UNIVERSIDAD DE ORIENTE
NÚCLEO DE SUCRE
ESCUELA DE CIENCIAS
DEPARTAMENTO DE INFORMÁTICA

IMPLEMENTACIÓN DE UN SISTEMA DE CÓMPUTO DE ALTO RENDIMIENTO PARA EL CENTRO DE INVESTIGACIÓN EN SERVICIOS DE CÓMPUTO DE LA UNIVERSIDAD DE ORIENTE

(Modalidad: Pasantía de grado)

VÍCTOR EZEQUIEL MAZA DÍAZ

TRABAJO DE GRADO PRESENTADO COMO REQUISITO PARCIAL PARA OPTAR AL TÍTULO DE LICENCIADO EN INFORMÁTICA

CUMANÁ, 2017

IMPLEMENTACION DE UN SISTEMA DE COMPUTO DE ALTO RENDIMIENTO PARA EL CENTRO DE INVESTIGACION EN SERVICIOS DE CÓMPUTO DE LA UNIVERSIDAD DE ORIENTE

APROBADO POR:

_____

Prof. José Sifontes
Asesor Académico

_____

Msc. José Romero
Asesor Institucional

_____

Jurado

_____

Jurado

ÍNDICE

DEDICATORIA

Dedico esta tesis A. DIOS, Por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos, además de su infinita bondad y amor.

A todos aquellos que no creyeron en mí, a aquellos que esperaban mi fracaso en cada paso que daba hacia la culminación de mis estudios, a aquellos que nunca esperaban que lograra terminar la carrera, a todos aquellos que pensaban que me rendiría a medio camino, a todos los que supusieron que no lo lograría, a todos ellos les dedico esta tesis.

A mis padres. Por brindarme su apoyo incondicional, además que han sido el eje fundamental de mi vida y mi carrera estudiantil.

A mi hermano por estar siempre presente en todo momento.

A mi familia. Quienes me inspiraron mi espíritu para la conclusión de esta tesis.

A mi tía Evelia Díaz (QEPD). Por quererme y apoyarme siempre, esto también se lo debo, gracias.

AGRADECIMIENTO

Dedico esta tesis a mis amigos de “la Legión”, Cecilia Rivas, Pablo Fajardo, David Acevedo, Carlos Madrigal, por hacer esta trayectoria más agradable.

A esos amigos que sin darme cuenta pasaron a ser parte de tu familia, José Torrens, Jessica Marcano, Huber Colina, José Peñuela, Quienes fueron un gran apoyo emocional durante el tiempo en que escribía esta tesis.

A la familia Torrens. Por regalarme ese apoyo incondicional y estar presente en toda ocasión.

A mis asesores José Francisco Romero y José Sifontes, por confiar en mí desde el principio, compartir sus conocimientos y brindarme todo su apoyo.

A todos aquellos profesores que me apoyaron a lo largo de esta carrera, por su tiempo y los conocimientos brindados para el desarrollo de este proyecto.

LISTA DE TABLAS

LISTA DE FIGURAS

RESUMEN

Se presenta una estrategia para la implementación de un sistema de cómputo de alto rendimiento en la Universidad de Oriente Núcleo de Sucre. Para lo cual se realizó un estudio de la situación de los profesores que se encuentran realizando proyectos que requieran una gran cantidad de procesamiento; en donde pudo apreciarse carencia de tecnología que le permita trabajar dentro del Núcleo, lo cual conllevó, mediante la aplicación del modelo incremental propuesto por Harlan Mills, con el objetivo de ofrecer soluciones adaptables a las necesidades de los requerimientos de los usuarios y reducir la repetición del trabajo en el proceso de desarrollo, así dar oportunidad de retrasar la toma de decisiones en los requisitos hasta adquirir experiencias con el sistema. Se evaluaron diferentes soluciones a la implementación del *Clúster*, a través del análisis de la tecnología de red, arquitectura del nodo, sistema de archivos, administración de procesos y herramientas de programación. Para el montaje de este *Clúster* fue necesaria la instalación y configuro del sistema operativo GNU/Linux Debian "jessie" en todas las máquinas por su estabilidad y funcionamiento, además la configuración del protocolo NFS encargado de compartir carpetas de forma tal que puedan ser accedidas remotamente, también SSH la cual brindaba una conexión segura entre las máquinas y poder acceder a ellas de forma remota, librerías MPI las cuales fueron necesarias para el procesamiento en paralelo, para el middleware se utilizó MOSIX como gestor entre las máquinas y lograr obtener un sistema distribuido, como también, como un punto extra se empleó el framework HADOOP para el análisis masivo de información, se implementó un sistema de motorización con la herramienta GANGLIA Monitoring System que funciona perfectamente para observar el estado de vida de cada uno de los nodos. El fin de la computación paralela es resolver problemas más grandes en menos tiempo y a bajo costo. Por último se realizaron las pruebas con el fin de demostrar la reducción del tiempo de procesamiento trabajando en la computadora de manera secuencial y la distribución de la misma tarea en varias máquinas trabajando de forma paralela. En síntesis, este trabajo muestra que el estudio de la performance y optimización de *Clúster* permite obtener una notable mejoría en el uso de los recursos facilitando el objetivo fundamental de la computación paralela que es permitir resolver problemas más grandes en menor tiempo.

Palabras Claves: *Clúster Beowulf*, MOSIX, HADOOP, *OPENMPI, NFS, SSH, GANGLIA Monitoring System,* Computación Paralela.

INTRODUCCIÓN

A lo largo de la historia de la computación se han presentado diversos problemas de tipo complejo, que en el pasado no podían ser resueltos debido al poco poder de computo existente para la época o el costo de sus soluciones era alcanzable para algunos. Sin importar que tan potentes puedan ser las máquinas actuales, comparadas con sus predecesoras, cada vez queremos resolver problemas de mayor complejidad que ameritan gran procesamiento y precisión de cómputo.

Inicialmente se plantearon algoritmos secuenciales para hallar soluciones a los problemas complejos, sin embargo estos algoritmos han pasado a ser ineficientes debido a su alto consumo de tiempo de ejecución (Mesa y Branch, 2008). Como respuesta a esta limitación nacieron los llamados supercomputadores, los cuales cuentan con arreglos de microprocesadores que trabajan en sincronía empleando procesamiento paralelo. Éstos, son máquinas más costosas que no se podrán encontrar en el mercado, debido a que usan la tecnología más avanzada posible.

El procesamiento en paralelo es empleado para acelerar el tiempo de ejecución de un programa, dividiéndolo en múltiples trozos que se ejecutarán al mismo tiempo, cada uno en sus propios procesadores. En los últimos años, el personal académico de diversas universidades y centros de investigación, se han dado a la tarea de aprender a construir sus propios supercomputadores como alternativa para lograr un cómputo en paralelo con altas prestaciones a menor costo, esto lo lograron conectando computadores personales y desarrollando software para enfrentar tales problemas. A esta solución se les conoce con el nombre de *clúster*, que pueden definirse como un grupo de múltiples computadores unidos por una red de alta velocidad, de tal forma que su conjunto puede ser visto como una única máquina.

El cómputo de alto rendimiento (HPC, por sus siglas en inglés) es un tipo de *clúster* que puede procesar grandes cantidades de información en poco tiempo con mínimo presupuesto, por esta razón es utilizado para aplicaciones de carácter científico y de cálculo intensivo como algoritmos genéticos, aplicaciones militares, análisis de sismos, entre otros. Un claro ejemplo de ello es Google o la NASA, los cuales poseen unos de los más grandes *clúster* del mundo, que les permite manejar su inmensa base de datos y procesar un gran nivel de cálculo computacional para sus estudios científicos.

El primer *clúster* de computadores fue construido en 1994, por Donald Becker y Thomas Sterling en la NASA, agruparon 16 procesadores Intel DX4 de 100 MHz, en equipo viejos con Linux instalado como sistema operativo, los interconectaron con tecnología Ethernet a 10 Mbps. Este tipo de *clúster* ha tenido mucha aceptación, debido a su capacidad para el cómputo paralelo y alto rendimiento, su objetivo es obtener una poderosa herramienta de cómputo con ordenadores comerciales, ésta podrá acelerar el tiempo de ejecución de los procesos suministrados.

En Venezuela distintas universidades han empleado la creación de *clúster* como alternativa para lograr sistemas de cómputo aceptables con respecto a las necesidades actuales, por ejemplo, el Centro Nacional de Calculo Científico de la Universidad de Los Andes (CeCalCULA), el cual ofrece servicios de cómputo de alto rendimiento en *clúster* Linux.

En la Universidad De Oriente Núcleo Sucre se formó El Centro de Investigación en Servicios de Computación (CISC), en el año 2011. Es un proyecto incubado del Parque Tecnológico de Oriente, que pretende canalizar la investigación en servicio de cómputo avanzado, investigación de operaciones, procesamiento distribuido, simulación y modelaje de procesos en diversas áreas (tales como: matemática, física, química, biología, medicina, sociología, entre otras), ofreciendo de manera inmediata un soporte de cómputo de alto rendimiento a docentes, estudiantes, institutos de investigación, e instituciones privadas que posean vínculos con la Universidad de Oriente.

Este trabajo de investigación tiene como propósito implementar una estrategia para creación de un sistema de cómputo de alto rendimiento para aprovechar de una forma eficiente la capacidad de procesamiento de los computadores personales o de escritorio ubicados en CISC para que funcionen como *Clúster* bajo la administración del *framework* HADOOP, de esta forma poder cubrir la necesidad de una infraestructura con un poder de computo aceptable que pueda utilizarse para calculo o análisis de datos.

Está estructurado en tres (3) capítulos, los cuales se especifican a continuación:

Capítulo I. Presentación: este capítulo corresponde al planteamiento del problema, donde se describe la problemática planteada y la importancia del trabajo, así como el alcance y limitaciones encontrados durante su desarrollo.

Capítulo II. Marco de referencia: Está conformado por dos (2) secciones principales: el marco teórico, en el cual se presenta los antecedentes de la investigación, antecedentes de la institución, además se exponen los fundamentos teóricos necesarios base para la aplicación desarrollada. El marco metodológico, presenta la metodología de la investigación y la metodología del área aplicada para dar la solución propuesta al problema planteado.

Capítulo III. Desarrollo: en este capítulo se presentan, describen, y desarrollan cada una de las fases que componen la metodología aplicada además se presentan los resultados de las pruebas realizadas para este proyecto.

Finalmente se prestan las conclusiones, recomendaciones, la bibliografía, apéndices y anexos los cuales complementan el contenido del trabajo realizado.

CAPÍTULO I. PRESENTACIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

Actualmente, la Universidad de Oriente, Núcleo Sucre cuenta con grupos de profesores/investigadores en las áreas de Física, Química y Matemáticas pertenecientes a dependencias adscritas a la universidad como: el Laboratorio de Física de Metales y el Laboratorio de Materiales que se encuentran trabajando con programación en paralelo y necesitan de sistemas de cómputo de alto rendimiento para agilizar y concluir sus investigaciones.

En estos momentos no existe una plataforma oficial dentro de la Universidad, capaz de cubrir los requerimientos y necesidades de estudiantes y profesores en las áreas científicas referentes al uso de sistemas de cómputo de alto rendimiento.

A pesar de las dificultades, algunos investigadores han elaborado, de forma independiente, sus propias soluciones para cómputo de alto rendimiento. Las cuales, están ubicadas en laboratorios pertenecientes a la universidad y son utilizados únicamente por sus desarrolladores para procesar sus propios cálculos e investigaciones.

Este esfuerzo, realizado de manera aislada, genera gastos innecesarios a la universidad y a los investigadores. Además, el acceso a esas herramientas está restringido para el resto de la comunidad universitaria, por no existir un ente dedicado a la administración de estos recursos.

Debido a la falta de instituciones competentes en el tema, en el 2011, se formó El Centro de Investigación en Servicios de Computación (CISC), el cual tiene entre sus objetivos la creación de un *clúster* que pueda resolver los requerimientos de la UDO, Núcleo de Sucre y extender sus servicios a otros usuarios en toda Venezuela. Sin embargo, dicho objetivo se ha retrasado por diversas razones; lo cual ha acrecentado la necesidad de una solución óptima que garantice la satisfacción de los investigadores que hacen vida dentro de la universidad.

Como solución al problema expuesto, se propone la Implementación de un Sistema de Cómputo de Alto Rendimiento para el Centro de Investigación en Servicios de Computación de la Universidad de Oriente (CISC). Este sistema estará instalado en el laboratorio perteneciente a la institución y permitirá el acceso a investigadores y estudiantes ligados a la UDO para la realización de cálculos y otras tareas computacionales.

1.2 ALCANCE Y LIMITACIONES

1.2.1 **Alcance**

El alcance de este proyecto se limita al planteamiento de una estrategia para la implementación de un Sistema de Alto Rendimiento usando herramientas y aplicaciones de distribución libre para el Sistema Operativo Linux. La propuesta contenida en esta tesis proveerá los lineamientos básicos de configuración de un clúster de N nodos. Sin embargo, el número total de nodos del prototipo del *clúster* desarrollado para demostrar la funcionalidad de la estrategia propuesta será de 3 nodos (un nodo Maestro y dos nodos esclavos), al mismo tiempo generara la construcción de conocimiento de nuevas áreas, crear una curva de aprendizaje de esta implementación.

1.2.1 Limitaciones

La carrera de licenciatura en Informática no cuenta con una línea de investigación formal para el estudio de plataformas de cómputo de alto rendimiento. Solo se dictan cátedras orientadas al desarrollo de software con capacidad de operar en este tipo de entornos. Lo anterior implicó mayor complejidad, tiempo y desarrollo en el análisis trabajo.

La carrera de Licenciatura en Informática no cuenta con la formación educativa en el área de cómputo y con tutores especializados en implementación de *Clúster*, esto

requirió ayuda de tutores extranjeros (Francia y Grecia) lo que implicó mayor complejidad en el desarrollo del análisis.

Durante la investigación se consiguió poco material bibliográfico sobre la implementación y administración de plataformas de alto rendimiento que utilicen de forma simultanea MOSIX, HADOOP y *GANGLIA*.

Los equipos adquiridos por la Universidad para la realización de este proyecto tuvieron algunas fallas físicas lo que ocasionó daños severos en algunas piezas del hardware esto implico tiemplo en la implementación del *Clúster*.

CAPÍTULO II. MARCO REFERENCIAL

2.1 MARCO TEORICO

2.1.1 Antecedentes de la investigación

A nivel nacional e internacional en las universidades se han desarrollado múltiples *Clúster* o *Grid* que sirven como antecedentes a esta investigación, tal es el caso de las siguientes:

León (2004), en la Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales, un trabajo titulado: análisis de performance y optimización en *clúster Beowulf*. El objetivo de este trabajo es presentar un estudio de la performance y optimización de *clúster Beowulf* a través del análisis de la tecnología de red, arquitectura del nodo, sistema de archivos, administración de procesos herramientas de programación, para obtener una notable mejoría en el uso de los recursos facilitando el objetivo fundamental de la computación paralela que es permitir resolver problemas más grandes en menor tiempo. Esta investigación hace referencia al estudio y configuración de un *clúster Beowulf*, que será de utilidad para el entendimiento de la investigación, como también, estrategias para el análisis y configuración del *clúster*.

Iturriaga, Maya y Pintos (2008), en Facultad de Ingeniería en Universidad de la República. Un trabajo titulado: Proyecto Fenton - *Clúster* de Computadores de Alto Desempeño con Acceso Remoto (CCADAR). El proyecto consistió en la implementación de un *clúster* de computadoras de alto desempeño con la capacidad de brindar acceso remoto para que sus usuarios interactúen con él y para que sus administradores lo regulen y monitoreen. El cual será de utilidad para el estudio de su estrategia metodología y la documentación.

Velázquez (2010), realizó en el Instituto Politécnico Nacional (IPN) de México, un trabajo titulado: construcción y diseño de un clúster tipo HPC virtualizado. Tuvo como objetivo la construcción de 2 tipos de *clúster*, uno físico y el otro virtual. A esto se le hizo una comparación con distintas pruebas computacionales complejos para una máquina secuencial, estas pruebas de desempeño ayudaron a escoger la mejor solución para el desarrollo de problemas que requerían. La utilidad de este trabajo es mejorar el entendimiento del *clúster* y de algunos protocolos y software que utilizaron como también su contenido teórico.

Cáceres (2012), realizó en la Universidad San Francisco de Quito, en Ecuador, un trabajo titulado: Estrategia de Implementación de un *Clúster* de Alta Disponibilidad de N Nodos Sobre Linux Usando Software Libre. El objetivo de este trabajo fue la construcción de un clúster de alta disponibilidad permite el acceso continuo a los servicios y aplicaciones más dentro de una organización. También explica como instalar, configurar y probar un clúster. Será de utilidad para el estudio de su estrategia metodológica y la documentación.

Echevarría (2014) realizo en la Universidad Pontificia Comillas, en Madrid, un trabajo titulado estudio, análisis y evaluación del *framework* HADOOP. El objetivo principal de este proyecto es la investigación en el ecosistema HADOOP para una mejor comprensión. La utilidad de este trabajo es mejorar el entendimiento del HADOOP y entender su comportamiento, también su contenido teórico.

Mays (2012), realizó en la UDO una investigación titulada: *Volunteer Computing* en la Universidad de Oriente Núcleo Sucre. Esté logró resolver problemas computacionalmente intensivos, apoyando y promoviendo a investigadores que necesitaban gran capacidad de cómputo y procesamiento. En este estudio, se recabó información sobre la construcción de una enlace *volunteer* para la UDO, por esta recopilación de información de utilidad para el entendimiento de la investigación, como a su vez, técnicas y materiales utilizados para la integración de esa implementación.

2.1.2 Bases Teóricas

2.1.2.1 Área de Estudio

Este proyecto se encuentra enmarcado dentro del área de redes y telecomunicaciones.

Telecomunicaciones

La telecomunicación (del prefijo griego tele, "distancia" o "lejos", "comunicación a distancia") es una técnica consistente en transmitir un mensaje desde un punto a otro, normalmente con el atributo típico adicional de ser bidireccional. El término cubre todas las formas de comunicación a distancia, incluyendo radio, telegrafía, televisión, telefonía, transmisión de datos e interconexión de ordenadores a nivel de enlace (Herrera, 1998).

Teleinformática

Son un conjunto de técnicas necesarias para transmitir datos dentro de un sistema informático o entre puntos situados en lugares remotos, a través de redes de telecomunicaciones (Castro, 1999).

Redes de datos

Una red de datos es un sistema que enlaza dos o más puntos (nodos) por un medio físico, el cual sirve para enviar o recibir un determinado flujo de información (Groth, 2005). Otro tipo de redes son las redes de computadoras las cuales son múltiples computadoras conectadas entre ellas que utilizan un sistema de comunicaciones. El objetivo de esta red es que las computadoras se comuniquen y compartan información, una computadora que forma parte de una red puede proveer de servicios a otras computadoras denominadas clientes mediante un servidor (Joyanes, 1998). Los servidores son el punto central en las redes modelo cliente/servidor. Existen muchos servicios que un servidor puede brindar a los clientes de red. Por ejemplo Sistema de Nombres de Dominio (DNS), Protocolo de Configuración Dinámica de Host (DHCP).

2.1.3 Área de investigación

Este proyecto se encuentra enmarcado dentro del área de la computación distribuida. Los conceptos y principios fundamentales a tener en cuenta en el área, son los siguientes:

Sistema

Es un conjunto de componentes que interaccionan entre sí para lograr un objetivo en común (Senn, J., 1992). El concepto de sistema se puede adaptar a diferentes ámbitos de nuestra vida, uno de esos ámbitos es el área computacional donde se adapta a los sistemas de información.

Sistemas Distribuidos

Un sistema distribuido se define como una colección de computadores autónomos, conectados por una red, y con el software distribuido adecuado para que el sistema sea visto por los usuarios como una única entidad, capaz de proporcionar facilidades de computación (Coloris, 2001).

Los sistemas distribuidos se implementan en diversas plataformas de hardware, desde unas pocas estaciones de trabajo conectadas por una red de área local hasta internet que enlazan millones de computadores.

Arquitectura de computadoras

La clasificación más popular la realizó Flynn (Bosque José Luis, 2006), que clasificó la arquitecturas de acuerdo a los flujos de datos (se les conoce también como *data streams*) y a los flujos de instrucciones (*instructions streams*). El concepto de flujos de datos se refiere al número de operandos que se pueden procesar al mismo tiempo y el de flujos de instrucciones se refiere a cuántos programas se pueden ejecutar al mismo tiempo. De acuerdo a esta clasificación existen cuatro tipos de computadoras:

Single Instruction Stream, Single Data Stream (SISD). Un solo flujo de instrucciones y un solo flujo de datos. Arquitectura Von Neuman.

Single Instruction Stream, Multiple Data Stream (SIMD). Un solo flujo de instrucciones y varios flujos de datos. Sus características principales son, entre otras, cuenta con una sola unidad de control que despacha instrucciones a todos los elementos de procesamiento (EP), todas las unidades de proceso ejecutan la misma instrucción. Algunas computadoras comerciales que cuentan con este tipo de arquitectura son: Illiac IV, MPP, DAP, CM-2, MasPar MP1 y MasPar MP2. Arquitecturas vectoriales.

Multiple Instruction Stream, Single Data Stream (MISD) Varios flujos de instrucciones y un solo flujo de datos. No hay implementaciones.

Multiple Instruction Stream, Multiple Data Stream (MIMD) Varios flujos de instrucciones y varios flujos de datos. Algunas características son: todos los elementos de procesamiento (EP) poseen su propia unidad de control, todas las unidades de proceso ejecutan su propio

programa. Algunas máquinas comerciales con esta arquitectura son el nCUBE2, iPSC, Symmetry, FX-8, FX-2800, TC-2000, CM-5, KSR-1 y la Paragon XP/S.

Es importante conocer la organización de la memoria en la arquitectura MIMD, esta se divide en:

Arquitectura de paso de mensajes, cada procesador tiene su memoria local o privada, cada procesador puede comunicarse con los demás solo a través del paso de mensajes, a estas computadoras suelen denominarse multicomputadoras (conjunto de máquinas Von Neumann (procesador + memoria local) conectadas a un bus común, a través del cual se pueden comunicar los procesadores y en particular pueden hacer operaciones de lectura y escritura sobre memorias de máquinas remotas). Cabe mencionar que las multicomputadoras están divididas en *MPPs* (*Massively Parallel Processors*) y *NOWs/COWs* (*Network/Clúster Of Workstations*). Los MPPs son supercomputadoras demasiado caras compuestas por varios *CPU's* "estrechamente acoplados" por una interconexión de alta velocidad, dos ejemplos conocidos de éste tipo de supercomputadoras son "CRAY T3E" y SP/2 de IBM. Por otro lado los *NOWs*, los cuales son conocidos también como *COWs*, consisten en PC's regulares o estaciones de trabajo (Workstation) "finamente acoplados" por la tecnología de interconexión comercial.

