortran environment. ACM SIGARCH Computer Architecture News. Volume 11 Issue 5, December 1983

[BAIL86] BAILEY D, Barton J. The NAS Benchmark Program. Numerical Aerodynamic Simulations Systems Division. 1986. http://www.tjhsst.edu/~rlatimer/mpi/nas-doc.pdf

[BOUT06] BOUTEILLER A, Herault T, Krawezik G, Lemarinier P, Cappello F. MPICH-V Project: A Multiprotocol Automatic Fault-Tolerant MPI. International Journal of High Performance Computing Applications 2006; 20; 319

[DONG95] DONGARRA L. Introduction to the HPC Challenge Benchmark Suite. 1995. http://icl.cs.utk.edu/hpcc/pubs/

[NUMR07] NUMRICH R. A note on scaling the Linpack benchmark. Journal of Parallel and Distributed Computing. v.67 n.4. 2007

[CHEN10] CHEN S. et all. TPC-E vs. TPC-C: Characterizing the New TPC-E Benchmark via an I/O Comparison Study. ACM SIGMOD V39, i3. 2010

[GARC08] GARCIA J. Proyecto Fin de Carrera: Una técnica para la caracterización de nodos en Redes de Sensores Inalámbricas. Escuela Técnica Superior de Ingeniería de Telecomunicación de la Universidad Politécnica de Cartagena. Julio de 2008

[LILJ05] LILJA D. Measuring Computer Performance: A Practitioner's Guide. Cambridge University. Pages: 133-134. 2005

[BERS92] BERSHAD B, et all. Using micro benchmark to evaluate System Performance. Schooll of computer science Carnegie Mellon university. IEEE Comput. Soc. Press, Pages: 148-153. 1992

[LIU04] LIU L. et all. Microbenchmark performance comparison of high-speed cluster interconnects. IEEE Computer Society. 2004.

[CHEN93] CHEN Peter M. and Patterson David A. Storage Performance - Metrics and Benchmarks. Computer Science Division, Dept. of EECS. University of California, Berkeley. 1993

[CURN76] CURNOW H J and Wichmann B A. A synthetic benchmark. Central Computer Agency, Riverwalk House, London SW1P 4RT. National Physical Laboratory, Teddington, Middlesex TW11 OLW. Computer Journal, Vol 19, No 1, pp43-49. 1976

[SZEP11] SZEPKUTI I. Difference Sequence Compression of Multidimensional Databases. http://arxiv.org/abs/1103.3857 . 2011

[SPEC11] Standard Performance Evaluation Corporation. [En línea] http://www.spec.org. 28/09/2011

[TPCC11] Transaction Processing Performance Council .[En línea] http://www.tpc.org 2011

[BERR89] BERRY M. et all. The Perfect Club Benchmarks: Effective Performance Evaluation of Supercomputers. 1989

[OSDB10] The Open Source Database Benchmark. [En línea] http://osdb.sourceforge.net/ 22/12/2010

[NAPB10] NAS Parallel Benchmarks. [En línea] http://www.nas.nasa.gov/Resources/Software/npb.html 29/06/2010

[HPLB08] HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers. [En línea] http://www.netlib.org/benchmark/hpl/. 28/09/2008

[WALK05] WALKER Bruce J, Hewlett-Packard. Open Single System Image Linux Cluster Project. [En linea] http://openssi.org/ssi-intro.pdf. Dec 2005

[SSID11] OPENSSI Documentation. [En línea] http://openssi.org/cgi-bin/view?page=docs. 04/10/2011

[IDGN11] INSTALLING DEBIAN GNU/Linux 5.0.9 [En línea] http://www.debian.org/releases/lenny/debian-installer/ 03/11/2011

[GRUB11] GNU GRUB [En línea] http://www.gnu.org/software/grub/ 10/08/2011

[LILO10] LILO Bootloader for GNU/Linux. [En línea] http://lilo.alioth.debian.org/ 2010

[DONG97] DONGARRA J. J, Meuer H. W. and Strohmaier E, eds. TOP500 Report 1996, SUPERCOMPUTER , volumne 13, number 1, January 1997

[GRAY05] GRAY J. A “Measure of Transaction Processing” 20 Years Later. 2005.

