Arquitectura Clúster de Alto Rendimiento Utilizando Herramientas de Software Libre
High Performance Cluster Architecture Using Free Software Tools

Leonardo Chuquiguanca, Edyson Malla, Freddy Ajila y Rene Guamán-Quinché

***Resumen*—En este artículo se presenta los resultados obtenidos de implementar una arquitectura clúster de tipo beowulf clase II con hardware convencional y herramientas de software libre, lo que permitió convertir un laboratorio utilizado para la enseñanza de programación de sistemas informáticos, en una sala de altas prestaciones mediante el uso de software: MPICH, Mosix, Blender, Hadoop y Ganglia Monitoring System.**

El método de resolución de problemas permitió: la comprensión del problema, diseñar una estrategia de solución y lograr la implementación de la arquitectura clúster con un rendimiento similar al de un supercomputador, optimizando el tiempo de procesamiento en: simulación de procesos, reconocimiento de patrones, renderizado de imagen y video, análisis de grandes volúmenes de datos Big Data, cifrado de códigos, evaluación de algoritmos, etc.

Las pruebas permitieron el procesamiento de millones de operaciones por segundo alcanzando una eficiencia del 85.2% de la capacidad total del clúster. Este logro importante facilita también la investigación académica en otros campos donde se requiera procesar grandes volúmenes de información y obtener resultados en un corto tiempo.

***Palabras clave*—Clúster, Middleware, Mosix, MPICH, Beowulf.**

***Abstract*— In this article the results of implementing a beowulf cluster type II architecture, with conventional hardware and free software tools is presented, allowed to convert a laboratory used for teaching programming systems, in a room with high performance, through a software: MPICH, Mosix, Blender, Hadoop and Ganglia Monitoring System.**

The method using for troubleshooting allowed; understanding the problem, and to design a solution strategy and achieve implementation of the cluster architecture with a performance like a supercomputer by optimizing the processing time: process simulation, pattern recognition, image and video rendering, and an analysis of Big Data, encryption codes, evaluation algorithms, etc.

The tests allowed the processing of millions of operations per second reaching an efficiency of 85.2% of total capacity of cluster. This important achievement expedite also the academy investigation in others realms where required to process large volumes of information and get results in a short time.

***Index Terms*— Cluster, Middleware, Mosix, MPICH, Beowulf.**

Este artículo fue enviado para revisión el 11 de febrero de 2015.
La implementación de la arquitectura clúster de alto rendimiento utilizando herramientas de software libre fue apoyada por la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja, como proyecto de grado.

L. Chuquiguanca graduado de la carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja actualmente administrador de sistemas en Protelcotelsa SA. Quito-Ecuador (e-mail: lrchuquihuancav@unl.edu.ec)

E. Malla graduado de la carrera de ingeniería en Sistemas de la Universidad Nacional de Loja Actualmente Ingeniero en Sistemas del Gobierno Autónomo Descentralizado Municipal San Felipe de Oña, Azuay − Ecuador (e-mail: ejmallab@unl.edu.ec)

F. Ajila docente Investigador de la Escuela de Ingeniería Industrial, Facultad de Mecánica de la Escuela Superior Politécnica de Chimborazo (e-mail: freddy.ajila@espoch.edu.ec)

R. Guaman-Quinche docente Investigador de la Carrera de Ingeniería en Sistemas Informáticos de la Universidad Técnica de Manabí (e-mail: eguaman@utm.edu.ec)

I. INTRODUCCIÓN

MULTITUD de aplicaciones dentro de la investigación científica requieren de una gran demanda de potencia de cómputo que solo pueden ser cubiertas por supercomputadores, que por su alto costo y los presupuestos reducidos para investigación para investigaciones académicas y científicas no pueden ser adquiridas por Instituciones de Educación Superior y centros de investigación, las inversiones en computación deben resultar rentables en cuanto a escalabilidad y rendimiento.

La informática para investigaciones, denominada computación de alto rendimiento usa potentes herramientas y procesos de computación para generar datos en investigaciones académicas avanzadas. Con un clúster de computación de alto rendimiento, los centros de investigación pueden obtener la velocidad y potencia de una costosa supercomputadora a una fracción del costo y con menos riesgo de sufrir tiempos de inactividad prolongados.

Una de las soluciones más fiables comparadas a la adquisición de supercomputadores es la implementación de un clúster de alto rendimiento, formado por hardware convencional y herramientas de software libre que unidos a una red de alta velocidad [1], [2], [3] ofrecen ventajas significativas en términos económicos y de escalabilidad comparadas al utilizar un único ordenador convencional, proporcionando así

resultados con un margen de error mínimo en tiempos relativamente bajos [4], [5].