Arquitectura de memoria compartida, Todos los procesadores comparten toda la memoria. Con esto se tiene tiempo de acceso a memoria uniformes, ya que todos los procesadores se encuentran igualmente comunicados con memoria principal, además el acceso a memoria es por medio de un solo canal. En esta configuración debe asegurarse que los procesadores no tengan acceso simultáneamente a regiones de memoria de una manera en la que pueda ocurrir algún error.

Arquitectura *Non Uniform Memory Arquitecture* (NUMA), además de la memoria compartida, cada procesador tiene una memoria local para guardar el programa y los datos no compartidos. En este caso disminuye el acceso a la red por parte de cada procesador haciendo más eficiente el acceso a memoria.

Arquitectura Cache *Only Memory Arquitecture* (COMA), solo quedan las memorias locales para cada procesador y cada procesador puede acceder a la memoria de cualquier otro, se le conoce como memoria virtual.

A continuación se presenta un pequeño cuadro comparativo de las arquitecturas SIMD y MIMD, como se muestra en la tabla 1.

Tabla 1. Comparación de arquitecturas SIMD y MIMD

SIMD y MIMD	**MIMD**
Requiere menos hardware: 1 unidad de control.	Requiere más hardware.
Necesita menos memoria: 1 sola copia del programa.	Necesita memoria para cada uno de los programas.
Adecuada para programas de datos paralelos: Se requiere el mismo programa sobre un gran número de datos.	Puede ejecutar varias tareas al mismo tiempo o emular un procesador SIMD mediante mecanismos de sincronización.
Menos tiempo de arranque para comunicarse con los vecinos: Debido a que posee un reloj global.	Para comunicarse es necesario usar mecanismos de sincronización.
Más costosa: Se requiere diseñar un microchip de arquitectura especial.	Más barata: Se pueden construir usando procesadores SISD de propósito general a gran escala, los cuales son muy baratos debido a la economía de escala.

Computación paralela

En la actualidad la computación paralela está siendo utilizada en multitud de campos para el desarrollo de aplicaciones y el estudio de problemas que requieren gran capacidad de cómputo, ya sea por el gran tamaño de los problemas que abordan o por la necesidad de trabajar con problemas en tiempo real. De esta forma, el paralelismo en la

actualidad, además de constituir diversas líneas abiertas de intensa labor investigadora, puede encontrarse en una infinidad de aplicaciones en campos muy variados, entre los que se destacan.

Modelo predictivo y simulación.- Se realiza mediante extensos experimentos de simulación por computador que con frecuencia acarrean computaciones a gran escala para obtener la precisión y el tiempo de respuesta deseado. Entre otros modelos se puede destacar la previsión meteorológica, numérica y la oceanografía.

El desarrollo industrial también reclama el uso de computadores para progresar en el diseño y automatización de proyectos de ingeniería, la inteligencia artificial y la detección remota de los recursos terrestres. En este campo se puede destacar la inteligencia artificial y automatización (procesamiento de imágenes, reconocimiento de patrones, visión por computador, comprensión del habla, ente otras.)

Investigación médica.- En el área médica las computadoras rápidas son necesarios en tomografía asistida, diseño de corazones artificiales, diagnostico hepático, estimación de daños cerebrales y estudios de ingeniería genética.

Modelos de cómputo en paralelo

En primer lugar se debe conocer que un clúster es un tipo de computador paralelo o distribuido que consiste en un tipo de computadoras independientes pero interconectadas entre sí que trabajan conjuntamente como un único recurso para resolver un problema común, de igual manera puede ser un multiprocesador “casero”, esto es que utiliza componentes comerciales, baratos y potentes (COTS). Ahora bien, los clústeres se pueden clasificar atendiendo a diversos factores como a continuación se menciona:

Clúster Beowulf

Es una tecnología para agrupar computadoras basadas en el sistema operativo Linux, para formar una supercomputadora virtual paralela; consiste de un nodo maestro y uno o más nodos conectados en red, construido con elementos de hardware comunes en el mercado, similar a cualquier PC capaz de ejecutar Linux, adaptadores de red y *switches* estándar. Los nodos en el *Clúster* están dedicados exclusivamente a ejecutar tareas asignadas al *Clúster* (Plaza, 2002).

Open Source Cluster Application Resources (OSCAR)

Es una recopilación de los mejores métodos conocidos para construir, programar y usar *Clúster* de alto rendimiento. Consiste de un paquete de software que contiene todos los programas necesarios para instalar, construir, mantener y utilizar un *Clúster* sobre Linux de tamaño modesto.

La arquitectura de un *Clúster* OSCAR es muy similar a la de un *clúster Beowulf*. Se configura un nodo como maestro, el cual provee los servicios a los nodos de cómputo. En el *clúster Beowulf*, una vez que se ha configurado el nodo maestro, se deben construir cada uno de los nodos de cálculo. OSCAR, en cambio, asiste en la configuración del nodo principal y construye los nodos de cálculo basados en una descripción especificada por el usuario. Esta construcción y configuración reduce considerablemente el esfuerzo y la necesidad de habilidades especiales para conformar un *clúster* (Des Ligneris, 2003).

Grid Computing (Computación en malla)

Tiene sus orígenes a principios de los 80’s donde la investigación intensiva en el campo de la algoritmia, la programación y arquitecturas que soporten paralelismo, provocaron que los primeros éxitos en este campo, ayuden a otros procesos de investigación en

diversas áreas; además de su exitoso uso en la ingeniería y en el campo empresarial (Smith, 2004).

En la actualidad, los entornos de computación en malla son las más prometedoras infraestructuras en la investigación computacional. Un *Grid Computing* es la infraestructura de hardware y software que provee acceso fiable, consistente y a bajo costo, a altas capacidades computacionales y bajo la certificación digital. Estas capacidades computacionales son explotadas mediante el desarrollo de aplicaciones diseñadas bajo la perspectiva de la programación paralela o distribuida, para la optimización de procesos que aletarguen el flujo que se desea optimizar. Estos procesos deben cumplir la precondición de poderse subdividir en procesos más pequeños, unos algorítmicamente independientes de los otros (Foster y Kesselman, 2008).

Una característica particular del *Grid Computing*, es que los nodos o equipos que lo conforman no tienen por qué ser computadores dedicados. Esta propiedad, denominada pertenencia dinámica, implica que cualquier equipo puede adherirse o abandonar un determinado *Grid* en tiempo de ejecución. Además, el hecho de que un equipo este formando, en un momento dado, parte de un determinado G*rid*, no implica que deje de ser operativo para cualquier otro tipo de tarea, sino, que en diferentes instantes de tiempo, un porcentaje de sus recursos, tales como el uso de CPU o almacenamiento secundario, serán utilizados por el mismo para la ejecución de determinadas tareas (Morillo, 2003).

Cloud Computing (Computación en nube)

Es un modelo tecnológico que permite el acceso ubicuo, adaptado y bajo demanda en red a un conjunto compartido de recursos de almacenamiento, aplicaciones y servicios, que pueden ser rápidamente aprovisionados y liderados con un esfuerzo de gestión reducido o interacción mínima con el proveedor del servicio (Urueña, 2012).

Algunas de las características del *Cloud Computing* son:

Pago por uso

Una de las características principales de las soluciones *Cloud* es el modelo de facturación basado en el consumo, es decir, el pago que debe abonar el cliente varía en función del uso que se realiza del servicio *Cloud* contratado.

Abstracción

Capacidad de aislar los recursos informáticos contratados al proveedor de servicios *Cloud* de los equipos informáticos del cliente. Esto se consigue gracias a la virtualización, con lo que la organización usuaria no requiere de personal dedicado al mantenimiento de la infraestructura, actualización de sistemas, pruebas y demás tareas asociadas que quedan del lado del servicio contratado.

Agilidad en la escalabilidad

Consiste en aumentar o disminuir las funcionalidades ofrecidas al cliente, en función de sus necesidades puntuales sin necesidad de nuevos contratos ni penalizaciones. De la misma manera, el coste del servicio asociado se modifica también en función de las necesidades puntuales de uso de la solución.

Multiusuario

Capacidad que otorga el *Cloud* que permite a varios usuarios compartir los medios y recursos informáticos, permitiendo la optimización de su uso.

Autoservicio bajo demanda

Esta característica permite al usuario acceder de manera flexible a las capacidades de computación en la nube de forma automática a medida que las vaya requiriendo, sin necesidad de una interacción humana con su proveedor o proveedores de servicios *Cloud*.

Acceso sin restricciones

Es la posibilidad ofrecida a los usuarios de acceder a los servicios contratados de *Cloud Computing* en cualquier lugar, en cualquier momento y con cualquier dispositivo que disponga de conexión a redes de servicio IP. El acceso a los servicios de *Cloud Computing* se realiza a través de la red, lo que facilita que distintos dispositivos, tales como teléfonos móviles, dispositivos PDA u ordenadores portátiles, puedan acceder a un mismo servicio ofrecido en la red mediante mecanismos de acceso comunes.

Volunteer Computing (computación voluntaria)

Tiene como base la compartición de recursos lo que supone un ahorro en la compra de equipos potentes y en suministro eléctrico. De esta forma se permite a centros que no poseen la última tecnología realizar investigaciones de forma rápida y eficiente, además es un sistema perfectamente escalable con relativamente poco esfuerzo (Taracón, 2009).

La computación voluntaria se ha consolidado en todo el mundo como una herramienta eficaz y fiable para el cálculo científico aprovechando las computadoras de ciudadanos e instituciones cuando están ociosos, es decir, cuando esos ordenadores están encendidos pero no se están utilizando o se usan con recursos computacionales mínimos (Taracón, 2009).

La donación de ciclos de procesamiento inutilizados ha sido una idea bastante aceptada entre las personas alrededor del mundo, quienes se han unido voluntariamente a proyectos de investigación en distintas áreas. Todos estos proyectos están basados en la idea de alistamiento de personas, siendo éstas últimas las que toman la decisión de

brindar ciclos computacionales de sus computadoras personales para ser utilizados por un proyecto de investigación, elegido según sus intereses (Abbas, 2004).
Luego de la suscripción a un proyecto específico, un pequeño programa de control es descargado al computador. Este programa es responsable de la comunicación con el servidor central del proyecto, así como el uso de la capacidad brindada, ejecutando procesos enviados por este servidor. Típicamente, estos proyectos utilizan pequeños paquetes de comunicación para manejar grandes procesamientos en la computadora asociada al proyecto (Abbas, 2004).

La computación voluntaria presenta las siguientes características (Kacsuk y cols., 2008):

Presenta un conjunto de computadoras unidas a una red compartida. Este conjunto puede poseer computadoras dedicadas, computadoras conectadas de forma intermitente y computadoras compartidas con otras actividades.

Cada computadora desconoce de la existencia de los demás computadoras, exceptuando el servidor central del proyecto.

Un mecanismo de control de envío, ejecución y recuperación de las tareas a procesar, todo esto bajo el control del servidor central del proyecto.

Al momento de especificar un algoritmo paralelo debe tenerse en cuenta los siguientes pasos a realizar (Grama y cols., 2003):

Identificar las porciones de código que puede ser ejecutada en paralelo.

Mapear las porciones de trabajo que se ejecutan en paralelo en los diferentes computadoras pertenecientes al *Volunteer Computing*.

Distribuir los datos de ingreso, las salidas y la información intermedia en la aplicación.

Controlar el acceso a la información compartida requerida por el programa por los diferentes computadores.

Sincronizar los diferentes computadores teniendo en cuenta el algoritmo paralelo.

El desarrollo de librerías de paso de mensaje entre procesadores ha contribuido considerablemente en el desarrollo de los sistemas distribuidos, una característica a destacar de esta metodología es que no importa la arquitectura ni plataforma de software o hardware para procesar la información, como PVM y MPI.

El desarrollo de *Parallel Virtual Machine* (PVM) dio inicio en el verano de 1989, cuando *Vaidy Sunderan*, profesor de la Universidad de *Emory* visitó el Laboratorio Nacional de *Oak* Ridge para realizar investigaciones con Al *Geist* en el área de la computación distribuida heterogénea. Ellos necesitaban de una plataforma para explorar esta nueva área y poder así desarrollar el concepto de Máquina Virtual Paralela (PVM). En el año de 1991, *Bob Manchek*, un investigador de la Universidad de *Tenesse* formó parte del grupo de investigación para desarrollar una versión portable y robusta de PVM (Gesit y Kohl, 1996).

El uso de PVM creció rápidamente alrededor del mundo, siendo la comunidad científica quien corría la voz de la utilidad de este software para la investigación (Gesit y Kohl, 1996).

La idea central del diseño de PVM fue la noción de una "Máquina Virtual". Un aspecto de la máquina virtual fue el cómo las tareas en paralelo intercambiaban datos. En PVM esto fue llevado a cabo mediante el uso de paso de mensajes. La portabilidad fue considerada más importante que el rendimiento, por dos razones, la comunicación a través de Internet era lenta y; la investigación estaba enfocada hacia problemas con escalamiento, tolerancia a fallos y heterogeneidad de la máquina virtual.

En Febrero de 1993, gracias a los numerosos aportes de los usuarios de PVM, se reescribió completamente el código de la herramienta, teniendo como resultado PVM. El software PVM ha sido distribuido gratuitamente, y es hoy en día uno de los más usados en el desarrollo de aplicaciones distribuidas alrededor del. *Message Passing Interface* (MPI) o llamado estándar MPI, cuyas especificaciones fueron culminadas en abril de 1994, es el sobrevenir de un esfuerzo comunitario para tratar de definir tanto la sintaxis como la semántica de un núcleo de rutinas de librerías de paso de mensajes que podrían ser muy útiles para los usuarios que desean implementarla de forma eficiente en un amplio rango de Sistemas de Procesadores Masivamente en Paralelo (MPPs) (Gesit y Kohl, 1996).

Uno de los objetivos del desarrollo de MPI es el de proveer a los fabricantes de MPPs un conjunto de instrucciones claramente definidas que puedan implementarse de forma eficiente, y en algunos casos, proveer el soporte para el hardware y por consiguiente el mejorar la escalabilidad de estos sistemas.

Clúster

Se define como un tipo de sistema distribuido o paralelo conformado por una colección de computadoras interconectadas usado como un único recurso de computación unificado (Pfister, 1998).

Un c*lúster* es un grupo de equipos independientes que cooperan para la ejecución de una serie de aplicaciones de forma conjunta que se comportan ante clientes como un solo sistema. Los Clúster permiten aumentar la escalabilidad, disponibilidad y fiabilidad de múltiples niveles de red.

Un c*lúster* está formado por dos o más servidores independientes pero interconectados. Algunos están configurados de modo tal que puedan proveer alta disponibilidad permitiendo que la carga de trabajo sea transferida a un nodo secundario si el nodo

principal deja de funcionar. Otros Clúster están diseñados para proveer escalabilidad permitiendo que los usuarios o carga se distribuya entre los nodos. Ambas configuraciones son consideradas Clúster.
Una característica importante es que se presentan a las aplicaciones como si fueran un solo servidor. Es deseable que la administración de los diversos nodos sea lo más parecida posible a la administración de una configuración de un solo nodo. El software de administración del Clúster debería proveer este nivel de transparencia.

Los supercomputadores tradicionales poseen costos excesivamente elevados cuya capacidad de procesamiento puede ser fácilmente reemplazada con un Clúster. Los *Clústers* son una buena solución cuando lo que se busca es mejorar la velocidad, fiabilidad y escalabilidad a un precio razonable.

Clasificación de *Clúster*

Los *Clústers* pueden ser clasificado según su origen: a nivel de software y a nivel de hardware.

A nivel de Software. Mediante el uso de un sistema operativo que brinde las herramientas necesarias para la creación de un Clúster, tal es el caso de un *kernel Linux* modificado, compiladores y aplicaciones especiales, los cuales permitan que los programas que se ejecutan en el sistema exploten todas las ventajas del Clúster.

A nivel de Hardware. Mediante la interconexión entre máquinas (nodos) del Clúster, las cuales se juntan utilizando redes dedicadas de alta velocidad como por ejemplo Gigabit Ethernet. Cuando se trata de brindar balanceo de carga mediante un Clúster el hardware y software trabajan conjuntamente para distribuir la carga de tráfico a los nodos, para de ésta manera poder atender eficientemente las sub-tareas encomendadas y con ello la tarea general asignada al Clúster.

Un servicio de alta disponibilidad en el Clúster normalmente no distribuye la carga de tráfico a los nodos (balanceo de carga) ni comparte la carga de procesamiento (alto rendimiento) sino más bien su función es la de estar preparado para entrar inmediatamente en funcionamiento en caso de que falle algún otro servidor.

Características de los *clúster*

Un Clúster debe cumplir con algunos requisitos o características que se detallan a continuación:

Los nodos de un Clúster están conectados entre sí por al menos un medio de comunicación.

Los *Clústers* necesitan software de control especializado. Para tener un funcionamiento correcto de un Clúster, el diseño y modelado del mismo depende del tipo de Clúster utilizado. El software y las máquinas conforman el Clúster.

Software de control para gestión del Clúster. Control que se refiere a la configuración del Clúster, el cual depende del tipo de Clúster y de la manera en que se conectan los nodos. El control puede ser de dos tipos según sea el caso:

Control centralizado. El cual consta de un nodo maestro o director, con el que se puede configurar todo el sistema.

Control descentralizado. Control en el que cada nodo se administra y gestiona individualmente.

Homogeneidad de un Clúster. Caracterizado por estar formado de nodos con arquitecturas y recursos similares. Este tipo de Clúster son implementados a nivel de sistema.

Heterogeneidad de un Clúster

Son caracterizados por tener diferencias a nivel de tiempos de acceso, arquitecturas, sistemas operativos, estas dos últimas conllevan a tener bibliotecas de carácter general las cuales van a ser utilizadas como interfaz para poder formar un sistema conjunto entre los nodos. Este tipo de *Clústers* son implementados a nivel de aplicación.

Entre las características de funcionalidad que presentan los *Clústers* se mencionan:

Escalabilidad

Los Clúster permiten agregar nuevos componentes para aumentar el nivel de prestaciones sin necesidad de eliminar los elementos ya existentes.

El balanceo de carga (LB) ofrece escalabilidad: la distribución de peticiones en varios servidores. LB consiste en el reenvío de paquetes y en el conocimiento del servicio cuya carga va a balancearse. Se basa en un monitor externo que recoge las estadísticas de carga de los servidores físicos para decidir donde se deben enviar los paquetes.

La escalabilidad es la capacidad de un equipo para hacer frente a volúmenes de trabajo cada vez mayores sin, por ello, dejar de prestar un nivel de rendimiento aceptable. Existen dos tipos de escalabilidad:

Escalabilidad del hardware (también denominada «escalamiento vertical»). Se basa en la utilización de un gran equipo cuya capacidad se aumenta a medida que lo exige la carga de trabajo existente.

Escalabilidad del software (también denominada: escalamiento horizontal). Se basa, en la utilización de un Clúster compuesto de varios equipos de mediana potencia que

funcionan en tándem de forma muy parecida a como lo hacen las unidades de un *RAID* (*Array* redundante de discos de bajo coste). Se utilizan el término *RAC* (*Array* redundante de equipos) para referirse a los *Clústers* de escalamiento horizontal. Del mismo modo que se añaden discos a un *array RAID* para aumentar su rendimiento, se pueden añadir nodos a un Clúster para aumentar también su rendimiento.

Disponibilidad

Existe redundancia natural, cada nodo posee sus propios componentes: bus, memoria, procesador. Se pueden implementar políticas para el reemplazo rápido en caso de falla del servidor maestro.

La alta disponibilidad

Ofrece fiabilidad: mantiene los servicios ejecutándose. Se basa en servidores redundantes, intercambio de mensajes del tipo "Estoy vivo", y un procedimiento para que en caso de fallos se sustituya el servidor donde se produjo el error por otro. El beneficio de éste diseño es el de proveer disponibilidad y confiabilidad. La confiabilidad se provee mediante software que detectan fallos y permiten recuperarse frente a los mismos, mientras que en hardware se evita tener un único punto de fallo.

Rendimiento

En principio las aplicaciones paralelas son más rápidas. Entre los factores que influyen se encuentran:

Comportamiento del programa.

Carga de la red.

En éste diseño se ejecutan tareas que requieren de gran capacidad computacional, grandes cantidades de memoria, o ambos a la vez. El llevar a cabo éstas tareas puede comprometer lo recursos del Clúster por largos períodos de tiempo.

El objetivo principal de un Clúster de Alto Rendimiento ó HPC "*High Performance Computing*" es alcanzar el mayor rendimiento en la velocidad de proceso de datos. Este tipo de tecnología nos permite que un conjunto de computadoras trabajen en paralelo, dividiendo el trabajo en varias tareas más pequeñas las cuales se pueden desarrollar de forma paralela.

Arquitectura de *clúster*

Como se puede observar la arquitectura posee varios componentes, ver figura 1.

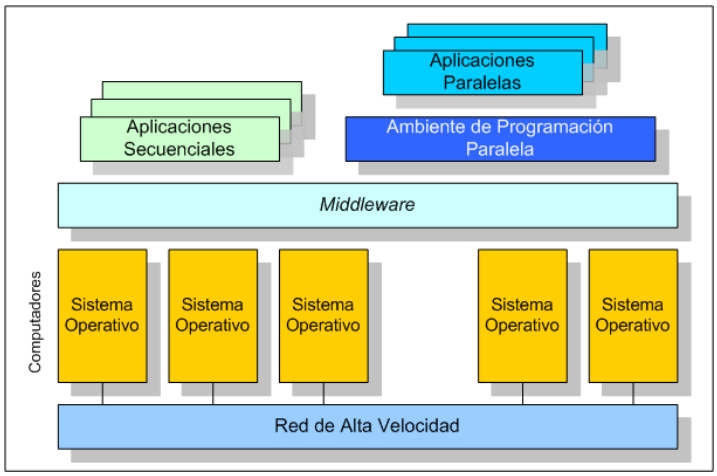

Figura 1. Componentes de un Clúster

Como se puede observar la arquitectura posee varios componentes que se detallan a continuación:

Componentes de un *clúster:*

- Nodos
- Sistemas Operativos

Conexiones de Red

Middleware

Protocolos de Comunicación y servicios

Ambientes de Programación Paralela

Aplicaciones

Nodos

Pueden ser simples ordenadores, sistemas multiprocesador o estaciones de trabajo *Workstation*). En informática de forma muy general, un nodo es un punto de intersección o unión de varios elementos que confluyen en el mismo lugar.

En redes de computadoras cada una de las máquinas es un nodo, y si la red es Internet, cada servidor constituye también un nodo. En un Clúster con nodos dedicados, los nodos no disponen de teclado, mouse ni monitor y su uso está exclusivamente dedicado a realizar tareas relacionadas con el Clúster. Mientras que, en un Clúster con nodos no dedicados, los nodos disponen de teclado, mouse y monitor y su uso no está exclusivamente dedicado a realizar tareas relacionadas con el Clúster.

Los nodos de cómputo están conformados a nivel de hardware por diferentes dispositivos los cuales permiten su funcionamiento individual y dentro del Clúster: procesadores, memoria, caché, dispositivos de entrada/salida y buses de comunicación.

Procesadores. Estos proveen la capacidad computacional del nodo. Pueden ser de diferentes fabricantes: Intel con sus familias Pentium 3, 4, Itanium y Xeon, HP y Compaq con su familia Alpha, dispositivos AMD (Athlon) entre otros.