[ZHOU99] ZHOU Y. et all. Fast Cluster Failover Using Virtual Memory-Mapped Communication. ICS Proceedings of the 13th international conference on Supercomputing 1999

[SHAN98] SHANLEY Kim. History and overview of TPC. [En línea] http://www.tpc.org/information/about/history.asp 1998

[TPCE10] TPC-Energy Specification V1.2.0. Transaction Performance Council. [En línea] http://www.tpc.org/tpc\_energy/default.asp 2010

[LIMA09] LIMA, Murilo Rodrigues de; Sunye, Marcos Sfair Execução distribuída de benchmarks em sistemas de bancos de dados relacionados. Universidade Federal do Paraná. Setor de Ciencias Exatas. Programa de Pós-Graduaçao em Informática. 2009

[TUVA02] HERNANDEZ J., Hernandez E., Llanos D. TPC-C UVA Implementación del Benchmark TPC-C. Escuela Universitaria Politécnica de Valladolid, Universidad de Valladolid, España, 2002

[GNPL07] GNU General Public License. Free Software. [En línea] Fundationhttp://www.gnu.org/copyleft/gpl.html 29/06/2007.

[XIAO06] XIAO W., Liu Y., Yang Q. K., Ren J., and Xie C. Implementation and performance evaluation of two snapshot methods on iSCSI target stores. In Proceedings of IEEE/NASA Conf. Mass Storage Systems, 2006.

[HILL06] HILLAR Gastón C. PostgreSQL 8.1.4 Robusto y fácil de administrar. Mundo Linux: Sólo programadores Linux, ISSN 1577-6883, Nº. 91, págs. 52-56, 2006

[CARRA08] CARRASQUILLA U. Capacity planning for virtualization and consolidation. Junio 2008.

[POESS00] POESS M., Floyd C. New TPC Benchmarks for Decision Support and Web Commerce. ACM Sigmod Record, 2000

[BRAG03] BRAGADO, Ignacio Martín. Gnuplot. Gráficos y dibujos desde la línea de comandos. Sólo programadores Linux, pags. 18-24, 2003.

[DBGE00] DBGEN, An object Relational Relational Tool. [En línea] http://dbgen.sourceforge.net/ 03/07/2000

[MENE05] MENÉNDEZ A., Pérez J., Sebastiá J. Red de sensores inalámbricos para monitorización de terrenos mediante tecnología IEEE 802.15.4. [En línea] http://w3.iec.csic.es/ursi/articulos\_modernos/articulos\_gandia\_2005/articulos/SC4/563.pdf 2005

[LINP08] THE LINPACK BENCHMARK. [En línea] http://people.sc.fsu.edu/~jburkardt/f\_src/linpack\_bench/linpack\_bench.html 07/03/2008.

[LAPA11] LAPACK: Lineal Algebra Package. [En línea] http://www.netlib.org/lapack/ 19/04/2011

Anexos

Instalación del Sistema Operativo Debian

La primera parte de la presente guía metodología para implementar una infraestructura clúster SSI consiste en instalar el nodo raiz. Para ello se utilizara la versión 5.0.8 de Debian, conocido como Lenny. Recomendada la version netinstall del mismo que se puede obtener desde la siguiente dirección.



**Netinstall** significa que únicamente el sistema base se encuentra en la imagen .iso, los demás servicios se podrán descargar de internet posteriormente. Se recomienda que la partición este vacía, si es la primera vez que va a realizar una instalación Linux. Grabe el ISO en un CD e inicie desde el cdrom en la terminal que va a instalar.



Fig. 10 Pantalla de instalacion

Instalación base de Debian

El proceso de la instalación se realiza en un modo consola. La terminal a instalar deberá de contar con dos placas de red, para mejor aislación entre la interconexión de nodos del cluster y la conexión a una red externa (internet). A continuación se detallan los pasos a pasos.

1. En la pantalla de Elegir Idioma, se selecciona “English”.
2. En la elección de País, se selecciona “United States”.
3. En la pantalla de seleccionar una configuración de teclado, se selecciona “American English”.
4. El hardware es detectado, los controladores y componentes se cargan, la red es ha configurada.
5. En la pantalla de configuración de red, se selecciona “eth0” como interfaz primaria de red.
6. En la pantalla de configuración de red, se selecciona como hostname “Lenny”
7. En la pantalla de configuración de red, se selecciona como nombre de dominio “clusterssi”.
8. Discos de almacenamientos son detectados y el proceso de partición es iniciado.
9. En la pantalla de partición de discos, se selecciona la opción “Manually edit partition table”
10. La partición se procederá de la siguiente manera. 1.0 Gb para la partición **/boot** con sistema de archivos *ext3*, 0.5 Gb para la partición **/swap** y el espacio sobrante del disco se asigna a la partición **/** con sistema de archivos *ext3.*
11. El sistema base de Debian ha sido instalado
12. Se consultara acerca de instalar el cargador GRUB en el registro de inicio maestro, se responde “Yes”.

