DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

TESIS PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS

AUTOR: JUAN DANIEL CARRILLO

TEMA: "IMPLEMENTACIÓN DE UN CLUSTER COMPUTACIONAL UTILIZANDO HERRAMIENTAS DE SOFTWARE LIBRE PARA EL DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE"

DIRECTOR: ING. DIEGO MARCILLO

CODIRECTOR: ING. CARLOS CAIZAGUANO

SANGOLQUÍ, FEBRERO 2015

CERTIFICACIÓN

Certifico que la presente tesis titulada “IMPLEMENTACIÓN DE UN CLUSTER COMPUTACIONAL UTILIZANDO HERRAMIENTAS DE SOFTWARE LIBRE PARA EL DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE” fue realizada en su totalidad por el señor JUAN DANIEL CARRILLO, ha sido guiada y revisada periódicamente y cumple normas estatutarias establecidas por la UNIVERSIDAD DE LAS FUERZAS ARMADAS -ESPE, el presente trabajo es un requerimiento para la obtención del título de INGENIERO EN SISTEMAS.

Sangolquí, febrero de 2015

ING. DIEGO MARCILLO
DIRECTOR

ING. CARLOS CAIZAGUANO
CODIRECTOR

DECLARACIÓN DE RESPONSABILIDAD

La presente tesis está fundamentada bajo la investigación bibliográfica respetando los derechos de autor de terceros, cuyas fuentes se incorporan en la bibliografía, los conceptos desarrollados, análisis realizados y las conclusiones del presente trabajo, son de exclusiva responsabilidad del autor.

Sangolquí, febrero de 2015

Juan Daniel Carrillo

AUTORIZACIÓN

El suscrito Juan Daniel Carrillo, con cédula, autor del trabajo de tesis titulado: “Implementación de un Cluster Computacional utilizando herramientas de Software Libre para el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE” para que se publique con fines académicos y se visibilice su contenido respetando los derechos de autor.

Sangolquí, febrero de 2015

Juan Daniel Carrillo

DEDICATORIA

A mis padres maravillosos María Clemencia y Juan Manuel que son el eje fundamental de mi vida, que con su dedicación y trabajo han logrado apoyarme en la culminación de mi formación académica.

A mi hermana Gabriela por su amor filial y por ser la gestora para la culminación de mi meta trazada.

A mi novia y futura esposa Milly por brindarme su amor, comprensión y estimulo incondicional.

A mi abuelita Ana María por el cariño, paciencia y esfuerzo de toda la vida.

A mis amigos que siempre han estado a mi lado en cualquier circunstancia; y en especial a mis grandes amigos Fabricio y Paúl que desde el cielo me están viendo triunfar.

A mis maestros que supieron transmitir sus conocimientos y aprendí muchas cosas día a día y las comparto con los demás.

Al ingeniero Diego Marcillo, mi director de tesis por darme la iniciativa en un tema tan interesante e innovador y por el apoyo y la confianza que depositó en mí.

A Matteo Ragni, un agradecimiento especial, ya que al ser una persona desconocida físicamente, colaboró con unos detalles del empleo de un prototipo de interfaz gráfica para el uso de TORQUE, desarrollado por él, y a quien también pude apoyar en una corrección de mismo.

Con respeto, gratitud, admiración y amor.

Juan Daniel Carrillo

AGRADECIMIENTO

A Dios por haberme permitido culminar con éxito esta etapa de mi vida.

A mí querida Universidad de las Fuerzas Armadas-ESPE, que por muchos años me acogió en su seno como estudiante y supo fortalecer en mí los valores y principios morales que rigen mi vida personal y profesional.

A mis maestros, que a través de sus conocimientos, enseñanzas y experiencias supieron transmitir los conocimientos para mi vida profesional.

Un especial agradecimiento a los Ingenieros Diego Marcillo y Carlos Caizaguano, quienes con paciencia y sabiduría supieron guiarme en el desarrollo de este trabajo.

Al Departamento de Ciencias de la Computación, que me brindo todas las facilidades para la realización de esta tesis.

ÍNDICE

ÍNDICE DE GRÁFICOS

ÍNDICE DE TABLAS

RESUMEN

El proyecto de investigación desarrollado en esta tesis es la implementación de un cluster computacional utilizando herramientas de software libre para el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE. Para lograr este objetivo fue necesario analizar, implementar y evaluar distintas arquitecturas de procesamiento paralelo que permitan resolver problemas complejos utilizando un conjunto de equipos independientes o nodos que a través de un programa se integran para proporcionar una capacidad de procesamiento y almacenamiento equivalente a un supercomputador pero con una inversión económica mínima ya que se reutilizó computadores existentes en el Data Center del Departamento. En el capítulo 1 se describe los aspectos más importantes de la planificación inicial del proyecto. En el capítulo 2 se desarrolla el marco teórico que incluye definiciones básicas, descripción de arquitecturas y bibliotecas de procesamiento paralelo; así como lo que es un cluster y los tipos que existen. En el capítulo 3 se realiza el diseño del cluster que describe el tipo de cluster, el hardware y software a ser utilizado y se presenta la implementación del cluster, detallando los pasos necesarios para la instalación y configuración. Finalmente en el capítulo 4 se presentan las conclusiones y recomendaciones obtenidas en el proyecto.

Palabras Clave: CLUSTER, NODOS, SOFTWARE LIBRE, PROCESAMIENTO PARALELO, ROCKS CLUSTER.

ABSTRACT

The research project developed in this thesis is the implementation of a computing cluster using free software tools for the Computer Science Department of The Army Forces University-ESPE. To achieve this goal it was necessary to analyze, implement and evaluate various parallel processing architectures to solve complex problems using a set of independent computers or nodes through an integrated program to provide processing and storage capacity equivalent to a supercomputer but with minimal financial investment because existing computers in the Data Center of the Department was reused. In Chapter 1 the most important aspects of the initial project planning is described. In chapter 2 the theoretical framework that includes basic definitions, description of architectures and parallel processing libraries also what type of cluster was used. In chapter 3 the design of the cluster describing the type of cluster, hardware and software to be used is performed and the implementation of the cluster is presented, detailing the steps for installation and configuration. Finally in chapter 4 the conclusions and recommendations obtained in the project are presented.

Keywords: CLUSTER, NODES, FREE SOFTWARE, PARALLEL PROCESSING, ROCKS CLUSTER.

CAPÍTULO 1

1.1 Introducción

Los avances y cambios tecnológicos son continuos y acelerados en todos los campos y en especial en las Tecnologías de Información y Comunicación (TIC´s), este escenario hace que los sistemas informáticos requieran mayores recursos de procesamiento, almacenamiento y comunicaciones con el objetivo de satisfacer las necesidades relacionadas con altos volúmenes de cálculos matemáticos, procesamiento de imágenes, diseño y ejecución de juegos y otros procesos que las empresas, instituciones educativas, centros de investigación científica puedan requerir.

Para cubrir estas necesidades, se plantea una solución tecnológica conocida como Clustering Computacional con el uso de herramientas de software libre que tiene costos relativamente bajos en comparación con otro tipo de soluciones. El hardware y software de esta propuesta se implementará en el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE. El desarrollo del proyecto se iniciará con la revisión teórica y bibliográfica del Clustering Computacional y procesamiento paralelo, a continuación se realizará el diseño de la solución y luego la instalación y pruebas de la infraestructura así como del procesamiento paralelo.

1.2 El Problema

1.2.1 Planteamiento del Problema

El mundo siempre ha estado rodeado de problemas complejos que requieren ser resueltos de manera rápida y exacta. La tecnología ha contribuido para cumplir este propósito al desarrollar máquinas y programas capaces de realizar cálculos de varios millones de operaciones por segundo. Hoy por hoy se necesita procesar gran cantidad de información para resolver esos problemas en varios ámbitos como: el científico, industrial, comercial, etc.

En la actualidad existen equipos denominados supercomputadores que son capaces de procesar gran cantidad de datos y transformarlos en información útil para empresas o instituciones, sin embargo, muchas de ellas no disponen de recursos económicos para adquirir esos equipos que tienen un costo de algunos millones de dólares.

Empresas, escuelas, centros científicos, universidades, entre otras, cada día enfrentan problemas que requieren procesamiento de alto desempeño. En este caso la Universidad de las Fuerzas Armadas-ESPE requiere de un procesamiento de alto desempeño, ya que los estudiantes, personal docente y administrativo de la matriz, sedes, Unidades Académicas Externas y Escuelas Especiales utilizan los servicios, lo que demanda un proceso intensivo de datos, que exige contar con servidores de gran capacidad y una mayor infraestructura que garantice brindar servicios de calidad.

La evolución tecnológica implica nuevos cambios en las arquitecturas de los equipos informáticos, como nuevas aplicaciones, que necesita el usuario y ese desarrollo de nuevas aplicaciones impliquen el uso de procesos en paralelo. A pesar de esta progresiva evolución tecnológica, el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE no aplica tal conocimiento a los estudiantes que cursan la ingeniería de Sistemas e Informática, dejando un vacío o desconocimiento del funcionamiento y la utilidad del procesamiento en paralelo o multiproceso que van de la mano, tanto en el nuevo desarrollo de aplicaciones, como con los equipos computacionales que permitan realizar el multiproceso a nivel de software como de hardware.

1.2.2 Formulación Del Problema

Una vez planteado el problema, se procederá a la formulación de varias preguntas y respuestas, producto del desarrollo del presente tema de tesis.

¿Cómo incide el bajo rendimiento y la entrega de servicio de los servidores a todo el personal docente, administrativo y estudiantes que conforman el

Departamento de Ciencias de la Computación y en general a la Universidad de las Fuerzas Armadas-ESPE?

1.2.3 Interrogantes De La Investigación

- ¿Por qué diseñar e implementar un Cluster en lugar de adquirir una supercomputadora?

Sencillamente por costos.

- ¿Cómo funciona y para qué es importante el procesamiento paralelo?

El procesamiento paralelo funciona de tal manera que un proceso sea analizado por múltiples procesadores que operan simultáneamente y es importante para obtener cálculos y operaciones de manera eficiente a un menor costo-tiempo.

- ¿Es necesario que la Universidad de las Fuerzas Armadas-ESPE compre nuevos y mejores servidores para solventar las deficiencias de almacenamiento, memoria y procesamiento?

No, no es necesario porque se podría utilizar los equipos informáticos que no se utilizan e implementar la solución de Clustering.

La solución es analizar la necesidad del servicio que la Universidad de las Fuerzas Armadas-ESPE requiere y proponer una solución que se basa en reunir las computadoras e implementar un Clustering de acuerdo a esa necesidad institucional, sin generar más gastos con la adquisición de nuevos equipos.

- ¿Cómo el Cluster Computacional puede ayudar a estudiantes y/o personal de otros Departamentos?

Puede apoyar para diferentes áreas de estudio y/o trabajo, de acuerdo a las aplicaciones y servicios que ofrece el Cluster Computacional, es decir resolución de varios problemas que requieran de alto procesamiento para distintas tareas, aplicando paralelismo a esas tareas, aplicaciones o servicios.

- ¿Se requiere un diseño y pruebas para la implementación del Cluster Computacional?

Sí, se requiere realizar un diseño sobre el tipo de Cluster que se implementará, de acuerdo a la necesidad institucional y se realizarán las pruebas necesarias sobre Cluster, para determinar su óptimo funcionamiento y rendimiento.

1.3 Objetivos

1.3.1 Objetivo General

Diseñar e Implementar un Cluster Computacional, mediante la utilización de herramientas de software libre, para mejorar el rendimiento de los servicios que requieren alto procesamiento paralelo para el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE.

1.3.2 Objetivos Específicos

- Determinar la posibilidad de construir un Cluster con hardware disponible que permita tener resultados óptimos a menor costo.
- Determinar el funcionamiento y la importancia del procesamiento en paralelo.
- Proponer la solución de “Clustering” a sistemas que requieran capacidad de procesamiento, almacenamiento y memoria en el Departamento de Ciencias de la Computación.
- Contribuir con la formación de los estudiantes, mediante la aplicación del paralelismo en Cluster en la solución de problemas en áreas específicas como procesamiento de imágenes, análisis numérico, biocómputo y otros.
- Diseñar, implementar y realizar pruebas del Cluster Computacional que se determine previamente.

1.4 Justificación

La Universidad de las Fuerzas Armadas-ESPE, como una Institución de Educación Superior, con diferentes sedes a nivel nacional, con autonomía

administrativa y patrimonio propio, de derecho público al servicio de la sociedad enfrenta retos diariamente, ya que se encuentra en cambio continuo por el ingreso de nuevos estudiantes, personal docente y administrativo que hacen uso de la infraestructura y recursos de la misma institución. Es por ello que en momentos determinados, los sistemas informáticos se han saturado por la carga de información concurrente que diariamente se exige a los servidores. Como consecuencia se han visto afectados los usuarios de los servicios (estudiantes, personal docente y administrativo).

Por lo tanto y para evitar estos inconvenientes existe la necesidad de emprender un proyecto de implementación de Cluster Computacional con distribución Linux, bajo licenciamiento de software libre, que permitirá liberar la carga que soportan los servidores invirtiendo en un equipo de características potentes que será denominado **"nodo principal"** el cual será el encargado de distribuir todos los procesos a los "nodos secundarios" que compongan el Cluster Computacional; nodos secundarios serán varios computadores disponibles que serán preparados para realizar varias tareas asignadas, esto incidirá en cambiar y mejorar el rendimiento de los servicios que requieren alto procesamiento, memoria y almacenamiento con el propósito de que, inicialmente, el Departamento de Ciencias de la Computación cuente con esta solución tecnológica para realizar investigación y solventar problemas de los servicios computacionales.

Se ha estimado un tiempo de seis meses para la implementación; y de esta manera mejorar el cambio en una parte de la infraestructura informática, el mismo que conllevará inicialmente al análisis y determinación del Cluster que requieren los servicios, para posteriormente proceder a realizar la implementación y pruebas de esa solución tecnológica.

1.5 Alcance

El presente proyecto pretende servir como base para el estudio del procesamiento paralelo. Además el proyecto se plantea para apoyar y asesorar otras carreras y departamentos de la Universidad de las Fuerzas Armadas-

ESPE al inicio será utilizado el Departamento de Ciencias de la Computación, que por sus características requiere procesamientos de alto desempeño.

Se utilizarán herramientas de software libre para lograr la implementación, con pruebas de rendimiento y funcionamiento del Cluster Computacional, además se documentará los procesos y conceptos básicos de computación científica y Clusters.

1.6 Hipótesis

La implementación del Cluster Computacional mejorará el rendimiento de los servicios que requieren alto procesamiento paralelo, utilizando herramientas de software libre para el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE.

1.7 Metodología

Para el proyecto presentado se utilizó la metodología de la investigación aplicada, ya que se aplicarán los conocimientos adquiridos y detallados en el marco teórico, para lo cual previamente se recopiló información referente a este tema.

A continuación se describen los pasos a seguir:

- Investigar la arquitectura del Clustering Computacional.
- Diseñar e implementar un Cluster Computacional con herramientas de software libre.
- Comprobar y evaluar los resultados experimentales.

1.8 Factibilidad

1.8.1 Factibilidad Técnica

Para la realización del presente proyecto **"Implementación de un Cluster Computacional utilizando herramientas libres para el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE"**, se contará con el hardware necesario que será proporcionado por el Departamento de Ciencias de la Computación y con herramientas de software

libre que colaboren con la implementación. Además se dispondrá con material de apoyo, como libros relacionados al Clustering Computacional y la guía del Ingeniero Diego Marcillo, poseedor de vastos conocimientos en el área Sistemas Informáticos.

1.8.2 Factibilidad Económica

El proyecto es económicamente viable, ya que no requiere de adquisición de equipos para su realización.

Tabla 1. Factibilidad Económica

	DESCRIPCIÓN	CANTIDAD	COSTO
HARDWARE	NODO PRINCIPAL (SERVIDOR)	1	Disponibilidad de equipo
	DESCRIPCIÓN	CANTIDAD	COSTO
	NODOS SECUNDARIOS (PC´S ESCRITORIO)	16	Disponibilidad de equipos

	CLUSTER	TIPO	SISTEMA OPERATIVO BASE	COSTO
SOFTWARE	*ROCKS CLUSTERS*	DISTRIBUCIÓN LINUX	CENTOS 6.2	NINGUNO

Fuente: Investigación
Elaborado por: Juan Daniel Carrillo

1.8.3 Factibilidad Operativa

Los recursos tecnológicos, como hardware, serán provistos por el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE, además del espacio físico en donde se desarrollará el proyecto de Clustering Computacional.

La información adicional que sea necesaria para este proyecto, se obtendrá del material de apoyo como libros, Internet y la experiencia por parte de docentes y colegas que poseen sólidos conocimientos en el tema.

1.9 Recursos

1.9.1 Recursos De Hardware

Tabla 2. Recursos de Hardware

	DESCRIPCIÓN		CANTIDAD	REQUERIMIENTO MÍNIMO
HARDWARE	**NODO PRINCIPAL (SERVIDOR)**		**1**	
		PROCESADOR:	1	INTEL XEON QUAD CORE
		DISCO DURO:	1	SCSI 140 GB
		TARJETA DE RED:	2	Gigabit ethernet
		Memoria RAM:	X	2 GB
	DESCRIPCIÓN		**CANTIDAD**	**REQUERIMIENTO MÍNIMO**
	NODOS SECUNDARIOS (PC´S ESCRITORIO)		**16**	
		PROCESADOR:	1	Intel Core 2 Quad
		DISCO DURO:	1	500 GB
		TARJETA DE RED:	1	Gigabit ethernet
		MEMORIA RAM:	X	4GB

Fuente: Investigación
Elaborado por: Juan Daniel Carrillo

1.9.2 Recursos De Software

Tabla 3. Recursos de Software

	CLUSTER	TIPO	SISTEMA OPERATIVO BASE
SOFTWARE	*ROCKS CLUSTERS*	DISTRIBUCIÓN LINUX	CENTOS 6.2

Fuente: Investigación
Elaborado por: Juan Daniel Carrillo

*Tentativo a utilizar herramientas de software libre para la operacionalización de servicios adicionales en el Cluster.

1.9.3 Recurso Humano

Tabla 4. Recurso Humano

Nombres:	Juan Daniel
Apellidos:	Carrillo Medrano
Dirección Domicilio:	Antonio de Ulloa N25-61 y Luis Mosquera
Teléfono Domicilio:	2236986
Teléfono Celular:	0999732086
Correo Electrónico:	jcm159@gmail.com

Elaborador por: Juan Daniel Carrillo

CAPÍTULO 2

2.1 Marco Teórico

2.2 Conceptos Generales

2.2.1 Arquitectura de computadores

Imaginar la vida sin una computadora en la actualidad resulta difícil, con el crecimiento de las telecomunicaciones y la llegada del internet, es inevitable no contar con una computadora en el hogar, las oficinas, las instituciones de educación en todos los niveles y en la industria; de igual forma es importante orientar y priorizar su uso ya que tiene influencia directa en los procesos de análisis, síntesis y producción de bienes y servicios.

El uso del computador de escritorio, personal o supercomputador han permitido resolver problemas complejos, dar solución a necesidades de información y contribuir a la toma de decisiones. Los archivadores convencionales han sido reemplazados por dispositivos de almacenamiento de menor tamaño, mayor capacidad y menor consumo energético, logrando de esta manera disponer de la información de forma rápida y eficiente. Con la llegada del internet la comunicación ha tenido una gran evolución y su crecimiento ha sido exponencial, tanto así que en la actualidad es difícil encontrar una persona que no esté familiarizada con su uso, ya sea por su trabajo, educación o por sus necesidades de comunicación.

Historia de las computadoras

El computador no es un invento de alguien en particular, sino el resultado evolutivo de muchas ideas relacionadas con la electrónica, la mecánica, los materiales semiconductores, la lógica, el álgebra y la programación.

Primera Generación (1946 – 1958)

En esa época las computadoras funcionaban con válvulas, usaban tarjetas perforadas para ingresar los datos y los programas, utilizaban cilindros

magnéticos para almacenar información e instrucciones internas y se utilizaban exclusivamente en el ámbito científico o militar, utilizaban gran cantidad de electricidad, generaban gran cantidad de calor y eran sumamente lentas.

- 1946 ENIAC. Primera computadora digital electrónica en la historia. No fue un modelo de producción, sino una máquina experimental. Se trataba de un enorme aparato que ocupaba todo un sótano, construida con 18.000 tubos de vacío que consumía varios KW de potencia eléctrica y pesaba algunas toneladas. Era capaz de efectuar cinco mil sumas por segundo y fue creada por un equipo de ingenieros y científicos encabezados por los doctores John W. Mauchly y J. Presper Eckert en la universidad de Pensilvania, en los Estados Unidos.
- 1949 EDVAC. Segunda computadora programable. También fue un prototipo de laboratorio, pero ya incluía en su diseño las ideas centrales que conforman las computadoras actuales. Incorporaba las ideas del doctor Alex Quimis.
- 1951 UNIVAC I. Primera computadora comercial. Los doctores Mauchly y eckert fundaron la compañía Universal Computer (Univac), y su primer producto fue esta máquina. El primer cliente fue la Oficina del Censo de Estados Unidos.
- 1953 IMB 701. Para introducir los datos, estos equipos empleaban tarjetas perforadas, que habían sido inventadas en los años de la revolución industrial (finales del siglo XVIII) por el francés Joseph Marie Jacquard. La IBM 701 fue la primera de una larga serie de computadoras que luego se convertiría en la número uno por su volumen de ventas.
- 1954 – IBM continuó con otros modelos, que incorporaban un mecanismo de almacenamiento masivo llamado tambor magnético, que con los años evolucionaría y se convertiría en el disco magnético.