Existen algunas implementaciones de clúster en centros de investigación y en instituciones de educación superior, uno de ellos se lo diseñó para la ejecución de un modelo de predicción climática, (Grupo de Ciencias de la Tierra y del Ambiente de la Dirección de Investigación (DIUC) de la Universidad de Cuenca, Ecuador. El objetivo de la implementación del clúster dentro del Grupo CTA, es fomentar a la investigación científica en áreas que requieren alta capacidad computacional, optimizando la utilización de los recursos de hardware disponibles, para contar con una herramienta potente y capaz de solventar, en lo posible esas necesidades. Se tiene como base el sistema operativo, GNU/Linux; el sistema de administración del clúster y el estándar utilizado en este proyecto para el desarrollo de software es PVM y MPI como sistemas de programación paralela, el middleware Mosix contiene lo necesario para ejecutar una aplicación que permitirán enlazar el entorno de programación paralela de la capa superior con compiladores paralelos MPI, PVM, GNU, JAVA, etc; software y herramientas de administración, que facilitan la gestión de archivos, balanceo de nodos, la imagen del sistema operativo, que ofrece a los usuarios el acceso unificado a los recursos del sistema [6].

Otro caso de éxito se basó en el diseño e implementación de un clúster de cómputo de alto rendimiento en el Centro de Investigación en matemáticas de la universidad de Guanajuato, México. El clúster de alto rendimiento denominado "El Insurgente" fue diseñado y construido específicamente para aplicaciones de cómputo científico que requieren grandes volúmenes de datos en memoria RAM, programación distribuida con un híbrido de MPI y OpenMPI, usando redes de bajo costo. De manera lógica, cada nodo del clúster tiene una parte de hardware y otra de software. El hardware está compuesto por procesadores, memoria, interfaz de red y discos duros entre otros. En cuanto al software, el nivel bajo corresponde al sistema operativo, el medio consiste en las librerías de paralelización y el alto está representado por la aplicación que se desea ejecutar en el clúster [7].

La implementación de la arquitectura clúster se realizó en un laboratorio de la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja, formado por 15 computadores personales con características homogéneas; las principales configuraciones que se realizaron son: instalación y configuración del sistema operativo Debian "wheezy", configuración e instalación del middleware Mosix [8], [9]en el kernel de Linux para obtener una imagen única de sistema orientada a la computación distribuida, para la programación paralela se utilizó MPICH [10], como una de las aplicaciones más implementadas del estándar Message Passing Interface (MPI). Para el proceso de renderizado se usó el software Blender [11]; para el análisis de datos (Big Data) [12], se implementó el framework Apache Hadoop [13]. Además se configuró el acceso remoto utilizando claves públicas a través del protocolo Secure Shell (SSH), para la configuración y compartición del sistema de ficheros en red se utilizó el protocolo Network File System (NFS) que trabaja a nivel de capa de aplicación según el modelo OSI, se realizó el diseño de la topología de red y la configuración del sistema de monitorización con Ganglia Monitoring System [14]. Todo ello permitió mejorar el tiempo de cómputo para calcular algoritmos o procesamiento de información.

La estructura del artículo es la siguiente:

La Sección II presenta como Marco Teórico, material bibliográfico sintetizado referente a arquitecturas clúster y la computación de alto rendimiento. La Sección III detalla la metodología de resolución de problemas utilizada para el desarrollo del proyecto. La Sección IV describe el proceso de implementación de la arquitectura clúster. En la Sección V se presentan los resultados obtenidos en diversas pruebas utilizando herramientas de software libre. Finalmente, la Sección VI se establece las conclusiones a las que se ha llegado al término del proyecto.

II. MARCO TEÓRICO

A. Arquitectura clúster de alto rendimiento

Un clúster es un conjunto de nodos de bajo costo conectados entre sí a través de una red de comunicaciones de alta velocidad, que operan bajo software que actúa como un sistema único de administración, responsable de distribuir las cargas de trabajo entre los nodos, de forma automática y transparente al usuario como si se tratara de un único ordenador [3], [4], [5].

De manera lógica, cada nodo del clúster está formado por hardware y software. El hardware está compuesto por las partes de un ordenador convencional. En cuanto al software, el nivel bajo corresponde al sistema operativo, el nivel medio consiste en las librerías de paralelización y el nivel alto está representado por la aplicación que se desea ejecutar; una aplicación se ejecuta en el nodo maestro, el sistema operativo y las librerías de paralelización se encargan de ejecutar copias de este programa en los nodos esclavos del clúster [8], [9], [10]. De las distintas arquitecturas disponibles, se ha elegido la arquitectura clúster de tipo beowulf, como se ilustra en la Fig. 1 la misma que está compuesta por hardware convencional y herramientas de software libre.