Memoria RAM (Random Access Memory). Es la memoria principal del computador la cual almacena los datos que está utilizando en ese instante. Esta memoria se actualiza

constantemente mientras el ordenador está en uso y pierde todos los datos cuando el sistema se apaga.

Caché. La memoria caché es pequeña y de alta velocidad de acceso que sirve de búfer para aquellos datos de uso frecuente. La caché está conectada a la memoria principal la cual tiene una velocidad de acceso menor.

Dispositivos de entrada/salida (I/O). Dispositivos de mucha utilidad para un computador, por medio de dispositivos de entrada puede escribirse en memoria desde el exterior, ubicándose junto con los datos sobre los cuales va a operar, de igual manera debe existir dispositivos de salida para conocer los resultados de la ejecución de los programas. Los dispositivos de entrada/salida (I/O) permiten la comunicación entre el computador(memoria) y el mundo exterior (periféricos), dentro de dispositivos periféricos que pueden interactuar con un computador se tiene:

Dispositivos de presentación de datos. Dentro de este grupo están periféricos como el teclado, pantalla, ratón e impresora. Dispositivos que interactúan directamente con el usuario.

Dispositivos de almacenamiento de datos. Dispositivos que interactúan con la máquina, dentro de estos se encuentran las cintas magnéticas y discos magnéticos.

Dispositivos de comunicación con otros procesadores. Comunicación con otros procesadores remotos ya sea a través de una red LAN (Local Area Networks), una red WAN (Wide Area Networks) o el bus de interconexión.

Bus del sistema. Periférico utilizado para transmisión de datos entre los diferentes dispositivos del computador con el procesador y la memoria.

Sistemas operativos

Un nodo en un Clúster (no siempre) es una entidad de cómputo autónoma completa y que posee su propio sistema operativo. Los *Clústers Beowulf* explotan las características sofisticadas de los sistemas operativos modernos para la administración de los recursos de los nodos y para la comunicación con los otros nodos a través de la red de interconexión.

Pueden utilizarse sistemas operativos modernos entre los que se tiene a Linux, Unix, Windows entre otros. Estos sistemas operativos son multitarea lo cual permite compartir recursos entre usuarios, dividir el trabajo entre procesos, asignar recursos cada proceso (memoria y ciclos de procesador) y tener al menos un hilo de ejecución.

Conexiones de red

Existen diferentes redes de alta velocidad utilizadas en un Clúster. Se puede citar algunas: Ethernet, ATM, SCI, cLAN, Myrinet, Infinibad y QsNet.

Ethernet, Fast Ethernet y Gigabit Ethernet. Tecnologías muy utilizadas en el ámbito local, ethernet ofrece un ancho de banda de 10 Mbps, el cual en la actualidad es pequeño para soportar nuevas aplicaciones de gran demanda de acceso, es por ello que aparecieron las tecnologías fast ethernet, que brinda un ancho de banda de 100 Mbps y Gigabit Ethernet, que ofrece un ancho de banda de 1 Gbps.

CLAN Tecnología creada por Gigante ofrece un alto rendimiento para redes de alta velocidad, posee adaptadores PCI y *switches* de 8 y 30 puertos, ofreciendo velocidades de 1.25 Gbps por puerto (2.5 Gbps bidireccional). Es una tecnología muy utilizada en la interconexión de los diferentes dispositivos pertenecientes a un Clúster.

Myrinet. Tecnología de alta velocidad diseñada por *Myricom* en Noviembre de 1998. El ancho de banda que maneja cada adaptador de red y los *switch* ha incrementado desde 640 Mbps hasta los 2.4 Gbps, teniendo tiempos de entrega de paquetes que fluctúan entre los 7 y 10 micro segundos. Cada nodo posee una tarjeta de red PCI-X con una o dos conexiones, las cuales pueden manejar velocidades 2 Gbps o 10 Gbps bidireccionales, estas tarjetas se interconectan a través de un cable Myrinet (fibra óptica) a un *switch* Myrinet de hasta 128 puertos. Esta tecnología posee un software el cual detecta automáticamente a la red Myrinet sin necesidad de configurar el conmutador o *switch*.

Infiniband. El estándar InfiniBand es un bus de comunicaciones de alta velocidad, diseñado tanto para conexiones internas como externas. Para la comunicación utiliza un bus bidireccional con lo cual ofrece dos canales de transmisión independientes. El ancho de banda básico de un enlace simple.

QsNet. Igual que las tecnologías myrinet e infiband, QsNet está conformada de dos partes, la interfaz de red Elan y el *switch* Elite, el cual dispone de 16 a 128 puertos, este *switch* posee dos canales virtuales bidireccionales por enlace, es decir cada enlace posee dos puertos de entrada/salida con una tasa de transferencia teórica de 400 Mbyte/s en cada dirección. El *switch* provee también dos niveles de prioridad, los cuales son de gran ayuda en la entrega de paquetes de mayor importancia en el menor tiempo posible.

Middleware

El *middleware* es un software que generalmente actúa entre el sistema operativo y las aplicaciones con la finalidad de proveer a un Clúster lo siguiente:

Una interfaz única de acceso al sistema, denominada SSI(*Single System Image*), la cual genera la sensación al usuario de que utiliza un único ordenador muy potente;

Herramientas para la optimización y mantenimiento del sistema: migración de procesos, *checkpoint-restart* (congelar uno o varios procesos, mudarlos de servidor y continuar su funcionamiento en el nuevo host), balanceo de carga, tolerancia a fallos, etc.

Escalabilidad: debe poder detectar automáticamente nuevos servidores conectados al Clúster para proceder a su utilización.

Recibe los trabajos entrantes al Clúster y los redistribuye de manera que el proceso se ejecute más rápido y el sistema no sufra sobrecargas en un servidor. Esto se realiza mediante políticas definidas en el sistema (automáticamente o por un administrador) que le indican dónde y cómo debe distribuir los procesos, por un sistema de monitorización, el cual controla la carga de cada CPU y la cantidad de procesos en él.

El *middleware* también debe poder migrar procesos entre servidores con distintas finalidades:

Balancear la carga: si un servidor está muy cargado de procesos y otro está ocioso, pueden transferirse procesos a este último para liberar de carga al primero y optimizar el funcionamiento.

Mantenimiento de servidores: si hay procesos corriendo en un servidor que necesita mantenimiento o una actualización, es posible migrar los procesos a otro servidor y proceder a desconectar del Clúster al primero.

Priorización de trabajos: en caso de tener varios procesos corriendo en el Clúster, pero uno de ellos de mayor importancia que los demás, puede migrarse este proceso a los servidores que posean más o mejores recursos para acelerar su procesamiento.

El usuario no necesita conocer donde se ejecutan las aplicaciones.

El usuario puede conectarse al Clúster como un sistema único, sin necesidad de hacerlo de manera individual a cada nodo como es el caso de un sistema distribuido.

Escalabilidad del sistema, ya que los *Clústers* pueden ampliarse fácilmente añadiendo nuevos nodos, las aplicaciones deben ser capaces de ejecutarse de forma eficiente en un amplio rango de tamaños de máquinas.

Es importante la disponibilidad del sistema para soportar las aplicaciones de los usuarios, por medio de técnicas de tolerancia a fallos y recuperación automática sin afectar a las aplicaciones de los usuarios.

Protocolos de comunicación

Los protocolos son reglas de comunicación que permiten el flujo de información entre computadoras distintas que manejan lenguajes distintos, por ejemplo, dos computadores conectados en la misma red pero con protocolos diferentes no podrían comunicarse jamás, para ello, es necesario que ambas "hablen" el mismo idioma, por tal sentido, el protocolo TCP/IP fue creado para las comunicaciones en Internet. Para que cualquier computador que se conecte a Internet, es necesario que tenga instalado este protocolo de comunicación.

Ambientes de programación paralela

Los ambientes de programación paralela permiten implementar algoritmos que hagan uso de recursos compartidos: CPU (*Central Processing Unit*), memoria, datos y servicios. En la figura 2. Se muestra gráficamente los elementos de un Clúster:

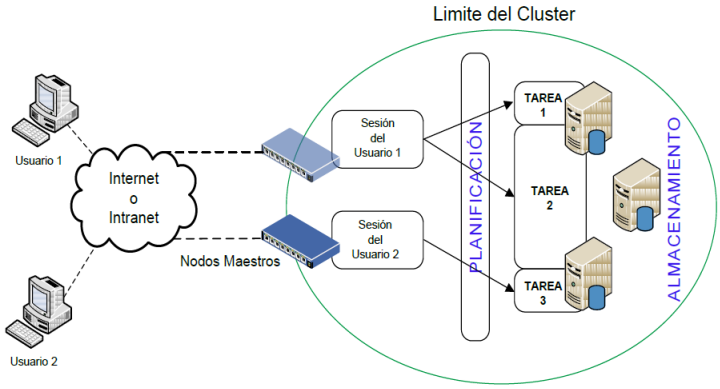

Figura 2. Elementos de un Clúster

Aplicaciones

Estas pueden ser paralelas o distribuidas y secuenciales

Aplicaciones Paralelas. Son aplicaciones que se distribuyen entre varias máquinas y plataformas para trabajar en forma integrada, típicamente sobre una red, para realizar una variedad de funciones relacionadas, tal es el caso de la arquitectura cliente/servidor caracterizado por dividir la funcionalidad de la aplicación en dos papeles: cliente y servidor. El servidor se encarga de proporcionar una serie de servicios al cliente, los cuales son utilizados por los clientes para completar la funcionalidad de una determinada aplicación.

Aplicaciones Secuenciales. Es aquella en la que una acción (instrucción) sigue a otra en secuencia. Las tareas se suceden de tal modo que la salida de una es la entrada de la siguiente y así sucesivamente hasta el fin del proceso.

Tipos de clúster

Los *Clústers* dependiendo de su aplicabilidad pueden clasificarse de diferentes maneras. La clasificación más generalizada es la que se presenta a continuación:

Alto rendimiento (HP, high performance)

Alta disponibilidad (HA, high availability)

Balanceo de Carga (Load Balancing)

Tolerantes a fallos

Alta Confiabilidad (HR, high reliability).

Clúster de alto rendimiento

Los Clústers de alto rendimiento han sido creados para compartir el recurso más valioso de un ordenador, es decir, el tiempo de proceso. Cualquier operación que necesite altos tiempos de CPU puede ser utilizada en un Clúster de alto rendimiento, siempre que se encuentre un algoritmo que sea paralelizable. Existen Clústers que pueden ser denominados de alto rendimiento tanto a nivel de sistema como a nivel de aplicación.

Este tipo de Clúster lo que busca es suplir las necesidades de súper computación para resolver problemas de determinadas aplicaciones que requieren un alto procesamiento, esto se logra mediante la utilización de un grupo de máquinas individuales las cuales son interconectadas entre sí a través de redes de alta velocidad y de esta manera se obtiene un sistema de gran rendimiento que actúa como uno solo.

La utilidad principal de este tipo de Clúster es principalmente en aplicaciones en las que se requieren gran capacidad de procesamiento computacional, la cual soluciona

problemas de alto procesamiento mediante la utilización de técnicas necesarias para la paralelización de la aplicación, distribución de los datos a los nodos, la obtención y presentación de resultados finales.

Generalmente estos problemas de cómputo suelen estar ligados a:

Estado del tiempo.
Cifrado y descifrado de códigos.
Compresión de datos.
Astronomía.
Simulación Militar.

Clúster de alta disponibilidad (ha, *high availability*).

Los *Clústers* de alta disponibilidad pretenden dar servicios 7/24 de cualquier tipo, son *Clústers* donde la principal funcionalidad es estar controlando y actuando para que un servicio o varios se encuentren activos durante el máximo periodo de tiempo posible.

El brindar alta disponibilidad no hace referencia a conseguir una gran capacidad de cálculo, si no lograr que una colección de máquinas funcionen en conjunto y que todas realicen la misma función que se les encomendó. La característica principal de éste Clúster es que ante la existencia de algún problema o fallo de uno de los nodos, el resto asumen ese fallo y con ello las tareas del nodo con problemas. Estos mecanismos de alta disponibilidad lo brindan de forma transparente y rápida para el usuario.

La escalabilidad en un Clúster de alta disponibilidad se traduce en redundancia lo cual garantiza una pronta recuperación ante cualquier fallo.

La flexibilidad y robustez que poseen este tipo de Clúster los hacen necesario en sistemas cuya funcionalidad principal es el intercambio masivo de información y el

almacenamiento de datos sensibles, dónde se requiere que el servicio esté presente sin interrupciones.

El mantenimiento es otra de las ventajas que ofrece. El mantenimiento se puede realizar de manera individual a cada máquina que compone el conglomerado evitando comprometer los servicios que este brinda.

Existen dos tipos de configuraciones aplicables a estos *Clústers*:

Configuración activo-pasivo: Esta configuración tiene dos actividades en los nodos que componen el Clúster, los activos son aquellos que se encargan de ejecutar las aplicaciones encomendadas, mientras que los nodos restantes actúan como respaldos redundantes para los servicios ofrecidos.

Configuración activo - activo: En este caso, todos los nodos actúan como servidores activos de una o más aplicaciones y potencialmente como respaldos para las aplicaciones que se ejecutan en otros nodos. Cuando un nodo falla las aplicaciones que se ejecutaban en él migran a uno de los nodos de respaldo.

Clúster de balanceo de carga

Técnica muy utilizada para lograr que un conjunto de servidores de red compartan la carga de trabajo y con ello el tráfico de sus clientes. Este proceso de dividir la carga de trabajo entre los servidores reales permite obtener un mejor tiempo de acceso a las aplicaciones y con ellos tener una mejor confiabilidad del sistema.

Además como es un conjunto de servidores el que atiende el trabajo, la falla de uno de ellos no ocasiona una falla total del sistema ya que las funciones de uno, las puede suplir el resto.

Balanceo de carga (*load balancing*)

El balanceo de carga permite suplir las necesidades del inminente crecimiento del tráfico en Internet. Existen dos alternativas para manipular el crecimiento del tráfico en internet: la primera permite usar una máquina de grandes características y de alto precio que probablemente a futuro quede obsoleta; la segunda alternativa consiste en utilizar un conjunto de servidores virtuales o granja de servidores de bajo costo que trabajan en conjunto para balancear la carga entre ellos.

El balanceo de carga consiste en compartir la carga de trabajo y tráfico de los clientes que éstos acceden. Al grupo de servidores que prestan este servicio se los conoce como servidores virtuales.

Al balancear la carga se mejora el tiempo de respuesta, acceso y confiabilidad. La caída de un servidor no influye en el funcionamiento de todo el Clúster ya que las funciones de éste son asumidas por el resto de servidores virtuales. Cuando la carga de trabajo sea mayor se pueden añadir más servidores al Clúster y escalar.

Clúster tolerante a fallos

En un sistema tolerante a fallos, cuando se produce un fallo hardware, el hardware asociado a este tipo de sistema es capaz de detectar el subsistema que falla y obrar en consecuencia para restablecer el servicio en segundos (o incluso décimas de segundo).

Clúster de alta confiabilidad (*hr, high reliability*)

Clúster caracterizado por ofrecer una alta confiabilidad al sistema. La idea es obtener respuestas eficientes del sistema a pesar de tener una sobrecarga de las capacidades de un servidor. Estos *Clústers* se caracterizan por ejecutar un mayor número de tareas en el menor tiempo posible.

Planificación y administración de clúster

Administración

Luego del proceso de instalación del hardware, configuración del Clúster, se necesita que estos recursos sean administrados para tener un control de rendimiento y funcionamiento del Clúster.

La administración de un Clúster es de mucha importancia en el funcionamiento del mismo, ya que es fundamental administrar adecuadamente los recursos del Clúster y con ello poder detectar fallos a nivel de software y hardware, además poder monitorear el rendimiento de cada servidor con la finalidad de constatar su correcto funcionamiento y en caso de tener algún inconveniente poder tomar medidas preventivas y evitar que estos colapsen.

Para poder lograr una administración adecuada de un Clúster se debe considerar los siguientes aspectos:

Registro de eventos (*logging*)
Falla y Recuperación de Hardware.
Falla de Software
Falla y Recuperación del Sistema de archivos.
Encolamiento (*Queing*)
Planificación (*Scheduling*)
Monitoreo (*Monitoring*)
Contabilidad (*Accounting*)

Existe una relación entre usuarios, recursos y las actividades involucradas en la administración del Clúster. El software de administración se ubica entre los usuarios y los recursos (servidores reales y director) del Clúster. En primera instancia los usuarios

envían sus peticiones a una cola que almacena los trabajos que se van a llevar a cabo (el usuario puede en cualquier momento pedir información sobre el estado de ese trabajo). Los trabajos esperan en la cola hasta el momento en que ingresan al Clúster para ser atendidos. Mientras todo está ocurriendo, el sistema de administración esta monitoreando el estado de los recursos del sistema y cuáles usuarios los están utilizando.

A continuación se va analizar con más detalle las actividades que se deben llevar a cabo en un sistema de administración:

Registro de eventos (*logging)*

El registro de eventos es el proceso por el cual todos los aspectos relacionados al funcionamiento de una máquina y operación del Clúster se guardan en un archivo de registro (logs) el cual puede ser usado para futuras consultas.

El administrador de un sistema Linux tiene a su disposición un programa de generación de registros que monitoriza todos los eventos que se producen en el mismo. Syslog guarda, analiza y procesa todos los archivos de registro sin requerir apenas intervención por parte del administrador. La ubicación del directorio donde se almacenarán los registros depende del servicio instalado, por lo general el directorio donde se almacena es el /var/log.

Falla y recuperación de hardware

Una de las principales responsabilidades de la administración de un Clúster es la falla a nivel de hardware. El impacto que ocasiona la falla del hardware puede conllevar a tener todo el sistema inoperante. Estos impactos pueden ocasionar pérdidas de los servidores de archivos, servidores de red, dispositivos de interconectividad, entre otros, siendo importante la necesidad de tener monitoreado el funcionamiento de estos y a la vez tomar medidas de prevención.

Se debe tener presente que muchos fallos a nivel de hardware que se presentan no afectan mucho al rendimiento general del Clúster, como es el caso de la falla del disco de un servidor, éste dejaría de operar y afectaría únicamente a los cliente que a él se conectan, pero en términos generales no afectarían a todo el sistema ya que el Clúster de por si se encarga de suplir ese fallo, pero a fin de cuentas esa falla si afecta al rendimiento del sistema, en especial al tiempo de respuesta de los servidores operativos en el Clúster.

Para evitar un gran impacto de estos inconvenientes y poder tener una recuperación ante fallos del hardware, la administración debe considerar varios aspectos como el aislar el componente fallido con la finalidad de que puedan interferir con el funcionamiento del resto de componentes del Clúster, tener respaldos (*backups*) a nivel de hardware para poder suplantar de manera rápida el componente fallido y así poder garantizar la disponibilidad del sistema.

Falla de software

La falla ocasionada por el software al igual que los fallos a nivel de hardware son críticos debido a que pueden dejar inoperante al Clúster, pero a la vez estas fallas involucran otros aspectos que deben ser tomados en cuenta. Las fallas de software muchas veces tienen arreglo, otras veces no, pero es de vital importancia detectarlas, para poder analizar las posibilidades de evitar que vuelvan a ocurrir, por lo general el Linux los errores se los evita con los denominados parches del sistema, los cuales vienen a suplir las fallas del sistema a nivel de software.

Para que el sistema tenga un funcionamiento adecuado y a su vez poder evitar los problemas de indisponibilidad que cada falla acarrea, se debe tomar en consideración varios aspectos tales como: la actualización del sistema debe hacérsela de manera continua, una vez que algún parche actualizado haya sido desarrollado. Realizar pruebas de funcionamiento del sistema posterior a la actualización del mismo, para poder de esta

manera constatar las debilidades que mencionada actualización trajo al sistema y si no las resuelve, tener la posibilidad de volver a las versiones de software anteriores.

Falla y recuperación del sistema de archivos.

La falla en un sistema de archivos es muy crítica, se podría decir que es la pérdida que mayor daño ocasionaría a un sistema, ya que se perdería toda la información almacenada por varios meses o años, en especial se perdería los datos de usuario y de las aplicaciones que hacen uso de éste.

Por ello es de vital importancia tomar medidas preventivas que logren evitar tener un punto de fallo en el sistema de archivos, teniendo sistemas de archivos redundantes mediante el uso de técnicas como RAID, poseer un sistema de archivos con *journaling* el cual permita recuperar rápidamente los datos en caso de existir alguna falla inesperada en el sistema, poseer un sistema de archivos paralelo, con lo cual la pérdida de un servidor de archivos no influya de manera total al funcionamiento general del sistema ya que la existencia de otros servidores de archivos podrían suplir su ausencia.

Encolamiento (*queing*)

Un aspecto a tomar en cuenta en la administración de un Clúster es el encolamiento (*Queing*) que consiste en el proceso de acumular los trabajos para que sean ejecutados por el conjunto de recursos disponibles. Las tareas y los trabajos que los usuarios desean realizar, son presentados por el sistema de administración como un conjunto llamado grupo de trabajo.

Este grupo de trabajo para ser ejecutado necesita dos aspectos fundamentales que el sistema debe proveer, en primer lugar proveer los recursos necesarios (como la cantidad de memoria o CPU´s necesitados), y en segundo lugar una descripción de las tareas a ser

ejecutadas (archivo a ejecutar, datos requeridos para procesar cualquier petición; entre otros).

El grupo de trabajo luego de presentado al sistema de administración, es colocado en una cola hasta que el sistema provea los recursos necesarios (por ejemplo: la cantidad correcta de memoria y CPU requerido) para procesar cualquier trabajo encomendado. El tiempo de espera para que los trabajos se ejecuten depende exclusivamente del tiempo de ocupación de los recursos.

Monitoreo (*monitoring***)**

El monitoreo involucra el observar que el rendimiento y funcionamiento del Clúster sea correcto. Una operación correcta implica tener a todos los recursos de hardware y software monitoreados y con ello constatar que el funcionamiento sea el esperado.

El tener monitorizado un sistema es muy importante en el proceso de administración del Clúster ya que mediante los datos que arroje dicho monitoreo se puede prever cual es el comportamiento de un determinado dispositivo que compone el Clúster, y con ello poder tomar los correctivos necesarios para evitar una caída del sistema.

Es indispensable chequear parámetros de los equipos cuando estén en funcionamiento, parámetros como: arquitectura y frecuencia del CPU, tipo y versión del sistema operativo, memoria total, número de CPU´s en cada servidor, tiempo de actividad, porcentajes de uso de CPU, utilización de disco, memoria disponible, actividad de procesos, entro otros; y de esta manera constatar su correcta operación.

Contabilidad (a*ccountin*g)

Mecanismo utilizado para la recolección de datos de cada grupo de trabajo que se ejecuta en el Clúster, el *accounting* es una herramienta muy importante que permite

generar información útil para el proceso de planificación de tareas, esta herramienta puede ser utilizada para varios propósitos tales como:}

Elaboración semanal de reportes de utilización del sistema.
Elaboración de reportes de utilización mensual de recursos por usuario.
Elaboración de un cronograma de políticas de planeamiento.
Prever requerimientos computacionales futuros.
Determinar áreas que deben ser mejoradas dentro del sistema.