Configuraciones de Debian

Una vez instalado el sistema base de Debian, se procede a realizar configuraciones básicas del sistema operativos.

1. En la pantalla de configuración de Huso Horario, se selecciona “No” cuando se consulta si el reloj interno del hardware esta configurado a GMT.
2. En la pantalla de configuración de Huso Horario, se selecciona “other” y luego “South America”, “Paraguay” como huso horario.
3. Se ingresa y verifica una contraseña para el administrador “root”.
4. Se crea un usuario regular.
5. En la pantalla de configuración de paquetes APT, se selecciona “No” cuando se consulta si se desea examinar otro CD.
6. En la pantalla de configuración de paquetes APT, se selecciona “Yes” cuando se consulta acerca de agregar otro repositorio APT. Seleccione http, luego “United States” y luego “http://ftp.us.debian.org”. No es necesaria información para proxy HTTP.
7. Las configuraciones están hechas.

Una vez reiniciada la terminal la siguiente pantalla se visualizará el inicio de sesión de usuarios.

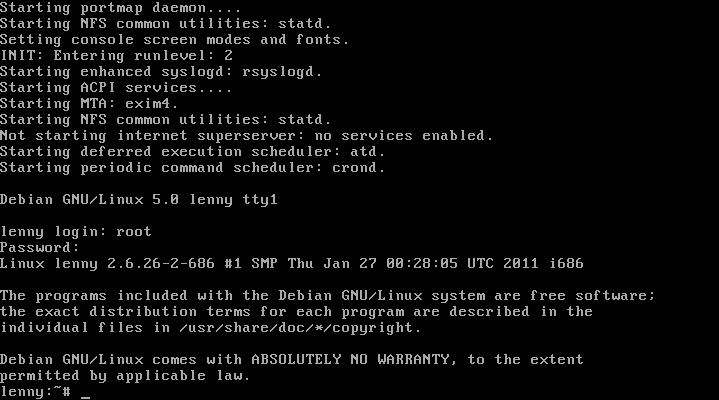


Fig. 11 Inicio de sesion

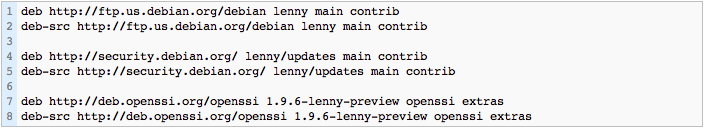
Instalación del nodo init o master

Posterior a la instalación del sistema operativo en la terminal que se desea usar como servidor o master (llamare master a partir de este momento) del clúster, se procede a las configuraciones para preparar al mismo. Se recomienda que se trabaje con únicamente la consola de Debian como entorno, ya que al instalar un entorno grafico, consumiría memoria ram, la cual puede ser utilizada de mejor manera en otros procesos del sistema.

Una vez iniciado el master con Debian Lenny instalado, se iniciara sesión como usuario administrador root, esto es para evitar escribir *sudo* a cada momento, luego de iniciada la sesión lo primero a realizar es la modificación del archivo “sources.list” de la siguiente manera.



Este se modificara para que quede de la siguiente forma:



Se crea el archivo “preferences” en la carpeta apt



Con la finalidad de que al momento de realizar cualquier tipo de instalación de un paquete, el sistema tenga preferencia por aquellos que se encuentre en el servidor de OpenSSI, se agregan las siguientes líneas:



Por ultimo, con el fin de autorizar los paquetes no autenticados se crea el archivo 90auth para autenticación.



Donde se ingresa la siguiente línea



Es importante que estos archivos contengan las líneas tal como se muestran expresadas para el correcta descarga e instalación de OpenSSI.

A continuación se ejecuta el comando *update* para actualizar repositorios y fuentes en Debian. Nota: si la salida de Internet es a través de un proxy se debe configurar primero, antes de continuar con los siguientes pasos.



Una vez realizada la operación anterior, se procede a instalar el paquete “initrd-tools”.



Se procede a desinstalar los siguientes paquetes:



Se procede a instalar el paquete “openbsd-inetd”



Se modificará el archivo de las interfaces de red:



Se agrega al final del archivo lo siguiente



Lo siguiente es agregar al modulo del NIC del grupo de interconexión al archivo modules, la tarjeta de red que el Master utilizara para comunicarse con los demás nodo, para obtener el NIC se usa la siguiente instrucción



La “X” es el numero de identificación de la tarjeta de red (para saber que numero tiene la tarjeta de red que deseas usar solo coloca “ifconfig” en la consola). En este caso, seria eth1. El NIC es el código que se encuentra al final de la línea después del ultimo “/” este código se debe agregar en la ultima línea en el siguiente archivo.