Un clúster de alto rendimiento es utilizado principalmente con fines académico-científicos, su objetivo principal es proporcionar altas prestaciones de capacidad de cómputo superior a los que pudiera ofrecer un ordenador común. Este tipo de arquitecturas son una alternativa a la utilización de grandes y costosas supercomputadoras [11], [12]

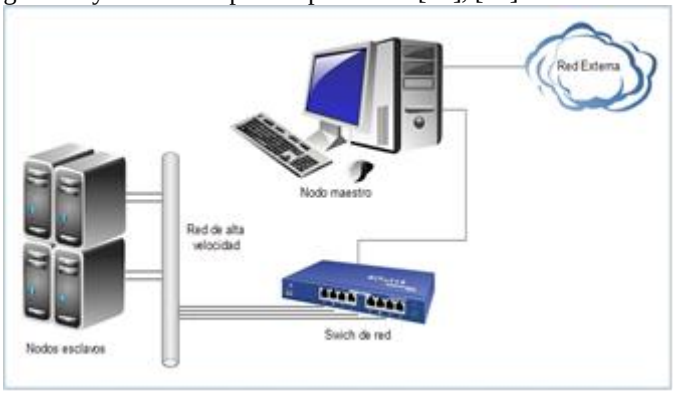

Fig 1. Clúster de computadoras formado por hardware convencional

B. Clasificación de los clústers

Existen dos tipos de clúster según la arquitectura de las computadoras que lo conforman:

Clúster homogéneo: Todos los nodos tienen las mismas características técnicas de hardware y software. Son idénticos y por lo tanto la capacidad de procesamiento y rendimiento de cada nodo es la misma.

Clúster heterogéneo: Al contrario de los clúster homogéneos, los nodos son completamente distintos en cuanto a hardware y software.

C. Importancia de la computación de alto rendimiento

La programación paralela se origina por las limitaciones de las aplicaciones secuenciales; integrando varios procesadores para llevar a cabo sus funciones, la programación paralela permite resolver problemas que requieren más memoria o mayor velocidad de cómputo. También existen razones económicas, pues el precio de los ordenadores secuenciales no es proporcional a su capacidad computacional, mientras que la conexión de varios procesadores utilizando una red nos permite obtener un aumento de prestaciones prácticamente proporcional al número de procesadores con un coste adicional mínimo [15].

Además con el uso del clúster se logra reducir el tiempo de resolución de problemas computacionales, o bien resolver problemas que no cabrían en la memoria de un solo procesador secuencial. Y para esto es necesario utilizar sistemas de altas prestaciones y algoritmos paralelos que utilicen estos sistemas de manera eficiente [16]

III. METODOLOGÍA

Durante el desarrollo de esta investigación, se utilizó la metodología de resolución de problemas que se organiza en siete etapas descritas a continuación:

1. **Identificación del problema**. Esta fase comprendió el estudio de la revisión bibliográfica y casos de éxito del funcionamiento de los clúster, en centros de investigación e instituciones de educación superior con la utilización de software libre y hardware convencional. Se analizó también la situación actual del laboratorio de cómputo de la Escuela de Ingeniería en Sistemas de la Universidad Nacional de Loja (CIS-UNL) para identificar si existían procesos que requieran procesar grandes volúmenes de información a altas velocidades y no se disponía de un supercomputador.
2. **Explicación del problema**. En el laboratorio de la CIS de la UNL no existían procesos que requieran grandes velocidades de procesamiento pero por la falta de un sistema computacional de alto rendimiento los docentes y estudiantes estaban privados de realizar proyectos académicos relacionados con el procesamiento de grandes volúmenes de información y se desconocía que con la utilización de hardware convencional y software libre se podían obtener grandes beneficios económicos, comparados con la adquisición de supercomputadores que realicen tareas dedicadas a la computación de alto rendimiento.
3. **Idear estrategias alternativas de intervención**. Para idear alternativas que permitieron solucionar el problema mencionado se realizó el análisis de los recursos técnicos de hardware, software y redes con las que cuenta el laboratorio de la CIS de la UNL, siendo el punto de partida para la implementación de la arquitectura clúster.
4. **Decidir la estrategia.** Una vez realizado el análisis de la situación actual del laboratorio de la CIS de la UNL, se procedió a realizar la búsqueda de las herramientas de software libre, la elección del middleware, la elección de la arquitectura clúster, el diseño de la topología y el direccionamiento de red.
5. **Diseño de la intervención.** En esta fase se determinó los tiempos de implementación del proyecto para el laboratorio CIS de la UNL y las actividades a cumplir en los plazos establecidos, logrando con éxito la culminación del mismo.
6. **Desarrollo de la intervención**. En esta fase se estableció la instalación y configuración de cada uno de los aplicativos de software libre mencionados anteriormente.
7. **Evaluación de los logros**. La evaluación de los logros obtenidos se la realizó aplicando pruebas de procesamiento a distintos proyectos, logrando evidenciar tiempos mínimos de ejecución al utilizar los recursos de la arquitectura clúster.

IV. PRUEBAS

Se realizaron diversas pruebas en la arquitectura clúster, para determinar la capacidad de procesamiento alcanzado, ejecutando aplicaciones que requieren alta capacidad computacional. Para realizar las pruebas a los siguientes proyectos se utilizó 1, 5, 10 y 15 nodos respectivamente.

A. Proyecto 1: Cálculo del valor aproximado de pi utilizando librerías MPI

El proyecto evalúa el algoritmo utilizado para calcular el valor de pi, utilizando librerías de paso de mensajes MPI.