Planificación de tareas

La planificación de tareas es una actividad de mucha utilidad en la administración de un Clúster, ya que permite programar un conjunto de actividades que se ejecutarán de acuerdo al tiempo, necesidad y circunstancia que un determinado sistema así lo requiera. Al hablar de tarea se hace referencia a un programa ejecutable que realiza determinadas funciones. Un proceso consiste en un número de tareas con cierta independencia que se coordinan para cumplir funciones lógicas.

.

Un planificador de tareas debe garantizar tres aspectos fundamentales:

Rendimiento. Optimizar la ejecución del número de tareas y procesos.
Tiempo de respuesta. Conseguir que el tiempo de ejecución de un determinado proceso sea mínimo
Optimización de CPU. Optimización constante de la carga de proceso de la CPU.

Este tipo de herramientas permiten la ejecución automática de tareas, esto mediante la ejecución de comandos los cuales se ejecutarán dependiendo del tiempo y las necesidades que se presenten.

En el caso de la planificación de tareas multiplataforma (desde Linux pasando por Unix o Windows), el principal objetivo es gestionar tantas tareas como sea posible de manera más rápida posible y con el menor número de anomalías.

Una solución de planificación de tareas debe ser capaz de interrumpir, parar o reiniciar tareas, así como el poder asignar prioridades a la ejecución de las mismas. También es importante la existencia de puntos de verificación, los cuales van a proporcionar una imagen del estado actual de la tarea en ejecución, de igual manera la tarea puede continuar su ejecución desde el punto de verificación.

Los planificadores de tareas completos y modernos pueden combinar múltiples tareas en grupos cuando lo necesitan y procesarlo como una única unidad cuyo resultado será utilizado como condición de inicio para otras tareas o grupos de tareas.

HADOOP

El marco HADOOP abarca un gran número de componentes de software de código abierto, con un conjunto de módulos básicos para la captura, procesamiento, gestión y análisis de grandes volúmenes de datos, que está rodeado por una gran variedad de tecnologías de soporte. Los componentes básicos incluyen:

El sistema de archivos distribuido HADOOP (HDFS), que es compatible con un directorio jerárquico y sistema archivos convencional que distribuye los archivos a través de los nodos de almacenamiento (es decir, *DataNodes*) en un clúster HADOOP.

MAPREDUCE, un marco modelo de programación y ejecución para el procesamiento paralelo de aplicaciones por lotes.

YARN (corto para el gracioso Otro Negociador de Recursos Más en inglés), que gestiona la planificación de trabajos y asigna los recursos del clúster a

aplicaciones en ejecución, arbitrando entre ellas cuando hay competencia por los recursos disponibles. También rastrea y monitorea el progreso de los trabajos de procesamiento.

HADOOP Common, un conjunto de librerías y utilidades utilizadas por los diferentes componentes.

Esas piezas centrales y otros módulos de software están puestos en capas en la parte superior de una colección de nodos de hardware de computación y almacenamiento de datos. Los nodos están conectados a través de una red interna de alta velocidad para formar un sistema paralelo de alto rendimiento y de procesamiento distribuido.

MOSIX

MOSIX es una extensión del *Kernel* de *Linux* (*Linux Kernel Patch*) el cual está dirigido a la computación de alto rendimiento, esto permite ejecutar aplicaciones normales (no paralelizadas) en un *clúster*. Una de las principales utilidades de MOSIX es la migración de procesos, un proceso puede iniciar en un nodo y posteriormente migrar y ejecutarse de manera trasparente en otros nodos.

El proceso de migración se realiza para optimizar el rendimiento global. La migración de un proceso se produce de forma automática y transparente en respuesta a la disponibilidad de los recursos. Este proceso de migración se repite según sea necesario entre los distintos nodos que componen el clúster, incluso puede volver al nodo que inició el proceso si las condiciones lo permiten.

Características principales MOSIX

Las características están destinadas a proporcionar a los usuarios y a las aplicaciones la impresión de que se ejecutan las tareas en un único nodo con varios procesadores.

Proporciona una imagen única del sistema (SSI).

Los usuarios pueden iniciar sesión en cualquier nodo y no necesitan saber dónde se están ejecutando sus aplicaciones.
No es necesario modificar las aplicaciones, ni enlazarlas a ninguna biblioteca en especial.
No es necesario copiar los archivos en los nodos remotos.

Detección automática de los recursos y distribución de la carga de trabajo.

Balanceo automático entre los procesos que han migrado.
Migración de procesos desde los más lentos a los nodos más rápidos.
Migración de procesos que necesitan más memoria de la disponible en un nodo.

Comunicación directa entre los procesos migrados.

Proporciona un entorno de tiempo de ejecución seguro para los procesos huésped.

Compatible con arquitecturas de x64 y x86.

Incluye herramientas para la instalación y configuración de manera automática.

Incluye monitoreo en línea.

2.2 MARCO METODOLÓGICO

2.2.1 Forma de investigación

La forma de investigación se considera de tipo aplicada, debido a que se basa en el estudio y aplicación de la investigación a problemas concretos, en circunstancias y características concretas (Tamayo y Tamayo, 2003). Ya que el objetivo de este proyecto es resolver el problema existente por la falta de equipos que permitan procesar una gran cantidad de cómputos en menor tiempo, beneficiando de esta manera a la comunidad universitaria en el desarrollo de sus investigaciones.

2.2.1.1 Tipo de investigación

Esta investigación es descriptiva, porque busca únicamente describir situaciones o acontecimientos; no está dirigida a comprobar explicaciones, ni probar determinadas hipótesis (Tamayo y Tamayo, 2003). Su finalidad es aportar una eficaz herramienta de cómputo y almacenamiento con la cual se podrá minimizar el tiempo de ejecución de los cálculos realizados en las diversas áreas de investigación de la UDO-NS.

2.2.1.2 Diseño de la investigación

El diseño de esta investigación es de campo porque los datos se recogieron directamente de la realidad (Tamayo y Tamayo, 2003), es decir, se aplicaron técnicas para la recolección de datos como entrevistas, cuestionarios que permitieron obtener la información necesaria para el desarrollo del prototipo de *clúster*.

Técnicas para la recolección de datos

En la recolección de la información necesaria para desarrollar esta investigación se utilizó las técnicas de consultas bibliográficas y consultas en Internet, lo cual permitió establecer el soporte teórico de la investigación.

2.2.1.3 Metodología del área aplicada

Para la realización de este trabajo se tomó como referencia el modelo incremental propuesto por Harlan Mills (1980), por lo cual planificar un proyecto con esta metodología es a menudo imposible, debido al grado incertidumbre con el que inicio, además se desconocía el número de iteraciones que iban a ser necesarios. Estas cuyas fases se adaptaron para poder implementar el prototipo de clúster de alto rendimiento. A continuación se describirá el modelo planteado:

El modelo incremental, propuesto por Harlan Mills, está basado en la idea que el sistema se particiona en subsistemas según sus funcionalidades. Al dividirlo se puede manejar cada funcionalidad por separado e ir integrándolas para formar el sistema final.

El modelo aplica secuencias lineales de manera escalonada conforme avanza el tiempo en el calendario. Cada secuencia lineal pasa por las siguientes fases:

Análisis y requerimientos
Diseño
Implementación
Pruebas

Estas secuencias producen lo que se denomina: incremento. El primer incremento es a menudo denominado "núcleo" y estará formado por aquellas funcionalidades más importantes.

Durante el desarrollo, se puede llevar a cabo un análisis adicional para los requisitos posteriores, pero no se aceptan cambios de los mismos para el incremento actual. Una vez que un incremento se completa, este es integrado al sistema y se le entrega al cliente un producto operacional puede poner en funcionamiento. Esto significa que tienen una entrega temprana de parte de la funcionalidad del sistema. Pueden experimentar con el sistema, lo cual les ayuda a clarificar sus requisitos para los incrementos posteriores.

Análisis de Requerimientos

En esta etapa, se logra claridad sobre lo que desea el usuario y la forma en la cual se le va a presentar la solución que está buscando. Esta fase se realiza por cada iteración que se realice.

Diseño

Determinar una implementación efectiva y eficiente que realice las funciones y que cumpla con los requerimientos dados por el cliente. Las siguientes actividades que se plantean en esta etapa.

Determinar la arquitectura inicial
Determinar el diseño lógico
Implementación física

Implementación

En esta etapa, se instalan y configura cada aplicación, se definen desde paradigma hasta herramienta.

Pruebas

Esta es la etapa donde se verifica cada componente del prototipo para encontrar los errores y corregirlos, como se muestra en la figura 3.

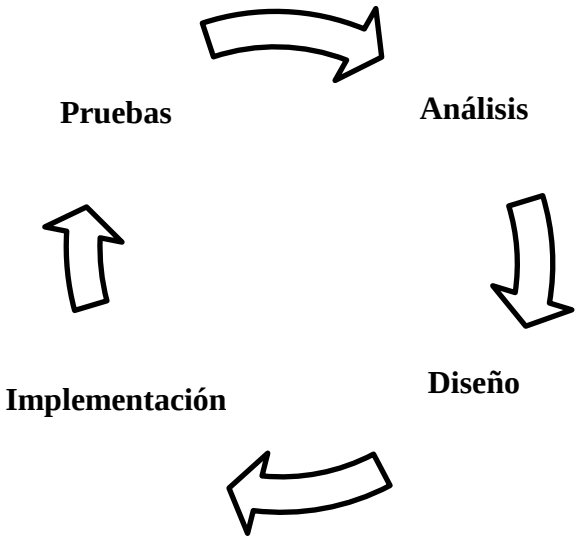

Figura 3. Fases de la metodología

Sin embargo, para la implementación, se usa el principio de trabajo en cadena o Pipeline. Con esto se mantiene al cliente en constante contacto con los resultados obtenidos en cada incremento. Es el mismo cliente el que incluye o desecha elementos al final de cada incremento a fin de que el software se adapte mejor a sus necesidades reales. El proceso se repite hasta que se elabora el producto completo. De esta forma el tiempo de entrega se reduce considerablemente.

Durante el proceso se trata de llevar a cabo al proyecto en diferentes partes que al final terminará siendo la solución completa requerida por el cliente, pero éstas partes no se pueden realizar en cualquier orden, sino que dependen de lo que el cliente este necesitando con más urgencia, de los puntos más importantes del proyecto, los

requerimientos más básicos, difíciles y con mayor grado de riesgo, ya que estos se deben hacer al comienzo, de manera que se disminuya la dificultad y el riesgo en cada versión.

Para el desarrollo de esta investigación se caracterizó este modelo de la siguiente forma:

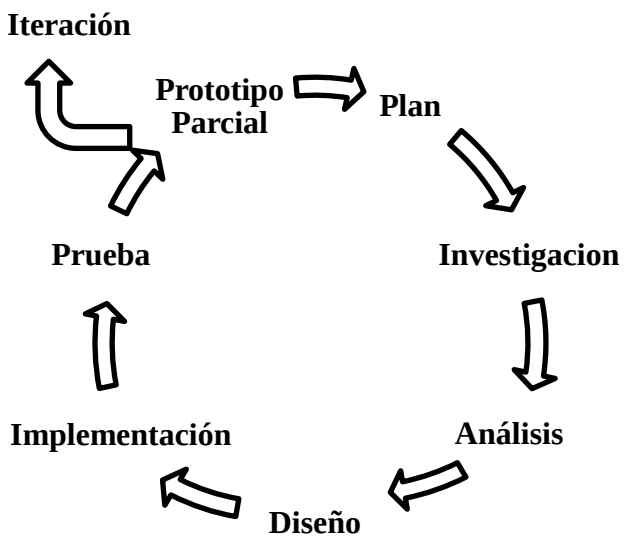

Figura 4. Adaptacion de la metodologia

Plan

Los requerimientos detallados del prototipo son identificados.

Investigación

Establecer objetivos de investigación, justificar la investigación, establecer un conocimiento previo

Análisis

Para efectos de este trabajo la fase análisis de requerimiento, Se basó en identificar, capturar y comprender las necesidades del *clúster*, servicios, aplicaciones y dispositivos

que conforman el sistema y sus características (interactividad, confiabilidad, seguridad, calidad, adaptabilidad, factibilidad, crecimiento esperado entre otros) para enfocar y mejorar su rendimiento.

La fase de implementación para adaptarse con las necesidades requeridas del prototipo se realizó una estrategia de implementación, la propuesta consiste en instalar, configurar e integrar todos los componentes del sistema. Para completar la estrategia es necesario realizar las siguientes actividades:

Investigación preliminar y cobertura de la curva de aprendizaje
Instalar y configurar el sistema operativo.
Instalar y configurar el sistema de archivos para clústeres o el administrador de sincronización de datos.
Instalar y configurar el administrador de recursos.
Instalar y configurar los servicios y/o aplicaciones.

CAPÍTULO III. DESARROLLO

En este capítulo se explica en detalle cada una de las actividades realizadas en la implementación del clúster de acuerdo a las etapas de la metodología antes citadas, Para dar inicio a la solución del problema se planificaron una serie de actividades esenciales, ofreciendo una visión general del proyecto, lo cual sirvió para concretar los objetivos del mismo, en función de estos se planificaron cuatro (4) iteraciones, tomando en cuenta que cada una de estas generara un incremento que representa cada nivel de operatividad del clúster, finalmente estos incrementos se integran para completar la solución esperada.

3.1 PRIMERA ITERACIÓN

Esta iteración fue una fase exploratoria y de descubriendo a nuevas tecnología donde se cubrió una ardua investigación sobre el tema, así como también un reconocimiento de nuevas herramientas y técnicas necesarias para la construcción del *Clúster* CISC.

3.1.1 Análisis

Se realizó revisión bibliográfica en instituciones de educación superior. Como también los sistemas distribuidos elaborados en la universidad. Se contó con una ayuda externa de diferentes universidades en el extranjero aportando asesoría y facilitando documentación con temas semejantes a lo que se requería elaborar.

Durante esta fase también se analizaron diferentes tecnologías existentes con el fin de seleccionar la que mejor se adaptara a las necesidades de la organización en función a los dispositivos tecnológicos que se encuentran disponibles en el Departamento de Licenciatura en Informática de la UDO Núcleo Sucre (*switches*, servidores, computadoras) y además que la tecnología seleccionada permita aprovechar los recursos de cómputo que en la actualidad se mantienen ociosos dentro del Núcleo.

Se procedió a estudiar la estructura y configuración del servidor MPI que se encontraba dentro de CISC (primer prototipo elaborado para el estudio de computación en paralelo) instalado para las asignaturas Fundamentos de Programación Paralela (230-4254) y Algoritmos Distribuidos (230-4214). Se observó que el servidor estaba conformado por solo un nodo y herramientas como OPENMPI, un compilador C y servidores virtuales, el servidor estaba destinado para el estudio y ejecución de programas en paralelo.

Entre los dispositivos disponibles en CISC para la construcción del *clúster* se encuentran:

2 computadores SIRAGON 1320

- Procesador intel pentium dual core 2 ghz
- Memoria ram 1g 555 x2
- Disco duro de 160gb

1 computador

- Procesador intel pentium dual core 2 ghz
- Memoria ram 1g 555 x2
- Disco duro de 160gb

1 *Switch Catalyst* 3560 24 puertos Ethernet y 4 puertos *GigaEthernet* a 1Gbps

Una vez analizadas las características de los equipos disponibles, se llegó a la conclusión de que el sistema operativo base a utilizar en la implementación del clúster es *Debian* en su versión 8 denominada "*Jessei*", ya que cuenta con soporte constante por parte de la comunidad de linux para la distribución, además cuenta con actualizaciones y eliminación de dependencias de otras librerías al momento de instalar paquetes, soporta múltiples arquitecturas como *alpha, amd64, armel, hppa, i386, ia64, mips, mipsel, powerpc, s390, sparc.*

Debían ofrece estabilidad, es rápido y ligero en memoria, los controladores para la mayoría del hardware son desarrollados por usuarios GNU/Linux, ofrece software de

seguridad que permite enviar correo entre usuarios preservando su privacidad. Los esquemas de cómputos analizados para el desarrollo de este proyecto se ven reflejados en la tabla 2. Se presenta:

Tabla 2. Esquema de cómputos analizados.

Esquema de Cómputo Requiere	*Grid Computing*	*Cloud Computing*	*Clúster Computing*	*Volunteer Computing*
Virtualización por hardware	Si	Si	No	No
Hardware homogéneos	No	No	Si	No
Hardware heterogéneos	Si	Si	No	Si
Multiplataforma (S.O.)	Si	Si	No	Si
Software propietario	Si	Si	No	No
Software GPL (libre)	Si	Si	Si	Si
Escalabilidad simple	No	No	No	Si
Redes de alta capacidad	Si	Si	No	No
Redes de rendimiento estándar	No	No	Si	Si
Certificación	Si	Si	No	No
Autenticación	Si	Si	Si	Si

El esquema de computación que mejor se adaptó a los dispositivos que se encuentran disponibles en CISC, fue El *Clúster Computing*, ya que requiere un hardware

homogéneo que esté 100% dedicado al procesamiento y almacenamiento de tareas.

Después del análisis de los equipos existentes, se determinó como requerimiento la elaboración de ciertas tareas, tales como, la instalación del sistema operativo *Debian* en todos los nodos, establecer un protocolo SSH para la comunicación entre los nodos y a su vez configurarlo para que se comuniquen sin claves, como también un sistema de archivos que para este caso se implementó un servidor NFS para pasar los archivos entre los nodos y por ultimo instalar y configurar una herramienta que contenga librerías MPI para la ejecución de código en paralelo.

3.1.2 Diseño

En base al análisis realizado se decidió utilizar una arquitectura cliente-servidor. Esta implementación del *clúster* HPC fue establecida para tres computadoras pertenecientes al CISC conformada por un nodo MAESTRO capaz de monitorear y ejecutar los programas en paralelo, como también de controlar la sobrecargar de los core´s de cada nodo esclavo y dos nodos ESCLAVOS los cuales están dedicados al procesamiento de datos o a ejecutar operaciones aritméticas. En este apartado se realizó el diseño de topología de red propuesta, para la implementación de la función de arquitectura clúster, así como la configuración con un direccionamiento IP estático en el laboratorio, con la finalidad de sustituir el direccionamiento dinámico DHCP, para evitar reconfiguraciones posteriores y establecer una comunicación mediante SSH.

La implementación del clúster se llevó a cabo con el sistema operativo GNU/Linux Debian “Jessie” por su estabilidad y su funcionamiento como servidor de alto rendimiento.

En función a la arquitectura de cliente-servidor, se muestra en la figura 4, la misma cuenta con la siguiente configuración.

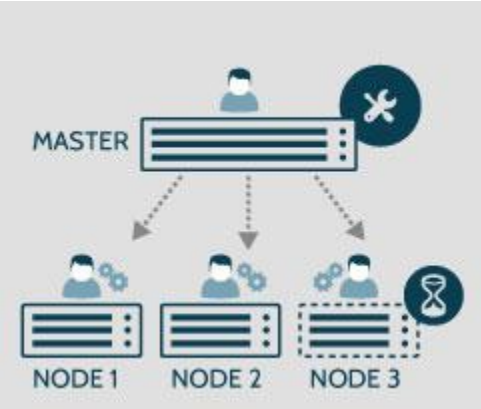

Figura 5. Arquitectura cliente servidor

El protocolo SSH (interprete de órdenes seguro: *Secure Shell*) fue diseñado pensando en la seguridad y la confiabilidad. Las conexiones que utilizan SSH son seguras, ya que se cifran todos los datos intercambiados. Para ello se instaló este protocolo para poder invocar comandos remotamente desde el nodo maestro hacia lo nodos esclavos. Se instaló el protocolo en cada uno de los nodos que integran la arquitectura clúster.

El sistema de archivos del *clúster* se contempla en un servidor *NFS*, para esto se requiere emplear el arranque desde la red y no utilizar el disco rígido de los nodos, de este modo lograr que el *clúster* represente la menor invasión posible en el ambiente del laboratorio y la mayor independencia posible de sus administradores. Para la programación paralela se empleó *OPENMPI*, es un estándar de funciones para llevar a cabo el pasaje de mensajes en la programación paralela. Puede invocarse desde *C* y *Fortran*, una función de pasaje de mensajes se encarga simplemente de transmitir datos de un proceso a otro a través de la red. *OPENMPI* es una de las tantas implementaciones de *MPI* que actualmente se encuentran disponibles.

3.1.3 Implementación

En esta fase se elaboró una de las tareas la cual era requerida como requisito principal para el funcionamiento e inicio de la configuración del *clúster*, se comenzó con la instalación y configuración del sistema operativo *GNU/Linux Debían 8* “Jessie” y

seguido a esto se establecieron las *IP* estáticas correspondientes a cada nodo, como también añadir el servidor *DNS* y repositorios adecuados.

Para la configuración, se implementó un direccionamiento *IP* estático para los ordenadores en el laboratorio (Direccionamiento IP clúster CISC), detalla el rango de *DIRECCIONES IP* estáticas utilizadas, desde la dirección 150.186.92.185 hasta 150.186.92.183, como también se le asignó un nombre característico a cada nodo para simplificar la forma de invocarlos, se puede observar en la tabla 3.

Tabla 3. *DIRECCIONES IP*.

Dirección de Red	Máscara de Subred	Gateway	Nodo
150.186.92.185	255.255.255.0	150.186.92.185	cesar
150.186.92.184	255.255.255.0	150.186.92.184	mau
150.186.92.183	255.255.255.0	150.186.92.183	koba

3.1.3.1 Instalación y configuración del sistema operativo del clúster

Luego de la selección del sistema operativo Debian 8 “Jessie” 64bits, se realizó la instalación y configuración en cada uno de los nodos que conforman la arquitectura, también se implementó un servidor SSH para una comunicación segura entre los nodos , definiendo así las dependencias que se instalaron en el maestro y los esclavos, al mismo era necesario como requerimiento para la fácil administración, tener una interfaz gráfica en el nodo maestro para el fácil manejo de algunas herramientas, se puede ver con mas detalle en el apéndice A.

3.1.4 Pruebas

Una vez instalado y configurado el sistema operativo en cada uno de los nodos se hicieron pruebas funcionalidad como arranque inicial de sistema operativo, pruebas de red local, conexión mediante el protocolo SSH y pruebas de conexión a internet las cuales fueron exitosas, no se logró hacer ningún otro tipo de pruebas debido a que se

desconoce totalmente el entorno y se basó en investigación de las herramientas que se iban a usar para la implementación del *clúster*.

3.2 SEGUNDA ITERACIÓN

3.2.1 Análisis

Con lo anteriormente analizado se seguirá con la implementación del *clúster* HPC siguiendo la arquitectura de *clúster* de alto rendimiento. Para entender el funcionamiento de dicha arquitectura fue necesario definir las funciones de cada una de las capas, dando como resultado lo siguiente.