Una vez agregado el código, se procede a remover el kernel actual del sistema, para ello se debe hacer *modprobe* a los módulos necesarios para construir el *initrd.*

**ADVERTENCIA**: a partir de este punto el sistema no será iniciable, por lo tanto no se puede ni se debe apagar el master o cometer algún error, o sino será necesario comenzar de nuevo. Se realiza el modprobe de la siguiente forma:



Ahora se debe instalar el kernel OpenSSI pre-compilado y se remueve el kernel actual, para ello se utiliza.



**Nota I**: si pregunta acerca abortar la remoción del kernel, se responde ”no”

**Nota II**: si durante la instalación pregunta acerca “portmap” se contesta “I” (i latina)(sin las comillas).

Una vez instalado el nuevo kernel se verifica que el “udev” este en la versión 0.080-1



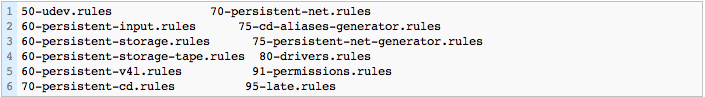
Si no se encuentra en esa versión, debe de ejecutarse:



Una vez concluida esta parte, se debe configurar el fichero /etc/udev/rules.d empezando asi:



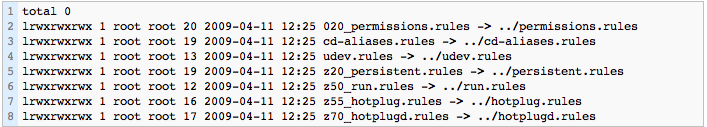
Una lista similar a esta aparecerá



Para solucionar se deben de ejecutar las siguientes líneas:



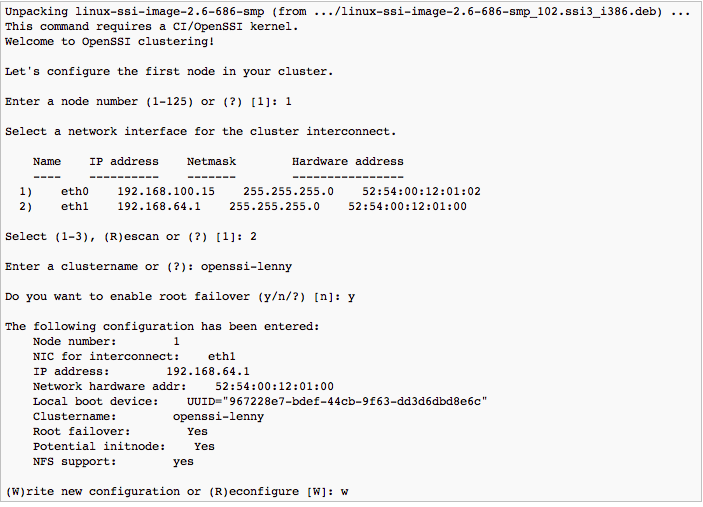
Obteniendo el siguiente resultado.



El paso a continuación es instalar openssi en el sistema con la siguiente instrucción.



En caso de que al finalizar la instalación, no se soliciten configuraciones del master, ejecutar el comando *ssi-create*.



Estas son las configuraciones del OpenSSI para el master. Recuerde que la tarjeta de red previamente configurada para la interconexión es la que se debe utilizar. El siguiente paso es instalar un kernel que corregirá el problema de inicialización del sistema operativo.

Para instalar un kernel de 32 bits se ejecuta



O para instalar un kernel de 64 bits se ejecuta



Lo siguiente a realizar es reiniciar el master usando el comando *init 6.* Una vez reiniciado el master con el kernel OpenSSI, el master se vuelve un cluster de un nodo, por el momento, posteriormente se agregara mas nodos. El siguiente paso a realizar es modificar el archivo dhcpd.proto:



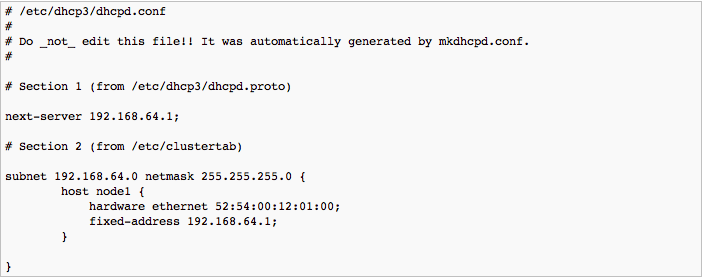
Se agrega al final del archivo la siguiente línea:



Luego se debe ejecutar el siguiente comando *mkdhcpd.conf*, se verifica que se haya generado el archivo dhcpd.conf



Debería de mostrar algo similar a lo siguiente:



Se debe de reiniciar el proceso de dhcp3-server:



Ahora se debe asegurar de enlazar los directorios “/tftpboot” y “/var/lib/tftpboot” para ello se ejecuta:



Lo cual debe de retornar en consola *total 0.* Luego se ejecuta:



Se obtiene:



A partir de este momento se pueden agregar nodos a la infraestructura OpenSSI.

Agregar nodos a la infraestructura.

Una vez instalado el sistema operativo en el nodo master, instalado el kernel openSSI en el mismo, se procede a agregar nodos para poder contar con una infraestructura Cluster. Para ello se deben realizar una serie de pasos y chequeos iniciando por la verificación si las tarjetas de red de los nodos esclavos no poseen soporte de booteo PXE. Se puede descargar un iso desde la siguiente url http://rom-o-matic.net/gpxe/gpxe-1.0.1/contrib/rom-o-matic/ Debido a que usaremos PXE, no es necesaria una imagen Etherboot.

Si el NIC de la tarjeta de red de los nodos esclavos no se encuentra en el archivo “/etc/mkinitrd/modules” del master. Se debe editar este archivo



Agregar el nombre del driver al final del archivo y luego reconstruir el ramdisk para contener los módulos necesarios para el inicio, debido a que este mismo ramdisk es el que se envía después a los nodos esclavos la imagen del SO. Para generar de nuevo el ramdisk se ejecutan los siguientes comandos:



Debido a que durante la instalación del sistema operativo al nodo master, se ingresó la opción noapic, es necesario realizar una modificación dentro del archivo /tftpboot/pxelinux.cfg/default.



Se agrega “noapic” antes del initrd de la siguiente manera:



Una vez hechas estas configuraciones en el nodo master. Se procede a iniciar el nodo esclavo con la imagen ISO gPXE. En el nodo esclavo, se podrá apreciar como la imagen gPXE se inicia y busca algún servidor DHCP, presionando de manera combinada las teclas CTRL + B se accede a la consola, al ingresar el comando *autoboot* se obtendrá la dirección MAC de la tarjeta de red que se utilizará para la interconexión con el cluster.

En el nodo master se ejecuta el comando *ssi-addnode* de la siguiente manera:



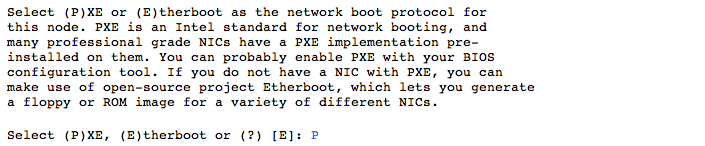
En donde MACADDRESS es la dirección MAC del nodo esclavo que se encuentra ejecutando gPXE. Al ejecutar el comando *ssi-addnode* se realizaran algunas preguntas acerca de cómo se desea configurar el nuevo nodo y que son los siguientes.



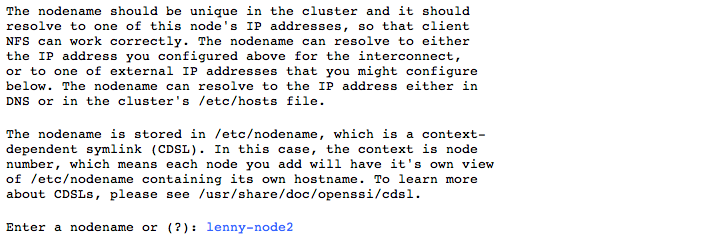
Se ingresa “Y”



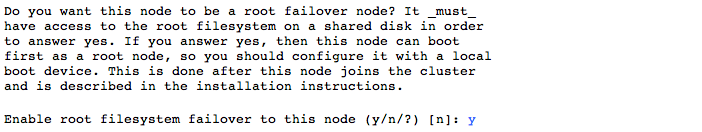
Se ingresa “2”



Se ingresa “P”



Se ingresa “Lenny-node2” o el nombre que se escoja para el nodo.