TABLA I. RESULTADOS C´ALCULO DEL VALOR APROXIMADO DE PI UTILIZANDO MPI

Número de nodos	Número de procesadores	Tiempo de ejecución (seg)
1	8	0.959801
5	40	0.321487
10	80	0.228983
15	120	0.136832

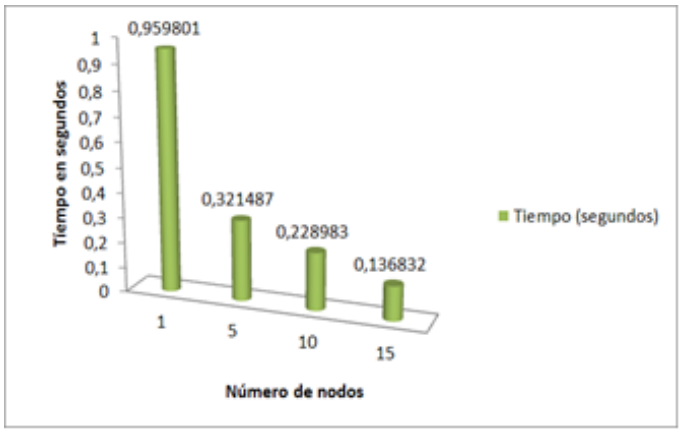

Fig 2. Tiempos de ejecución cálculo del valor aproximado de pi.

De la prueba realizada se deduce que utilizando un solo nodo, el tiempo de ejecución es de 0.959801 segundos, mientras que al utilizar los 15 nodos de la arquitectura el tiempo es de 0.136832 segundos, como se observa en la Tabla. I, obteniendo una disminución de 0.822969 segundos de tiempo de procesamiento, lo que se interpreta como una eficiencia de aproximadamente 88%, como se ilustra en la Fig. 2.

B. Proyecto 2: Cifrado de códigos con John The Ripper (JTR) y MPI.

Las pruebas se realizaron sobre uno de los ficheros más importantes de Linux como lo es shadow, este fichero almacena información cifrada de las contraseñas de cada una de las cuentas de usuario del sistema operativo. La contraseña utilizada tiene una longitud de 8 símbolos, y consta de 62 combinaciones de caracteres (26 letras del abecedario mayúsculas + 26 letras del abecedario minúsculas + 10 dígitos), por lo que se realizaran cerca de 218 trillones de posibles combinaciones.

TABLA II. RESULTADOS CIFRADO DE CÓDIGOS CON JOHN THE RIPPER (JTR) Y LIBRERÍAS MPI

Número de nodos	Número de procesadores	Tiempo de ejecución (horas)
1	8	09:48:34
5	40	07:21:22
10	80	03:06:57
15	120	00:27:24

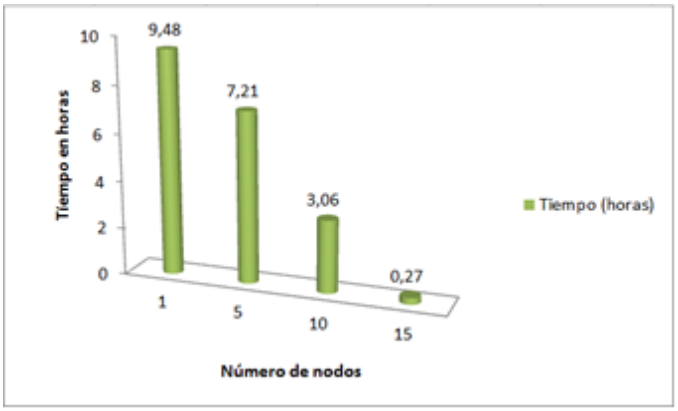

Fig 3. Tiempo de ejecución de cifrado de código utilizando John The Ripper

El tiempo de ejecución al descifrar el fichero shadow, en un solo nodo es de 9.48 horas, mientras que al utilizar los 15 nodos de la arquitectura el tiempo de la decodificación es de 27 minutos, como se detalla en la Tabla II obteniendo una disminución de 9.21 horas de tiempo de procesamiento, lo que se interpreta como una eficiencia de aproximadamente 97%, ver Fig. 3.

En la Fig. 4, se observa el balanceo de carga al ejecutar el proceso de cifrado de código utilizando John The Ripper y librerías MPI en los 15 nodos de la arquitectura clúster.

Fig 4. Balanceo de carga en el clúster al ejecutar MPICH+JTR

C. Proyecto 3: Renderización de imagen y video con Blender

Este proyecto trata sobre el armado final de un cubo de rubik que está formado por 115 frames, una vez terminado el renderizado se obtiene como producto final, un video con una duración aproximada de 11 segundos y un tamaño total de 18.5 MB.