3.2.1.1 Capa de aplicación

Permite la ejecutar aplicaciones secuenciales, aplicaciones paralelas (desarrolladas específicamente para la ejecución en el clúster) y ambiente de programación paralela (permite implementación de algoritmos que hacen uso de recursos compartidos*: CPU*, memoria, y servicios*).*

3.2.1.2 Capa de *middleware*

Es el software que generalmente actúa entre el sistema operativo y las aplicaciones con la finalidad de proveer:

Librerías MPI, enlaza a los compiladores con el ambiente de programación paralela que se encuentra en la capa de aplicación.

Herramientas del clúster, permiten su correcto funcionamiento: migración de procesos, *checkpoint-restart* (detener y restaurar varios procesos migrarlos a otro nodo) balanceo de carga entre los nodos, tolerancia a fallos, etc.

3.2.1.3 Sistema operativo

Autoriza a los usuarios acceso unificado a los recursos del sistema.

3.2.1.4 Red de alta velocidad

Permite la comunicación entre los distintos nodos del clúster con tecnología de alta velocidad (Gigabit).

Adicionalmente se especificaron las actividades de cada una de las aplicaciones que se iban a instalar en el clúster, también se acordaron las herramientas usadas y analizadas en la iteración anterior para la construcción del mismo.

La implementación del clúster de alto rendimiento "*CLUSTER CISC*" comprendió algunas actividades que se detallan a continuación**,** cabe desatacar que dependiendo del rol del nodo la configuración para cada tipo de nodo (nodo Maestro y nodo Esclavo) es diferente en algunas herramientas:

Asignación de los nodos al archivo host.
Instalación y configuración del protocolo Secure Shell (SSH).
Instalación y configuración del protocolo Network File System (NFS).
Instalación y configuración de OPENMPI.

3.2.2 Implementación

Se procedió a la instalación de los paquetes *OPENSSH, NFS, OPENMPI* para cada tipo de nodo, cabe destacar que en algunos casos el nodo MAESTRO se configura de manera distinta a los nodos ESCLAVOS.

Para comenzar primer se editó el archivo */etec/hosts* para asignarle los nombres a cada

nodo dentro del *clúster* y así definir el grupo de trabajo para invocarlos por sus nombres y no por las IP´s (se usó un súper-usuario para poder editar este archivo).

```
#nano /etc/hosts

150.186.92.185 cesar
150.186.92.184 mau
150.186.92.183 koba
```

3.2.2.1 Instalación el protocolo SSH

Se utilizó este protocolo para poder invocar comandos remotamente desde el nodo maestro hacia lo nodos esclavos. Para ello se instaló el protocolo en cada uno de los nodos que integran la arquitectura clúster, ejecutado en una terminal como usuario ***root,*** el siguiente comando.

```
# apt-get install SSH
```

3.2.2.2 Configuración

Posteriormente se configuro el protocolo SSH para poder logar y ejecutar comandos remotamente sin necesidad de introducir cada vez la contraseña, esto se realiza generando una clave pública con el protocolo SSH, el cual hace uso del algoritmo de cifrado RSA. Esto es importante para que el nodo maestro pueda invocar comandos remotamente en cada uno de los nodos esclavos. (Generación de la clave pública *SSH* en el nodo maestro) muestra el proceso para generar la clave pública.
Generación de la clave pública *SSH* en el nodo maestro:

```
# SSH-keygen –t rsa
```

Se debe tener en cuenta que la clave se genera tecleando el comando desde el directorio personal ***home*** del usuario ***clúster*** y no como ***usuario root.*** En la opción ***Enter passphrase*** se deja vacío y presionando la tecla [Enter], esto es importante para poder acceder remotamente a cada uno de los nodos sin introducir la contraseña. (Añadir clave pública al fichero de claves autorizadas) se observa el paso para añadir la clave pública al fichero de claves autorizadas ***authorized_keys***, para ello ingresamos al directorio ***/home/clúster/.SSH*** donde se generó la clave.

```
# cat id_rsa.pub >> authorized_keys.
```

se copia de manera segura, la clave pública generada en el nodo maestro hacia los nodos esclavos [nodo1-nodo2].

```
# scp authorized_keys fnd@150.186.92.185/home/fnd/.SSH/
```

De esta manera se accede mediante el protocolo SSH al koba (en este caso con la dirección IP: 150.186.92.183). Este proceso debe realizarse desde el nodo maestro hacia cada uno de los nodos que componen el clúster nodo1-nodo3, para que el nodo maestro pueda invocar comandos remotamente a dichos nodos.

3.2.2.3 Instalación y configuración del protocolo Network File System (NFS)

NFS es un protocolo a nivel de aplicación del modelo OSI, que es utilizado para sistemas distribuidos en un entorno de red local. Se este protocolo para que los usuarios de los 3 nodos que componen el clúster puedan compartir el directorio personal por medio de la red. Para ello: hay que instalar en el nodo maestro el servidor NFS. Instalación del servidor NFS:

```
# apt-get install NFS-kernel-server NFS-common
```

En los nodos esclavos el cliente NFS (Instalación del cliente *NFS*), para que así el nodo maestro pueda compartir su directorio ***home*** a los nodos esclavos koba y mau:

```
# apt-get NFS-common
```

Se crear del directorio el cual se compartirá en todo los nodos

```
# mkdir nombre_directorio
# chmod 755 nombre_directorio
```

Se edita el fichero ***/etc/exports*** del nodo maestro, el nodo que actúa como servidor NFS, añadiendo al final del archivo las dos últimas líneas.

```
# nano /etc/exports
```

Se edita en el nodo maestro de la siguiente manera:

```
/home/fnd/nombre_directorio mau(rw,ro) koba(rw,ro)
```

El modo en que se comparte (solo lectura 'ro' o lectura y escritura ‘rw’). Una vez realizado el paso anterior, se inicia el servidor (como usuario ***root***)

```
# /etc/init.d/NFS-kernel-server start
```

De esta manera el directorio ***/home/fnd*** del nodo maestro se montará en los nodos esclavos mediante la red con permisos de lectura y escritura.
Lo siguiente es añadir en el fichero ***/etc/fstab*** de cada uno de los nodos esclavos: koba-mau. *(*Montar el directorio del nodo esclavo al nodo maestro*)*.

```
#nano /etc/fstab
```

```
Cesar:/home/fnd/nombre_directorio
/home/fnd/nombre_directorio  NFS    rw,sync,hard,  intr      0
0
```

3.2.2.4 Instalar OPENMPI

Es necesario tener instalado un compilador de C/C++ ANTES de realizar la instalación de *OPENMPI.*

```
# apt-get install -y autotools-dev g++ build-essential
```

Luego se instalan los paquetes de OPENMPI.

```
# apt-get install OPENMPI-bin OPENMPI-common libOPENMPI1
libOPENMPI-dev
```

3.2.3 Pruebas

Se hicieron pruebas de conectividad. Se chequea si existe conexión con el nodo koba desde el nodo maestro, se puede observar en la figura 5.

```
# ping koba
```

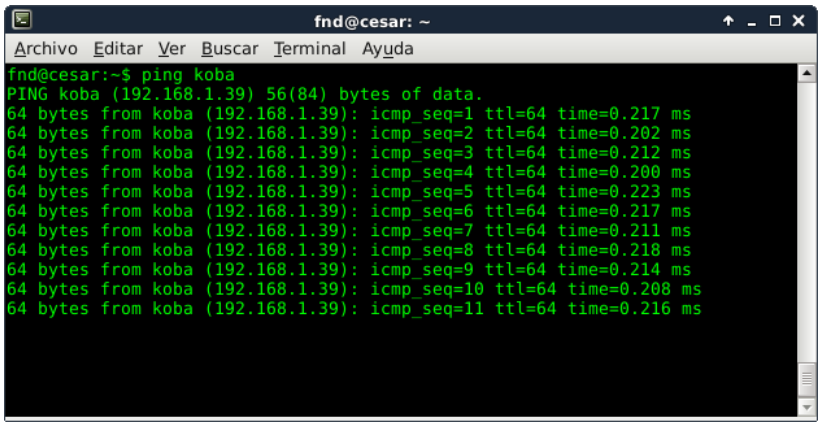

Figura 6. Prueba de convertibilidad

Se procede a conectar el nodo koba, se muestra en la figura 6.

```
# SSH koba
```

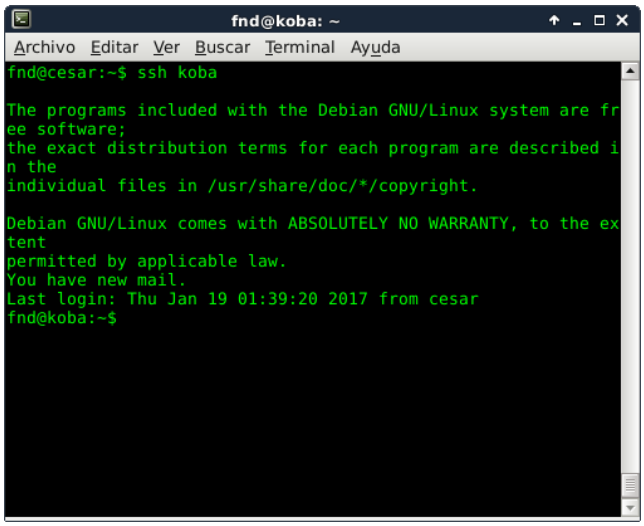

Figura 7. Prueba SSH nodo maestro-esclavo

Se verifica que existe conexión entre todos los nodos y se elaboran pruebas con el sistema de ficheros NFS. Dirigirnos al fichero /home/fnd/babana, como se muestra en la figura 7.

```
# cd /home/fnd/banana
```

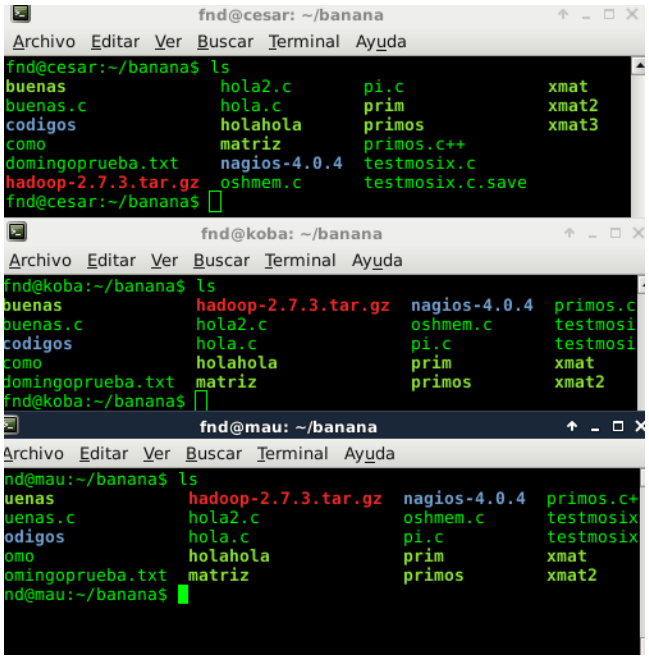

Figura 8. Prueba NFS

Se puede observar el fichero replicado en los todos los nodos del *Clúster*. Ya hecho esto se puede comenzar con la pruebas usando OPENMPI, se muestra en la figura 8.

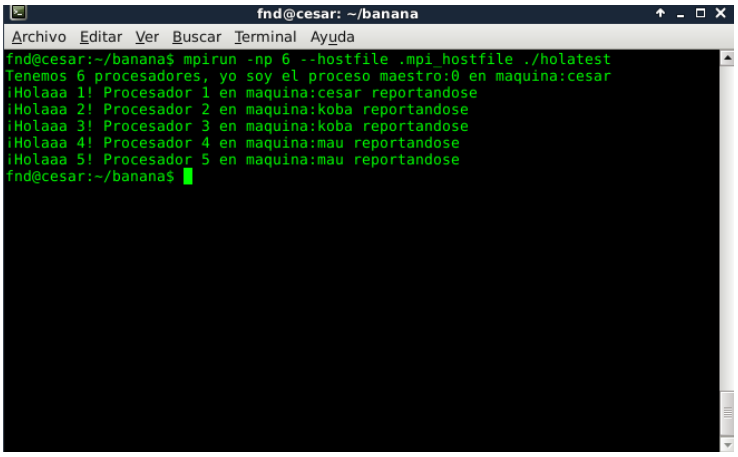

Figura 9. Ejecución de la aplicación con OPENMPI

Se observa el comportamiento del clúster utilizando librerías de paso de mensajes (MPI)

3.3 TERCERA ITERACIÓN

3.3.1 Análisis

Implementado y siguiendo con la arquitectura de *Clúster HPC*, se analizaron algunos Sistema *Middleware* para la distribución de procesos y poder compartir recursos de la tecnología *Clúster Computing* la cual permitirá la migración dinámica y eficiente de procesos, entre diferentes computadoras. Así como también una plataforma para la planificación de los trabajos y lograr ver los programas que se están ejecutando. Como se muestra en la tabla 4.

Tabla 4. Sistemas *middleware*.

Middleware	**Descripción**	**Componentes**
Condor	Utiliza eficazmente la potencia de cálculo de las estaciones conectadas en la red. Migran sus tareas a distintas máquinas que estén disponibles.	Condor permite computación Grid. Sistema Operativo: Unix y Windows. Interfaz de usuario: Consola/Terminal
OpenMOSIX	Actúa como un sistema multiprocesador. Balanceo de carga. Migración entre los procesos. No migran los procesos multi-hilo.	Parche para el kernel Linux. Herramientas para línea de comandos y para entorno gráfico.

Continuación de la tabla 4.

Middleware	Descripción	Componentes
MOSIX	Sistema operativo que proporciona una imagen única del sistema. Detección automática de recursos. Soporta aplicaciones secuenciales y paralelas. Asignación dinámica de recursos. Equilibrio de carga.	Un entorno no virtualizado requiere parchar el kernel de Linux y proporciona un mejor rendimiento.
Rock	Construido bajo Red Hat, soporta las arquitecturas x86 y x86_64. Soporta los procesadores AMD Athlon, Opteron, Itanium. Una de las distribuciones más utilizadas en el ámbito de los clúster por su facilidad de instalación e incorporación de nuevos nodos.	Basado inicialmente en Linux pero las últimas versiones están basadas en CentOS, las instalaciones pueden ser personalizadas utilizando CD's especiales, que extienden el sistema integrando automáticamente los mecanismos de gestión y empaquetamiento usados por el software base.
OSCAR	Incluye una arquitectura robusta y extensible lista para iniciar la producción. Crea imágenes de disco personalizadas para la prestación de los nodos con todo el software necesarios.	Utilizado principalmente en la computación científica mediante una interfaz de paso de mensajes.

Se utilizó MOSIX como gestor de procesos para obtener una imagen única de sistema orientada a la computación distribuida. MOSIX es un software que transforma un conjunto de computadoras conectadas en red bajo un sistema operativo GNU/Linux en un clúster. Equilibra la carga automáticamente entre las diferentes PC´s que contiene el clúster y pueden unirse o dejar el clúster sin interrumpir el servicio.

En MOSIX se pueden ejecutar las aplicaciones desde cualquier nodo pero normalmente se lo debe realizar desde un nodo principal (nodo maestro), en el cual se encuentran instaladas las diferentes herramientas de administración y monitoreo, para la correcta administración del clúster y así los usuarios se puedan acceder al clúster y ejecutar las distintas aplicaciones de una manera más fácil.

Se implementó el *framework* HADOOP, el cual trabaja con una arquitectura Maestro-Esclavo con dos tipos de nodo: nodo *máster* (maestro) y nodos *slaves* (esclavos), lo cual es perfecto para montar una arquitectura clúster para el análisis de Big Data.

HADOOP ofrece un enfoque de alto rendimiento y bajo costo para establecer una gran arquitectura de gestión de *big data* para soportar las iniciativas de analítica avanzada. Es el marco de trabajo más eficiente para obtener los máximos beneficios de la ciencia de los datos, permite el desarrollo de aplicaciones para procesar grandes volúmenes de datos, distribuido a través de un modelo de programación sencillo. Está diseñado para ser escalable puesto que trabaja con almacenamiento y procesamiento local, de manera que funciona tanto para clústeres de un solo nodo como para los que estén formados por miles.

Otra característica importante de HADOOP es la detección de errores a nivel de aplicación, pudiendo gestionar los fallos en los distintos nodos y ofreciendo un buen nivel de tolerancia a errores.

3.3.2 Implementación

Finalmente para esta fase de análisis surgieron para la implementación de los *middleware* las siguientes tareas:

Instalación y configuración MOSIX
Instalación y configuración del *framework Apache* HADOOP.

3.3.2.1 Instalación y configuración de MOSIX

Antes de comenzar a instalar MOSIX es necesario instalar las siguientes librerías:

```
# apt-get install dpkg-dev
# apt-get install debconf-utils
# apt-get install debhelper
# apt-get install linux-headers-$(uname -r)
# apt-get install libncurses5-dev
# apt-get install kernel-package
# apt-get install cmake
```

Una vez instaladas las librerías se procede a descargar MOSIX desde la página oficial http://www.MOSIX.com/ para posteriormente descomprimirlo. Se debe estar ubicados en el directorio donde se descargó el MOSIX.

Se descomprime la herramienta MOSIX

```
# tar jxf MOSIX-4.4.1.tbz
```

Se accede al directorio donde se descomprime

```
# cd MOSIX-4.4.1/
```

Se procede a crear los siguientes directorios:

```
# mkdir /etc/MOSIX
# mkdir /etc/MOSIX/var
```

Se mueven la carpeta a /etc/MOSIX

```
# cp –r MOSIX-4.4.1/ /etc/MOSIX
```

Ir a /etc/MOSIX y dentro buscar el directorio que se descomprimió anteriormente y se ejecuta el siguiente comando:

```
# ./MOSIX.install
```

Se procede a la instalación, ver Figura 9.

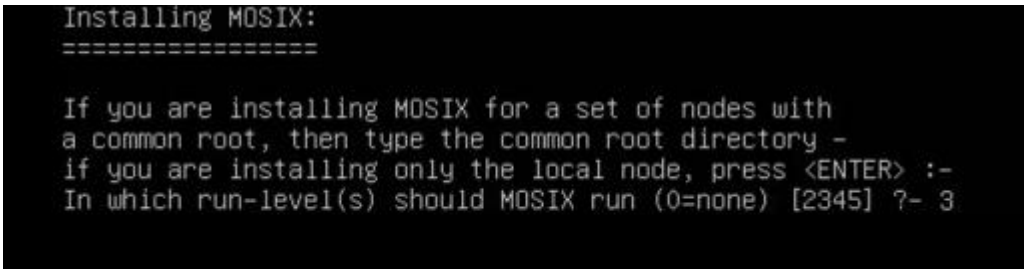

Figura 10. Primer paso para instalar MOSIX

Se indica el número de nodos que se necesita agregar. Siguiente a esto comienza la configuración del MOSIX, ver en la figura 10.

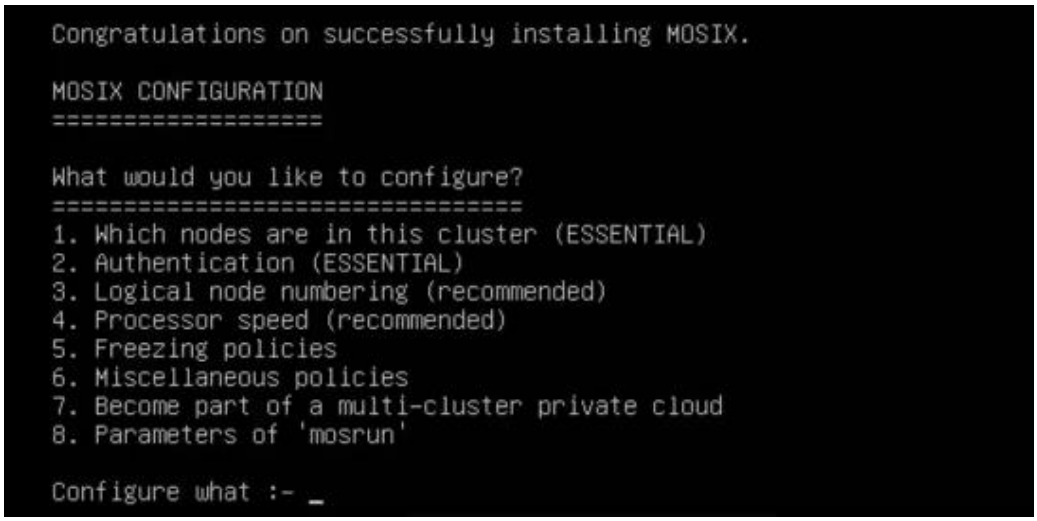

Figura 11. Menú MOSIX

Una vez instalado en cada uno de los nodos se copia *mosconf* en el nodo maestro para comenzar a configurar:

Opción 1: cuales nodos están en este *clúster.* Se colocó la IP del nodo principal:

Ejemplo: `150.186.92.185`

Seguidamente agregar el número de nodos y salir con la letra "q".

Opción 2: autentificación de claves. Definiremos el password único para todos los nodos que comprende el *clúster*.

Ejemplo: `123456.`

Opción 4: velocidad de procesador. Se establecerá la velocidad de los procesadores. (Límite del CPU en Mhz)

Ejemplo: `2000`

De la misma forma se edita en los modos esclavos.

```
# mosconf
```

Los nodos esclavos deben estar configurados de la misma forma, por último se edita el archivo /etc/crontab, con esto el MOSIX se cargara al encender la pc.

```
# nano /etc/crontab
```

3.3.2.2 Instalación y configuración HADOOP

Antes de iniciar con la implementación de HADOOP es necesario tener instalado la máquina virtual de java, en este caso se trabajara con la versión de JDK 1.8. En primer lugar, se debe verificar la existencia de Java en el sistema mediante *"java -version*". La sintaxis del comando versión de Java es la siguiente.

```
# java –version
```

Si todo funciona bien que le dará la siguiente salida.

```
java version "1.7.0_71"
Java(TM) SE Runtime Environment (build 1.7.0_71-b13)
Java HotSpot(TM) Client VM (build 25.0-b02, mixed mode)
```

Si Java no está instalado en su sistema, a continuación, siga los pasos que se indican para la instalación de Java. En primer lugar, agregue PPA de Oracle, a continuación, luego actualice el repositorio de paquetes.

```
# add-apt-repository ppa:webupd8team/java
# apt-get update
```

Luego, se instala la versión que se necesita

```
# apt-get install oracle-java8-installer
```

Puede haber varias instalaciones de JAVA en un servidor. Se configurara la versión por defecto para su uso mediante la línea de comandos usando *update-alternatives*, que gestiona cuales enlaces simbólicos se utilizan para diferentes comandos

```
# update-alternatives --config java
```

Una vez instalado java, es recomendable exportar las siguientes variables

```
# nano ~/.bashrc
```

```
export JAVA_HOME=/usr/lib/jvm/jdk1.8.0_45
export PATH=$JAVA_HOME/bin:${PATH}
```

Hecho esto ahora se probara **echo $JAVA_HOME** y tendría que mostrar en la consola **/usr/lib/jvm/jdk1.8.0_45.**

Finalizado la instalación de java, ahora si estamos en condiciones de instalar HADOOP. Comenzamos descargando *apache* HADOOP de http://hadoop.apache.org/releases.html. Para este proyecto se decidió trabajar con HADOOP-*2.7.3*. A continuación se debe copiar archivo *.tar* en */usr/local*. La instalación en modo clúster implica descomprimir el software en cada uno de los nodos y realizar las debidas configuraciones. El nodo maestro hace de *NameNode y ResourceManager* mientras tanto el resto de nodos o nodos esclavos hacen de *DataNode y NodeManager*.