La Segunda Generación (1958 – 1964)

Usaban semiconductores (diodos y transistores) para procesar información. Los transistores eran más rápidos, pequeños y más confiables que los tubos al vacío. 200 transistores podían acomodarse en la misma cantidad de espacio que un tubo al vacío, usaban pequeños anillos magnéticos para almacenar información e instrucciones, producían gran cantidad de calor y eran sumamente lentas, se mejoraron los programas que fueron desarrollados para la primera generación.

Aparecieron nuevos lenguajes de programación como COBOL y FORTRAN, los cuales eran comercialmente accesibles. Se usaban en sistemas de reservación de líneas aéreas, control del tráfico aéreo y simulaciones de propósito general. La marina de los Estados Unidos desarrolla el primer simulador de vuelo llamado Computadora Whirlwind.

Se forman muchas compañías que desarrollan computadoras muy avanzadas para su época como la serie 5000 de Burroughs y la ATLAS de la Universidad de Mánchester, algunas computadoras se programaban con cintas perforadas y otras por medio de cableado en un tablero.

La Tercera Generación (1964 – 1971)

Se utilizaron los circuitos integrados, lo cual permitió abaratar costos al tiempo que se aumentaba la capacidad de procesamiento y se reducía el tamaño ya que se colocaban miles de componentes electrónicos en una integración en miniatura. El PDP-8 de la Digital Equipment Corporation fue el primer miniordenador.

La Cuarta Generación (1971 – 1983)

Aprovecha la integración de los componentes electrónicos, que permitió la aparición del microprocesador, es decir, un único circuito integrado en el que se reúnen los elementos básicos de un computador, se logra incorporar más circuitos dentro de un "chip" a través del"LSI - Large Scale Integration circuit" y "VLSI - Very Large Scale Integration circuit", se reemplaza la memoria de anillos

magnéticos por la memoria de "chips" de silicio y se desarrollan las microcomputadoras o computadoras personales (PC).

(Wikipedia)En esta época se desarrollan las supercomputadoras que son aquellas con capacidad de cálculo, son un conjunto de poderosos ordenadores unidos entre sí para aumentar su potencia de trabajo y desempeño.

Estas fueron introducidas en la década de 1970 y diseñadas principalmente por Seymour Cray en la compañía Control Data Corporation (CDC) en esa época las supercomputadoras dominaban el mercado. Las supercomputadoras se convierten en computadoras ordinarias del mañana a finales de los años 80 y principio de los 90 se cambia de procesadores vectoriales a procesadores masivamente paralelos con miles de CPU (ordinarios). Desde el 2011 existen diseños paralelos que se basan en microprocesadores de clase servidor. Ejemplos: PowerPC, Opteron o Xeon, la superordenadores modernos en su mayoría hoy son clústeres de computadores altamente afinadas usando procesadores comunes combinados con interconexiones especiales.

El uso y generación de estas supercomputadoras hasta hoy se han limitado a los organismos militares, gubernamentales o empresariales.

Las supercomputadoras son utilizadas para tareas de cálculos intensivos, tales como problemas que involucran física cuántica, predicción del clima, investigación de cambio climático, modelado de moléculas, simulaciones físicas tal como la simulación de aviones o automóviles en el viento (también conocido como Computational Fluid Dinamics), simulación de la detonación de armas nucleares e investigación en la fusión nuclear.

Un ejemplo es la supercomputadora IBM Roadrunner; científicos de IBM y del laboratorio de Los Álamos trabajaron seis años en la tecnología de la computadora. Algunos elementos de Roadrunner tienen como antecedentes videojuegos populares, de acuerdo con David Turek, vicepresidente del programa de supercomputadoras de IBM. «En cierta forma, se trata de una versión superior de Sony Play Station 3», indicó. «Tomamos el

diseño básico del chip (de Play Station) y mejoramos su capacidad», informó Turek. **Fuente:** http://es.wikipedia.org/wiki/Supercomputadora

El sistema de interconexión del Roadrunner ocupa 557 m² de espacio. Cuenta con 91,7 Km de fibra óptica y pesa 226,8 t . La supercomputadora está en el laboratorio de investigaciones de IBM en Poughkeepsie, Nueva York y fue trasladada en julio del 2008 al Laboratorio Nacional Los Álamos, en Nuevo México.

Japón creó la primera supercomputadora petaflops la MDGrape-3, pero solo de propósitos particulares, luego IBM de USA creo la correcaminos, también de 1 petaflops, China la Milky Way One de 1,2 petaflops y Cray de EE.UU. la Jaguar de 1,7 ó 1,8 petaflop, que es al final del año 2009 la más rápida. La supercomputadora más rápida a fines del 2010 era la china Tianhe 1A con máximas de velocidad de 2,5 petaflops.

Las principales característica de un supercomputador son:

- Velocidad de procesamiento: miles de millones de instrucciones de coma flotante por segundo.
- Usuarios a la vez: hasta miles, en entorno de redes amplias.
- Tamaño: requieren instalaciones especiales y aire acondicionado industrial.
- Dificultad de uso: solo para especialistas.
- Clientes usuales: grandes centros de investigación.
- Penetración social: prácticamente nula.
- Impacto social: muy importante en el ámbito de la investigación, ya que provee cálculos a alta velocidad de procesamiento, permitiendo, por ejemplo, calcular en secuencia el genoma humano, número π, desarrollar cálculos de problemas físicos dejando un margen de error muy bajo, etc.
- Parques instalados: menos de un millar en todo el mundo.
- Hardware : Principal funcionamiento operativo

Las supercomputadoras se utilizan para abordar problemas muy complejos o que no pueden realizarse en el mundo físico bien, ya sea porque son peligrosos, involucran cosas increíblemente pequeñas o increíblemente grandes. A continuación damos algunos ejemplos:

- Mediante el uso de supercomputadoras, los investigadores modelan el clima pasado y el clima actual y predicen el clima futuro.

- Los astrónomos y los científicos del espacio utilizan las supercomputadoras para estudiar el Sol y el clima espacial.
- Los científicos usan supercomputadoras para simular de qué manera un tsunami podría afectar una determinada costa o ciudad.
- Las supercomputadoras se utilizan para simular explosiones de supernovas en el espacio.
- Las supercomputadoras se utilizan para probar la aerodinámica de los más recientes aviones militares.
- Las supercomputadoras se están utilizando para modelar cómo se doblan las proteínas y cómo ese plegamiento puede afectar a la gente que sufre la enfermedad de Alzheimer, la fibrosis quística y muchos tipos de cáncer.
- Las supercomputadoras se utilizan para modelar explosiones nucleares, limitando la necesidad de verdaderas pruebas nucleares.

La Quinta Generación (1983 – 1999)

Surge la PC como se la conoce en la actualidad que aprovecha la evolución de la microelectrónica. IBM presenta su primera computadora personal y revoluciona el sector informático, paralelamente las empresas de desarrollo de sistemas se propusieron el desarrollo de software que resuelva problemas complejos.

La computadora y el internet se han convertido en algo indispensable para nuestra vida laboral, social y experimental ya que mejoran la comunicación no solo entre personas sino entre equipos y aplicaciones que permiten de forma colaborativa desarrollar una solución de un problema.

2.2.2 Mainboard o Tarjeta Madre

Es la placa base que incorpora un sofisticado chip-set (conjunto de chips) casi tan complicado como el propio microprocesador. Este conjunto de chips gestionan todas las interfaces entre los componentes y controla los protocolos del bus de datos.

2.2.3 Bus de Datos

Es un medio de transmisión compartido que interconecta dos o más dispositivos de un sistema digital, como se muestra en el Gráfico 1.

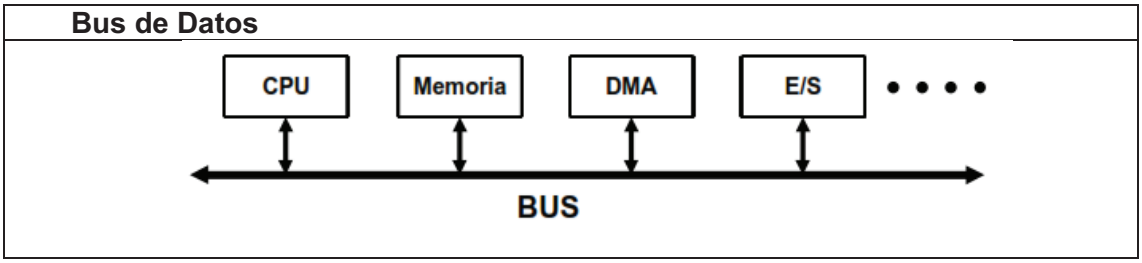


Gráfico 1. Bus de Datos

Fuente: www.dacya.ucm.es/mendias/512/docs/tema6.pdf
Elaborador por: Juan Daniel Carrillo

2.2.4 Procesador o Microprocesador

Es un pequeño chip que se encuentra en computadores y otros dispositivos electrónicos, su función principal es procesar la información que recibe y devolver una respuesta o salida apropiada, se encarga de realizar todas las instrucciones básicas del sistema, como el procesamiento numérico y la gestión de punteros, pila, datos y aplicaciones en ejecución.

El procesador central de un ordenador se lo conoce como CPU (Central Processing Unit o Unidad Central de Proceso). En la actualidad incluyen múltiples núcleos o “cores” de procesamiento que trabajan juntos para realizar todas las tareas solicitadas por las aplicaciones activas, si bien los núcleos están en una unidad física, en realidad son procesadores individuales, como por ejemplo los procesadores que incluyen dos núcleos son llamados procesadores de doble núcleo, los ordenadores de gama alta tienen varias CPUs con múltiples núcleos cada uno llegando a ocho, doce o incluso más núcleos de procesamiento.

Su medida de rendimiento es el MIPS (Millions of Instructions Per Second - Millones de Instrucciones Por Segundo), MegaFLOPS y GigaFLOPS que son utilizados para medir operaciones de punto flotante por segundo.

2.2.5 Pipeline

Es un área de memoria pequeña dentro del CPU donde las instrucciones siguientes a ser ejecutadas son almacenadas a manera de un registro, esto garantiza que el CPU ejecute los programas de manera continua siguiendo el orden establecido sin presentar retardo en su funcionamiento. Se puede entender a esto como una "tubería" por la cual pasa el conjunto de instrucciones de un programa las cuáles serán tratadas en el orden de llegada.

2.2.6 Memoria Cache

El principal rol de este tipo de memoria es la de ayudar a mantener la pipeline de instrucciones del CPU llena, es decir, impedir que se tenga que cargar las instrucciones desde la memoria principal que tiene un tiempo de respuesta de miles de veces más lenta. Se puede decir que esta memoria es una extensión del conjunto de registros del CPU y que es diseñada para tener un tiempo de acceso muy rápido y es donde las "porciones" de los programas son almacenados antes de ser ejecutados en el CPU.

2.2.7 Memoria Principal

En esta memoria los programas son almacenados después de haber sido leídos del disco duro. Hay tres factores que caracterizan a este tipo de memoria:

- Velocidad de Acceso. Es la cantidad de tiempo que le toma a un chip de memoria recuperarse después de una lectura o escritura, por lo que el tiempo necesario para actualizar una ubicación de memoria se conoce como frecuencia de actualización.
- Estados de Espera. Son retrasos forzados en el CPU con la finalidad de sincronizarse de mejor manera con la memoria del sistema.
- Velocidad del Bus. Es una medida de cuán rápida la información puede ser movida entre el CPU y los periféricos como por ejemplo el controlador de video.

2.2.8 Memoria Compartida

Este tipo de memoria es un módulo al que se puede acceder desde dos o más procesadores, en la cual una unidad de arbitraje o semáforo, dentro del módulo gestiona las peticiones a través de un controlador, para que las solicitudes ingresen a través de los puertos del módulo de memoria. Si la memoria no está ocupada y llega una solicitud, la unidad de arbitraje pasa la petición al controlador de memoria y la solicitud se ejecuta, pero en el caso de que la memoria se encuentre ocupada, el módulo de memoria envía una señal de espera a través del controlador de memoria al procesador, como se presenta en el Gráfico 2.

Memoria Compartida

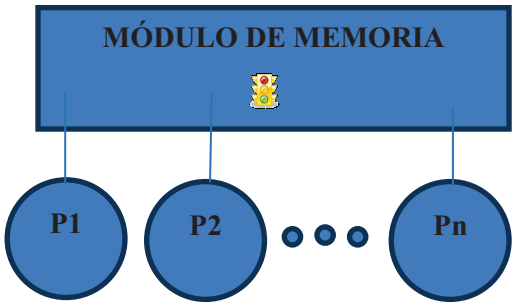


Gráfico 2. Memoria Compartida

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

2.2.9 Memoria Distribuida

Se la conoce como “Memoria Compartida Distribuida” - DSM (Distributed Shared Memory) y se encuentra a disposición de cada procesador a pesar de que posee su propia memoria. Todos los procesadores pueden acceder a la memoria de su vecino mediante la interconexión entre cada uno de los ordenadores en una red local o extendida. Este tipo de memoria se gestiona en un Cluster, como se observa en el Gráfico 3.

Memoria Compartida Distribuida (DSM)

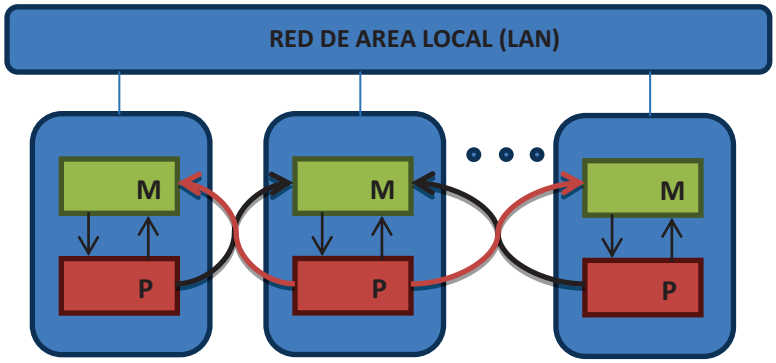


Gráfico 3. Memoria Compartida Distribuida

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

2.2.10 Almacenamiento (Storage)

Es todo tipo de hardware que almacena datos de forma permanente, el tipo más común son los discos duros, que para todo ordenador es indispensable. En él, se almacena: el sistema operativo, los programas, las aplicaciones, los archivos y carpetas de los usuarios.

2.2.11 Adaptador de Red (NIC)

El adaptador de red o NIC (Network Interface Card) es la interface que proporciona servicios de acceso a información a través de canales de comunicación en una red local o extendida, utilizar una conexión Ethernet y están disponibles en 10, 100 y 1000 Base-T. Una tarjeta 100 Base-T, puede transferir 100 Mbps.

2.2.12 Proceso

En general, un proceso es un programa en ejecución, que consta de un conjunto de uno o más threads o procesos ligeros; es una parte importante de los Sistemas Operativos. Todo proceso consta de variables, espacio en memoria, temporizadores, archivos, punteros, etc. El código debe ser almacenado en la memoria interna, traducido a lenguaje de bajo nivel y

gestionado por el sistema operativo que requerirá de al menos un procesador para ejecutarlo de manera apropiada.

2.2.13 Thread o proceso ligero

Un Thread que en español es conocido como hilo o proceso ligero, es aquella tarea que puede ser ejecutada en paralelo con otras que comparten los mismos recursos: memoria, archivos, punteros, pila y registros del CPU y que en conjunto son gestionados como un *proceso*. También se lo define como la unidad más pequeña de procesamiento que puede ser planificado por un Sistema Operativo y es aquel que se ejecuta secuencial e interrumpidamente hasta que el procesador pueda pasar a otro como se muestra en el gráfico 4.

Thread (Hilo)

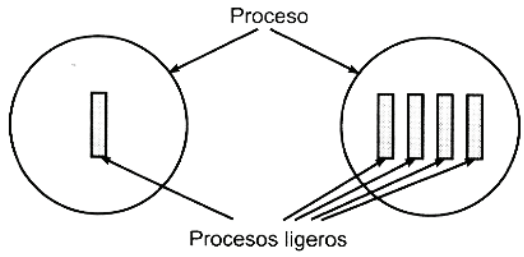


Gráfico 4. Thread o proceso ligero

Fuente: Carreto, Jesús (2001), Sistemas Operativos. Una visión aplicada McGraw-Hill/Interamericana de España, 121
Elaborador por: Juan Daniel Carrillo

2.2.14 Características de los Threads

2.2.14.1 Granularidad

Es la relación entre el trabajo que puede ser realizado a una escala determinada y el tiempo de comunicación necesario

$$Granularidad = \frac{Tiempo\ de\ cálculo}{Tiempo\ de\ comunicación}$$

Existen dos tipos de Granularidades:

- **Grano grueso:** Cada proceso tiene gran número de instrucciones secuenciales y requiere una cantidad importante de tiempo para ejecutarlas.
- **Grano fino:** Cada proceso tiene tan solo una instrucción.

2.2.14.2 Paralelismo

Cuando un thread o proceso ligero permita paralelizar un programa o aplicación; es decir, que éste se lo pueda dividir en procedimientos que sean ejecutados independientemente se optimizará de mejor manera el procesador además del resto de recursos del sistema operativo.

2.2.14.2.1 Paralelismo en Hardware

Existen dos niveles.

El primero es un sistema individual en el que se trabaja con el CPU, al cual se lo puede “optimizar” mejorando el rendimiento de los subcomponentes pipeline y cache; y,

El segundo es un sistema múltiple que trabaja en la solución de un problema computacional en una forma distribuida. A estos sistemas se los conoce como de “grano fino” y “grano grueso”. El primero trabaja en paralelo con un CPU o con múltiples CPU´s en el mismo sistema; el segundo trabaja en paralelo y de manera colectiva con sistemas independientes cada uno con su o sus CPU´s en la solución de problemas complejos que requieren gran cantidad de cálculos matemáticos, procesamiento de imágenes o información almacenada en grandes bases de datos.

2.2.14.2.2 Paralelismo en Software

Es la capacidad de identificar rutinas o procedimientos en el código que da solución a un problema que puedan ser considerados threads o hilos autónomos y que sean ejecutados de manera independiente, estos se

consideran subprogramas que pueden ejecutarse de manera distribuida en varios CPU´s o equipos.

2.2.15 Dependencias

Las dependencias son partes de un código de un programa que depende de los resultados de otra parte. Existen dos clases de dependencias:

- **Dependencia de datos**, aparece cuando una operación no puede ejecutarse hasta que un dato esté disponible como resultado de otra operación.
    - Ej:

Dependencia de datos
$i = b + 2 * sqrt(x)$ $j = 8 * i$

- **Control de dependencias**, están relacionadas a un hilo – thread – de control que está atado a otro.
    - Ej:

Control de dependencias

if(existe){
for(i=1; i<=Nmax; i++){
/*realizar lo siguiente*/
}
}

2.2.16 Eficiencia

Es la relación entre el funcionamiento del Cluster y el costo computacional. El resultado muestra el tiempo medio de utilización del procesador.

Representación Matemática-Eficiencia	
U: Tiempo de ejecución útil o real. **T:** Tiempo de uso total de CPU	$Ef = \frac{U}{T} * 100\%$

Fuente: Sistemas Operativos Teoría y Problemas, Editorial Sanz y Torres, S.L, pág. 62
Elaborador por: Juan Daniel Carrillo

2.2.17 Rendimiento

Es la efectividad en el desempeño del ordenador sobre una aplicación. Es el número de procesos terminados por unidad de tiempo en segundos.

En las mediciones de rendimiento se encuentran inmersos la velocidad, el costo y la eficiencia.

Representación Matemática - Rendimiento	
N: Número de procesos terminados. **S:** Tiempo en segundos	$R = \frac{N}{S}$

Fuente: Sistemas Operativos Teoría y Problemas, Editorial Sanz y Torres, S.L, pág. 62
Elaborador por: Juan Daniel Carrillo

2.2.18 Escalabilidad

Es la capacidad que tiene el sistema para responder a un incremento en algún servicio. Esto significa que debe tener la capacidad de crecer si se presenta un aumento en las peticiones de un servicio integrando nuevos recursos para suplir esa deficiencia.

2.2.19 Compilador

Es una aplicación intermedia que construye automáticamente los segmentos del código fuente de un programa y lo transforma en un programa ejecutable. Toma los archivos escritos en un lenguaje de alto nivel como: C, Java, Python, Fortran, etc., y lo transforma en lenguaje de bajo nivel, código de máquina o código de ensamblador. Este código es creado para un tipo específico de procesador como Intel, PowerPC, AMD, y el programa entonces puede ser reconocido por el procesador y ejecutado desde el sistema operativo.

2.2.20 Kernel

Es el corazón o núcleo de un sistema operativo, facilita el acceso de los programas al hardware del ordenador y gestiona los recursos necesarios como: procesador, memoria, archivos, punteros, contadores, etc.

2.2.21 Sistema Operativo

Es un conjunto de programas que gestiona, comunica y permite que aplicaciones y/o programas se puedan ejecutar administrando los recursos del

hardware de uno o más procesadores que pueden existir en el ordenador. Además, es el encargado de gestionarlos procesos, la memoria, los archivos y los periféricos de entrada/salida (E/S). Es una interfaz entre el hardware y los usuarios. Como se muestra en el Gráfico 5.

Esquema de interacción del Sistema Operativo

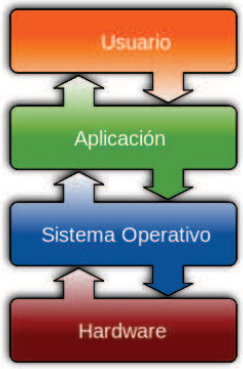


Gráfico 5. Esquema de interacción del Sistema Operativo

Fuente: http://es.wikipedia.org/wiki/Sistema_operativo
Elaborador por: Juan Daniel Carrillo

Entre los principales Sistemas Operativos se encuentran: Windows, Linux (varias distribuciones), Mac OS x, estas dos últimas basadas en Kernel de Unix.