TABLA III. RESULTADOS RENDERIZACIÓN DE IMÁGENES Y VIDEOS CON BLENDER

Número de	Número de	Tiempo de

nodos	procesadores	ejecución (min)
1	8	15:40
5	40	09:13
10	80	04:06
15	120	01:31

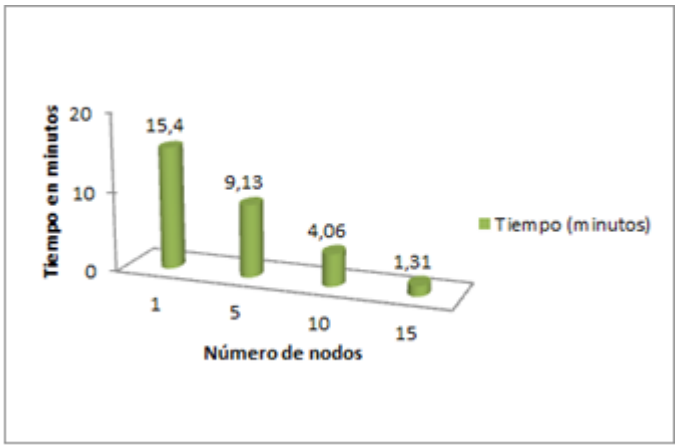

Fig 5. Tiempo de ejecución de renderización de imágenes y videos con Blender.

El tiempo de ejecución utilizando Blender para renderizar el proyecto en un solo nodo es de 15.4 minutos, en cambio utilizando los 15 nodos de la arquitectura el tiempo de renderizado es de 1.31 minutos, ver Tabla III, obteniendo una disminución de 14.09 minutos de tiempo de procesamiento, alcanzando un rendimiento de aproximadamente 92%, como se observa en la Fig. 5.

D. Proyecto 4: Análisis de datos (Big Data) con Apache Hadoop

Para esta prueba se utilizó un dataset sobre la medición de la calidad del aire en España en las estaciones de Castilla y León, en el que se encuentra información recopilada desde el año 1997 hasta el año 2013, el mismo que se encuentra disponible en el siguiente enlace http://goo.gl/jZO6OT. El fichero de extensión .csv, fue modificado a nuestros requerimientos con lo que se logró obtener un archivo que contiene más de 16 millones de líneas y un peso mayor a 950 Mb.

TABLA IV. RESULTADOS DE ANÁLISIS DE DATOS (BIG DATA) CON HADOOP.

Número de nodos	Número de procesadores	Tiempo de ejecución (seg)
1	8	92
5	40	78
10	80	54
15	120	32

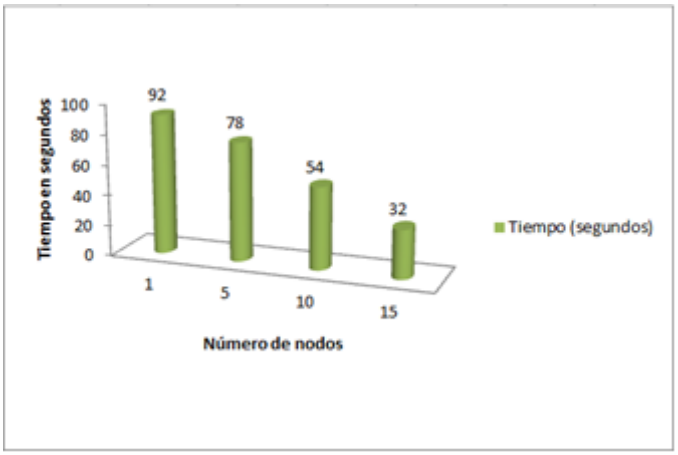

Fig 6. Tiempo de ejecución de análisis de datos (Big Data) con Hadoop

Utilizando el framework Apache Hadoop para el análisis de datos (Big Data), se deduce que utilizando un solo nodo, el tiempo de ejecución es de 1 minuto con 32 seg, mientras que al utilizar los 15 nodos de la arquitectura el tiempo es de 32 segundos de tiempo de procesamiento, como se detalla en la Tabla IV obteniendo una disminución de 60 segundos de diferencia, lo que se interpreta como una eficiencia de aproximadamente 74% de aceleración de los procesos relacionados al análisis de datos, ver Fig. 6.

E. Proyecto 5: Pruebas de compresión de música con MOSIX

En el proyecto realiza la compresión de 32 canciones en formato .wav a un formato de ficheros de audio .flac, haciendo uso de Mosix para la migración automática de los procesos en cada uno de los nodos del clúster. El tamaño de los archivos antes de realizar la compresión es de 5.2 Gb, luego de realizar el proceso el tamaño es de 1.9 Gb.

