UNIVERSIDAD DE ORIENTE
NÚCLEO DE SUCRE
ESCUELA DE CIENCIAS
DEPARTAMENTO DE MATEMÁTICAS
PROGRAMA DE LA LICENCIATURA EN INFORMÁTICA

ENLACE *VOLUNTEER COMPUTING* EN LA UNIVERSIDAD DE ORIENTE NÚCLEO DE SUCRE

(Modalidades: Pasantía de Grado)

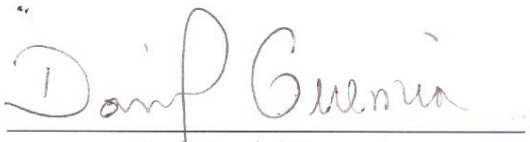
MARÍA INÉS MAYS BELMONTE

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

CUMANÁ, 2013

ENLACE *VOLUNTEER COMPUTING* EN LA UNIVERSIDAD DE ORIENTE NÚCLEO DE SUCRE

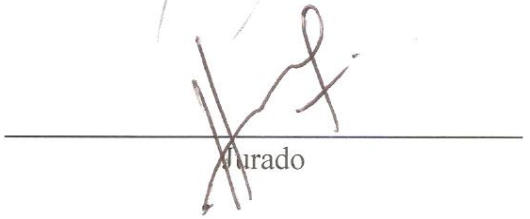
APROBADO POR:



Prof. Daniel Geremia
Asesor Académico

Ing. Napoleón Milá de la Roca
Asesor Industrial

Jurado



Jurado

DEDICATORIA

A:

Dios y mis cómplices Uds. Que jamás me juzgaron y me ayudaron sin esperar recibir nada a cambio, gracias totales.

El amor de mi vida, el motor que me impulsó a llegar hasta el final, quién siempre me escuchó y me dio ánimos en los momentos en los que deseaba dejarlo todo, mi abue Cruz, gracias de corazón, espero tengamos el tiempo de retribuirte todo lo que has hecho por mí.

Mi hermano Antonio Mays, mi bebé, espero servir de ejemplo, los sueños si pueden hacerse realidad y vale más subir una montaña con obstáculos que bajar por una pendiente lisa, esos obstáculos son los que te ayudarán a ser mejor persona y valorar cada cosa que consigas, ahora te cedo el testigo a ti Bro, gracias por ser mi confidente y alegría en muchos momentos.

Mis primitos y futura generación, en especial a mi negris Ana Victoria, que esto les sirva de motivación para lo que deseen ser en la vida, elijan lo que más les guste, así lo disfrutarán más y el camino será más placentero, los quiero.

AGRADECIMIENTOS

Al Profesor Daniel Geremia por confiar en mí desde el principio, por ser más que mi asesor; un consejero y amigo, un ejemplo para mí de lo que en un futuro me gustaría ser y hacer como profesional.

Al Ing. Napoleón Milá de la Roca por su tiempo y los conocimientos brindados para el desarrollo de este proyecto, gracias por adentrarme al mundo del software libre y de los servidores virtuales.

Al Lcdo. José Francisco Romero quien de forma desinteresada se comprometió en este proyecto y tomó el testigo en los últimos meses ayudándome a llegar a la meta, siempre viéndole el lado positivo y práctico de las cosas, gracias MAESTRO.

A los compañeros y amigos que esta casa de estudios me permitió conocer y quienes siempre me ayudaron y tuvieron palabras de ánimos en este viaje cuya travesía compartíamos, al Lcdo. Antonio Patiño, Daniel Villalba, Carlos López, Hansell Rojas, Aarón Pérez, Francelys Guerra y a Stephanie Contreras.

A los viejos amigos, esos que sin entender nada de lo que les explicaba me decían “tranquila, ya falta poco” (cuando en realidad faltaba mucho) a Alejandra López, Michelle Vall, Emily Rengel y Luis José Romero.

A Carlos G. Vicci, per essere così presente, per essere la mia spalla negli ultimi mesi, per credere in me e darmi coraggi, eternamente gradito.

Por último pero no menos importante a mi familia, a mi tía Norma y mi tío Antonio, mi papá Güicho, mis primos, gracias por su apoyo incondicional, a mi madre, tarde pero seguro, lo logré.

ÍNDICE

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

RESUMEN

El presente trabajo da cuenta del proceso de desarrollo de un enlace de *Volunteer Computing* 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 y almacenamiento; en donde pudo apreciarse carencia de tecnología que le permita trabajar dentro del Núcleo, lo cual conllevó, mediante la aplicación de la metodología de análisis y diseño de redes de computadores planteada por James McCabe (1998), a determinar los requerimientos de los usuarios, con el objetivo de ofrecer soluciones adaptables a las necesidades de los mismos y a las tecnologías disponibles para el desarrollo de este proyecto; se evaluó el flujo de datos de los servidores del *Volunteer Computing* para determinar la manera como se divide la aplicación entre el conjunto de computadoras que forman dicha plataforma. Se trabajó con el diseño lógico y direccionamiento existente en el Núcleo específicamente en el Programa de la Licenciatura en Informática, por último se realizaron los 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. El trabajo propuesto pretende implementar una solución que aproveche los ciclos de cómputo no utilizados de las computadoras de escritorio en laboratorios informáticos y demás computadoras personales que se encuentren dentro de la Universidad, y con ello resolver problemas computacionalmente intensivos, ayudando y promoviendo a investigadores que necesiten gran capacidad de cómputo y procesamiento, para que puedan empezar proyectos de investigación propios o concluir de manera efectiva los ya empezados por ellos.

Palabras claves: *Volunteer Computing, Grid Computing,* BOINC, Computación Paralela, Virtualización.

INTRODUCCIÓN

Desde el surgimiento de la primera computadora se hizo evidente que el poder de cálculo de ésta no era suficiente para dar respuesta en un tiempo razonable a múltiples problemas que por su naturaleza presentan un elevado costo computacional o que deben manejar enormes volúmenes de datos. El enfoque tradicional para lidiar con estos inconvenientes ha seguido un modelo centralizado basado en costosas supercomputadoras. Sin embargo durante la última década se ha podido apreciar un notable incremento en el rendimiento de las redes de computadoras como resultado del desarrollo de hardware cada vez más rápido y software más eficiente, lo cual ha permitido que empezaran a surgir otras alternativas distribuidas capaces de obtener rendimientos comparables a los ofrecidos por los modelos centralizados más avanzados pero a un costo mucho menor, dando lugar a lo que se conoce como computación distribuida (Abreu y cols., 2006).

La computación distribuida consiste en un modelo de computación en paralelo donde típicamente intervienen dos o más computadoras que se comunican a través de una red para cumplir una tarea u objetivo común. El tipo de hardware, lenguajes de programación y sistemas operativos pueden ser muy variados (Abreu y cols., 2006).

Estos esquemas de computación facilitan la capacidad de procesamiento dentro de la organización. En el mundo globalizado actual la información transciende mucho más allá de los límites geográficos de un país o de una región, permitiendo que las organizaciones de todas partes del mundo, sean capaces de compartir información y recursos entre ellas, gracias a las bondades que ofrecen la red de computadoras conectadas que permiten ejecutar procesos de forma distribuida (Suppi, 2003).

El *Grid Computing*, es un modelo de computación distribuida que permite resolver problemas demasiado grandes para cualquier simple computadora o incluso supercomputadora. Motivado a la necesidad que tienen las universidades de implementar mecanismos que le permitan procesar y optimizar gran cantidad cálculo el *Grid Computing* se ha convertido en una eficaz alternativa a las limitaciones actuales de

hardware para procesar los grandes volúmenes de información que actualmente demanda el auge científico. El *Grid Computing* está destinado a la resolución de problemas que requieren de almacenamiento masivo o cálculo intensivo. Muchas de estas aplicaciones tienen que ver con el desarrollo de modelos computacionales para la resolución de problemas de pronósticos meteorológicos, aplicaciones de la computación gráfica al diagnóstico por imágenes de alta resolución, el cálculo de números irracionales con millones de dígitos, la criptografía con números primos del orden de los 400 dígitos, entre otros (Aguiar, 2005).

A pesar de la invención de costosas computadoras con gran cantidad de núcleos de procesamiento, memoria y capacidad de almacenamiento (Golatch y cols., 2008), muchos de estos recursos son desperdiciados, ya que la cantidad de carga computacional, que generan los usuarios, no siempre aprovecha todo el potencial que tienen las computadoras de escritorio.

Por lo general el recurso ocioso en las computadoras, en promedio, es del 85% durante el día y del 95% durante la noche, además, se presenta un uso de procesador menor al 5% (Foster y Kesselman, 2003). En los últimos años, el *Volunteer Computing* han emergido como un importante método para utilizar recursos ociosos de computadores de escritorio, y además han sido orientadas hacia computación de alto rendimiento, para procesar una gran cantidad de tareas independientes en aplicaciones científicas o empresariales (Gorlatch y cols., 2008).

Aunque individualmente, las computadoras personales son inferiores en muchos aspectos a los supercomputadores, el poder combinado de cientos de computadoras personales, representa un recurso computacional importante (Ghazali y Zomaya, 2007).

El propósito de este proyecto es aprovechar la disponibilidad de tiempo ocioso de las computadoras personales o de escritorio ubicados en los laboratorios del Programa de la Licenciatura en Informática de la Universidad de Oriente Núcleo de Sucre (UDO-NS) bajo un sistema de *Volunteer Computing*, además se pretende aprovechar los recursos no utilizados mientras existan alumnos trabajando en cada uno de estas computadoras,

orientando el uso de las mismas como soporte para el desarrollo de proyectos científicos dentro de la UDO-NS, que necesiten realizar procesos computacionalmente intensivos. El mismo se estructura de la siguiente manera:

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: en este capítulo se resaltan los antecedentes de la investigación y la organización, el área de estudio y el marco metodológico que describe la metodología de la investigación y la metodología del área aplicada en esta investigación.

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

CAPÍTULO I. PRESENTACIÓN

PLANTEAMIENTO DEL PROBLEMA

Las universidades por ser centros donde se hace investigación necesitan contar con la infraestructura tecnológica de punta que permita procesar los altos requerimientos de cómputo y almacenamiento que sus investigaciones requieren. La Universidad de Oriente no escapa a esta necesidad y en el caso particular del Núcleo de Sucre no cuenta con los equipos adecuados que permitan realizar el almacenamiento y procesamiento que involucren gran cantidad de datos. Según indagación preliminar realizada, actualmente un grupo de profesores de las áreas de Física, Química y Matemáticas se encuentran trabajando con cómputos distribuidos, así como también en algunas dependencias adscritas a la universidad como son el Centro de Física Teórica y el Laboratorio de Materiales. Estos investigadores ejecutan los cálculos de sus trabajos científicos en computadoras personales, los cuales se tardan hasta 10 días para realizar cómputos tales como: simulaciones, elaboración de modelos matemáticos dinámicos, cálculo de valores utilizando punto flotante y alta precisión, limitándolos en el desarrollo de sus investigaciones y retrasando el proceso investigativo y por ende la producción científica. Es por ello que surge la necesidad de implementar mecanismos que permitan incrementar la capacidad de cómputo y almacenamiento en el menor costo posible haciendo uso de la tecnología disponible en el Programa de la Licenciatura en Informática sin necesidad de adquirir nuevos equipos que conllevaría además a un inversión muy costosa, uno de estos mecanismos fue el desarrollo de un enlace de *Volunteer Computing* en la UDO-NS, que permite poseer una estructura de recursos virtualizados para optimizar los procesos, con el fin de obtener un mayor poder de procesamiento, aplicaciones, almacenamiento y recursos de red, con el cual no solo podrá hacer uso de los ciclos ociosos de las computadoras que se encuentran dentro de Núcleo sino también las que se encuentren fuera y deseen unirse a los proyectos.

ALCANCE Y LIMITACIONES

Alcance

El alcance de este proyecto se basó en el diseño y la implementación de un enlace de *Volunteer Computing* que apoye el desarrollo de investigaciones científicas que requieran gran cantidad de cómputo y almacenamiento.

Limitaciones

Poca documentación para desarrollar aplicaciones y proyectos propios bajo la plataforma BOINC.

CAPÍTULO II. MARCO DE REFERENCIA

MARCO TEÓRICO

Antecedentes de la investigación

El *Volunteer Computing* ha permitido conectar entornos de ejecución, redes de alta velocidad y bases de datos distribuidos geográficamente, permitiendo obtener una potencia de procesamiento y con excelentes resultados (Suppi, 2003), *SETI@Home* es el proyecto que realmente popularizó la computación distribuida y el computo voluntario, *SETI* por sus siglas en inglés, Búsqueda de Inteligencia Extraterrestre, es el proyecto por cuya plataforma *Berkeley Open Infrastructure for Network Computing* **(**BOINC) fue creado, para salvaguardar algunos aspectos de seguridad y de control de los resultados que eran enviados por los usuarios pertenecientes a la red.

SETI es una red científica liderada por David Anderson, que también lidera el equipo encargado de BOINC, cuya meta es la detección de vida inteligente fuera de nuestro planeta. El proyecto posee radio telescopios que monitorean el espacio captando ondas de radio de banda estrecha, las cuales no ocurren de forma natural, por lo que su detección sería una evidencia de tecnología extraterrestre.

Las señales de los radiotelescopios consisten en ruidos provenientes de fuentes celestiales y señales producidas por el hombre. Los proyectos de radio *SETI* se encargan del análisis de estas señales, las cuales se procesan digitalmente, por lo que una potencia mayor de cálculo permite que la búsqueda cubra un mayor rango de frecuencias con una mayor sensibilidad.

Inicialmente estos cálculos se realizaban mediante computadoras en los mismos radiotelescopios, las cuales se encargaban de procesar la mayor cantidad de información. En 1995, David Geyde, propuso que radio *SETI* tuviera una supercomputadora virtual conectada a través de Internet. El proyecto *SETI@Home* fue lanzado en mayo de 1999.

El proyecto *SETI@Home* posee diversos recursos computacionales, como servidores de base de datos para información de usuarios, listas, unidades de trabajo, resultados, información procesada; servidores web para el hosting de la web del proyecto. Cada uno de estos servicios está dividido en servidores, la mayoría de los cuales utilizan tecnología Intel (Anderson y cols., 2002).

EINSTEIN@Home: Al igual que *SETI@Home, EINSTEIN@Home* es un proyecto de computación voluntaria que analiza ondas gravitacionales producidas por fuentes de ondas continuas. El nombre proviene del científico alemán, Albert Einstein, quien a principios del siglo XX predijo la ocurrencia de estas ondas gravitacionales.

El proyecto fue lanzado oficialmente el 19 de Febrero del 2005 como contribución por parte de la Sociedad Americana de Física para el *World Year of Physics* 2005.

El objetivo científico del proyecto es la búsqueda de fuentes de radiación gravitacional de onda continua. El éxito en la detección de ondas gravitacionales constituiría un hito importante en la física, ya que nunca antes se ha detectado un objeto astronómico únicamente por radiación gravitacional.

La información es obtenida mediante dos fuentes, por el *Laser Interferometer Gravitational-Wave Observatory*, y mediante GEO 600, otro detector de ondas gravitacionales (Anderson y cols., 2002).

Rosetta@Home es un proyecto orientado a determinar las formas tridimensionales de las proteínas a través de investigaciones científicas experimentales que a la larga podrían llevar a descubrir curas para las más grandes enfermedades humanas, como el VIH, la malaria y el cáncer (Anderson y cols., 2002).

Ibercivis es un proyecto complejo y multidisciplinar que engloba distintas dimensiones de la actividad científica. Es, al mismo tiempo, una plataforma de computación voluntaria basada en BOINC y con una infraestructura distribuida que soporta múltiples aplicaciones; una herramienta de difusión de la ciencia, la tecnología y de la

metodología científica; y un lugar de encuentro de investigadores en Tecnologías de la Información y la Comunicación, Física, Bioquímica, Matemáticas, etc. con objetivos y tareas comunes (Tarancón, 2009).

Otro proyecto que pone en práctica el uso de la tecnología *Grid Computing*, es *Grid5000*, cuya finalidad es la construcción de una plataforma experimental que abarca 8 sitios distribuidos geográficamente en Francia. El principal objetivo de esta plataforma es la de servir como banco de pruebas experimentales para la investigación en *Grid Computing* (Suppi, 2003).

El Centro Europeo para la Investigación Nuclear (CERN) es una de las instituciones que ha puesto en marcha el desarrollo de un proyecto con la tecnología, titulado *DataGrid*, que tiene como objetivo unir a grandes bases de datos y usuarios en una red informática de alta velocidad. *DataGrid* se define como una alternativa para superar los límites de capacidad de tratamiento de datos que padece la *World Wide Web* y su lentitud debido a la multiplicación del número de usuarios. Según el comunicado del CERN, *DataGrid* ofrece a los científicos de todo el mundo y a todos los usuarios, un acceso rápido a los recursos informáticos (Suppi, 2003).

En Venezuela son pocas las universidades que cuentan con tecnologías de gran capacidad de procesamiento para atender estos requerimientos, entre otras razones, por las limitaciones presupuestarias que éstas tienen. Por ello el *Grid Computing* ha sido considerado como una alternativa económica para ofrecer capacidad de cómputo. Entre las universidades que han implementado esta tecnología se encuentra la Universidad de los Andes, quien en el año 2007 firmó un convenio con la empresa IBM de Venezuela para poner en marcha el programa *Grid Computer* ULA, enfocado en el mejoramiento de la investigación en el área de la tecnología, como soporte en programas desarrollados en diversas áreas del saber (Universidad de los Andes, 2008).

En el caso particular del Programa del Licenciatura en Informática en el año 2009 se hizo un intento de desarrollar un *ClusterBeowulf* enmarcado en el trabajo de grado del Br. Ranses José Jiménez Maza intitulado "Evaluación del rendimiento de la plataforma

de *Clustering Beowulf-OpenMosix"* el cual no obtuvo los resultados esperados de acuerdo a las conclusiones mostradas en el mismo.

Antecedentes de la organización

La UDO fue creada el 21 de noviembre de 1958, mediante el Decreto Ley No. 459 dictado por la Junta de Gobierno presidida por el Dr. Edgard Sanabria, siendo Ministro de Educación el Dr. Rafael Pizani, bajo la conducción de su Rector fundador Dr. Luis Manuel Peñalver.

La Universidad comienza sus funciones el 12 de febrero de 1960 en Cumaná, con los Cursos Básicos; en octubre de 1962 con la Escuela de Medicina y la Escuela de Geología y Minas, en el Núcleo de Anzoátegui comenzaron el 9 de enero de 1963 con la Escuela de Ingeniería y Química, en el Núcleo de Nueva Esparta se iniciaron con los 5 Núcleos que la conforman.

El Programa de la Licenciatura en Informática, adscrito a la Escuela de Ciencias del Núcleo de Sucre de la Universidad de Oriente, cuya creación fue aprobada en 1989, pero es solo en el primer trimestre de 1993 cuando dio inicio a las actividades académicas, con dos secciones y un total de 85 estudiantes.

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), almacenamiento de archivos, alojamiento de aplicaciones, alojamiento de sitios Web, etc (Cisco CCNA *Exploration* 4.0, 2008).

Área de investigación

Este proyecto se encuentra enmarcado dentro del área de la computación distribuida.

Virtualización

La virtualización se refiere a integración uniforme de sistemas heterogéneos geográficamente distribuidos, permitiéndoles a los usuarios hacer uso de los servicios de forma transparente, es decir; que el usuario no necesita estar enterado donde está localizado los recursos computacionales que lo van a satisfacer (Barrera, 2006).

Máquina Virtual

Una máquina virtual (VM) es un contenedor de *software* aislado que puede ejecutar su propio sistema operativo y aplicaciones como si fuera una computadora física (El Rafaey, 2009). Para ejercer cierto control y administración se requiere de un "Administrador de máquinas virtuales" (VMM), se define como el enlace entre el *Gateway* y los recursos, el *Gateway* no comparte recursos físicos directamente, pero depende de la tecnología de virtualización para abstraerlos.

Asimismo el VMM depende del "Motor de infraestructura virtual" (VIE) para administrar máquinas virtuales en un conjunto de recursos físicos. Generalmente los VIE son capaces de crear y detener máquinas virtuales en un clúster físico (Di Constanzo y cols., 2009).

Servidor virtual

Es una tecnología que permite dividir el hardware de un servidor en distintas partes (máquinas virtuales) cada una independiente completamente, de esta manera permite la instalación de múltiples máquinas virtuales y sistemas operativos corriendo de forma concurrente en un mismo servidor (Di Constanzo y cols., 2009).

Middleware

Es un software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, software, redes, hardware y/o sistemas operativos. Éste simplifica el trabajo de los programadores en la compleja tarea de generar las conexiones que son necesarias en los sistemas distribuidos. De esta forma se provee una solución que mejora la calidad de servicio, seguridad, envío de mensajes, directorio de servicio, etc. (Bishop y Karne, 2010).

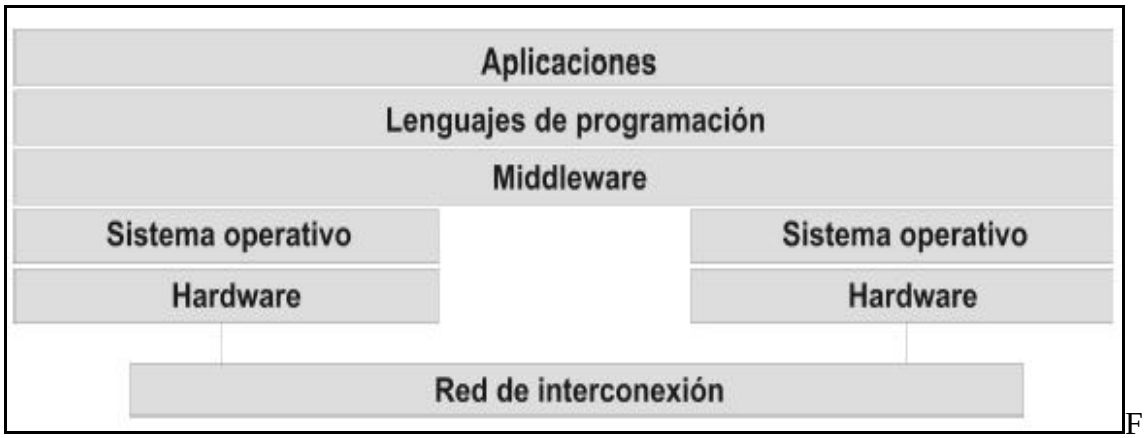


Figura 1. Concepto de *Middleware.*

Diversos autores han escrito acerca de los tipos o categorías de *Middleware,* lo que hace muy amplia la gama de clasificaciones de los *Middleware*; sin embargo a continuación se presenta la dada por Ariannejad (2004) quien hace una clasificación bastante completa, donde cada categoría está incluida en algún punto de la definición dada anteriormente:

Middleware Distributed Tuples (DT)

Es el *Middleware* para acceso a base de datos que permite desarrollar sistemas independientes del manejador de base de datos que lo soporta. Por ejemplo el *framework* Linda.

Remote Procedure Call (RPC)

Es el *Middleware* diseñado como servicio síncrono para permitir la gestión remota de redes. Esconde las operaciones de envío y recepción bajo el aspecto de una llamada convencional a una rutina o procedimiento. Los RPC tienen la misma semántica que las llamadas a procedimientos ordinarios; es decir, se realiza la llamada y se pasa el control al procedimiento servidor; cuando éste devuelve el resultado, el cliente recupera el control. El *software* que soporta RPC debe ocuparse de tres tareas importantes: la interfaz del servicio, la búsqueda del servidor, y la gestión de comunicación.

Messaging Oriented Middleware (MOM)

Es el *Middleware* orientado a mensajes; está diseñado para el servicio de mensajes con tecnología asíncrona. Permite el envío de mensajes entre aplicaciones, las aplicaciones sólo ponen y sacan mensajes de las colas, no se conectan. El cliente y el servidor pueden ejecutarse en diferentes tiempos (mensajes asíncronos), por lo que no necesariamente se requiere respuesta.

Distributed Object Middleware (DOM)

Es el *Middleware* para tecnologías orientadas a objetos; los objetos piden servicio a otros objetos que se encuentran en la red. Se encarga de establecer comunicación entre los clientes y los objetos de forma transparente respecto a la distribución. Permite localizar a un objeto remoto dada una referencia a ese objeto. El núcleo de estos *Middleware* es *Object Request Broker* (ORB). Ejemplo de este *Middleware* son: *Common Object Request Broker Architecture* (CORBA) de OMG, *Remote Method Invocation* (RMI) de SUN *Microsystems* y *Distributed Component Object Model* (DCOM) de *Microsoft*®.

Common Object Request Broker Architecture (CORBA)

Es un modelo de soporte para la programación distribuida orientada a objetos. Hace posible que los objetos interactúen a través de lenguajes de programación, protocolos de comunicación y plataformas heterogéneas. Este modelo no especifica cómo hacer el soporte, sino qué debe hacer, basado en cinco aspectos de los sistemas distribuidos: *Interface Definition Lenguaje* (IDL), que permite la descripción de la interfaz que ofrece un objeto; *CorbaServices* (servicios CORBA), complementan a los objetos que sirven para la construcción de aplicaciones; *CorbaFacilities* (facilidades CORBA), cubren servicios de alto nivel, como interfaces, administración de sistemas y redes; *CorbaDomains* (interfaces de dominio CORBA), proveen funcionalidad a usuarios finales en áreas de interés particular y *General Inter-ORB Protocol* (GIOP) que define

los mensajes y el empaquetado de datos que se transmiten entre objetos. Además, define su implementación sobre otros protocolos.

Remote Method Invocation (RMI)

Posee la misma finalidad que el RPC, invocar de la manera más transparente posible un servicio en una máquina virtual distinta a la que reside el cliente (en la misma máquina física pueden existir varias máquinas virtuales de Java). La diferencia entre estas dos tecnologías radica en que RPC se utiliza en diseños no orientados a objetos, mientras que RMI está soportado por el lenguaje orientado a objetos Java. RMI es un Middleware específico que permite a clientes invocar métodos de objetos como si estuviesen en la misma máquina virtual.

Distributed Component Object Model (DCOM)

Al igual que CORBA y RMI, permiten la comunicación de objetos, pero únicamente para las diferentes versiones del sistema operativo *Windows*®. DCOM surge como evolución de *Object Linking and Embedding* (OLE) y *Component Object Model* (COM).

Transaction Processing Monitors (TP Monitors)

El *Middleware* para Procesamiento de Transacciones ya que facilita la conectividad y el acceso a un gran número de usuarios con servicios de *back-end* limitados. Este tipo de *Middleware* requiere del soporte de un Monitor; es decir, un programa que supervise las transacciones entre procesos, con el propósito de asegurar el éxito de la transacción, o en caso de ocurrir un error, tomar acciones apropiadas. Su principal uso es coordinar el flujo de solicitudes entre los dispositivos y las aplicaciones que procesan esas solicitudes.

Database Access Technology (DBAT)

Son las *Aplication Programming Interface* (API) creando una capa transparente para el acceso a base de datos, ocultando la complejidad dada por el manejador de base de

datos. Ejemplo de estas API son *Java Database Connectivity* (JDBC) desarrollado por SUN y *Open Database Connectivity* (ODBC) desarrollado por *Microsoft*®.

Java DataBase Connectivity (JDBC)

Es una API que facilita programar el acceso a bases de datos para Java sin que se tenga en cuenta a que Servidor se dirige (*SQLServer, Oracle, Sybase, Informix*, etc.). JDBC hace tres cosas: establece la conexión con una BD, envía sentencias SQL y procesa los resultados.

Open Database Connectivity (ODBC)

*E*s una API abierta para acceder a bases de datos. Especifica un conjunto de funciones para manejar conexiones a bases de datos, ejecutar declaraciones SQL (*Structure Query Language*) y consultar las capacidades del sistema de base de datos.

Component Oriented Framework (COF)

Este *Middleware* está soportado en el modelo de desarrollo de aplicaciones basado en componentes, permite la creación de una aplicación como conjunto de componentes reusables. Los componentes son una evolución del software orientado a objetos para dar respuesta a la reusabilidad. Los *Middleware* basados en componentes permiten a éstos "pegarse" con otros componentes para lograr la integración. Algunos ejemplos son: *Enterprise Java Beans* (EJB), especificación creada por SUN *Microsystems*; define un *framework* para los componentes Java del lado del servidor. COM+ es la siguiente generación de DCOM; simplifica la programación y es igualmente diseñado por *Microsoft*®. *CORBA Component Model* (CCM).

Directory Services (DS)

Es el *Middleware* que se basa en directorios que permiten reducir costos administrativos, simplifican y/o distribuyen tareas administrativas, reducen el número de *passwords* para un usuario mediante una única combinación de *login/password*, aumentan la seguridad y

proporcionan una única localización para la información de los usuarios. Son similares a la bases de datos, pero contienen información más descriptiva; la frecuencia de lectura sobre ellos es bastante más alta que la de escritura y en consecuencia su manejo es más simple que el de una base de datos ya que no hace actualizaciones complejas. *LDAP Lightweight Directory Access Protocol* (LDAP) es un protocolo para acceder a los DS. Como ejemplo, se tienen los *Domain Name System* (DNS) que conforman una fuente de datos distribuidas por múltiples máquinas en Internet, cuya tarea es convertir nombres en direcciones.

Application Servers (AS)

Es el *Middleware* que se enfoca en la parte de la de aplicación o lógica del negocio. Conceptualmente un sistema puede tener muchas capas; sin embargo, la arquitectura más popular es de tres capas: interfaz, aplicación y base de datos.

Como se aprecia, las tecnologías *Middleware* fundamentalmente dependen de los mecanismos que permiten la interconexión de los sistemas. Es decir, todas las tecnologías *Middleware* permiten comunicación. La diferencia entre ellas se encuentra en el énfasis que presentan como apoyo en aspectos como datos y procesos (Bakken, 2003).

Certificado digital

Es un punto de unión entre la clave pública de una entidad y uno o más atributos referidos a su identidad. El certificado garantiza que la clave pública pertenece a la entidad identificada y que la entidad posee la correspondiente clave privada (Talens-Oliag, 2008).

Los certificados digitales sólo son útiles si existe alguna Autoridad Certificadora (*Certification Authority* o *CA*) que los valide, ya que si uno se certifica a sí mismo no hay ninguna garantía de que su identidad sea la que anuncia, y por lo tanto, no debe ser aceptada por un tercero que no lo conozca.

Los certificados digitales proporcionan un mecanismo criptográfico para implementar la autentificación; también proporcionan un mecanismo seguro y escalable para distribuir claves públicas en comunidades grandes.

Cluster

Se denomina *Cluster* (agrupación) a un grupo de computadoras que trabajan con un fin común. Estas computadoras agrupan hardware, redes de comunicación y *software* para trabajar conjuntamente como si fueran un único sistema. Generalmente, un *Cluster* trabaja sobre una red de área local (LAN) y permite una comunicación eficiente de las máquinas que se encuentran dentro de un espacio físico próximo (Suppi, 2009).

Cluster 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* estándare. Los nodos en el *Cluster* están dedicados exclusivamente a ejecutar tareas asignadas al *Cluster* (Plaza, 2002).

Open Source Cluster Application Resources (OSCAR)

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

La arquitectura de un *Cluster* OSCAR es muy similar a la de de un *Cluster Beowulf*. Se configura un nodo como maestro, el cual provee los servicios a los nodos de cómputo. En el *Cluster 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 *Cluster* (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.

Computación paralela

La computación paralela consiste en agrupar computadores independientes de forma que la imagen de éstos hacia el usuario final sea la de un único computador cuyas propiedades, en la medida de lo posible, sea la suma de las prestaciones de los equipos que formen el sistema paralelo (Morillo, 2003).

Esta agrupación de computadoras tiene como objetivo realizar una tarea o proceso computacionalmente intensivo, y para ello, la tarea necesita ser descompuesta en distintas subtareas independientes, debiendo redireccionar cada una de éstas hacia un computador asociado al proyecto, el cual tenga los recursos necesarios para la resolución

de las mismas. A medida que las subtareas sean completadas por cada uno de las computadoras, se centralizan en uno de ellos para formar la tarea inicial, obteniendo el resultado en un tiempo mucho menor si es que se realizara en una solo computadora (Morillo, 2003).

Así, como al resolver un problema computacional con una sola computadora, en el que se necesita una secuencia ordenada de pasos para la resolución del problema, al cual se denomina algoritmo; en la resolución de un problema utilizando varias computadoras, se requiere también un algoritmo con el cual se resuelve el problema, además de especificar los pasos que pueden ser ejecutados simultáneamente, donde cada uno de ellos debe ser independiente del resto. Esto es esencial para obtener un buen rendimiento a la hora de ejecutar el algoritmo en forma paralela (Grama cols., 2003)

Al momento de especificar un algoritmo paralelo debe tenerse en cuenta los siguientes pasos a realizar (Grama 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, MPI, y BOINC que es una de las más importantes plataformas para el desarrollo de aplicaciones distribuidas hoy en día.

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 (PVM 2.0) (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 “Maquina Virtual”. Un aspecto de la maquina 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 maquina 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.

BOINC, es una plataforma de código abierto que permite a proyectos hacer uso de la capacidad de cómputo de diversos computadores, ubicados alrededor del mundo de manera voluntaria. Es decir, que cuando un computador conectado a la red BOINC se encuentra inactivo, inmediatamente empieza a trabajar para el proyecto al cual se ha unido, procesando los datos que le solicita al proyecto al cual está unido. Estos proyectos usualmente están relacionados a investigaciones científicas de diversos tipos (Anderson y cols., 2002).

Además, esta plataforma puede utilizarse para realizar computación en red dentro de una misma organización, por ejemplo para realizar cálculos propios que la misma empresa necesita dentro de su negocio.

Las características más importantes de BOINC se muestran a continuación:

Autonomía de proyectos: varios proyectos pueden utilizar BOINC para realizar el procesamiento, cada uno de los cuales se maneja como un proyecto independiente, el cual posee su propio servidor y base de datos.

Flexibilidad del voluntariado: los voluntarios pueden participar en múltiples proyectos, dependiendo de sus intereses, y además, decidir la cantidad de recursos donados a cada uno de los proyectos en los cuales participa.

Multiplataforma. El BOINC cliente es encontrado para múltiples plataformas, como Windows, Linux y MacOS.

BOINC ha sido diseñado para soportar aplicaciones que necesiten grandes cantidades de recursos de cómputo, siendo el único requerimiento que esta aplicación sea divisible en

gran cantidad de tareas pequeñas que puedan ser realizadas, cada una, de manera independiente (Anderson y cols., 2002).

BOINC *Manager*

Es un programa que se instala directamente en el host y permite indicar los parámetros de configuración directamente a la máquina. Además, permite añadir proyectos que se encuentren disponibles. El cliente BOINC-*Manager* es multiplataforma y puede correr bajo cualquier arquitectura (Anderson y cols., 2002).

El usuario puede elegir si BOINC-*Manager* solo debe trabajar cuando el mouse / teclado están inactivos, a qué hora puede trabajar, cuanto espacio de disco y ancho de banda se le permite usar.

BOINC *Wrapper*

Es una de las formas de procesar una aplicación existente bajo BOINC, en cualquier archivo o secuencias de archivos ejecutable sin modificación del código fuente de la aplicación, trabajándola como subproceso y manejando toda la comunicación entre el cliente y el servidor (Anderson y cols., 2002).

Work Unit (WU)

Una WU define la aplicación y el conjunto de datos que tienen que ser ejecutados y procesados por el cliente (Anderson y cols., 2002).

Core Client

Es un servicio (o demonio, dependiendo de la plataforma) que hace la conexión con el servidor y necesita instrucciones previas. Dichas indicaciones de funcionamiento pueden ser asignadas a través del BOINC-*Manager* (Anderson y cols., 2002).

MARCO METODOLÓGICO

Metodología de la Investigación

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.

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

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 enlace de *Volunteer Computing*.

Técnicas para la recolección de datos

En la recolección de la información necesaria para desarrollar esta investigación se realizaron encuestas y entrevistas a los posibles usuarios del *Volunteer Computing*, de igual manera se utilizó las técnicas de consultas bibliográficas y consultas en Internet, lo cual permitió establecer el soporte teórico de la investigación.

Metodología del área aplicada

Para la realización de este trabajo se tomó como referencia la metodología descrita en el análisis y diseño de redes de computadoras propuesta por Mc Cabe (1998), cuyo proceso general se describe a continuación:

Análisis de requerimientos

Consiste en identificar, capturar y comprender las necesidades de los usuarios, 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.

Análisis de flujo

En esta fase se especifica la información y las características de las aplicaciones y protocolos que se emplean para la transmisión de información en la red.

Diseño lógico

Esta etapa especifica la topología lógica seleccionada, la cual muestra de qué forma las redes WAN y LAN se interconectan y a través de cual medio. Para la realización del diseño lógico se debe establecer metas de diseño como especificación de flujo y requerimientos, en particular el presupuesto. Desarrollar un criterio para evaluación de tecnologías: costo, rapidez, confiabilidad, movilidad, a través de cuadros comparativos se realizará la selección de la tecnología, analizando la información disponible en el mercado.

Diseño físico

Para esta fase se especifica la topología física a diseñar, en donde se describe mediante un esquema, los equipos de interconexión de la red y su distribución. Para la realización del diseño físico se debe: seleccionar la ubicación de los equipos y realizar el diagrama físico de la red.

Direccionamiento y ruteo

Por medio de esta fase se calcula el direccionamiento IP para la red y se aplica la estrategia de separación lógica de la red en VLANs, y se documentan los segmentos de red de acuerdo a su ámbito.

Ejecución del diseño

Una vez que se propone el diseño debidamente documentado es posible ejecutarlo, definiendo las estrategias de acuerdo a los recursos humanos y computacionales disponibles bajo la supervisión de expertos en el área de redes, con la finalidad de que la prueba ofrezca resultados óptimos basados en los requerimientos establecidos.

La fase de direccionamiento y ruteo no se llevó a cabo puesto que se usó el direccionamiento establecido en las redes existentes en los laboratorios del Programa de la Licenciatura en Informática de la UDO-NS.

CAPÍTULO III. DESARROLLO

ANÁLISIS DE REQUERIMIENTOS

El desarrollo de esta fase consistió en analizar y describir la situación actual de los profesores e investigaciones que requieren gran cantidad de procesamiento y almacenamiento de datos en sus actividades de investigación; para ello se realizaron entrevistas a diferentes profesores del área de Física, Química y Matemáticas, cuyos resultados indican que en su mayoría carecen que la tecnología que les permita procesar y almacenar el gran volumen de datos que este tipo de trabajos amerita, así mismo algunos expresaron que han logrado construir pequeños *clusters* para trabajos personales, mientras que otros deben acudir a centros como el Instituto Venezolano de Investigaciones Científicas (IVIC) o hacer uso de estas tecnologías en las universidades en las cuales se encuentran haciendo cursos de postgrado (especialización, maestrías y doctorados). Lo que evidencia la necesidad de implementar este tipo de tecnología dentro de las instalaciones de la UDO-NS.

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 Programa de la Licenciatura en Informática de la UDO-NS (*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.

Entre los dispositivos disponibles en el Programa de la Licenciatura en Informática de la UDO-NS se encuentran:

22 computadoras HP Intel Core 2 Duo 2.40 GHz, 2Gb RAM, 160Gb Disco Duro.
6 computadoras HP Intel Core 2 Duo 3.00 Ghz, 2Gb RAM, 250Gb Disco Duro.
20 computadoras Dell Intel Pentium 4 3.00 Gbz, 512Mb RAM, 80Gb Disco Duro.
20 Laptop Dell Inspiron 1300 Intel Pentium M 1.7GHz. 512Mb de RAM, 40Gb Disco Duro.

2 Servidores HP PROLIANT ML 110 G5 Procesador XEON de 2.3 GHz, 4Mb Cache, 2Gb RAM, 500Gb Disco Duro

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

1 *Switch TRENDNET* 24 puertos a 10/100Mbps

1 *Switch Advantek Networks* 24 puertos a 10/100Mbps *Fast Ethernet* y 2 puestos a 10/100/1000MMbsp

Una vez analizada la diversidad de quipos disponibles en el Programa de la Licenciatura en Informática de la UDO-NS, se estudiaron los diferentes esquemas de cómputos para determinar cuál de ellos era el que mejor se adaptaba a la gran variedad de equipos disponibles.

Los esquemas de cómputos analizados para el desarrollo de este proyecto se ven reflejados en la tabla que a continuación se presenta:

Tabla 1. Requerimientos de Esquema de Cómputo.

Esquema de Cómputo / Requiere	*Grid Computing*	*Cloud Computing*	*Cluster 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

Los esquemas de computación *Grid Computing* y *Cloud Compuing* fueron descartados ya que uno de los requerimientos para el desarrollo de los mismos es poseer un sistema de certificación digital que actualmente la UDO-NS no posee.

El *Cluster Computing*, requiere de hardware homogéneo que esté 100% dedicado al procesamiento y almacenamiento de tareas, se descartó ya que no se contaba con los

recursos económicos para adquirir los dispositivos que permitieran el desarrollo de esta esquema de computación.

El esquema de computación que mejor se adaptó a los dispositivos que se encuentran disponibles en el Programa de la Licenciatura en Informática fue el *Volunteer Computing*, ya que el mismo permite hacer uso de los ciclos ociosos de las computadoras de los laboratorios sin alterar el trabajo que a diario realizan los estudiantes y profesores que hacen vida en los mismos, no requiere de certificación digital y puede desarrollarse con hardware heterogéneo, por lo tanto resultó ser el esquema de computo más factible a los requerimientos planteados.

Además de los esquemas de cómputo también se estudiaron diferentes plataformas de virtualización que para la implementación del *Volunteer Computing.*

Tabla 2. Requerimientos de Plataforma de Virtualización.

Plataforma de Virtualización / Requiere	Xen	Virtual PC	Vmware	Virtual Box	BOINC
Virtualización por hardware	Si	No	No	No	No
Virtualizacion por software	Si	Si	Si	Si	No
Instalación del SO en las MV	Si	Si	Si	Si	No
Reserva de memoria y almacenamiento	Si	Si	Si	Si	No
Multiplataforma	Si	Si	Si	Si	Si
Capacidad de para-virtualización	Si	No	No	No	No
Soporte Network Bridge	Si	Si	Si	Si	No

La plataforma de virtualización XEN fue descartada ya que requiere un soporte de virtualización por hardware el cual no poseen las maquinas de los laboratorios del programa de la Licenciatura en Informática de la UDO-NS.

En los casos de Virtual PC, VMWare y VirtualBox como el sistema operativo a instalar funciona sobre la máquina virtual y no sobre el hardware de las computadoras, su desempeño es más lento y al momento de realizar cálculos que requirieran gran cantidad de procesamiento el tiempo de respuesta sería más largo, además la implementación de la virtualización sobre cada una de las computadoras de los laboratorios implica un mayor uso de los recursos de los equipos puesto que en cada uno deben reservarse

recursos de almacenamiento y memoria así como de procesamiento para el sistema operativo anfitrión lo que afectaría las actividades regulares que se realizan en las máquinas de los laboratorios, adicional a esto la documentación para construcción de algún esquema de cómputo como este tipo de software es muy escasa.

La plataforma de virtualización que mejor se adaptó al esquema de cómputo *Volunteer Computing* fue BOINC ya que no solo podía hacerse uso de los dispositivos disponibles en los laboratorios de el Programa de la Licenciatura en Informática de la UDO-NS sino que además podían utilizarse las máquinas personales que se encontraran dentro y fuera de las instalaciones de la Universidad, motivado a que es una plataforma de cómputo voluntario, que no requiere características similares de hardware, ni que estén dentro de la misma red, ni es indispensable que tengan el mismo sistemas operativo, pudo instalarse directamente en el hardware de las computadoras sin necesidad de crear una máquina virtual ya que solo debe reservarse el espacio para instalar la aplicación cliente (BOINC-*Manager*), corre directamente en el sistema operativo que reside en las computadoras, sin afectar de forma agresiva sus recursos, por tal motivo no se ve afectado el funcionamiento de las tareas que a diario se realizan en las computadoras ya que BOINC usará el tiempo ocioso de las mismas.

Entre otras ventajas del uso de BOINC como sistemas de cómputo distribuido y voluntario para el desarrollo del *Volunteer Computing* se pueden detallar (Sommerville, 2004):

Recursos compartidos: un sistema distribuido permite compartir tanto software como hardware, mediante computadores asociados a través de la red, en este caso, mediante los computadores asociados al *Volunteer Computing*, cada uno de los cuales brinda capacidad de procesamiento para realizar los procesos requeridos por los investigadores.

Apertura: los sistemas distribuidos están diseñados para soportar equipos de diferentes fabricantes, ya que se basan en protocolos definidos como estándares. Como se ha mencionado anteriormente, la plataforma de BOINC; se encuentra disponible para diferentes arquitecturas y plataformas.

Concurrencia: en un sistema distribuido, muchos procesos pueden operar al mismo tiempo en diferentes computadoras de la red. Para el caso específico de la solución planteada, cada computador asociado al *Volunteer Computing*, se encarga de la ejecución de una parte del proceso.

Escalabilidad: BOINC solo necesita que cada computadora que desea ingresar al *Volunteer Computing* cuente con el BOINC cliente y se una al proyecto deseado, indicando la cantidad de recursos que desea compartir.

ANÁLISIS DE FLUJO

BOINC presenta una arquitectura cliente-servidor, cuya comunicación se soporta bajo el protocolo HTTP, siendo el servidor BOINC quien ofrece los servicios necesarios para que los clientes BOINC puedan realizar el procesamiento requerido. Estos dos componentes son conectados mediante una red de trabajo que permiten a los clientes acceder a los servicios.

Los servicios brindados por el servidor BOINC son la administración de los trabajos, procesamiento de resultados parciales y la administración de cuentas de usuario entre los principales; mientras que el cliente BOINC, que se encuentra corriendo en las computadoras participantes, realiza peticiones o ejecuta una aplicación específica al proyecto al que se encuentra unido.

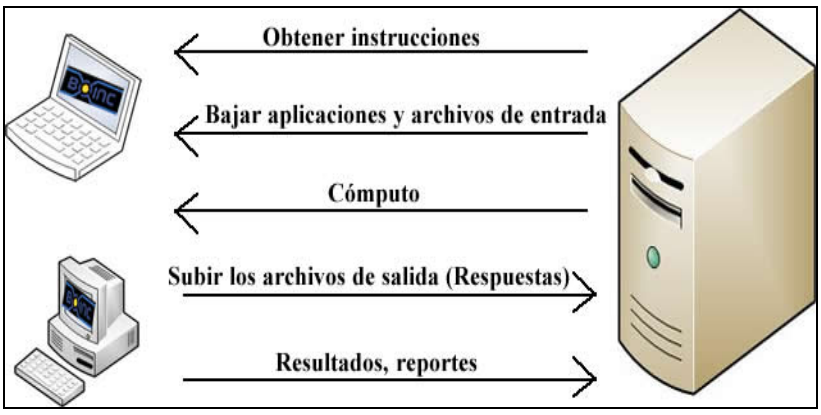


Figura 2. Muestra la interacción entre las computadoras y el servidor.

DISEÑO LÓGICO

En la figura 3 se visualiza el funcionamiento lógico de una tarea en una plataforma BOINC, como se puede observar si el usuario BOINC está ocupado no procesa ningún tipo de tarea pero una vez la computadora no esté realizando alguna actividad el servidor BOINC realiza la asignación de la WU a esa máquina, ésta empieza a realizar los cálculos y una vez culminados los envía al servidor, si durante el periodo de procesamiento de la tarea, la máquina empieza a realizar una actividad adicional se detiene el cálculo.

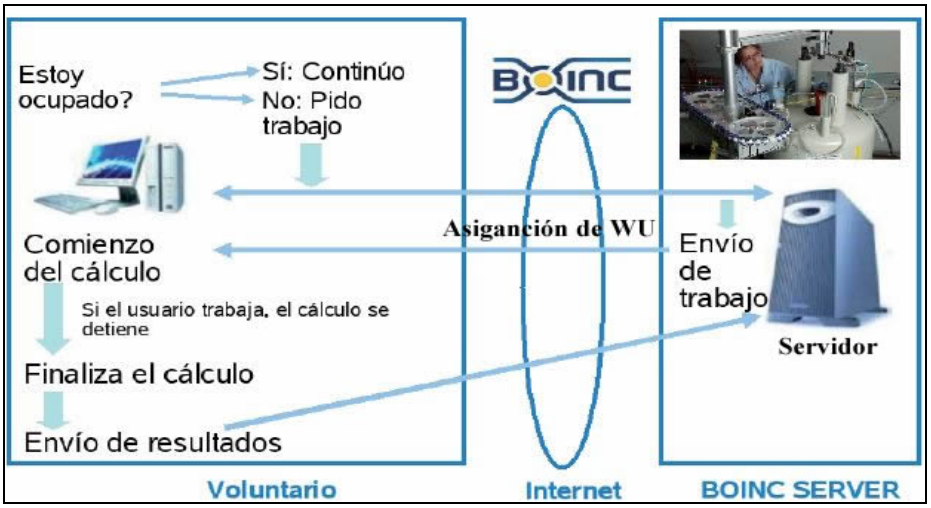


Figura 3. Diseño lógico del BOINC.

DISEÑO FÍSICO

Para el desarrollo de este proyecto se tomaron como equipos de prueba los ubicados en las instalaciones del Programa de la Licenciatura en Informática de la UDO-NS, los cuales están organizados físicamente en forma de estrella extendida, ya que, las computadoras están conectadas a un nodo central el cual a su vez se enlaza con otros nodos centrales. Es de indicar que la configuración del servidor BOINC fue implementado en un servidor virtual, optimizando de esta manera la disponibilidad de recursos de hardware para el desarrollo de este proyecto.

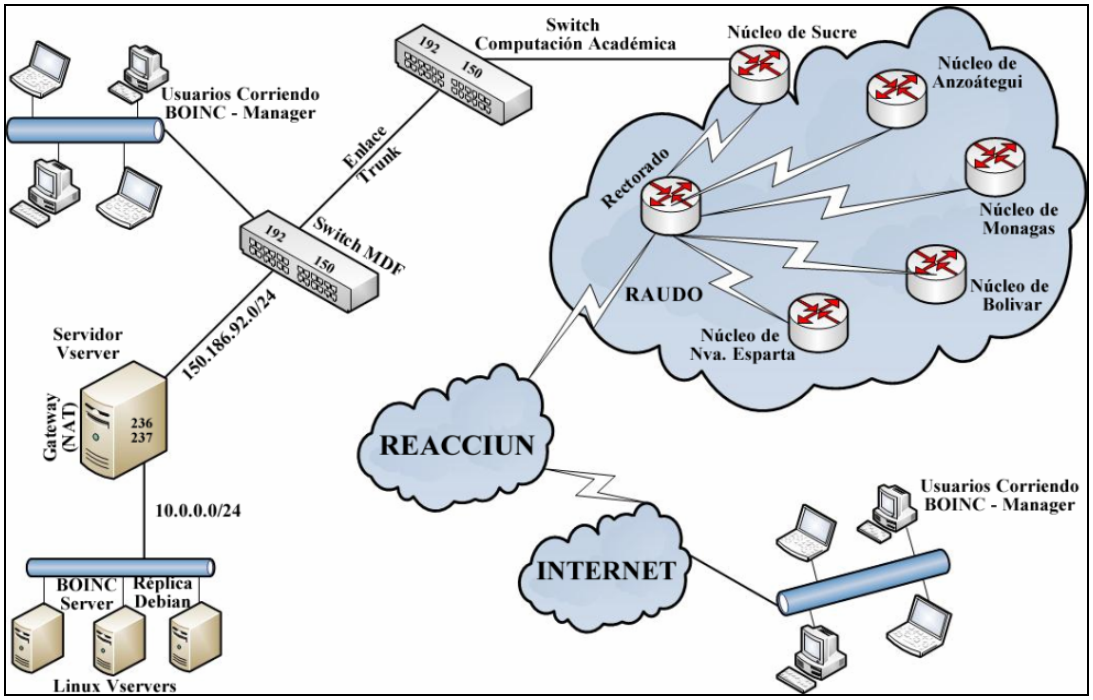


Figura 4. Plano del diseño físico.

EJECUCIÓN DEL DISEÑO

Para efecto de este proyecto como se trabajó con un diseño lógico y físico que ya estaban desarrollados, esta fase se basó en la configuración del servidor BOINC, el cual se configura de la siguiente manera:

Instalar archivos base:

Copiar en la línea de comandos:

```
apt-get update
apt-get install subversion build-essential apache2 php5
mysql-server php5-gd php5-cli php5-mysql python-mysqldb
libtool automake autoconf pkg-config libmysql++-dev libssl-
dev phpmyadmin
```

Definir los *passwords* de root y mysql

Copiar en la línea de comandos:

```
passwd root CONTRASEÑA ROOT
```
Si el *password* no se estableció durante la instalación del mysql-server

Copiar en la línea de comandos:

```
mysqladmin -u root password CONTRASEÑA MSQL
mysqladmin -u root -h host_name password CONTRASEÑA MSQL
```

Solo para efectos de la instalación dentro de servidores virtuales debe asegurarse que los servicios corran en la IP propia del servidor y no en el "localhost", es decir: 127.0.0.1

Reiniciar el gestor de bases de datos: detener los servicios

Copiar en la línea de comandos:

```
/etc/init.d/apache2 stop
/etc/init.d/mysql stop
```

Editar la dirección del servicio de apache2

Copiar en la línea de Comandos:

```
vi /etc/apache2/ports.conf
```

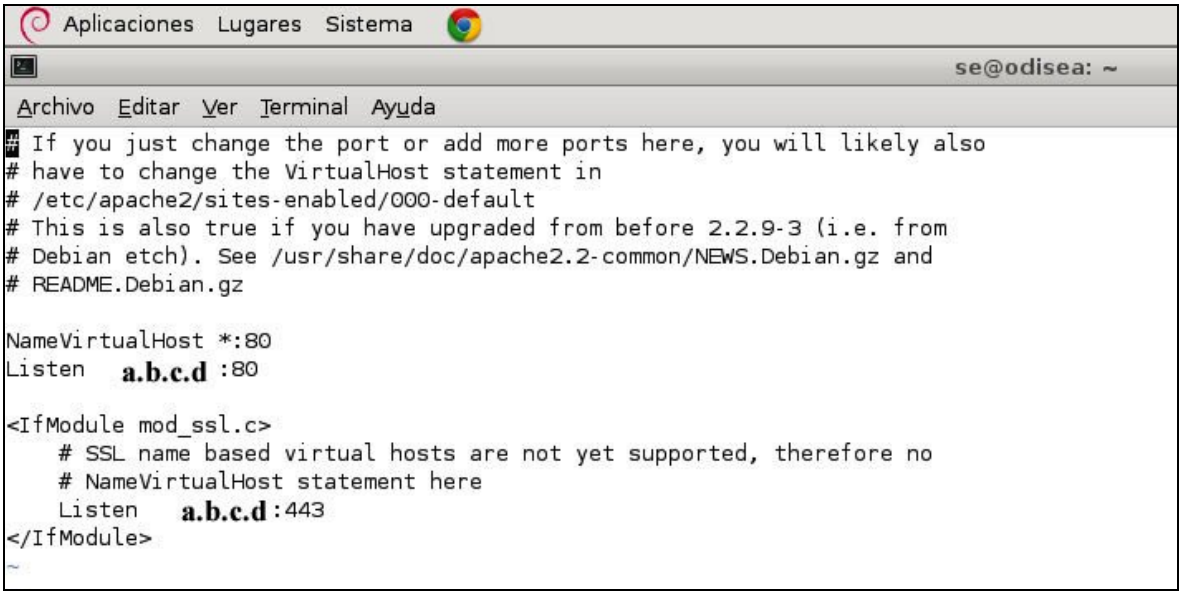


Figura 5. Vista del archivo ports.conf.

Editar la dirección IP del servicio de mysql:

Copiar en la línea de comandos:

```
vi /etc/mysql/my.cnf
```

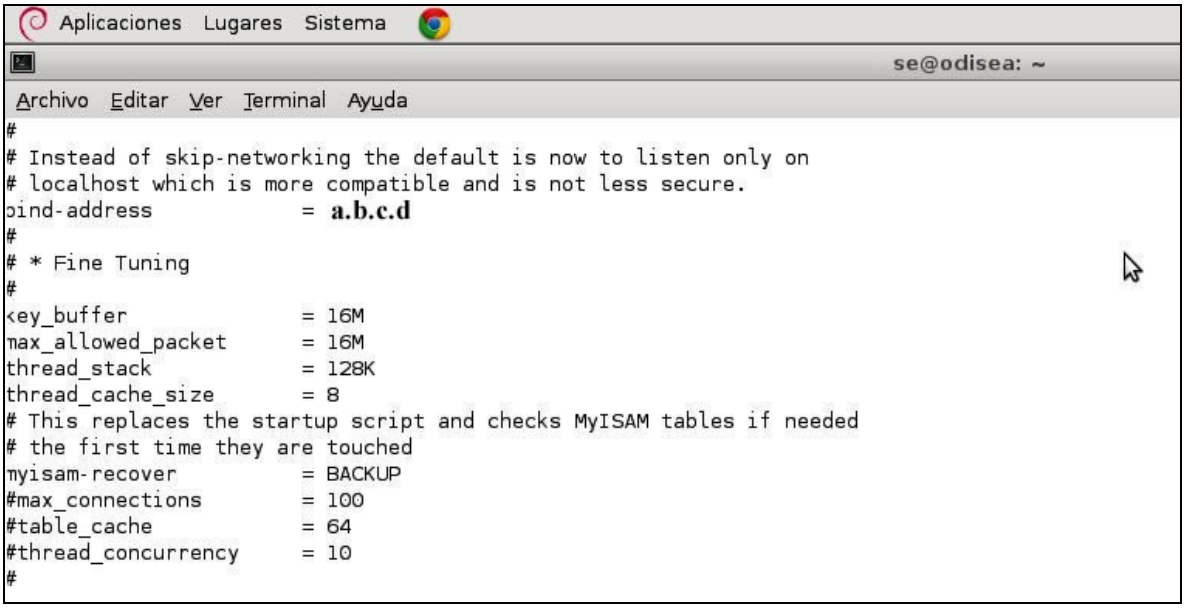


Figura 6. Vista del archivo my.cnf.

a.b.c.d = a la dirección IP elegida

Iniciar los servicios de Apache2 y Mysql

Copiar en la línea de comandos:

```
/etc/init.d/apache2 start
/etc/init.d/mysql start
```

Crear Grupo y Usuario para el BOINC, y añadir www-data a este nuevo grupo

Copiar en la línea de comandos:

```
addgroup --system boincadm
adduser www-data boincadm
```

Se debe definir el directorio donde se procederá a la compilación, en nuestro caso fue:

```
/root/boinc/
```

Copiar a la línea de Comandos:

```
cd /root
```

Descargar el código fuente de BOINC mediante SVN:

Copiar en la línea de comandos:

```
svn co http://boinc.berkeley.edu/svn/branches/server_stable
boinc
```

Compilar el código fuente de BOINC

Verificar si con el comando SVN se crea la carpeta BOINC, en caso contrario copiar en la línea de comandos:

```
mkdir boinc
cd boinc
```

Copiar en la línea de comandos:

```
cd boinc; ./_autosetup; ./configure --disable-client; make
```

Creando un proyecto, como ejemplo tomamos el nombre “boinctest”:

Primero creamos el archivo donde se definirán los parámetros y permisos para el nuevo proyecto:

Copiar en la línea de comandos:

```
touch ~/.boinc_test.conf
chmod 700 ~/.boinc_test.conf
```

Definir los parámetros de acceso a la base de datos dentro del archivo ".boinctest.conf"

El "pw=PASSWORD BD BOINCTEST" la cual es diferente a la definida al usuario root de msql:

Copiar en la línea de comandos:

```
cat << EODBCONFIG >> ~/.boinc_test.conf
# Se define la variable pw con la clave que conceda los
permisos de escritura a la base de datos del proyecto.
pw= PASSWORD BD BOINCTEST
# se define la variable dbprojectname con el nombre de la
Base de datos.
dbprojectname=boinctest
EODBCONFIG
```

Para crear la base de datos del proyecto "boinctest":

Copiar en la línea de comandos:

```
if [ -r ~/.boinc_test.conf ] && . ~/.boinc_test.conf; then
cat <<MYSQLUSER | mysql -u root -p mysql
DROP USER 'boincadm'@'localhost';
MYSQLUSER
echo "If the removal of the previous user fails because the
user is not existing, then this does not matter. Other
errors would be required a manual removal."
if  [ -z "$dbprojectname" ]; then echo "Variable
'dbprojectname' not set";
elif [ -z "$pw" ]; then echo "Variable 'pw' not set";
else
  # piping commands to mysql shell
  cat <<EOMYSQL | mysql -u root -p mysql
DROP DATABASE IF EXISTS $dbprojectname;
CREATE DATABASE IF NOT EXISTS $dbprojectname;
CREATE USER 'boincadm'@'localhost' IDENTIFIED BY '$pw';
GRANT ALL PRIVILEGES ON $dbprojectname.* TO
'boincadm'@'localhost';
EOMYSQL
fi
fi
```

Definir los parámetros que definirán la URL, la carpeta contenedora de los datos y la carpeta donde residirán los códigos fuentes del proyecto.

Copiar en la línea de comandos:

```
cat <<EOCONF >> ~/.boinc_test.conf
# address of host (via DNS or IP number) at which project
server shall be reached
hosturl=http://A.B.C.D
# name of folder in which data shall be stored, also
becomes part of project URL
fileprojectname=$dbprojectname
# more human-compatible way to read the project name
niceprojectname="BoincTestProject@Home"
# location at which sources shall be kept
installroot=/home/cliente/project/boinctest
EOCONF
```

Para verificar que los parámetros establecidos en ".boinc_test.conf" son correctos.

Copiar en la línea de comandos:

```
if [ ! -r ~/.boinc_test.conf ]; then
 echo "Configuration file '~/.boinc_test.conf' not
existing."
else
 . ~/.boinc_test.conf
 if [ -z "$installroot" -o -z "$hosturl" -o -z
"$dbprojectname" -o -z "$pw" \
  -o -z "$niceprojectname" -o -z "$fileprojectname" ]; then
    echo "Variables de Entorno no definidas!!! :(";
 else
   echo "Variables de Entorno definidas!!! :)"
 fi
fi
```

Para crear el proyecto para la aplicación "boinctest"

Copiar en la línea de comandos:

```
cd /root/boinc

if [ -d "$installroot" ] || mkdir -p "$installroot"; then
./tools/make_project --url_base "$hosturl" --db_name
"$dbprojectname" --db_user boincadm --db_passwd "$pw" \--
delete_prev_inst --drop_db_first --project_root
```

```
"$installroot/$fileprojectname" "$fileprojectname"
"$niceprojectname";
fi
```

Para verificar que las variables estén definidas

Copiar esto en la línea de comandos:

```
[ -r ~/.boinc_test.conf ] && . ~/.boinc_test.conf
```

Cambiar los permisos de los directorios dentro del "installroot"

Copiar en la línea de comandos:

```
if [ -z "$installroot" -o -z "$fileprojectname" ]; then
 echo "Not all variables are set for the configuration"
 echo "Error, do not continue."
elif [ ! -d "$installroot"/"$fileprojectname" ]; then
 echo "The directory '$installroot/'$fileprojectname' is
not existing"
 echo "Error, do not continue."
else
 cd "$installroot"/"$fileprojectname"
 chown root:boincadm -R .
 chmod g+w -R .
 chmod 02770 -R upload html/cache html/inc html/languages
html/languages/compiled html/user_profile
fi
```

Copiar a la línea de comandos:

```
cd "$installroot"/"$fileprojectname"
```

Copiar a la línea de Comandos:

```
if [ -d html/inc -a -d cgi-bin ]; then
 chmod  o+x html/inc
 chmod -R o+r html/inc
 chmod  o+x html/languages/
 chmod  o+x html/languages/compiled
else
 echo "You are not in your project directory"
fi
```

Anadir el Proyecto al CRON

Copiar a la línea de Comandos:

```
crontab -e
```

Añada en el editor del cron la siguiente línea:

```
0-59/5 * * * *
/home/cliente/project/boinctest/boinctest/bin/start --cron
```

Guarde las modificaciones

Para verificar si los cambios fueron guardado copiar en la línea de comandos:

```
crontab -l
```

Configurar el *password* de la página de Administración de BOINC:

Copiar a la línea de comandos:

```
cd /home/cliente/project/boinctest/boinctest
htpasswd -c html/ops/.htpasswd boincadm
```

Copiar *password* de su preferencia

Configuración de los alias en Apache para las llamadas del proyecto

Copiar a la línea de comandos:

```
cd /home/cliente/project/boinctest/boinctest
cp ${fileprojectname}.httpd.conf /etc/apache2/sites-
available/
a2ensite ${fileprojectname}.httpd.conf
/etc/init.d/apache2 restart
```

Configuración del sitio web del proyecto "boinctest"

Copiar a la línea de Comandos:

```
cd /home/cliente/project/boinctest/html/project
```

```
vi project.inc
```

Reemplazar la cadena: *"REPLACE WITH PROJECT NAME"* con los valores apropiados al proyecto. Una vez modificado el documento recuerde guardar los cambios.

Pasos para unirse a un proyecto BOINC

Para que un usuario pueda unirse a un proyecto que este ejecutándose con la plataforma BOINC es necesario que tenga los siguientes requisitos:

Instalar BOINC-*Manager* en su computadora, para poder tener acceso al proyecto a ejecutarse (Ver apéndice A).

Ingresar la dirección IP o Web del proyecto al que desea unirse, esto le permitirá que el servidor BOINC le asigne las WU del proyecto a ejecutarse.

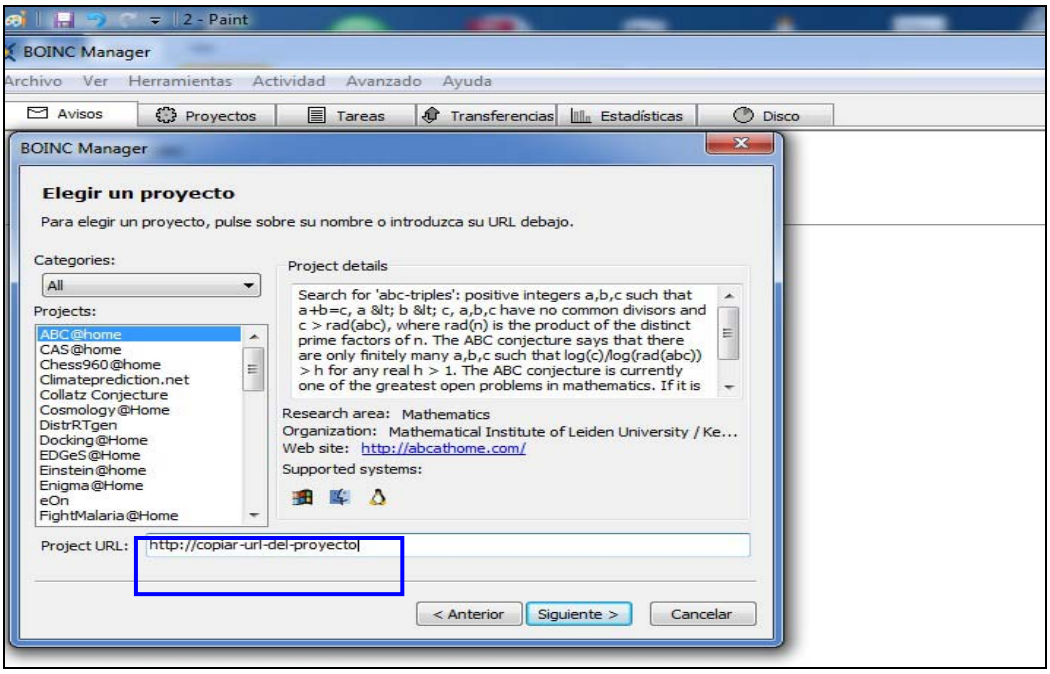


Figura 7. Ingresar URL desde el BOINC *Manager*.

Registrarse con una cuenta de correo válida en el BOINC-*Manager*, de esta manera se realizará la autenticación de los clientes que se unirán a los proyecto.

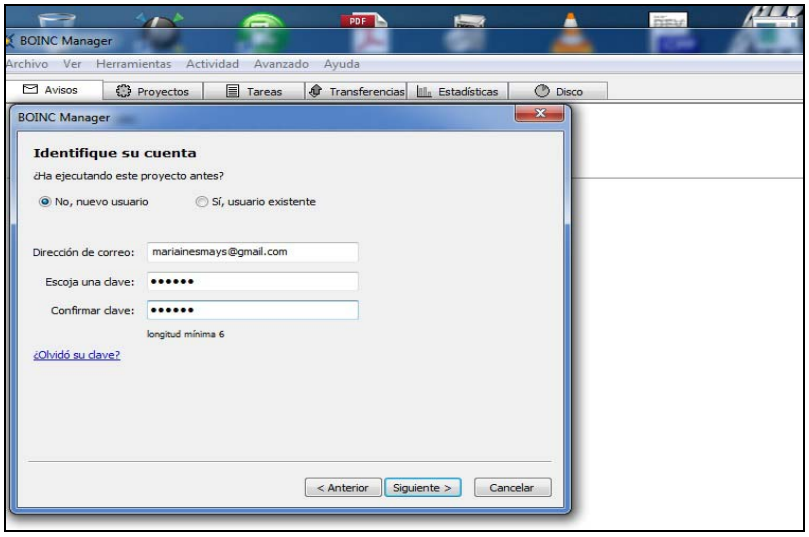


Figura 8. Ingresar la dirección de correo desde el BOINC *Manager*.

Una vez el usuario haya ingresado el proyecto el BOINC-*Manager* se conectará con el *Core-Client* del servidor BOINC y le asignará las WU a ser procesadas por la(s) computadora(s).

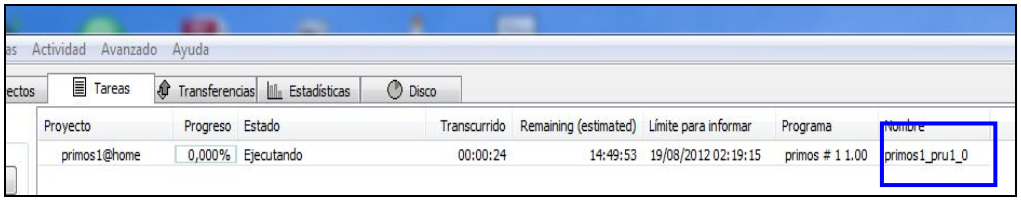


Figura 9. Asignación de WU del proyecto.

Caso de estudio para la ejecución del diseño: números primos.

Un número primo es un número natural mayor que 1 que tiene únicamente dos divisores distintos: él mismo y el 1.

El algoritmo utilizado para el cálculo de los números primos fue el siguiente:

```
#include <stdio.h>
#include <iostream> //Librerías utilizadas para la entrada y
#include <time.h>  // salida de datos
#include <stdlib.h>
```

```
#define min 1
#define max 5000000 //Definición de variables y funciones
bool primo(unsigned long);

int main(int argc, char *argv[]){ //Programa principal
FILE *fp;
fp = fopen("out","w");
char numero[16];
unsigned long i,j=0;
for(i=min;i<max;i++)
 if(primo(i))
 {sprintf(numero,"%d \n",i);
  fputs (numero, fp);
   j++;
}
fclose(fp);
}

bool primo(unsigned long a){ //Algoritmo de ejecución de números
 bool sw = true;                    //primos
 unsigned long i = 2;
 while (sw&& a>i){
    if(((a%i) != 0))
     i++;
    else
    sw = false;
    }
   return sw;
 }  //Fin del programa
```

Es preciso indicar que este método no es el más eficiente para calcular los números primos, existen otros métodos como los son la *criba de Eratóstenes* es una manera sencilla de hallar todos los números primos menores o iguales que un número dado. Se basa en confeccionar una lista de todos los números naturales desde el 2 hasta ese número y tachar repetidamente los múltiplos de los números primos ya descubiertos. La *criba de Atkin*, más moderna, tiene una mayor complejidad, pero si se optimiza apropiadamente también es más rápida. También existe una reciente *criba de Sundaram* que genera únicamente números compuestos, siendo los primos los números faltantes.

Análisis de los números primos en BOINC

Para trabajar un proyecto con BOINC puede realizarse de dos maneras una es la modificación del código del proyecto agregándole las funciones y librerías propias de

BOINC y otra es la implementación del BOINC *Wrapper* que fue la opción utilizada para el desarrollo de este proyecto.

Para analizar el cálculo de los números primos se te tomó en cuenta el cálculo de los números primos en un rango de 1 – 5 000 000. Este ejemplo se realizó tomando en cuenta tres escenarios:

Escenario 1: tiempo de ejecución del algoritmo en un servidor, sin BOINC.

La ejecución del escenario 1 fue realizada en un servidor HP PROLIANT ML 110 G5 Procesador XEON de 2.3 GHz, 4Mb Cache, 2Gb RAM, 500Gb Disco Duro

En el parámetro “real” que se visualiza en la figura 10 el tiempo que tardó la máquina en ejecutar el cálculos de los números primos en el rango de 1 – 5 000 000 fue de 79’ 30”.

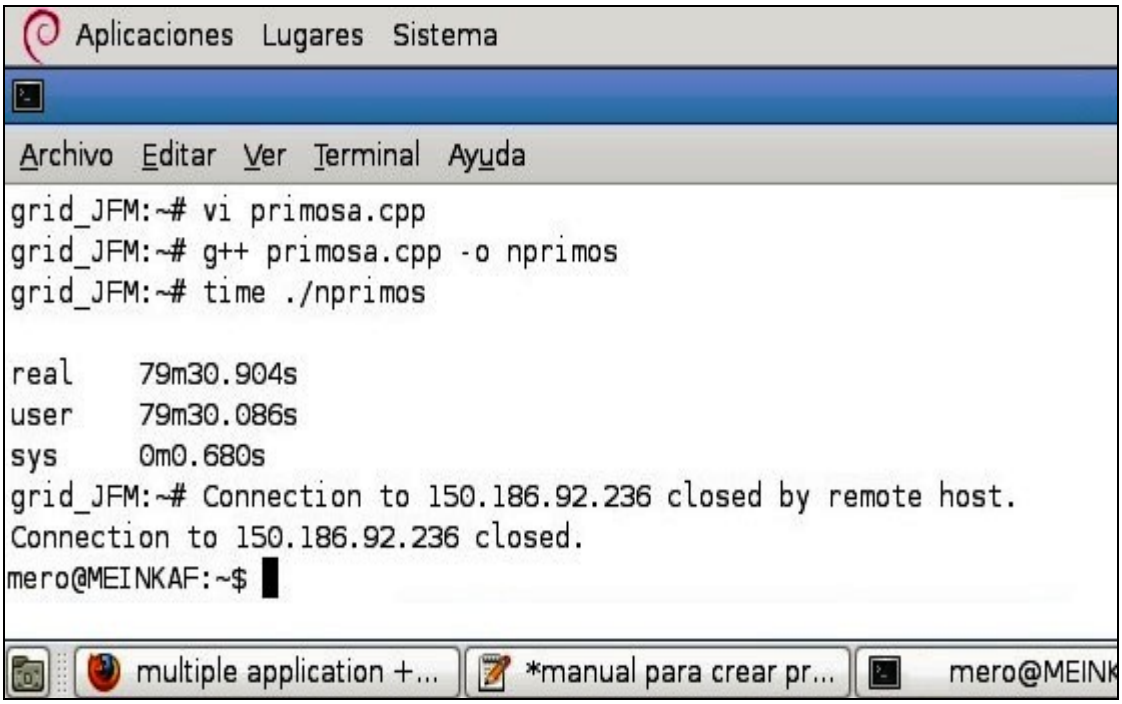


Figura 10. Tiempo que tardó la tarea en el escenario 1.

Escenario 2: tiempo de ejecución del algoritmo en una computadora, usando BOINC.

La ejecución del escenario 2 fue realizada en una laptop Dell Intel Pentium M 1.7GHz. 512Mb de RAM, 40Gb Disco Duro.

El tiempo que tardó la máquina en ejecutar el cálculos de los números primos en el rando de 1 – 5 000 000 fue de 198’ 89”.

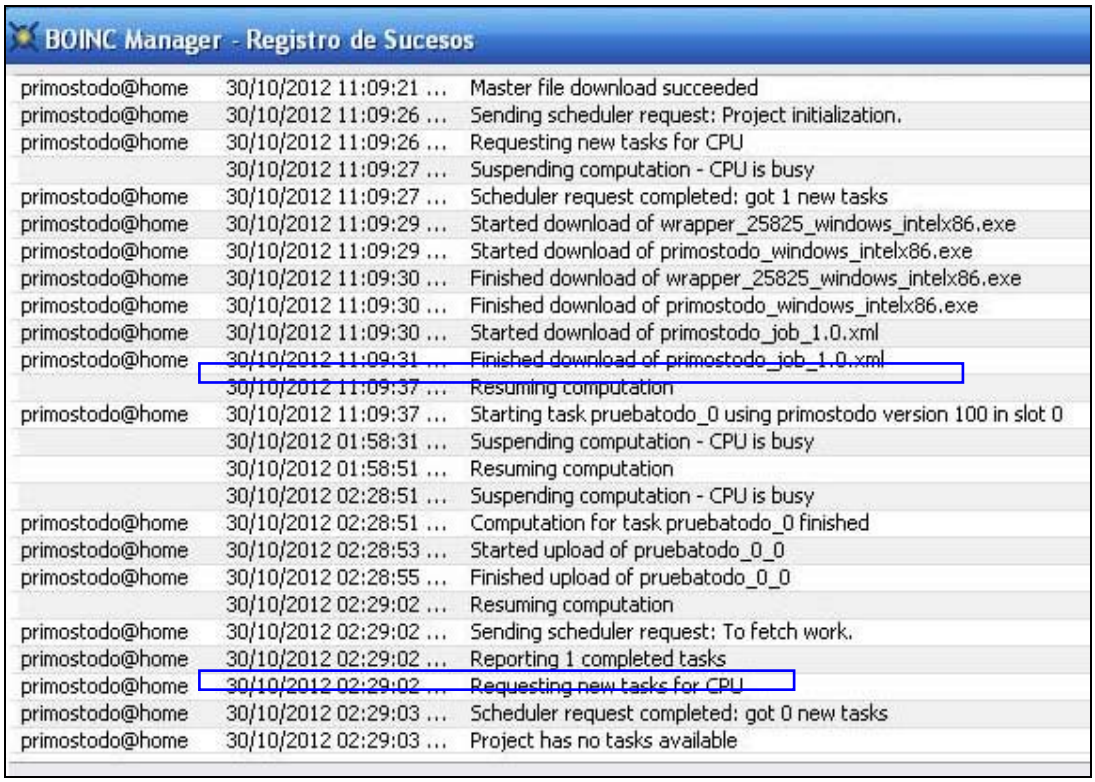


Figura 11. Tiempo que tardó la tarea en el escenario 2.

Escenario 3: tiempo de ejecución del algoritmo utilizando cinco (5) máquinas con características disimiles en una red fuera de las instalaciones del Núcleo de Sucre de la Universidad de Oriente bajo la arquitectura BOINC y asignándole a cada máquina un intervalo de 1 000 000 números para calcular los primos.

Para trabajar con el escenario 3, se utilizaron cinco máquinas cuyas características y tareas asignadas se muestran en la siguiente tabla:

Tabla 3. Características de las máquinas del escenario 3.

Computadora	Descripción	Sistema Operativo
1	Laptop Compaq Presario c700 PentiumDual 1.60GHz, 1Gb RAM, 120Gb Disco Duro	Windows XP
2	Laptop Toshiba Satillite C645D Dual Core 2.3GHz, 2Gb, 250Gb Disco Duro	Windows 7
3	Laptop Lenovo Thinkpad sl 500 Core2Duo 2.0GHz, 992Mb RAM, 120Gb Disco Duro	Windows XP
4	Laptop Síragon Sn50 Corei3 2.1GHz 4Gb RAM, 320Gb Disco Duro	Windows 7
5	Desktop Clon Core2Duo 2.19GHz 0.99Gb RAM, 80Gb Disco Duro	Windows XP

En la tabla 4 se puede apreciar el intervalo de números primos asignados a cada computadora.

Tabla 4. Asignación de las tareas a las máquinas.

Computadora	Tarea Asignada
1	Calcular los números primos en el rango 1 – 1 000 000
2	Calcular los números primos en el rango 1 000 001 – 2 000 000
3	Calcular los números primos en el rango 2 000 001 – 3 000 000
4	Calcular los números primos en el rango 3 000 001 – 4 000 000
5	Calcular los números primos en el rango 4 000 001 – 5 000 000

En la tabla 4 se muestra el tiempo que tardó cada máquina para computar la tarea asignada.

Tabla 5. Relación máquina-tiempo tardado en computar en el escenario 2.

Máquina	Tiempo
1	4’ 59”
2	10’ 30”
3	20’ 51”
4	22’ 10”
5	27’ 58”

Tiempo total en ejecución: 27’ 58”

Escenario 4: tiempo de ejecución del algoritmo utilizando cinco máquinas con características similares en la red del Núcleo de Sucre de la Universidad de Oriente bajo la arquitectura BOINC y asignándole a cada máquina un intervalo de 1 000 000 números para calcular los primos.

La ejecución del escenario 4 fue realizada en cinco (5) laptops Dell Intel Pentium M 1.7GHz. 512Mb de RAM, 40Gb Disco Duro.

Tabla 6. Relación máquina-tiempo tardado en computar en el escenario 4.

Máquina	Tiempo
1	9’ 01”
2	25’ 49”
3	41’ 49”
4	55’ 59”
5	69’ 24”

Tiempo total en ejecución: 69’ 24”

Mediante las pruebas realizadas en los diferentes escenarios, se confirmó que con el cómputo distribuido se reducen los tiempos de respuesta para este caso de estudio.

El escenario 4 muestra como las computadoras a pesar de no poseer una arquitectura actualizada (en relación a procesador y memoria) a través del cómputo distribuido, lograron brindar un menor tiempo de respuesta a la prueba realizada en comparación al escenario 1. Confirmando que los equipos al pasar a un estado de inactividad (por la devaluación de su arquitectura) pueden seguir ofreciendo sus recursos para el desarrollo de tareas que requieren una gran capacidad de cómputo. En relación a lo que tardó la tarea realizada en una sola máquina (escenario 2) y las cinco máquinas con una misma arquitectura (escenario 4) la reducción del tiempo de respuesta fue de un 67,78%.

En algunas computadoras del Programa de la Licenciatura en Informática de la UDO-NS no pudieron realizarse pruebas de desempeño del BOINC debido a que estas se encuentran en áreas donde la configuración de la red impedía su conexión con el

servidor. Se determinó que el problema era originado por un error en la comunicación a nivel del Protocolo de Transferencia de Hipertexto (HTTP) y que se requiere que el Centro de Computación Académica realice configuraciones adicionales para corregir esta situación. La figura 12 muestra el error de comunicación presentado por los clientes BOINC en algunas áreas del Programa de la Licenciatura en Informática de la UDO-NS.

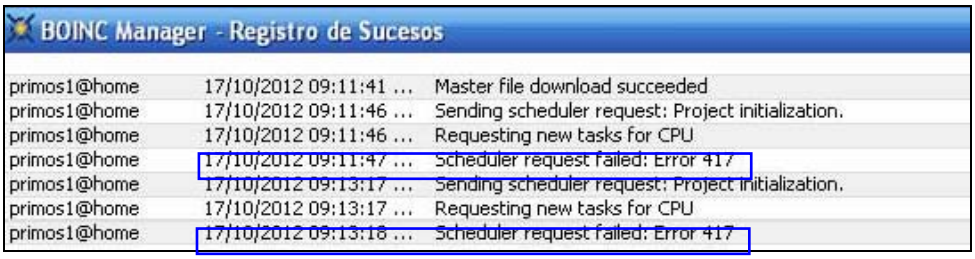


Figura 12. Error de comunicación presentado.

CONCLUSIONES

A través de la presente investigación pudo demostrarse que una tarea que requería un elevado poder de procesamiento el tiempo se redujo considerablemente demostrando que el *Volunteer Computing* se perfila como una poderosa herramienta para el procesamiento de tareas que ameriten gran poder de cómputo.

Mediante la presente investigación se pudo constatar que el cómputo voluntario es una opción viable que permite reducir el tiempo de cómputo de proyectos que requieran una mayor capacidad de procesamiento usando los equipos disponibles sin necesidad de adquirir tecnologías de elevados costos.

BOINC resultó ser una herramienta muy eficiente que permite integrar y utilizar los ciclos ociosos de las computadoras sin importar el sistema operativo que las mismas posean en función de poder utilizarlos para realizar otro tipo de tareas.

La metodología de James McCabe (1998) se adaptó significativamente al desarrollo de la presente investigación, ya que a través de sus fases se pudo culminar con éxito y obtener los resultados esperados.

RECOMENDACIONES

Crear áreas de investigación que incentiven el estudio del cómputo paralelo y distribuido ya que son áreas que se perfilan en el futuro para hacer uso de los tiempos ociosos de las computadoras.

Desarrollar un portal Web que permita la difusión rápida y masiva de los proyectos de investigación que se involucrarán con el presente proyecto de tesis, lo cual permitirá que muchos investigadores, docentes y alumnos, se beneficien con los resultados del mismo y que puedan unirse a los proyectos que estén realizándose.

Incentivar el desarrollo de una plataforma que permita integrar todas las computadoras asociadas al *Volunteer Computing*, con lo que se tendría un acceso remoto sobre cada cliente BOINC instalado en todas estas computadoras.

Solicitar al Centro de Computación Académica de la UDO-NS realizar las configuraciones necesarias, para garantizar que todas las computadoras conectadas a la red del Núcleo, puedan integrarse como voluntarias a los proyectos de cómputo que se realicen.

BIBLIOGRAFÍA

Abreu, J. y Cols. 2006. Introducción a la Tecnología Grid. Facultad de Informática. Universidad de Matanzas "Camilo Cienfuegos" Cuba.

Academia CISCO. 2008. Currículo CCNA (CISCO *certified network associate*) version 4.0. Semestre 1.

Aguiar, G. 2005. Grid Computing. Página web de la Universidad Nacional de Nordeste. Facultad de Ciencias Exactas y Naturales y Agrimensura. <http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Gridmonogcarol.pdf. > (20/06/2010).

Ahmar Abbas, 2004. *Grid Computing: A Practical Guide to Technology and Applications*, Primera edición. Editorial Charles River Media. Massachusetts.

Ananth Grama y cols. 2003. *Introduction to parallel Computing*. Segunda edición. Editorial Pearson, Harlow

Anderson D., y cols. 2002. *SETI@home: An experiment in publicresource computing* Communications of the ACM.

Ariannejad M. 2001. *Trends in Middleware Systems.* Computer Engineering Research Group, 2001.

Bakken, D., 2002. "*Middleware*". *Chapter in Encyclopedia of Distributed Computing*. Editorial Kluwer Academic Publishers.

Barrera, R. y Cols. 2009. Computación GRID. Página web de la Escuela Superior Politécnica del Litoral <http://www.dspace.espol.edu.ec/ simple/search?query=computacion+grid&submit=Ir>. (10/04/2010).

Bishop T. y Karne R. 2010. A survey of middleware**.** Computer & Information Science Dept Towson University Maryland. USA

Castro, A. 1999. Teleinformática para ingenieros en sistemas de información. Segunda Edición. Editorial Reverté. España.

Des Ligneris, B. 2003 *Open Source Cluster Application Resources (OSCAR): design, implementation and interest for the computer scientific community."* OSCAR Symposium, Sherbrooke.

Di Constanzo A. y cols. 2009. *Building a Virtualized Distributed Computing Infrastructure by Harnessing Grid and Cloud Technologies.* Universidad de Melbourne.

Foster I., y Kesselman C. 2003 The Grid 2: Blueprint for a New Computing Infrastructure. Segunda edición. Editoral Morgan Kaufmann.

Gesit G. y Kohl J., 1996. *PVM and MPI: a Comparison of Features P. M. Papadopoulos.*

Ghazali E. y Zomaya A. 2007 *Grid Computing for Bioinformatics and Computational Biology*, Primera edición. Editorial Wiley-Interscience.

Gorlatch S., Fragopoulou P., y Priol F. 2008. *"Grid Computing: Achievements and Prospects"*, Primera edición. Editorial Springer.

Groth, D. 2005. Guía del estudio de redes. Cuarta edición. Editorial John Wiley & Sons. Estados Unidos.

Herrera, E. 1998. Introducción a las telecomunicaciones modernas. Editorial Limusa. México.

Joyanes, L. 1998. Programación orientada a objetos. Segunda edición. Editorial McGraw-Hill, España

Kacsuk P. y cols. 2008. *Distributed and Parallel Systems: In Focus: Desktop Grid Computing*. Primera edición. Editorial Springer.

Mc Cabe, J. 1998. Introducción al Análisis. Metodología de Red de las Computadoras. Primera Edición. Editorial Mogan Kauffman S.A. México

Morillo P., 2003. *Grid Computing:* Compartición de recursos y optimización del hardware. Editorial Mundo Electrónic.

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

Smith, R. 2004. *Grid Computing: A Brief Technology Analysis*. <http://citeseerx.ist.psu.edu/viewdoc/download? (02/08/2012)

Sommerville, I. 2004. *Software engineering.* Septima edición. Editorial Pearson. Boston

Suppi, R. 2003. *Clustering*. Página web de la Universitat Oberta de Catalunya <http://www.uoc.edu/opencms_portal2/opencms/CA/_config/search/index.html?searchWords=Remo+Suppi+Boldrito&search.x=0&search.y=0&searchBase=UmVtbyBTdXBwaSBCb2xkcml0bw%3D%3D> (15/05/2010).

Talens-Oliag, S. 2008. Introducción a los certificados digitales. España

Tamayo y Tamayo, M. 2003. El Proceso de Investigación Científica. Cuarta edición. Ediciones Limusa. S.A. México.

Tarancón, A. 2009. IBERCIVIS MEMORIA. España

Universidad de los Andes. 2008. Memoria y Cuenta. Tomo I. Mérida – Venezuela

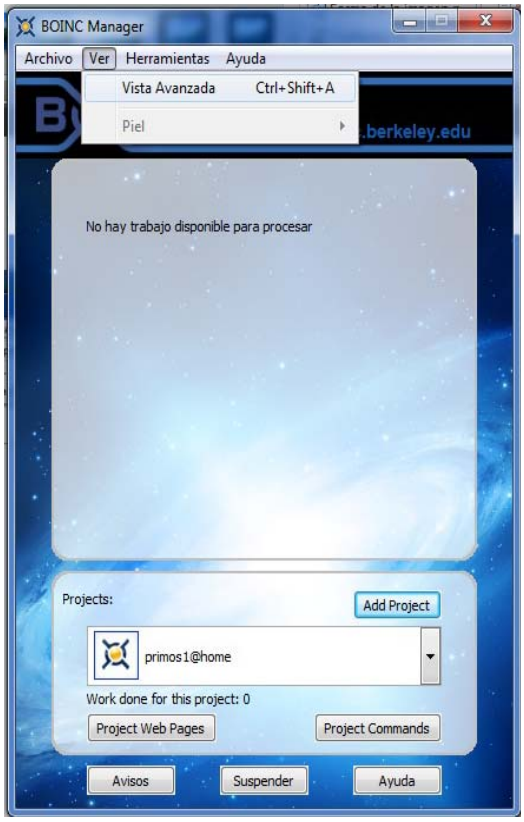
Urueña A. y cols. 2012. *Cloud Computing* Restos y Oportunidad. España

APÉNDICES

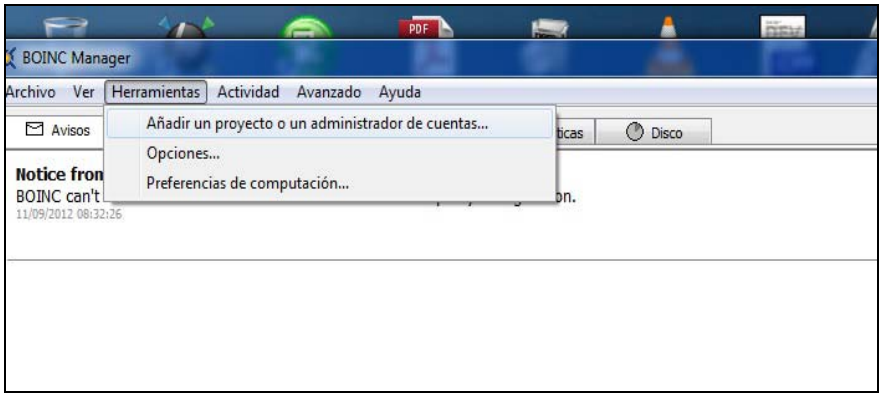
APÉNDICE A

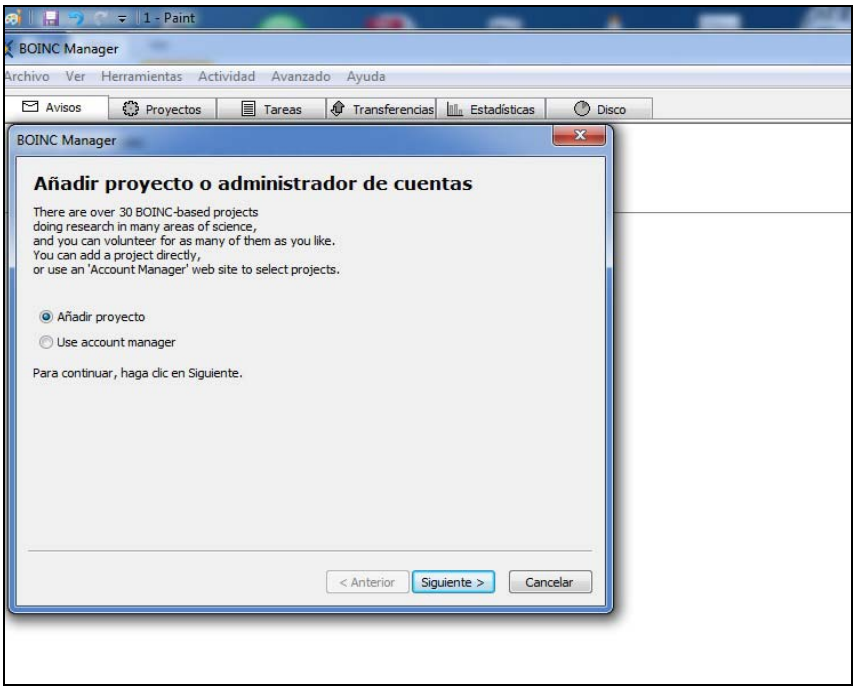
Pasos para unirse a un proyecto

Vista simple del BOINC *Manager*.

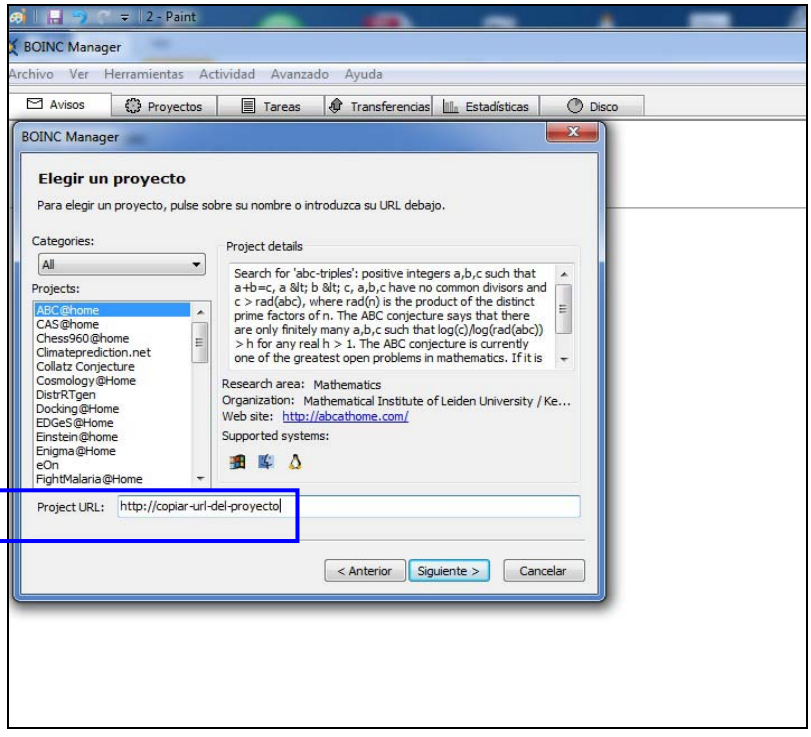


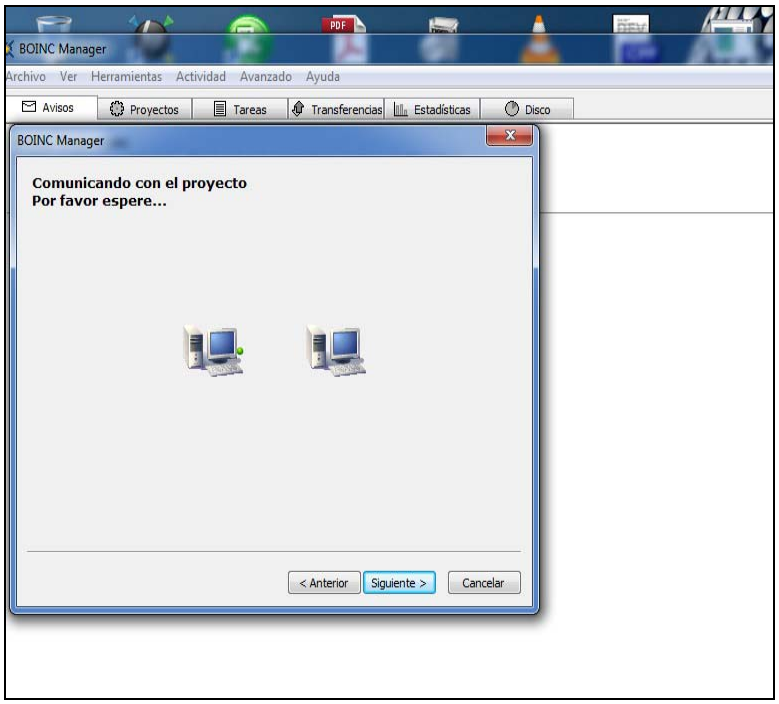
Añadir un proyecto.



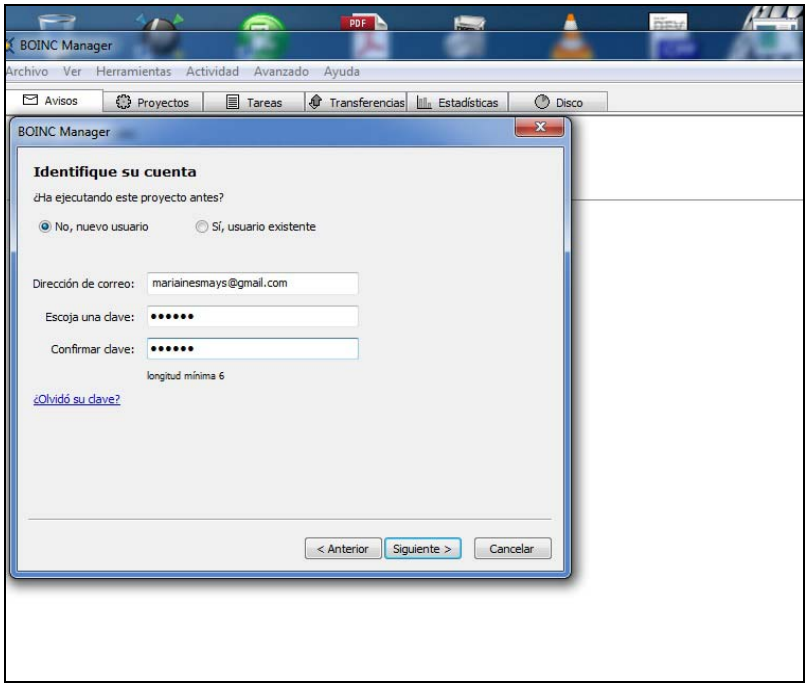


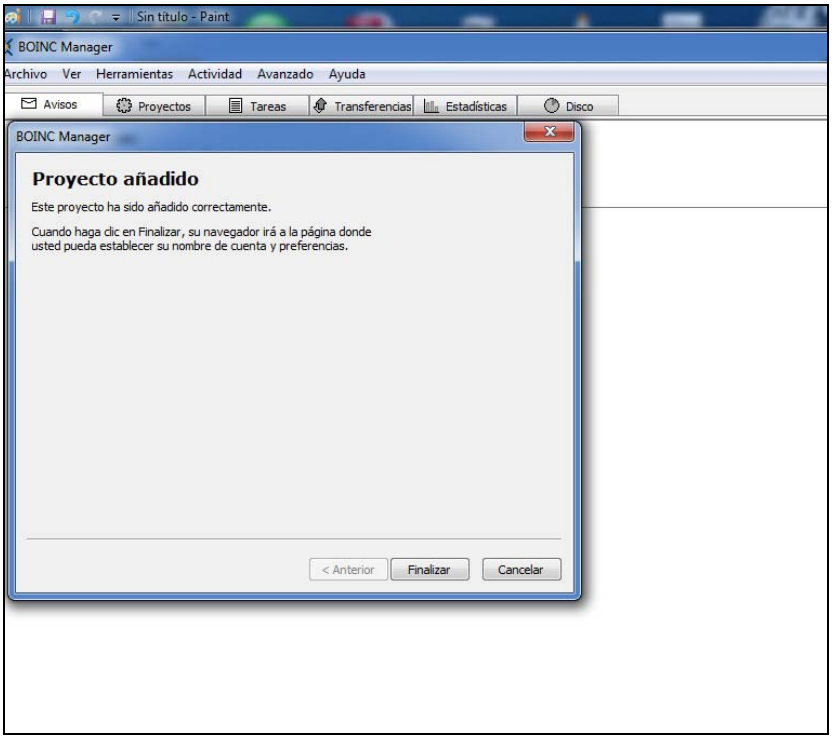
Agregar la URL del proyecto.



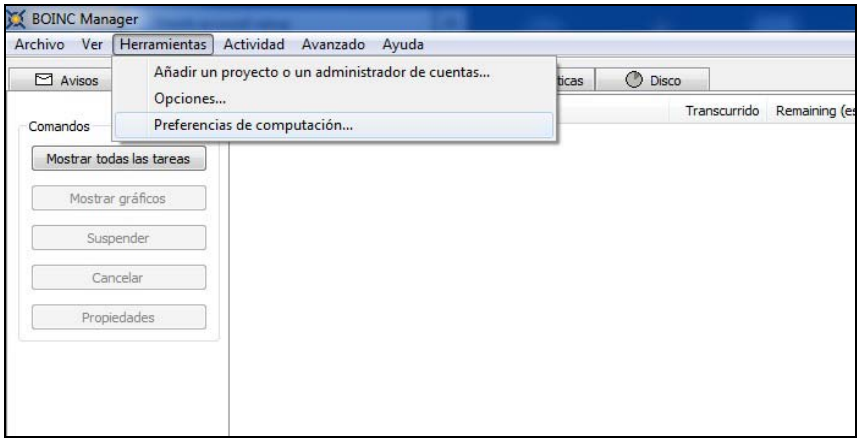


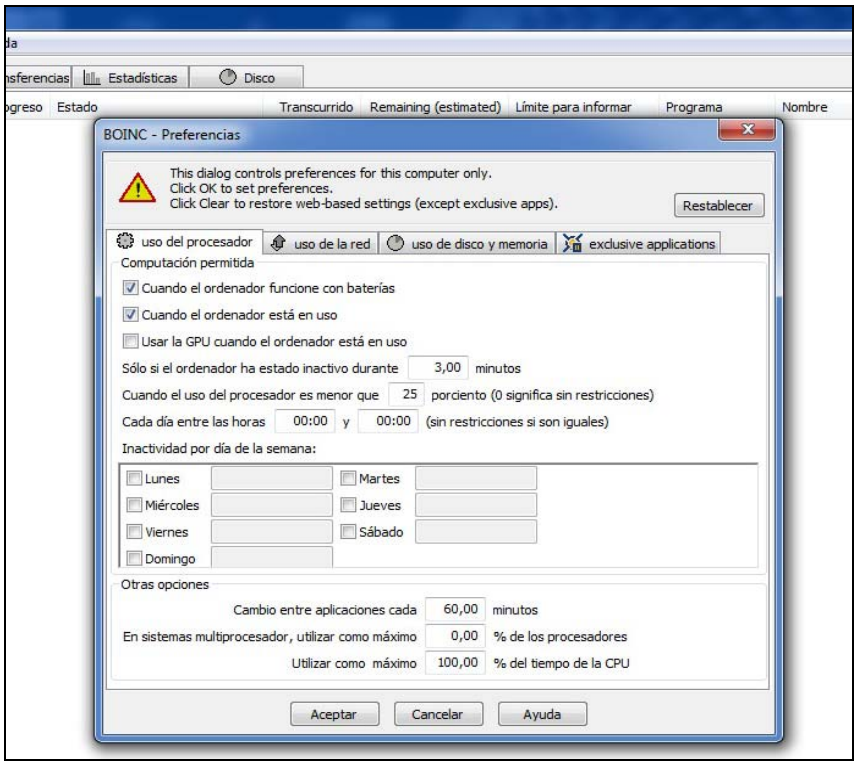
Ingresar una cuenta de correo y un password.



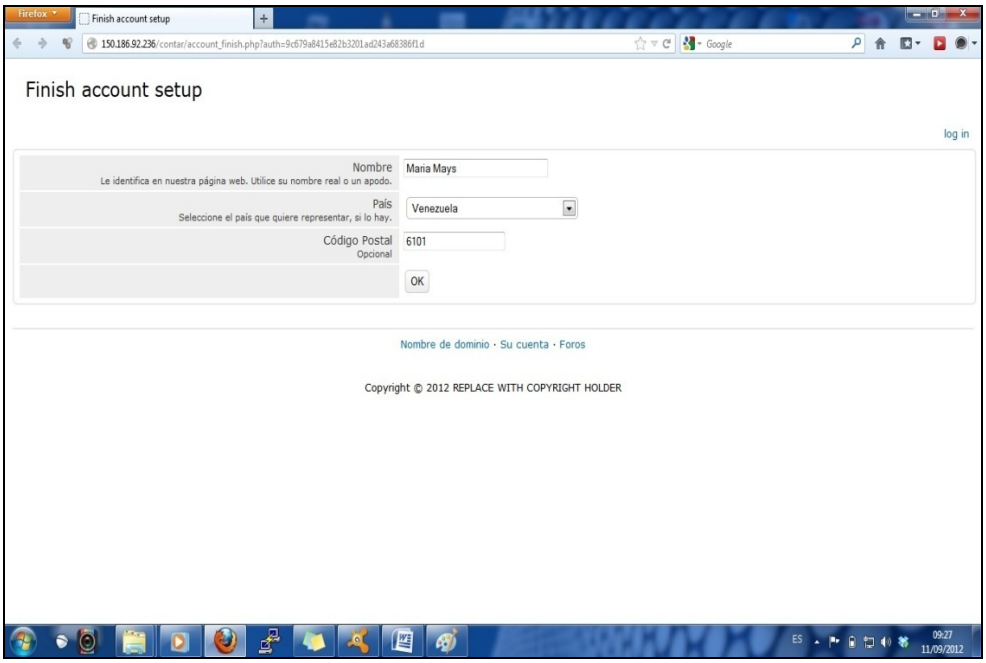


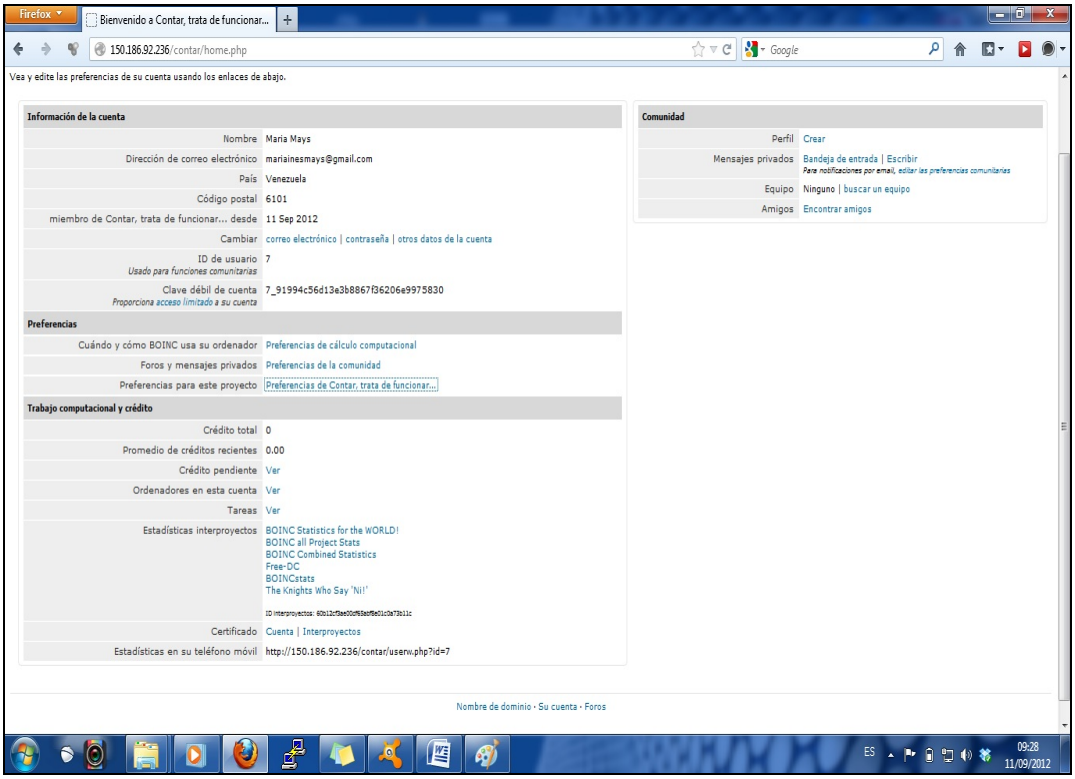
En preferencias de computación podrá configurar la cantidad de memoria y disco que desee que sea utilizado en el proyecto.





Página para monitorear su estado en relaciones a los trabajos en que los que ha participado.

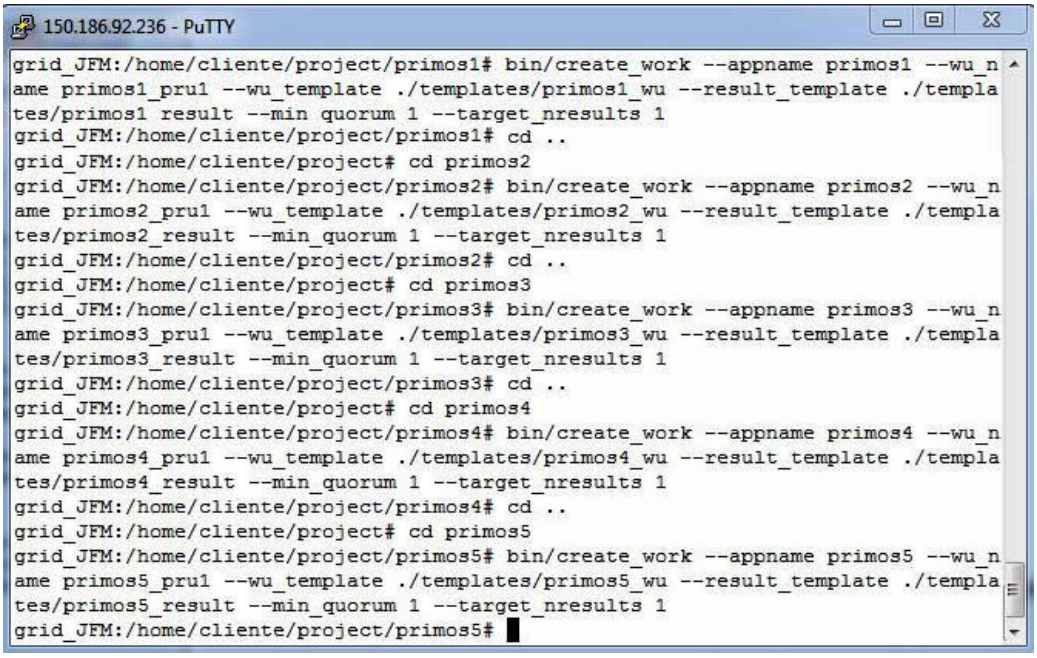



Firefox
Bienvenido a Contar, trata de funcionar...
150.186.92.236/contar/home.php
Google
Vea y edite las preferencias de su cuenta usando los enlaces de abajo.
Información de la cuenta
Nombre Maria Mays
Dirección de correo electrónico mariainesmays@gmail.com
País Venezuela
Código postal 6101
miembro de Contar, trata de funcionar... desde 11 Sep 2012
Cambiar correo electrónico | contraseña | otros datos de la cuenta
ID de usuario
Usado para funciones comunitarias 7
Clave débil de cuenta
Proporciona acceso limitado a su cuenta 7_91994c56d13e3b8867f36206e9975830
Preferencias
Cuándo y cómo BOINC usa su ordenador Preferencias de cálculo computacional
Foros y mensajes privados Preferencias de la comunidad
Preferencias para este proyecto Preferencias de Contar, trata de funcionar...
Trabajo computacional y crédito
Crédito total 0
Promedio de créditos recientes 0.00
Crédito pendiente Ver
Ordenadores en esta cuenta Ver
Tareas Ver
Estadísticas interproyectos BOINC Statistics for the WORLD!
BOINC all Project Stats
BOINC Combined Statistics
Free-DC
BOINCstats
The Knights Who Say 'Ni!'
ID interproyectos: 60b12cf3ae30cf63a0f8e01c0a73b11c
Certificado Cuenta | Interproyectos
Estadísticas en su teléfono móvil http://150.186.92.236/contar/userw.php?id=7
Comunidad
Perfil Crear
Mensajes privados Bandeja de entrada | Escribir
Para notificaciones por email, editar las preferencias comunitarias
Equipo Ninguno | buscar un equipo
Amigos Encontrar amigos
Nombre de dominio · Su cuenta · Foros
ES
09:28
11/09/2012

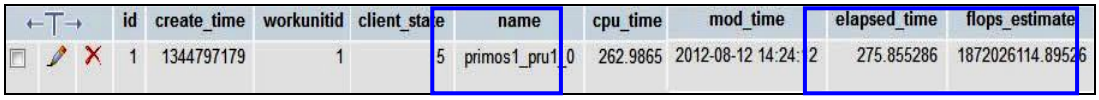
APÉNDICE B

Creando las wu del proyecto

Creando las WU.

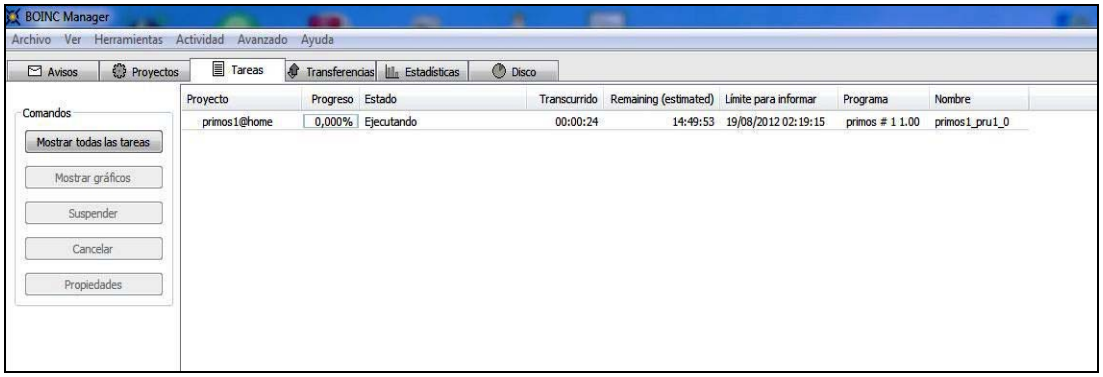


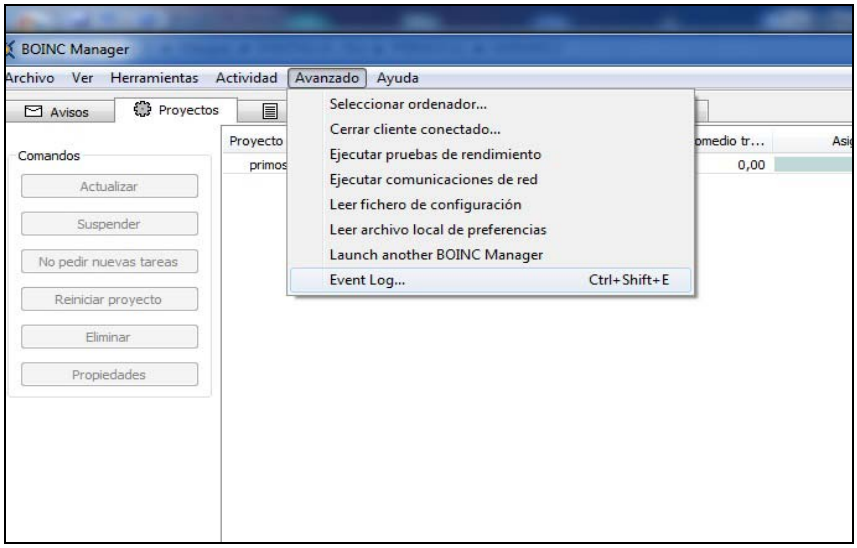
Vista de la base de datos de una de las WU creadas



	id	create_time	workunitid	client_state	name	cpu_time	mod_time	elapsed_time	flops_estimate
	1	1344797179	1	5	primos1_pru1_0	262.9865	2012-08-12 14:24:12	275.855286	1872026114.89526

Seguimiento del estado de la tarea mediante el BOINC-*Manager*.





BOINC Manager - Registro de Sucesos

Proyecto	Hora	Mensaje
	12/08/2012 02:34:00	No general preferences found - using defaults
	12/08/2012 02:34:00	Reading preferences override file
	12/08/2012 02:34:00	Preferences:
	12/08/2012 02:34:00	max memory usage when active: 2003.57MB
	12/08/2012 02:34:00	max memory usage when idle: 3606.42MB
	12/08/2012 02:34:00	max disk usage: 10.00GB
	12/08/2012 02:34:00	don't use GPU while active
	12/08/2012 02:34:00	suspend work if non-BOINC CPU load exceeds 25 %
	12/08/2012 02:34:00	(to change preferences, visit the web site of an attached project, or select Preferences in the Manager)
	12/08/2012 02:34:00	Not using a proxy
	12/08/2012 02:34:00	This computer is not attached to any projects
	12/08/2012 02:34:00	Visit http://boinc.berkeley.edu for instructions
	12/08/2012 02:36:03	Fetching configuration file from http://150.186.92.236/primos1/get_project_config.php
primos1@home	12/08/2012 02:36:26	Master file download succeeded
primos1@home	12/08/2012 02:36:32	Sending scheduler request: Project initialization.
primos1@home	12/08/2012 02:36:32	Requesting new tasks for CPU
primos1@home	12/08/2012 02:36:34	Scheduler request completed: got 1 new tasks
primos1@home	12/08/2012 02:36:36	Started download of wrapper_25825_windows_intelx86.exe
primos1@home	12/08/2012 02:36:36	Started download of primos1_windows_intelx86.exe
primos1@home	12/08/2012 02:36:44	Sending scheduler request: To fetch work.
primos1@home	12/08/2012 02:36:44	Requesting new tasks for CPU
primos1@home	12/08/2012 02:36:45	Finished download of wrapper_25825_windows_intelx86.exe
primos1@home	12/08/2012 02:36:45	Started download of primos1_job_1.0.xml
primos1@home	12/08/2012 02:36:45	Scheduler request completed: got 0 new tasks
primos1@home	12/08/2012 02:36:45	Project has no tasks available
primos1@home	12/08/2012 02:36:46	Finished download of primos1_windows_intelx86.exe
primos1@home	12/08/2012 02:36:46	Finished download of primos1_job_1.0.xml
primos1@home	12/08/2012 02:36:46	Starting task primos1_pru1_0 using primos1 version 100 in slot 0
primos1@home	12/08/2012 02:41:23	Computation for task primos1_pru1_0 finished
primos1@home	12/08/2012 02:41:25	Started upload of primos1_pru1_0_0
primos1@home	12/08/2012 02:41:26	Finished upload of primos1_pru1_0_0
primos1@home	12/08/2012 02:41:29	Sending scheduler request: To fetch work.
primos1@home	12/08/2012 02:41:29	Reporting 1 completed tasks, requesting new tasks for CPU
primos1@home	12/08/2012 02:41:31	Scheduler request completed: got 0 new tasks

HOJA DE METADATOS

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

Título	ENLACE *VOLUNTEER COMPUTING* EN LA UNIVERSIDAD DE ORIENTE NÚCLEO DE SUCRE
Subtítulo	

Autor

Apellidos y Nombres	**Código CVLAC / e-mail**	
Mays Belmonte, María Inés	**CVLAC**	**18210403**
	e-mail	**mariainesmays@gmail.com**
	e-mail	

Palabras o frases claves:

Volunteer Computing, *Grid Computing*, BOINC, Computación Paralela, Virtualización.

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

Líneas y sublíneas de investigación:

Área	Subárea
Ciencias	**Informática**

Resumen (abstract):

El presente trabajo da cuenta del proceso de desarrollo de un enlace de *Volunteer Computing* 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 y almacenamiento; en donde se pudo apreciar carencia de tecnología que le permita trabajar dentro del Núcleo, lo cual conllevó, mediante la aplicación de la metodología de análisis y diseño de redes de computadores planteada por James McCabe (1998), a determinar los requerimientos de los usuarios, con el objetivo de ofrecer soluciones adaptables a las necesidades de los mismos y a las tecnologías disponibles para el desarrollo de este proyecto; se evaluó el flujo de datos de los servidores del *Volunteer Computing* para determinar la manera como se divide la aplicación entre el conjunto de computadoras que forman dicha plataforma. Se trabajó con el diseño lógico y direccionamiento existente en el Núcleo específicamente en el Programa de la Licenciatura en Informática, por último se realizaron los 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. El trabajo propuesto pretende implementar una solución que aproveche los ciclos de cómputo no utilizados de las computadoras de escritorio en laboratorios informáticos y demás computadoras personales que se encuentren dentro de la Universidad, y con ello resolver problemas computacionalmente intensivos, ayudando y promoviendo a investigadores que necesiten gran capacidad de cómputo y procesamiento, para que puedan empezar proyectos de investigación propios o concluir de manera efectiva los ya empezados por ellos.

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

Contribuidores:

Apellidos y Nombres	ROL / Código CVLAC / e-mail	
Geremía, Daniel	**ROL**	**CA** [ ] **AS** [X] **TU** [ ] **JU** [ ]
	CVLAC	**8645325**
	e-mail	**geremiada1@hotmail.com**
	e-mail	
Milá de la Roca, Napoleón	**ROL**	**CA** [ ] **AS** [ ] **TU** [X] **JU** [ ]
	CVLAC	**11833462**
	e-mail	**milan@udo.edu.ve**
	e-mail	
Márquez, Henry	**ROL**	**CA** [ ] **AS** [ ] **TU** [ ] **JU** [X]
	CVLAC	**8443874**
	e-mail	**henrylmarquez@gmail.com**
	e-mail	
Rodríguez, Carmelys	**ROL**	**CA** [ ] **AS** [ ] **TU** [ ] **JU** [X]
	CVLAC	**13539531**
	e-mail	**carmelysrodriguez@gmail.com**
	e-mail	

Fecha de discusión y aprobación:

Año	Mes	Día
2013	**01**	**17**

Lenguaje: **SPA**

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

Archivo(s):

Nombre de archivo	**Tipo MIME**
Tesis-maysm.doc	**Aplication/word**

Alcance:

Espacial: **(Opcional)**

Temporal: **(Opcional)**

Título o Grado asociado con el trabajo: Licenciada en Informática

Nivel Asociado con el Trabajo: Licenciada

Área de Estudio: Informática

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

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

UNIVERSIDAD DE ORIENTE
CONSEJO UNIVERSITARIO
RECTORADO

CU Nº 0975

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.

UNIVERSIDAD DE ORIENTE
SISTEMA DE BIBLIOTECA
RECIBIDO POR
FECHA 5/8/09 HORA 5:30

Comunicación que hago a usted a los fines consiguientes.

Cordialmente,

JUAN A. BOLAÑOS CURIEL
Secretario

REPÚBLICA BOLIVARIANA DE VENEZUELA
UNIVERSIDAD DE ORIENTE
SECRETARÍA
CONSEJO UNIVERSITARIO

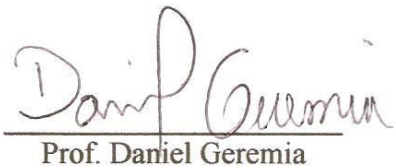
C.C: *Rectora, Vicerrectora Administrativa, Decanos de los Núcleos, Coordinador General de Administración, Director de Personal, Dirección de Finanzas, Dirección de Presupuesto, Contraloría Interna, Consultoría Jurídica, Director de Bibliotecas, Dirección de Publicaciones, Dirección de Computación, Coordinación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

JABC/YGC/maruja

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".

María Inés Mays B.
Autor



Prof. Daniel Geremia
Asesor 1