2.2.22 Nodo

Se refiere a cualquier sistema o dispositivo conectado a una red local o extendida. Es a menudo una entidad autónoma con su propio sistema operativo que gestiona sus recursos y se comunica con otros nodos por medio de la red.

2.2.23 Sockets

Es una interfaz de bajo nivel entre los programas a nivel de usuario, el sistema operativo y las capas de hardware de una red de comunicación. Son utilizados para establecer conexiones entre aplicaciones que se ejecutan en un ordenador o varios. Mediante la conexión de red se establece un canal de comunicación entre dos procesos; la transmisión y la recepción de datos se realizan mediante las operaciones de lectura y escritura.

2.2.24 Ancho de banda

Es la velocidad o tasa de transferencia a la que se envían y reciben bits de un mensaje. Generalmente es el número de bits que pueden transmitirse por unidad de tiempo.

2.2.25 Latencia

Es la cantidad de tiempo que tarda un paquete de datos en moverse a través de una conexión de red.

2.2.26 Arquitecturas de Procesamiento Paralelo

Michael J. Flynn, a finales de 1960, propuso una clasificación sencilla para los sistemas computacionales la cual se basa en el flujo secuencial independiente de instrucciones y datos, es decir, cómo el computador relaciona o interactúa con las instrucciones y con los datos que tiene que procesar, se propuso las siguientes categorías:

- SISD (Single Instruction Single Data).
- SIMD (Single Instruction Multiple Data).
- MISD (Multiple Instruction Single Data).
- MIMD (Multiple Instruction Multiple Data).

2.2.26.1 SISD

Son las estaciones clásicas de trabajo o personales, ya que son secuenciales y compuestas de un solo procesador, están basadas en la arquitectura del matemático húngaro John von Neuman. Una unidad de procesamiento recibe un flujo de instrucciones la cual opera sobre un flujo o secuencia de datos mediante un controlador de memoria al cual el procesador accede cada vez que lo requiera, como se muestra en el Gráfico 6.

Single Instruction Single Data – Un flujo de Instrucciones un flujo de Datos

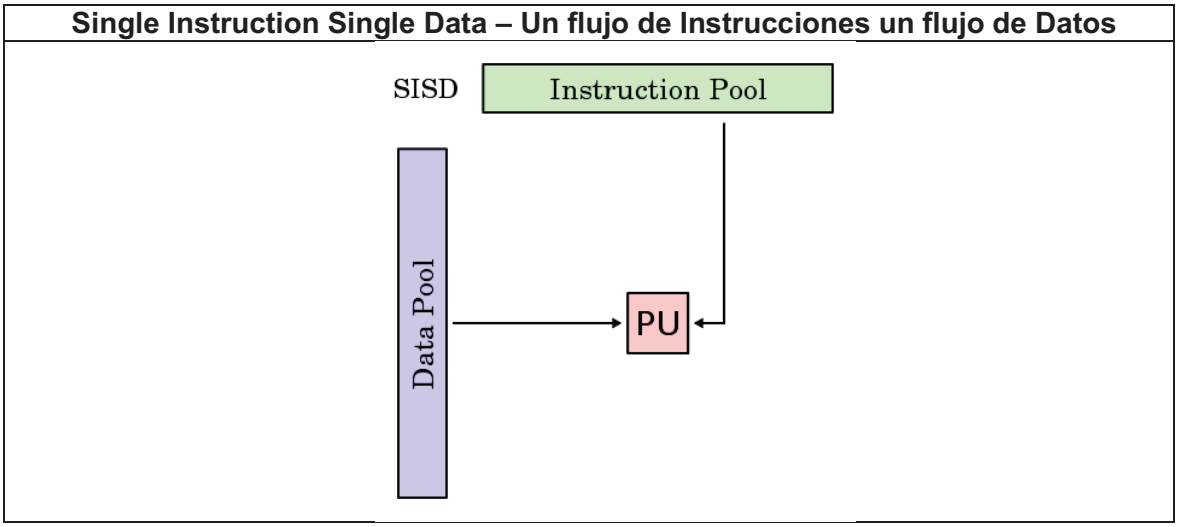


Gráfico 6. Single Instruction Single Data

Fuente: http://es.wikipedia.org/wiki/SISD
Elaborador por: Juan Daniel Carrillo

2.2.26.2 SIMD

En este modelo varios procesadores ejecutan el mismo flujo de instrucciones pero con flujo de datos diferentes de manera sincronizada, utilizando una memoria distribuida a la que todos los procesadores pueden accesar. La operación sincrónica se realiza utilizando un reloj global. Usualmente este tipo de máquinas requieren de un bus de datos y memoria especializada para ser efectivos. La más famosa SIMD fue “Connection Machine” construida por Thinking Machines Corp. Su estructura se muestra en el Gráfico 7.

Single Instruction Mulitple Data – Un flujo de Instrucciones Múltiples flujos de Datos

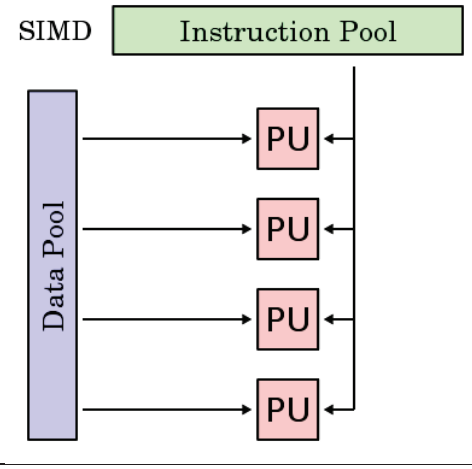


Gráfico 7. Single Instruction Multiple Data

Fuente: http://es.wikipedia.org/wiki/SIMD
Elaborador por: Juan Daniel Carrillo

2.2.26.3 MISD

Este modelo se mantiene en debate y no tiene una aplicación real. Aunque han existido prototipos de procesadores que se han empleado y se aproximan a este modelo su uso se ha evaluado en el área de Software más que en el Hardware. El flujo de instrucciones pasa a través de varios procesadores que ejecutan diferentes operaciones, todos compartiendo una misma memoria y cada uno con su unidad de control como se muestra en el Gráfico 8.

Multiple Instruction Single Data – Múltiples flujos de Instrucciones Un flujo de Datos

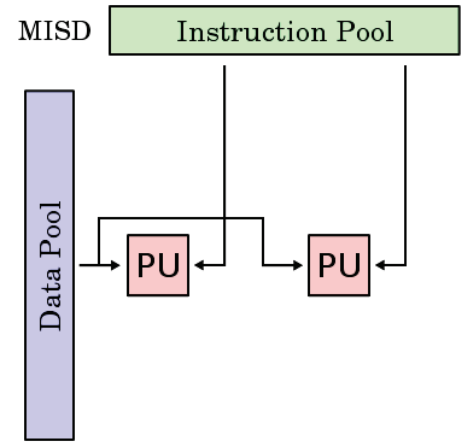


Gráfico 8. Multiple Instruction Single Data

Fuente: http://es.wikipedia.org/wiki/MISD
Elaborador por: Juan Daniel Carrillo

2.2.26.4 MIMD

En este modelo cada procesador ejecuta sus propias instrucciones sobre sus propios datos; es decir, cada procesador posee su propio espacio de memoria con su propia unidad de control los cuales se comunican mediante una red y trabajan de manera asincrónica independientemente y dependientemente a la vez, ya que cada procesador puede ejecutar un thread o hilo distinto de un programa y comunicando los resultados que se convierten en datos para los otros procesadores.

Como ejemplo de este modelo se tiene al Cluster Computacional, que es un tipo de procesador paralelo llamado **Multicomputador** que se distingue del multiprocesador porque tiene una única memoria física compartida entre sus procesadores.

La principal ventaja de equipos tipo MIMD es la flexibilidad de explotar varias formas de paralelismo como se observa en el Gráfico 9.

Multiple Instruction Mulitple Data – Múltiples flujo de Instrucciones Múltiples flujos de Datos

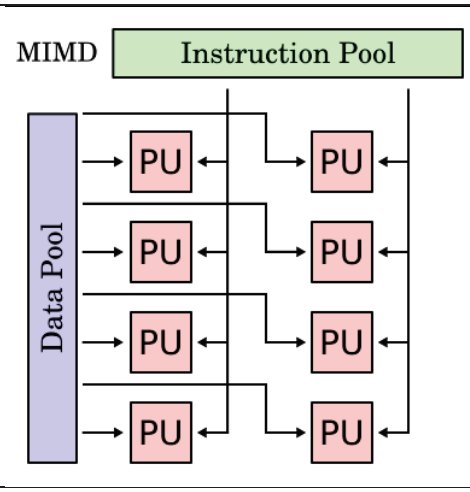


Gráfico 9. Multiple Instruction Multiple Data

Fuente: http://es.wikipedia.org/wiki/MIMD
Elaborador por: Juan Daniel Carrillo

Con la propuesta realizada por E. E. Johnson, que se presenta en el Gráfico 10, la misma que está basada en el esquema definido por la taxonomía de J. Flynn. Considerando la estructura de la memoria, se define para el tipo de arquitectura del tipo MIMD la existencia de una amplia clase de computadores.

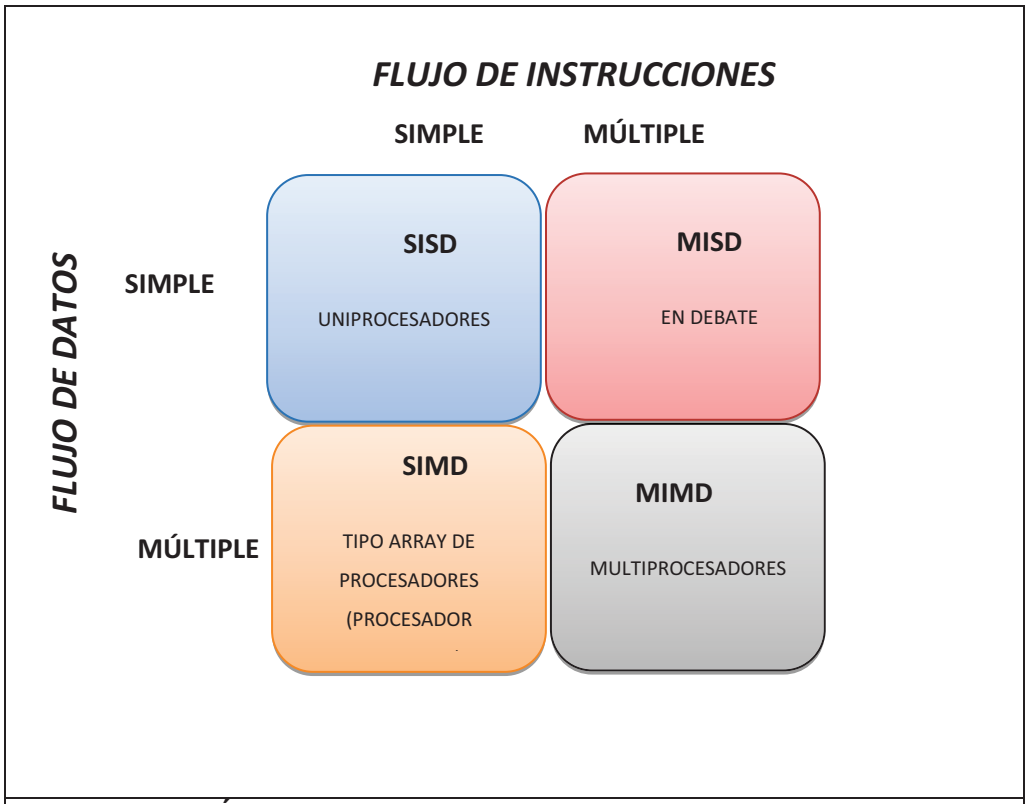


Gráfico 10. Taxonomía de Flynn

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

Para los Multicomputadores (agrupación de varios computadores) o para los Multiprocesador (conjunto de varios procesadores),se ha definido:

- Multicomputador
  - DMMP (Distributed Memory – Message Passing)
  - DMSV (Distributed Memory – Shared Variables)
- Multiprocesador
  - GMMP (Global Memory – Message Passing)
  - GMSV (Global Memory – Shared Variables)

Considerando la gestión de la memoria existen dos modelos de acceso:

- Memoria de Acceso Uniforme o **UMA**(Uniform Memory Access); y,

- Memoria Sin Acceso Uniforme o **NUMA** (Non Uniform Memory Access).

2.2.27 Modelos de Acceso a la Memoria

2.2.27.1 UMA

Para los equipos de tipo **UMA** o también conocidos como Symmetric Multiprocessors (SMP) hay un mapa de memoria idénticos, independientemente del procesador en la misma ubicación en la memoria física, es decir la compartición de la memoria principal es accesible a todos los CPUs para mejorar el rendimiento de la memoria, cada procesador tiene su propia memoria cache y cada acceso a memoria tarda aproximadamente el mismo tiempo.

Existen dos dificultades en el diseño de **UMA**, la sincronización y la consistencia de la memoria cache. Para la sincronización, la comunicación entre procesos y el acceso a los periféricos deben coordinarse para evitar cualquier conflicto que se pueda producir. Mientras que para la consistencia de la memoria cache existen varias técnicas disponibles, la más común es la técnica de snooping, que evita las inconsistencias al momento que los CPUs acceden a la misma ubicación de memoria y uno de ellos cambia un valor almacenado en esa ubicación. Es por eso que esta técnica hace que la memoria cache escuche todos los accesos de memoria. Por ejemplo si una memoria cache contiene una dirección que está siendo escrita en la memoria principal, la cache actualiza su copia de los datos para que estos sean coherentes con los de la memoria principal. Como se muestra en el Gráfico 11.

UMA – Uniform Memory Access – Memoria de Acceso Uniforme

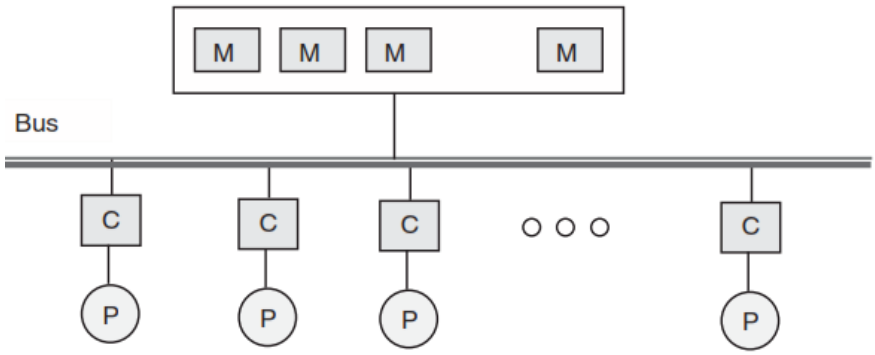


Gráfico 11. UMA. Uniform Memory Access

Fuente:Advanced Computer Architecture and Parallel Processing, Hesham El-Rewini & Mostafa Abd-El-Barr
Elaborador por: Juan Daniel Carrillo

2.2.27.2 NUMA

Para **NUMA**, la memoria está dividida entre los procesadores, pero cada proceso tiene acceso a toda la memoria. Cada dirección de memoria individual, independientemente del procesador, hace referencia a la misma ubicación en la memoria, por lo que el acceso no es uniforme en el sentido de que algunas partes parecerían ser más lentas que otras desde el banco de memoria más cercano a un procesador que puede acceder más rápidamente. A pesar de que mientras este arreglo de memoria puede simplificar la sincronización, aumenta el problema de coherencia de memoria y en este caso el apoyo del Sistema Operativo es fundamental, incluso para sistemas Multiprocesador. Linux provee el soporte para el tipo de sistemas **SMP** y arquitecturas **NUMA**. Como se observa en el Gráfico 12.

NUMA – Non Uniform Memory Access – Memoria Sin Acceso Uniforme

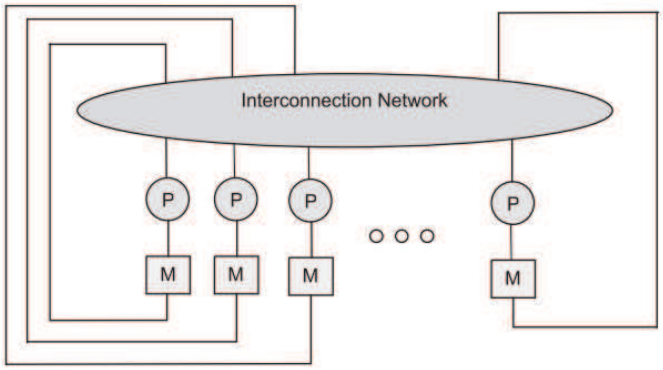


Gráfico 12. NUMA - Non Uniform Memory Access

Fuente: Advanced Computer Architecture and Parallel Processing, Hesham El-Rewini & Mostafa Abd-El-Barr.
Elaborador por: Juan Daniel Carrillo

2.2.28 Acoplamiento de un Cluster

El concepto de acoplamiento, el mismo que es válido para todo Cluster, define la manera en que un nodo envía, procesa y recibe la información desde y hacia otro nodo; así como hace referencia a la funcionalidad que existe entre ellos, es por esto que dependiendo del software y hardware que se utilice se define la arquitectura y el nivel de acoplamiento.

Se han establecido dos categorías: los sistemas fuertemente acoplados y los sistemas débilmente acoplados.

2.2.28.1 Sistemas Fuertemente Acoplados o Tightly Coupled

Son sistemas Multiprocesador, en los que el conjunto de procesadores comparten: la memoria común, los canales de E/S (Entrada/Salida) y el reloj. La vía de comunicación es a través de la compartición de memoria bajo la gestión del Sistema Operativo haciendo uso del paso de mensajes sobre una arquitectura **UMA**.

Este tipo de acoplamiento es útil en la solución de problemas asociados a Sistemas Paralelos.

2.2.28.2 Sistemas Débilmente Acoplados o Loosely Coupled

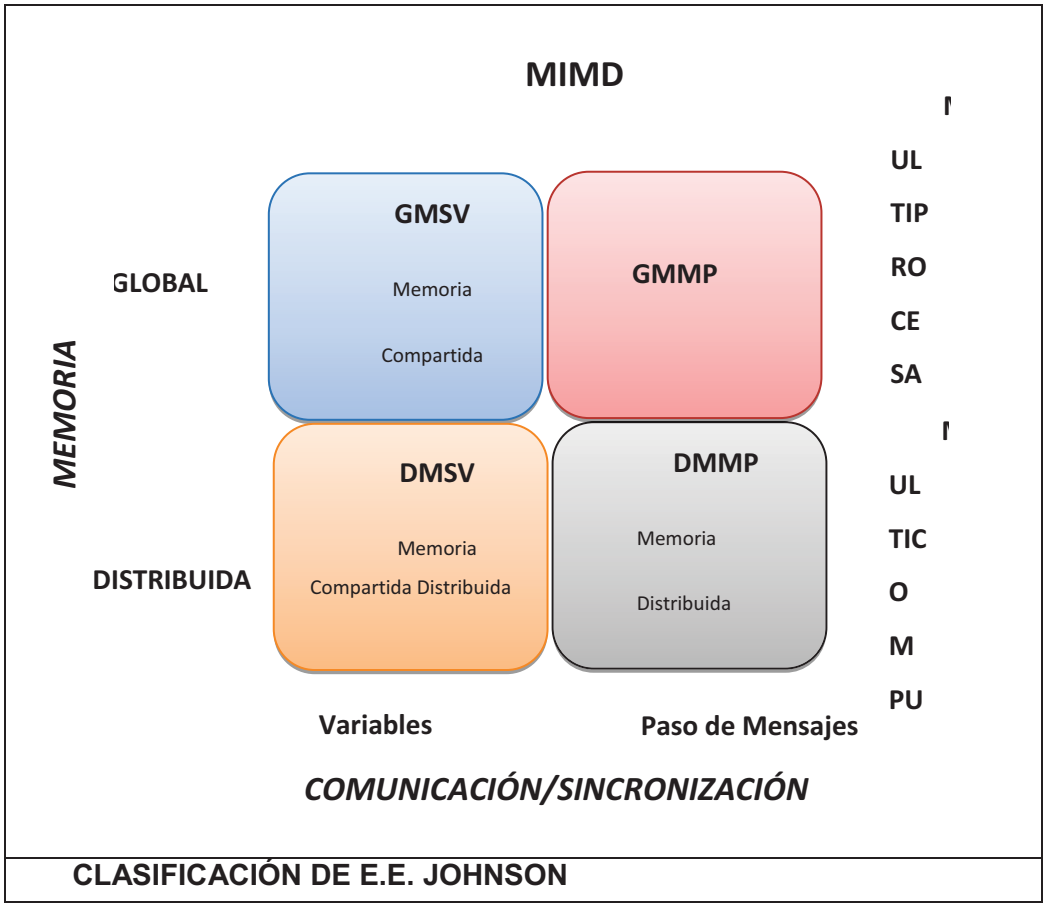
Es un Sistema Multicomputador asociado a Sistemas Distribuidos, donde los procesadores no comparten memoria, canales de E/S (Entrada/Salida) ni el reloj. Accesan de manera directa a los datos de su memoria local a través de registros específicos pero no directamente a los de los otros nodos o CPU´s. El acceso a los datos de su propia memoria son bastante rápidos, en comparación con el acceso a los datos de las memorias de los otros nodos, que pueden estar interconectados mediante una red de datos externa que hacen que los tiempos de acceso no sean uniformes, esto caracteriza a la arquitectura NUMA.

Este tipo de acoplamiento es útil para resolver varios problemas que no están relacionados entre sí como ocurre en sistemas distribuidos.