TABLA V. RESULTADO DE LA EJECUCIÓN, PRUEBAS DE COMPRESIÓN

Número de nodos	Número de procesadores	Tiempo de ejecución (min)
1	8	22:16
5	40	11:23
10	80	09:34
15	120	07:45

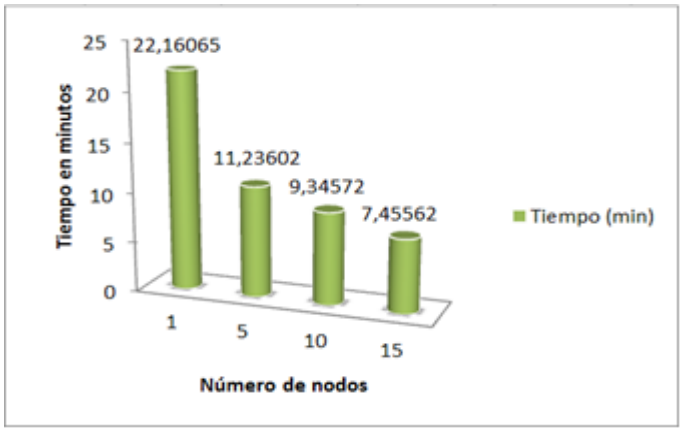

Fig 7. Tiempo de ejecución de pruebas de compresión de música con Mosix

De la prueba realizada con el middleware Mosix para la compresión de los ficheros de audio se deduce que utilizando un solo nodo, el tiempo de ejecución es de 22.16065 minutos, mientras que al utilizar los 15 nodos de la arquitectura el tiempo es de 7.45562 minutos, ver Tabla V, obteniendo una disminución de 14.70503 tiempo de procesamiento, lo que se interpreta como una eficiencia de aproximadamente 75%, ver Fig. 7.

F. Monitorización de la arquitectura clúster

Gracias al sistema de monitoreo Ganglia Monitoring System, tenemos registradas las actividades de cada uno de los nodos de la arquitectura clúster, como se observa en la Fig. 10 permitiendo recolectar métricas como: ocupación de los procesadores, uso de memoria, espacio en disco, etc [15].

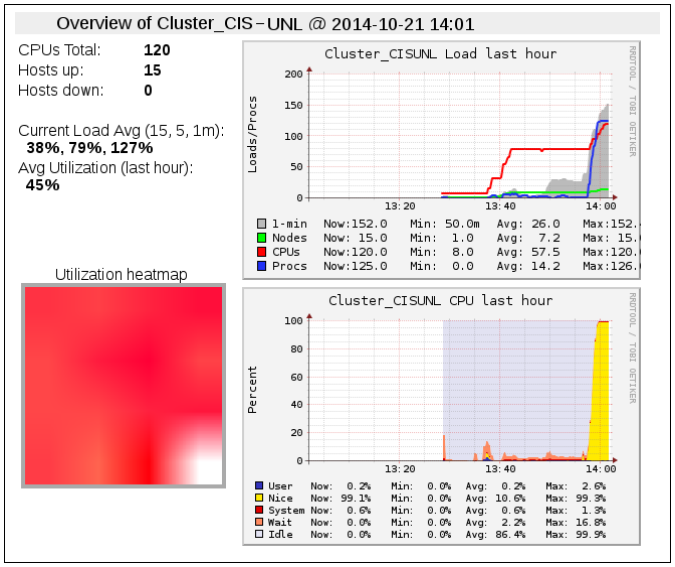

Fig 8. Monitorización arquitectura clúster de alto rendimiento CIS-UNL

V. RESULTADOS

A. Esenario real

Los resultados se obtubieron en base al siguiente esenario real:

El diseño lógico de la arquitectura clúster fue la siguiente:

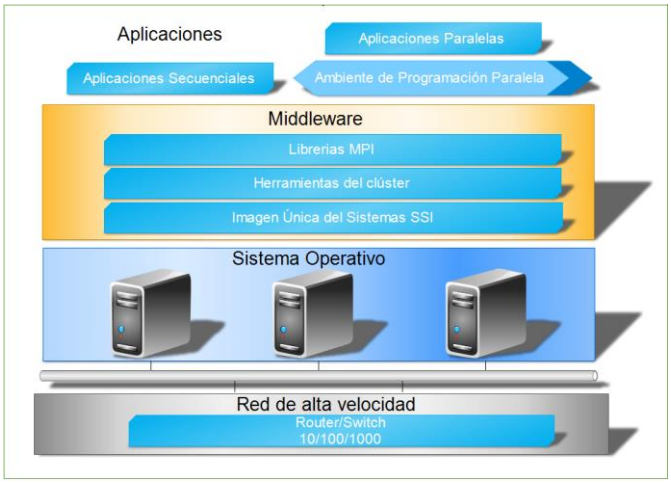

Fig 9. Diseño lógico arquitectura clúster de alto rendimiento CIS-UNL

Para la capa inferior o la red de datos de alta velocidad se utilizaron:
Tarjetas de red PCI Adapter 802.11 que soportan velocidades de hasta 1000 Mbps; un router CISCO LINKSYS EA4500 Giga bit Ethernet con una velocidad de transmisión de 450 Mbps y un Switch: CISCO Catalyst 2960 de capa 2 con 48 puertos Gigabit Ethernet.

La topología de la red implementada fue de tipo estrella ya que con esta topología no es necesario que los nodos esclavos tengan conexión a internet, para poder hacer uso de este servicio en el nodo maestro se instalaron dos tarjetas de red, una para conectarse a la red LAN en la cual están enlazados todos los nodos que componen el clúster y la segunda tarjeta para la conexión a Internet. Ver Fig. 10.