Antes de comenzar crearemos una fichero donde se va a descomprimir HADOOP

```
# mkdir /usr/local/HADOOP
```

```
# cp HADOOP-2.7.0.tar.gz /usr/local/HADOOP
```

Se descomprime el archivo

```
# tar -xvf HADOOP-2.7.0.tar.gz
```

A continuación se debe configurar HADOOP. Para esto vamos a tener que editar unos cuantos archivos **~/.bashrc**

```
# nano ~/.bashrc
```

```
#SSH
export P4_RSHCOMMAND=SSH
#HADOOP
export HADOOP_PREFIX=/usr/local/HADOOP/HADOOP-2.7.3
export HADOOP_HOME=/usr/local/HADOOP/HADOOP-2.7.3
export HADOOP_CONF_DIR=HADOOP_HOME/etc/HADOOP
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_HOME=/usr/lib/jvm/java-8-oracle
export PATH=$PATH:$HADOOP_PREFIX/bin

#mpi
export PATH=/usr/lib64/mpi/gcc/OPENMPI/bin:$PATH
export PATH="$PATH:/home/$fnd/.OPENMPI/bin"
export
LD_LIBRARY_PATH="$LD_LIBRARY_PATH:/home/$fnd/.OPENMPI/lib/"
```

Una vez editado el fichero. Se cargan los cambios que se hicieron en el fichero bashrc.

```
# source ~/.bashrc
```

Para comprobar que todo ha salido correcto teclearemos en la terminal **HADOOP versión**, el cual nos tendrá que devolver la versión de HADOOP con la que estamos trabajando.

```
# hadoop version
```

Se procede a configurar apache HADOOP. Para poder utilizar de manera correcta este *framawork* se deben modificar los siguientes ficheros, se ubica en el directorio **/usr/local/HADOOP/etc/HADOOP/**

hadoop-*env.sh*
core-site.xml
hdfs-site.xm
mapred-site.xm
yarn-site.xml
slaves

Comenzaremos editando el archivo **HADOOP-env.sh** en el cual se configuraran las variables de ambiente que correrá HADOOP.

```
 # nano /usr/local/HADOOP/etc/HADOOP/HADOOP-env.sh
```

Se edita el fichero **HADOOP-*env.sh***, al cual indica el directorio donde se encuentran el ***JAVA_HOME*** y el **HADOOP_*PREFIX***.

```
export JAVA_HOME="/usr/lib/java-8-oracle"
export HADOOP_PREFIX="/usr/local/HADOOP/hadooop-2.7.3"
```

Se edita el fichero **core-site.sh**, el cual determinara el nodo servidor que contendrá el sistema HDFS y el puerto a través recibirán las peticionas del cliente.

```
# nano /usr/local/HADOOP/etc/HADOOP/core-site.sh
```

Se edita el fichero core-site.xml de la siguiente manera en el nodo maestro y esclavo:

```
?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
   <configuration>
     <property>
      <name>fs.defaultFS</name>
      <value>hdfs://cesar:9000</value>
     </property>
   </configuration>
```

Se edita el fichero hdfs-site.xml, define las opciones para el sistema de archivo de HADOOP, también el factor de replicar cada bloque.

```
# nano nano /usr/local/HADOOP/etc/HADOOP/hdfs-site.xml
```

El fichero hdfs-site.xml solo el nodo maestro

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>
 <property>
  <name>dfs.namenode.name.dir</name>
  <value>file:/home/fnd/HADOOP_data/hdfs/namenode</value>
 </property>

 <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/home/fnd/HADOOP_data/hdfs/datanode</value>
 </property>
```

```
   <property>
      <name>dfs.replication</name>
      <value>3</value>
   </property>
</configuration>
```

Se procede a crear los directorios

```
# mkdir /home/clúster/workspace/dfs/name
# mkdir /home/clúster/workspace/dfs/data
```

Se puede observar la configuración de este fichero en los nodos esclavos del clúster.

```
# nano nano /usr/local/HADOOP/etc/HADOOP/hdfs-site.xml
```

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
   <property>
      <name>dfs.replication</name>
      <value>3</value>
   </property>
</configuration>
```

Se indica la configuración del framework ***Yarn*** y el ***Job Tracker***, el cual es el encargado de ejecutar las aplicaciones desarrolladas con MapReduce. Esta configura es necesaria para ambos tipos de nodos.

```
# nano nano /usr/local/HADOOP/etc/HADOOP/mapred-site.xml
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
 <property>
   <name>mapreduce.framework.name</name>
```

```
   <value>yarn</value>
 </property>

 <property>
  <name>mapreduce.jobtracker.address</name>
  <value>cesar:9001</value>
 </property>

</configuration>
```

En el fichero *yarn-site.xml*, habilitamos las opciones de *Suffle* para poder ejecutar los trabajos entre Map y Reduce ya que *YARN por defecto no lo incluye, configuramos las opciones del resourcemanager del framework YARN.* Se debe editar en ambos nodos.

```
# nano nano /usr/local/HADOOP/etc/HADOOP/yarn-site.xml
```

```
<?xml version="1.0"?>

<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>

  <property>
    <name>yarn.nodemanager.aux-
services.mapreduce_shuffle.class</name>
    <value>org.apache.HADOOP.mapred.ShuffleHandler</value>
  </property>

  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>cesar:8030</value>
  </property>

  <property>
   <name>yarn.resourcemanager.resource-
tracker.address</name>
   <value>cesar:8031</value>
  </property>
```

```
<property>
   <name> yarn.resourcemanager.address</name>
   <value>cesar:8032</value>
  </property>

  <property>
   <name>yarn.resourcemanager.admin.address</name>
   <value>cesar:8033</value>
  </property>

  <property>
   <name>yarn.resourcemanager.webapp.address</name>
   <value>cesar:8088</value>
  </property>

</configuration>
```

Se edita el fichero *slaves* en los nodos del *clúster*, se observa que en el fichero se detallan los nombres de cada uno de los nodos esclavos.

```
# nano nano /usr/local/HADOOP/etc/HADOOP/slaves
```

koba

mau

3.3.3 Pruebas

Se verificar el estado de MOSIX

```
# /etc/init.d/MOSIX status
```

Para verificar el funcionamiento de todos los nodos configurados, se puede ver en la figura 11, ejecutamos el siguiente comando:

```
# mosmon
```

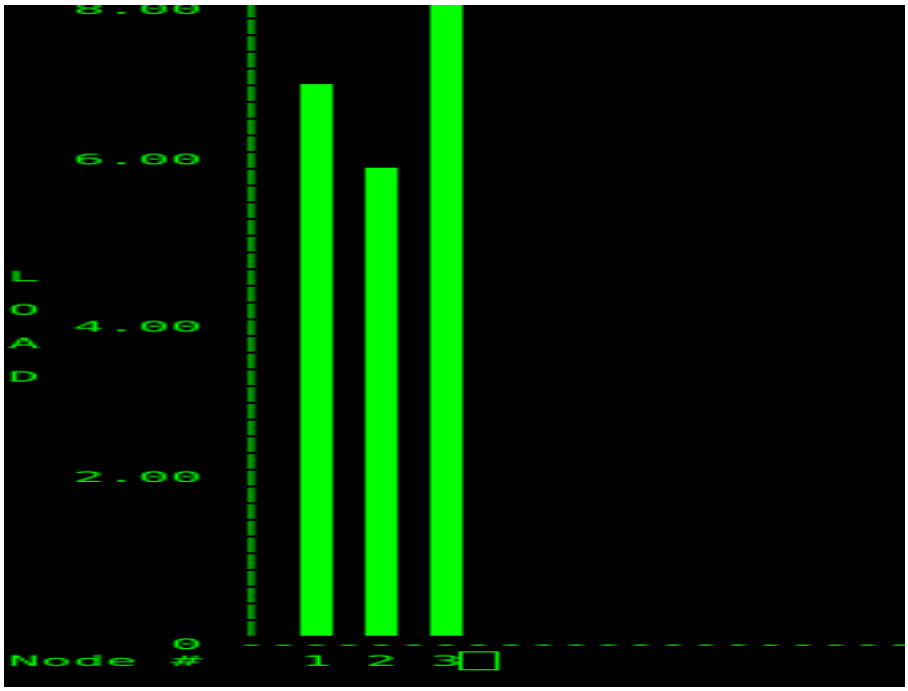

Figura 12. Prueba de funcionalidad MOSIX

Se podrá mostrar la carga de servidor en cada uno de los nodos configurados. Para más opciones. Comando:

```
# -help ./mosmon
```

3.3.3.1 Verificar el funcionamiento de HADOOP

Una vez realizada la instalación y configuración de framework Apache HADOOP se debe formatear el sistema de ficheros HDFS mediante el siguiente comando:

```
# hdfs namenode –format
```

Hecho esto se iniciaran los diferentes servicios de HADOOP en el *clúster* encargados de la ejecución de las tareas MapReduce y de la gestión del sistema de ficheros HDFS.

```
# start-all.sh
```

Posteriormente se tiene listo HADOOP para trabajar. Se verifica que el *clúster* local está funcionando correctamente con el comando **jps (java process status tool)** el cual mostrara que servicios están activos.

```
# jps
```

NameNode
DataNode
SecondaryNameNode
ResourceManager
NodeManager
Jps

Lo siguiente es acceder mediante un navegador web a la interfaz web de HADOOP mediante la dirección http://localhost:8088/, se puede ver en la figura 12, en el cual se observar la información de los trabajos ejecutados

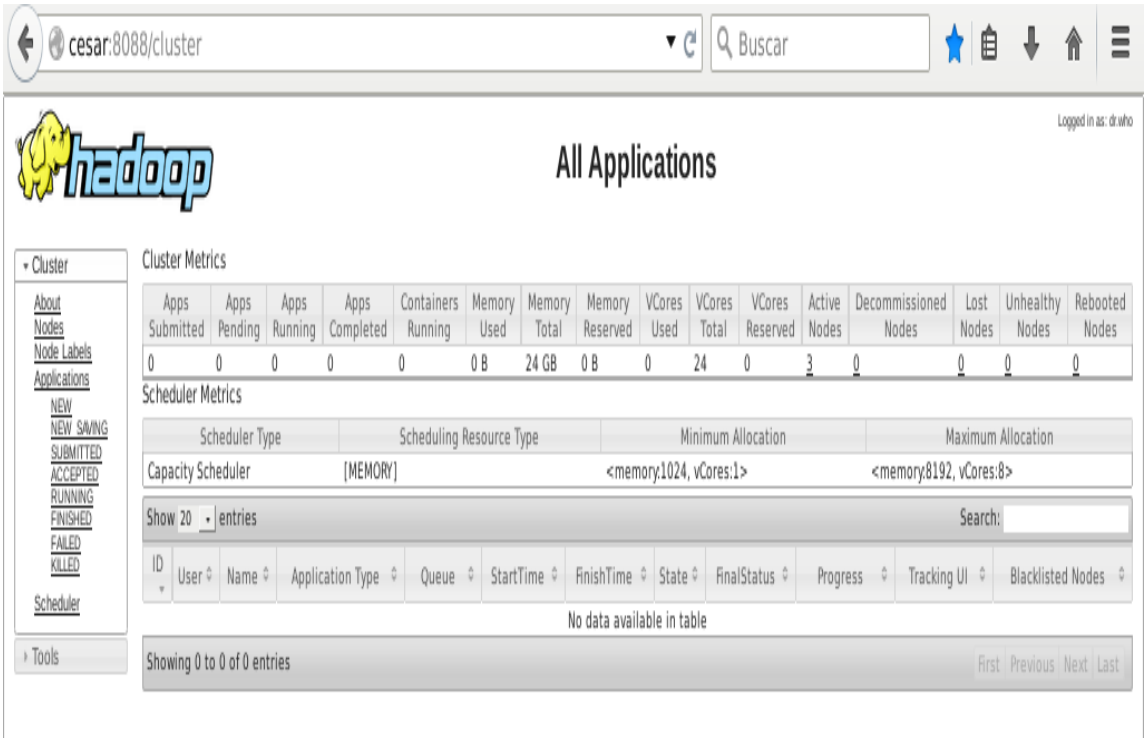

Figura 13. Interfaz YARN

A través del navegador web http://localhost:50070, se puede observar en la figura 13., se muestra una interfaz muy útil que muestra la información del NameNode y de los DataNodes.

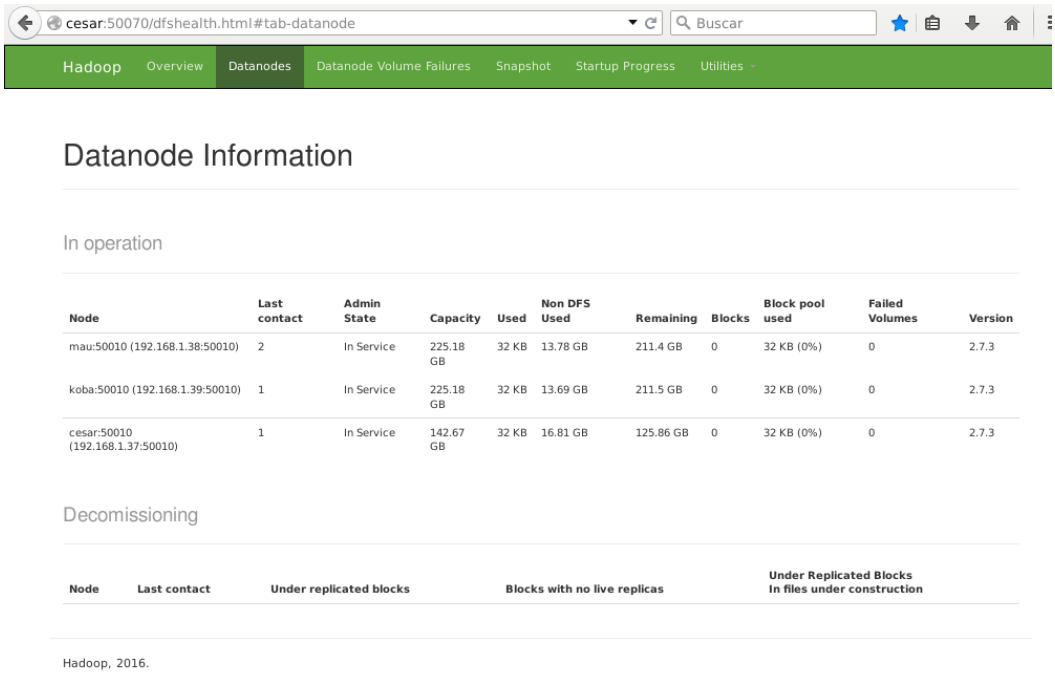

Figura 14. Interfaz de información de HADOOP

3.4 CUARTA ITERACIÓN

3.4.1 Análisis

Es necesario contar con un sistema para monitorear el estado de salud del *clúster* permitiendo recolectar métricas como: ocupación de los procesadores, uso de memoria, espacio en disco, etc. Para el sistema de monitoreo de *clúster* CISC se acordó poner en funcionamiento la herramienta *GANGLIA Monitoring System*, debido a que es muy popular en monitoreo de *clúster y grid*, de tal forma que ayude a tener un control del estado de cada nodo perteneciente al *clúster*. Esta herramienta se utiliza para ver estadísticas en vivo o grabadas que cubren métricas como los promedios de carga de la CPU o la utilización de la red para muchos nodos.

3.4.2 Implementación

Una de las actividades a elaborar en esta fase es la instalación y configuración de GANGLIA Monitoring System. Debido a lo cual es necesario instalar unas dependencias necesarias para su correcto funcionamiento.

En el terminal es necesario estar como súper usuario, los paquete a instalar son los siguientes:

```
# aptitude install libapr1-dev
# aptitude install libconfuse-dev
# aptitude install libpcre-ocaml-dev
# aptitude install librrds-perl
# aptitude install librrd2-dev
# aptitude install apache2
# aptitude install libapache2-mod-php5
# aptitude install php5-gd
# aptitude install rrdtool
```

Luego de instalar todas las dependencias anteriores se procede a instalar el sistema de monitoreo:

```
# apt-get install GANGLIA-monitor
```

Luego de instalar se reinicia el servicio:

```
# service GANGLIA-monitor restart
```

Se procede a instalar el servidor web:

```
# apt-get install GANGLIA-webfrontend gmetad
```

Se muestra una ventana, ver en la figura14, por consiguiente se elige la opción “SI” para reiniciar el servicio.

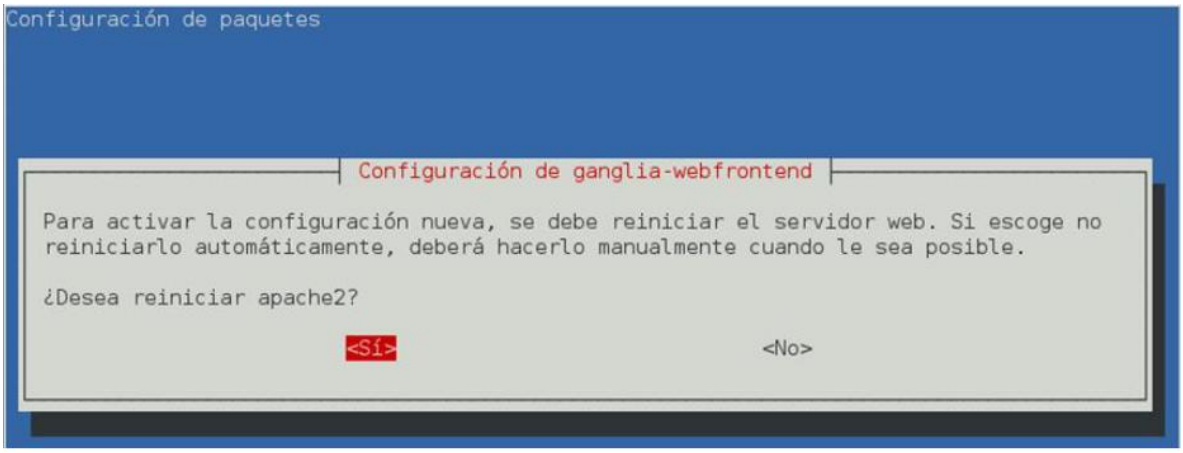

Figura 15. Reinicio del servicio web

Luego se debe activar sitio web *GANGLIA* donde ver la información gráfica:

Ir a /etc/GANGLIA-webfronted, localizar el .conf de Apache

```
# cd /etc/GANGLIA-webfronted/
```

Crear un link simbólico del mismo en /etc/apache/conf.d

```
#  ln –s /etc/apache/conf.d
```

Reiniciar Apache

```
# service apache2 restart
```

Se dirige al directorio **/etc/ganglia/gmond.conf** para comenzar a configurar el servidor GANGLIA.

Se edita el archivo /**etc/ganglia/gmond.conf** y solo se modifica el nombre (name= "unspecified").

```
# nano etc/GANGLIA/gmond.conf
```

```
Name="clúster CISC"
```

Se reinicia el servicio *GANGLIA*

```
# service ganglia-monitor restart
```

Se edita el archivo /etc/contab, se puede observar en la figura 15.

```
# nano /etc/crontab
```

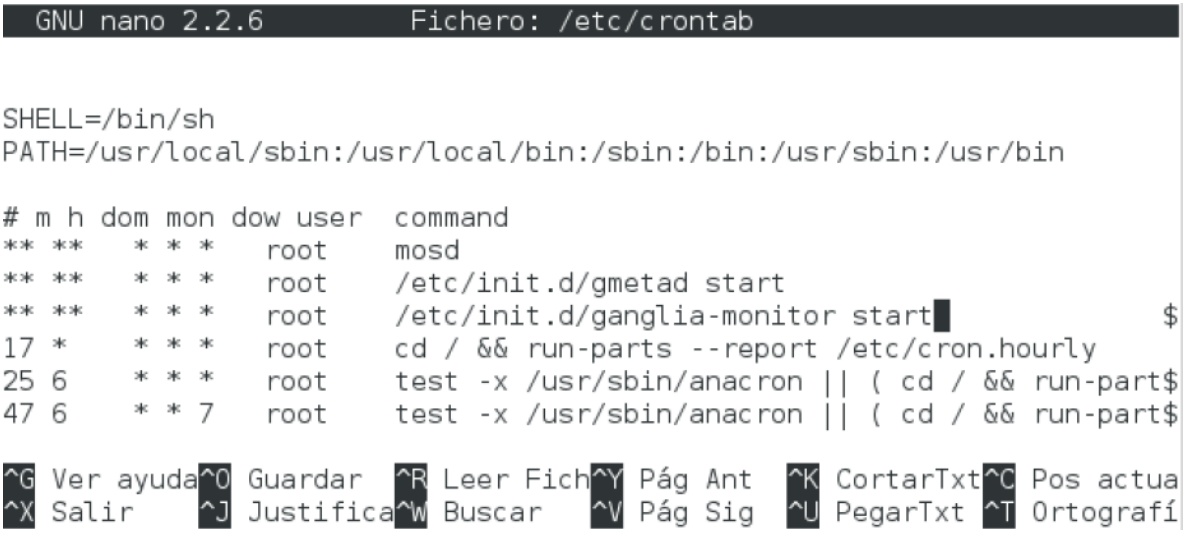

Figura 16. Edición del fichero *crontab* para el inicio automático de los servicio *GANGLIA* en el nodo maestro.

Se procede a configurar los nodos esclavos, se debe tener en cuenta que en cada uno de los nodos a monitorear se realiza la siguiente implementación.