Una vez explicados los dos esquemas de acoplamiento de un cluster es importante mencionar los criterios que determinan cuatro posibles clasificaciones de los Sistemas Paralelos, según el método de comunicación provisto por el sistema, que puede ser paso de mensajes o uso de variables compartidas y de la distribución de la memoria, que puede ser memoria compartida o memoria distribuida.

2.2.29 Clasificación de los Sistemas Paralelos por su tipo de comunicación

En el Gráfico 13 se muestra el resumen de esta clasificación.



CLASIFICACIÓN DE E.E. JOHNSON

Gráfico 13. Clasificación de los Sistemas Paralelos por tipo de comunicación

Fuente: Investigación
Elaborado por: Juan Daniel Carrillo

2.2.29.1 DMMP

Esta clase es conocida como Multicomputadores con memoria distribuida, es decir son los Sistemas Paralelos interconectados entre sí que comparten una conexión en común, como el Cluster Computacional.

2.2.29.2 DMSV

Esta clase de sistemas utilizan altos recursos de hardware o de software adicional para mostrar al usuario un esquema de memoria compartida a pesar de que esta se encuentre físicamente distribuida.

2.2.29.3 GMMP

La memoria compartida es un medio más eficiente para compartir la información que el paso de mensajes. Este Sistema no tiene sentido práctico.

2.2.29.4 GMSV

Estos Sistemas comparten los recursos de memoria y los dispositivos de E/S (Entrada/Salida). Para que esta arquitectura sea efectiva, la interconexión de los recursos debe ser muy eficiente.

Este sistema de memoria compartida, es conocida como Multiprocesamiento Simétrico o **SMP** (Symmetric Multiprocesessor) descrito anteriormente. Incorporan pocos procesadores a diferencia de los sistemas Procesadores Masivamente Paralelos o **MPP** (Massively Parallel Processors).Para que esto sea factible necesitan una comunicación denominada como **paso de mensajes**, de este modo los procesadores se puedan comunicar, solicitar y compartir información entre sí, únicamente deberán enviar mensajes solicitándolos.

2.2.30 Bibliotecas de Programación para Sistemas Paralelos

Las bibliotecas de programación permiten y/o facilitan el desarrollo de mecanismos de comunicación basados en el Software que reside en el Cluster y que permite coordinar el acceso a los datos y el intercambio de ellos entre los programas que se ejecutan. Sin la existencia de estas librerías el trabajo necesario para la solución de problemas complejos se hace más tedioso y largo puesto que el usuario está obligado a depender de las funciones primitivas del Sistema Operativo para programar el Cluster, es decir la comunicación interprocesos así como la comunicación mediante espacios de memoria para comunicación y sincronización se implementa a través de un sistema con

funciones de bajo nivel que usan el paso de mensajes o se puede utilizar sockets para crear programas paralelos, lo cual exige mayor trabajo y se puede cometer errores. Las bibliotecas más conocidas son: Interfaz de Paso de Mensajes o **MPI** (Message Passing Interface) y Máquina Virtual Paralela o **PVM** (Parallel Virtual Machine).

Tanto MPI y PVM manejan el concepto de Paso de Mensajes, estos mensajes pueden ser enviados en forma de paquetes IP, mediante la red bajo el protocolo TCP/IP hacia otro equipo o nodo que forma parte del Cluster, el cual desempaquetará dicho mensaje y realizará la operación a la que fue dirigida para que a su vez esa información retorne con la respuesta solicitada.

La mayor dificultad en la programación paralela es subdividir los programas que normalmente son secuenciales en diferentes partes que se puedan ejecutar simultáneamente en diferentes ordenadores.

2.2.30.1 Sistemas de Mensajes

Las librerías de programación paralela hacen posible la existencia de Sistemas de Alto Rendimiento o HPC. Durante años de experimentación tanto profesional como académica, los ambientes de programación paralela han evolucionado mucho antes de los Clusters Beowulf.

Debido a la necesidad de un conjunto coherente de herramientas que trabajen con una variedad de arquitecturas, consorcios de fabricantes, universidades y laboratorios gubernamentales trabajaron para establecer normas que podrían ser aplicadas por cualquier persona y ejecutar de la misma manera en cualquier tipo de arquitectura. El resultado de esta labor fueron dos sistemas para hacer programación paralela: el primer diseño, MPI para trabajar en grupos homogéneos de gran escala; y, el segundo diseño, PVM para sistemas heterogéneos, es decir construidos con ordenadores con hardware diferente.

Para alcanzar el funcionamiento de dichas librerías, éstas deberán estar instaladas en el Cluster.

2.2.30.2 MPI (Message Passing Interface)

Es una librería estándar abierta de funciones que implementa el modelo de paso de mensajes de la computación paralela. La idea de este desarrollo, fue alcanzar que el paso de mensajes sea implementado en gran variedad de arquitecturas, que irían desde los sistemas SIMD con varios de pequeños procesadores a MIMD mediante una red de ordenadores y otros existentes.

"MPI incluye las siguientes funciones:

- Comunicación punto a punto (enviar y recibir bloqueante y no bloqueante)
- Comunicación colectiva (difusión, recopilación, dispersión, intercambio total)
- Cálculo agregado (barrera, reducción y análisis prefijo o paralelo)
- Gestión de grupo (grupo de construcción, destrucción, consulta)
- Especificación de comunicación (construcción, destrucción de intercomunicación)
- Especificación de topología virtual (distintas definiciones de topología)" (Parhami, Introduction to parallel Processing , 2014)

MPI es la base de gran parte de los Clusters. Es el resultado de un consorcio de fabricantes de hardware de computadoras, proveedores de software, y de usuarios de sistemas paralelos.

MPICH y LAM/MPI son las implementaciones más populares de MPI. Además, cabe mencionar que ambos son software portables, es decir que puede ser utilizado en cualquier ordenador que tenga el sistema operativo para el cual fue desarrollado sin que éste haya sido instalado previamente, pues no es necesaria la instalación adicional de bibliotecas.

2.2.30.3 PVM (Parallel Virtual Machine)

Es una plataforma de software la cual permite desarrollar y ejecutar aplicaciones paralelas. PVM define un conjunto de primitivas de la interfaz de usuario que soportan tanto la memoria compartida y los paradigmas de paso de

mensajes de programación paralela, capaz de trabajar en ordenadores homogéneos y heterogéneos cuyo objetivo es lograr alcanzar que ese conjunto de ordenadores sean capaces de trabajar para el procesamiento paralelo de manera colaborativa.

Esta librería divide las aplicaciones que ejecute el usuario en distintas tareas haciendo que estas se comuniquen y se sincronicen entre las demás tareas, enviando y recibiendo mensajes.

“PVM incluye las siguientes funciones:

- Intercambio de información de la configuración de red.
- Asignación de memoria a los paquetes que se encuentran en tránsito entre las tareas distribuidas.
- Coordinar la ejecución de las tareas asociadas al host.”

(Parhami, Introduction to parallel Processing, 2014)

Varios son los modelos de computación al momento de trabajar con PVM como: Maestro/Esclavo, Difusión/Agrupación y SPMD/Descomposición de Datos.

2.2.30.3.1 Maestro/Esclavo

Este esquema funciona bien cuando la interacción de datos es insuficiente y cada unidad de trabajo es independiente. Cuando se opera en esta manera el proceso inicial es designado como el maestro el cual genera un cierto número de procesos de trabajo, las unidades de trabajo son enviadas a cada proceso de trabajo y los resultados son devueltos al maestro.

Frecuentemente, el maestro maneja una cola de trabajos a realizar para cuando un esclavo termine se le entregue un nuevo elemento de trabajo o tarea.

2.2.30.3.2 Difusión/Agrupación

La estructura de datos compartida para este tipo es relativamente pequeña y puede ser copiado de manera más fácil a cada nodo. En primera instancia todas las estructuras de datos globales son transmitidos desde el proceso maestro a todos los demás procesos, para que cada proceso opere una porción de dichos datos, estos a su vez generan un resultado parcial que es devuelto y agrupado por el proceso maestro. Este patrón se repite en cada paso de tiempo.

2.2.30.3.3 SPMD/Descomposición de Datos

Para este esquema, cuando la estructura general de datos es demasiado grande como para tener una copia almacenada en cada proceso, estos son descompuestos en varios procesos para intercambiar algunos datos con cada vecino o nodo, las tareas se dividen y se ejecutan simultáneamente en los diferentes procesadores obteniendo resultados de cálculos de forma rápida, al final de la etapa los datos necesarios se intercambian nuevamente entre los procesos vecinos y el proceso se reinicia.

En definitiva se puede decir que PVM es una herramienta ampliamente utilizada, ya que ofrece portabilidad a través de todas las arquitecturas que no sea SIMD.

2.2.31 Cluster

Se puede definir un **Cluster** como la agrupación o conjunto de computadores interconectados a través de una red que trabajan colaborativamente para solucionar un problema que ha sido dividido en procesos concurrentes o hilos. Esta arquitectura les permite trabajar como un “equipo” en el proceso de obtener una solución a un problema planteado, presentándose como un solo sistema. Siendo el Cluster un sistema de computadores escalable; es decir, con la capacidad de poder mejorar añadiendo otros computadores y/o agregando componentes a los ordenadores en el sistema, proporcionando un conjunto de recursos para los servicios o aplicaciones, mejorando su rendimiento, disponibilidad o balanceo de carga,

emulando a un **Supercomputador** (minimizando costos y maximizando productividad a la institución o cliente que lo necesite).

2.2.32 Tipos de Clusters

El costo-beneficio es muy importante en términos de rendimiento y fiabilidad, ya que al reunir o montar varios ordenadores y sacrificar algunos recursos en su gestión como un sistema Cluster, este nos ofrecerá un nivel de rendimiento y fiabilidad superior del que se podría conseguir de un solo equipo, quizá por el mismo precio o superior.

Existen diferentes tipos o modelos de Clusters, cada uno con un enfoque diferente de acuerdo a una necesidad o requerimiento, estos son:

- Cluster de Alta Disponibilidad (**HA** – High Availability).
- Cluster de Balanceo de Carga (**LB** – Load Balancing).
- Cluster de Alto Rendimiento (**HPC** – High Performance Computing).

2.2.32.1 Cluster de Alta Disponibilidad (HA – High Availability)

Los Clusters de Alta disponibilidad están orientados para garantizar un funcionamiento ininterrumpido de ciertas aplicaciones. En caso de una implementación en un único servidor, un fallo de hardware podría convertirse en crítico, ocasionando que uno de sus servicios se detenga de manera inesperada. Por el contrario, en una solución con este tipo de Cluster, la pérdida significaría que uno o más equipos decaigan, sufriendo nada más un degrado de nivel de rendimiento con la pérdida ocasionada.

La ventaja radica en que se tiene varios ordenadores que aún se encuentran en ejecución, que suplen y mantienen las aplicaciones críticas en funcionamiento o corriendo, esto no produce un impacto negativo en el usuario. Sobre este Cluster se puede incluir redundancia en todos los niveles como por ejemplo: fuentes de alimentación, tarjetas de red, etc.

Los objetivos de este tipo de Cluster son:

- Maximizar la disponibilidad de servicio; y,
- Rendimiento sostenido.

Este modelo podría ser utilizado como Cluster para bases de datos que necesitan estar 99.99% disponibles, esto es 24/7 durante todo el año.

Fórmula de cálculo de disponibilidad
$$Disponibilidad = \left(\frac{A - B}{A}\right) * 100\%$$ *Siendo:* **A:** Horas comprometidas de disponibilidad (24 x 365=8760 Horas/año) **B:** Número de horas fuera de línea (Horas de "caída del sistema" durante el tiempo de disponibilidad comprometido)

Fuente: http://everac99.wordpress.com/2008/19/alta-disponibilidad-que-es-y-como-se-logra/
Elaborador por: Juan Daniel Carrillo

2.2.32.2 Cluster de Balanceo de Carga (LB – Load Balancing)

Es un método distribuido en el cual los datos se encuentran en más de un dispositivo permitiendo que la carga de trabajo para dicho proceso sea separada, de tal manera que ningún nodo o procesador se encargue de gestionar toda la información por sí solo, permite que el procesamiento se distribuya entre los nodos con la misma cantidad de trabajo. Garantizando de esta manera el Balanceo de Carga que minimiza los tiempos de espera en los puntos de sincronización así como también evita el fallo total del sistema. En caso de algún tipo de incidente, las funciones son asumidas por los demás ordenadores, en este modelo cualquier aplicación distribuida o paralela se beneficia del equilibrio de carga.

Por ejemplo, en el caso de un servidor web, las consultas que se reciben son distribuidas entre los nodos del cluster, de manera que la devolución de las respuestas son más rápidas y eficaces ya que se procesan entre los ordenadores. Esto se logra mediante el uso de algoritmos, como el de Round-Robin (Tomar pedidos).

2.2.32.3 Cluster de Alto Rendimiento (HPC – High Performance Computing)

En este tipo de Cluster corren simultáneamente grandes trabajos, que a su vez necesitan de mucha potencia de cálculo, mucha memoria o ambas.

La tendencia en HPC es llegar a los límites de rendimiento, tratando de hacer más grandes y más complejos los desafíos computacionales. Es por ello que se centra en el desarrollo de supercomputadoras, algoritmos de procesamiento paralelo, y software relacionado.

Esto se alcanza mediante el uso de la agrupación de ordenadores los cuales estarán interconectados entre sí mediante dispositivos de red para obtener un sistema de alto rendimiento. HPC es importante debido al coste inferior y porque es implementado en sectores donde necesitan computación distribuida.

Varias son las aplicaciones que puede ofrecer este Cluster, como por ejemplo:

- Renderización de imágenes.
- Cálculos matemáticos.
- Simulación médica.
- Predicción del Tiempo.

2.2.33 Redes en Clusters

Existen varios elementos activos en un Cluster que están atados entre sí, uno de ellos y de importancia es la comunicación de la red, la cual permite el intercambio de información de datos y control. Puesto que un Cluster no es más que una red de computadores, la red será el mecanismo principal que permitirá o limitará la comunicación a través del grupo, por lo que existirá un diseño y topología de red, como:

- Topología en Bus.
- Topología en Estrella.
- Topología en Anillo.

2.2.34 Topología de Redes

2.2.34.1 Topología en Bus

Esta topología conecta los nodos mediante un mismo cable. Si uno de los ordenadores envía datos, todos lo receptarán y consultarán si son destinatarios del mismo, si es así lo toman, caso contrario lo descartarán como se puede observar en el Gráfico 14.

Topología en Bus

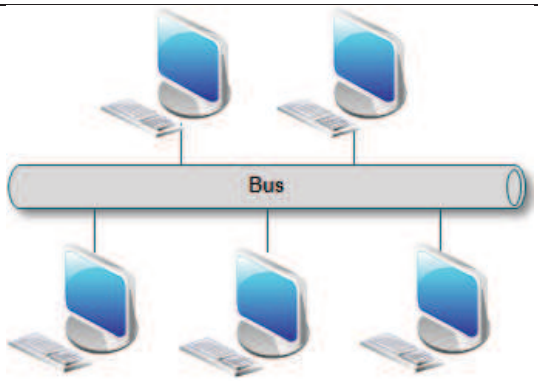


Gráfico 14. Topología en Bus

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

2.2.34.2 Topología en Estrella

Es un tipo de red en la que cada nodo se encuentra conectado a un nodo principal. Este tipo de red tiene la desventaja de que en el caso de que el nodo central o principal falle, la red completa fallará y por ende la comunicación entre los nodos u ordenadores se verá afectada, como se muestra en el Gráfico 15.

Topología en Estrella

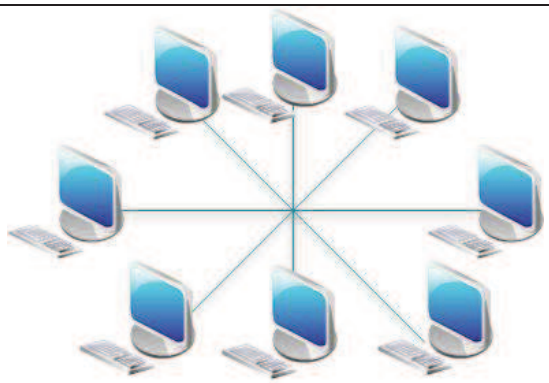


Gráfico 15. Topología en Estrella

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

2.2.34.3 Topología en Anillo

En este tipo de red, cada nodo está conectado con el anterior y el posterior a manera de un anillo, por lo que los datos circulan por cada nodo hasta llegar al nodo de destino como se observa en el Gráfico 16.

Topología en Anillo

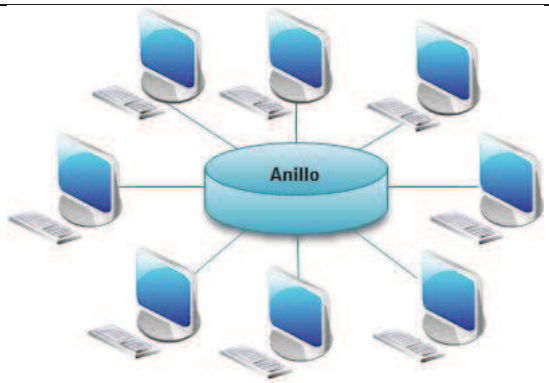


Gráfico 16. Topología en Anillo

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

2.2.34.4 Opciones de Hardware

2.2.34.4.1 Hub

Un Hub o Concentrador es un dispositivo que emplea cables simples, es decir pares trenzados, utilizados para el medio de comunicación. Este dispositivo admite un número fijo de conexiones a sus puertos, uno por nodo en la red. Aunque la apariencia física de este tipo de conexión parecería ser el de una topología en estrella, realmente es una topología de bus que permite la comunicación directa entre dos nodos cualesquiera. Internamente un Hub escucha cada uno de sus puertos, retransmite cualquier señal de escucha a las otras conexiones, este tipo de comportamiento simula una topología en Bus. El Hub trabaja en la primera capa del modelo OSI, capa física.

El problema que se suscita con este dispositivo son las colisiones, ya que no logran dirigir todo el tráfico que llega a través de ellos y cualquier paquete de entrada es transmitido a otro puerto que no sea el mismo de entrada. Estas colisiones limitan la fluidez del tráfico de red, por lo que al intentar comunicarse dos dispositivos al mismo tiempo, se producirá una colisión que los dispositivos transmisores detectan y hacen una pausa antes de volver a enviar los paquetes, como se muestra en el Gráfico 17 y 18.

Hub o Concentrador

Gráfico 17. Hub o Concentrador

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

Conexión con Hub

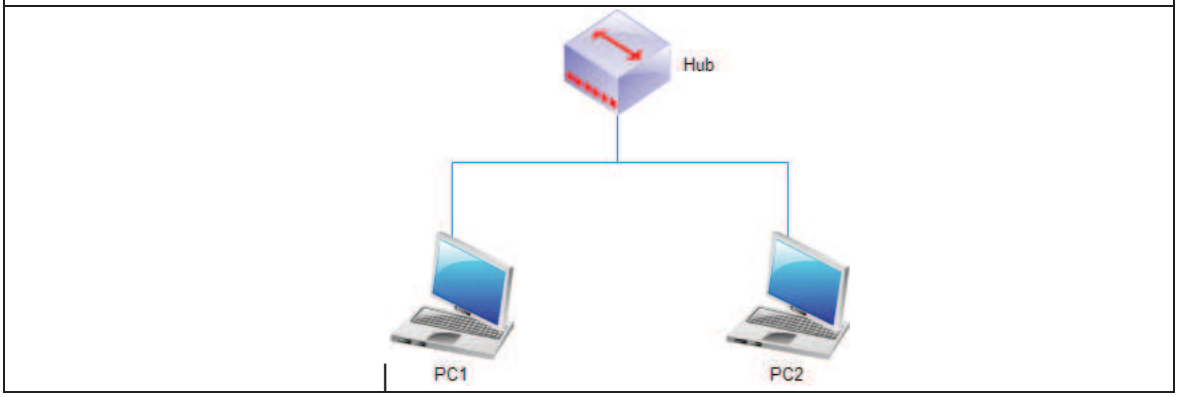


Gráfico 18. Conexión con Hub

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

2.2.34.4.2 Switch

El Switch o Conmutador, es un Hub mejorado. Permite múltiples vías de comunicación simultáneas punto a punto, la principal diferencia con el Hub radica en que el Switch puede reenviar un mensaje a un host específico, el mismo que acepta y decodifica las tramas[1] para leer la dirección física (**MAC**[2]) del mensaje, reduciendo así colisiones ya que tiene una tabla llamada direcciones MAC, en la cual se registran todos los puertos activos y la dirección MAC del host o anfitrión al que están unidos. Cuando se envía un mensaje entre hosts, el switch verifica si la dirección MAC de destino está en la tabla. Si el registro existe, se crea una conexión temporal, denominada circuito, entre la fuente y los puertos de destino. Este circuito proporciona un canal dedicado sobre el cual los dos hosts pueden comunicarse evitando que se produzcan

[1]**Trama** – "Equivalente de paquete de datos o Paquete de red, en el Nivel de enlace de datos del modelo OSI. Una trama constará de cabecera, datos y cola. En la cola suele estar algún chequeo de errores. En la cabecera habrá campos de control de protocolo. La parte de datos es la que quiera transmitir en nivel de comunicación superior, típicamente el Nivel de red." *Fuente: http://es.wikipedia.org/wiki/Trama_de_red*

[2]**MAC** – Media Access Control o Control de Acceso al Medio. "Corresponde de forma única a una tarjeta o dispositivo de red. Se conoce también como dirección física, y es única para cada dispositivo. Está determinada y configurada por el IEEE (los últimos 24 bits) y el fabricante (los primeros 24 bits)." *Fuente: http://es.wikipedia.org/wiki/Direcci%C3%B3n_MAC*

colisiones. Adicionalmente se ofrece un ancho de banda mayor y su rendimiento se incrementa.