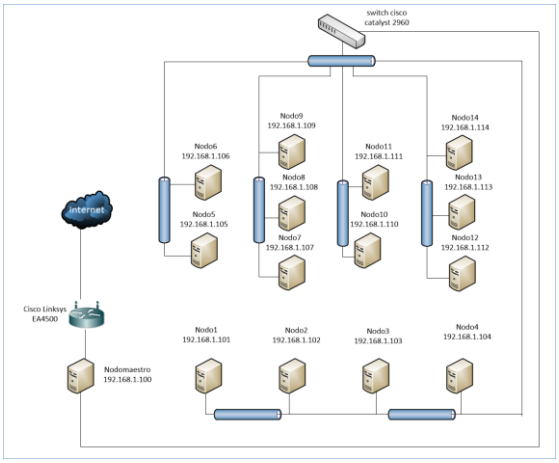

Fig 10. Diagrama de topología de la arquitectura clúster CIS-UNL

Para la capa del sistema operativo y el hardware se utilizó:

El Sistema operativo Debian 7.5 "Wheezy" (64 bits). El hardware del clúster de procesamiento fue armado con 15 computadoras personales Corei7-2600 de 3.64 GHz, memoria RAM 4 GB de 1333 MHz y disco duro de 500 GB con 7200 rpm. Al final los recursos hardware del clúster de procesamiento son los indicados en la tabla VI.

TABLA VI. RECURSOS TOTALES DE LA ARQUITECTURA CLÚSTER

Recursos	Descripción
Capacidad en memoria RAM	60 GB
Número de procesadores	120 de 3.4 GHZ
Capacidad en disco duro	7.32 TB

Para el Middleware se utilizó herramientas de software libre como:

Middleware Mosix 3.4.0.12, Compiladores: GCC 4.5.2. Soporte para C, C++, MPICH 3.1.2, como implementación del estándar Message Passing Interface (MPI), John The Ripper 1.7.9-jumbo-7.

Para la capa superior o de aplicacioes se utilizó herramientas de software libre como:

Blender 2.72b, Apache Hadoop 2.5.1, Ganglia Monitoring System 3.3.8.

B. Evaluación de resultados

Los principales resultados obtenidos son los que se indican en la tablas II, III, IV, V y VI; es fácil deducir entonces que el clúster de procesamiento implementado con los recursos hardware y software antes mencionados cumple con las expectativas académicas para el procesamiento de grandes volúmenes de información logrando con ello los resultados esperados al finalizar el proyecto.

Al no contar con un supercomputador para realizar mediciones de tiempos de procesamiento de grandes volúmenes de información, no fue posible realizar un estudio comparativo con los resultados obtenidos del clúster de procesamiento armado con hardware convencional y software libre por lo que asumimos que el clúster mencionado tiene características técnicas de procesamiento similares a las de un supercomputador.

VI. CONCLUSIONES Y TRABAJOS FUTUROS

La arquitectura clúster tipo beowulf es la más óptima en ambientes universitarios ya que se implementó utilizando hardware convencional y herramientas de software libre, con lo cual se obtiene un ahorro generalizado en costes de administración, mantenimiento y monitorización, comparadas con la adquisición de un supercomputador. Se debería armar una arquitectura clúster de procesamiento con al menos 15 computadores personales con 8 procesadores de 3.6 GHz cada uno para tener una arquitectura eficiente, ya que al realizar la ejecución de varios tipos de algoritmos complejos como: cifrado de código, renderización de imágenes, análisis de datos (Big Data), compresión de ficheros de audio, esta arquitectura clúster alcanzó un rendimiento del 85.2%, en relación a la capacidad de procesamiento de un solo computador convencional. Además la arquitectura clúster de alto rendimiento fue diseñada para proporcionar capacidad de procesamiento de grandes volúmenes de datos, lo que permite realizar investigaciones en campos como: minería de datos, Big Data, sistemas de gestión de base de datos, evaluación de estructuras de datos, evaluación de algoritmos, programación paralela, reconocimiento de patrones, entre otros. En esta arquitectura clúster se puede ejecutar una amplia gama de aplicaciones de cómputo científico, para la evaluación de métodos numéricos y técnicas de optimización, solucionando problemas principalmente de ciencia e ingeniería.

VII. AGRADECIMIENTOS

Los autores agradecen a la Universidad Nacional de Loja, el Área de la Energía, las Industrias y los Recursos Naturales no Renovables, y a la Carrera de Ingeniería en Sistemas por el apoyo técnico y logístico desinteresado para la culminación exitosa del proyecto. De igual manera los autores agradecen el apoyo académico desinteresado de los docentes investigadores de la Escuela Superior Politécnica de Chimborazo y de la Universidad Técnica de Manabí.

VIII. REFERENCIAS

[1] G. Cáceres, "Estrategia de implementación de un clúster de alta disponibilidad de N nodos sobre linux usando software libre", 2012.