En el terminal como usuario ***root*** se ejecuta el comando ***# apt-get install GANGLIA-monitor.***

```
# apt-get install GANGLIA-monitor
```

Se edita el administrador de procesos de linux ***crontab*** para arrancar automáticamente el servicio de monitorio de GANGLIA en cada uno de los nodos esclavos, se puede observar en la figura 16.

```
# nano /etc/crontab
```

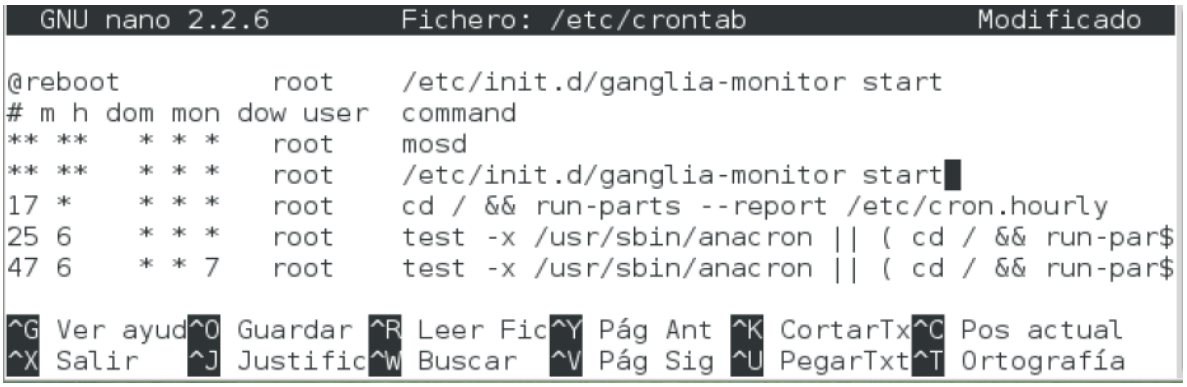

Figura 17. Edición del fichero *crontab* para el inicio automático de los servicios *GANGLIA* en los nodos esclavos

Se configura el archivo etc/GANGLIA/gmond.conf para configurar los nodos esclavos, en este caso se edita el nombre y host, se coloca la ip del nodo maestro.

```
# nano etc/GANGLIA/gmond.conf
```

Name="*clúster* CISC"

Hosts= 150.186.92.185

3.4.3 Pruebas

Una vez finalizadas las configuraciones, se accede al navegador web, a la dirección ***cesar/GANGLIA***, desde el nodo maestro, se observa en la figura 17 y 18.

```
# service GANGLIA-monitor restart
# service gmetad restart
```

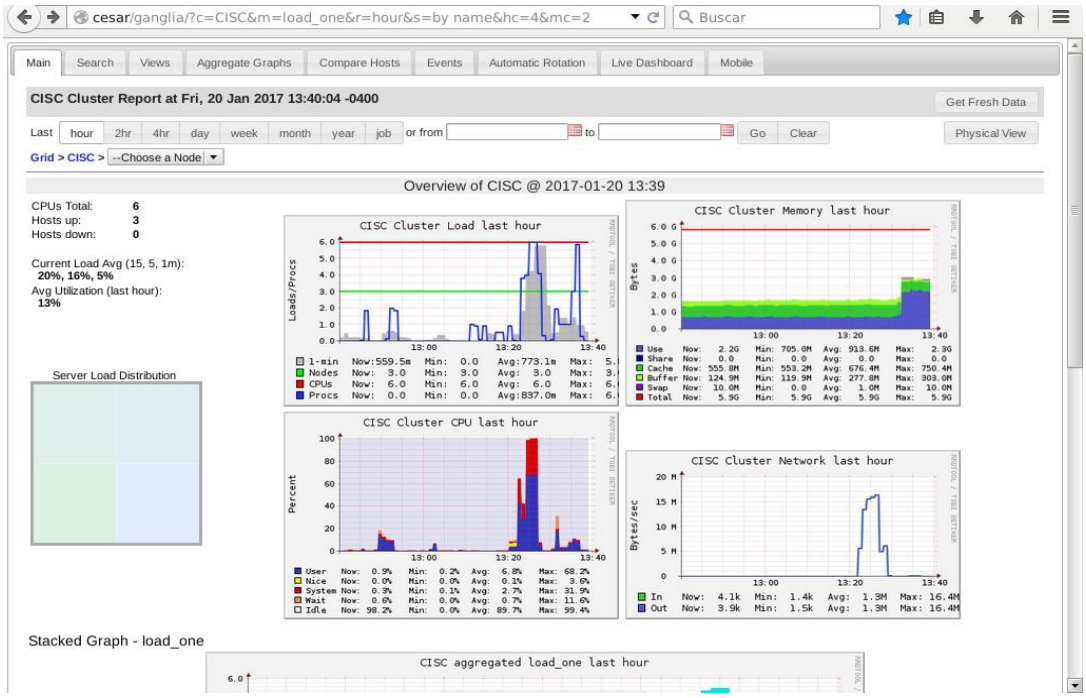

Figura 18. Estado de los nodos del clúster

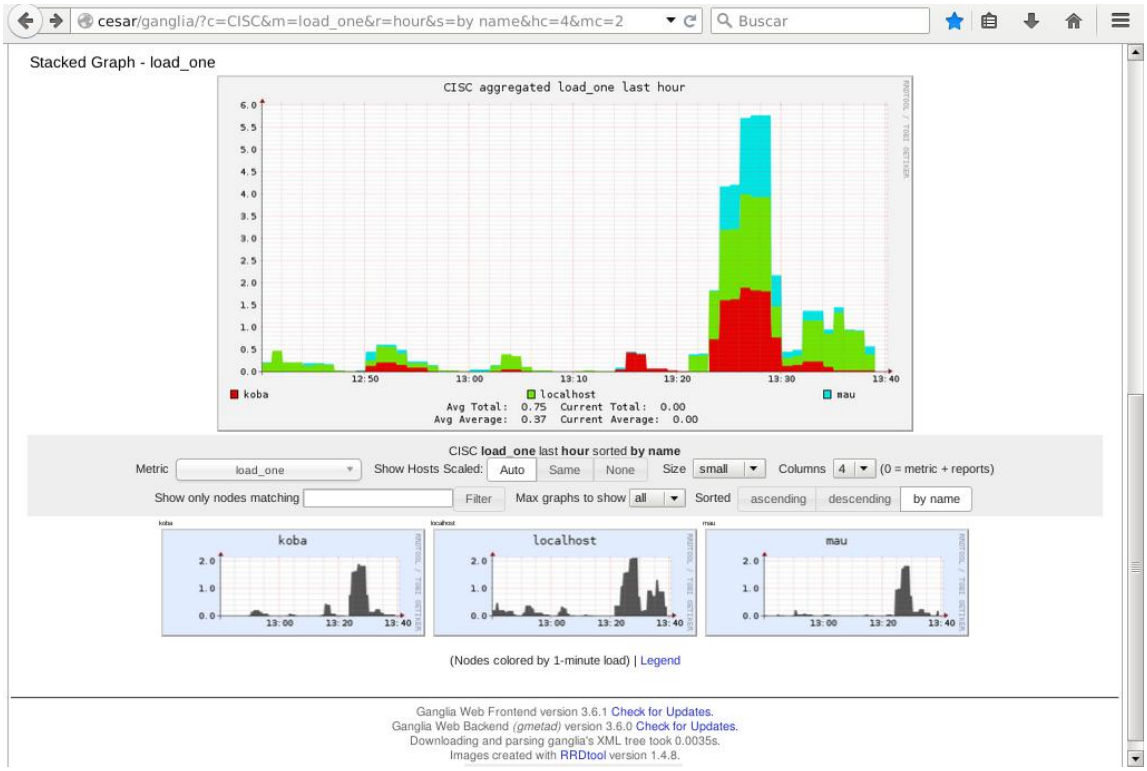

Figura 19. Estadísticas de cada uno de los nodos

Con esta última iteración se deja en funcionamiento el sistema de cómputo de alto rendimiento de forma operativa y estable, cubriendo todos los flancos posibles para su perfecto funcionamiento, además cumple con cabalidad cada uno del los objetivos planteados logrando así un clúster diseñado para 3 máquinas, un nodo maestro y 2 nodos esclavos, el cual es totalmente escalable lo que facilita la integración de más máquinas, así aumentar su capacidad de cómputo para agilizar aún más la obtención de los resultados.

CONCLUSIONES

La finalización del presente proyecto de grado permitió la integración de tecnología y diversas áreas de conocimiento, adquiridas durante la formación académica, que ayudaron a comprender de forma practica la importancia de realizar un buen levantamiento de información (tanto de lo que se requiere hacer como de los recursos disponibles para llevarlo a cabo y sus limitaciones).

A pesar que se utilizaron computadoras con velocidad de procesamiento lento, se pudo implementar una arquitectura factible pero limitada (aunque se les haya dotado de hardware necesario para contar con lo requerido).

Este proyecto usa herramientas de software libre disponibles en internet, tales como: el sistema operativo debían 8, MOSIX, HADOOP, librerías de programación, entre otros; su documentación en cuanto a soporte es muy escasa. Debido a ello, se requirió una extensiva investigación basada mayormente en internet, revisión bibliográfica, grupos de temas tecnológicos, entre otros.

Las herramientas utilizadas para este *clúster* resultaron ser muy eficientes, estas permitieron integrar y utilizar los recursos de las computadoras para un mejor desempeño y uso para ejecutar procesos en paralelo.

Después de desarrollar el proyecto de la implementación del sistema de cómputo de alto rendimiento se puede asegurar que incluso proyectos simples pueden acabar complicándose bastante.

Una buena planificación de proyecto es vital para poder llevarlo a cabo en los tiempos deseados. Por desgracia en muchas ocasiones imprevistos con los que no se habían contado pueden hacer que esa planificación se vea alterada y no se pueda tener el proyecto en las fechas marcadas.

Los problemas pueden venir de muy diversas formas. En el caso de éste proyecto han sido causas ajenas al trabajo académico que han hecho que el proyecto se demorase durante un largo periodo.

Es importante mencionar que pese a que los problemas encontrados durante la instalación de los componentes para formar el clúster fueron solucionados probando diferentes versiones del software, los problemas encontrados durante la configuración e integración de cada componente demandaron más lectura e investigación que la esperada, sobre todo por problemas de incompatibilidad y errores de configuración ya que la información encontrada en los documentos oficiales, libros y artículos en Internet está desactualizada e incompleta, incluso fue necesario leer el código fuente de algunas aplicaciones.

Básicamente este informe estuvo destinado a conocer los elementos necesarios para la construcción de un *clúster*, no tanto en su aplicación práctica pero si mayoritariamente teórica debido a los pocos recursos con los que se contaban para el desarrollo del tema.

RECOMENDACIONES

A continuación se listan algunas de las recomendaciones más importantes para la implementación de un clúster:

Nunca actualizar el software de los componentes del clúster en un ambiente de producción sin haber realizado las pruebas respectivas.

Revisar periódicamente los archivos de logs de los nodos del clúster.

Incentivar el desarrollo de la plataforma que permita integrar más nodos al *clúster*, con el fin de aumentar el nivel de cálculo y obtención de resultados.

Crear planes para realizar tareas de mantenimiento de hardware en los nodos.

Familiarizarse con la administración de clúster antes de ponerlo en producción.

Un clúster de Alto rendimiento, así como cualquier otro equipo que esté conectado a una red de computadores, puede ser objeto de ataques por personas malintencionadas; por lo tanto deben implementarse las medidas de seguridad necesarias para proteger a los componentes del clúster, como por ejemplo: control de acceso físico a los servidores, implementación de un firewall para el sistema operativo y los servicios del clúster, etc.

BIBLIOGRAFÍA

Abarca, M. 2008. Estructuración de un *clúster Beowulf*. Tesis de grado en ing. En informática, Universidad Católica de Temuco, Araucanía, Chile.

Boitnott, B. (2.009). Maintaining High Availability in Distributed Mobile Systems. Trabajo de grado presentado para optar al título de Magister en ciencias de la Computación, Escuela Naval de Postgrado, Monterrey – California.

Braastad, E. (2.006). Management of high availability services using virtualization. Trabajo de grado presentado para optar al título de Magister en Administración de Redes y Sistemas, Universidad de Oslo, Oslo – Noruega.

A. Barack and A. Shiloh, "The MOSIX *Clúster* Operating System for Distributed Computing on Linux *Clúster*s, Multi-*Clúster*s and Clouds", 2014.

Apache, "HADOOP - Apache HADOOP 2.5.1." [Online]. Available: http://HADOOP.apache.org/docs/r2.5.1/index.html. [Accessed: 13-Nov-2014]

Buyya, R. (1999). High Performance *Clúster* Computing: Architectures and Systems, Volume I, Prentice Hall, Upper Saddle River, New Jersey.

Buyya, R. (1999). High Performance *Clúster* Computing: Programming and Applications, Volume II, Prentice Hall, Upper Saddle River, New Jersey.

Bookman, Charles. (2002). "Linux *Clúster*ing: Building and Maintaining Linux *Clúster*s". Sams, 1era edition.

Brauss, S., Frey, M., Gunzinger, A., Lienhard, M. y Nemecek J. (1999). Swiss-Tx Communication Libraries, HPCN Europe 1999, Springer-Verlag.

Clark, T. (2003). Designing Storage Area Networks: A Practical Reference for Implementing Fibre Channel and IP SANs, 2nd edition, Addison Wesley.

Culler, D. y Singh J. (1999). Parallel Computer Architectures: A hardware/Software Approach, Morgan Kaufmann, San Francisco.

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

G. M. System, "GANGLIA Monitoring System" [Online]. Available: http://GANGLIA.sourceforge.net/. [Accessed: 04-Dec-2014].

[13] 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.

Gordon, B. y Gray J. (2001). “High Performance Computing: Crays, *Clúster*s, and Centers. What Next?” Technical Report MSR-TR-2001-76, Microsoft Research, Microsoft Corporation, USA.

Gropp, W., Lusk, E. y Sterling,T. (2003). Beowulf *Clúster* Computing with Linux, 2nd edition, The MIT Press.

Free Software Foundation. (2008). “What is GNU?”. Obtenido en línea el 4 de Agosto del 2011. Disponible en: http://www.gnu.org/home.es.html

Henesey, J. y Patterson D. (1996), Computer Architecture: A quantitative Approach, Morgan Kaufmann, San Francisco.

Hwang, K.; Xu, Z. (2008). “Scalable parallel computing”. WCB/McGraw-Hill.

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.

Kvasnicka D., Hlavacs H. y Ueberhuber C. (2001). “*Clúster* Configuration Aided by Simulation”. Proceedings of Computational Science – ICCS 2001, USA, 2073, 243--252.

Lucke, R. (2005). Building *Clúster*ed Linux Systems, Prentice Hall, Upper Saddle River, New Jersey.

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.

Pacheco P. (1997). Parallel Programming with MPI, Morgan Kaufmann, San Francisco.

Pfister G. (1995). In Search of *Clúster*s: The Coming Battle in Lowly Parallel Computing, Prentice Hall, Upper Saddle River, New Jersey.

Plaza E., 2002. *Clúster* Heterogéneo de Computadoras, <http://es.tldp.org/Manuales-LuCAS/doc-*clúster*-computadoras/doc-*clúster*-computadoras-html/node50.html>, (25/07/2012).

Pressman, R. 2010. Ingeniería del Software un enfoque Práctico, 7ma. ed. México:Mc Graw

Quinn, M. (2003). Parallel Programming in C with MPI and OpenMP, McGraw-Hill.

Sloan, J. (2004). High Performance Linux *Clúster*s: With Oscar, Rocks, openMOSIX, And MPI, O'Reilly & Associates.

Sterling, T. (2002). Beowulf *Clúster* Computing with Windows, The MIT Press.
Tamayo y Tamayo, M. 2003. El Proceso de Investigación Científica. Cuarta edición. Ediciones Limusa. S.A. México.

Tanenbaum, A. (2001). Modern Operating System, 2nd edition, Prentice Hall, Upper Saddle River, New Jersey.

Vrenios, Alex. (2000). "Building Linux *Clúster*s". O'Reilly.

APENDICES

INDICE

APENDICE A: IMPLENTACION DEL SISTEMA OPERATIVO LINUX

IMPLENTACION DEL SISTEMA OPERATIVO LINUX

Instalación debian 8

Iniciando la implementación del *clúster*, se instala en cada nodo el sistema operativo *debían 8 netinstall*, se encuentra disponible en https://www.debian.org/CD/netinst/.

Tras entrar en Debian 8 por primera vez, se continúa con el siguiente paso para la configuración de los nodos.

Configuración Clúster CISC

Configurar red

```
# nano /etc/network/interfaces

auto lo eth0
iface lo inet loopback
iface eth0 inet static
address  x.x.x.x     (ejem.:192.168.1.90)
netmask  x.x.x.x    (ejem.:255.255.255.0)
gateway  x.x.x.x    (ejem.:192.168.1.1)
```

Se reinicia el servicio

```
# /etc/init.d/networking restart
```

Configuración del DNS

```
# nano /etc/ resolv.conf

nameserver 8.8.8.8
nameserver 8.8.4.4
```

Añadir repositorios en Debian

Una vez establecida la conexión de red, procedemos a modificar la lista de repositorios en caso de ser necesario

Configuración:

```
# nano /etc/apt/sources.list

deb http://http.debian.net/debian jessie main contrib non-free
deb-src http://http.debian.net/debian jessie main contrib non-free
```

Se procede actualizando la lista de paquetes

```
# apt-get update
```

Se actualizan los paquetes instalados

```
# apt-get upgrade
```

Configuración de los grupos de redes

Esta configuración se realizó en todos los nodos de la misma forma se comienza editando el archivo /etc/hosts:

```
150.186.92.185 cesar
150.186.92.183 koba
150.186.92.184 mau
```

Interfaz gráfica para el nodo maestro

LXDE

Escritorio mínimo: pcmanfm, lxpanel, lxsession, openbox, xscreensaver (instalación manual de aplicaciones después)

```
# apt-get install lxde-core
# apt-get install lightdm
```

INSTALAR GOOGLE CHROME USANDO LA TERMINAL

Se agregan los repositorios a nuestra lista tecleando en el terminal Root

```
# nano /etc/apt/sources.list
```

```
deb http://dl.google.com/linux/deb/ stable main
```

se guardan los cambios y Luego de esto se ejecutan las siguiente llaves.

```
# gpg --keyserver hkp://subkeys.pgp.net --recv-keys
A040830F7FAC5991
```

```
# gpg --export --armor A040830F7FAC5991 | sudo apt-key add
-
```

Se actualiza

```
# apt-get update
```

Implementación de openSSH

OPENSSH Tiene dos conjuntos diferentes de archivos de configuración: uno para los programas cliente (SSH, scp, y sftp) y otro para el demonio del servidor (SSHd). La información de configuración SSH para todo el sistema está almacenada en el directorio /etc/SSH/:

```
# apt-get install SSH
```

Configuración del nodo maestro

Todas las configuraciones del servidor SSH se encuentran en el archivo /etc/SSH/SSHd_config.
Para configurar el servidor deben indicarse las direcciones donde el servicio debe responder. En este caso serán conexiones ligadas a la dirección 150.186.92.185, a través del puerto 26667 y utilizando la versión 2 del protocolo SSH:

```
 # nano /etc/SSH/SSHd_config

Port 2667
ListenAddress 150.186.92.185
Protocol 2
```

Por seguridad, se debe desactivar el login como root. En esta configuración, para adquirir los privilegios del root, se debe hacer un login usuario normal y, después, adquirir los privilegios de root. De este modo, prevenimos que el password del root sea objeto de un ataque.

```
#nano /etc/SSH/SSHd_config

LoginGraceTime 120
PermitRootLogin no
StrictModes yes
```

Luego hacemos un restart al servicio:

```
# etc/init.d/SSH restart
```

Configuración de los nodos esclavos

```
#nano /etc/SSH/SSH_config

Port 2667
ListenAddress 150.186.92.183
Protocol 2
```

APENDICE B: BITÁCORA

BITÁCORA

Fecha: 15-06-2015
Título: Asesoría

Se planifico una asesoría con un experto en la elaboración de clúster, dicha asesoría era extranjera y se hizo mediante la herramienta *skype*, esta charla duro alrededor de una hora y treinta minutos, y sirvió como guía para el arranque del proyecto, como a limitar las herramientas básicas que se deberían implementar.

Un *clúster* para cálculo debe estar conformado por maquinas en paralelo, un sistema de archivos distribuidos y librerías para poder programar en paralelo. Además de esto el clúster debe estar conformado por herramientas libres (requisitos de CISC).

Fecha: 18-06-2015
Título: Configuración de Equipos disponibles en CISC

Condiciones iniciales: Entre los dispositivos disponibles en CISC para la construcción del prototipo de *clúster*, se encontraban 3 máquinas con las siguientes características:

Descripción de equipos

Descripción	Nodo maestro	Nodos esclavos
Cantidad	1 genérica	2 x siragon 1320
Procesador	Procesador intel pentium dual core 2 ghz	Procesador intel pentium dual core 2 ghz
Memoria	Memoria ram 1g 555 x2	2 x Memoria ram 1g 555
Disco duro	Disco duro de 160gb	Disco duro de 160gb

Estas máquinas se encontraban en buen estado y sin ningún tipo de formato, como requisito de CISC cada uno de estos dispositivos debe estar bajo el Sistema Operativo Linux DEBIAN debido a que era el sistema operativo utilizado dentro del centro del investigación.

Configuración de la red

No se llevó el direccionamiento del IP puesto que se utilizó el direccionamiento establecido en las redes existentes en CISC, al mismo tiempo se le asignaros nombres a cada nodo.

Direcciones IP

Dirección de Red	Máscara de Subred	Gateway	Nodo
150.186.92.185	255.255.255.0	150.186.92.185	cesar
150.186.92.184	255.255.255.0	150.186.92.184	mau
150.186.92.183	255.255.255.0	150.186.92.183	koba

Archivo /etc/network/interfaces

Auto eth0

Iface eth0 inet static

Address 10.1.1.1

Netmask: 255.0.0.0

Gateway: 10.1.1.1

Implementación openmpi nodo maestro

Lista de actividades a Implementar en un nodo.

- Instalar librerías MPI
- Prueba con Mpi

En la investigación se decide utilizar OPENMPI por su fácil uso y modo de instalación.

Evento:

Durante la prueba de Openmpi se generó un error al ejecutar un programa en paralelo.

Fecha: 24-06-2015

Error Openmpi.

Se cuenta con un error al ejecutar un programa en paralelo.

No existía ningún compilador previamente instalando. Openmpi tiene como requisito tener instalado compiladores.

Pasos a seguir:

Se desinstala Openmpi y siguiente a esto volver a instalar, como usuario root.

apt-get purge openmpi-bin openmpi-common libopenmpi libopenmpi-dbg libopenmpi-dev

apt-get install gcc g++ build-essentials

apt-get install openmpi-bin openmpi-common libopenmpi libopenmpi-dbg libopenmpi-dev

Se ejecutaron pruebas con éxito.

Implementación Openmpi nodos esclavos

Lista de actividades a Implementar en cada nodo

- Instalación de Compiladores
- Instalación de Openmpi
- Prueba Openmpi de manera individual
- Pruebas en comunicación

Se ejecutaron pruebas individuales con éxito, las pruebas de procesamiento distribuido no funcionaron.

Fecha: 20-07-2015

Error encontrado

Es necesario instalar un sistema de archivos distribuido para hacer réplica del fichero en el clúster. Durante la investigación se consideran instalar SSH, NFS, NIS y LDAP.

Lista de actividades a Implementar en cada nodo

Implementar servidor NFS
Pruebas al servidor NFS
Implementar servidor SSH
Implementar NIS (opcional)
Implementar LDAP (opcional)

Implementación del servidor NFS

Se inicia desde una consola de root en el nodo maestro

apt-get install nfs-common nfs-kernel-server

Antes de arrancar el servicio NFS, es necesario indicar que carpeta se desea compartir y si se quiere que los usuarios accedan con **permisos de solo lectura o de lectura y escritura**. También existe la posibilidad de establecer desde que nodos es posible conectarse. Estas opciones se configuran en el archivo /etc/exports

Se crear la carpeta llamada banana desde el usuario creado.

mkdir /home/fnd/banana

Configuración del nodo maestro

nano /etc/exports

Se muestra el archivo /etc/exports para configurar algunas carpetas compartidas. Ejemplo:

```
/home 192.168.0.0/255.255.255.0(rw)
```

Se reinicia el servicio mediante el demonio nfs-kernel-server

/etc/init.d/nfs-kernel-server restart

Configuración del nodo esclavos:

Se inicia desde una consola de root en los nodos esclavos

apt-get install nfs-common

Configuración de los nodos esclavos

Ir a la carpeta /etc/exports de cada uno de los nodos esclavos.

nano /etc/exports

```
/home/fnd/banana cesar(rw,no_subtree_checkk, async,no_root_squash)
```

Se ejecuta el siguiente comando para levantar los servicios:

showmount –e cesar

Evento:

Error al montar las carpetas, el nodo maestro no replica los archivos.