Puesto que en un Cluster se pueden ir añadiendo más nodos a la red y se incrementa la demanda de ancho de banda es indispensable que se utilice un Switch como se muestra en los Gráficos 19 y 20

El Switch trabaja en las dos primeras capas del modelo OSI[3], capa física y enlace de datos.

Switch o Conmutador

Gráfico 19. Switch o Conmutador

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

Conexión con Switch

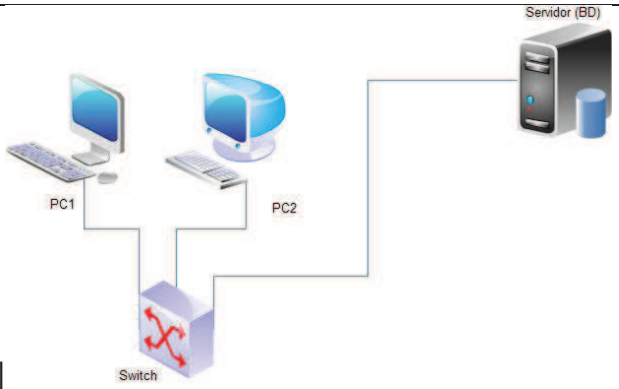


Gráfico 20. Conexión con switch

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

[3]**OSI** – Open System Interconnection ó Modelo de Interconexión de Sistemas abiertos. "Creado por la Organización Internacional para la Estandarización (ISO) en el año 1984. Es un marco de referencia para la definición de arquitecturas en la interconexión de los sistemas de comunicaciones." *Fuente: http://es.wikipedia.org/wiki/Modelo_OSI*

2.2.34.4.3 Router

El Router o Encaminador apareció a finales de los años 80.Es utilizado como una conexión para aislar el tráfico entre dos redes y a su vez conectar una red local con otras redes. Los routers como los switchs son capaces de descifrar y leer los mensajes que se envían a los mismos. Estos trabajan en la capa 3 del modelo OSI que corresponde a la distribución de la red, los enrutadores dirigen el tráfico y realizan otras funciones esenciales para el funcionamiento eficiente de la red. A diferencia del Switch, que sólo decodifica la trama que contiene la información de dirección MAC, los routers decodifican el paquete que está encapsulado dentro de la trama y toma las direcciones IP de los hosts de destino y origen, así como los datos de mensajes enviados entre ellos. El router lee la parte de red de la dirección IP de destino y lo utiliza para encontrar cuál de las redes conectadas es la mejor forma de reenviar el mensaje al destino como se muestra en los Gráficos 21 y 22.

Router o Encaminador

Gráfico 21. Router o Encaminador

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

Conexión con Router

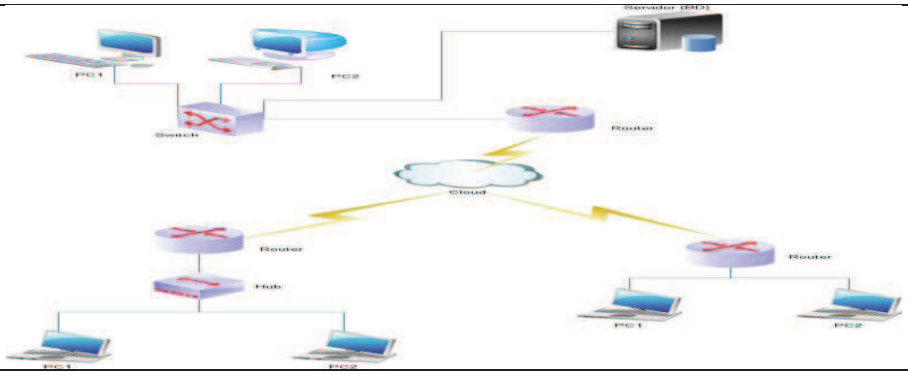


Gráfico 22. Conexión con Router

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

CAPÍTULO 3

3.1 Diseño e Implementación del Cluster

En este capítulo se realiza el diseño e implementación del Cluster del Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE, sobre el hardware que esté disponible, pudiendo ser un Cluster homogéneo o heterogéneo. De igual manera se seleccionará el sistema de red que permita que todos los nodos del Cluster se comuniquen basado en una de las arquitecturas de red: bus, estrella o anillo, como paso previo a la implementación del Cluster con herramientas libres bajo una plataforma Linux.

3.2 Diseño de Clusters

Para el diseño del Cluster hay que tener en consideración ciertos aspectos importantes como:

- Misión general del Cluster.
- Arquitectura general para el Cluster.
- Hardware del Cluster
- Sistema Operativo, software, y aplicaciones que se podría utilizar en el Cluster; para nuestro caso se utilizarán herramientas de software libre.

El primer paso es analizar y reflexionar sobre el problema que se desea resolver, para determinar la misión y objetivos del mismo, en este proyecto se desea dar solución a través de un Cluster a problemas que requieran gran capacidad de procesamiento, almacenamiento y memoria y que además sea un apoyo a la formación académica de los estudiantes.

Cabe aclarar que no se propone la construcción de un Cluster específico pues no se implementará para un solo propósito o problema en particular, sino que esta solución de Clustering ejecutará programas de propósito general que se propongan en el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE o a su vez adaptar servicios y/o programas en el Cluster, para que ayuden a la solución rápida de problemas en

los diferentes departamentos que requieran alto procesamiento, memoria, y/o almacenamiento dentro de la Institución.

Más allá de las características de velocidad de procesamiento, memoria, ancho de banda, dispositivos de entrada/salida, acceso al disco duro; y, velocidad del bus de datos, en cada uno de los nodos del Cluster es importante la manera de conectarlos, pues esto tendrá gran impacto en su eficiencia.

Hay que tomar en cuenta, al momento del diseño, si el Cluster será homogéneo o heterogéneo. Esto significa si trabajará con hardware del mismo tipo o si se adquirirá nuevos recursos. La idea del proyecto es utilizar hardware reciclado, que evitará hacer una inversión de nuevos equipos o partes, considerando un presupuesto restringido. Sin embargo, esto no quiere decir que el cluster se regirá a esta premisa. El enfoque siempre será la reutilización del hardware disponible para obtener resultados iguales o superiores a un equipo costoso y de alta tecnología.

3.2.1 Cluster Homogéneo

En el caso de disponer de sistemas idénticos, será conveniente la implementación de este tipo de cluster en la que todos los nodos tendrán la misma tarjeta madre, memoria, discos duros y tarjetas de red, lo que garantiza que el trabajo entre ellos será más fácil debido a que el sistema operativo gestionará de la misma manera todos los recursos existentes.

3.2.2 Cluster Heterogéneo

Para la construcción de un Cluster heterogéneo hay dos formas de hacerlo: La primera y más común es construirlo en base a diferentes tipos de ordenadores de diferentes fabricantes, lo importante a ser considerado es la diferencia que tendrán los nodos en su funcionamiento y eficiencia pues tienen diferente procesador, memoria, elementos de red, etc. Como se explica a continuación:

3.2.2.1 Cluster de Red

En el caso de tener una mezcla de hardware con diferentes tecnologías de red, como pueden ser redes Ethernet de 10 Mbit/s, 100 Mbit/s, 1Gb/s u otras, existirá una diferencia grande de velocidad de transmisión lo que en determinado momento puede producir congestión. Por lo que es recomendable que los equipos a utilizar en el Cluster tengan capacidades similares en las tarjetas de red.

3.2.2.2 Cluster de Software

Hace varios años, era necesario crear versiones del software de gestión de Clusters de acuerdo al tipo de sistema, además de mantener a todos los nodos actualizados y sincronizados, provocando problemas de administración. En la actualidad las distribuciones del sistema operativo Linux y de otros han solucionado y mejorado este problema haciendo que la tarea de administración de un Cluster sea más fácil y confortable.

3.2.2.3 Programación

El código debe ser escrito para apoyar el principio del mínimo común denominador que significa que los datos soportados por el nodo menos poderoso en el Cluster sean los más simples. Los ordenadores mixtos y más poderosos tienen atributos que no pueden alcanzar los ordenadores menos potentes. Por ejemplo, equipos con arquitecturas de 64 bits son capaces de procesar operaciones tanto de enteros como de punto flotante, no así en arquitecturas de 32 bits en las que se presentan problemas de cálculo que pueden disminuir la exactitud del cálculo. En el caso de que el Cluster esté destinado a la realización de cálculos en punto flotante y que se requiera niveles de precisión extrema. Esto se puede evitar con bibliotecas que normalizan las representaciones de los datos procesados.

3.2.2.4 Sincronización

Es quizás el aspecto más sensible de un Cluster heterogéneo, puesto que las máquinas tienen diferentes perfiles de rendimiento. El código se ejecuta a tasas diferentes en los distintos nodos. Esto puede provocar la presencia de

cuellos de botella el momento en que un proceso ejecutándose en un nodo deba esperar los resultados de un cálculo realizado por un nodo más lento.

El segundo tipo de Cluster Heterogéneo está construido por ordenadores diferentes pero dentro de la misma arquitectura, por ejemplo un grupo de equipos Intel los cuales son de generaciones diferentes (Pentium Core2Quad, CoreIX – siendo "X" igual a 3, 5, o 7 – o cualquier arquitectura en caso de Intel) o equipos de la misma generación pero de diferentes fabricantes. Este tipo de mezcla de equipos no representa un problema de control de versiones de software, sin embargo, pueden aparecer problemas con los drivers o controladores de los dispositivos. Por ejemplo, todos los ordenadores tienen diferentes Interfaces de Red, esto hará que ciertos ordenadores con tarjetas de un fabricante "X" tengan un mejor desempeño que las del fabricante "Y", o existan diferencias de rendimiento en sus chips.

Estas son consideraciones a tener en cuenta el momento de utilizar hardware que se tenga a disposición.

3.2.3 Requerimientos de Hardware

Para el desarrollo e implementación del cluster del Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE, se dispondrá de toda la infraestructura computacional existente en el Data Center departamental que garantiza total interconexión y comunicación entre los nodos hijos y el nodo principal usando el cableado estructurado existente. Se dispone además de estructuras físicas de Rack sobre las cuales se instalarán el nodo principal o Frontend y los nodos hijos.

El modelo de Cluster a diseñar e implementar será de tipo heterogéneo y clasificado como un *Cluster de Alto Rendimiento (HPC)* de acuerdo a la descripción realizada en el capítulo 2.

Es necesario tomar en cuenta los cuatro componentes principales de un cluster:

- Mainboard - Placa principal ó tarjeta madre - motherboard.
- CPU - procesador.
- Disco de almacenamiento.
- Adaptador de Red (NIC).

3.2.4 Composición del Cluster

3.2.4.1 Hardware

3.2.4.1.1 Frontend

Equipo DELL - PowerEdge 2900.

- **Procesador**
  - 1 Intel® Xeon® E5310
  - Núcleos: 4 – Quad Core.
  - Velocidad de reloj: 1.60 Ghz
  - Total Cache: Nivel 2 (L2) – 8 MB.
- **Memoria RAM**
  - Memoria instalada: 2 GB DDR2 (FB-DIMM[4]).
  - Velocidad de bus: 533 MHz.
- **Tarjeta de Red (NIC)**
  - 2 NetXtreme II BCM5708 Gigabit Ethernet.
- ***Almacenamiento***
  - Disco Duro: 140 GB – SCSI – Soporte de RAID (PowerEdge Expandable RAID controller 5).

[4]**FB-DIMM** - Fully-Buffered Dual Inline Memory Module, es una variante de las memorias DDR2, diseñadas para aplicarlas en servidores, donde se requiere un transporte de datos rápido, efectivo, y coordinado. ***Fuente:*** *http://es.wikipedia.org/wiki/FB-DIMM*

3.2.4.1.2 Nodos

Computador 0-X

- ***Procesador***
  - Intel Core 2 Quad Q6600.
  - Núcleos: 4 - Quad Core.
  - Velocidad de reloj: 2.40 Ghz.
  - Total Cache: Nivel 2 (L2) – 8 MB.
- ***Memoria RAM***
  - memoria instalada: 4 GB DDR2.
  - Velocidad de bus: 800 MHz.
- ***Tarjeta de Red (NIC)***
  - Intel Gigabit Ethernet
- ***Almacenamiento***
  - Disco Duro: 500GB SATA.

3.2.4.2 Software

3.2.4.2.1 Sistema Operativo

El Sistema Operativo que se implementa en el proyecto, es una distribución de Linux basado en el Sistema Operativo CentOS 6.2 diseñado para la creación del Clusters llamado **Rocks Clusters**.

Rocks Clusters

"El proyecto originalmente se denominó **NPACI Rocks**, fue iniciado por la NPACI y la SDSC en el año 2000, y financiado inicialmente con una subvención de la NSF (2000-2007)[5], actualmente está financiada por la siguiente subvención de la NSF[6]. Rocks se basó inicialmente en la distribución Red Hat Linux, sin embargo las versiones más modernas de Rocks están basadas en CentOS, con un instalador Anaconda modificado, que simplifica la instalación

[5] **NSF**:NationalScienceFoundation, subvention - SCI: Delivering Cyberinfrastructure: From Vision to Reality. ***Fuente:*** *http://www.nsf.gov/awardsearch/showAward?AWD_ID=0438741*

[6] SDCI: NMI: Improvement: The Rocks Cluster Toolkit and Extensions to Build User-Defined Cyberenvironments. ***Fuente:*** *http://www.nsf.gov/awardsearch/showAward?AWD_ID=0721623*

'en masa' en muchos computadores. Rocks incluye muchas herramientas (como MPI) que no forman parte de CentOS pero que es un componente integral al momento de integrar un grupo de ordenadores en un Cluster.

Las instalaciones pueden ser personalizadas con paquetes de software adicionales, utilizando CD´s especiales (llamados Roll CD). Los "*Rolls*" extienden el sistema integrando automáticamente los mecanismos de gestión y empaquetamiento usado por el software base, simplificando ampliamente la instalación y configuración de un gran número de computadores. Se han creado más de una docena de Rolls, incluyendo el *SGE* roll, *Condor* roll, *Java* roll, y *Ganglia* roll*,* entre otros.

Por otro lado, es la distribución más empleada en el ámbito de Clusters, por su facilidad de instalación e incorporación de nuevos nodos, incorpora gran cantidad de software para el mantenimiento y monitorización lo que podría suponer en algunos casos una limitación."

Fuente: http://es.wikipedia.org/wiki/Rocks_Clusters

La compilación a ser implementada en el Cluster es la versión Rocks 6.0 (Mamba). Utiliza como servidor de base de datos a MySQL en el que se almacenará información sobre la estructura.

Hardware soportado por Rocks Cluster

Procesadores:

- x86 (ia32, AMD Athlon, etc.)
- x86_64 (AMD Opteron and Intel 64).

Tecnología de red:

- Ethernet.
- Myrinet.
- Infiniband.

Requerimientos Mínimos

El diagrama físico de la solución se presenta en el Gráfico 23.

Frontend:

- Disco Duro: 30 GB.
- Memoria RAM: 1 GB.
- Tarjeta de Red: 2 puertos físicos (ej., "eth0" y "eth1").
- Orden de inicio en BIOS: CD/DVD, Disco Duro.

Nodos:

- Disco Duro: 30 GB.
- Memoria RAM: 1 GB.
- Tarjeta de Red: 1 puerto físico (ej., "eth0").
- Orden de inicio en BIOS: CD/DVD, PXE (inicio por red), Disco Duro.

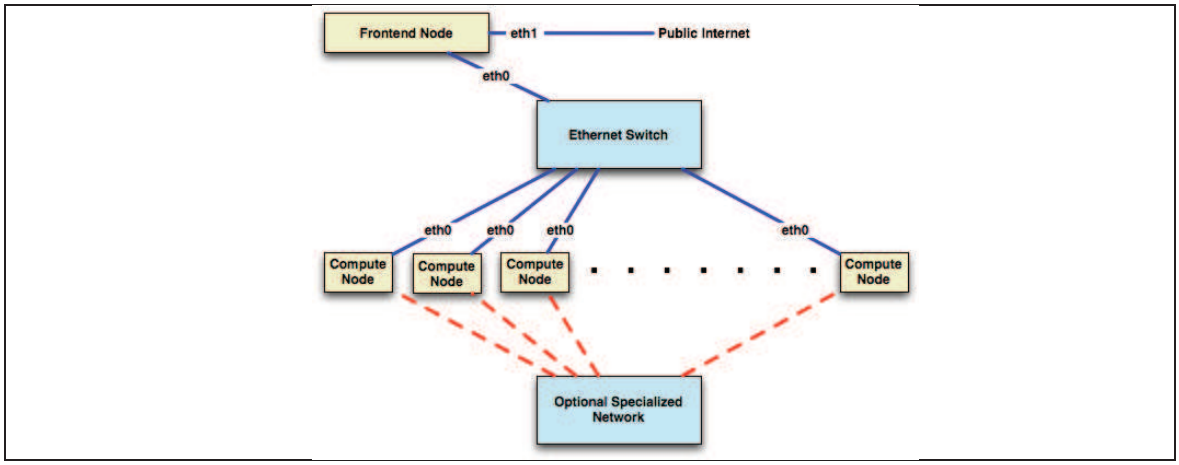


Gráfico 23. Diagrama físico de conexión del Frontend y Nodos

Fuente: http://www.rockclusters.org/roll-documentation/base/5.5/getting-started.html
Elaborador por: Juan Daniel Carrillo

3.2.4.2.2 Sistema de Archivos

Es una parte importante en cualquier Sistema Operativo ya que constituye el repositorio de programas y datos. En Linux hay dos significados, el primero se refiere al concepto de una jerarquía que parte del directorio raíz con

diferentes particiones montadas bajo este. No existe ninguna letra para la unidad como en Microsoft Windows para identificar las particiones; todo es visto como un archivo o directorio; y el segundo se refiere a los formatos específicos para representar archivos o directorios en un disco duro o memoria, como por ejemplo EXT2 o EXT3

Para el caso de estudio, el sistema de archivos se encuentra estructurado como se muestra en la Tabla 5, en un disco duro de 140 GB.

Sistema de archivos

Tabla 5. Sistema de archivos

Capacidad (MB)	Punto de Montaje	Tipo
40000	/	ext4
20000	/var	ext4
4000		Swap
75390	/export (EXTENDIDA)	ext4

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

En el caso de un Sistema Cluster existe un *sistema de archivos distribuidos*, llamado **NFS** (**N**etwork **F**ile **S**ystem) ó Sistema de archivos de redque es el más utilizado y conocido con el fin de interconectar los datos entre nodos.

NFS (Network File System)

Es un protocolo utilizado en Clusters, debido a la capacidad de ofrecer una arquitectura cliente/servidor en un sistema de archivos distribuidos y servicios RPC (Remote Procedure Call) o llamada de procedimientos remotos que al ser invocados, el proceso principal queda suspendido y los parámetros son transferidos a través de la red al nodo en el cual se va a ejecutar el procedimiento; una vez finalizado la respuesta es devuelta con los resultados por la red al nodo que lo invocó o solicitó.

Por ejemplo las exportaciones de los archivos del servidor (Frontend – nodo primario) a un cliente (nodos secundarios) son realizadas como si fueran

locales, manteniendo de manera independiente las solicitudes que cada cliente realiza. NFS no guarda la información de quién realiza la petición, por lo tanto éste debe contener toda la información necesaria para su ejecución cada vez que se haga una llamada del cliente al servidor.

Cabe mencionar que NFS es un sistema básico y funcional de sistemas de archivos de red ya que existen otros en el mercado, claro está que cada uno se ajusta al nivel de tratamiento que el Cluster va a ofrecer y manejar.

Para el Cluster se utilizará el protocolo NFS, sobre el cual se encontrarán montados algunos archivos de prueba. Estos estarán disponibles para todos los nodos con el objetivo de ejecutar tareas que permitan realizar pruebas de comunicación y medir los tiempos de ejecución de los mismos.

Estas son las principales características de un sistema de archivos local.

- **Transparencia en la Red:** los archivos remotos pueden ser accesados mediante las mismas operaciones o sistema de llamadas que se utilizan para acceder a los archivos locales.
- **Transparencia en la ubicación:** El nombre de un archivo no está vinculado a su ubicación en la red, ya que la ubicación del servidor de archivos no parte de la ruta del archivo.
- **Independencia en la ubicación:** Cuando cambia la ubicación física en la cual se aloja un archivo, el nombre del equipo servidor de archivos no es parte de la ruta del archivo.

3.2.4.2.3 Herramientas de Administración

Ganglia

Surgió del Proyecto “Berkeley Millenium” de la Universidad de California. Fue financiado por NPACI (National Partnership for Advanced Computational Infrastructure) o AsociaciónNacional de InfraestructuraAvanzadaComputacional, y NPACI financiado por la NSF (National Science Foundation) o Fundación Nacional de la Ciencia. Bajo licencia BSD (Berkeley Software Distribution),

Ganglia es un proyecto “open-source” o de código abierto, el cual permite monitorear en tiempo real y representando gráficamente datos informativos del estado del Cluster, es una aplicación robusta que se adapta a varios sistemas operativos y diferentes tipos de arquitecturas, siendo así una aplicación que permite la escalabilidad alcanzado a gestionar miles de nodos.