[2] L. M. Santos Jaimes, S. Peñaloza, and E. R. Cruz Cruz, "Cluster implementation of a prototype for the resolution of a particular problem", Journal Article, vol. 1 2010.

[3] N. Pérez Otero, S. Méndez, C. V. Ayusa, M. I. Aucapiña, and V. J. Lopez, "Aplicaciones del cómputo de altas prestaciones", in XI Workshop de Investigadores en Ciencias de la Computación, 2013.

[4] D. Zhao, K. Qiao, and I. "Raicu, HyCache+: Towards Scalable High-Performance Caching Middleware for Parallel File Systems", in IEEE/ACM CCGrid, 2014.

[5] J. d. J. R. Quezada, S. B. Rionda, J. M. V. Félix, and I. A. M. Torres, "Diseño e implementación de un clúster de cómputo de alto rendimiento", Acta Universitaria, vol. 21, pp. 24-33, 2011.

[6] R. Gualán, A. Vásquez and O. Vega, "Una primera aproximación a la implementación de un clúster para la ejecución de un modelo de predicción climática", 2012

[7] R. Quezada S. B. Rionda, J. Felix, and I. Torres, "Diseño e implementación de un clúster de cómputo de alto rendimiento, Acta Universitaria, vol.21, pp. 24-33, 2011.

[8] A. Barack and A. Shiloh, "The MOSIX Cluster Operating System for Distributed Computing on Linux Clusters, Multi-Clusters and Clouds", 2014.

[9] M. C. O. System, "Administrator's, User's and Programmer's Guides and Manuals", July 2014.

[10] R. Latham and A. J. Pe, "MPICH Installer's Guide", Mathematics and Computer Science Division Argonne National Laboratory, 2014.

[11] Blender, "Blender is a free and open source 3D animation suite" [Online]. Available: http://www.blender.org/. [Accessed: 05-Nov-2014].

[12] R. Serrat Morros, "Big Data: análisis de herramientas y soluciones", 2013.

[13] Apache, “Hadoop - Apache Hadoop 2.5.1.” [Online]. Available: http://hadoop.apache.org/docs/r2.5.1/index.html. [Accessed: 13-Nov-2014].
[14] G. M. System, “Ganglia Monitoring System” [Online]. Available: http://ganglia.sourceforge.net/. [Accessed: 04-Dec-2014].
[15] B. Otero, R. Astudillo, and Z. Castillo, “Un esquema paralelo para el cálculo del pseudoespectro de matrices de gran magnitud,” Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería, 2014.
[16] A. Sheharyar and O. Bouhali, “A Framework for Creating a Distributed Rendering Environment on the Compute Clusters”, arXiv preprint arXiv:1401.0608, 2014
[17] R. Bhatnagar and J. Patel, “Performance Analysis of A Grid Monitoring System-Ganglia”, International Journal of Emerging Technology and Advanced Engineering, vol. 3, pp. 362-365, 2013.

Leonardo Chuquiguanca, Administrador de Sistemas en la empresa Protelcotelsa S.A**.** Ingeniero en Sistemas graduado en la Universidad Nacional de Loja - Ecuador, Activista de software libre, Administrador de servidores, redes y telecomunicaciones, Seguridad de información. Provincia de Loja, Ciudad Loja, Ecuador, 2015.

Edyson Malla Ingeniero en Sistemas del Gobierno Autónomo Descentralizado Municipal San Felipe de Oña, Azuay − Ecuador (Marzo 2015 hasta la actualidad). Ingeniero en Sistemas graduado en la Universidad Nacional de Loja − Ecuador (Febrero 2015). Conocedor de software libre, redes, telecomunicaciones, análisis y diseño de sistemas. Provincia de Loja, Ciudad Loja, Ecuador, 2015.

Freddy Ajila, Docente Investigador de la Escuela de Ingeniería Industrial, Facultad de Mecánica de la Escuela Superior Politécnica de Chimborazo ESPOCH (Desde Octubre 2014 hasta la actualidad). Profesor de Sistemas Operativos, Arquitectura de Computadoras y Estructuras de Datos de la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja – Ecuador (Abril 2013 – Julio 2014). Magister en Telemática graduado en la Universidad de Cuenca – Ecuador (Agosto 2011). Ingeniero en Informática graduado en la Universidad Técnica Particular de Loja – Ecuador (Junio del 2006). Activista de software libre, Administrador de servidores, redes y telecomunicaciones. Provincia de Chimborazo, Ciudad Riobamba, Ecuador, 2015.

Rene Guamán Quinché Docente Investigador de la Carrera de Ingeniería en Sistemas Informáticos de la Universidad Técnica de Manabí. Experto en tecnologías y accesibilidad Web. Magister en Sistemas Informáticos Avanzados en la Universidad del País Vasco y candidato a Doctor por la Escuela Politécnica Nacional. Provincia de Manabí, Ciudad Portoviejo, Ecuador, 2015