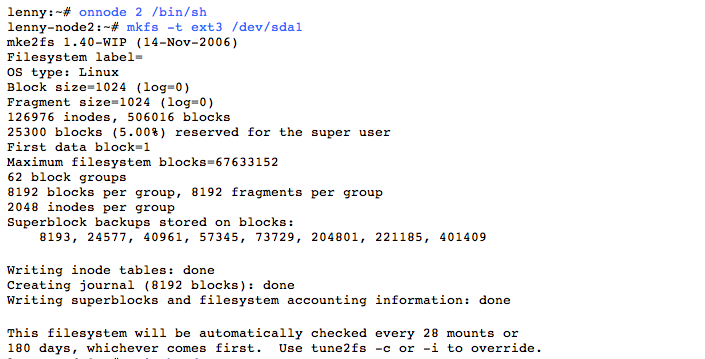


Si en el master se activo la tolerancia a fallos, se ingresa aquí “Y”. Una vez guardadas estas configuraciones, el nodo esclavo se unira al cluster por red.

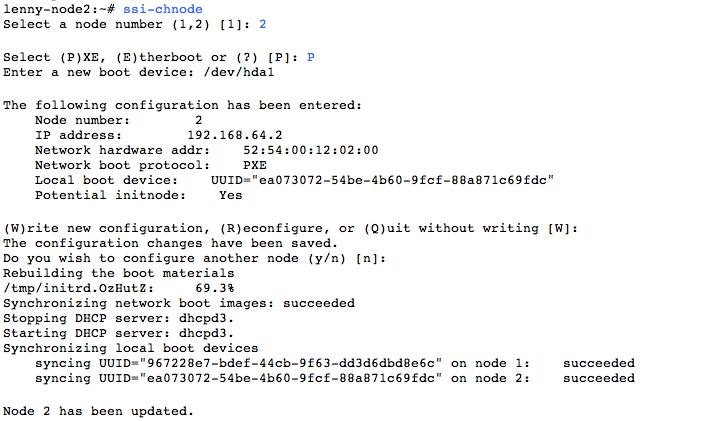
Para comprobar el estado del cluster ejecutar el siguiente comando



Para la tolerancia de falla en el cluster, se deben realizar los siguientes pasos.



Luego se ejecuta el comando *ssi-chnode*



A partir de este momento, el nodo puede iniciar desde su propio disco. Repita los pasos para agregar mas nodos al cluster OpenSSI.

Monitorización visual del clúster OpenSSI

Una vez lograda la instalación del nodo master, y agregado nodos esclavos, el siguiente paso es la instalación de la herramienta de monitoreo OpenSSI webView. OpenSSI webView es una herramienta PHP desarrollada por Kilian Cavalotti y extendida por el propio equipo de la tesis. Se encuentra disponible es esta dirección <http://darthvinsus.github.com/webview/>

La herramienta permite visualizar la carga en cada nodo del clúster, monitorizar el estado de cada nodo y ver las graficas con diversas estadísticas. También es posible monitorizar los procesos en cada nodo y migrar manualmente un proceso a otro nodo. Para utilizar esta herramienta es necesario realizar los siguientes pasos.

1. Extraer directamente en la carpeta del servidor el archivo descargado desde la dirección <http://darthvinsus.github.com/webview/>

*# cd <DESTDIR>*

*# tar xfvj DarthVinsus-webview-1902726.tar.gz*

1. Establecer los permisos adecuados en el directorio de gráficos, para la generación de gráficos y la recolección de datos en el archivo config.php, para poder modificar y guardar la configuración desde la interfaz web.

*# cd <DESTDIR>/openssi-webview*

*# chown -R <USER> graphs*

*# chown <USER> config.php*

1. Agregar al cron la tarea de mantener actualizada la interfaz a través de la recolección de datos por medio de un script.

*\*/5 \* \* \* \* <USER> [ -d <DESTDIR> ] && (cd <DESTDIR> && ./graphs/update\_all.sh )*

1. Habilitar la migración de nodos por medio de la interfaz web, para ello modificar el archivo *etc/sudoers* y permitir la migración de procesos a todos los usuarios.

*<USER> ALL = NOPASSWD:/usr/bin/migrate*

Una vez realizado los pasos, acceda desde algún navegador web de su preferencia a la dirección *http:/IP\_NODO\_MASTER/webview*, donde IP\_NODO\_MASTER es la dirección IP del nodo master, configurado en la sección

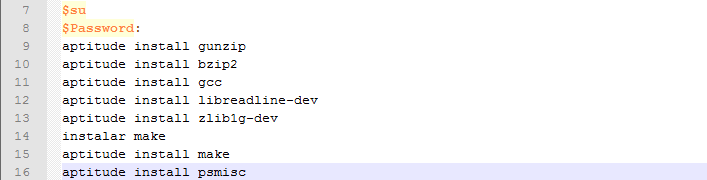
Instalación de Postgres SQL

Una vez instalado el sistema operativo con las herramientas necesarias se procede a instalar el motor de banco de datos Postgres en su versión 8.3

Inicialmente se obtiene el paquete provisto entre los utilitarios de la Universidad de Valladolid mediante la ayuda del comando wget.