Solución:

En el nodo maestro se desmonta el servicio de la siguiente manera:

#unmount /home/fnd/banana

Se verifico cara fichero y se encontró un error de sintaxis en el archivo /etc/exports del nodo maestro.

nano /etc/exports

```
/home/fnd/banana mau(rw,ro) koba(rw,ro)
```

Se reinicia el servicio

/etc/init.d/nfs-kernel-server restart

Se vuelve a montar el servicio en cada nodo

Error encontrado

Problemas al replicar el fichero del nodo maestro. Se recomendó instalar el servicio portmap, puesto que NFSv4 ya no requiere no fue necesario instalarlo, además, no se encuentra dentro de los repositorios de debían 8.

Fecha: 10-08-2015

Error encontrado

Se verifica cada paso de la instalación, reinician los servicios y se encontraron algunas posibles fallas:

Permisos de directorio

Propietario del directorio
Conexiones ssh.

Durante la investigación para resolver los problemas se verifico cada uno de los posibles errores, también es necesario crear un único usuario para ejecutar los programas y

Listas de actividades

- Crear el mismo usuario en cada nodo
- Cambiar permisos a cada directorios

Se cambian los permisos a los directorios en cada nodo.

chmod 755 –r /home/fnd/banana

Se cambian de propietario a los directories en cada nodo

chown –R fnd:fnd /home/fnd/banana

Se crean claves rsa para establecer una conexión sin claves.

Error encontrado

Problemas con la conexión ssh.

Se verifica el error ya que muestra en pantalla problemas con los puertos establecidos, el nodo maestro se encuentra con un puerto distinto a los nodos esclavos.

nano /etc/ssh/sshd_conf

Después de corregir los errores se reinicia el servicio

service sshd restart

Fecha: 17-08-2015

Se verifica lo anteriormente instalado y no se tiene novedad, todo ha funcionado con éxito.

Es necesario crear claves rsa para dar permisos de seguridad y se puedan comunicar sin pedir claves, de esta forma no se interrumpe el flujo de los procesos.

Siguiendo la lista de actividades

- Implementar NIS
- Implementar LDAP

Se investiga sobre cada servicio, NIS se utiliza para distribuir bases de datos, ficheros y directorios, en cambio el protocolo LDAP el cual es utilizado para autentificarse y almacenar cierto tipo de información, es un protocolo de acceso unificado a un conjunto de información sobre la red, se utiliza para replicar cuentas de usuarios en una red y crear carpetas personales para cada usuario.

Fecha: 20-08-2015

Implementación NIS

Después de investigar sobre el servicio NIS, se establece una lista de actividades:

Establecer nombre de dominio NIS

Configurar e iniciar el demonio servidor, *ypserv.*

Iniciar los mapas NIS

Iniciar el demonio de password NIS

Iniciar el demonio de transferencia

Modificar el proceso de inicialización del sistema para incluir los demonios NIS cada vez que se reiniciar la pc.

Observación:

Al reiniciar cada nodo es necesario tomar en cuenta que el nodo maestro debe estar encendido antes de los demás nodos, debido a que el nodo maestro es el que distribuirá los ficheros a los demás nodos, de no ser así los nodos esclavos tardaran en iniciar y no funcionara de forma correcta.

Fecha: 27-08-2015

Siguiendo la lista de tarea anteriormente planteada

Implementación LDAP

Después de investigar sobre el protocolo LDAP, se establece una lista de actividades:

Configurar demonio sldap

Evitar la configuración automática

Borrar bases de datos

Habilitar el protocolo LDAPv2

Reiniciar servicios

Modificar el proceso de inicialización del sistema para incluir los demonios LDAP cada vez que se reiniciar la pc.

Observación:

Tras instalar LDAP se crearon claves de súper usuarios para la administración, como también se crea una clave para el dominio DNS de LDAP.

Fecha: 05-09-2015

Problemas con el hardware el nodo maestro no prende, Se verifica y se identifica la falla, la fuente de poder no funciona. Esta falla es causada por apagones consecutivos en la ciudad.

Al no contar con el nodo maestro se investiga la forma de cómo implementar un planificador de lotes de procesos para el *clúster* (batch scheduler)

Fecha: 18-09-2015

Se pretende implementar OAR batch scheduler, para framework es necesario contar con 1 nodo maestro el cual se llamara frontend y los nodos esclavos llamados servernode. Para la implementación de este batch era necesario tener un conocimiento amplio de Linux, además era necesario conocer el funcionamiento de herramienta, instalar algunas dependencias necesarias para su correcto funcionamiento y configurar algunas aplicaciones previamente instaladas para la sincronía y arranque del *clúster*.

Fecha: 22-09-2015

Existen 2 modos de instarla OAR. Agregando los repositorios de oar o descargar el paquete e instalarlo por MAKEFILE

Se planea

Investigar sobre variables MAKEFILE para poder lograr instalar OAR
Investigar usuario virtual de OAR

Definir paquetes para el nodo frontend
Definir paquetes para los servernode
Definir que gestor de bases de datos usar
Investigar sobre servidor NTP

Investigar sobre variables MAKEFILE para poder lograr instalar OAR
Después de revisar la documentación se decide usar el método de agregar los repositorios de OAR e instalarlo con todas sus dependencias ya que no se entendía usar variables MAKEFILE.
Investigar usuario virtual de OAR
El usuario virtual de OAR viene agregado de forma predeterminada y es el responsable de control de todo el framework.

Fecha: 29-09-2015

Continuando con el plan anterior

- Definir paquetes para el nodo frontend
- Definir paquetes para los servernode
- Definir que gestor de bases de datos usar
- Investigar sobre servidor NTP

Definir que gestor de bases de datos usar
Se decide usar Mysql ya que se manejaba el entorno.

Investigar sobre servidor NTP

Network Time Protocol (NTP) es un protocolo de Internet para sincronizar los relojes de los sistemas informáticos a través del enrutamiento de paquetes en redes con latencia variable.

Definir paquetes para el nodo frontend

Además de las herramientas anteriormente instaladas (ssh, mpi, nfs) era necesarios intalar apache, algunas dependencias de PERL además: Mysql. Además oar-adminoar-server oar-docoar-web-status las cuales eran necesarias para instalar OAR.

Definir paquetes para los nodos servernode

oar-node como el paquete mediante el nodo frontend se podrá comunicar con los demás nodos esclavos y las herramientas anterior mente instaladas.

Fecha: 12-10-2015

Se logra conseguir una fuente de poder para el nodo maestro, ya instalada podemos comenzar a hacer pruebas con OAR.

Arranque de esta implementación

Nodo maestro

Aptitude install oar-adminoar-server oar-docoar-web-status

Editar los archivos

#nano /var/lib/oar/.ssh/authorized_keys

nano /var/lib/oar/.ssh/id_rsa.pub

Y agregar *environment="OAR_KEY=1”* al comienzo y sin olvidar después dejar un espacio entre lo anterior escrito y el contenido del archivo.

Inicializar la base de datos desde OAR

oar_mysql_db_init

Inconveniente presentado

No arroja los resultados correctos, no reconoce la base de datos.

Fecha: 22-10-2015

Asesoría con externa

Se tuvo una charla con un experto en instalación de OAR de al menos 30 min. Se determinó el problema, resulto ser la clave ssh del usuario virtual OAR.

Inconveniente

No se había trabajado anteriormente con usuarios virtuales, estos tipos de usuarios no tienen claves y en el momento de enviar una clave RSA mediante ssh era necesario ingresar clave. Como alternativa a esto se decide hacer todo de forma manual y dando los permisos respectivos.

Luego de esto seguía dando error de conexión.

Fecha: 26-10-2015

Cambio de monitor por problemas técnicos, debido a cambios anterior mentes hechos se contaba con error de conectividad y el nodo koba no lograba establecer conexión con el nodo cesar.

Resuelto el problema de los permisos se pretende hacer una revisión de documentos y revisión bibliográfica para encontrar solución.

Fecha: 15-11-2015

Debido a los problemas presentados en oar por faltas de conocimientos con el entorno debían, se decide descargar el batch scheduler OAR ya que su implementación fue muy completa. Se decide formatear todos los nodos y empezar de nuevo, de esta forma trabajar con un servidor más limpio de otros demonios.

Fecha: 20-11-2015

Tras implementar los paquetes básicos del clúster (ssh, mpi y nfs) y hacer pruebas de conexión y de funcionalidad, se descarta la instalacion de LDAP y Nis ya que solo se contaban con 3 máquinas y esas eran herramientas administrativas del cluster para tener un control de los usuario y permisos (era necesario adquirir otro nodo extra para crear un nodo para administrar usuarios) de esta forma el nodo maestro no tendría mucha carga de procesos al momento de ejecutar los programas.

Se propone investigar sobre otras alternativas de middleware.

Fecha: 5-12-2015

Se decide implementar MOSIX como plataforma middleware. Para comenzar era necesario:

Instalar cmake previamente (# apt-get install cmake)

Crear una carpeta dentro del fichero /etc con el nombre “MOSIX”

Dar permisos a esa carpeta “MOSIX”

Ubicarse en la carpeta para luego descomprimir o descargar el paquete MOSIX

Instalar MOSIX

Configurar MOSIX

Es necesario implementar los siguientes pasos en cada uno de los nodos antes de instalar y configurar la conexión entre ellos.

Es necesario investigar la configuración ya que se desconoce la forma de como aglomerar los nodos.

Fecha: 10-01-2016

Luego de descargar e instalar en cada uno de los nodos se procede a configurar, con MOSIX no es necesario configurarlo de forma manual, con el comando MOSCONF y seguir las instrucciones se configuro de forma satisfactoria.
Aunque ya se implementó MOSIX se desconoce el manejo de la herramienta, se panea investigar el uso de la herramienta de forma básica para administrar y uso de esta herramienta.

Fecha: 15-01-2016

Dentro de las aplicaciones que trae MOSIX, se encuentra el programa **mon** que se ejecuta en modo texto (consola) y que permite monitorear la carga de los nodos del clúster, obtener información acerca de las velocidades de los nodos, cantidad de memoria, etc.
Usando el **mon** pudimos apreciar lo siguiente:

Al ejecutar nuestra rutina X! en el nodo más lento, MOSIX migraba el proceso hacía el nodo más rapido, quedando el nodo lento sin carga.

Al ejecutar la rutina en el nodo más ràpido, toda la carga se quedaba en el mismo nodo, es decir, MOSIX no migraba el proceso hacia otros nodos.

Después de conocer los comandos básicos de MOSIX se procede a hacer pruebas de funcionalidad. Dichas pruebas no salieron con éxito, debido a un error.

Fecha: 16-01-2016

Solución

Existia un problema de configuración, problemas a comunicarse, los nodos esclavos estaban mal configurados. Donde se encontraba la IP del nodo maestro contenía su propia IP. Luego de hacer los respectivos cambios y los nodos esclavos reconfigurados se procede a hacer pruebas.

Las pruebas se ejecutaron con éxito.

Fecha: 18-01-2016

Se decide agregar un framework capaz de procesar información masiva y hacer la distribución median el clúster para esto se planea hacer una revisión bibliografía sobre herramientas herramientas de para análisis masivo de datos.

Se chequea lo anterior mente instalado y se notó un que accidentalmente se ejecuto

apt-get update
apt-get upgrade

Se notó que algunos paquetes se actualizaron y no eran compatibles con las configuraciones implementadas. Se procede a investigar cómo solucionar este conflicto.

Fecha: 12-02-2016

No se encontro solución, no se ejecutaban los programas anteriormente probados.

Se decide formatear y volver a implementar el cluster.

Fecha: 25-02-2016

Luego de implementar cada uno de las herramientas y despues de tener una asesoria con expertos extranjeros entre las herramientas sugeridas, el framework que mas se comprende y se adapta a la meta que se quiere es HADOOP.

Como plan para arrancar con la implementacion de este framework se decide:

Investigar como instalar esta plataforma
Investigar las diferentes configuraciones
Depentencias que necesitan.
Fecha: 22-02-2016

HADOOP por ser un ecosistema de software e incluir diferentes framework, las herramientas que se necesitan configurar son: YARN, HDFS, MAPREDUCE. Como requisito preliminar es necesario tener actualizado el JDK de JAVA.

Actividades a realizar:

- Verificar y actualizar el JDK al 1.8 (en tal caso que lo requiera)
- Definir que topologia de software utilizar para implementar HADOOP
- Instalar HADOOP
- Configurar HADOOP
    - hadoop-env.sh
    - core-site.xml
    - hdfs-site.xm
    - mapred-site.xm
    - yarn-site.xml
    - slaves

existen 3 topologias para configurar HADOOP: un nodo, modo virtual, modo cluster.
De esta forma se decide implementar el modo cluster para lograr distribuir los trabajos en cada uno de los nodos que se cuentan.

Para la configuracion se estudio el manual que brinda la pagina oficial de APACHE HADOOP, http://hadoop.apache.org/.

Al terminar de configurar cada uno de los archivos (maestro-esclavos) se procede a ejecutar la herramienta HADOOP.

Error

No se ejecutaron los servicios completos, se procede a revisar y solucionar el error.

Fecha: 26-02-2016

Verificado cada uno de los archivos configurados, se corrigieron multiples errores, pero al volver a encender los serivicos ocurrio un inconveniente que no dejaba ejecutar todos los servicios.

Solucion

En la investigacion del problema se noto que no se configuro el el fichero .bashrc,
luego de editar este fichero se lograron levantar todos los servicios

Fecha: 05-03-2016

Se intenta ejecutar un ejemplo con la herramienta HADOOP, pero se desconocia completamente el uso del mismo, asi como la administracion de la herramienta.

Actividades

Investigar acerca del uso de hadoop y su administracion.

Fecha: 10-03-2016

Al haber investigado y lograr contar con un conocimiento basico sobre el uso de HADOOP, se noto que la ejecutar el programa probar la herramienta se contaba con un error en la ejecucion.

Fecha: 11-03-2016

Solucion

El error era ocurrido por no haber subido el paquete a analizar al HADOOP, sabiendo esto se procede a ejecutar los siguentes comandos, es importante que antes de subir un paquete por primera vez en el servidor, es necesario darle formato.

Subir ficheros a HDFS

Si se quiere hacer una prueba de manejo de ficheros, se pueden ejecutar los siguientes comandos.

Crear un directorio HDFS

```
# hdfs dfs -mkdir /user
# hdfs dfs -mkdir /user/hduser
```

Ingresar archivo

hdfs dfs –put prueba.csv /nombre del fichero

Copiar archivos en el sistema de ficheros distribuido

hdfs dfs -put <***carpeta_local_origen***> <***carpeta_hdfs_destino***>

Borrar un archivo del sistema de ficheros distribuido

hdfs dfs -rm <***carpeta_hdfs***>/<***nombre_fichero***>

Ejecución de un aarchivó

hadoop jar ***nombre_programa***.jar \

Descargar el resultado

hadoop fs -get output/* output_ejemplo

Existe un error al ejecutar cada uno de los comandos.

Fecha: 15-03-2016

Solución del error

Era necesario agregar el usuario al grupo de HADOOP para poder ejecutar los comandos anteriormente descritos, también se notó que el usuario ROOT debe mantener activo los servicios para que los otros usuarios puedan utilizarlo.

HOJA DE METADATOS

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 1/6

Título	Implementación De Un Sistema De Cómputo De Alto Rendimiento Para El Centro De Investigación En Servicios De Cómputo De La Universidad De Oriente
Subtítulo	

Autor(es)

Apellidos y Nombres	**Código CVLAC / e-mail**	
Maza Díaz, Víctor Ezequiel	**CVLAC**	18.777.635
	e-mail	victormaz4@hotmail.com
	e-mail	vicmaz477@gmail.com

Palabras o frases claves:

Clúster Beowulf
MOSIX
HADOOP
OPENMPI
NFS
SSH
GANGLIA Monitoring System
Computación Paralela

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 2/6

Líneas y sublíneas de investigación:

Área	Subárea
CIENCIAS	INFORMÁTICA

Resumen (abstract):

Se presenta una estrategia para la implementación de un sistema de cómputo de alto rendimiento en la Universidad de Oriente Núcleo de Sucre. Para lo cual se realizó un estudio de la situación de los profesores que se encuentran realizando proyectos que requieran una gran cantidad de procesamiento; en donde pudo apreciarse carencia de tecnología que le permita trabajar dentro del Núcleo, lo cual conllevó, mediante la aplicación del modelo incremental propuesto por Harlan Mills, con el objetivo de ofrecer soluciones adaptables a las necesidades de los requerimientos de los usuarios y reducir la repetición del trabajo en el proceso de desarrollo, así dar oportunidad de retrasar la toma de decisiones en los requisitos hasta adquirir experiencias con el sistema. Se evaluaron diferentes soluciones a la implementación del *Clúster*, a través del análisis de la tecnología de red, arquitectura del nodo, sistema de archivos, administración de procesos y herramientas de programación. Para el montaje de este *Clúster* fue necesaria la instalación y configuro del sistema operativo GNU/Linux Debian "jessie" en todas las máquinas por su estabilidad y funcionamiento, además la configuración del protocolo NFS encargado de compartir carpetas de forma tal que puedan ser accedidas remotamente, también SSH la cual brindaba una conexión segura entre las máquinas y poder acceder a ellas de forma remota, librerías MPI las cuales fueron necesarias para el procesamiento en paralelo, para el middleware se utilizó MOSIX como gestor entre las máquinas y lograr obtener un sistema distribuido, como también, como un punto extra se empleó el *framework* HADOOP para el análisis masivo de información, se implementó un sistema de motorización con la herramienta *GANGLIA Monitoring System* que funciona perfectamente para observar el estado de vida de cada uno de los nodos. El fin de la computación paralela es resolver problemas más grandes en menos tiempo y a bajo costo. Por último se realizaron las pruebas con el fin de demostrar la reducción del tiempo de procesamiento trabajando en la computadora de manera secuencial y la distribución de la misma tarea en varias máquinas trabajando de forma paralela. En síntesis, este trabajo muestra que el estudio de la performance y optimización de *Clúster* permite obtener una notable mejoría en el uso de los recursos facilitando el objetivo fundamental de la computación paralela que es permitir resolver problemas más grandes en menor tiempo.

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 3/6

Contribuidores:

Apellidos y Nombres	ROL / Código CVLAC / e-mail	
José Sifontes	**ROL**	**CA** [ ] **AS** [x] **TU** [ ] **JU** [ ]
	CVLAC	12123953
	e-mail	jasifontes@yahoo.com
	e-mail	
José francisco Romero	**ROL**	**CA** [ ] **AS** [x] **TU** [ ] **JU** [x]
	CVLAC	13631597
	e-mail	jromero@udo.edu.ve
	e-mail	jfromeropino@gmail.com
Carmelys Rodriguez	**ROL**	**CA** [ ] **AS** [ ] **TU** [ ] **JU** [x]
	CVLAC	13539531
	e-mail	carmelysrodriguez@gmail.com
	e-mail	
Ana Fuentes	**ROL**	**CA** [ ] **AS** [ ] **TU** [ ] **JU** [x]
	CVLAC	
	e-mail	afuentes_marquez@hotmail.com
	e-mail	

Fecha de discusión y aprobación:

Año	Mes	Día
2017	**3**	**10**

Lenguaje: **ESPAÑOL**

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 4/6

Archivo(s):

Nombre de archivo	Tipo MIME
Tesis-Maza.docx	**Aplication/word**

Alcance:

Espacial: Universal **(Opcional)**

Temporal: Intemporal **(Opcional)**

Título o Grado asociado con el trabajo:

Licenciado en informática

Nivel Asociado con el Trabajo: Licenciado

Área de Estudio: Sistemas Distribuidos / Computo de alto Rendimiento

Institución(es) que garantiza(n) el Título o grado:
Universidad de Oriente (UDO)

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 5/6

UNIVERSIDAD DE ORIENTE
CONSEJO UNIVERSITARIO
RECTORADO

CU Nº0975

Cumaná, 04 AGO 2009

Ciudadano
Prof. JESÚS MARTÍNEZ YÉPEZ
Vicerrector Académico
Universidad de Oriente
Su Despacho

Estimado Profesor Martínez:

Cumplo en notificarle que el Consejo Universitario, en Reunión Ordinaria celebrada en Centro de Convenciones de Cantaura, los días 28 y 29 de julio de 2009, conoció el punto de agenda ***"SOLICITUD DE AUTORIZACIÓN PARA PUBLICAR TODA LA PRODUCCIÓN INTELECTUAL DE LA UNIVERSIDAD DE ORIENTE EN EL REPOSITORIO INSTITUCIONAL DE LA UDO, SEGÚN VRAC Nº 696/2009".***

Leído el oficio SIBI – 139/2009 de fecha 09-07-2009, suscrita por el Dr. Abul K. Bashirullah, Director de Bibliotecas, este Cuerpo Colegiado decidió, por unanimidad, autorizar la publicación de toda la producción intelectual de la Universidad de Oriente en el Repositorio en cuestión.

Comunicación que hago a usted a los fines consiguientes.

UNIVERSIDAD DE ORIENTE
SISTEMA DE BIBLIOTECA
RECIBIDO POR
FECHA 5/8/09 HORA 5:20

Cordialmente,

REPÚBLICA BOLIVARIANA DE VENEZUELA * UNIVERSIDAD DE ORIENTE * SECRETARÍA CONSEJO UNIVERSITARIO

JUAN A. BOLAÑOS CURIEL
Secretario

C.C: *Rectora, Vicerrectora Administrativa, Decanos de los Núcleos, Coordinador General de Administración, Director de Personal, Dirección de Finanzas, Dirección de Presupuesto, Contraloría Interna, Consultoría Jurídica, Director de Bibliotecas, Dirección de Publicaciones, Dirección de Computación, Coordinación de Teleinformática, Coordinación General de Postgrado.*

JABC/ YGC/ maruja

Apartado Correos 094 / Telfs: 4008042 - 4008044 / 8008045 Telefax: 4008043 / Cumaná - Venezuela

Hoja de Metadatos para Tesis y Trabajos de Ascenso- 6/6

Artículo 41 del REGLAMENTO DE TRABAJO DE PREGRADO (vigente a partir del II Semestre 2009, según comunicación CU-034-2009) : "los Trabajos de Grado son de la exclusiva propiedad de la Universidad de Oriente, y sólo podrán ser utilizados para otros fines con el consentimiento del Consejo de Núcleo respectivo, quien deberá participarlo previamente al Consejo Universitario para su autorización".

Autor

Asesor Institucional

Asesor Académico