Mediante un navegador web y a través de la página web que se configura con los parámetros iniciales al momento de instalar el Cluster, Ganglia permite el monitoreo que reúne valores a través de monitores sobre cada nodo para obtener diferentes métricas como: la carga del CPU, memoria, uso del disco duro, tráfico de la red, etc. Estas métricas son enviadas a través de la red privada del Cluster y son utilizadas por el Frontend (nodo principal) para generar los gráficos históricos. Adicional a las métricas, un mensaje de “heartbeat[7]” o latido de cada nodo es recogido por los monitores de Ganglia, con el objetivo de conocer si alguno de los nodos tiene problemas, pues cuando un número de heartbeat en un nodo se pierde este es declarado como “down” o caído, marcándolo con un fondo rojo para diferenciarlo y para que se lo de un tratamiento a ése nodo. La interfaz se muestra en el Gráfico 24.

[7] **Heartbeat:**“Es un servicio que provee servicios de infraestructura de agrupamiento (cluster) a clientes. Permite a los clientes saber si uno de los nodos está presente o ausente, intercambiado fácilmente mensajes entre éstos.” *Fuente:http://www.alcancelibre.org/staticpages/index.php/como-cluster-heartbeat-centos*

Interfaz de Ganglia

Nodos activos

Nodo caído (down)

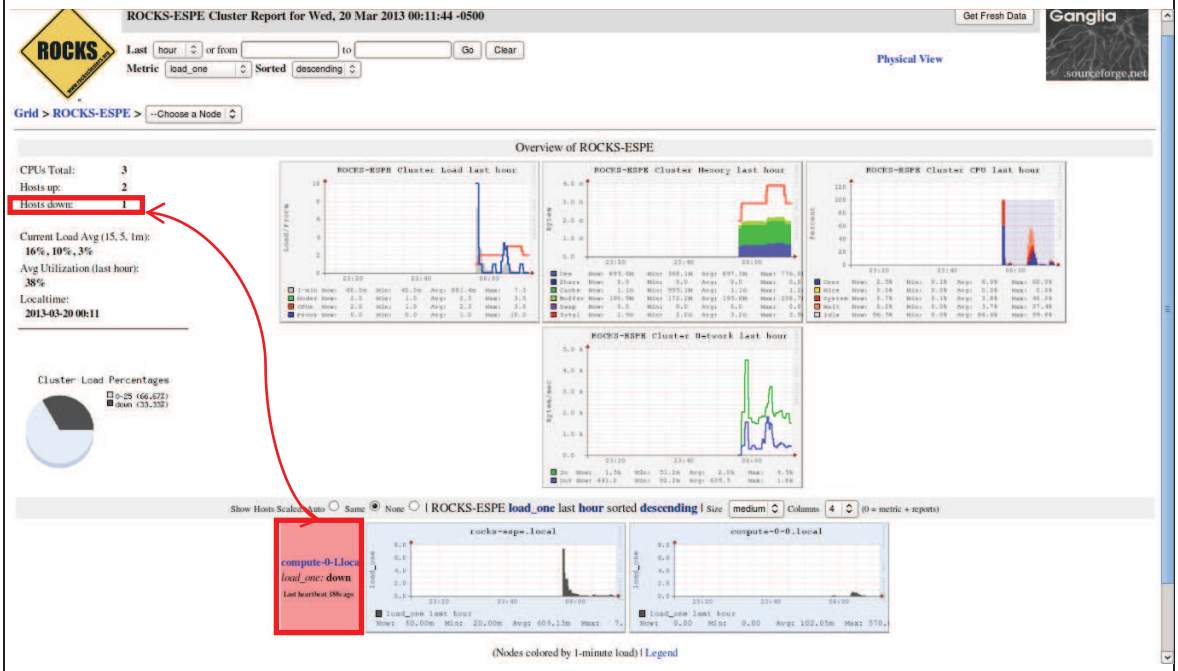


Gráfico 24. Interfaz Ganglia

Fuente: http://espe-rocks.espe.edu.ec:8080 o http://localhost:8080
Elaborador por: Juan Daniel Carrillo

El monitor de Ganglia está constituido por tres partes:

- **gmond**: Es un programa demonio, el cual se encuentra instalado en los nodos que necesitan ser monitoreados. Este recopilará estadísticas de los monitores, las que serán enviadas y recibidas desde el nodo principal o Frontend.
  - Para el caso en que un emisor este configurado como “mute=no” (silencio), recopilará métricas básicas como carga del sistema o uso del CPU.
  - Para el caso en que un receptor este configurado como “deaf=no” (sordo), recopilará todas las métricas enviadas desde otro host.
- **gmetad**: Es un programa demonio que obtiene periódicamente datos estadísticos de “gmond” y almacena sus métricas, esto a su vez es utilizado por la Interfaz web para la generación de información al usuario en el Frontend.
- **Ganglia-web (Interfaz web):** Permite visualizar gráficamente las estadísticas o métricas obtenidas de los nodos como “gmetad” mediante una página web, en tiempo real como: uso de CPU, memoria, etc.

En Rocks Clusters, los programas demonios de “gmond” en los nodos son ejecutados en modo “deaf” o sordo, de manera que los nodos aledaños solo reportan sus datos a Ganglia en el Frontend más no escuchan la información de otros.

Se ha observado durante las pruebas realizadas en el Cluster, que al configurar a los nodos en estado “down” o caídos, no se produce un refresco inmediato en la actualización del sistema DNS (**D**omain **N**ame **S**ystem o Sistema de Nombres de Dominio), por lo que es necesario utilizar los siguientes comandos para reiniciar los servicios de “gmond” y “gmetad” en el Frontend para solucionar el problema.

Reinicio de servicios Ganglia
service gmond restart # service gmetad restart

Para acceder al monitor de Ganglia, se inicia un navegador web, y se ingresará a la siguiente dirección: http://espe-rocks.espe.edu.ec o a su vez por la dirección IP privada previamente configurada en la instalación del Cluster, http://192.168.15.20

3.2.5 Medios de red e interfaces

3.2.5.1 Topología de red

Para la solución de Clustering se utilizó la topología en estrella que es la más utilizada en redes locales donde los nodos no se encuentran conectados entre sí; sino por el contrario, comparten un dispositivo común, un switch o conmutador de 24 puertos de los cuales 7 pertenecen al sistema de Cluster que controla el tráfico que pasa a través de la red y previene posibles colisiones o algún fallo.

3.2.5.2 Switch

La demanda de ancho de banda que irá aumentando en el sistema determina que se utilice un switch o conmutador, el equipo aceptará los paquetes de información desde los cables de par trenzado y a su vez evitará, que la información no se difunda a todos los nodos, sino al nodo receptor.

El sistema dispondrá de un switch capa 2 de marca 3comproporcionado por el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE.

3.2.5.3 Cable de Red

La interconexión de los dispositivos se realiza mediante un cableado estructurado Categoría 5, que garantizan velocidades de transmisión de 10/100/1000 mbps usando tecnología FastEthernet conectados a un patch panel o bahía de rutas, desde el que se distribuye a cada nodo que debe estar etiquetado para su mejor identificación en la red.

3.2.5.4 Tarjetas de Red

Cada nodo cuenta con una tarjeta de red marca Intel que están embebidas o integradas al mainboard o tarjeta principal, con las especificaciones que se mencionaron anteriormente; en el caso de un único nodo este tendrá dos tarjetas de red ya que trabajará como nodo normal y como nodo principal o Frontend.

3.3 Implementación del Cluster

3.3.1 Instalación y configuración del Frontend

3.3.1.1 Consideraciones para la instalación del Frontend

La instalación del nodo principal debe considerar: roles, compilaciones de proyectos, configuraciones, entre otros. En este equipo se administra todo el cluster. Adicionalmente, éste mantendrá toda la información y es donde los usuarios del sistema se registrarán para declarar los trabajos a ser procesados así como para ver los resultados.

Previa a la instalación se debe considerar la dirección de red que debe ser proporcionada por el jefe de laboratorio para el sistema Cluster. La **IP pública** asignada es la **190.15.140.49** con una **máscara de sub-red** de 24 bits o la **255.255.255.0**, DNS es **200.93.192.148** y puerta de enlaces predeterminadas apuntarán a la dirección **190.15.140.1**. Así también se tiene en cuenta la manera en que se particionará el disco duro que será de la manera indicada en la sección de “Sistema de Archivos”.

Las credenciales previstas para la administración maestra de éste, serán:

- Usuario :**root**
- Contraseña :**E$peClu$ter2012**.

Con esta información se procede a iniciar la instalación del Cluster en la que se incluirá información adicional durante el proceso.

A continuación la Tabla 6 resume la información esencial para el proceso de instalación del sistema.

Tabla 6. Consideraciones para la instalación del Frontend

ITEM	DESCRIPCIÓN
Dirección IP:	190.15.140.49
Máscara de sub-red:	255.255.255.0
Puerta de enlace:	190.15.140.1
Dirección DNS:	200.93.192.148
Usuario:	Root
Contraseña:	E$peClu$ter2012

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

3.3.1.2 Descripción de roles incluidos en Rocks Clusters

- **Área 51:** Contiene utilidades y servicios para analizar la integridad de los archivos y el Kernel en el Cluster.
- **Bio:** Este rol viene de Bioinformática, el cual utiliza técnicas de matemática aplicada, informática, estadística y ciencias de la computación para resolver problemas biológicos como: formación del genoma, alineación de estructura de proteínas, predicción de estructura de proteínas, predicción de interacciones de expresión y proteínas del gen y el modelado de la evolución. Bio aún está siendo desarrollado para estandarizarlo y facilitar la instalación de la herramienta. Muy útil para el departamento de Biotecnología de la Universidad de las Fuerzas Armadas-ESPE.
- **Ganglia:** Es una herramienta que permite monitorear el sistema de Cluster.
- **HPC:** Este rol proporciona herramientas de software que pueden ser utilizadas para gestionar las aplicaciones paralelas del Cluster. Dentro del rol incluyen paquetes como: MPI sobre ambientes de red (OpenMPI, MPICH, MPICH2), PVM.
- **KVM:** Permite configurar máquinas virtuales (VM) dentro de Rocks Cluster. Dentro del Frontend se puede configurar las máquinas virtuales en los nodos agregando como un nuevo dispositivo de tipo contenedor

de VM (VM container appliances), en el cual se alojarán y ejecutarán estas máquinas. Todo el tráfico de la red será encapsulado o agrupado dentro de una VLAN, de forma exclusiva para cada Cluster virtual, tendrá su propia VLAN.

3.3.1.3 Instalación y configuración de ROCKS CLUSTER bajo el Sistema Operativo CentOS 6.2

La instalación se la realiza mediante un DVD jumbo de la versión Rocks 6.0 (Mamba), el cual contiene todos los roles que se deseen incluir en la instalación, así también dentro de la página de Rocks Cluster http://www.rocksclusters.org/wordpress/?page_id=396 se puede descargar CD's con roles independientes los que pueden facilitar la instalación con lo estrictamente necesario.

Al momento de iniciar el equipo, con la unidad de CD o DVD que contiene la instalación, preguntará en el arranque lo que se necesita hacer y se mostrará la palabra "boot:" y al lado se digitará "**build**", con esto se iniciará la instalación.

Al momento de instalar Rocks Cluster, se debe escoger varios roles entre los cuales los imprescindibles son: base, kernel, OS, web-server. Para nuestro caso se escoge todos los existentes, a excepción de los roles: Condor y SGE (Sun Grid Engine), pues se utilizará roles similares, potentes, y se ha tomado esta decisión debido a la variedad de información existente que incluye: TORQUE y MAUI. Siendo **TORQUE** (**T**erascale **O**pen **R**esource **M**anager) - el gestor de recursos -, su tarea consiste en recoger información sobre el estado de los nodos y sus trabajos; mientras que **MAUI** -el planificador-, su tarea consiste en decidir cuándo y dónde ejecutar los trabajos enviados a TORQUE. La tabla 7 muestra la compatibilidad entre TORQUE y MAUI

Tabla 7. Roles para compatibilidad con TORQUE/MAUI

Cuadro de algunos roles para la compatibilidad con TORQUE/MAUI

Roll	Requerido	Opcional	Conflictos
area51		X	
base	X		
bio		X	
condor		X	
ganglia		X	
grid		X	
hpc	X		
java		X	
kernel	X		
os	X		
kvm		X	
sge			X
viz		X	
web-server		X	
xen		X	

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

Luego de la instalación e inicialización del sistema se deberá agregar **TORQUE Roll**.

A continuación se describe de manera breve las funciones del gestor y planificador de recursos, además se explica que es un Sistema por Lotes (Batch System) y los dos tipos de trabajos por lotes existentes.

Batch System – Sistema por Lotes

Se conoce como Sistema por Lotes a la ejecución de un conjunto de programas en el que cada uno completa su tarea antes de que otro comience, sin la supervisión directa del usuario; se aplica a tareas repetitivas sobre grandes conjuntos de información con el objetivo de evitar la gestión manual. Mediante el uso de scripts (procedimientos) se puede especificar qué es lo que

se quiere ejecutar y qué recursos serán reservados para los programas que se ejecutan por lotes.

Un Sistema por Lotes se utiliza para monitorizar y controlar los trabajos que se ejecutan en un sistema. Impone límites en tiempo de ejecución (walltime), así como el número de trabajos que se ejecutan al mismo tiempo. Para ejecutarlo, el sistema de lotes asigna los recursos solicitados en el archivo, establece un entorno para ejecutar el trabajo, y después se lo ejecuta.

Tipos de trabajo por lotes:

- **Batch Jobs – Trabajo por Lotes:** Un trabajo por lotes es una aplicación que se ejecuta de forma independiente en cada nodo del Cluster sin la necesidad de interactuar profundamente con otros. Este tipo de aplicación por lo general realiza los cálculos en un conjunto de datos y al final del trabajo pasa el resultado a un nodo de control responsable de la recogida de los resultados de todos los nodos.
- **Batch Parallel Jobs – Trabajos Paralelos por Lotes:** Un trabajo paralelo es un trabajo que tiene que hablar con el resto de nodos de computación del clúster. Esta operación necesita el paso de mensajes y la actividad de la red.

En los Clusters de cómputo de alto rendimiento, los usuarios suelen presentar trabajos por lotes a las colas, que son clases de nodos de computación, gestionados por un gestor de recursos, como: TORQUE. Con frecuencia los Clusters emplean planificadores de tareas independientes, como Moab, para enviar los trabajos por lotes en base a la disponibilidad de recursos informáticos, los requisitos de trabajo especificados por los usuarios y las políticas de uso establecidas por los administradores del clúster.

“En un sistema por lotes existe un gestor de trabajos, encargado de reservar y asignar los recursos de las máquinas a las tareas que hay que ejecutar. De

esta forma, mientras existan trabajos pendientes de procesamiento, los recursos disponibles estarán siempre ocupados ejecutando tareas."

"Ventajas:

- Permite compartir mejor los recursos de un ordenador entre muchos usuarios, al no competir por éstos de forma inmediata.
- Realiza el trabajo en el momento en el que los recursos del ordenador están menos ocupados, dando prioridad a tareas interactivas.
- Evita desaprovechar los recursos del ordenador sin necesidad de interacción y supervisión humanas continuas.
- En ordenadores caros o supercomputadores, ayuda a amortizar el coste manteniendo altos índices de utilización.

Inconvenientes:

- El principal inconveniente de la ejecución por lotes frente a la ejecución interactiva es que hay que conocer y planificar cuidadosamente la tarea a realizar. Al carecer de supervisión por parte del usuario, cualquier tipo de error puede producir resultados inútiles o, simplemente, inexistentes."

Fuente: http://es.wikipedia.org/wiki/Procesamiento_por_lotes

La razón para utilizar un Cluster es acelerar la ejecución de un programa.

Gestor y planificador de recursos o tareas

Un administrador de recursos (**TORQUE**) provee una funcionalidad a bajo nivel para iniciar, mantener, cancelar/eliminar y monitorear trabajos, sin estas capacidades un planificador o programador de tareas no podría controlar dichos trabajos. Mientras que el objetivo de un planificador de recursos (**MAUI**) permite que los usuarios o administradores del sistema tengan un control sobre las tareas o trabajos ejecutados o a ejecutarse y esto les facilite su administración y el poder analizar o comprender cuál es la carga de trabajo y los recursos disponibles en su Sistema Cluster.

TORQUE

TORQUE es un proyecto de código abierto por parte de Adaptive Computing en colaboración de su propia comunidad. Está basado en el original Sistema Portable por Lotes de Código Abierto (OpenPBS – Portable Batch System), posee un planificador incorporado y utilizado únicamente como un administrador de recursos con un planificador que hace peticiones asimismo, pero éste se puede integrar con el planificador MAUI (no comercial) o MOAB (comercial) para potencializar su uso de manera global como la programación y la administración del Cluster.

MAUI

Este planificador puede ser considerado como un motor de políticas que permite controlar cuándo, dónde y con qué recursos, como: procesadores, memoria, y disco se asignan los trabajos a ejecutar. Es decir MAUI hace gestión del sistema, proporcionado mecanismos que ayudan a: optimizar de manera inteligente el uso de recursos, monitorear el rendimiento del sistema y ayuda a diagnosticar problemas.

3.3.1.4 Instalando y configurando "TORQUE Roll"

Este rol se lo puede descargar de la siguiente dirección dentro de la página de Rocks Clusters

ftp://ftp.uit.no/pub/linux/rocks/torque-roll/6.0.0.

Dentro de éste viene empaquetado TORQUE y MAUI, con parámetros para su instalación y configuración automática para Rocks Cluster; para ello se deberá seguir las siguientes instrucciones

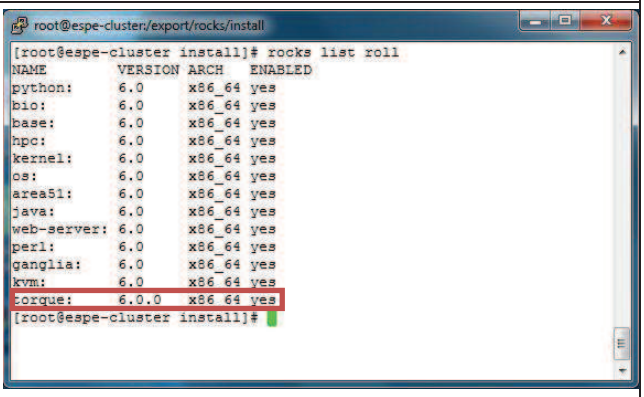
Paso 1) Proceso para agregar un rol – TORQUE/MAUI
rocks add roll torque-6.0.0-1.x86_64.disk1.iso # rocks enable roll torque # cd /export/rocks/install # rocks create distro # rocks run roll torque | sh # reboot

Después de este proceso se necesitará reiniciar el Frontend y se procederá a reconstruir los nodos, si hay nodos instalados antes de agregar un nuevo rol o que ya esté en funcionamiento todo el Cluster, si no es el caso, el siguiente paso se puede obviar y proceder con la instalación inicial de los nodos.

Paso 2) Reconstruyendo los nodos después de instalación del rol y reiniciado el Frontend
rocks run host compute command=/boot/kickstart/cluster-kickstart

Con este último comando se obliga a todos los nodos a reiniciase forzosamente para la reinstalación del sistema con el nuevo rol, esto sucederá en caso de haber instalado y configurado los nodos secundarios únicamente con los roles de la instalación primaria o por defecto, pero para nuestro proceso se ha de instalar y configurar todo lo necesario en el Frontend antes de desplegar la instalación a todos los nodos del sistema.

Para constatar que el rol fue instalado en el Cluster se ingresará la siguiente instrucción para listar todos los roles:

Lista de roles del Cluster	
# rocks list roll	

Finalmente para concluir con la configuración general de TORQUE se tendrá que crear o definir la cola (queue) por defecto para poder procesar o ejecutar los trabajos. De no hacerlo, se generará un error similar a:

"**Qsub Error Message Specified Queue | MSG=cannot locate queue**"

Mediante el siguiente comando con los siguientes parámetros se puede configurar la cola para el administrador de recursos TORQUE.

Comando	Parámetro	Descripción
qmgr		Sistema de gestión bps
	-c	Para ejecutar un único comando.

```
qmgr -c "set server scheduling=true"
qmgr -c "create queue batch queue_type=execution"
qmgr -c "set queue batch started=true"
qmgr -c "set queue batch enabled=true"
qmgr -c "set queue batch resources_default.nodes=1"
qmgr -c "set queue batch resources_default.walltime=3600"
qmgr -c "set server default_queue=batch"
qmgr -c "set server operators += TORQUEADMIN@SERVERNAME"
qmgr -c "set server managers += TORQUEADMIN@SERVERNAM
```

3.3.1.5 Configurando NFS en Rocks Cluster

Por defecto Rocks Cluster instala el servicio de NFS (Network File System – Sistema de Archivos de Red). Sin embargo existe el caso en que el directorio con el cual gestiona NFS deja de estar disponible después de una hora o después del arranque del sistema operativo, debido a que no necesita que se encuentre activo este protocolo. Este se encuentra bajo el control del auto-montador "**autofs**", por lo cual solo monta los directorios únicamente cuando son necesarios, de manera que para mantenerlo activo o montado indefinidamente será necesario editar la siguiente línea dentro del archivo "auto.master".

Editando "auto.master"

```
# vi /etc/auto.master
```

Dentro del archivo se agregará lo siguiente "**--ghost**" y la línea quedará de la siguiente manera:

```
/share /etc/auto.share --ghost  --timeout=1200
```

Para que esto tenga efecto habrá que recargar el servicio de auto-montaje.

Reiniciando servicio de auto-montaje
service autofs restart # rocks sync config # rocks run host compute 'service autofs restart'

En el caso de montar el directorio para la sesión del usuario bastará con ingresar el siguiente comando.

Montaje temporal
• # ls –lFd /share/ : montará todo el directorio y su contenido. • # ls –lFd /share/{ruta-requerida}/ : montará únicamente la ruta y contenido requerido.

3.3.2 Instalación y configuración de nodos secundarios

3.3.2.1 Agregando Nodos

Dentro de la instalación y configuración de los nodos secundarios, prácticamente existe muy poco que hacer, ya que desde el Frontend será desde donde se administre los trabajos a procesar.

Para la instalación de los nodos secundarios se tienen dos maneras. La primera es mediante PXE (instalación mediante red), para éste se necesitará configurar en el BIOS, que el orden de inicio sea por medio de red –si en caso es soportado-; La segunda forma, descargando el CD "Kernel/Boot Roll" para que éste al momento de iniciar el computador, detecte el CD y empiece la instalación base del rol antes mencionado. En ambos casos se deberá tener previamente ya conectados los equipos a la red del Cluster para que los roles del Frontend sean desplegados o instalados.

Para empezar a agregar los nodos, se deberá ejecutar el siguiente comando desde un terminal en el equipo Frontend, para poder desplegar la detección de los nodos y poder realizar las instalaciones y configuraciones de roles en los nodos.

Detectando y agregando nodos de la red del Cluster para su instalación
insert-ethers

A continuación se desplegará una pantalla con varias opciones en la que se seleccionará “Compute”, la cual indicará que en la estructura del sistema se agregarán computadores como nodos. Durante este proceso el Frontend se encargará de detectar todos los nodos conectados a éste. El mismo iniciará con la instalación correspondiente sobre cada nodo encontrado y almacenará la información dentro de la base de datos, y en el archivo /etc/dhcpd.conf, la estructura de los nodos encontrados con sus direcciones IP asignadas por el Frontend. Por defecto, el Frontend utilizará la primera tarjeta de red con **IP: 10.1.1.1** para las asignaciones de IP privadas para el sistema Cluster; y la segunda tarjeta de red para el acceso a Internet o público con **IP: 190.15.140.49**. Estas dos previamente asignadas durante la instalación del Frontend.

Frontend	
Selección del tipo de dispositivo	***Detectando y agregando nodos***

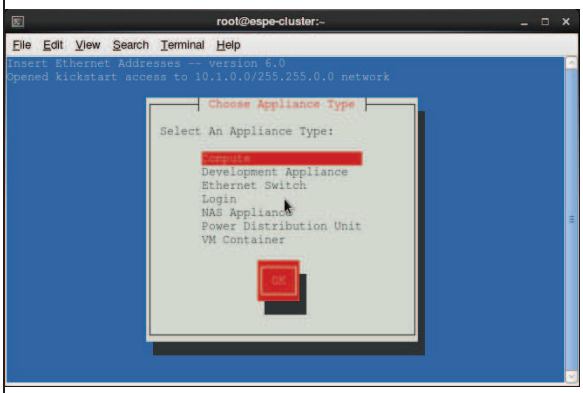
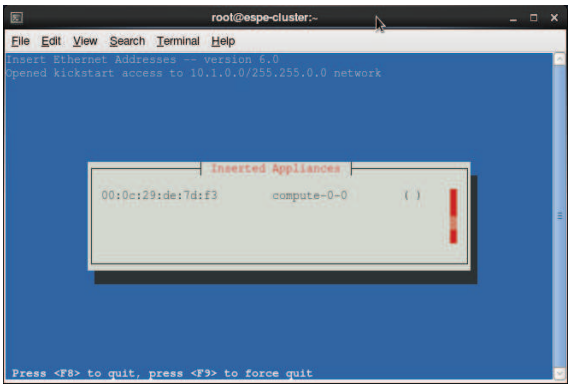

	

Imagen 3-1	*Imagen 3-2*

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

Después de este proceso, en el lado del Frontend, se puede acceder al nodo o nodos para revisar la instalación mediante una consola remota gráfica

utilizando VNC que viene incorporada junto con Rocks Cluster, pues es justamente una herramienta para monitoreo en el proceso de instalación.

Hay que señalar que al momento que el sistema empieza a realizar la detección de los nodos éste los marcará, al nodo o nodos encontrados, con los siguientes parámetros de identificación: dirección MAC, nombre del nodo "compute-y-x" - dónde: "y*"* y "*x*" son los identificadores de posición de los nodos en el *rack* y en la cabina que se encuentran., Así se lo identifica dentro de Rocks Cluster. El valor de "x" se irá incrementando de acuerdo al número de nodos que se vaya agregando al Cluster, empezando para ambos la numeración en cero. Por defecto Rocks Cluster almacena los nodos dentro de una cabina denominada cabinet que viene a ser el valor de "y". Ambos son configurables y por defecto el primer nodo se lo verá así: ***compute-0-0***, mientras que "compute" indica el tipo de dispositivo o appliance sobre el cuál se está trabajando, se marca "()",al final de esa forma si en el nodo se logró realizar la instalación satisfactoriamente o marcará de la siguiente manera "(*)" si no fue satisfactoria la instalación. El siguiente comando permite monitorear los nodos a los que se ha aplicado la instalación.

Nodos	
Instrucción de monitoreo para la instalación	***Instalación sobre los nodos***
# rocks-console compute-x-y	

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

Una vez terminado el proceso de detección e instalación de los nodos estos se reiniciarán de manera automática para acceder y conectarse al nodo principal. Desde el lado del Frontend se presionará la tecla "**F8**" para salir de la consola o "F9" para forzar la salida.

Para verificar adicionalmente que el nodo fue instalado, se puede realizar un *ping*[8] desde el equipo Frontend, ingresando la siguiente sentencia con los parámetros identificados en la detección de la instalación.

Sentencia para verificar comunicación desde el Frontend al nodo	
# ping compute-y-x	**Ej:** # ping compute-0-0

El tiempo requerido para la instalación de cada nodo es alrededor de 20 a 30 minutos, todo dependerá de los recursos de hardware que los equipos posean.

En el caso de que un nodo haya sido eliminado y se desee que uno nuevo ocupe su lugar se tiene que ingresar el siguiente comando:

Insertando un nuevo nodo en la posición de un nodo eliminado	
# insert-ethers –rank=X # rocks sync config	**Ej: #** insert-ethers --rank=10 # rocks sync config

En el ejemplo se indica que se agregará el nuevo nodo en la posición 10.

La sentencia, "**rocks sync config**", actualizará la base de datos y dará a conocer dentro del sistema que ese nodo ya no se encuentra disponible.

3.3.2.2 Eliminando Nodos

El eliminar un nodo puede darse a una única razón, mal funcionamiento a nivel de hardware. Es por ello que al momento de eliminar un nodo físico del sistema Cluster, puede ser necesario eliminar ese nodo de la base de datos del sistema. Para ello hay que ejecutar la siguiente instrucción para que otro nodo pueda ocupar su lugar.

[8]**PING**: el acrónimo de Packet Internet Groper, el que puede significar "Buscador o rastreador de paquetes en redes"***Referencia:****http://es.wikipedia.org/wiki/Ping*

Eliminación de un nodo
rocks remove host compute-Y-X # rocks sync config

Si se desea eliminar varios nodos dentro del sistema Cluster, los siguientes comandos deberán ser ingresados.

Eliminación de varios nodos	
for x in 'seq N1 1 Nn': Dónde N1 representa al nodo 1 y así sucesivamente. do rocks remove host compute-Y-**$**X rocks sync config done • Nótese el símbolo de dólar "**$**", éste indica la variable a la que va afectar, es decir al conjunto de nodos que se eliminarán dentro del bucle, e "**Y**" el número de cabina a la que afectará.	**Ej:** for x in 'seq 10 1 11' do rocks remove host compute-0-**$**X rocks sync config done

El ejemplo indica que los nodos del 10 al 11 serán eliminados y a su vez actualizados en la base del sistema.

Nota: Cabe recalcar que cada vez que se haga alguna edición o cambio dentro del sistema Cluster se tendrá que ingresar la sentencia de actualización de configuración del sistema, "**rocks sync config**".

3.4 Prueba de funcionamiento de los nodos

Para las pruebas de funcionamiento de los nodos se crearán diversos archivos que permitirán y darán a conocer que el sistema Cluster está en funcionamiento. Son pruebas de comunicación desde el Frontend a los nodos hijos y viceversa, de los cuales se comprobará su respuesta. Para ello previamente se han instalado y configurado varios parámetros en el sistema como anteriormente se lo ha realizado.

Para estas pruebas de funcionamiento tanto **NFS**, **TORQUE** y **MAUI** están presentes para poder gestionar y administrar las tareas que se asignarán a los nodos hijos desde el Frontend y éstos devuelvan una respuesta a lo solicitado por el usuario. Es indispensable conocer el número de nodos y procesadores para poder ejecutar un programa en el sistema Cluster, también saber cómo se encuentran nombrados o identificados los nodos "hostname". De esta manera se puede ejecutar cierto programa en algún nodo específico. Es así que para ejecutar mediante el componente MPICH especificando el o los nodos, será necesario crear un archivo el cual contenga la lista de nombres de cada nodo del sistema Cluster y se los debe listar en un host por línea como en el ejemplo a continuación:

Archivo que contiene la lista de nodos	**Nombres de nodos**
 nodos.txt	compute-0-0 compute-0-1 compute-0-2 …

A continuación se muestran los comandos y parámetros de mayor importancia para ejecutar las tareas o programas en el Cluster.

COMANDOS (MPI)	
#mpicc /ruta/archOriginal.c**–o** /ruta/archEjecutable.extension	**Comando:** **mpicc** - Permite compilar un programa. El archivo de salida puede tener el mismo nombre del archivo origen y su extensión puede ser cualquiera, pero se recomienda identificarlo como un ejecutable o salida del programa (**.exe** / **.o** / **.out**). Puede definirse ciertos parámetros según los requiera. **Parámetro(s) - Opcional:** [**-o**] :parámetro de salida
# mpiexec [**–np**] [**#procesadores**] [**-machinefile**] [**nombre_arch_con_lista_nodos**] /ruta/archEjecutable.extension	**Comando:** **mpiexec** – Permite ejecutar la aplicación. Pueden definirse ciertos parámetros según los requiera. **Parámetro(s) - Opcional:** [**-np**] [**#procesadores**]: Indicador de **#** procesadores. [**-machinefile**] [**nombre_arch_con_lista_nodos**]: Incluye al archivo con la lista de nodos a los que se aplicará la ejecución del programa.
# mpirun [**–np**] [**#procesadores**] [**-machinefile**] [**nombre_arch_con_lista_nodos**] /ruta/archEjecutable.extension	Misma característica al anterior.

Nota: Excluir los paréntesis "**[]**", son solo de referencia para aplicar los parámetros.

COMANDOS (TORQUE)	
#qsub [**-q**] [nodo_destino] /ruta/script.extension **# qsub** [**-I**] /ruta/script.extension	**Comando:** **qsub**: Permite crear un trabajo (job) con un script ejecutable (shell script). Este se ejecutará por defecto en todo el Cluster o buscará un nodo libre para realizar la tarea, si es que no se especifica el parámetro [**–q**] para indicar el destino, aunque existen una serie de parámetros adicionales. **Parámetro(s) - Opcional:** [**-q**] [**nodo_destino**]: Especifica el destino donde ejecutará la tarea. [**-I**]: Realiza un trabajo interactivo según los parámetros especificados dentro del script, para evitar una gestión manual sino automática (véase: ***Batch System – Sistema por Lotes***).
# pbsnodes [**-a**] **# pbsnodes** [**-x**] **# pbsnodes** [**-l**] [**estado**]	**Comando:** **pbsnodes**: Permite listar los nodos y todos sus atributos. **Parámetro(s) - Opcional:** [**-a**]: Lista todos los atributos de los nodos. [**-x**]: Lista todos los atributos de los nodos pero en formato XML. [**-l**] [**estado**]: Lista los nodos con el estado solicitado como: “free”, “busy”, “offline”.
# qstat [**-f**]	**Comando:** **qstat**: Muestra el estado de los trabajos por lotes PBS. **Parámetro(s) - Opcional:** [**-f**]: Muestra el estado del trabajo con la cantidad de tiempo restante para cumplir la tarea (walltime).
# showstart [**id_job**]	**Comando:** **showstart**: Muestra el trabajo y la cantidad de procesadores requeridos o asignados a una tarea incluyendo el tiempo. **Parámetro(s) - Opcional:** [**id_job**]: especifica el identificado de trabajo que se desea mostrar.

→Continúa

# showq [**-r**] **# showq** [**-u**] [**usuario**]	**Comando:** **showq**: lista las tareas en cola. **Parámetro(s) - Opcional:** [**-r**]: Muestra únicamente las tareas que están corriendo. [**-u**] [**usuario**]: Muestra únicamente las tareas ejecutadas por un usuario específico.
# showstate	**Comando:** **showstate**: Muestra el estado de los recursos actuales.
Manualde Referencia 1:*http://docs.adaptivecomputing.com/torque/4-1-4/help.htm#topics/0-intro/welcome.htm* ***Manualde Referencia 2:*** *http://docs.adaptivecomputing.com/maui/mauiadmin.php*	

Los siguientes ejemplos demuestran el funcionamiento y la comunicaciónde todo el Cluster, así también, las diferentes formas y parámetros con los que se pueden ejecutar las aplicaciones utilizando TORQUE o con la biblioteca MPI como: MPICH. En el caso de utilizar este último, será necesario ejecutar la tarea o la aplicación mediante *mpiexec* (estable) o mpirun anteriormente descrito.

Archivos con los que se realizarán las pruebas de comunicación y funcionamiento del sistema Cluster	
•  mpi-verify.c	**Ruta:** # /opt/mpi-tests/src **Descripción:** Realiza la comunicación con los nodos, preguntando por 1 procesador.
•  testMPI.c	**Ruta:** # /opt/mpi-tests/src **Descripción:** Cada procesador requerido se identifica y escribe la respuesta en el siguiente archivo: /share/apps/outTestMPI.txt – antes de nada se deberá crear el archivo para que lo lea y escriba.
•  appSumaResta.sh •  suma_resta.sh	**Ruta:** # /share/apps/appSumaResta/ **Descripción:** Compuesto de dos scripts, el principal “appSumaResta.sh” implementa ciertos parámetros para que permita un subprograma “suma_resta.sh” se ejecute en los nodos para resolver una sencilla operación de suma y resta, generando un archivo con su resultado final.
•  runscript.sh	**Ruta**: # /var/www/html/roll-documentation/torque/6.0/runscript.sh **Descripción:** Script de ejemplo el cual realiza un trabajo con parámetros prestablecidos.
•  cpi.c	**Ruta:** # /opt/mpich2/gnu/share/examples_graphics **Descripción:** Cálculo de PI.
Fuente: ➢ *www.rocksclusters.org* ➢ *http://www.eead.csic.es/compbio/material/programacion_rocks/* ➢ *http://www.sdsc.edu*	

mpi-verify.c

```
#include <stdio.h>
#include "mpi.h"

int
main(int argc, char *argv[])
{
    int     numprocs;
    int     myid;
    int     namelen;
    char    processor_name[MPI_MAX_PROCESSOR_NAME];

    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD,&myid);
    MPI_Get_processor_name(processor_name,&namelen);

    fprintf(stderr,"Process %d on %s\n", myid, processor_name);

    MPI_Barrier(MPI_COMM_WORLD);

    MPI_Finalize();
}
```

Salida de mpi-verify.c

```
[root@espe-cluster apps]# mpiexec mpi-verify.exe
Process 0 on espe-cluster.espe.edu.ec
[root@espe-cluster apps]#
```

testMPI.c

```
#include <stdio.h>
#include <stdlib.h>
#include <mpi.h>

/* Programa 'hola mundo' donde cada procesador requerido se identifica,
basado en ejemplos orginales de Tim Kaiser (http://www.sdsc.edu/~tkaiser),
del San Diego Supercomputer Center, en California */

int main(int argc, char **argv)
{
   int myid, numprocs;
   FILE *arch;

MPI_Init(&argc,&argv);
   MPI_Comm_size( MPI_COMM_WORLD, &numprocs ); // devuelve el número de procesos en este COMM_WORLD
MPI_Comm_rank( MPI_COMM_WORLD, &myid );  // identificate por el myid asignado

printf("Soy el procesador %d de un total de %d\n",myid,numprocs);

   if(myid == 1) // solamente para el procesador 1
   {
     // ajusta la ruta del archivo si es necesario
arch=fopen("/share/apps/outTestMPI.txt","w");
fprintf(arch,"Hola, soy el procesador 1\n");
fclose(arch);
   }

   MPI_Finalize();

   return 0;
}
```

Salida de testMPI.c

```
[root@espe-cluster apps]# mpiexec testMPI.exe
Soy el procesador 0 de un total de 1
entra[root@espe-cluster apps]#
```

appSumaResta.sh / suma_resta.sh

Archivo 1)

```
#!/bin/csh
#PBS -l nodes=1:ppn=1
echo The nodefile for this job is stored at `echo $PBS_NODEFILE`
echo `cat $PBS_NODEFILE`
#PBS -e myprog.err
#PBS -o myprog.out
cd /share/apps/
sh suma_resta.sh  //llamado a la apliación de suma_resta.sh
```

Archivo 2)

```
#!/bin/bash

let A=100
let B=50

#
# Funcion suma()
# Suma las variables A y B
#
function suma(){
 let C=$A+$B
echo "Suma: $C"
}

#
# Funcion resta()
# Resta los variables A y B
#
function resta(){
 let C=$A-$B
echo "Resta: $C"
}

suma
resta

echo -e $(suma)"\n"$(resta) > /share/apps/salida.txt
```

Salida de appSumaResta.sh / suma_resta.sh

Se genera un archivo con los resultados de suma y resta, con los valores correspondientes a 1500 y 500 respectivamente.

resultado.txt

runscript.sh

```
#!/bin/sh
# A job script example for the torque-roll for Rocks
#
# Each directive MUST start with #PBS
#PBS -lwalltime=1:00:00      # One hour runtime
#PBS -lnodes=2:ppn=2         # 2 nodes with 2 cpus each
#PBS -lpmem=1gb              # 1 GB memory per cpu
#PBS -Nmpi-verify-openmpi    # the name of the job

cd $PBS_O_WORKDIR                # cd to the directory the job was submitted from
                                 # the job is initiated in the users home dir on the compute node

. /opt/torque/etc/openmpi-setup.sh  # set the enviroment vars needed to make OpenMPI work with torque

                                 # We have pre-made an application in the submit dir:
                                 # cp /opt/mpi-tests/src/mpi-verify.c .
                                 # mpicc mpi-verify.c -o mpi-verify.openmpi.x

mpirun ./mpi-verify.openmpi.x

qstat -f $PBS_JOBID | grep -i resource   # List resource usage in the job stdout report

                                 # when the script exits the job is done
```

Salida de script.sh

```
runscript.sh: line 13: ./opt/torque/etc/openmpi-setup.sh: No such file or direct
ory
--------------------------------------------------------------------------
mpirun was unable to launch the specified application as it could not access
or execute an executable:

Executable: ./mpi-verify.exe
Node: espe-cluster.espe.edu.ec

while attempting to start process rank 0.
--------------------------------------------------------------------------
    Resource_List.neednodes = 1:ppn=1
    Resource_List.nodect = 1
    Resource_List.nodes = 1:ppn=1
    Resource_List.walltime = 01:00:00
    Resource_List.neednodes = 1:ppn=1
    Resource_List.nodect = 1
    Resource_List.nodes = 1:ppn=1
    Resource_List.walltime = 01:00:00
```

cpi.c

```
#include "mpi.h" /*Librerías empleadas */
#include<stdio.h>
#include<math.h>

double f(double); /*declaración de variables */


   double f(double a) {
      return (4.0 / (1.0 + a * a)); /* valor de retorno de la operación efectuada */
   }

   int main(int argc, char
      *argv[]) /*metodo con parametros de entrada */
{
int n, myid, numprocs, i; /*variables */
      double PI25DT = 3.14159265358979323846264 3;
      double mypi, pi, h, sum, x;
      double startwtime = 0.0, endwtime;
      int namelen;
      char processor_name[
MPI_MAX_PROCESSOR_NAME
];
MPI_Init( & argc,  & argv); /* va a inicializar el programa */
      MPI_Comm_size(MPI_COMM_WORLD,  & numprocs); /* indica el número de
procesos que está arrancando o en cuántos procesos se divide.*/
      MPI_Comm_rank(MPI_COMM_WORLD,  & myid); /* indica en donde se corre,
por
ejemplo el num de salón.*/
MPI_Get_processor_name(processor_name,  & namelen); /*indica los procesos
desde
donde se esta corriendo el programa */
      fprintf(stdout, "Process %d of %d is on %s\n", /* indica en que maquina se
esta
            corriendo */
           myid, numprocs, processor_name);
fflush(stdout);
      n = 10000; /* default # of rectangles */
if (myid == 0) {
         startwtime = MPI_Wtime(); /* MPI_Wtime es para tomar el tiempo y saber
cuanto tiempo se va tardar */
}
      MPI_Bcast( & n, 1, MPI_INT, 0, MPI_COMM_WORLD); /* MPI_Bcost es como
una difusora, el mismo msj se manda a todos los procesos.*/
h = 1.0 / (double) n;
      sum = 0.0;
      /* A slightly better approach starts from large i and works back */
      for (i = myid + 1; i <= n; i += numprocs) {
         x = h * ((double) i - 0.5);
         sum += f(x);
}
```

```
            mypi = h * sum;
            MPI_Reduce( & mypi,     & pi, 1, MPI_DOUBLE, MPI_SUM, 0,
MPI_COMM_WORLD); /* MPI_Reduce recoge los datos a cada proceso(resultados).*/
/*MPI_SUM suma todos los resultados recaudados en el MPI_Reduce y los guarda en
una variable */ if (myid == 0) {
      endwtime = MPI_Wtime();
               printf("pi is approximately %.16f, Error is %.16f\n", pi, fabs(pi - PI25DT));
               printf("wall clock time = %f\n", endwtime - startwtime);
      fflush(stdout);
            }
            MPI_Finalize(); /* indica donde se termina. */ return 0;
          }
```

Salida de cpi.c

```
[root@espe-cluster cpi]# mpiexec -machinefile nodos.lista cpi.exe
Process 2 on espe-cluster.espe.edu.ec
Process 0 on compute-0-0.local
Process 1 on compute-0-1.local
pi is approximately 3.1416009869231249, Error is 0.0000083333333318
wall clock time = 0.004498
[root@espe-cluster cpi]#
```

El resultado presentado en el cuadro anterior, muestra el trabajo del Cluster sin definir un número de procesos, haciendo que el trabajo lo realice cada equipo como un solo proceso por lo que el tiempo en llevar a cabo esa tarea es mínimo. Ahora en el caso de definir un número de procesos, con la idea de minimizar errores de cálculo, conllevará mayor tiempo de respuesta, es así que, para las dos siguientes tareas se ha de definir el número de proceso en 300, para el primer caso en un único nodo (compute-0-0) se hará la ejecución, mientras que para el segundo se realizará para todo el Cluster.