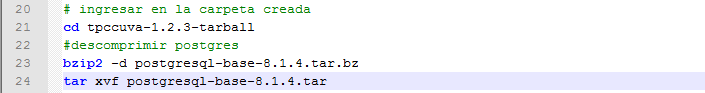
A continuación como usuario root se debe de instalar todos los paquetes necesarios para continuar.



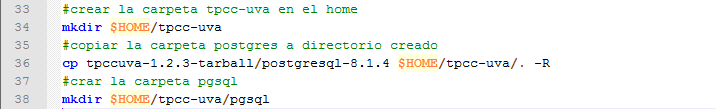
Luego es necesario ingresar como un usuario distinto al root y continuar la descompresión del paquete anteriormente obtenido.



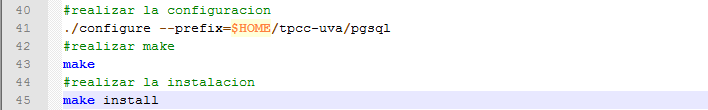
Seguidamente se ingresa a la carpeta descomprimida y se repite la tarea de descomprimir el archivo relacionado a postgres sql



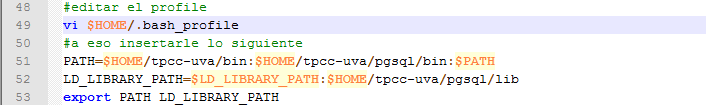
Luego se crea las carpetas necesarias para la instalación del postgres sql.



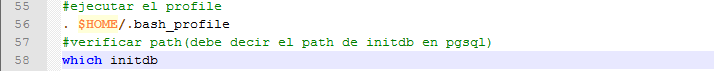
El paso siguiente consiste en realizar la configuración e instalación del postgres



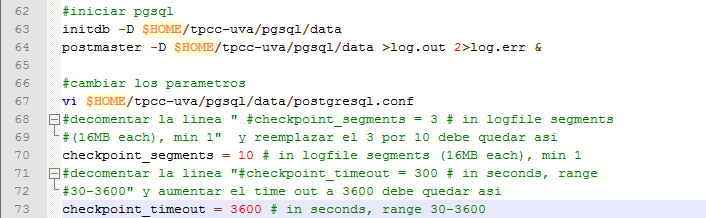
Seguidamente es necesario editar el perfil para que se considere las configuraciones realizadas



Luego de guardar y salir del archivo, ejecutar el perfil creado y verificar su estado.



Una vez realizada la instalación se debe iniciar el postgres sql y cambiar los parámetros del archivo de configuración.



Seguidamente se debe de forzar la salida para que lea el nuevo archivo de configuración.



El postgres SQL se encuentra listo para ser ejecutado.

Siempre que se requiera levantar nuevamente la instancia se debe ejecutar



Y para finalizar la instancia se debe realizar



Instalación de la suite TPCC-UVA

Una vez instalado el postgres SQL se debe crear las carpetas necesarias



Seguidamente editamos las variables en el archivo Makefile

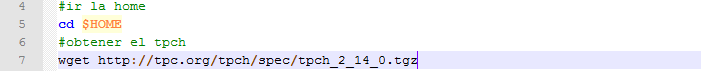


Y finalmente con ayuda del comando make complementamos la instalación

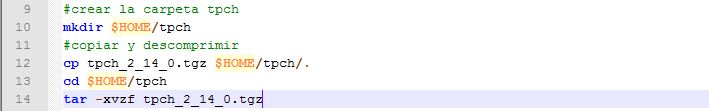


Instalación del Benchmark TPC-h

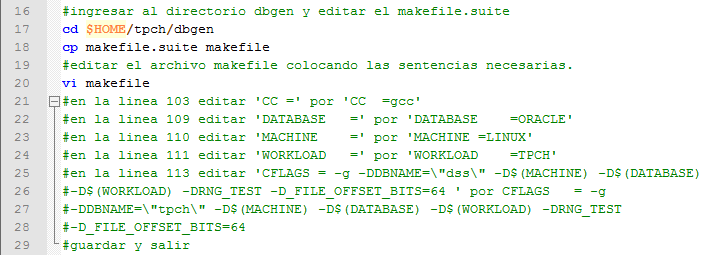
Luego de tener instalado el Postgres SQL se procede a instalar el benchmark TPC-H, para ello primeramente ingresar al home y obtener los paquetes necesarios



Seguidamente crear la carpeta que contendrá al benchmark, copiar el archivo obtenido y descomprimir en ese lugar.