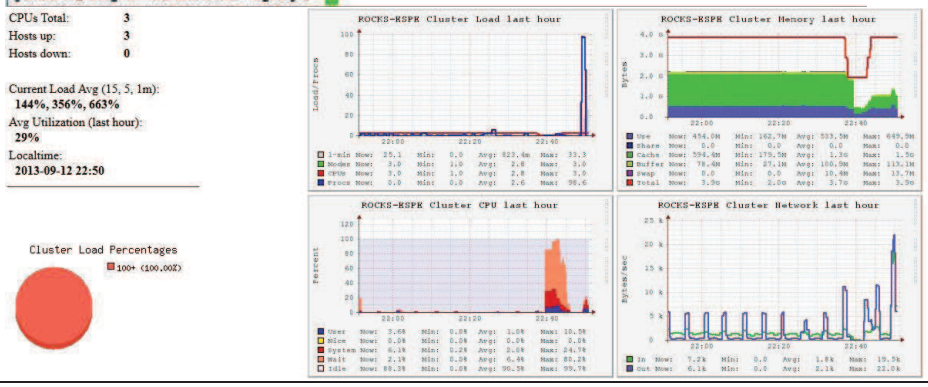
Para revisar el proceso, la carga del nodo y del Cluster es proporcionada por Ganglia.

Proceso y carga del nodo – compute-0-0

```
Process 299 on compute-0-0.local
Process 48 on compute-0-0.local
Process 76 on compute-0-0.local
Process 267 on compute-0-0.local
Process 285 on compute-0-0.local
Process 147 on compute-0-0.local
Process 269 on compute-0-0.local
Process 138 on compute-0-0.local
Process 134 on compute-0-0.local
Process 67 on compute-0-0.local
Process 297 on compute-0-0.local
Process 257 on compute-0-0.local
Process 136 on compute-0-0.local
pi is approximately 3.1416009869231249, Error is 0.0000083333333318
wall clock time = 172.923755
[root@espe-cluster cpi]#
```

Proceso y carga de todo el Cluster

```
Process 297 on compute-0-0.local
Process 292 on compute-0-1.local
pi is approximately 3.1416009869231249, Error is 0.0000083333333318
wall clock time = 1.326471
Process 295 on compute-0-1.local
Process 298 on compute-0-1.local
[root@espe-cluster cpi]#
```



Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

Otro sistema que se adaptó a Rocks Clusters y se generó un rol como tal, es APBS Roll. APBS es un software para la evaluación de las propiedades electrostáticas de sistemas biomoleculares nanoescala o por sus siglas en Ingles Adaptable Poisson-Boltzmann Solver (APBS). Este software tiene desarrollo para utilizar en diferentes sistemas operativos como: Linux, Windows, MacOS y también poder utilizar en Clusters. “APBS es un paquete de

software para el modelado biomolecular solvatación a través de la solución de la ecuación de Poisson-Boltzmann (PBE), uno de los modelos continuos más populares para describir las interacciones electrostáticas entre solutos moleculares en medios salinos y acuosos".

El siguiente ejemplo muestra la información procesada y generada por el software, para ello se utilizó el rol de la versión 5.4 de Rocks Clusters, pero para el pleno funcionamiento del rol se han realizado ciertas modificaciones e instalaciones adicionales; la versión del Cluster, es la 6.0 (Mamba) y existen varios cambios en su estructura.

Se maneja el mismo esquema de instalación y configuración para los roles.

Paso 1) Proceso para agregar el rol – APBS
rocks add roll apbs*.iso # rocks enable roll apbs # cd /export/rocks/install # rocks create distro # rocks run roll apbs > /tmp/install-apbs.sh # sh -x /tmp/install-apbs.sh

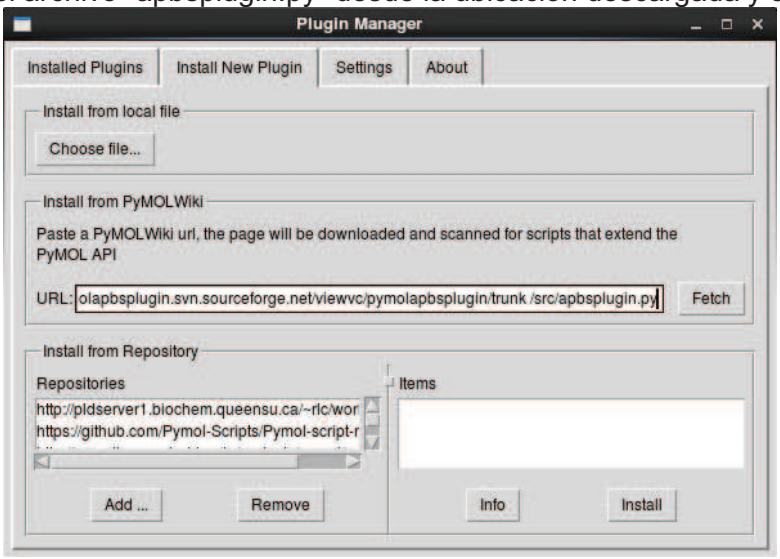
Paso 2) Reconstruyendo los nodos después de instalación del rol y reiniciado el Frontend
rocks run host compute command=/boot/kickstart/cluster-kickstart

Paso 3) Instalación y configuración adicional – PyMOL
Instalación de módulo PyMOL para gráficas de APBS # yum – y install pymol-wxpython pymol *Descargar el plugin en PyMOL de APBS.* # wget http://pymolapbsplugin.svn.sourceforge.net/viewvc/pymolapbsplugin/trunk /src/apbsplugin.py

→Continúa

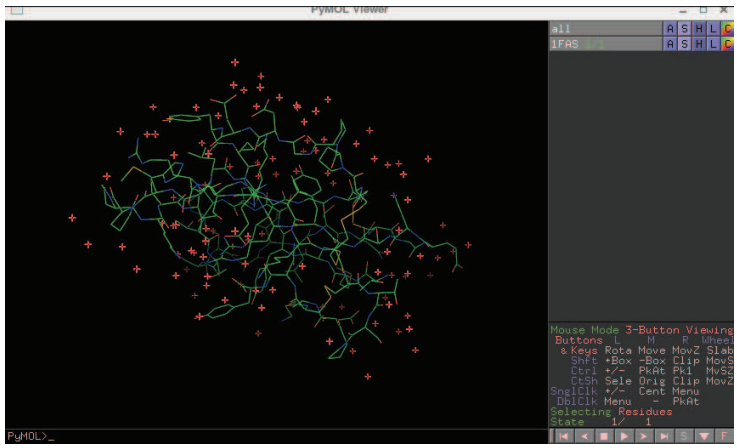
Paso 3.1) Instalación y configuración adicional – Plugin PyMOL

Se escoge el archivo “apbsplugin.py” desde la ubicación descargada y se instala.



Finalmente se puede descargar un ejemplo, ejecutar y visualizar el resultado en una grafica.

- *Descarga (1FAS.pdb): http://www.pdb.org/pdb/explore.do?structureId=1FAS*
- Abrir el archivo.



Además se puede montar APBS para que esté visible y se pueda utilizar a través del Internet.

Para ello se debe crear un acceso de la carpeta APBS de la ruta /opt/apbs a /var/www/html/ y luego cambiar la configuración de dos archivos: “index.html”, “aconf.py” y “querystatus.cgi”.

Paso 1) Cambios en algunos archivos de software APBS

3. Editar archivo: "**index.html**"
   Ruta: /var/www/html/apbs/pdb2pqr/
   Línea 35 : http://190.15.140.49/apbs/pdb2pqr/pdb2pqr.css
   Línea 141: http://190.15.140.49/apbs/pdb2pqr/pdb2pqr.cgi

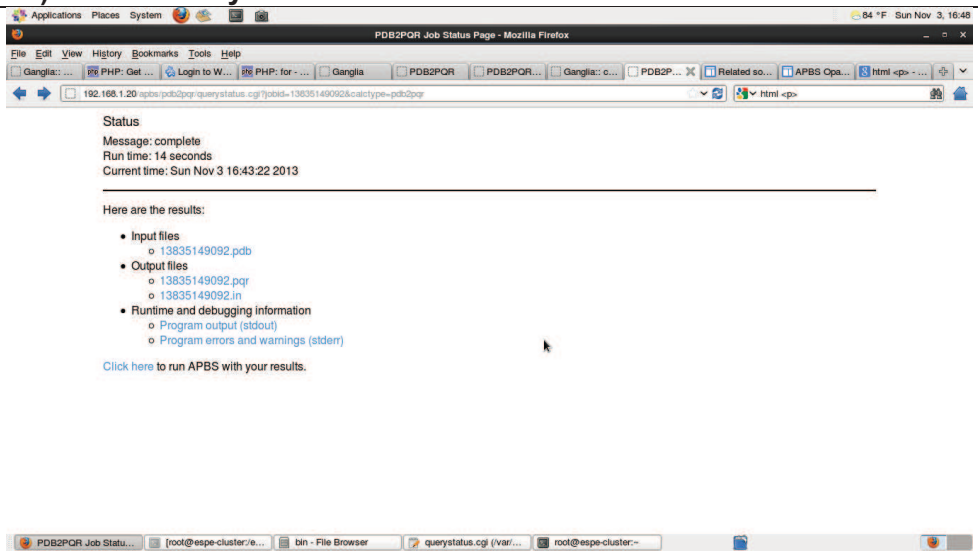
4. Editar archivo: "**aconf.py**"
   Ruta: /var/www/html/apbs/pdb2pqr/src/
   Línea 141: http://190.15.140.49/apbs/pdb2pqr/pdb2pqr.cgi

5. Editar archivo: "**querystatus.cgi**"
   Ruta: /var/www/html/apbs/pdb2pqr/
   Línea 502: print " Click here to run APBS with your results. " % nexturl

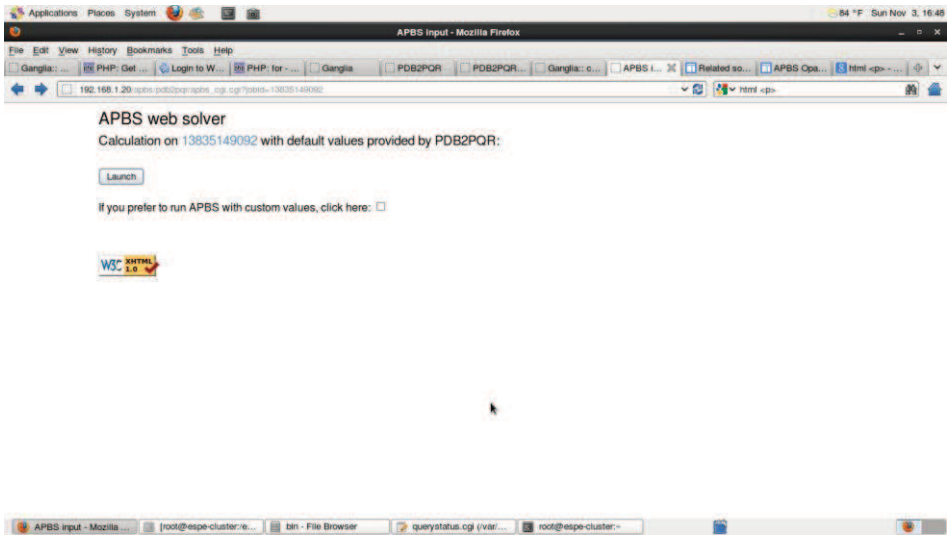
Paso2) Llenado del formulario y selección de archivo PDB

→Continúa

Paso 3) Generación y status de las salidas del archivo

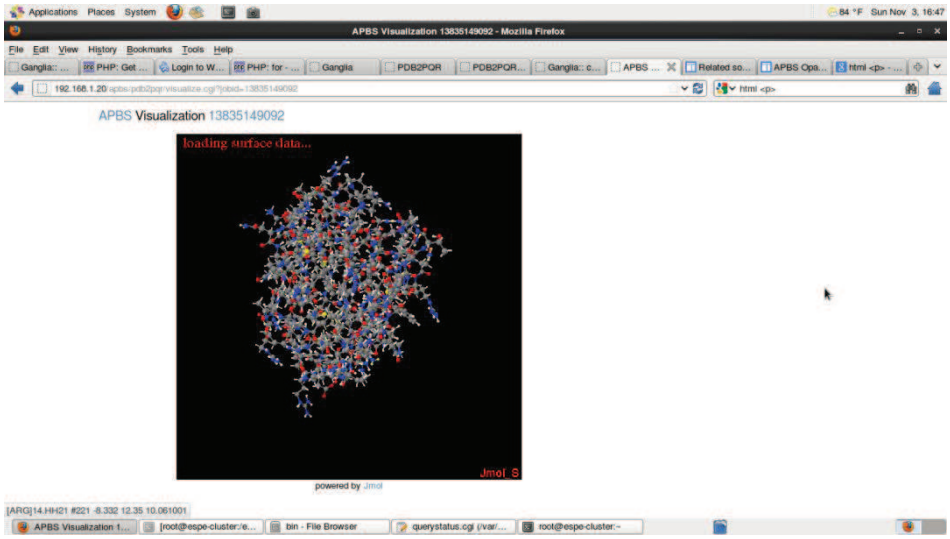


Paso 4) Status de la generación para visualizar la gráfica



→Continúa

Paso 5) Generación de la gráfica



Carga del Cluster

Fuente: Investigación
Elaborador por: Juan Daniel Carrillo

3.5 Seguridad del Cluster

A continuación se describe el uso de una herramienta para la seguridad del Cluster, pues es indispensable mantener restricciones a accesos ajenos al

Sistema que puedan malograr, dañar la información u ocasionar algo indebido, sea intencional o no.

La arquitectura habitual de un Cluster es un conjunto de máquinas en una subred dedicada. El nodo principal se conecta a esa red con el mundo externo, es decir a la red de la empresa o directamente a Internet, haciendo que el nodo principal o Frontend sea el único acceso a la subred del Cluster, ningún nodo hijo del Sistema está vinculado a otra red. Con este modelo, la seguridad normalmente recae sobre el nodo principal y la subred del Cluster es una red abierta basada en la confianza. Son varias las razones para este enfoque, pues la adición de capas de seguridad en la red del Cluster puede afectar de manera negativa al rendimiento del mismo. Al centrarse en el nodo principal, la administración de seguridad es más simple de manejar. Este enfoque también simplifica la configuración de filtrado de paquetes, por ejemplo firewall. Configurando de manera incorrecta el filtrado de paquetes, podría ocasionar problemas en el Cluster. De tal manera que con el enfoque de red aislada, se puede configurar la interfaz interna para permitir todo el tráfico y aplicar el filtro de paquetes solo a la interfaz pública en el nodo principal.

Hay ciertos puntos a ser considerados para empezar y mantener la seguridad:

- Hay que asegurar de aplicar todos los parches de seguridad adecuada, al menos en el nodo principal y preferiblemente a todos los nodos. Esta es una tarea que tendrá que hacerse de forma rutinaria, y no sólo cuando se configura el clúster.
- Saber lo que está instalado en el Sistema.
- Diferenciar entre lo que está disponible en el interior del Cluster y lo que está disponible fuera del clúster. Por ejemplo, no ejecutar NFS fuera del clúster.
- No ejecutar servicios innecesarios.

- Utilizar la cuenta de “root” - cuenta de superusuario - sólo cuando sea necesario. No ejecutar programas como “root” a menos que sea absolutamente necesario.

La seguridad es implementada para precautelar la confidencialidad, integridad y la disponibilidad del Sistema definidas como:

- “**Confidencialidad:** La información solo debe ser conocida y accedida por quienes estén debidamente autorizados, es decir la información llegue solamente a las personas autorizadas.
- **Integridad:** La información sólo puede ser modificada y/o eliminada por quienes estén autorizados para ello.
- **Disponibilidad:** Los sistemas que albergan datos e información deben garantizar que éstos podrán accederse cuando así se requiera por quienes tienen derecho a ello.”

La seguridad de Rocks Cluster basado en CentOS, éste viene por defecto con el servicio de **IPTABLES**, el cual permite crear reglas de acceso como por ejemplo bloquear o permitir puertos, direcciones IP, entre otras opciones.

Para este caso se procederá a instalar **Firewall Builder**, con una interfaz gráfica amigable y documentación, ésto hace que el administrador pueda configurar y administrar el servicio de forma sencilla.

3.5.1 Instalación y configuración de Firewall Builder (FWBUILDER)

Cuatro son los pasos a seguir para la instalación.

- Crear un archivo de repositorio.
  - vi /etc/yum.repos.d/fwbuilder.repo
- Dentro del archivo se incluirá lo siguiente:

fwbuilder.repo

```
[fwbuilder]
name=Firewall Builder
failovermethod=priority
baseurl=http://packages.fwbuilder.org/rpm/stable/rhel-$releasever-$basearch
enabled=1
```

→Continúa

```
priority=14

[fwbuilder-testing]
name=Firewall Builder Test Builds
failovermethod=priority
baseurl=http://packages.fwbuilder.org/rpm/testing/rhel-$releasever-$basearch
enabled=0
priority=14
```

- Obtener e importar la firma digital de FWBUILDER.
  - wget http://www.fwbuilder.org/PACKAGE-GPG-KEY-fwbuilder.asc
  - rpm --import PACKAGE-GPG-KEY-fwbuilder.asc
- Instalar FWBUILDER.
  - yum –y install fwbuilder

Para que las políticas creadas se carguen al iniciar el Cluster. Firewall Builder genera un script de Shell para el firewall basado en **iptables**, para activar la política en el arranque del sistema se deberá localizar el script generado por fwbuilder dentro del directorio "/etc/firewall" y editar el archivo "/etc/rc.d/rc.local" y añadir la ruta en el que se encuentra dicho script.

Ruta de script generado por Fwbuilder en archivo "rc.local"
/etc/firewall/script-generado.fw

Para el funcionamiento correcto de la política se deberá desactivar el script de iptables que viene por defecto en la distribución del sistema "CentOS".

Descativación script de iptables
chkconfig –level 2345 iptables off

En caso de que se desee reversar o volver activar, se puede editar en la línea anterior el estado "off" por "on".

Para mayor información se puede ingresar a: www.fwbuilder.org

CAPÍTULO 4

4.1 Conclusiones y Recomendaciones

4.1.1 Conclusiones

- Se diseñó, implementó y evaluó un cluster para el Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas ESPE, con el hardware y cableado estructurado disponibles optimizando costos.

- Luego de las pruebas realizadas se evidencio que el tiempo de respuesta en la solución de problemas complejos utilizando un cluster y procesamiento paralelo es mucho menor que el obtenido al utilizar un solo computador de altas características.

- La infraestructura que conforma el cluster puede ir creciendo mediante la agregación de nodos al sistema incrementándose el procesamiento, memoria y almacenamiento y comportándose para el usuario como un solo computador y no como un conjunto de equipos.

- Se realizaron pruebas con varios tipos de problemas complejos relacionados con procesamiento de imágenes, análisis numérico, biocomputo y otros, obteniéndose resultados muy satisfactorios.

- La construcción e implementación de un Cluster no conlleva incurrir en gastos económicos altos, ya que se puede disfrutar del uso de esta tecnología aprovechando los recursos de hardware o cómputo existentes.

- El Departamento de Ciencias de la Computación cuenta ahora con este tipo de tecnología y queda abierta la posibilidad de mejora y buen uso que se le dé.

4.1.2 Recomendaciones

- Dar seguimiento al desarrollo e implementación de programas en paralelo o la migración de programas que actualmente se desarrollan y que se basen en estos nuevos conceptos con el fin de aprovechar los recursos de hardware.

- Se realicen cambios en la Planificación Curricular de la Carrera de Ingeniería en Sistemas e Informática a fin de que se incluya el estudio de esta tecnología y el paradigma de la programación paralela.

- Adquirir e incrementar la bibliografía relacionada con el tema, así como la adquisición de infraestructura de alto rendimiento a fin de mejorar los resultados obtenidos en este trabajo y poner a disposición de la comunidad universitaria estos recursos.

BIBLIOGRAFÍA

Correto, J. (2001). *Sistemas Operativos .* Madrid: Mc Graw Hill .

Hesham, E.-R. &.-E. (2005). *Advanced Computer Arquitecture and Parallel Procesing.*John Wiley & Sons Inc.

Joaquin, A. A. (2010). *Sistemas Operativos Teoria y Problemas.* Madrid: Sanz y Torres, SL .

Parhami, B. (2014). *Introduction to parallel Processing.*

Wikipedia. (s.f.). Obtenido de http://es.wikipedia.org/wiki/Supercomputadora

Wikipedia. (2014). *es.wikipedia.org/wiki/Arquitectura_de_computadoras*. Obtenido de http//es.wikipedia.org

wikipedia Commos. (s.f.). Obtenido de http://commons.wikimedia.org/wiki/Main_Page