Luego ingresar al directorio dbgen y editar el archivo makefile



Luego de guardar y salir del archivo, realizar la instalación con el comando make.



Finalmente solo se debe crear la base de datos de pruebas con los script utilitarios que son provistos.

Script para instanciar el Postgres para pruebas OLTP y OLAP

Siempre que se requiera levantar nuevamente la instancia se debe ejecutar



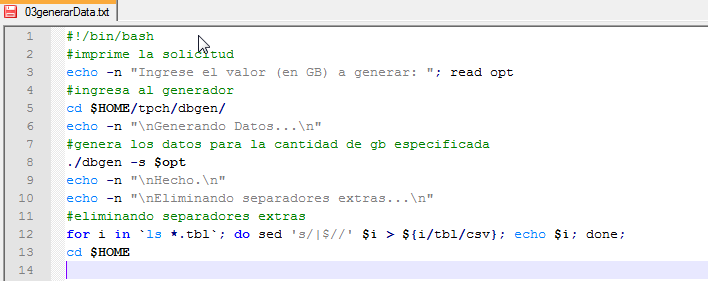
Y para finalizar la instancia se debe realizar



Scripts utilitarios para pruebas OLAP

A continuación se definen los script utilitarios para ser utilizados como soporte en las pruebas OLAP realizadas con el benchmark TPC-H

1. Para la creación de nuevas bases de datos indicando el tamaño de las mismas se recomienda utilizar el script 03generarData.txt que crea un archivo .csv de datos generados por el utilitario dbgen instalado en el benchmark TPC-H y está compuesto por:

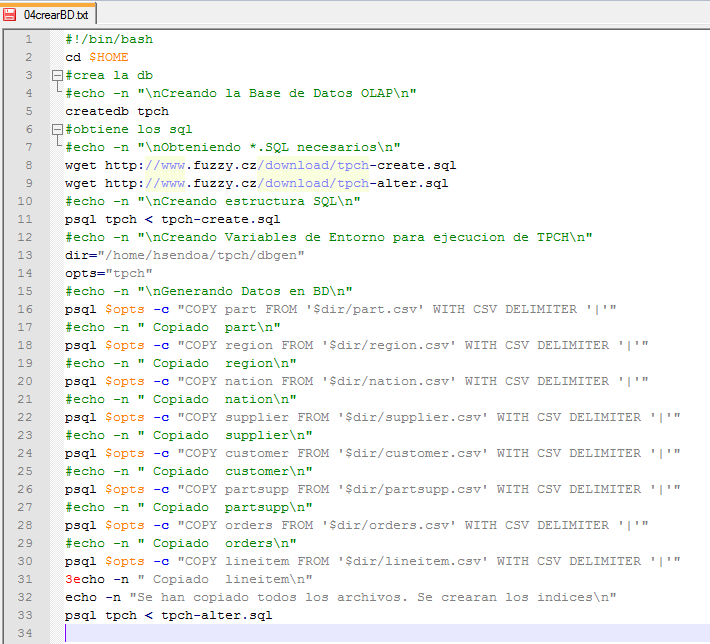


El mismo puede ser ejecutado mediante consola de la siguiente manera,



El cual solicita ingresar un valor entero correspondiente a la base de datos a ser generada.

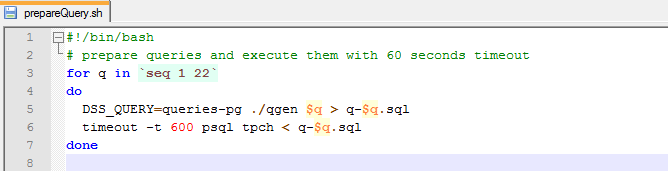
1. Una vez realizado los archivos .csv es necesario crear la estructura de tablas, migrar los datos y realizar los ajustes necesarios para ejecutar el benchmark TPC-H. Para ello es provisto el siguiente script contenido en 04crearBD.txt:



El mismo puede ser ejecutado con el siguiente comando



1. La ejecución del benchmark TPC-H requiere preparar las consultas antes de iniciar las pruebas. Para ello tenemos disponible es script siguiente:



1. Las ejecuciones se realizan mediante el siguiente script:

