IMPLEMENTACIÓN DE LOS SERVICIOS Y MODULOS FUNDAMENTALES PARA LA CONSTRUCCIÓN Y PUESTA EN MARCHA DE UNA INFRAESTRUCTURA COMPUTACIONAL GRID USANDO EL MIDDLEWARE GLITE

IVAN DARIO RODRIGUEZ SALGUERO

JULIAN MAURICIO NOBSA VARGAS

INGENIERÍA DE SISTEMAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA, NOVIEMBRE 2008

IMPLEMENTACIÓN DE LOS SERVICIOS Y MODULOS FUNDAMENTALES PARA LA CONSTRUCCIÓN Y PUESTA EN MARCHA DE UNA INFRAESTRUCTURA COMPUTACIONAL GRID USANDO EL MIDDLEWARE GLITE

AUTORES

IVAN DARIO RODRIGUEZ SALGUERO

JULIAN MAURICIO NOBSA VARGAS

Trabajo de grado para optar por el titulo de ingeniero de sistemas

DIRECTOR

MSC. GILBERTO JAVIER DÍAZ TORO

CODIRECTOR

ING. ERICK RAMON MENESES CUADROS

INGENIERÍA DE SISTEMAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA, NOVIEMBRE 2008

Este libro esta dedicado a los menos, serán sin duda, los que lo comprendan. Pues bien, solo esos son mis lectores, a los hiperbóreos.
A aquél que salió de primero de la cueva, aquél que se sumergió sin miedo en el océano, aquél que no fue animal doméstico.
A mi Madre que hizo su mejor esfuerzo y me ha dado hasta donde ha podido siempre apoyándome
A Julián Prieto donde se encuentre

Julián

Sobra decir, que va a dedicado a todos los que nos han colaborado y acompañado.
Sin embargo es necesario mencionar a
Betty, Lucho, Mis hnos, Diana, Mis amigos...

Iván

A mis padrinos Efrain y Socorro por creer en mí y darme estudio, el arma mas poderosa contra la desigualdad

A mis amigos y amigas que me han acompañado en muchas causas y que espero lo sigan haciendo

A CUSOL por ser escenario de ideas y amistad

Al profesor Gilberto Díaz por su colaboración y sugerencias

Al Ingeniero Erick Meneses por apostar por nosotros, ser un amigo y guiarnos

A Noé Vazquez por su ayuda desinteresada y a todos los que de alguna manera nos ayudaron en esta tarea.

Julián.

A mis padres Betty y Lucho, que me brindaron siempre una incondicional libertad.

A CUSOL espacio altruista donde la diversidad reina y quiero mucho.

A Erick que siempre se intereso en lo personal como en lo académico.

A Los amigos de la Universidad... que los aprecio demasiado.

Iván.

INDICE DE CONTENIDOS

INDICE DE TABLAS

INDICE DE FIGURAS

RESUMEN.

Titulo: IMPLEMENTACIÓN DE LOS SERVICIOS Y MODULOS FUNDAMENTALES PARA LA CONSTRUCCIÓN Y PUESTA EN MARCHA DE UNA INFRAESTRUCTURA COMPUTACIONAL GRID USANDO EL MIDDLEWARE GLITE.*

Autores: IVÁN DARÍO RODRÍGUEZ SALGUERO**
JULIÁN MAURICIO NOBSA VARGAS***

Palabras Clave: Grid Computacional, Paravirtualización, gLite, Procesamiento Paralelo y/o distribuido.

Descripción: El acelerado avance de la ciencia ha ocasionado que se estudien fenómenos cada vez mas complejos, esto ha llevado a que surjan nuevas necesidades tecnológicas en la investigación mucho mas exigentes, hecho que ha incentivado el desarrollo de nuevas tecnologías que cumplan con los requisitos que imponen estas necesidades. Es así como en el área de la computación de alto rendimiento, el Grid Computacional nace como una respuesta a las necesidades de cómputo intensivo, almacenamiento masivo y trabajo colaborativo ofreciendo la posibilidad de compartir recursos geográficamente distribuidos y organizando los recursos disponibles en una Universidad, empresa u otro tipo de organización.

El Grid Computacional no sólo es un mecanismo enfocado en compartir los recursos físicos, también los recursos lógicos como datos, software, resultados y experiencias con otras organizaciones virtuales que tengan objetivos comunes, constituyendo un entorno que facilita las investigaciones y desarrollos en la medida que se pueden compartir todos estos elementos junto a la capacidad de calculo y almacenamiento que las entidades poseen y ponen a disposición del Grid.

En este proyecto se presenta y se discute la implementación de los módulos y servicios fundamentales para la construcción y puesta en marcha de una infraestructura computacional Grid usando el middleware gLite como un primer paso en la investigación sobre la computación grid en nuestro entorno.

* Trabajo de Grado

** Facultad de Fisico-Mecánicas, Ingeniería de Sistemas e Informática.

***Facultad de Fisico-Mecánicas, Ingeniería de Sistemas e Informática.

ABSTRACT.

Title: IMPLEMENTATION OF BASIC SERVICES AND MODULES FOR THE CONSTRUCTION AND TUNE OF AN INFRASTRUCTURE GRID COMPUTING USING MIDDIEWARE GLITE*

Authors: IVÁN DARÍO RODRÍGUEZ SALGUERO**
JULIÁN MAURICIO NOBSA VARGAS***

Key Words: Grid Computing, Paravirtualization, gLite, Parallel and/or distributed processing.

Description: The fast advancement of science has led to study phenomena increasingly complex, this has led to the emergence of new technology needs in the investigation much more demanding, a fact that has stimulated the development of new technologies that meet the requirements imposed by these needs. This is how in the area of High Performance Computing, The *Computational Grid* born as a response to the needs of computation-intensive, mass storage and collaborative work by offering the possibility of sharing resources geographically distributed and organizing the resources available in a university, company or other organization.

The Computational Grid is not only a mechanism focused on the sharing of physical resources, logical resources as well as data, software, results and experiences with other virtual organizations that have common goals, providing an environment that facilitates the research and developments as they can share all these elements together with the computation and storage capacity of the entities that own and made available to the Grid.

This project is presented and discussed the implementation of the modules and services essential to the construction and commissioning of a Grid computing infrastructure using the middleware glite as a first step in research on grid computing in our environment.

* Grade Work

** Faculty of Physics – Mechanics Engineering Engineering of Systems and Informatics.

***Faculty of Physics – Mechanics Engineering Engineering of Systems and Informatics.

GLOSARIO

CERN: European Organization for Nuclear Research, Organización Europea para la Investigación Nuclear.

Certificado Digital: Es un documento digital mediante el cual un tercero confiable (autoridad de certificación) garantiza la identidad de un sujeto, para esto el certificado digital contiene datos de la identidad del sujeto y está firmado digitalmente con la clave privada de la Autoridad de Certificación. Es la estructura de datos que enlaza la clave pública con los datos que permiten identificar al titular. El formato estándar manejado es el X.509 y su sintaxis, se define empleando el lenguaje ASN.1 (Abstract Syntax Notation One).

Concurrencia: Es la propiedad de los sistemas que permiten que varios procesos sean ejecutados al mismo tiempo, los procesos concurrentes pueden ser ejecutados realmente de forma simultánea cuando cada uno es ejecutado en diferentes procesadores. Opuesto a ello, la concurrencia es simulada cuando sólo existe un procesador encargado de ejecutar los procesos, simulandola y ocupándose de forma alternada entre procesos a intervalos de tiempo muy pequeños dando la impresión de que se están ejecutando a la vez.

Tareas (Informática) : Una sección lógica de un trabajo computacional, comúnmente es una o un conjunto de instrucciones de un programa a ejecutarse en un procesador.

EGEE: (Enabling Grids for E-Science) es un proyecto financiado por la Comisión Europea y tiene por objeto desarrollar una infraestructura Grid cuyo servicio esté a disposición de los científicos las 24 horas del día.

Ejecución Paralela: Ejecución de un programa dividido en múltiples tareas (la mayoría de veces en diferentes equipos) que se ejecutan la misma acción pero con diferentes valores en las variables locales para cada tarea.

Ejecución Serial: Ejecución de programas secuencialmente, una instrucción al tiempo.

Escalabilidad: Es la capacidad de un sistema informático de cambiar su tamaño o configuración para adaptarse a las circunstancias cambiantes.

Large Hadron Collider: Se trata del acelerador de partículas europeo utilizado para recrear las condiciones después del Big Bang, permitirá estudiar y obtener conocimiento acerca de las partículas más pequeñas conocidas y el origen del

universo. Para su construcción se hizo necesario el trabajo conjunto de los países de la unión europea en una nueva tecnología llamada Grid Computing. Debido a que la cantidad de datos a almacenar, 15 Petabytes por año, y el tratamiento de esta información, procesamiento y comunicación requiere de una infraestructura de dimensiones exorbitantes.

Ley de Amdahl's: También conocida como argumento Amdahl's, nombrada en reconocimiento al arquitecto de computadores Gene Amdahl y es usada para buscar la máxima mejora provista para un sistema en general cuando solo una parte del sistema es mejorado. A menudo se utiliza en la computación en paralelo para predecir la velocidad máxima teórica en el uso de múltiples procesadores.

Middleware: Es un software de conectividad que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas, funciona como una capa de abstracción de software distribuida, que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo, red). El Middleware abstrae de la complejidad y heterogeneidad de las redes de comunicaciones subyacentes, sistemas operativos y lenguajes de programación, proporcionando una API para la fácil programación y manejo de aplicaciones distribuidas.

MPI (Message Passing Interface) es un estándar que define la sintaxis y la semántica de las funciones contenidas en una librería de paso de mensajes diseñada para ser usada en programas que exploten la existencia de múltiples procesadores.

Threads: (Hilos) Es una característica que permite a una aplicación realizar varias tareas concurrentemente. Los distintos hilos de ejecución comparten una serie de recursos tales como el espacio de memoria, los archivos abiertos, situación de autenticación, etc.

1. INTRODUCCIÓN

Actualmente los investigadores se enfrentan a fenómenos y problemas de alta complejidad y para apoyar estas labores se han valido de diferentes herramientas como los llamados supercomputadores.

Los supercomputadores se han utilizado para el pronóstico del tiempo, dinámica de fluidos, simulaciones de explosiones nucleares y el estudio mismo del origen del universo,-entre otros fenómenos, a través de aplicaciones con un gran número de variables y ecuaciones que deben resolverse numéricamente a través de una serie de pasos casi incomprensible.

Desde una perspectiva histórica en 1929 el periódico New York World, hace referencia por primera vez al término "Super Computing"[1] comparando un dispositivo que podría ser igual, en términos de resolución de problemas, a 100 matemáticos. Sin embargo, no fue sino hasta los años 70 y 80 en el cual debido a la necesidad de velocidad en la resolución de problemas, principalmente matemáticos, se diseña y comercializa, los "supercomputadores", permitiendo a varios fabricantes entrar en el mercado naciente de la época, encontrándose entre los más destacados Seymour Roger Cray[2].

Las máquinas desarrolladas por Cray eran conocidas como *Supercomputadores Vectoriales* y tenían hardware especial que intentaba manejar grandes cadenas de números en problemas computacionales científicos bastantes complejos.[3] Lo cual sitúa una idea y filosofía que posteriormente, conforme avanzó la tecnología, maduró la supercomputación y junto a la implementación de redes que permitieron compartir recursos se dió paso a nuevos conceptos como *Computación Distribuida y Computación Paralela*.

"Resultaba frustrante que en esos días había diferente información en diferentes computadores, pero tenias que iniciar sesión en cada uno de ellos para obtenerla. A veces también tenías que aprender a manejar diferentes programas en cada

1 Eames, Charles; Eames, Ray (1973). A Computer Perspective. Cambridge, Mass: Harvard University Press, 95. . Page 95 identificó el articulo como "Super Computing Machines Shown", New York World, March 1, 1920. . Sin embargo el articulo mostrado en la pagina 95 referencia The Statistical Bureau in Hamilton Hall y un articulo en el sitio web Columbia Computing History menciona que este articulo no existió hasta 1929

2 Seymour Roger Cray (Septiembre 28, 1925 - Octubre 5, 1996) Ingeniero Eléctrico, y Arquitecto de Supercomputadores

3 John Markoff (2003, August 4). Supercomputing's New Idea Is Old One. New York Times (Late Edition (east Coast)), p. C.1. Retrieved April 8, 2008, from Banking Information Source database. (Document ID: 378956161).

computador" afirma Berners-Lee[4] en su sitio web.

Estas y muchas otras situaciones evidenciaron la necesidad de un suceso que marcara el aceleramiento del desarrollo científico, el suceso se dió y fué el nacimiento de la World Wide Web; red que nació debido a la necesidades interacción y colaboración que tenían los científicos del CERN (**European Organization for Nuclear Research**) en sus proyectos de investigación, entre ellos uno de los más importantes en los últimos tiempos ha sido el intento por revelar el origen del Universo, simular el BigBang bajo un entorno controlado a un nivel Microscópico.

El CERN desde entonces, junto con centros de investigación, universidades y empresas a nivel mundial han hecho parte de muchas innovaciones computacionales que han permitido avanzar en el desarrollo científico donde cada vez se acelera mas con el objetivo de suplir las demandas en soluciones computacionales, lo que implica el desarrollo de nuevas tecnologías que posteriormente se han convertido en tendencias comerciales tecnológicas para la humanidad.[5]

4 Sir Timothy "Tim" John Berners-Lee, OM, KBE (TimBL o TBL) Londres, 1955, Físico de la Universidad de Oxford. Es considerado el padre de la web.

5 John-Paul Kamath (2008, March). From web to grid, Cern's huge data demands forge a path for business. Computer Weekly,12. Retrieved April 7, 2008, from ProQuest Computing database. (Document ID: 1456394681).

2. ESPECIFICACIONES DEL PROYECTO

2.1. TITULO

IMPLEMENTACIÓN DE LOS SERVICIOS Y MODULOS FUNDAMENTALES PARA LA CONSTRUCCIÓN Y PUESTA EN MARCHA DE UNA INFRAESTRUCTURA COMPUTACIONAL GRID USANDO EL MIDDLEWARE GLITE

2.2. DIRECTOR

MsC. Gilberto Javier Díaz Toro.
Universidad de Los Andes, Mérida Venezuela.
gilberto@cecalc.ula.ve

2.3. CODIRECTOR

Ing. Erick Ramón Meneses Cuadros.
Universidad Industrial de Santander, Bucaramanga Colombia.
Erick.Meneses@imag.fr

2.4. AUTORES

Iván Darío Rodríguez Salguero.
Estudiante de Ingeniería de Sistemas
Código. 2022093.
ivan@linux.com.co

Julián Mauricio Nobsa Vargas
Estudiante de Ingeniería de Sistemas
Código. 2012017.
jnobsa@ciencias.uis.edu.co

3. ENTIDADES INTERESADAS EN EL PROYECTO

- Universidad Industrial de Santander – Escuela de Ingeniería de Sistemas e Informática, Bucaramanga - Colombia.

- Grupo de Investigación en Ingeniería Biomédica – GIIB, Bucaramanga - Colombia.

- Centro Nacional de Calculo Científico – CeCalCULA, Mérida - Venezuela.

- Centro de Tecnologías de la Información y la Comunicación de la UIS – CENTIC , Bucaramanga - Colombia.

4. PLANTEAMIENTO DEL PROBLEMA

La Universidad Industrial de Santander en su propósito de generar y adecuar conocimiento, promueve la investigación en las diferentes áreas con las que se ha comprometido, aprovechando los avances tecnológicos existentes para incentivar y facilitar las actividades de investigación promoviendo el desarrollo de proyectos que impacten en la región. Un aspecto de vital importancia desde el siglo pasado, ha sido la evolución tecnológica, específicamente en el área de la computación, evolución que ha permitido detectar enfermedades, pronosticar el estado del clima acertadamente, hacer análisis macroeconómicos, simulaciones de colisiones y situaciones de riesgo e innumerables estudios e investigaciones sobre diferentes áreas del conocimiento. Este tipo de fenómenos o problemas del conocimiento tienen en común la complejidad que es intrínseca en ellos y la alta demanda de recursos computacionales como capacidad de procesamiento , almacenamiento y comunicación.

En la cúspide de este avance tecnológico se ha encontrado con un concepto que a pesar de ser del siglo pasado, es revolucionario y recién toma forma pasando de ser un planteamiento teórico a ser implementado en la practica de las investigaciones más grandes a nivel mundial. El concepto de Grid Computacional, está permitiendo realizar estudios, pruebas, análisis e investigación sobre los datos arrojados en experimentos que antes no se contemplaban. Por otro lado, las empresas e industrias han visto como esta infraestructura puede ayudarles en su crecimiento y expansión, pues nos encontramos en un mundo en el cual las relaciones económicas trascienden fronteras y necesitan tener estrategias de negocio efectivas junto a un respaldo tecnológico que las soporte, aprovechando todos los recursos disponibles en la empresa, brindando un ente organizado en almacenamiento y capacidad de cómputo de las empresas.

Bajo este nuevo panorama, la investigación tiene una base sólida y muy robusta en la cual apoyarse, pero existen diferentes obstáculos para ejercerla de una forma mas avanzada, uno de ellos es que la dimensión de los problemas a tratar en muchas áreas exigen recursos computacionales de altas prestaciones, dichos recursos generalmente son costosos y por lo tanto escasos en las organizaciones, además existen múltiples instrumentos de apoyo como telescopios, microscopios, scanners y otros dispositivos que están fuera del alcance de muchas organizaciones por su costo y mantenimiento, sumado a esto, la falta de un canal de comunicación interactivo y funcional en el que diferentes investigadores en diferentes lugares puedan compartir datos, experiencias y trabajar juntos simultáneamente a través de aplicaciones mediante las cuales interactúen superando la barrera de la distancia y avanzando mas rápidamente, debido a que

trabajando en conjunto la experiencia es compartida y no son obstáculos los problemas ya superados por otros.

Se observa que las necesidades planteadas anteriormente sobre los recursos y práctica de la investigación también se encuentran en la Universidad Industrial de Santander dado que los diferentes centros y grupos de investigación trabajan en problemas altamente demandantes en recursos computacionales, sobre todo en las áreas de ciencias naturales e ingenierías. Dichas necesidades son el terreno para proponer interrogantes en torno a la creación de una infraestructura computacional Grid que en base al intercambio de recursos, ofrezca alto poder computacional tanto en procesamiento como en almacenamiento permitiendo la ejecución de aplicaciones distribuidas que procesan grandes volúmenes de datos con algoritmos complejos como simulaciones de fenómenos, facilitando la comprensión de estos a los investigadores de la UIS.

De igual manera, dicha infraestructura Grid, deberá abrir la posibilidad mediante convenios inter-institucionales, de acceder a otro tipo de recursos que no estén presentes en la Universidad Industrial de Santander y a su vez también debe establecerse como canal de cooperación entre diferentes institutos y Universidades facilitando el trabajo colaborativo entre investigadores a través de proyectos como Grid Colombia.
Es por esto que este proyecto busca dar solución a cuestiones importantes en nuestro entorno Universitario, tales como:

- ¿Como construir una infraestructura grid que cumpla con las necesidades de cómputo de alto rendimiento, almacenamiento e intercambio de experiencias, conocimiento y recursos en las labores de investigación de la Universidad Industrial de Santander ?
- ¿Como hacer de este grid un recurso real, funcional y de fácil uso y acceso a los científicos, profesores, investigadores, ingenieros que hacen parte de la comunidad universitaria en la UIS?
- ¿Este Grid Computacional podría conectarse con otro grid similar con el objetivo de avanzar en las relaciones de confianza entre los Grid Computacionales académicos, constituyendo comunidades entre instituciones distantes compartiendo recursos geográficamente distribuidos?
- ¿Como montar y ejecutar una aplicación para grid, con el fin de comprobar su funcionamiento a través de pruebas de conexión con otro grid?

5. OBJETIVOS

5.1. OBJETIVO GENERAL

Implementar los componentes fundamentales del middleware gLite para la construcción de una plataforma computacional distribuida Grid, que cumpla con las necesidades de cómputo de alto rendimiento presentes en las labores de investigación de la Universidad Industrial de Santander

5.2. OBJETIVOS ESPECIFICOS

- Realizar una revisión bibliográfica y estudio del estado del arte de los conceptos fundamentales de la infraestructura Grid Computacional y del middleware gLite.
- Configurar los equipos de la sala de supercomputación del CENTIC con los requerimientos necesarios para la implementación del middleware gLite.
- Implementar los componentes y servicios fundamentales que hacen parte de la infraestructura del middleware gLite con el fin de configurar, realizar pruebas, puesta a punto y poner en marcha el Grid Computacional.
- Realizar pruebas de conexión (velocidad, latencia) e intercambio de recursos computacionales(almacenamiento, procesamiento) con la Universidad de los Andes en Mérida, Venezuela, utilizando la infraestructura grid.

6. JUSTIFICACIÓN

Actualmente la investigación requiere de recursos de alta capacidad de computo, integración y cooperación de diferentes organizaciones para alcanzar objetivos comunes, y es aquí donde aparece el concepto de computación en grid, una infraestructura que propone un modelo de trabajo en el cual se comparten los recursos de forma descentralizada y coordinada para resolver problemas altamente demandantes en recursos computacionales (almacenamiento, procesamiento, comunicación y cooperación) en organizaciones geográficamente distribuidas, dependiendo de la disponibilidad, capacidad, desempeño y calidad de servicio requerida por los usuarios. La Universidad Industrial de Santander debido a su naturaleza es una entidad que contribuye en el desarrollo social, tecnológico y económico de la sociedad y evidencia la necesidad de tener una infraestructura de tal magnitud, que permita contribuir y acelerar el avance científico e ingenieril brindándole el soporte a los científicos, ingenieros, especialistas y doctores para las investigaciones que estos realicen.

La implementación de este grid computacional, ubicará a la Universidad Industrial de Santander a la vanguardia de esta tecnología y posibilitará el desarrollo de futuras aplicaciones y proyectos enfocados en el área de Super-Computación, entre ellos la iniciativa de Grid Colombia y el proyecto EELA (E-Infraestructure shared between Europe and Latin America), aprovechando los recursos del Centro de Tecnologías de la Información y Comunicaciones (CENTIC). A su vez la Escuela de Ingeniería de Sistemas incursionará en esta área tan importante a nivel mundial en la actualidad, permitiendo la interacción científica y académica con otros centros de investigación, entre ellos el Centro Nacional de Cálculo Científico de la Universidad de los Andes en Mérida, Venezuela.

Finalmente la implementación de esta infraestructura computacional, permitirá generar una nueva área en el desarrollo de software, anexando el desarrollo de aplicaciones para grid y fortaleciendo los proyectos en el área de supercomputación estableciéndola como una corriente de trabajo interdisciplinario al interior de la Universidad y así mismo, posibilitará el trabajo mancomunado entre ingenieros y científicos de la Universidad Industrial de Santander.

7. MARCO TEORICO

7.1. COMPUTACIÓN PARALELA Y DISTRIBUIDA

La Computación Paralela, es un modelo de cómputo que permite trabajar instrucciones de manera simultanea en cada uno de los equipos que se tengan conectados entre si a través de una red, subdividiendo de esta manera, un problema grande en tareas más pequeñas las cuales pueden correr de manera concurrente, trabajando todos como una unidad resolviendo un mismo problema. Teniendo en cuenta que los computadores están diseñados para funcionar de manera secuencial, este nuevo modelo de desarrollo representa un desafío para los programadores, debido al cambio de paradigma en la programación ya que es necesario realizar una buena división de trabajo, además de comunicación y sincronización entre los procesos que corren en paralelo. El resultado de la paralelización en los sistemas computacionales es mayor rendimiento en la ejecución de las instrucciones (condicionado a la Ley de Amdahl's).

Existen varios modelos de programación que permiten distinguir una ejecución en paralelo:

- Shared Memory: En este modelo, las tareas comparten una espacio de direcciones común, en el cual leen y escriben de manera asíncrona.

- Threads: En el modelo Threads de programación en paralelo, un proceso sencillo puede tener múltiples caminos de ejecución concurrente.

- Message Passing: Grupo de tareas que usa su propia memoria local durante el proceso de computo, múltiples tareas pueden residir en la misma máquina física así como a través de un número arbitrario de máquinas.

- Data Parallel: La mayoría de los trabajos paralelos se enfocan en realizar operaciones a un conjunto de datos. Estos conjuntos de datos están típicamente organizados en una estructura común, como un arreglo o un cubo, sin embargo cada trabajo opera en una diferente partición de la misma estructura de datos.

- Hybrid: En este modelo, varias formas de programación paralela son combinados, un ejemplo común es la combinación de Message Passing Modle (MPI) con el modelo de Hilos (POSIX Threads) o el modelo de memoria compartida (OpenMP).

Algunas características de la computación en paralelo son:

- Uso de múltiples CPUs para correr aplicaciones computacionales
- Un problema es dividido en partes que pueden correr en forma concurrente.
- Cada subdivisión es ejecutada por cada CPU secuencialmente de manera simultánea.
- Ahorro de tiempo en la ejecución del programa computacional.
- Posibilita la solución de problemas complejos que puedan paralelizarse.

Por otro lado la computación distribuida, al igual que la computación paralela, es un método que permite correr aplicaciones simultáneamente en varios computadores que se comuniquen por medio de una red, pero presentando diferencias con la computación paralela, la más notoria es que la computación en paralelo necesita de sistemas dedicados y homogéneos para poder hacer la ejecución paralela, mientras que la computación distribuida permite ejecutar una aplicación en sistemas que a su vez trabajan de manera independiente sin la necesidad de que la ejecución de las tareas sea concurrente, de esta forma, con la computación distribuida se puede aprovechar el tiempo que pasan los computadores desocupados o con poca carga, es decir el tiempo de ocio, esto sumado a la popularidad de internet permite que surjan posibilidades de compartir recursos geográficamente distribuidos.

Una analogía es pensar que en computación en paralelo puede atribuirse a 10 hombres tirando de una única cuerda para levantar una carga, pero si los 10 hombres tuviesen cada uno una cuerda y tiraran de diferentes lados de la carga seria un método análogo a la computación distribuida.

7.2. VIRTUALIZACIÓN

Virtualización es un término amplio que en el contexto de la computación se refiere a la abstracción de los recursos de una computadora. El objetivo en común de todas las tecnologías de virtualización es el de ocultar los detalles técnicos a través de la encapsulación.

Cuando se habla de virtualizar hoy en día, se refiere a ejecutar un sistema operativo dentro otro. Esto presenta ventajas como:

- La consolidación, que consiste en tener en un único computador varios servicios que no utilizan completamente los recursos de una máquina ejecutándose en diferentes instancias de sistema operativo.
- Aprovechamiento al máximo de los recursos computacionales.

- Escalabilidad en la creación de nuevos sistemas operativos virtuales, debido a que no se necesita instalar y configurar desde cero sino que basta con copiar la imagen del S.O a virtualizar al disco, disponiendo de un sistema ya creado y configurado para personalizar.
- Ejecución de aplicaciones críticas en sistemas operativos viejos, mediante la virtualización se puede hacer que el S.O que necesita la aplicación se ejecute en hardware moderno.
- Seguridad, en caso de un ataque solo se vería comprometida una máquina virtual ya que se encuentran aisladas entre si.
- Facilidad para la realización de pruebas, en un entorno de desarrollo, la virtualización permite realizar las pruebas que se necesiten en un entorno controlado, sin poner en riesgo el equipo y sin reinstalar el S.O.

De igual manera la virtualización proporciona ahorro de espacio físico y ahorro energético ya que hay menos máquinas físicas conectadas y menor necesidad de refrigeración.

7.3. CLUSTER COMPUTACIONAL

Los dos enfoques , *Computación Distribuida* y *Computación Paralela* permitieron y aún lo hacen, resolver de manera eficiente los problemas altamente demandantes en recursos computacionales y que necesitan ser resueltos en el menor tiempo posible, introduciendo nuevas ideas acerca de arquitecturas como es el Cluster Computacional, el cual consiste en un grupo desacoplado de computadores que trabajan juntos y que puede ser visto como si fuese un solo computador, generalmente la interconexión entre estos computadores es por medio de redes de alta velocidad. Esta solución brinda la alternativa de poder usar en lugar de un gran computador de altas prestaciones y costoso, varios computadores de menores capacidades en hardware permitiendo ser una arquitectura escalable, confiable y segura.

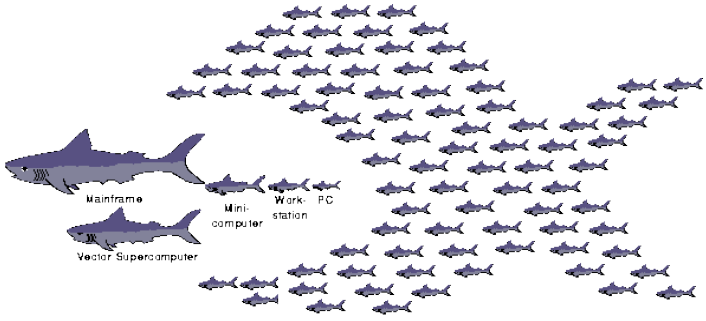


Figura 1: Cluster Computacional.
Fuente: http://clusterfie.epn.edu.ec/clusters/Images/clusters.gif

Existen varios tipos de cluster, dependiendo de la necesidad que se tenga; entre ellos se encuentran:

- Cluster de Alto Rendimiento:

  Es una agrupación de máquinas interconectadas que colaboran conjuntamente para la solución de una tarea que exige altos niveles de cómputo, esta diseñado para dar altas prestaciones en cuanto a capacidad de cálculo, la estructura mas común consiste en nodos esclavos que son controlados por un nodo maestro, este es quien envía trabajos a los diferentes nodos esclavos y estos procesan las tareas asignadas, reportando al final sus resultados al nodo maestro, de igual forma existen otras estructuras que son réplicas y combinaciones de ésta, como varios maestros controlando cada uno un grupo de esclavos pero siendo controlados a su vez por un maestro principal, formando una estructura de árbol.

- Cluster de Alta Disponibilidad:
  Es un conjunto de dos o más máquinas que se caracterizan por mantener una serie de servicios que redundan entre ellos y por estar constantemente monitoreandose entre sí, de tal forma que cuando una de las máquinas del cluster falla de alguna manera, la otra toma automáticamente toma la carga de esta sustituyéndola, manteniendo la integridad de la información y la oferta de servicios evitando conflictos a los usuarios quienes no notan que se produjo un problema.

- Cluster de Balanceo de Cargas:
  Está compuesto por uno o más nodos que actúan como frontend del cluster y que se ocupan de recibir las peticiones de servicio que este reciba y repartirlas a otras máquinas del cluster que forman el back-end de éste. Su función es repartir la carga de proceso entre los nodos.

Para implementar un Cluster Computacional, se necesita incorporar un middleware que permita la ejecución de código en paralelo, El middleware es un software que generalmente actúa entre el Sistema Operativo y las aplicaciones con la finalidad de proveer a un cluster de:

- Interfaz de acceso al sistema , que genera la sensación que el usuario se encuentra en un único ordenador muy potente.
- Escalabilidad
- Optimización y mantenimiento del sistema.

Existen diversos middlewares como Mosix, OpenMosix, Condor, OpenSSI, PVM,

MPI. Estos middleware incorporan librerías y funcionalidades para que a través de lenguajes de programación como C, Fortran, Python, Ocaml, Ruby, Python, Smalltalk se puedan escribir programas de ejecución paralela.

7.4. GRID COMPUTACIONAL

En un cluster computacional, los nodos deben estar ubicados en el mismo lugar, pero si se pudiese utilizar recursos que están distribuidos geográficamente en otros lugares a través de redes de área extensa, se tendría una inmensa capacidad de cómputo y almacenamiento sin preocuparse por la ubicación de estos, tal como ocurre con otro tipo de recursos como la electricidad o el agua potable, donde el recurso es distribuido mediante redes y el usuario sencillamente lo aprovecha.

En el mundo moderno, es común poder sentarse en cualquier café pedir un expreso y hacer uso de internet para lo que se crea necesario, sin saber de donde viene el agua, el café, la electricidad ó la conexión a internet que se está usando.

La disponibilidad de estos recursos ejemplifica el concepto de "virtualización", que se refiere a esconder las funciones de utilidad detrás de una interfaz que oculta los detalles de como están implementadas. Cuando el empleado del café, por ejemplo, gira el grifo del agua, es como si el grifo fuese un barril sin fondo, el mismo fenómeno ocurre cuando se conecta el computador portátil en un enchufe, dada la enorme red eléctrica invisible más allá del enchufe, quien sabe cómo y dónde se generó este poder?. No se concibe actualmente tener en nuestras casas u oficinas generadores de corriente o reservas de agua, ¿porque no pensar lo mismo para la computación?.[6]

Disponer de estos recursos da una idea de lo grandioso y revolucionario que son las redes eléctricas, pues el verdadero desarrollo revolucionario no es, de hecho, la electricidad sino las redes eléctricas y las correspondientes tecnologías de transmisión y distribución. Con un bajo costo de acceso se estandarizó un servicio para toda la humanidad permitiendo el desarrollo de nuevos dispositivos y nuevos modelos industriales con una aceleración que generó una era que marcaría la diferencia. Por analogía Ian Foster[7] y Carl Kesselman[8] en 1998 adoptan el término *"Computational Grids" para una infraestructura de hardware y software que provea*

6 Ian Foster (2003, April). The Grid Computing without bounds, Scientific American v288 no4 p78-85.

7 Científico Senior en la División de Computer Sciencie y Matemáticas de el Laboratorio Nacional de Argonne, denotado comúnmente como “El padre del Grid”. Pagina web personal: http://www-fp.mcs.anl.gov/~foster/

8 Es el lider de proyecto en University of Southern California's Information Sciences Institute, es muy reconocido en el campo de las tecnologías Grid Computacionales.

de un acceso a un gran capacidad recursos computacionales de gama alta frecuente, fiable, coherente y de bajo costo.[9]

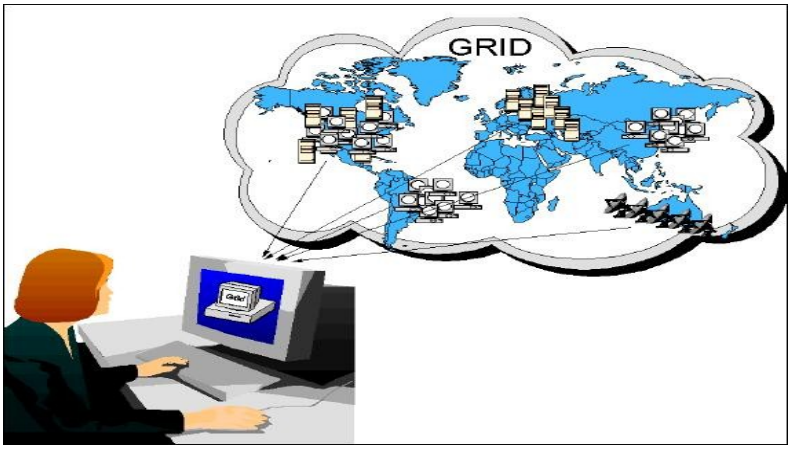


Figura 2: Grid Computacional.
Fuente:http://www.it.uom.gr/teaching/unc_charlottePPG/images/fig1_1.jpg

Esta definición ha de ser refinada después, refiriéndose a que la Computación grid concierne a "Poder compartir recursos de manera coordinada y solucionar problemas entre organizaciones virtuales dinámicas multi-institucionales", esto junto a un contexto social y político en el cual la idea es poder negociar distribución de recursos compartidos entre un conjunto de las partes participantes y posteriormente utilizar estos recursos para sus fines.[10]

Se podría resumir que la esencia de la definición se refiere a una lista en la cual se describe que debe tener/hacer un Grid computacional:

- Recursos coordinados sin estar sujetos a control centralizado.
- Usar protocolos e interfaces estándares, abiertos y de propósito general.
- Ofrecer una calidad de servicio no trivial (en relación a tiempo de respuesta, rendimiento, disponibilidad y seguridad).

En la lista anterior Ian Foster menciona lo que es un Grid más no lo que se considera "The Grid", es importante hacer la distinción, la visión de "The Grid" requiere introducir protocolos, interfaces y políticas que sean abiertas, estándares y de propósito general. La estandarización es lo que garantiza establecer el conjunto de recursos compartidos dinámicamente con cualquier parte interesada y crear algo más que un incompatible y no inter-operable sistema distribuido,

9 Ian Foster (1998). Computational Grids, Chapter 2 of The Grid: Blueprint for a New Computing Infrastructure.

10 Ian Foster (2001). The anatomy of the grid: Enabling scalable virtual organizations

además de un conjunto de normas como medio que permita el uso general de los servicios y herramientas.[11]

Existe un conjunto integrado de componentes y servicios que hacen posible el funcionamiento de las aplicaciones, el intercambio de recursos y el control y sincronización de toda la estructura que conforma el Grid, funcionando como una capa de abstracción de software distribuida situada entre la capa de aplicaciones y las capas inferiores como la del sistema operativo o la red, abstrayéndonos de la complejidad y heterogeneidad de estas últimas, éste conjunto se conoce como middleware y han sido desarrollados por diferentes personas en todo el mundo, algunos de estos son: Globus Toolkit, gLite, GRIA, DAT, LCG y OAR.

Algunas características del Grid Computacional son:

- Compartir recursos a gran escala. Poder compartir una serie de recursos de forma coordinada entre los usuarios, sin tener en cuenta la ubicación geográfica de los recursos.
- Computadores encargados de proporcionar potencia de cálculo y almacenamiento.
- Redes de comunicaciones que permiten acceder a nuevos recursos interconectando diversos sistemas.
- Instrumentos generalmente de carácter científico necesarios en algunas aplicaciones.
- Recursos heterogéneos.
- Coordinación de recursos distribuidos que pueden pertenecer a una o varias organizaciones ubicadas en diferentes sitios que tengan sus propios administradores donde se determina que compartir, a quien y las condiciones bajo las cuales se realiza este procedimiento.

7.4.1. ESTADO DEL ARTE

Un problema crítico al cual se enfrenta la comunidad de Grid hoy en día proviene por parte de los estándares para los middleware, haciendo notar un progreso en el Global Grid Forum, con mas de 6 años de trabajo y experiencia se desarrolla con código abierto el Globus Toolkit, el cual permite implementar la idea de Grid Computacional, este es desarrollado por la Globus Alliance.

Este avance en la tecnología Grid, ha hecho posible la creación de diferentes implementaciones de infraestructuras como:

11 Ian Foster (July 20, 2002). What is the Grid? A Three Point Checklist. Argonne National Laboratory & University of Chicago

- Teragrid: Infraestructura científica abierta que combina grandes recursos computacionales (Supercomputadores, Almacenamiento y sistemas de Visualización Científica)[12]
- EPCC: Es un puente al mundo de la computación avanzada para la industria, comercio e investigación. Fundada en la Universidad de Edinburgo.[13]
- OSG: Infraestructura para trabajar ciencia a gran escala, construido y operado por un consorcio de Universidades Estadounidenses y laboratorios de ese país.[14]
- Gilda: Proyecto italiano del Instituto Nacional de Física Nuclear que pone a disposición de la comunidad mundial un grid de prueba y que además permite que se anexen sitios voluntariamente a esta infraestructura.[15]

Además estas implementaciones de Grid Computacionales permitieron desarrollar proyectos como:

- Genome@home : Proyecto que busca descifrar el Genoma Humano y su aplicación.[16]
- Folding@home : Proyecto que estudia el plegamiento de las proteínas.[17]
- FightAIDS@home : Estudio para descubrir la cura del SIDA.[18]
- Predictor@home : Estudio de enfermedades relacionadas con proteínas[19]

De igual manera otros proyectos como el Colisionador de Hadrones (Large Hadron Collider LHC), experimento del CERN que fué iniciado el 21 de octubre de 2008 y postpuesto para los primeros meses de 2009 proporcionará gran cantidad de datos (15 Petabytes de información) que hacen necesaria la implementación de un Grid Computacional para realizar un análisis, almacenamiento y disposición de estos por medio del proyecto EGEE (Enabling Grids for E-sciencE)[20].

El proyecto EGGE opera con más de 250 sitios, centros de investigación,

12 http://www.teragrid.org/
13 http://www.epcc.ed.ac.uk/
14 http://www.opensciencegrid.org/
15 https://gilda.ct.infn.it/
16 Desarrollado por el departamento de Química y Biología estructural de la Universidad de Stanford, sitio web: http://genomeathome.stanford.edu/
17 Desarrollado por el departamento de Química y Biología estructural de la Universidad de Stanford, sitio web:http://folding.stanford.edu/
18 Desarrollado por el departamento de Biología molecular del Instituto de inve4stigación Scripps en California, sitio web http://fightaidsathome.scripps.edu/
19 Desarrollado como parte del proyecto Berkeley Open Infraestructure for Network Computing BOINC, sitio web http://predictor.chem.lsa.umich.edu/
20 http://www.eu-eela.eu/

Universidades, Compañías y demás interesados. La infraestructura comenzó a implementarse con el middleware LCG-2, utilizado y basado en el middleware que usa el proyecto TeraGrid, mientras en paralelo han desarrollado el middleware gLite, creando por medio de re-ingeniería de varias fuentes un middleware ligero que provee un rango muy amplio de servicios y componentes básicos para Grids Computacionales.

Este nuevo desarrollo incentivó un proyecto llamado EELA; el objetivo es llevar la e-infraestructura de los países de América Latina al nivel de las de Europa aprovechando el apoyo específico de la Comisión Europea. EELA se beneficiará del estado de madurez del proyecto ALICE[21] y de la red RedCLARA[22] para centrarse en una infraestructura Grid y aplicaciones de e-Ciencia relacionadas, identificando y promoviendo un marco sostenible para la e-Ciencia. El enfoque de EELA es crear una red de colaboración que se ocupe de la organización de la formación en las tecnologías Grid y del despliegue de una infraestructura grid piloto para aplicaciones de e-ciencia con latinoamérica.

En latinoamérica el país pionero en computación en grid es Venezuela, mediante la Universidad de los Andes en Mérida la cual a través del Centro Nacional de Cálculo Científico CeCalCULA[23] realizó el primer taller latinoamericano de grid el 15 de noviembre de 2004 y poco a poco fue consolidando una base de investigadores que dedicados a tiempo completo y mediante proyectos de estudiantes de pregado y maestría lograron construir una infraestructura grid con gLite como middleware que hoy en dia sirve de soporte para las actividades de investigación de varios institutos de todo Venezuela. De igual manera han realizado varios convenios con institutos de Europa para impulsar a través del proyecto EELA la implementación de grid en latinoamérica realizando talleres y compartiendo su infraestructura de recursos.

En Colombia, la iniciativa Grid Colombia, es otro proyecto que tiene como fundamento ser *"una organización virtual en proceso de formación, con una base fundamentalmente académica, destinada originalmente a centralizar los esfuerzos para la creación de una malla computacional de propósito académico en Colombia usando las* ***Redes de Tecnología Avanzada (RENATA)[24]*** *a escala regional y nacional*".

Grid Colombia lo conforman actualmente las Universidades:

- Universidad de Antioquia.

21 http://www.comp.nus.edu.sg/~teoym/alice.htm
22 http://www.redclara.net/
23 http://www.cecalc.ula.ve/
24 http://www.renata.edu.co/

- Universidad Pontificia Bolivariana.
- Universidad Industrial de Santander.
- Universidad Pontificia Javeriana.
- Universidad Nacional de Colombia.
- Universidad de los Andes.
- Politécnico Gran Colombiano.
- Universidad Manuela Beltran.

Al igual que EELA, Grid Colombia usara el middleware gLite el cual ha tenido bastante aceptación en las iniciativas de construcción de Grids Computacionales. Desafortunadamente este proyecto avanza muy lentamente y no es algo consolidado aún, los trabajos adelantados en la computación en grid son pocos y aislados.

7.4.2. ARQUITECTURA GENERAL DE GRID

La arquitectura de un Grid independiente del middleware se puede generalizar en las siguientes 4 capas:

- Capa de red *Network Layer*
- Capa de recursos *Resource Layer*
- Capa de middleware *Middleware Layer*
- Capa de aplicaciones y servicios software *Application and serviceware Layer*

cada una de ellas agrupa un conjunto de servicios que se encargan de realizar labores enfocadas a un mismo objetivo específico. Las dos primeras están más enfocadas hacia el hardware y las dos últimas hacia el usuario.

7.4.2.1. CAPA DE RED

Las redes son una pieza fundamental de la capa subyacente al hardware, estas se caracterizan según su cobertura (local, nacional, internacional) y su desempeño, entendido como la cantidad de datos que puede transmitir en determinado intervalo de tiempo. El desempeño generalmente se mide en Kilo, Mega o Giga Bits por segundo, sin embargo ya que con el grid se utilizan recursos geográficamente dispersos, son necesarias redes de área extensa y de alto desempeño que puedan transmitir a altas velocidades.

En esta capa se encuentran protocolos estándar de seguridad y comunicación para transacciones de red. Los protocolos de comunicación permiten el

intercambio de datos entre la capa más inferior y la de recursos mientras que los protocolos de seguridad brindan mecanismos de criptografía para identificar usuarios y recursos.

7.4.2.2. CAPA DE RECURSOS

Los recursos computacionales que serán compartidos por los diferentes grupos de usuarios: servidores de almacenamiento masivo, supercomputadores, clusters y hardware especializado, junto con la infraestructura de red y sus mecanismos de gestión y control se ubican en esta capa.

7.4.2.3. CAPA DE MIDDLEWARE

El middleware es el software que organiza e integra los servicios en un grid, consiste en un conjunto de servicios software que implementan el acceso uniforme a los datos y recursos, la autenticación y autorización, el manejo de la información de los recursos y la planificación de la asignación de los recursos. Acá también se encuentran los servicios que permiten obtener la información de un recurso en particular y gestionarlo controlando el acceso, arranque de procesos, gestión, monitorización y auditoría.

7.4.2.4. CAPA DE APLICACIONES Y SERVICIOS SOFTWARE

Las aplicaciones son construidas en términos de servicios definidos para alguna de las capas antes mencionadas, pudiendo por ejemplo comunicarse directamente con una capa en particular. Las aplicaciones permiten que los usuarios interactúen con ellas de diferentes maneras brindándoles facilidades para trabajar en diferentes áreas.

Las aplicaciones de grid necesitan entenderse con nuevas capas de software, por ejemplo una aplicación de grid que haga análisis de datos localizados en diferentes fuentes debe poder obtener las credenciales necesarias para autenticación para poder acceder a los archivos, consultar el ***Information Service*** y el ***File Catalog*** para determinar donde están los archivos y que recursos pueden analizarlos, enviar solicitudes a la capa de recursos y de redes para extraer datos, iniciar cálculos y proveer los resultados y finalmente monitorizar el progreso de los cálculos y las transferencias de datos así como notificar al usuario cuando lo análisis hayan finalizado y detectar y responder a fallos.

Esta arquitectura se puede observar en la siguiente figura:

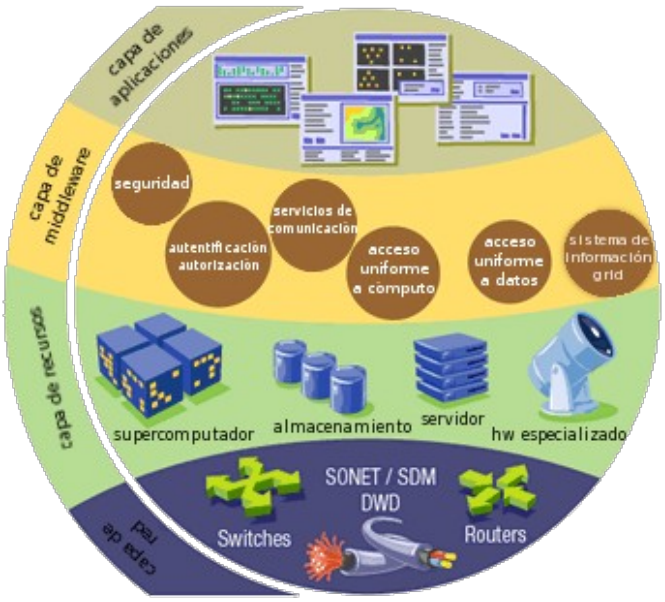


Figura 3: Diagrama de Capas.
Fuente:http://gridcafe.web.cern.ch/gridcafe/gridatwork/images/gridlayer.gif.

7.4.3. VENTAJAS DE LA COMPUTACIÓN GRID

La computación en grid presenta una serie de ventajas algunas son:

- Es una solución altamente escalable.
- Integra sistemas y dispositivos heterogéneos.
- Mejora la colaboración entre las organizaciones diversificadas.
- Permite afrontar la solución a nuevos procesos que no se podían realizar por su complejidad.
- Incrementa la productividad, proporcionando a usuarios finales un acceso sin restricción a los recursos informáticos de datos y de almacenamiento que necesitan.
- Debido al concepto de Computación Grid, de estándares abiertos permite crear una infraestructura única y unificada, liberando a las organizaciones de tecnologías de la información del peso de administrar sistemas no integrados, reduciendo de esta forma la supervisión.
- Se pueden enviar tareas para que sean ejecutadas en aplicaciones que están disponibles sólo en algunos equipos, esta clasificación conlleva ahorro en los costos de licencias.

- Utilización de diferentes equipos de alto costo ubicados en lugares distantes y que no son asequibles en el sitio de trabajo de algunas organizaciones.

7.5. DESCRIPCIÓN DE LA ARQUITECTURA DEL MIDDLEWARE gLite

Es la nueva generación de middleware para *Grid Computing*, ha sido desarrollado inicialmente por el esfuerzo colaborativo de mas de 80 personas en 12 diferentes centros de investigación industriales y académicos.

El modelo de gLite proviene de contribuciones de otros proyectos como el LCG y VDT, este modelo consiste en construir diferentes tipos de servicios ('node-types') a partir de los componentes y permitir una fácil instalación y configuración en la plataforma que se trabaja, Scientific Linux, versiones 3 y 4.

7.5.1. MÓDULOS DE SERVICIOS EN EL MIDDLEWARE GLITE

EL middleware gLite esta estructurado en una serie de módulos que se encargan de aspectos específicos del grid como el esquema de seguridad, los servicios de información y monitoreo, gestión de trabajos y servicios de datos. A su vez estos módulos están compuestos por una serie de actores que configurados, sincronizados y relacionados fuertemente entre si, permiten que los módulos de cada tipo de servicio funcionen correctamente. Los módulos en gLite son los siguientes:

Security Services (servicios de seguridad):

En un grid las comunidades de usuarios se agrupan en organizaciones virtuales. Antes hacer uso de los recursos, el usuario debe leer y aceptar las reglas de uso y las normas de la organización virtual a la que desea unirse, también debe registrar algunos datos personales en la inscripción.

Una vez que el usuario ha completado el registro, se puede acceder al Grid. La Infraestructura de Seguridad del Grid (GSI) permite la autenticación y la comunicación segura a través de una red abierta, esta infraestructura se basa en el cifrado de clave pública, certificados tipo X.509 y el protocolo de comunicación *Secure Sockets Laye*r (SSL), con extensiones para el inicio de sesión único y delegación.

Con el fin de autenticarse en los recursos del Grid, el usuario debe tener un certificado digital X.509 emitido por una Autoridad de Certificación (CA) de

confianza por el Grid.

El certificado de usuario, cuya clave privada está protegida por una contraseña, se utiliza para generar y firmar un certificado temporal llamado certificado *proxy* (o simplemente un *proxy*) el cual es presentado para la autenticación ante los servicios del Grid y no necesitar una contraseña. La posesión de un certificado *proxy*. Como la posesión de un certificado *proxy* es una prueba de identidad, el contenido del archivo solo debe ser leído por el usuario, el *proxy* tiene por defecto un tiempo de vida corto (por lo general 12 horas) para reducir los riesgos de seguridad en caso de que fuese robado.

Los principales componentes que implementan el módulo de servicios de seguridad en gLite son la autoridad certificadora encargada de emitir los certificados digitales, el ***Virtual Organization Membership Service*** que se encarga de autorizar a los usuarios para el uso de determinados recursos del grid. Además de estos autores existen diferentes esquemas, protocolos y servicios que soportan a niveles mas bajos las operaciones entre los elementos mas importantes.

Information and Monitoring Services (Servicios de información y monitoreo):

Proporciona información acerca de los recursos del Grid y su estado. Esta información es esencial para el funcionamiento de todo el Grid, ya que es a través de éste que los recursos son descubiertos. Esta conformado por dos servicios de información (*Information Service*) principales; el *Globus Monitoring* and *Discovery Service (MDS)* que descubre recursos y publica su estado, y el *Relational Grid Monitoring Architecture (R-GMA*) que contabiliza, monitorea y publica información a nivel de usuario. Estos a su vez se basan en componentes como el *GLUE Schema* el cual es un modelo de datos común para descubrir y monitorear recursos que funciona con el *Berkeley Database Index Information;* una base de datos OpenLDAP que es poblada por servicios de información a diferentes niveles con la información del estado de los recursos.

Para la gestión de la información del estado de los trabajos, existe un componente llamado *Logging and Bookeeping* que va registrando en cada acción el estado de los trabajos manteniendo información actualizada de estos.

Data Services (servicios de datos)

La unidad primaria para el manejo de datos en Grid, así como en la informática tradicional es el archivo. En un entorno Grid, los archivos pueden estar replicados en diferentes lugares. Debido a que todas las réplicas deben ser coherentes, los archivos en Grid no pueden modificarse después de su creación, sólo leerlos y

eliminarlos. Idealmente, los usuarios no necesitan saber donde está ubicado un archivo ya que este servicio se encarga de manejar las tareas de almacenamiento, transacción, control e información de los datos a usar en los trabajos. Para esto se soporta en elementos de almacenamiento *(storage element),* los cuales son servidores o dispositivos con capacidad de almacenamiento masivo, el *File Catalog* o catálogo de archivos que permite la ubicación de determinado archivo que necesite un trabajo y protocolos especiales de transferencia de archivos.

Job Management Services

Es mediante el cual se gestiona todo lo referente a la ejecución de los trabajos. Se conforma de los elementos de cómputo (*computing elements*) quienes coordinan la ejecución de los trabajos en las máquinas finales, el *Workload Management System* quien se encarga de recibir los trabajos enviados y encontrar los recursos apropiados para su ejecución.

Mas adelante se explica a fondo la instalación configuración, sincronización y funcionamiento de estos elementos y de otros que se ubican entre estos para posibilitar la comunicación y los objetivos de los componentes mas generales.

7.5.2. COMPONENTES DEL MIDDLEWARE GLITE

Los servicios antes mencionados se componen de los siguientes nodos:

- User Interface (UI):
  La interfaz de usuario (UI) es el punto de acceso al Grid, esta puede ser cualquier máquina donde los usuarios tienen una cuenta personal y en la cual el certificado de usuario está instalado. Desde una interfaz, el usuario puede ser autenticado y autorizado para utilizar los recursos, e inmediatamente acceder a las funcionalidades que ofrecen los sistemas de Información, Workload y Data Manager. La interfaz de usuario proporciona herramientas para realizar algunas operaciones básicas en el Grid como obtener una lista de todos los recursos adecuados para ejecutar un determinado trabajo, enviar los trabajos para su ejecución, cancelar trabajos, recuperar los resultados de trabajos terminados, mostrar el estado de los trabajos enviados, copiar, reproducir y borrar archivos del grid y recuperar el estado de los diferentes recursos desde el Sistema de Información.

- Computing Element (CE):
  El Computing Element (CE) es el servicio que representa un recurso de computo. Su principal funcionalidad es la gestión de trabajos (envío de trabajos, control de trabajos, etc.). El CE puede ser utilizado por un cliente

genérico: un usuario final interactúa directamente con el Computing Element, o el Workload Manager, el cual envía un determinado trabajo a un Computing Element adecuado encontrado mediante un proceso de emparejamiento. Para el envío de trabajos, el CE puede trabajar con el modelo “push” (cuando el trabajo es llevado a un CE para su ejecución) o con el modelo “pull” (en el que el Computing Element está preguntando al servicio Workload Manager por los trabajos).

Además de la capacidad de gestión de trabajos, un CE también debe proporcionar información sobre sí mismo. En el modelo “push” esta información es publicada en el Servicio de información, y es utilizada por el motor de emparejamiento, asignando los recursos disponibles a los trabajos en cola. En el modelo “pull” la información del CE es embebida en un mensaje de “disponibillidad de CE”, que es enviado por el CE al servicio de Workload Manager. El matchmaker (emparejador) entonces utiliza esta información para encontrar un trabajo adecuado para el CE.

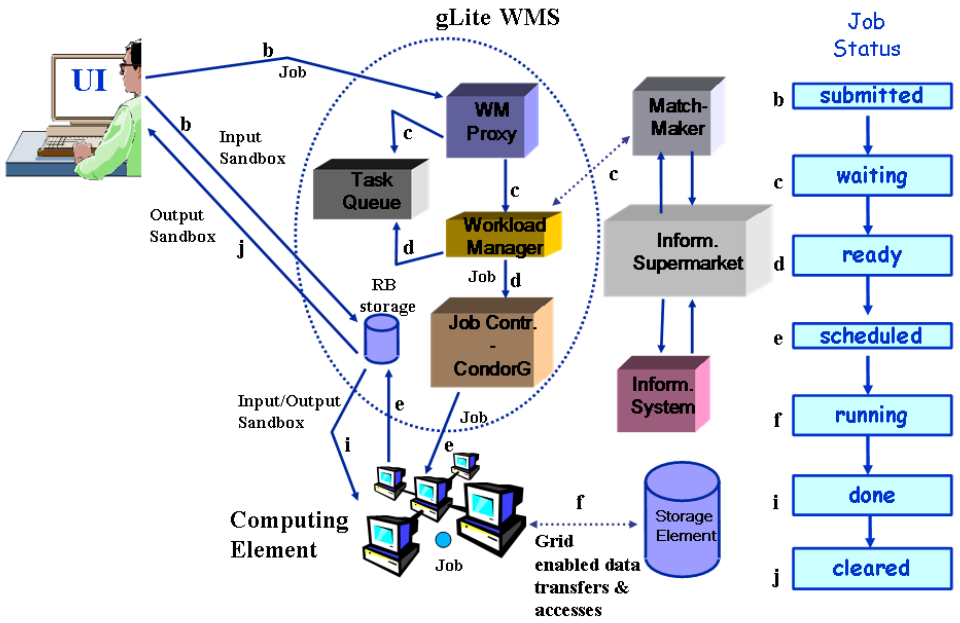
- Storage Element (SE):
  Es el servicio que permite a un usuario o una aplicación guardar datos para la futura recuperación y uso de los mismos. Es consultado por las aplicaciones que requieran estos datos. Un Storage Element (SE) proporciona acceso uniforme a los recursos de almacenamiento de datos. El SE puede controlar servidores de disco simple, grandes arrays de disco o cinta, basados en los sistemas de almacenamiento masivo (MSS).

- Workload Management (WM):
  Su propósito es el de aceptar y satisfacer las solicitudes de gestión de trabajos procedentes los clientes. Para un cálculo de trabajos, hay dos tipos principales de petición: presentación y cancelación. En particular el significado de la petición de presentación es pasar la responsabilidad del trabajo al WM. El WM luego pasará el trabajo a un "CE" apropiado para su ejecución, teniendo en cuenta las necesidades y preferencias expresadas en la descripción de trabajo. La decisión de cual recurso debe utilizarse es el resultado de un proceso de vinculación entre la presentación solicitudes y los recursos disponibles

La interacción de estos componentes es lo que va a proveer la funcionalidad del Grid.

7.5.3. FLUJO DE TRABAJO CON UN GRID COMPUTACIONAL USANDO MIDDLEWARE gLite

Cuando un usuario del Grid Computacional requiere hacer uso de el, comienza un proceso entre los servicios en el cual el trabajo pasa por unos estados en los cuales cada servicio interactúa con los otros para cumplir su función.



Dibujo 1: Sistema de Manejo de Trabajos de gLite.
Fuente: https://edms.cern.ch/file/795221//jobflow.png

1. Después de obtener un certificado digital desde una Autoridad Certificadora confiable, registrarse en la Organización Virtual (VO) y obtener una cuenta, el usuario puede iniciar sesión el el Grid autenticándose y presentar un certificado que lo autenticará de ahora en adelante frente a todos los componentes.

2. El usuario posteriormente envía un trabajo desde la interfaz de usuario (UI) al sistema de manejo de trabajos (WMS). Estos trabajos son comúnmente llamados ***Input Sandbox***, este evento es grabado posteriormente en el ***Loggin & Bookkeeping*** (LB) y el estatus del trabajo pasa a hacer ***Submitted.***

3. El sistema de manejo de trabajos (WMS) chequea por la mejor disponibilidad de los elementos de computo (CE) para ejecutar el trabajo. En este proceso interroga al ***Information Supermarket*** (ISM), un cache interno de información el cual lee la información del sistema actual desde la BDII, para determinar el estado de los recursos de cómputo y de almacenamiento, también interroga al ***File Catalogue*** para buscar la ubicación de archivos que sean necesarios. Otro evento es registrado en el ***Logging & Bookkeeping*** (LB) y el estatus del trabajo pasa a hacer ***Waiting.***

4. El sistema de manejo de trabajos (WMS) prepara el trabajo a enviar, creando un script de envoltura que será enviado conjunto a parámetros adicionales al elemento de computo seleccionado (CE). Otro evento es registrado en el ***Loggin & Bookkeeping*** (LB) y el estatus del trabajo pasa a hacer ***Ready***.

5. El elemento de computo (CE) recibe la petición y envía el trabajo para la ejecución al sistema local de manejo de recursos (LRMS), Otro evento es registrado en el ***Loggin & Bookkeeping*** (LB) y el estatus del trabajo pasa a hacer ***Scheduled***.

6. El sistema local de manejo de recursos maneja la ejecución de los trabajos en los nodos de trabajos (WN) locales, los archivos del ***Input Sandbox*** son copiados del WMS a un WN disponible donde el trabajo será ejecutado. Otro evento es registrado en el ***Loggin & Bookkeeping*** (LB) y el estatus del trabajo pasa a hacer ***Running***.

7. Mientras el trabajo corre, los archivos del Grid pueden ser vistos directamente desde un elemento de almacenamiento (SE) usando unos protocolos específicos o copiándolos en el sistema de archivos en el WN.

8. El trabajo produce nuevos archivos de salida que a su vez pueden ser colocados en el Grid y hacerlos disponibles para que otros usuarios del Grid lo puedan usar. Este proceso involucra copiarlo desde el elemento de almacenamiento (SE) y registrarlo en el ***File Catalogue***.

9. Si el trabajo termina sin errores la salida, llamada comúnmente ***Output Sandbox***, es enviada al nodo WMS y otro evento es registrado en el ***Loggin & Bookkeeping*** (LB) y el estatus del trabajo pasa a hacer ***Done***.

10. En este punto el usuario del Grid puede disponer la salida de su trabajo a la interfaz de usuario (UI) y otro evento es registrado en el ***Loggin & Bookkeeping*** (LB) y el estatus del trabajo pasa a hacer ***Cleared***.

11. Consultas para el estado del trabajo pueden ser direccionadas al LB desde la UI. También es posible desde la UI consultar la BDII para el estado de los recursos.

12. Si el sitio en el cual el trabajo es enviado no está disponible para aceptar o correrlo, el trabajo puede ser automáticamente re-enviado a otro CE que satisfaga los requerimientos del usuario. Después del máximo números permitidos de re-envíos es usado, el trabajo es marcado como ***aborted*** y el usuario puede obtener información de la historia del trabajo consultando el servicio de LB.

8. IMPLEMENTACIÓN DEL GRID COMPUTACIONAL UIS

Para la implementación de un Grid Computacional experimental en la Universidad Industrial de Santander fueron asignados recursos ubicados en la sala de supercomputación del Centro de Tecnologías de la Información y Comunicación, estos recursos consisten en 5 computadores con las siguientes características:

Procesador	: **Intel(R) Pentium(R) 4 CPU 3.20GHz**
Extensiones de VT en Procesador	: **No**
Memoria	: **1024 Mbytes**
Disco Duro	: **160 GB**
Caché	: **2048 KB**

Adicionalmente se asignaron las conexiones de red y direcciones ip necesarias para cada uno de los equipos.

8.1. RECURSOS PARA EL PROYECTO

Como se puede apreciar en la descripción de la arquitectura del middleware gLite [25], este está conformado por una cantidad de nodos que sobrepasan los equipos de cómputo asignados, para superar éste problema se realizó una búsqueda de soluciones que permitieran ejecutar varios sistemas operativos de manera simultánea e independiente en una misma máquina física, encontrando diferentes tipos de técnicas de virtualización que permiten hacer esto y posteriormente se realizó un estudio comparativo entre las técnicas que mas se ajustaban a las necesidades siguiendo la siguiente metodología:

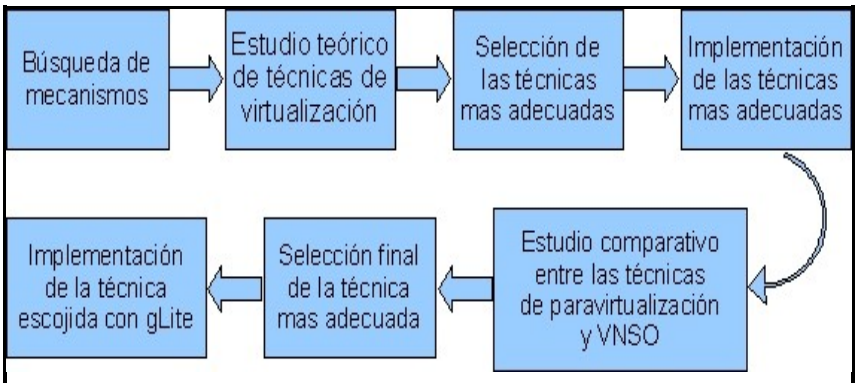


Figura 4: Estructura Virtualización Grid Computacional UIS. Fuente:Autor.

25 7.5 Descripción de la arquitectura del middleware gLite

En la primera etapa se realizó una búsqueda de los mecanismos que permiten ejecutar varios sistemas operativos de manera simultánea e independiente en un mismo computador, en ésta etapa se concluyó que la virtualización era la tecnología que ofrecía mediante varias técnicas ésta posibilidad. Una vez realizado esto, se procedió en la segunda etapa a estudiar teóricamente mediante la literatura las técnicas mas destacadas de virtualización, estas fueron:

- Emulación
- Virtualización Nativa
- Paravirtualización
- Virtualización a Nivel de Sistema Operativo (VNSO)

8.1.1. TIPOS DE VIRTUALIZACIÓN ESTUDIADAS

8.1.1.1. EMULACIÓN

Esta técnica consiste en crear una máquina virtual en el sistema anfitrión que emula el hardware que se desee, incluyendo el procesador y periféricos, esto permite hacer creer al sistema operativo invitado que está ejecutándose en determinada arquitectura. La emulación incluso permite que un sistema operativo diseñado para alguna plataforma específica pueda correr sobre otra con la cuál no es compatible técnicamente sin ser modificado gracias a que es posible emular distintas arquitecturas (x86, x86_64, ARM, SPARC, PowerPC, MIPS, etc).

Dicha emulación se fundamenta en la traducción dinámica de instrucciones de la arquitectura emulada a la que realmente se tiene, lo que ocasiona que sea bastante lenta en comparación con otras técnicas, por lo que generalmente es usada para probar y desarrollar software sobre diferentes arquitecturas que no siempre se tienen.

La estructura se puede apreciar en la figura 5:

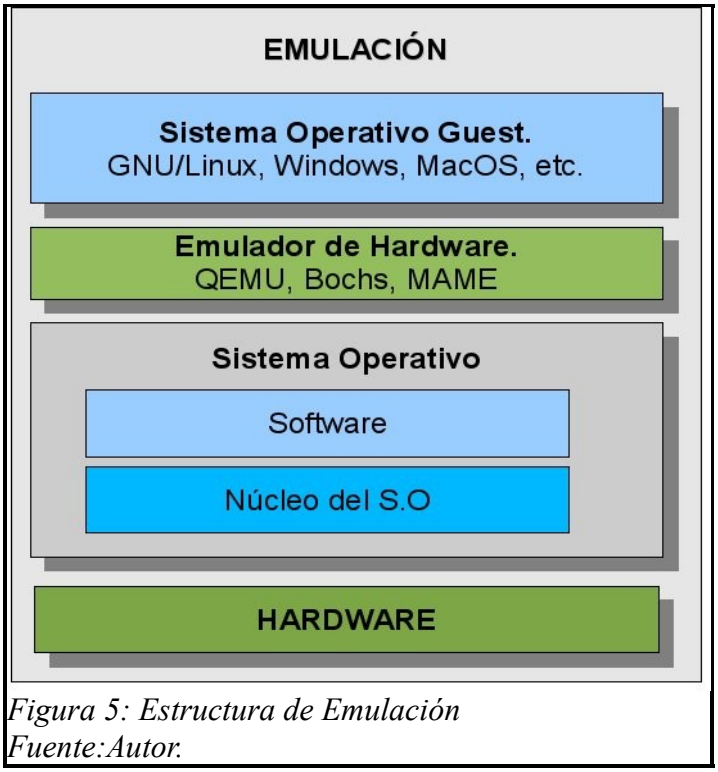


Figura 5: Estructura de Emulación
Fuente:Autor.

8.1.1.2. VIRTUALIZACIÓN NATIVA / COMPLETA

Esta técnica es similar a la emulación y consiste en que el sistema operativo anfitrión no emula todo pero si lo suficiente del hardware para que los sistemas operativos invitados puedan ser ejecutados de forma nativa, es decir, sin modificaciones en su kernel, permitiendo la ejecución de varios sistema aislados.

Los sistemas invitados deben estar diseñados para la misma arquitectura de CPU que el sistema anfitrión, ya que esta no se emula, esto hace que no sea necesaria la traducción de instrucciones que se realiza en la emulación ya que la arquitectura de ambos sistemas operativos es la misma dando mayor velocidad de ejecución.

Esta técnica se utiliza generalmente para realizar pruebas de software en diferentes sistemas operativos pero sobre una misma arquitectura de procesador, también para poder utilizar herramientas que en el sistema anfitrión no sean compatibles. Esta técnica aun consume altos ciclos de procesador y tiempo de ejecución para la virtualización, dado que sigue emulando dispositivos.

La estructura de la técnica de virtualización Nativa/Completa se aprecia en la figura 6:

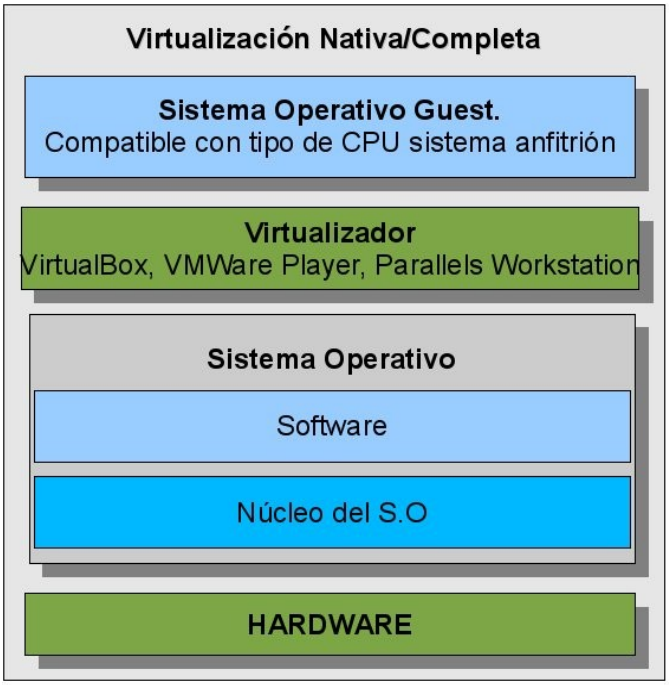


Figura 6: Estructura Virtualización Nativa/Completa
Fuente:Autor.

8.1.1.3. VIRTUALIZACIÓN A NIVEL DE S.O

Permite que una máquina física ejecute diferentes instancias de sistemas operativos aislados entre si. Cada uno de los sistemas invitados se conoce como VPS por sus siglas en ingles (Virtual Private Server) o VE (Virtual Environment). Cada VPS es una entidad separada que desde la perspectiva de sus usuarios se ve como un servidor físico real que tiene sus propios archivos de configuración, usuarios, procesos, librerías de sistema y aplicaciones entre otros.
Se caracteriza por que los sistemas invitados comparten el kernel del sistema anfitrión, y este kernel modificado para soportar esto es quien determina para quien trabaja en determinado momento, logrando gran rapidez de ejecución, ya que los sistemas invitados se comunican directamente con este y aprovechan los procesos comunes. Esto se traduce en mejor rendimiento y aprovechamiento de los recursos, menor penalización y alta escalabilidad.
De igual forma permite una fácil administración ya que hasta ahora los sistemas

operativos invitados deben ser del mismo tipo, evitando incompatibilidades y permitiendo la administración masiva de VPS's.

La estructura de la técnica de virtualización a Nivel de S.O se encuentra en la figura 7:

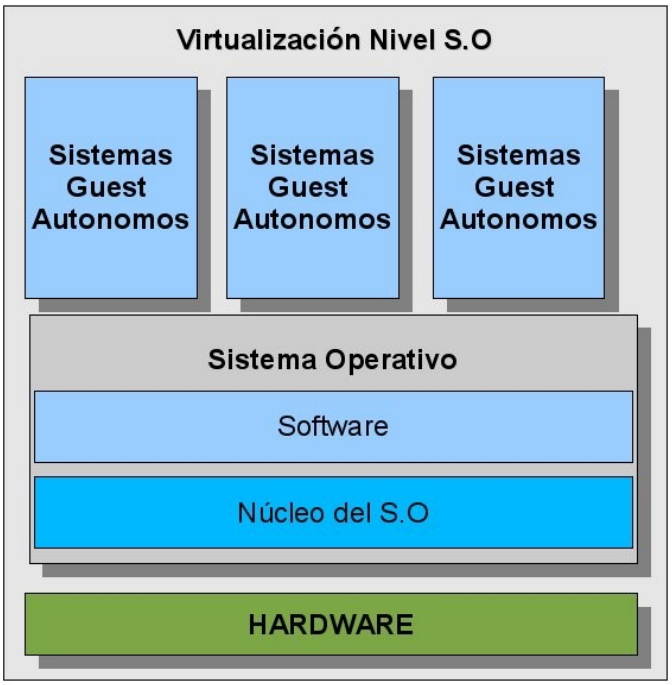


Figura 7: Estructura Virtualización a Nivel de S.O
Fuente:Autor.

8.1.1.4. PARAVIRTUALIZACIÓN

Esta técnica se basa en la arquitectura de niveles de los actuales procesadores que consiste en que cada procesador tiene 4 niveles de privilegios representados por 4 anillos enumerados del 0 al 3.

El anillo 0 es quien permite mayores privilegios de ejecución, permitiendo acceso sin restricciones al hardware, es en este anillo donde se ejecuta el kernel o núcleo del sistema operativo y normalmente los controladores de dispositivos.

Los anillos 1 y 2, tienen menos privilegios que el 0 y en estos se ejecutan servicios básicos del sistema, los cuales pueden ser llamados por las aplicaciones, por ejemplo protocolos de red y gestores gráficos, esto permite que los servicios

tengan acceso a los datos de las aplicaciones pero a su vez estén protegidos de estas, ya que las aplicaciones se ejecutan en el anillo 3.

El anillo 3 es el de menor privilegio, y en este se ejecutan las aplicaciones, manteniendo así un esquema de seguridad para mantener la integridad de los recursos.

La siguiente figura explica esta estructura:

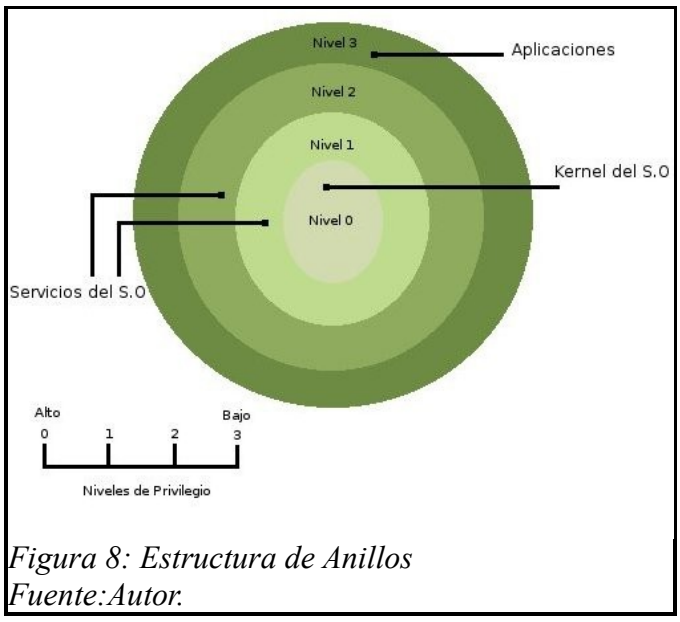


Figura 8: Estructura de Anillos
Fuente:Autor.

La técnica de paravirtualización consiste en la ubicación de un Hypervisor en el anillo 0, este Hypervisor es un software que tiene la funcionalidad de comunicarse directamente con el hardware, el kernel del sistema anfitrión pasa a ejecutarse en un nivel superior, dejando el anillo 0 asociado al hypervisor.

Para poder realizar esto, se deben modificar tanto el kernel del sistema operativo anfitrión, como el de los sistemas operativos a virtualizar, esto podría parecer una limitante para poder paravirtualizar sistemas operativos no modificables como es el caso de los S.O's privativos, sin embargo es posible virtualizar este tipo de sistemas por medio de la paravirtualización haciendo uso de las tecnologías de virtualización VT-X de Intel y Pacifica de AMD presentes en los procesadores modernos, las cuales permiten ejecutar sistemas operativos en el nivel 0 sin tener que modificarlos, esto gracias a un nivel de privilegio especial en el anillo 0 para el hypervisor llamado root-mode también conocido como “nivel -1”, los demás componentes se ejecutan en otro nivel llamado non-root-mode.

De esta manera quedan redistribuidos los componentes del sistema, en el anillo 0

se encuentra el hypervisor y en los anillos superiores se encuentran tanto el S.O anfitrión como de los sistemas operativos invitados y sus kernel que se comunican con el Hypervisor y este a su vez con el hardware.

La estructura de la técnica de Paravirtualización se detalla en la siguiente figura:

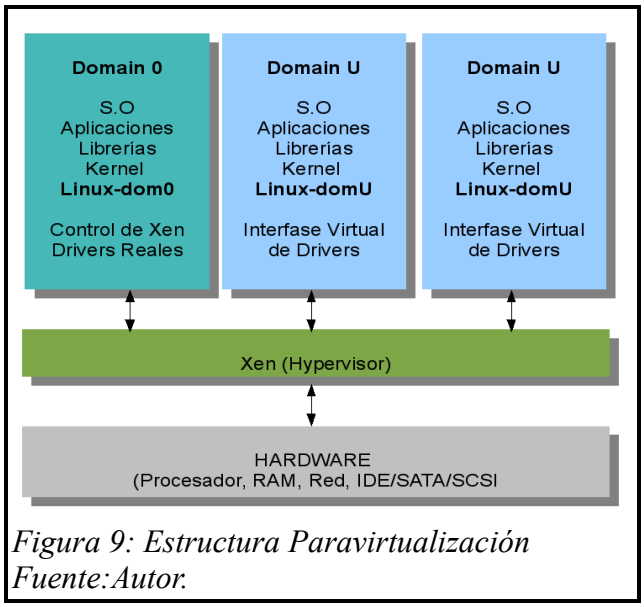


Figura 9: Estructura Paravirtualización
Fuente:Autor.

Entre las ventajas de esta técnica se encuentran:

- Facilidad para la migración en caliente sin perdida de desempeño en las máquinas que están siendo virtualizadas.
- Heterogeneidad en los S.O's a virtualizar ya que los sistemas operativos invitados no tienen que compartir un mismo kernel.
- Gran rapidez de ejecución al no tener que realizar emulación alguna.
- Portabilidad, ya que las imágenes de los S.O invitados se pueden extraer con facilidad y es posible convertir imágenes de sistemas operativos virtualizados con otras técnicas en imágenes de paravirtualización.
- Escalabilidad, debido a que inicializar una imagen puede hacerse con mucha facilidad y son totalmente aisladas de tal forma que no se afectan en caso de problemas.

En esta segunda etapa se tuvo una comprensión de los diferentes enfoques y ventajas de cada una de las técnicas mencionadas.

Posteriormente en la etapa 3 se seleccionaron las técnicas de paravirtualización y virtualización a nivel de sistema operativo debido a las características de baja pérdida de desempeño para realizar un estudio mas completo y técnico con el objetivo de establecer cual era la mas adecuada para las necesidades del proyecto.

En la etapa 4 se implementaron estas dos técnicas para proceder a realizar el estudio comparativo.

En la quinta etapa se procedió a realizar el estudio comparativo entre las dos técnicas escogidas en la etapa 3 realizando pruebas de disco, memoria, uso de CPU y transmisión de datos por red. La metodología, implementación y resultados de este estudio se pueden ver en Nobsa Julián, Rodriguez Iván y Díaz Gilberto. Estudio Comparativo de las Técnicas de Paravirtualización y Virtualización a nivel de Sistema Operativo. Bucaramanga: 2008.

Luego de realizar el estudio, en la sexta etapa se seleccionó la técnica de paravirtualización para aplicarla en el proyecto y resolver el problema de limitación de recursos debido al buen rendimiento mostrado durante el estudio y a las características mencionadas en el parágrafo 8.1.1.4[26].

Finalmente se realizó la implementación de la técnica de paravirtualización a través del software Xen [27], con la instalación de los nodos fundamentales del middleware gLite montando toda la infraestructura paravirtualizada.

26 8.1.1.4 Paravirtualización

27 Herramienta software que implementa la paravirtualizacion a través de un hypervisor, www.xen.org

El esquema de distribución de los nodos en los recursos de cómputo se puede observar en la siguiente figura:

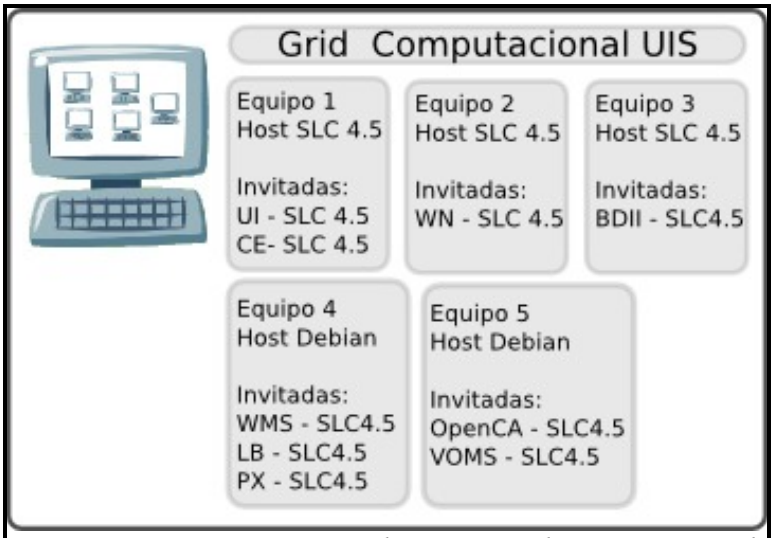


Figura 10: Estructura Virtualización Grid Computacional UIS.
Fuente:Autor.

Los notación de los nodos alojados en cada máquina corresponde a:

- SLC : Scientific Linux CERN
- UI : User Interface
- CE : Computing Element
- BDII : Berkeley Database InformatonIndex
- WMS : Workload Management System
- LB : Logging and Bookkeeping
- PX : My Proxy
- OpenCA : Open Certification Authority
- VOMS : Virtual Organization Membership Service

De esta manera el grid computacional UIS, consta de 5 equipos en los cuales se encuentran paravirtualizados los nodos fundamentales para lograr el funcionamiento del Grid. Estos nodos tienen el sistema operativo Scientific Linux Cern 4.5 y el sistema base se encuentra con el sistema operativo Scientific Linux Cern 4.5 y Debian 4.0, aprovechando al máximo los recursos ejecutando de manera paravirtualizada múltiples sistemas operativos que facilitan las labores de instalación, configuración y pruebas de los servicios y componentes de glite.

8.2. SISTEMA OPERATIVO ANFITRIÓN

En el proceso de instalación de una distribución GNU/Linux que permita realizar trabajos de paravirtualización, se decide utilizar Scientific Linux Cern 4.4, distribución que se puede obtener del sitio público del CERN (European Organization for Nuclear Research).

Esta distribución es una versión adaptada de Scientific Linux y está a su vez de RHEL (RedHat Enterprise Linux), distribución amparada por la empresa RedHat que ha invertido en el desarrollo de Xen permitiendo la integración a nivel del kernel como máquina dom0, adicionalmente la distribución Scientific Linux Cern 4.4 tiene compatibilidad con el middleware gLite al ser la distribución usada por el CERN en el proceso de pruebas y verificación de los componentes del middleware.

Paso inicial: Boot.
El proceso de instalación comienza con las opciones de arranque, del DVD SLC4.4, se escoge las opciones por omisión y se comienza la instalación, después de realizar una revisión del estado del DVD arranca el sistema gráfico y se procede con la instalación.

Posteriormente se visualiza **Pantalla de Bienvenida** con explicación acerca de la distribución SLC4.4 y las opciones durante el proceso de instalación.

Escoger el tipo de instalación.
En esta sección pregunta que tipo de instalación se quiere hacer, para lo cual se debe realizar una configuración personalizada a manera de ajustar la instalación a los parámetros que se desea.

Particionamiento del Disco Duro.
La partición del Disco Duro, para los propósitos del proyecto fue de tamaño fijo de 160 GB, se realizara en 3 particiones primarias organizadas de la siguiente forma:

Punto de Montaje	**Capacidad**	**Tipo**
/	6 GB	ext3
-	2 GB	swap
LVM(Physical Volume)	158GB	LVM

El uso de LVM se emplea con el objetivo de obtener facilidad en torno al re-dimensionado de los volúmenes lógicos y copias de respaldo en imágenes instantáneas de los sistemas que van a ser paravirtualizados.

En el proceso de instalación, se crea un Grupo de Volúmenes llamado *vg* con el

espacio disponible restante, 158 GB en total, y se ubica un volumen lógico llamado *home* de 4 GB de tamaño, el cual va alojar las cuentas de usuario del sistema base a correr inicialmente.

El resto de espacio se deja sin asignar para posteriormente crear volúmenes lógicos que permitirán la paravirtualización de los componentes y servicios del middleware gLite.

Configuración de la Red
En la configuración de red se especifica la IP Clase C correspondiente a la red interna asignada para la sala de Super-Computación del CENTIC, IP's en el rango de 192.168.109.141 hasta 192.168.109.159, además de introducir la puerta de enlace de la red y el DNS.
En el siguiente paso es necesario deshabilitar el firewall para realizar una configuración manual con base a los requisitos de conectividad necesarios posterior a la instalación.

Selección de paquetes a instalar
En la selección de paquetes se usan la menor cantidad de paquetes y se configura la hora y fecha de acuerdo a la posición geográfica de las máquinas.

Después de establecer la **Contraseña root** procede la **Copia de Archivos**.

En la **finalización de instalación** el sistema de instalación reinicia el equipo y comienza la configuración básica.

8.3. IMPLEMENTACIÓN DE LA PARAVIRTUALIZACIÓN

8.3.1. CONSIDERACIONES INICIALES

Para el montaje y configuración de las máquinas paravirtualizadas es necesario primero configurar el arranque del computador permitiendo que el Hypervisor se ubique en el modo privilegiado (anillo 0) y posteriormente el sistema operativo que va acompañado del hypervisor, llamado de otra forma el dom0.

Con el fin de iniciar el hypervisor en el anillo 0 es necesario modificar el sistema operativo GNU/Linux Scientific Linux Cern 4.4 instalándole un kernel modificado que permita ejecutarse en un anillo superior diferente del anillo 0 otra forma es obteniendo una imagen modificada del sistema operativo que incluya dicho kernel instalado a partir del arranque del dom0 los S.O paravirtualizados serán dom1,

dom2 ó mejor conocidos como domU.

Scientific Linux Cern 4.4 distribución construida en base a Scientific Linux la cual a su vez está basada en RedHat Enterprise Linux 4, incluye paquetes del CERN el cual desarrolla gLite, tiene gran soporte en la comunidad que hace uso de ella y tiene una gama amplia de paquetes precompilados, entre ellos los paquetes necesarios para iniciar el S.O como dom0.

El proceso de instalación y configuración de Xen está explicado en el anexo 12.1.

8.3.2. SISTEMAS OPERATIVOS INVITADOS DOMU

El sistema operativo invitado debe estar compuesto de una estructura de archivos y kernel que permita correr en otro nivel de ejecución, se utiliza la imagen de virtualización de Scientific Linux CERN 4 disponible en el proyecto del CERN en http://project-xen.web.cern.ch/project-xen/xen/img/slc4.gpe.img.gz

Para examinar el proceso de instalación y configuración de los sistemas domU véase el anexo 12.2.

8.4. PREREQUISITOS PARA LA INSTALACIÓN DEL MIDDLEWARE gLite

8.4.1. HORA DEL SISTEMA Y REPOSITORIOS DE PAQUETES

La hora del sistema de cada uno de los componentes y servicios del Grid Computacional deben estar en el mismo rango, lo cual se logra configurando en cada máquina que la hora del sistema se actualice por la hora de servidores en internet.

El protocolo encargado de realizar esta acción es el NTP (Network Time Protocol)[28] y su instalación y configuración se puede ver en el anexo 12.3.

Para instalar diferentes paquetes adicionales antes del hacerlo con los componentes y servicios del middleware gLite, es necesario configurar los repositorios de los manejadores de paquetes utilizados (APT, YUM).

APT

```
[root@vscientific02 ~]# rpm -ivh
http://glitesoft.cern.ch/scientific/30rolling/i386/apt-
```

28 *Protocolo de internet para sincronizar los relojes de los sistemas informáticos a través de ruteo de paquetes en redes con latencia variable. NTP utiliza UDP como su capa de transporte, usando el puerto 123. Está diseñado para resistir los efectos de la latencia variable.*

```
rpm/RPMS.os/apt-0.5.15cnc6-4.SL.i386.rpm
[root@vscientific02 ~]# nano /etc/apt/sources.list.d/glite-rhel30.list
```

Se descarga el rpm que instalará el manejador de paquetes APT y después se configura los repositorios creando el archivo *glite-rhel30.list* con el contenido:

Archivo: **glite-rhel30.list**

```
rpm http://glitesoft.cern.ch/EGEE/gLite/APT/R3.0/ sl4-compat externals
Release3.0 updates
rpm http://www.dutchgrid.nl/mirror/apt/centos 3/i386 os updates
rpm http://www.dutchgrid.nl/mirror/jpackage 1.7/generic free non-free
rpm http://www.dutchgrid.nl/mirror/jpackage 1.7/redhat-el-4.0 free non-
free
rpm http://linuxsoft.cern.ch  cern/slc4X/i386/apt os updates extras
```

después de guardar el archivo se realiza la operación **apt-get update** e inmediatamente se obtiene acceso a los nuevos archivos alojados en el repositorio.

YUM

Para el caso de YUM es necesario ubicar los archivos ubicados en http://grid-deployment.web.cern.ch/grid-deployment/yaim/repos/ ,dependiendo del componente ó servicio a instalar en el directorio /etc/yum.repos.d/ y realizar un yum update para actualizar la disposición de los archivos.

Adicionalmente se deben instalar dos paquetes importantes, el JDK y el Jpackage necesarios para soportar los componentes y scripts en java. Para ver el proceso de instalación de estos dos paquetes véase el anexo 12.4.

8.5. INFRAESTRUCTURA DE SEGURIDAD EN EL GRID

Las comunidades pertenecientes al Grid Computacional están agrupadas en organizaciones virtuales(***VO***). Deben existir reglas y condiciones de uso para cada usuario perteneciente a una organización virtual reconocida por el Grid.

El *Grid Security Infrastructure* (***GSI***) [Infraestructura de seguridad en Grid] habilita una autenticación y comunicación segura a través de la red basada en encriptación de llave publica con certificados X.509 y el protocolo SSL (***Secure Socket Ley***).

Para que un usuario logre autenticarse en el Grid y pueda hacer uso de los

recursos que posee, es necesario que tener un certificado digital X.509 generado por una *Certification Authority* (***CA***) de confianza al Grid Computacional y además estar registrado en una VO asociada al Grid. Los servicios y componentes que hacen parte de la infraestructura del middleware gLite también necesitan de certificados que permita tener una comunicación segura y de confianza entre ellos.

8.5.1. INFRAESTRUCTURA DE LLAVE PÚBLICA.

La infraestructura de llave pública (PKI) es una combinación de hardware, software, políticas y procedimientos de seguridad que permiten la ejecución de aplicaciones con garantías de operaciones criptográficas como el cifrado, la firma digital o el no repudio de transacciones electrónicas.[29]

8.5.1.1. LLAVES PÚBLICAS Y PRIVADAS

Cada usuario tiene dos llaves una privada y una pública. Estas llaves están relacionadas por transformaciones lineales haciendo improbable deducir la llave privada a partir de la llave pública.

La información se encripta con la llave pública y se desencripta únicamente con la llave privada. De esta forma no se compromete la seguridad en el intercambio de llave para desencriptar el contenido.

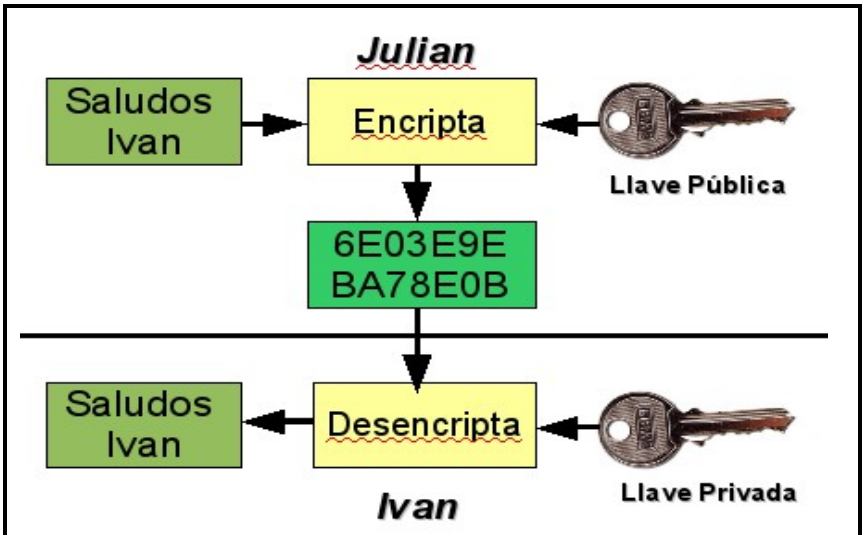


Figura 11: Encriptación de Mensaje
Fuente: Autor

29 http://es.wikipedia.org/wiki/Certificaci%C3%B3n_Electr%C3%B3nica

En la figura 11, *Encriptación de Mensaje* se muestra como ***Julian*** envía un mensaje encriptado usando la llave pública de ***Ivan*** y este a su vez desencripta con la llave privada. Para ***Julian*** es importante saber si la llave pública que el tiene es efectivamente la que ha generado ***Ivan***.

8.5.1.2. CERTIFICADOS DIGITALES: DEFINICIÓN Y ESTRUCTURA

Los certificados digitales, es un archivo adjunto el cual permite verificar la autenticidad del mensaje que ha sido enviado. El certificado tiene una estructura la cual se explica en la siguiente figura:

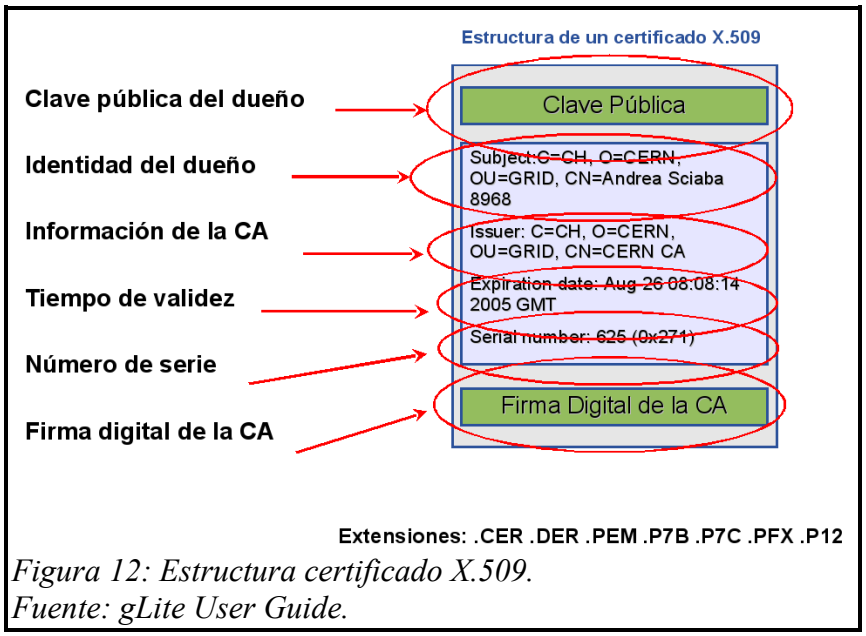


Figura 12: Estructura certificado X.509.
Fuente: gLite User Guide.

Los certificados son documentos digitales mediante los cuales la Autoridad Certificadora garantiza la relación entre la identidad de un sujeto o una entidad y su clave pública, legitimando ante terceros que confían en sus certificados dicha relación.

8.5.1.3. AUTORIDAD DE CERTIFICACIÓN

Existe una entidad llamada Autoridad de Certificación (***CA***) la cual es de confianza

tanto para quien emite como quien recibe la comunicación, es la encargada de emitir, revocar y administrar los certificados digitales.

El proceso de obtención de un certificado digital se realiza de acuerdo a la figura 13, *Proceso certificado autoridad de certificación*. El usuario envía una solicitud a la Autoridad de Registro (***RA***) con un las llaves pública y privada. Los datos son verificados por la Autoridad de Registro y posteriormente son dirigidos con aprobación a la Autoridad de Certificación para su firma. La Autoridad Certificadora firma con su llave pública el certificado estableciendo los parámetros descritos en la sección anterior ***CERTIFICADOS Y ESTRUCTURA.***

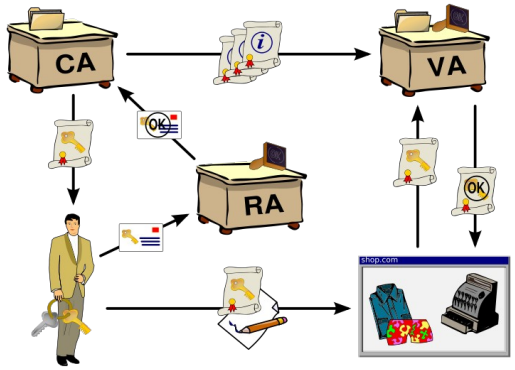


Figura 13: Proceso certificado autoridad de certificación.
Fuente: Wikipedia.

Cuando el usuario presenta el certificado a otro ente, este realiza una consulta a la Autoridad de Certificación que emitió el certificado para realizar la comprobación de identidad del usuario.

Para implementar una autoridad de certificación y de registro existen diferentes paquetes de software, entre los más conocidos existe OpenCA software que se ha escogido en el desarrollo del proyecto debido a las interfaces presentes para consulta, generación y petición de certificados.

8.5.2. OpenCA

Openca es una herramienta desarrollada con el fin de tener la implementación de una Infraestructura de Llave Pública(***PKI***) propia. Nace de un proyecto colaborativo Open Source y tiene como objetivo ofrecer una implementación robusta de Autoridad de Certificación implementando los protocolos mas usados con toda la potencia de la criptografía en la web.

El desarrollo de OpenCA está dividido en dos lineas principales: estudiar y refinar el esquema de seguridad para garantizar el mejor modelo para ser usado en una autoridad de certificación y desarrollar software para instalar y administrar fácilmente una autoridad de certificación.

8.5.2.1. PREREQUISITOS E INSTALACIÓN DE OpenCA

Para la instalación y configuración de OpenCA se debe tener previamente un S.O instalado, en este caso Debian GNU/Linux, además los paquetes básicos de las herramientas necesarias para el funcionamiento de OpenCA, esta son:

- Apache2 http://httpd.apache.org/download.cgi
- Perl http://www.perl.com/
- OpenSSL http://www.openssl.org/source/
- MySQL http://dev.mysql.com/downloads/mysql/5.0.html#source

La instalación de estas herramientas se muestra en el anexo 12.5 y la de la autoridad certificadora OpenCA en el anexo 12.6.

El mecanismo de trabajo usado por OpenCA en la mayoría de CA con certificados digitales firmados por otras CA de confianza exige que dos nodos (CA y RA) sean usados de manera separada. El nodo encargado de firmar (CA) debe estar aislado de internet y la única forma de importar solicitudes de firma de certificados es por medio de un archivo de solicitudes exportado previamente por el nodo con el software de RA instalado. El caso del Grid en el presente trabajo se utiliza la combinación de ambas configuraciones en la máquina encargada de OpenCA.

Después de realizar los pasos de instalación, configuración y puesta en marcha, OpenCA ha generado una llave secreta y auto firmado digitalmente un certificado. Si se desea se puede crear un certificado digital para el administrador de la CA y de la RA a manera de desacoplar el trabajo de cada uno como se muestra en la Figura 13, *Certificado inicial.*

Figura 14: Certificado inicial
Fuente: Autor.

8.5.3. VIRTUAL ORGANISATION MEMBERSHIP SERVICE (VOMS)

El servidor VOMS actúa como un repositorio central para la información de autorización del usuario, proporcionando soporte para el ordenamiento de usuarios dentro de grupos, control de roles, privilegios, entre otros, pertenecientes a una organización virtual (VO). El VOMS Admin Service, es una aplicación web que provee las herramientas para administrar las base de datos de miembros de una VO asociada a la VOMS . El VOMS puede ser configurado con ORACLE ó MySQL. Para el desarrollo del proyecto se ha usado MySQL.

En el proceso de autorización se crea un certificado proxy, el cual es firmado con las llaves del usuario y contiene una llave pública, privada y un certificado temporal. Datos generados en el proceso de creación del certificado proxy y solo validos durante el tiempo de vida del certificado proxy. Posteriormente existe un proceso llamado delegación el cual, consiste en almacenar en un servidor llamado MyProxy, el certificado proxy para que los demás nodos puedan usarlo.

El uso del servidor VOMS proporciona una estructura basada en grupos y roles

que permite organizar de una manera eficiente los miembros de una organización virtuales y los permisos sobre los recursos, en la figura 15, *Multiples VO'S* se distingue varias VO'S a nivel mundial y el acceso a recursos específicos en un Grid interconectado con varias organizaciones en diferentes ubicaciones geográficas.

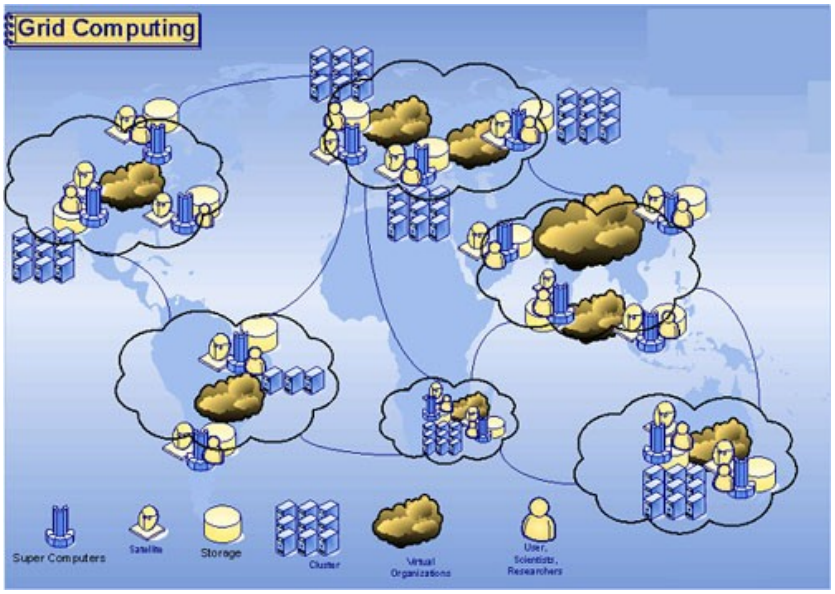


Figura 15: Múltiples VO'S.
Fuente: gLite User Guide.

8.5.4. INSTALACIÓN Y CONFIGURACIÓN DE gLite-VOMS

En la instalación del servidor VOMS se siguen los siguientes pasos:

1. Configurar la base de datos (MySQL ó Oracle).
2. Definir parámetros en los archivos de configuración para el servidor VOMS y para las VO pertenecientes al Grid Computacional.
3. Configurar el servidor VOMS.
4. Iniciar el servidor VOMS

Para la instalación de MySQL se utiliza YUM.

```
[root@glite-VOMS ~]# yum install glite-VOMS_mysql
```

Se asigna el password para mysql.

```
[root@glite-VOMS ~]# /usr/bin/mysqladmin –u root password 'yourPassword'
```

La instalación de gLite-VOMS se realiza mediante el metapaquete glite-VOMS_mysql, el repositorio debe estar preparado con anterioridad como se menciona en *"PREREQUISITOS PARA EL MIDDLEWARE gLite"*.

```
[root@glite-VOMS ~]# yum install glite-VOMS_mysql
```

Los scripts y archivos de configuración de VOMS y de VOMS Admin se encuentran en la ubicaciones:

$GLITE_LOCATION = /opt/glite/etc/config/scripts/glite-voms-server-config.py.

Las plantillas de configuración se encuentran en los directorios:

$GLITE_LOCATION/etc/config/templates/glite-voms-server.cfg.xml
$GLITE_LOCATION/etc/config/templates/glite-global.cfg.xml
$GLITE_LOCATION/etc/config/templates/glite-security-utils.cfg.xml
$GLITE_LOCATION/etc/config/templates/vo-list.cfg.xml

Se modifican las plantillas de configuración y se ubican en el directorio /opt/glite/etc/config *(Ver Anexo 12.8 y 12.9)*.

Después de asignar los parámetros de configuración se revisa la configuración con el script *./glite-voms-server-config.py -c ,* la salida a este comando se muestra en el anexo 12.10.

Posteriormente see realiza la configuración con ./glite-voms-server-config.py --configure. Terminada la configuración el script agrega en la base de datos los datos correspondientes a las organizaciones virtuales y el administrador por omisión de la misma, previamente establecido en los archivos de configuración, junto al certificado emitido por la CA.

Finalizado el proceso de instalación se inicia el servidor VOMS con el script ./glite-voms-server-config.py --start y el script iniciara el VOMS-Admin que es una aplicación web corriendo sobre tomcat permitiendo administración de las VO.

Es necesario obtener el certificado en el formato PKCS#12[30] del administrador del VOMS proveniente de la CA para descargarlo y ubicarlo en el navegador y así

30 Estándar de sintaxis de intercambio de información personal, formato de fichero usado comúnmente para almacenar claves privadas con su certificado de clave pública protegido mediante clave simétrica.

poder ingresar al servidor VOMS y realizar los trabajos de Administración ó de registro de usuario a una VO.

8.5.5. TRABAJANDO CON EL VOMS

Después de realizar la instalación y configuración se procede a agregar el primer usuario a una VO, en este caso se utiliza la VO UIS previamente definida en el proceso de instalación y se crea un certificado nuevo firmado digitalmente por la CA para agregarlo a la VO UIS.

Como se ve en la figura 16 el usuario carga el certificado digital en el navegador para proceder al registro de en la VO que desee. Para ver la lista de las VO disponibles en el servidor VOMS debe ingresar en el navegador la dirección: https://ipdevoms:8443/vomses/

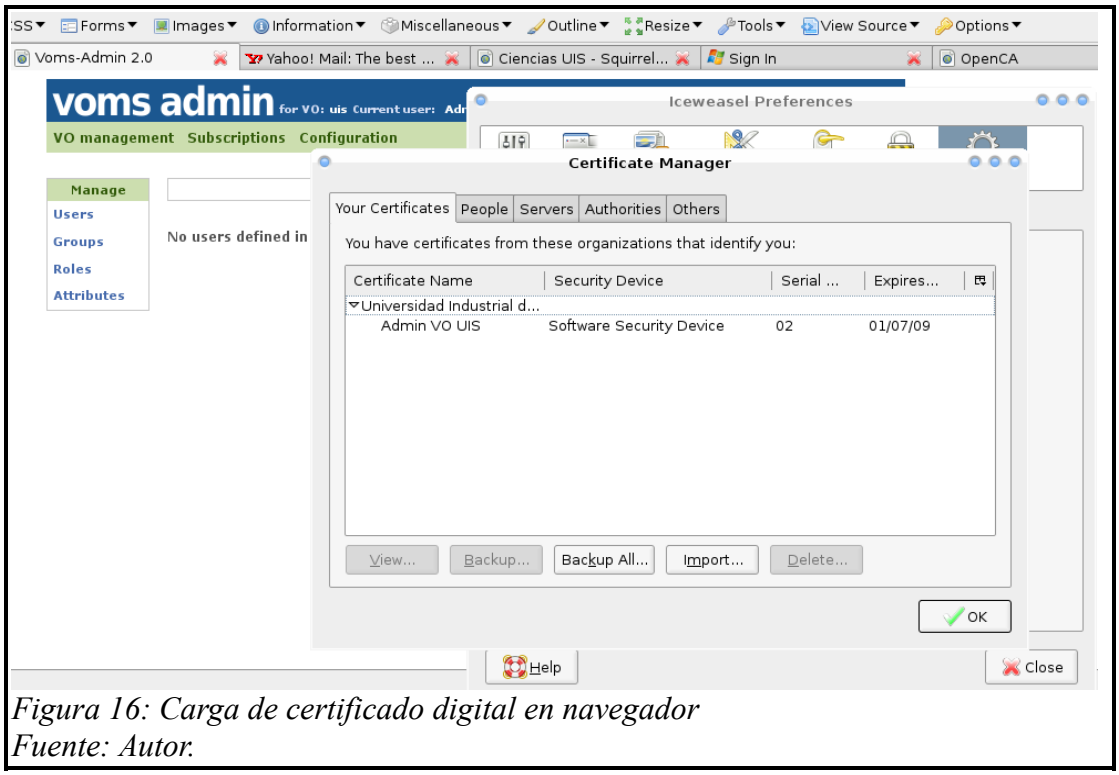


Figura 16: Carga de certificado digital en navegador
Fuente: Autor.

Se elige la VO a registrarse, se introduce en el formulario la información correspondiente e inmediatamente los datos son enviados al administrador de la VO elegida para verificar la veracidad de los datos y proceder el registro a la VO solicitada.

Para casos no experimentales y Grid Computacionales en producción la obtención de certificados digitales y el registro en organizaciones virtuales pertenecientes a servidores VOMS requiere de un procedimiento largo en el cual se establecen citas personales, entrevistas y solicitudes formales.

8.6. USER INTERFACE (UI)

El acceso al Grid Computacional se realiza mediante la User Interface (UI). En esta máquina están alojadas cuentas reales de los usuarios pertenecientes a las VO usadas por el Grid Computacional. El usuario después de haberse registrado en una VO por medio del servidor VOMS debe solicitar una cuenta de usuario al administrador del sitio que provea la UI para posteriormente acceder a ella y ubicar los certificados de usuario con extensión PEM[31] dentro del directorio personal en la carpeta .globus. La UI provee de herramientas CLI (Command Line Interface) para realizar operaciones básicas de Grid como son:

- Listado de los recursos disponibles para ejecutar un trabajo.
- Enviar trabajos para la ejecución.
- Cancelar trabajos.
- Obtener la salida de los trabajos finalizados.
- Mostrar el estado de los trabajos enviados.
- Obtener la información del servidor *Logging and Bookkeeping* correspondiente a los trabajos.
- Copiar, replicar y borrar archivos del Grid Computacional.
- Obtener el estatus de los diferentes recursos disponibles por medio del servidor Information System.

8.6.1. INSTALACIÓN Y CONFIGURACIÓN UI

La instalación de la UI se realiza por medio de el sistema de paquetes yum y yaim.

Inicialmente se instala el metapaquete glite-UI

```
[root@glite-ui-grid ~]# yum install glite-UI
```

Este paquete instala toda los archivos necesarios para la configuración del UI,

31 (Privacy Enhanced Mail) Base64 encoded DER certificate

adicionalmente instala el sistema de paquetes YAIM que se ejecuta al iniciar la configuración del nodo UI. Antes de ejecutar el comando encargado de la configuración del gLite User Interface, es necesario ubicar los ficheros correspondientes a las autoridades certificadoras en el directorio /*etc/grid-security/certificates/* para el caso se ha creado un paquete llamado *uis-openca.tgz* el cual contiene:

b541feed.0 → Certificado de la CA UIS.
b541feed.crl_url → Dirección URL de la lista de revocación de certificados.
b541feed.r0 → Certificado de la lista de revocación de certificados.
b541feed.signing_policy → Política de firma para los certificados emitidos por la CA.

Estos archivos se obtienen del sistema web de la OpenCA y contienen la información definida anteriormente, Adicionalmente debe ubicarse en el directorio */etc/grid-security/vomsdir* los certificados correspondientes a los servidores VOMS usados por los usuarios que hacen parte de las VO.

Posteriormente debe crearse y editar un archivo site-info-UI.def el cual contiene los parámetros suficientes para que YAIM configure el nodo UI *(Ver Anexo 12.11)*.

Luego se ejecuta el comando yaim para configurar el nodo UI.

```
[root@glite-ui-grid ~]# /opt/glite/yaim/bin/yaim -s /root/site-info-
UI.def -n glite-UI -c
```

8.6.2. TRABAJANDO CON LA UI

Luego de instalar y configurar la UI el usuario se loguea mediante el servicio ssh en la máquina asociada al User Interface y ubica los certificados de usuario firmados digitalmente y agregados a una VO en el directorio .globus del usuario personal con permisos chmod 444 para usercert.pem y 400 para userkey.pem.

```
[uis001@ui-grid ~]$ ls -l .globus/
total 8
-r--r--r--  1 uis001 uis001 2283 Jul  9 16:58 usercert.pem
-r--------  1 uis001 uis001 1015 Jul  9 16:58 userkey.pem
```

Después de ubicar los certificados el usuario tiene derecho a crear un certificado proxy el cual junto a la llave privada del certificado proxy y el certificado de la OpenCA serán la carta de presentación ante los diferentes nodos que validaran ante el servidor VOMS esta información.

Para la creación del certificado se ejecuta el comando :

```
[uis001@ui-grid ~]$ voms-proxy-init --voms uis
Cannot find file or dir: /home/uis001/.glite/vomses
Your identity: /C=CO/O=UIS/OU=Internet/CN=uis001
Creating temporary proxy ..................................... Done
Contacting  voms-grid.uis.edu.co:15000 [/C=CO/O=UIS/OU=Internet/CN=VOMS
Grid UIS] "uis" Done
Creating proxy ............................... Done
Your proxy is valid until Mon Jul 14 05:59:04 2008
```

Existen diferentes opciones en la creación del certificado proxy, opciones que pueden variar la seguridad del funcionamiento del usuario en el Grid Computacional. Tal como la duración de la validez del certificado proxy.

8.7. INFORMATION SERVICE

En la arquitectura de un Grid con gLite existen sistemas que se encargan de manejar la información acerca de los recursos en diferentes niveles, de esta manera es como se pueden monitorear el estado de los recursos. Esta información es esencial para el funcionamiento de todo el Grid, y es mediante este servicio que los recursos son descubiertos. El servicio recibe el nombre de Information Service.

En gLite 3.1 dos servicios de información (Information Service) son utilizados; el *Globus Monitoring* and *Discovery Service (MDS)* que se encarga de descubrir recursos y de publicar su estado, y el Relational Grid Monitoring Architecture (R-GMA), encargado de contabilizar, monitorear y publicar información a nivel de usuario.

Existe un modelo conceptual de datos común que es usado para monitorear y descubrir recursos llamado GLUE Schema[32]. El MDS implementa el GLUE Schema utilizando OpenLDAP, una implementación de código abierto del protocolo liviano de acceso a directorios LDAP[33].

El Information Service comprende una serie de servicios que se encargan de monitorizar y publicar la información de los recursos a diferentes niveles, estos niveles son: nivel de recursos, nivel de sitio y nivel top.

32 https://edms.cern.ch/file/722398//gLite-3-UserGuide.html#SECTION00016000000000000000

33 Base de datos optimizada para leer, navegar y buscar información.

8.7.1. NIVEL DE RECURSOS

En este nivel los recursos de cómputo y almacenamiento del Grid como los *computing elements* y *storage elements* y algunos otros como *resource broker, MyProxy* ejecutan cada uno un servicio llamado *Information Provider*, el cual consiste en un conjunto de scripts y sensores que se encargan de extraer la información estática y dinámica de un recurso. Por ejemplo en un storage element el tipo de almacenamiento; Classic SE, DPM, dCACHE o CASTOR es información estática, mientras que la información dinámica es el espacio en disco usado. Esta información es publicada por un servicio llamado Grid Resource Information Server (***GRIS***) el cual es un servidor LDAP que se ejecuta sobre el recurso y se encarga de llevar esta información a un nivel superior, el nivel de sitio.

8.7.2. NIVEL DE SITIO

La información publicada por cada Grid Resource Information Server (***GRIS***) es referente a cada recurso que pertenezca a un sitio, comprendiendo un sitio como una agrupación de recursos compartidos a través de un Grid. Para obtener la información de todos los recursos presentes en un sitio se cuenta con un servicio llamado Grid Index Information Server (***GIIS***) el cual recolecta la información de todos los GRISes de un sitio y la lleva al nivel mas alto, el nivel top, en gLite 3.1 los GIIS se conocen como Site BDII.

8.7.3. NIVEL TOP

Este nivel posee un servicio centralizado llamado TOP-Level Berkeley Database Information Index (TOP-Level BDII) que también se conoce como BDII Server, el cual recoge toda la información proveniente de los Site BDII y los almacena en una caché, el BDII Server consta de dos servidores LDAP uno para el acceso de escritura y otro para lectura que mediante redireccionamiento de puertos permite habilitar una base de datos para publicar la información mientras la otra está siendo actualizada. El TOP-Level BDII puede ser configurado para recolectar la información publicada de los recursos de todos los sitios en el Grid o un subconjunto de ellos, generalmente es configurado un TOP-Level BDII por organización virtual (VO) con el fin de tener separada la información de los recursos a los que tiene acceso una organización virtual.

La arquitectura del ***Information Service*** para un sitio es la siguiente:

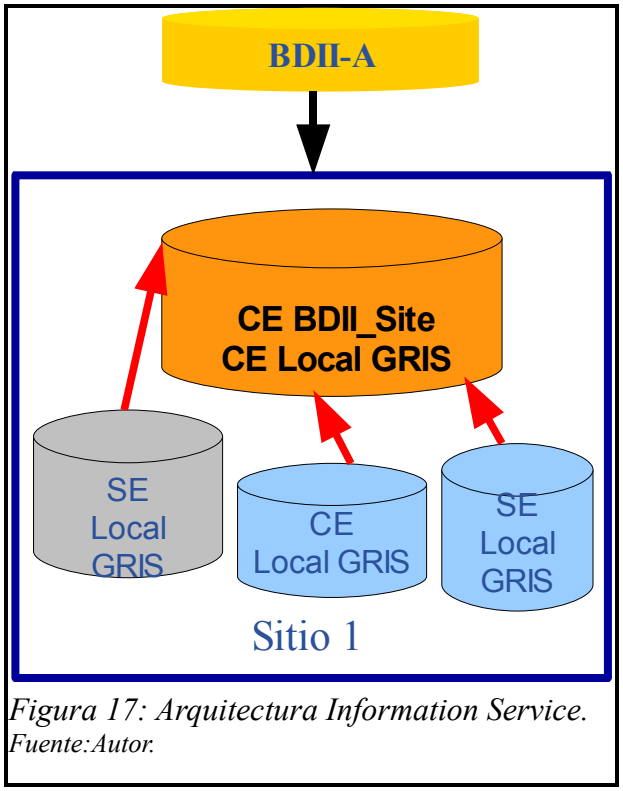


Figura 17: Arquitectura Information Service.
Fuente:Autor.

Se aprecia que el *BDII* es el principal componente de el Information Service siendo el encargado de gestionar la información en los dos niveles mas altos. En un Grid con gLite pueden existir varios sitios, generalmente se configura un BDII (**TOP-Level BDII**) por cada organización virtual, y estas a su vez pueden tener derecho a utilizar recursos de varios sitios, si este es el caso la arquitectura sería la siguiente:

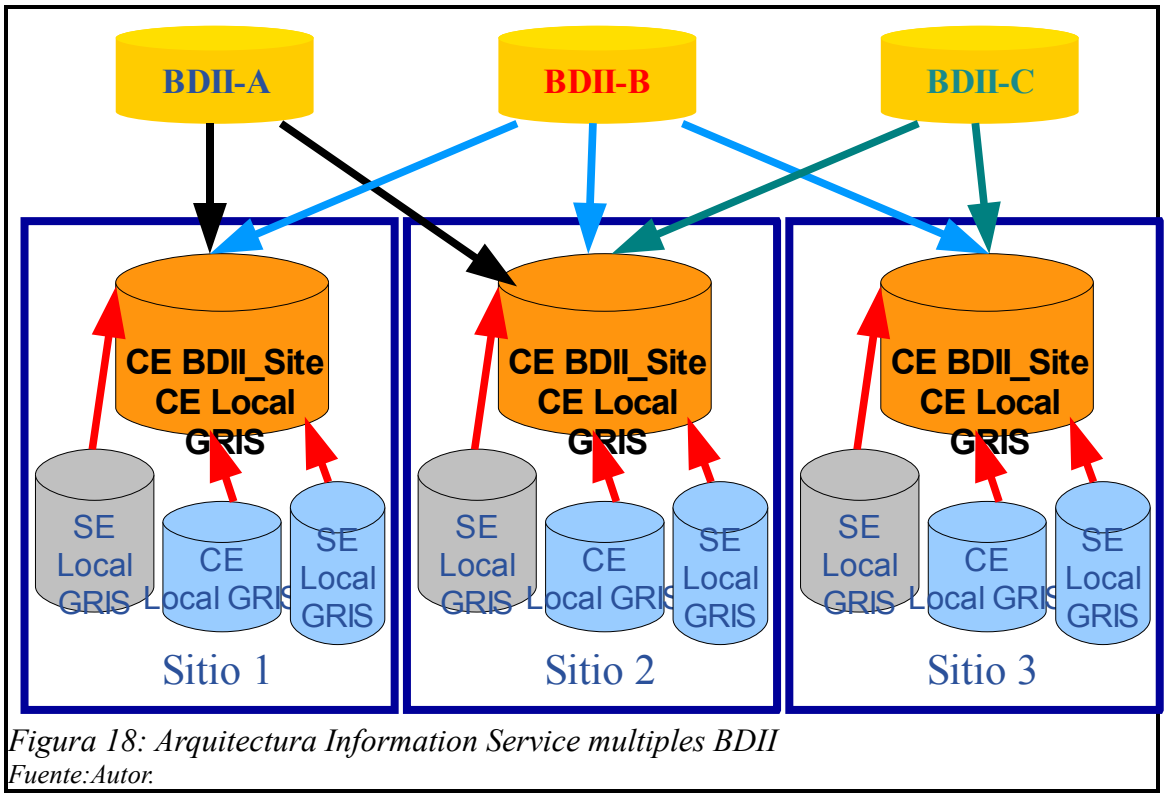


Figura 18: Arquitectura Information Service multiples BDII
Fuente:Autor.

Suponiendo que existan 3 organizaciones virtuales se puede instanciar un *BDII-Top* por cada una de ellas, por ejemplo: la organización virtual **A** tendría permisos para utilizar recursos del sitio *1*, la organización **B** para recursos de todos los sitios y la organización **C** para recursos pertenecientes a los sitios *2* y *3*. Cuando un usuario envía un trabajo, se verifica con su certificado a que organización virtual pertenece, posteriormente el ***Resource Broker*** busca un ***Computing Element*** adecuado, esto lo hace consultando al *BDII-Top* que tiene la información de los recursos a los que la organización virtual a la que pertenece el usuario tiene derecho a usar.

8.7.4. INFORMATION SERVICE EN UN GRID

Cuando un usuario envía un trabajo desde la User Interface al Resource Broker éste último revisa el archivo con formato jdl y con base a su contenido inicia la búsqueda de los recursos que cumplan con los requisitos especificados para llevar a cabo el cumplimiento del trabajo, este procedimiento es realizado consultando al Information Service.

El Information Service mantiene la información del estado actual de los recursos

en un Grid, cuando el Resource Broker realiza una consulta, obtiene la información de los Computing Elements disponibles, posteriormente el Resource Broker compara los requisitos especificados con las características de los diferentes Computing Elements y asigna el trabajo al mas apropiado.

8.7.5. FUNCIONAMIENTO INTERNO DEL INFORMATION SERVICE

El componente principal del Information Service es el Berkeley Database Information Index (BDII). El BDII consiste en dos o mas bases de datos LDAP que son alimentadas mediante un proceso de actualización. El redireccionamiento de puertos es usado para habilitar una base de datos para ser servidor de datos mientras la otra se actualiza. El proceso de actualización es el siguiente:

- Si está activado, actualiza el archivo de configuración que contiene la URL LDAP de la web.
- Escribe el archivo LDIF en un directorio temporal.
- Si está habilitado intenta obtener un archivo LDIF que modificará (filtrar) los datos.
- Modifica los datos leídos de los archivos temporales.
- Elimina la mas antigua instancia de la base de datos.
- Limpia la correspondiente base de datos.
- Inserta los datos nuevos a la nueva base de datos.
- Inicia la nueva base de datos.
- Redirecciona el trafico desde el puerto de lectura del BDII a la nueva base de datos.
- Duerme por un tiempo.

El proceso de actualización del LDIF se puede obtener haciendo un *ldapsearch* en las url de LDAP o ejecutando un script local que genere el LDIF. Existen opciones para actualizar la lista de URL's de LDAP desde una página web y hacer uso de un archivo LDIF desde una página web para modificar los datos antes de insertarlos en la base de datos. Por defecto el BDII es configurado para correr el GIP (Generic Information Provider).

8.7.6. INSTALACIÓN Y CONFIGURACIÓN.

Para la instalación y configuración del BDII se siguen los pasos similares a los anteriores en cada componente, esto significa instalar el paquete *uis-openca.rpm* el cual contiene los certificados de la OpenCA que valida los certificados y las configuraciones del /etc/hosts para los nodos disponibles, en el caso del BDII se instala vía yum el meta-paquete que instalara todas las dependencias asociadas al BDII.

```
[root@bdii-grid ~]# yum install glite-BDII
```

Posteriormente, se crea copia el archivo site-info.def *(Ver Anexo 12.11)* usado para la configuración con el manejador de paquetes yaim y se configura el nodo correspondiente.

Configuración del nodo BDII_Site

```
[root@bdii-grid ~]# /opt/glite/yaim/bin/yaim -c -s /root/site-info.def -n
BDII_site
```

Configuración del nodo BDII_Top

```
[root@bdii-grid ~]# /opt/glite/yaim/bin/yaim -c -s /root/site-info.def -n
BDII_top
```

Se edita el fichero *bdii.conf* con los parámetros personalizados al proyecto.

```
[root@bdii-grid ~]#  nano /opt/bdii/etc/bdii.conf  (Ver Anexo 12.12)
```

Por ultimo se reinicia el servidor.

```
[root@bdii-grid ~]# /etc/init.d/bdii restart
```

Para realizar una prueba de búsqueda ejecutamos el comando ldapsearch

```
[root@bdii-grid ~]# ldapsearch -x -h bdii -p 2170 -b 'mds-vo-
name=resource,o=grid'
```

8.8. WORKLOAD MANAGEMENT SYSTEM.

El Workload Management System, creado para gLite 3.1, es el componente del grid por el cual un usuario puede enviar trabajos y realizar todas las tareas requeridas para su ejecución sin tener que conocer toda la complejidad del grid computacional. El usuario se encarga de describir el trabajo que va a ejecutar y obtener la salida del mismo cuando ha sido ejecutado.

Existen 2 tipos de Workload Management System desarrollados en el grid WLCG/ EGEE, el LCG-2 y el nuevo sistema desarrollado a partir del proyecto EGEE.

La interacción de un usuario con el WMS se realiza mediante la descripción de características y requerimientos de la solicitud del trabajo a enviar definida por el lenguaje de descripción de trabajo [Job Description Language](JDL). Para un ejemplo vease el anexo 12.19 tiempos.jdl.

El WMS traduce estos requerimientos en un conjunto de recursos disponibles en

el grid que cumplan con ellos.

Los tipos de trabajos que se pueden enviar a ejecución son:

- Batch-like.
- DAG Workflow.
- Collection.
- Parametric.
- MPI.
- Interactive.

8.8.1. COMPONENTES DEL WMS

Después de realizar una solicitud de envío con los comandos instalados por defecto para manejo de trabajos desde la UI. La solicitud pasa a través de diferentes componentes que hacen parte del WMS.

En este proceso la solicitud cambia de estado a medida que cada uno de los componentes interviene. Los componentes para el manejo de trabajos del WMS son:

• WMProxy/Network Server :

Cualquiera de los dos es encargado de aceptar solicitudes desde el UI (envios de trabajos, cancelar trabajos) y los trabajos son válidos pasan al Workload Manager. El WMProxy es una implementación nueva la cual reemplaza el antiguo componente llamado Network Server. Este nuevo componente es una interface Web Services, para la misma función que el Network Server.

• Workload Manager (WM):

Es el componente principal del WMS. Para un trabajo hay dos tipos de solicitudes: envíos y cancelaciones. En el caso del envío la responsabilidad del trabajo a ejecutar es asignada al WM, esté a su vez tiene la tarea de enviar el trabajo a un CE apropiado para su ejecución con base en los requerimientos y preferencias consignadas en el JDL. Asignar los recursos no solamente depende del estado de los mismos, también de la políticas de uso que han sido establecidas por los administradores de los mismos y de los administradores de las VO a la cual el usuario pertenece.

• Resource Broker (RB):

Llamado de otra forma “Matchmaker” es una de las “classes” que ayudan al WM en la decisión, el provee un servicio de búsqueda para el recurso que más coincida con la solicitud.
El WM posteriormente puede adoptar diferentes políticas para programar un trabajo. Por un lado se podría considerar que el trabajo corresponde al uso de un recurso lo más pronto posible y una vez la decisión es tomada el trabajo es enviado al recurso para su ejecución, en otro esquema los trabajos están ocupados por el WM hasta que un recurso este disponible, en ese punto el recurso se compara con el trabajo a ejecutar y el trabajo que se adapte mejor pasa al recurso para la ejecución inmediata.

• CondorC/DAGMan:

CondorC es el módulo que hace el manejo de las operaciones reales del trabajo actual expedidas a petición del WM. Mientras que DAGMan (DAG Manager) es un meta-planificador cuyo principal objetivo es determinar cuales nodos se encuentran libres de dependencias y realizar el seguimiento de la ejecución del trabajo correspondiente. Una instancia DAGMan es generada por CondorC por cada DAG.

• Logging and Bookkeeping:

Este servicio provee soporte para el monitoreo del trabajo, almacenando información sobre registros y acontecimientos generados por los diferentes componentes del WMS. Usando esta información el servicio LB mantiene una vista de estados para cada uno de los trabajos.

El usuario puede saber en que estado se encuentra su trabajo realizando consultas al servicio LB (usando los comandos que provee el UI para el LB). Además de las consultas el usuario también puede registrarse para recibir notificaciones de cambios de estado de un trabajo específico (ejemplo, cuando un trabajo termina su ejecución). La notificación es enviada mediante el uso de una infraestructura apropiada.

• Log Monitor:

El monitor de registros es el responsable de mirar el archivo de registro de CondorC, capturar eventos interesantes concernientes a los trabajos, esto quiere decir eventos que afectan el estado del trabajo (Ejemplo: Trabajo terminado, cancelado, etc), y posteriormente realizar las respectivas acciones.

8.8.2. INSTALACIÓN Y CONFIGURACIÓN WMS

La instalación y configuración del WMS, representa un proceso extenso debido a la cantidad de nodos que lo conforman, para el respectivo proyecto se ha utilizado una versión estable de wms basado en gLite 3.0 con Scientific Linux 3.0.8 debido a bugs presentados en la versión de gLite 3.1 que no permiten un óptimo funcionamiento.

El primer paso es configurar los repositorios de APT, para Scientific Linux 3.0.8 a descargar de los repositorios:

```
[root@wms-grid root]# cd /etc/apt/sources.list.d/
[root@wms-grid sources.list.d]# wget grid-
it.cnaf.infn.it/mrepo/lists/sl3.list
[root@wms-grid sources.list.d]# wget grid-
it.cnaf.infn.it/mrepo/lists/lcg-ca.list
[root@wms-grid sources.list.d]# wget grid-
it.cnaf.infn.it/mrepo/lists/glite.list
[root@wms-grid sources.list.d]# wget grid-
it.cnaf.infn.it/mrepo/lists/glite-wms.list
[root@wms-grid sources.list.d]# wget grid-
it.cnaf.infn.it/mrepo/lists/ig.list
[root@wms-grid sources.list.d]# echo "rpm http://grid018.ct.infn.it/rep
gilda_app-i386 app 3_0_0" >  /etc/apt/sources.list.d/gilda.list
```

Se ejecuta apt-get update y se procede a instalar el metapaquete de WMS.

```
[root@wms-grid root]# apt-get install  ig_WMSLB
```

Después se ubican los ficheros correspondientes a los certificados del nodo WMS con los permisos adecuados en */etc/grid-security* como en los nodos anteriores, y se procede a configurar el WMSLB con yaim.

Configuración via yaim de WMS

```
[root@wms-grid  root]#  /opt/glite/yaim/bin/ig_yaim  -c  -s  /root/site-
info.def -n ig_WMS
```

Configuración via yaim de LB

```
[root@wms-grid  root]#  /opt/glite/yaim/bin/ig_yaim  -c  -s  /root/site-
info.def -n ig_LB
```

Después de instalar y configurar el WMS con los manejadores de paquetes, es

necesario editar */opt/glite/etc/glite_wms_wmproxy.gacl (Ver Anexo 12.14),* este archivo es modificado con el fin de agregar los roles y grupos de usuarios que podrán hacer uso de los recursos, aquí se define de acuerdo al certificado que se presente si puede o no hacer uso del WMS.

Se reinician los servicios y se prueba el envío de trabajos desde la UI.

```
[root@wms-grid root]# /etc/init.d/gLite restart
```

Cualquier información que se necesite saber adicional de los registros que ocurren durante el envío de un trabajo, considerando la cantidad de servicios que tiene el WMS se puede encontrar en la ruta /var/log/glite

8.9. COMPUTING ELEMENT

Un Computing Element es un servicio que representa un recurso de cómputo en el grid, su principal funcionalidad es la gestión de trabajos una vez le son asignados. Consiste en un conjunto de servicios que hacen posible recibir trabajos y enviarlos a ejecutar en los Worker Nodes que se encuentran bajo su administración. Un Computing Element incluye: un servicio denominado Grid Gate (GG) el cual actúa como una interfase genérica a un cluster, un Local Resource Management System (LRMS) también llamado batch system y un cluster como tal, que es una colección de Worker Nodes, los nodos donde los trabajos se ejecutarán.

El *Grid Gate* es responsable de aceptar trabajos y enviarlos para su ejecución en los ***Worker Nodes*** a través del *LRMS*. En gLite 3.1 los tipos de *LRMS* soportados son OpenPBS/PBSPro, LSF, MAUI/Torque, BQS y Condor, y se encuentra en desarrollo el soporte para Sun GridEngine.

Los ***Worker Nodes*** generalmente tienen los mismos comandos y librerías instalados que la ***User Interface***, además de los comandos para administrar los trabajos. Pueden pre-instalarse aplicaciones software para una VO específica en un área dedicada, generalmente en un sistema de archivos compartido accesible para todos los ***Worker Nodes***.

8.9.1. COMPUTING ELEMENT EN UN GRID

Cuando un usuario ha enviado un trabajo a través de la *User Interface* al *Resource Broker*, éste consulta al *Information Service* acerca del estado de los recursos de cómputo, luego el *Resource Broker* escoge un *Computing Element* que cumpla con los requerimientos del trabajo enviado y le asigna la ejecución de dicho trabajo. Posteriormente el *Computing Element* acepta el trabajo y lo envía

para su ejecución a los Worker Nodes que tiene a su disposición a través del *LRMS*.

Al inicio del envío del trabajo algunos archivos de entrada son especificados, este conjunto de archivos conforma los que se conoce como *Input Sandbox*, estos son copiados inicialmente de la *User Interface* al *Resource Broker* y cuando el trabajo es enviado por el *LRMS* a los Worker Nodes el *Input Sandbox* es copiado del *Resource Broker* a un Worker Node disponible. A partir de este momento el *Computing Element* se encarga de monitorear la ejecución del trabajo en los Worker Nodes mediante el *LRMS.*

Si el trabajo termina de ejecutarse sin errores la salida *Output Sandbox* es transferida al *Resource Broker.* Si por algún motivo el *Computing Element* seleccionado no acepta el trabajo para ser ejecutado el trabajo es reenviado a otro *Computing Element* que cumpla los requerimientos.

8.9.2. FUNCIONAMIENTO INTERNO DEL COMPUTING ELEMENT

El *Computing Element* puede ser utilizado por un cliente genérico o por el *Workload Management System*, en el primer caso un usuario final interactúa directamente con el *Computing Element*, en el segundo el *Resource Broker* envía un determinado trabajo a un *Computing Element* adecuado encontrado mediante un proceso de selección. Para la asignación de trabajos, el *Computing Element* puede trabajar bajo dos modelos: el modelo *"push"* y el *"pull"*.
En el modelo *"push"* el *Computing Element* está en espera de que le sea asignado un trabajo para su ejecución. En el modelo *"pull"* el *Computing Element* pregunta por posibles trabajos al *Resource Broker* para que este se los asigne.

Un componente importante del *Computing Element* es el *LRMS* o *Batch System*, que es un software que se encarga de controlar de manera eficaz la ejecución de trabajos en los recursos disponibles sin que un usuario tenga que estar interviniendo para esto. Los trabajos enviados a través del *LRMS* son colocados en una cola y son ejecutados cuando los recursos estén disponibles. Es mediante este servicio que el *Computing Element* envía los trabajos a Worker Nodes.

A su vez el *LRMS* está compuesto de diferentes servicios, en el caso de TORQUE el *LRMS* utilizado en el proyecto se compone de:

- *pbs_server:* el cual provee funcionalidades básicas como recibir trabajos, crear colas y proteger los trabajos ante fallas del sistema
- *job_sheduler:* se encarga de decidir cual es el trabajo que debe ser ejecutado en base a la política asignada

- *pbs_mom:* éste coloca los trabajos en ejecución, es también el responsable de retornar la salida de los trabajos a los usuarios

De esta manera es como el *Computing Element* realiza su función en el Grid valiéndose de diferentes componentes ya existentes, ajustados a las necesidades del middleware.

8.9.3. INSTALACIÓN Y CONFIGURACIÓN COMPUTING ELEMENT

La instalación y configuración de un Computing Element es similar al proceso empleado en los otros nodos, inicialmente se debe instalar el metapaquete lcg-CE el cual se encuentra en los repositorios del CERN.

```
[root@ce-grid ~]#  yum install lcg-CE
```

Luego, se crea el archivo /etc/glite/yaim/wn-list.conf *(Ver Anexo 12.15)* y se agregan linea por linea los hostname de los ***Worker Nodes*** donde se ejecutarán los trabajos. Posteriormente, es necesario con YAIM configurar el nodo usando el archivo site-info.def

```
[root@ce-grid ~]# /opt/glite/yaim/bin/yaim -c -s /root/site-info.def -n
lcg-CE -n TORQUE_server -n TORQUE_utils
```

El script configura el CE y el servidor TORQUE así, como el MAUI que será el encargado de manejar la cola de trabajos a enviar a los Workers Nodes.

Es necesario que la comunicación entre el Worker Node y el Computing Element sea de manera transparente, es decir, que no haya necesidad de especificar contraseñas para poder responder a los trabajos enviados.

Se modifican las plantillas correspondientes a los archivos /opt/edg/etc/edg-pbs-knownhosts.conf *(Ver Anexo 12.16)* y /opt/edg/etc/edg-pbs-shostsequiv.conf *(Ver Anexo 12.17)* y se ejecuta los siguientes comandos:

```
[root@ce-grid ~]#  /opt/edg/sbin/edg-pbs-knownhosts
[root@ce-grid ~]#  /opt/edg/sbin/edg-pbs-shostsequiv
```

Estos comandos crean el archivo /etc/ssh/shostsequiv y el /etc/ssh/knownhosts estos archivos contienen la llave pública que será aceptada por el CE para la conexión sin contraseñas basada en confianza.
Se modifica el archivo de configuración del servidor ssh del Computing Element anexandole las lineas:

HostbasedAuthentication yes
IgnoreUserKnownHosts yes
IgnoreRhosts yes

Se realiza la transferencia de los archivos generados a los Workers Nodes que se conectaran al Computing Element.

```
[root@ce-grid ~]#  scp /etc/ssh/ssh_known_hosts root@wn:/etc/ssh
[root@ce-grid ~]#  scp /etc/ssh/shosts.equiv root@wn:/etc/ssh
```

Para comprobar el estado de los Worker Nodes se ejecuta :

```
[root@ce-grid ~]# pbsnodes -a
wn-grid.uis.edu.co
state = free
np = 1
properties = lcgpro
ntype = cluster
status = opsys=linux,uname=Linux wn-grid.uis.edu.co
2.6.9-42.0.3.EL.xs0.4.0.263xenU #1 SMP Fri Jan 5 05:46:45 EST 2007
i686,sessions=? 0,nsessions=?
0,nusers=0,idletime=95,totmem=786524kb,availmem=751668kb,physmem=262244kb
,ncpus=1,loadave=0.00,netload=136382321,state=free,jobs=,varattr=,rectime
=1224454511

wn.ula.ve
state = free
np = 1
properties = lcgpro
ntype = cluster
status = opsys=linux,uname=Linux wn-grid.ula.ve 2.6.18-5-xen-amd64 #1 SMP
Tue Jun 17 00:10:27 UTC 2008 x86_64,sessions=? 0,nsessions=?
0,nusers=0,idletime=120,totmem=2097144kb,availmem=2013416kb,physmem=10485
76kb,ncpus=1,loadave=0.26,netload=513245678,state=free,jobs=,varattr=,rec
time=1224454514

wn.utp.ac.pa
state = free
np = 1
properties = lcgpro
ntype = cluster
status = opsys=linux,uname=Linux wn-grid.utp.ac.pa 2.6.18-6-xen-vserver-
amd64 #1 SMP Tue Jun 17 00:10:27 UTC 2008 x86_64,sessions=? 0,nsessions=?
0,nusers=0,idletime=443,totmem=2097144kb,availmem=2013416kb,physmem=10485
76kb,ncpus=1,loadave=0.26,netload=51712551,state=free,jobs=,varattr=,rect
ime=1224454514
```

Para el envío de un trabajo de prueba se accede como usuario, se crea un archivo hostname.sh el cual contiene la ejecución de un comando en el bash:

```
[uis001@ce-grid ~]$ nano hostname.sh
      #!/bin/sh
      hostname
[uis001@ce-grid ~]$ qsub -q uis hostname.sh
27.ce-grid.uis.edu.co
```

La salida del envío del trabajo genera 2 archivos los cuales tienen el nombre del id del trabajo y una letra correspondiente a “e” indicando los errores si han ocurrido y otra con la letra “o” indicando la salida generada por el script.

```
[uis001@ce-grid ~]$ ls
hostname.sh  hostname.sh.e27  hostname.sh.o27
[uis001@ce-grid ~]$ cat hostname.sh.o27
wn-grid.uis.edu.co
```

En cualquier momento de la ejecución es posible mirar el estado de la cola de trabajos del *LRMS* con el siguiente comando:

```
[uis001@wn-grid ~]$ qstat -a
ce-grid.uis.edu.co:
                                                                  Req'd  Req'd   Elap
Job ID               Username Queue    Jobname    SessID NDS   TSK Memory Time  S Time
-------------------- -------- -------- ---------- ------ ----- --- ------ ----- - -----
28.ce-grid.uis.edu.c uis001   uis      hostname.s    --    --   --    --  48:00 Q   --
```

8.10. WORKER NODE.

Los Worker Nodes son los nodos que finalmente ejecutan los trabajos enviados a un Grid Computacional, pueden ser computadores de altas prestaciones o los nodos de un cluster de alto desempeño. Estos nodos poseen un conjunto de clientes requeridos para ejecutar los trabajos enviados por un ***Computing Element*** a través de el *LRMS*. El metapaquete gLite-WN actualmente incluye el cliente de acceso de gLite, el cliente del ***Logging and Bookkeeping***, el cliente *R-GMA* y la librería Checkpointing del ***Workload Management System***.

8.10.1. WORKER NODE EN UN GRID

Cuando se asigna un trabajo a un ***Computing Element*** éste lo envía a ejecutarse a los Worker Nodes que tenga a su disposición a través del *LRMS.* Los Worker Nodes poseen el cliente del *LRMS* y reciben los trabajos a ejecutar, una vez ejecute el trabajo satisfactoriamente, cada ***Worker Node*** envía su resultado al ***Computing Element*** de nuevo a través del *LRMS*.

8.10.2. FUNCIONAMIENTO INTERNO DEL WORKER NODE

Los Worker Nodes están configurados con el cliente del *LRMS* para recibir los

trabajos enviados desde el ***Computing Element***, en los Worker Nodes debe estar instalado todo lo necesario para poder ejecutar los trabajos; librerías, aplicaciones etc. A su vez se puede configurar un sistema de archivos compartido entre los Worker Nodes para tener un área dedicada para ciertas aplicaciones.

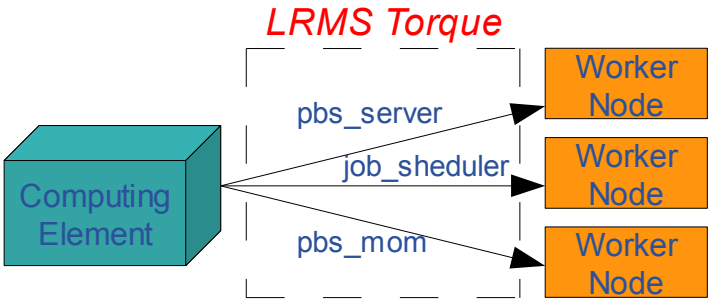


Figura 19: Arquitectura de conexión entre Computing Element y Worker Nodes.
Fuente:Autor.

8.10.3. INSTALACIÓN Y CONFIGURACIÓN WORKER NODE

En la instalación y configuración de un Worker Node luego de instalar los requisitos para cualquier nodo se ubica en */etc/yum.repos.d* los archivos que permitirán instalar desde los repositorios del CERN los respectivos metapaquetes para el Worker Node:

```
[root@wn-grid yum.repos.d]# cd /etc/yum.repos.d/
[root@wn-grid yum.repos.d]# wget http://grid-deployment.web.cern.ch/grid-deployment/glite/repos/glite-WN.repo

[root@wn-grid yum.repos.d]# wget http://grid-deployment.web.cern.ch/grid-deployment/glite/repos/glite-TORQUE_client.repo
```

Posteriormente se instalan los metapaquetes:

```
[root@wn-grid yum.repos.d]# yum install glite-WN glite-TORQUE_client
```

Una vez terminada la instalación se procede a editar el archivo /etc/glite/yaim/wn-list.conf de tal forma que coincida con el presente en el Computing Element:

```
[root@wn-grid ~]# nano /etc/glite/yaim/wn-list.conf

wn-grid.uis.edu.co
wn-grid.ula.ve
```

```
wn-grid.utp.ac.pa
```

luego se configura a través de yaim:

```
[root@wn-grid]# /opt/glite/yaim/bin/yaim -c -s site-info.def -n glite-WN
-n TORQUE_client
```

Luego de esto se dispone de un *Worker Node* en el cual se pueden ejecutar trabajos ya sea como parte de un cluster o como una máquina aislada.

9. PRUEBA Y PUESTA A PUNTO GRID COMPUTACIONAL UIS

Con el fin de realizar una puesta a punto del Grid Computacional UIS, es necesario verificar la velocidad, latencia y accesibilidad de los *Wokers Nodes*, en los sitios en los cuales ha sido instalados. En el desarrollo del proyecto se ha instalado *Worker Nodes* en las máquinas de la Universidad de Los Andes, Mérida – Venezuela y en la Universidad Tecnológica de Panamá.

9.1. PRUEBAS DE CONEXIÓN

Para la realización de pruebas de conexión se utilizó la herramienta Iperf. Iperf se trata de una aplicación usada comúnmente en pruebas de redes que puede usar los protocolos TCP y UDP para medir el ancho de banda a través de la red. Iperf permite al usuario configurar varios parámetros, entre ellos el tamaño de información a transmitir, consiste en un cliente y servidor este último se prepara para aceptar una cantidad de paquetes y el cliente los envía dependiendo del protocolo hacia el servidor.

Para ejecutar desde la linea de comandos en modo servidor se realiza con el comando *iperf -s -p puerto*, siendo *-s* el parámetro para que actué como servidor y *-p* es opcional para indicar el puerto donde escucha. En el caso del cliente ejecuta el comando *iperf -c ip -p* puerto el cual *-c* indica que funciona en modo cliente y *-p* el puerto el cual el servidor está escuchando.

Las pruebas se realizaron durante 5 dias, en cada uno se ejecutó la prueba tres veces en cada intervalo de tiempo definido, obteniendo de esta forma resultados suficientes para estimar un promedio que sea representativo cuando se mide el ancho de banda de una red. Para las pruebas realizadas con Iperf en el Grid Computacional entre la conexión del nodo UI y un Worker Node en la ULA y UTP, se configuraron los siguientes parámetros:

- Ancho de la ventana: 16 Kbyte (Valor tomado por omisión para el calculo)
- Datos Transferidos: 1 MByte

Las pruebas se realizaron con la siguientes lineas de comandos:

Para el Servidor:

```
[root@ui-grid ~]# iperf -s -p 22
```

Para los clientes:

```
[root@wn-ula ~]# iperf -c ui-grid.uis.edu.co -p 22 -n 1M
[root@wn-utp ~]# iperf -c ui-grid.uis.edu.co -p 22 -n 1M
```

9.2. RESULTADOS.

Los resultados de las pruebas realizadas con iperf, arrojaron resultados que se resumen en la siguiente tabla:

Rango de tiempo	Velocidad Promedio [Kbits/sec] ULA	Velocidad Promedio [Kbits/sec] UTP
8 AM – 12 M	367	361
12 – 6 PM	421	385
6 PM – 12 AM	440	404
12 AM – 8 AM	463	489

Tabla 1: Resultados de pruebas de conexión con iperf.

Como se puede observar en los resultados, en la franja horaria de 12 de la noche a 8 de la mañana la velocidad de conexión es relativamente mayor. Lo cual indicaría un espacio de tiempo óptimo para el envío de trabajos en el Grid.

9.3. PROCEDIMIENTO ENVÍO DE TRABAJOS

Para el envío de un trabajo en un Grid con gLite, después de haber instalado los nodos pertinentes se siguen los siguientes pasos:

Acceso a la UI con la cuenta de usuario asociada al certificado.

```
ivostro:~# ssh uis001@200.21.228.158
Scientific Linux CERN SLC release 4.6 (Beryllium)
uis001@200.21.228.158's password:
Last login: Thu Aug 28 09:51:34 2008 from 192.168.6.248
[uis001@ui-grid ~]$
```

Si es la primera vez que se va a hacer uso del Grid, se ubican los certificados en el directorio .globus, previamente obtenidos de la CA reconocida por el VOMS. Los permisos de estos archivos deben ser 400 para el userkey.pem y 644 para el

usercert.pem. Adicionalmente se debe configurar el archivo vomses en el directorio .glite con las lineas que indican las VOMS a utilizar.

```
[uis001@ui-grid ~]$ ls -l .globus/
total 8
-rw-r--r--  1 uis001 uis001 2121 Jul 28 16:13 usercert.pem
-r--------  1 uis001 uis001 1015 Jul 28 16:12 userkey.pem
[uis001@ui-grid ~]$ ls .glite/
vomses
[uis001@ui-grid ~]$ cat .glite/vomses
"uis"  "voms-grid.uis.edu.co"  "15000"  "/C=CO/O=UIS/OU=Internet/CN=voms-
grid.uis.edu.co" "uis" "/C=CO/O=UIS/OU=Internet/CN=voms-grid.uis.edu.co"
```

Después se valida el certificado frente a la VO UIS y se inicia la autenticación.

```
[uis001@ui-grid ~]$ glite-voms-proxy-init --voms uis
Your identity: /C=CO/O=UIS/OU=Internet/CN=uis001
Creating temporary proxy ............................... Done
Contacting   voms-grid.uis.edu.co:15000  [/C=CO/O=UIS/OU=Internet/CN=voms-
grid.uis.edu.co] "uis" Done
Creating proxy ........................................... Done
Your proxy is valid until Mon Oct 13 09:29:00 2008
```

Se buscan los CE disponibles con el comando:

```
[uis001@ui-grid ~]$ lcg-infosites --vo uis ce
#CPU  Free  Total Jobs  Running     Waiting     ComputingElement
----------------------------------------------------------
   2     2     0            0 444444      ce-
grid.uis.edu.co:2119/jobmanager-lcgpbs-uis
```

Solo existe un CE disponibles con varios WN asociados, a este CE se le envía el trabajo:

```
[uis001@ui-grid ~]$ glite-job-submit -a hostname.jdl
======================glite-job-submit Success ======================
The job has been successfully submitted to the Network Server.
Use glite-job-status command to check job current status.
Your job identifier (edg_jobId) is:
- https://wms-grid.uis.edu.co:9000/rIBubkFFKhnSQ6CjiLUY8Q
=====================================================================
```

Se puede verificar el estado del mismo

```
[uis001@ui-grid ~]$ glite-job-status https://wms-
grid.uis.edu.co:9000/rIBubkFFKhnSQ6CjiLUY8Q
*****************************************************************
BOOKKEEPING INFORMATION:
Printing status info for the Job:
https://wms-grid.uis.edu.co:9000/rIBubkFFKhnSQ6CjiLUY8Q
Current Status: Scheduled
```

```
Status Reason: unavailable
Destination: ce-grid.uis.edu.co:2119/jobmanager-lcgpbs-uis
reached on: Mon Oct 13 09:30:15 2008
**************************************************************
```

Finalmente obtenemos la salida del trabajo

```
[uis001@ui-grid ~]$ glite-job-output https://wms-
grid.uis.edu.co:9000/rIBubkFFKhnSQ6CjiLUY8Q

Retrieving files from host wms-grid.uis.edu.co
**************************************************************
JOB GET OUTPUT OUTCOME
Output sandbox files for the job:
- https://wms-grid.uis.edu.co:9000/rIBubkFFKhnSQ6CjiLUY8Q
have been successfully retrieved and stored in the directory:
/tmp/jobOutput/snPegp1YMJcnS22yF5pFlg
```

La siguiente gráfica presenta el esquema de envío de trabajos anteriormente descrito:

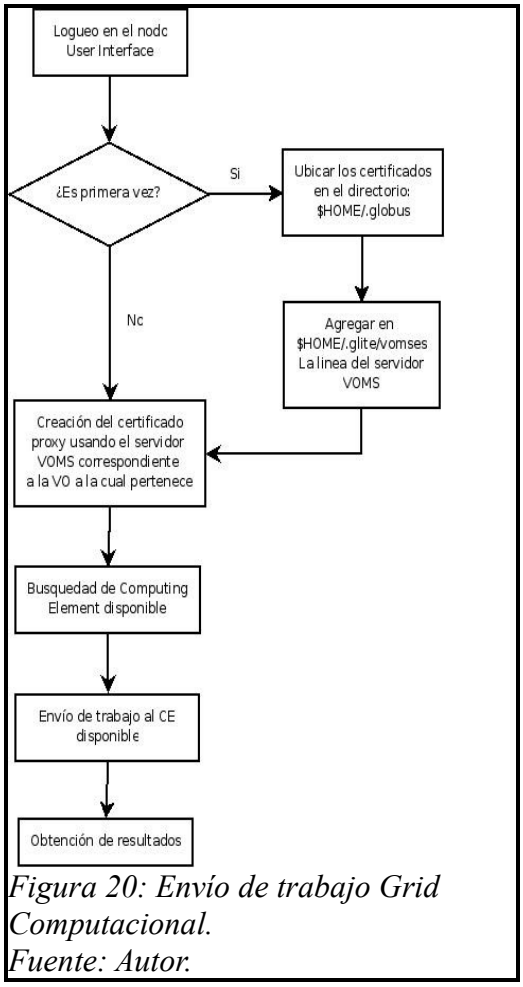


Figura 20: Envío de trabajo Grid Computacional.
Fuente: Autor.

9.4. PRUEBA DE ENVIO DE TRABAJOS

Para realizar pruebas en el Grid se escogen dos programas hechos para *Octave*[34]. Estos programas realizan operaciones típicas en las labores de todos los procesos de investigación, la descripción y codificación de los programas se encuentran en los anexos.

El primer programa a ejecutar llena dos matrices de valores aleatorios con un tamaño de 1000x1000 y posteriormente realiza el cálculo del producto de ambas. Se miden los tiempos y con base en estos se calcula en *MFlops* el número de operaciones flotantes por elementos de la matrix resultante.

Para el caso del segundo programa consiste en un benchmark que corre sobre *Octave*, realiza pruebas de programación, cálculos y funciones con matrices. Para cada una de las operaciones se toma el tiempo en segundo que se demoró.

Estas pruebas serán ejecutadas en los *Worker Nodes* UIS, ULA y UTP, nodos que tienen diferentes características técnicas y que hacen parte de los recursos con los que cuenta el Grid Computacional.

9.4.1. RESULTADOS

Los resultados para la prueba del software de producto de matrices se observan en la siguiente tabla:

Resultados Pruebas Producto Matrices		
UIS (Mflops)	ULA (Mflops)	UTP (Mflops)
692.95	460.9	450.51

Tabla 2: Resultados pruebas producto de matrices

En el caso del programa que realiza pruebas de rendimiento de octave se obtuvo la siguiente tabla de resultados:

34 Programa libre para realizar cálculos numéricos. MATLAB es considerado su equivalente comercial.

Resultados Pruebas Benchmark Octave			
Tipo de Prueba	UIS (sec)	ULA (sec)	UTP (sec)
Matrix calculation			
Creation, transp., deformation of a 1500x1500 matrix	1,11	1,16	1,02
800x800 normal distributed random matrix ^1000	0,15	0,14	0,1
Sorting of 2,000,000 random values	6,3	6,75	7,08
700x700 cross-product matrix (b = a' * a)	1,13	1,37	1,54
Linear regression over a 600x600 matrix (c = a \ b')	0,23	0,24	0,23
Trimmed geom. mean (2 extremes eliminated)	0,66	0,78	0,71
Matrix functions			
FFT over 800,000 random values	0,53	0,49	0,43
Eigenvalues of a 320x320 random matrix	0,64	0,63	0,5
Determinant of a 650x650 random matrix	0,3	0,29	0,26
Cholesky decomposition of a 900x900 matrix	0,31	0,29	0,26
Inverse of a 400x400 random matrix	0,17	0,21	0,18
Trimmed geom. mean (2 extremes eliminated)	0,36	0,37	0,31
Programmation			
750,000 Fibonacci numbers calculation (vector calc)	0,43	0,44	0,35
Creation of a 2250x2250 Hilbert matrix (matrix calc)	0,91	0,75	0,5
Creation of a 220x220 Toeplitz matrix (loops)	1,08	0,98	0,79
Escoufier's method on a 37x37 matrix (mixed)	1,18	1,14	0,92
Trimmed geom. mean (2 extremes eliminated)	0,75	0,66	0,52
Total time for all 15 tests	14,46	14,41	14,16
Overall mean (sum of I, II and III trimmed means/ 3)	0,56	0,6	0,48

Tabla 3: Resultados de pruebas de benchmark con Octave.

Se han configurado tres Worker Nodes para éste grid, ubicados en la Universidad Industrial de Santander, Colombia, en la Universidad de Los Andes, Mérida, Venezuela y el otro en la Universidad Tecnológica de Panamá en ciudad de Panamá. En los resultados obtenidos en las pruebas realizadas se puede observar un comportamiento similar entre los nodos lo cual indicaría que ninguno llega a ser considerablemente mejor que otro. Este comportamiento es debido a que los servicios del middleware usado se encuentran en fase de de construcción y desarrollo para la arquitectura 64 bits en el modo de paravirtualización, lo cual obliga a correr la imagen de 32 bits ocasionando una pérdida de rendimiento del nodo a usar debido a la incompatibilidad como sucede en los casos de los nodos en la ULA y la UTP.

10. CONCLUSIONES

Los objetivos trazados en el desarrollo del proyecto de grado se han cumplido, el Grid Computacional que se ha implementado es un Grid experimental de pequeña escala el cual sienta las bases para el desarrollo futuro de una infraestructura de Grid a nivel nacional que sirva como un recurso de cómputo de gran escala que proporcione soporte a la investigación científica.

La construcción de este Grid Computacional ha permitido conocer al mismo tiempo la complejidad en el proceso de implementar una infraestructura Grid; los mecanismos que deben utilizarse para tal fin, algunos aspectos a considerar en términos organizativos y el estado actual de esta tecnología.

Respecto a los aspectos técnicos se encuentra material de documentación necesario para implementar servicios básicos, sin embargo es necesario hacer una extensa búsqueda, análisis y experimentación cuando se trata de servicios muy especializados. La documentación existente está estandarizada y se encuentra al alcance de cualquier centro de cómputo con recursos humanos experimentados en administración de servicios y modelo de documentación de herramientas libres.

Cabe destacar que el middleware gLite es un software avanzado que hace parte de un complejo e inmenso proyecto de investigación y se encuentra en fase de experimentación y desarrollo, lo cual trabajar con él implica poseer una amplia experiencia técnica en sistemas operativos, servidores y redes. A su vez trabajar en la implementación de plataformas grandes y complejas con arquitecturas definidas a varios niveles y fuerte relación entre sus componentes exige aplicar y ayuda a fortalecer habilidades sistémicas ya que se debe tener una visión holística de todo el sistema y comprender el modelo de funcionamiento a diferentes niveles y como inciden los cambios de parámetros en toda la infraestructura, así como descubrir isomorfismos con otro tipo de plataformas en aspectos específicos como la seguridad y la ejecución de trabajos.

El estado actual de las redes de interconexión en las instituciones no favorecen la implementación de infraestructura grid ya que es claro que el tamaño de la información a manipular en un Grid Computacional es demasiado grande para la transferencia de los datos a través de una red común y que las velocidades de transmisión debe ser altas ya que se busca reducir tiempos de ejecución. Sin embargo la dirección a tomar es centrarse en el uso de mejores recursos para demostrar que con estos y una conectividad más apropiada se obtiene una infraestructura más eficiente.

En cuanto a los aspectos organizativos se encontró que los temas que cubren la tecnología de grid son de una gran variedad, y que van desde la parte técnica hasta un fuerte componente de relaciones humanas por lo que es importante establecer grupos de trabajo independientes y con suficiente cantidad de integrantes que puedan interactuar con rapidez para resolver los problemas concretos en temas como seguridad, redes, middleware, aplicaciones y relaciones de convenios, colaboración y compartir recursos.

Es necesario dar importancia a que los operadores de las diferentes redes de interconexión tengan una relación estrecha con la comunidad de grid para identificar y resolver eficazmente cualquier problema en la infraestructura de telecomunicaciones. También es de destacar el hecho de que la instalación, configuración y operación de grids, y la utilización y desarrollo de aplicaciones requieren del recurso humano adecuado y suficiente. Finalmente, debe resaltarse la importancia de interactuar con otros proyectos de grid a nivel internacional, ya que esto permite asimilar otras experiencias, lo que acelera el desarrollo de la implementación de estas nuevas tecnologías, además de establecer lazos que puedan resultar en la constitución de proyectos de grids mas grandes.

11. RECOMENDACIONES

- Para tener un Grid de producción estable que soporte una gran cantidad de usuarios se recomienda invertir en hardware de mejores características que el actual, de igual manera tener una infraestructura de redes de alta velocidad con el objetivo de que estos aspectos no influyan negativamente en el rendimiento de la infraestructura Grid.

- La computación en Grid es un concepto bastante complejo que permite abordar diferentes aspectos, con base en lo realizado en éste proyecto, se debe impulsar la ejecución de proyectos en esta área que complementen la infraestructura a diferentes niveles aprovechando las bases, conocimiento y documentación que se realizan con el desarrollo del presente trabajo de grado.

- La implementación de una infraestructura Grid requiere de un gran esfuerzo, los futuros trabajos deben ser dimensionados correctamente ya que la temática es muy extensa y consideramos que para tener una infraestructura Grid estable y disponible se deben tener personas capacitadas trabajando en el área a tiempo completo.

- La implementación del modelo de seguridad para un Grid en producción mediante el cual se compartan recursos importantes entre diferentes organizaciones debe incluir el montaje y puesta a punto de una autoridad de certificación con todos sus aspectos de seguridad y reconocida frente a las organizaciones involucradas.

- Se considera que debido al creciente y continuo desarrollo del middleware gLite en su última versión se debe mantener un sistema de vigilancia del estado del middleware para adoptar los correctivos y parches que vayan saliendo en cada uno de los componentes que conforman el Grid UIS.

- Es recomendable continuar con el crecimiento de esta infraestructura Grid mediante convenios inter-institucionales que permitan a la Universidad Industrial de Santander compartir recursos, experiencias, datos y trabajos colaborativos con otras entidades similares con el fin de que se aproveche el resultado de este trabajo ya que posibilita grandes oportunidades para la labor de investigación en conjunto con instituciones en cualquier lugar.

12. ANEXOS

12.1. INSTALACIÓN Y CONFIGURACIÓN DE XEN

SLC4.4 permite configurar el manejador de paquetes tipo rpm *yum*, accediendo a los repositorios de SLC4.4 y a su vez a los paquetes específicos para instalar y configurar Xen.

```
[root@scientific03 yum.repos.d]# /bin/rpm -ivh
http://linuxsoft.cern.ch/cern/xen/slc4X/i386/RPMS/yum-xen-
conf-4X-1.slc4.cern.noarch.rpm
```

Dos nuevos archivos que especifican donde obtener los paquetes de Xen junto a las fuentes, se encuentran en el directorio de configuración de *yum*.

```
[root@scientific03 yum.repos.d]# pwd
/etc/yum.repos.d
[root@scientific03 yum.repos.d]# ls
atrpms.repo         cern-only.repo         cern-srpms.repo       cern-
update.repo         cern-xen-srpms.repo    rhaps2.repo
cern-extra.repo     cern-only-srpms.repo   cern-test.repo        cern-
update-srpms.repo   dag.repo               rhaps2-srpms.repo
cern-extra-srpms.repo  cern.repo           cern-test-srpms.repo  cern-
xen.repo            jpackage.repo
```

Los archivos resaltados, son agregados por la instalación del paquete yum-xen-conf-4X-1.slc4.cern.noarch.rpm. El contenido de los archivos especifíca donde obtener los paquetes, un ejemplo del contenido del archivo cern-xen.repo es:

/etc/yum.repos.d/cern-xen.repo

```
#
# Scientific Linux CERN 4 (SLC4) Xen domU/dom0 packages
#
[main]
[slc4-xen]
name=Scientific Linux CERN 4 (SLC4) Xen dom0/domU packages
baseurl=http://linuxsoft.cern.ch/cern/xen/slc4X/$basearch/yum/xen/
gpgkey=http://linuxsoft.cern.ch/cern/slc4X/$basearch/docs/RPM-GPG-KEY-
cern
       http://linuxsoft.cern.ch/cern/slc4X/$basearch/docs/RPM-GPG-KEY-
jpolok
gpgcheck=1
enabled=1
protect=1
```

Después de configurar el manejador de paquetes *yum*, se instala Xen para dom0.

```
[root@vscientific03 ~]# yum install xen kernel-xen
Loading "kernel-module" plugin
```

```
Setting up Install Process
Setting up repositories
slc4-extra                100% |=========================|  951 B
00:00
slc4-base                 100% |=========================| 1.1 kB
00:00
slc4-xen                  100% |=========================|  951 B
00:00
dag                       100% |=========================| 1.1 kB
00:00
slc4-update               100% |=========================|  951 B
00:00
Reading repository metadata in from local files
primary.xml.gz            100% |=========================| 1.9 MB
01:18
dag      : ###############                                    2467/8591
```

yum actualiza la lista de paquetes y dependencias disponible verificando la existencia del paquete deseado a instalar. Posteriormente muestra una lista de paquetes a instalar que satisfacen la dependencia del paquete que se ha seleccionado inicialmente: xen y kernel-xen

```
Dependencies Resolved
================================================================================
====
 Package                  Arch       Version          Repository
Size
================================================================================
====
Installing:
 kernel-xen               i686       2.6.18-53.1.13.slc4  slc4-xen
14 M
 xen                      i386       3.0.3.0-41.slc4.1  slc4-xen
1.7 M
Installing for dependencies:
 SDL                      i386       1.2.7-8          slc4-base
202 k
 ati-fglrx                i686       1:8.28.8-1.slc4.cern  slc4-extra
7.2 M
 bridge-utils             i386       1.0.4-4          slc4-base
27 k
 compat-libstdc++-33      i386       3.2.3-47.3       slc4-base
226 k
 device-mapper-multipath  i386        0.4.5-27.el4_6.3  slc4-update
900 k
 fontconfig               i386       2.2.3-13.el4     slc4-update
118 k
 freetype                 i386       2.1.9-6.el4      slc4-update
768 k
```

```
 j2re                    i386       1.4.2_16-5.cern  slc4-update
20 M
 libjpeg                 i386       6b-33            slc4-base
126 k
 libmng                  i386       1.0.8-1          slc4-base
136 k
 libpng                  i386       2:1.2.7-3.1      slc4-update
157 k
 qt                      i386       1:3.3.3-13.RHEL4  slc4-update
2.9 M
 sysfsutils              i386       1.2.0-1          slc4-base
56 k
 xen-libs                i386       3.0.3.0-41.slc4.1  slc4-xen
124 k
 xorg-x11-Mesa-libGLU    i386       6.8.2-1.EL.33.0.2  slc4-update
457 k
 xorg-x11-deprecated-libs  i386       6.8.2-1.EL.33.0.2  slc4-update
274 k
 xorg-x11-libs           i386       6.8.2-1.EL.33.0.2  slc4-update
2.7 M
Transaction Summary
================================================================================
====
Install     19 Package(s)
Update       0 Package(s)
Remove       0 Package(s)
Total download size: 52 M
Is this ok [y/N]:
```

Se van a instalar 19, actualizar 0 y liminar 0 paquetes, el tamaño total en MegaBytes a descargar del repositorio (52 Mbytes).

Después de instalado el kernel, con soporte para que el S.O trabaje como dom0, es necesario agregar la entrada en el archivo de configuración del gestor de arranque Grub, la linea en la cual se especifica el nuevo kernel instalado.

```
[root@scientific03 yum.repos.d]# vim /etc/grub.conf

title Scientific Linux CERN SLC (2.6.18-53.1.13.slc4xen)
  root (hd0,0)
  kernel /boot/xen.gz-2.6.18-53.1.13.slc4
  module /boot/vmlinuz-2.6.18-53.1.13.slc4xen ro root=LABEL=/1 rhgb quiet
  module /boot/initrd-2.6.18-53.1.13.slc4xen.img
```

Al reiniciar e ingresar por la opción del nuevo kernel se verifica el kernel que se encuentra en ejecución.

```
[root@scientific03 ~]# uname -r
2.6.18-53.1.13.slc4xen
```

12.2. INSTALACIÓN Y CONFIGURACIÓN DE S.O. DOMU

```
[root@scientific03 ~]# wget http://project-xen.web.cern.ch/project-
xen/xen/img/slc4.gpe.img.gz
[root@scientific03 ~]# gunzip slc4.gpe.img.gz
```

Se cambia el tamaño al archivo que contiene la imagen con el objetivo de poder personalizar la imagen descargada.

```
[root@scientific03 ~]# dd if=/dev/zero of=slc4.gpe.img bs=1M conv=notrunc
count=1000 seek=2000
[root@scientific03 ~]# losetup /dev/loop1 slc4.gpe.img
[root@scientific03 ~]# e2fsck -f /dev/loop1
[root@scientific03 ~]# resize2fs /dev/loop1
[root@scientific03 ~]# losetup -d /dev/loop1
```

Es necesario montar la imagen, usando el dispositivo loopback, en un directorio temporal para realizar cambios.

```
[root@scientific03 ~]# mkdir temporal
[root@scientific03 ~]# mount -o loop slc4.gpe.img temporal/
```

Posteriormente se cambia el directorio temporal como consola root, se procede a instalar el kernel domU en la imagen.

```
[root@scientific03 ~]# chroot temporal/
[root@scientific03 /]# rpm -ivh
http://xenbits.xensource.com/kernels/rhel4x/RPMS/i686/kernel-
xenU-2.6.9-42.0.3.EL.xs0.4.0.263.i686.rpm
```

Se ubica el kernel domU (vmlinuz e initrd) en el directorio /boot para efectos de arranque de la imagen a paravirtualizar.

```
[root@scientific03 ~]# cp
temporal/boot/vmlinuz-2.6.9-42.0.3.EL.xs0.4.0.263xenU
/boot/vmlinuz-2.6.9-42.0.3.EL.xs0.4.0.263xenU
[root@scientific03 ~]# cp
temporal/boot/initrd-2.6.9-42.0.3.EL.xs0.4.0.263xenU.img
/boot/initrd-2.6.9-42.0.3.EL.xs0.4.0.263xenU.img
```

Los archivos de configuración de Xen se encuentran ubicados en el directorio ***/etc/xen/***, este directorio puede contener archivos de configuración de ejemplos que guían el proceso de paravirtualización con Xen.

Esta configuración se compone de varias lineas que definen información del S.O a paravirtualizar información como:

- Ubicación y opciones del kernel con el cual arrancar el S.O domU.

```
ramdisk = '/boot/initrd-2.6.9-42.0.3.EL.xs0.4.0.263xenU.img'
kernel = '/boot/vmlinuz-2.6.9-42.0.3.EL.xs0.4.0.263xenU'
extra = 'fastboot'
root = '/dev/sda1 ro'
```

- Memoria en MegaBytes asignada para la máquina virtual.

  memory = 320

- Configuración de la red

```
ip = '192.168.109.132'
gateway = '192.168.109.1'
netmask = '255.255.255.0'
vif = []
```

- Ubicación de la imagen a usar como disco principal.

  disk = ['file:/root/slc4.gpe.img,sda1,w']

- Nombre de la máquina ante Xen y la red.

```
hostname = 'vscientifico3'
name = 'vscientifico3'
```

- Acciones a realizar en casos específicos.

```
on_poweroff='destroy'
on_reboot='restart'
on_crash='restart
```

después de crear el archivo de configuración se inicia la máquina virtual.

```
[root@scientific03 xen]# xm create -c /etc/xen/vscientific03iso
```

Se autentica en el sistema y se realizan los siguientes cambios:

- Deshabilitar SELinux en el archivo de configuración ***/etc/selinux/config***
- Asignación de password de root.
- Instalación de paquetes varios (openssh, vnc, wget, editores)
- Actualización de todos los paquetes por medio del comando yum update

Para la instalación de los servicios y componentes fundamentales del middleware gLite se usa la técnica LVM para el almacenamiento de los sistemas virtualizados.

Apagada la máquina virtual se crea un volumen lógico en el volumen de grupo vg.

El nuevo volumen lógico tiene un tamaño de 10Gb y de nombre vscientific03.

```
[root@scientific03 xen]# lvcreate -L 10G -n vscientific03 vg
[root@scientific03 xen]# mkfs.ext3 /dev/vg/vscientific03
```

Paso de información de la imagen al volumen lógico

```
[root@scientific03 xen]# cat /root/slc4.gpe.img > /dev/vg/vscientific03
```

Es necesario cambiar el tamaño del volumen lógico especificado (10GB) debido a que después de la copia, el volumen lógico queda con el tamaño de la imagen (2Gb)

```
[root@scientific03 xen]# e2fsck -f /dev/vg/vscientific03
[root@scientific03 xen]# resize2fs /dev/vg/vscientific03
```

Creación del volumen lógico asignado al swap.

```
[root@scientific03 xen]# lvcreate -L 2G -n vscientific03_swap vg
[root@scientific03 xen]# mkswap /dev/vg/vscientific03_swap
```

Cambio de dispositivo de almacenamiento en la configuración de la máquina a usar por xen.

```
disk = ['phy:vg/vscientific03,sda1,w','phy:vg/vscientific03_swap,sda2,w']
```

Arranque de la máquina paravirtualizada.

```
[root@scientific03 xen]# xm create /etc/xen/vscientific03
```

Verificación de las máquinas en ejecución

```
[root@scientific03 xen]# xm list
Name                              ID   Mem VCPUs      State   Time(s)
Domain-0                           0   738     2     r-----   1168.3
vscientific03                      1   256     1     -b----    183.0
```

12.3. Instalación y configuración de NTP

Para instalar NTP, se instala con el gestor de paquetes APT para RPM.

```
[root@vscientific02 etc]# apt-get install ntp
```

Parte de la salida del proceso...

```
Building Dependency Tree... Done
The following NEW packages will be installed:
  ntp
0 upgraded, 1 newly installed, 0 removed and 3 not upgraded.
```

```
Need to get 1236kB of archives.
After unpacking 2220kB of additional disk space will be used.
Get:1 http://www.dutchgrid.nl 3/i386/os ntp 4.1.2-5.el3 [1236kB]
70% [1 ntp 874104/1236kB 70%]
30.8kB/s 11s
```

La configuración de NTP se basa en el archivo ubicado en /etc/ntp.conf

```
[root@vscientific02 etc]# nano ntp.conf
```

Archivo: ***ntp.conf***

```
restrict 1.pool.ntp.org mask 255.255.255.255 nomodify notrap noquery
server 1.pool.ntp.org
```

después de guardar se reinicia el servidor ntp y se mueve el archivo que indica la zona usada por el sistema, se asigna la nueva zona, reinicio del servicio ntp y por ultimo se asigna la hora al reloj del hardware.

```
[root@vscientific02 etc]# mv localtime localtime.old
[root@vscientific02 etc]# ln -s /usr/share/zoneinfo/America/Bogota
/etc/localtime
[root@vscientific02 etc]# service ntpd restart
Shutting down ntpd:                                        [  OK  ]
ntpd: Synchronizing with time server:                      [  OK  ]
Starting ntpd:                                             [  OK  ]
[root@vscientific02 etc]# date
Sat May  3 11:26:33 COT 2008
[root@vscientific02 etc]# hwclock –systohc
```

12.4. JDK Y JPackage

Se debe realizar la instalación de JDK 1.5.0 o superior para realizar la instalación del middleware gLite, inicialmente se instala el programa jpackage-utils.

```
[root@vscientific02 etc]# apt-get install jpackage-utils
Fetched 50.3kB in 2s (17.9kB/s)
Committing changes...
Preparing...                ##########################################
[100%]
   1:jpackage-utils         ##########################################
[100%]
Done.
```

con YUM, se configuran la instalación de JDK para el middleware gLite.

```
[root@vscientific02 ~]# yum install xml-commons-jaxp-1.3-apis
```

```
Dependencies Resolved

================================================================================
 Package                 Arch       Version          Repository
Size
================================================================================
Installing:
 xml-commons-jaxp-1.3-apis  noarch     1.3.03-11jpp     jpackage17-
generic  167 k
Installing for dependencies:
 xml-commons             noarch     1.3.03-11jpp     jpackage17-generic
9.4 k

Transaction Summary
================================================================================
Install      2 Package(s)
Update       0 Package(s)
Remove       0 Package(s)
Total download size: 176 k
Is this ok [y/N]: y
```

Se descarga el archivo para rpm de JDK de la página oficial de SUN http://java.sun.com/javase/downloads/index_jdk5.jsp y se instala por medio de YUM.

```
[root@vscientific02 ~]# yum localinstall  jdk-1_5_0_14-linux-i586.rpm

Dependencies Resolved

================================================================================
 Package                 Arch       Version          Repository
Size
================================================================================
Installing:
 jdk                     i586       2000:1.5.0_15-fcs  jdk-1_5_0_14-
linux-i586.rpm   80 M

Transaction Summary
================================================================================
Install      1 Package(s)
Update       0 Package(s)
Remove       0 Package(s)
Total download size: 80 M
Is this ok [y/N]: y
```

```
Downloading Packages:
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing: jdk                          ##########################
[1/1]

Installed: jdk.i586 2000:1.5.0_14-fcs
Complete!
```

por ultimo se instala el paquete **java-1.5.0-sun-compat**

```
[root@vscientific02 ~]# yum install java-1.5.0-sun-compat
```

12.5. Instalación de herramientas necesarias para OpenCA

Instalación de Perl y módulos necesarios

```
openca:/# apt-get install libxml-perl libxml-regexp-perl libdbi-perl perl
perl-modules libldap2 libldap2-dev libdbd-mysql-perl libauthen-sasl-perl
libx500-dn-perl libcgi-session-perl libconvert-asn1-perl libdigest-md2-
perl libdigest-md4-perl libdigest-sha1-perl libio-socket-ssl-perl libio-
stringy-perl libmime-lite-perl libmime-perl libmailtools-perl libnet-
server-perl liburi-perl libxml-twig-perl libintl-perl libnet-ldap-perl
```

Instalación de OpenSSL

```
openca:/# apt-get install openssl
```

Instalación de Apache2

```
openca:/# apt-get install apache2
```

Instalación de MySQL

```
openca:/# apt-get install mysql-server
```

12.6. Instalación, configuración y puesta en marcha de OpenCA

Se descarga el software de OpenCA (http://www.openca.org/), posteriormente se instalan y configuran.

```
openca:/# cd /usr/src
openca:/usr/src/# tar xvfz openca-0.9.3-rc.tar.gz
openca:/usr/src/# cd openca-0.9.3-rc1
openca:/usr/src/openca-0.9.3-rc1# touch config_ra
openca:/usr/src/openca-0.9.3-rc1# touch config_ca
```

Los scripts creados necesitan permiso de ejecución.

```
openca:/usr/src/openca-0.9.3-rc1# chmod 755 config_ra
openca:/usr/src/openca-0.9.3-rc1# chmod 755 config_ca
```

Se edita el archivo config_ra y se introduce las lineas para configurar y compilar la *Registration Authority* (***RA***)

```
#!/bin/sh
make distclean
./configure --prefix=/usr/local/openca --with-web-host=localhost --with-
httpd-user=www-data        --with-httpd-group=www-data        --with-ext-
prefix=/usr/local/openca.0.9.3 --with-htdocs-fs-prefix=/var/www/ --with-
cgi-fs-prefix=/usr/lib/cgi-bin  --with-ca-organization="UIS"  --with-ca-
locality="Bucaramanga" --with-ca-country= "CO" --with-module-prefix=/usr/
local/openca/modules --enable-dbi --disable-db --disable-rbac --with-
hierarchy-level=ra --with-service-mail-account="root@loca lhost" --with-
db-type=mysql --with-db-name=openca --with-db-host=localhost --with-db-
port=3306 --with-db-user=openca --with-db-passwd="openca"
make
make install-ra
make install-pub
make install-node
```

Se realiza lo mismo con el script config_ca

```
#!/bin/sh
./configure --prefix=/usr/local/openca --with-web-host=localhost --with-
httpd-user=www-data        --with-httpd-group=www-data        --with-ext-
prefix=/usr/local/openca.0.9.3 --with-htdocs-fs-prefix=/var/www/ --with-
cgi-fs-prefix=/usr/lib/cgi-bin  --with-ca-organization="UIS"  --with-ca-
locality="Bucaramanga" --with-ca-country= "CO" --with-module-prefix=/usr/
local/openca/modules --enable-dbi --disable-db --disable-rbac --with-
hierarchy-level=ca --with-service-mail-account="root@loca lhost" --with-
dbi-type=mysql --with-dbi-name=openca --with-dbi-host=localhost --with-
dbi-port=3306 --with-dbi-user=openca --with-dbi-passwd="openca"
make
make install-ca
make install-node
```

Se graban los datos y se ejecutan ambos scripts.

```
openca:/usr/src/openca-0.9.3-rc1# ./config_ra
openca:/usr/src/openca-0.9.3-rc1# ./config_ca
```

Los módulos de perl serán descargados por el script de instalación de la OpenCA. Después se debe crear la base de datos donde se va a guardar la información de la CA y la RA, esta base de datos puede estar alojada en un servidor postgresql, mysql u oracle.

```
 openca:/# mysql -u root -p
      <password>
      create database openca;
      create database openra;
      grant all privileges on openca.* to openca@localhost identified by
"password";
      grant all privileges on openra.* to openra@localhost identified by
"password";
```

Uno de los aspectos más importantes es la revisión y modificación del fichero config.xml:

```
openca:/# cd /usr/local/openca/OpenCA/etc
openca:/usr/local/openca/OpenCA/etc# cp config.xml config.xml.original
openca:/usr/local/openca/OpenCA/etc# vim config.xml
```

Se revisan y se configuran las opciones generales, configuración de servidor web, base de datos, configuración de módulos, enlaces relativos y absolutos. La mayoría de los parámetros se encuentran por omisión configurados en el *configure*, sin embargo generalmente hay algunos parámetros a cambiar:

```
      <name>ca_organization</name>
      <value></value>
      <value>GridUIS</value>

      <name>ca_locality</name>
      <value></value>
      <value>Bucaramanga</value>

      <name>ca_country</name>
      <value></value>
      <value>CO</value>

      <name>service_mail_account</name>
      <value></value>
      <value>root@localhost</value>
```

Se modifican los archivos de ejemplo.

```
openca:/usr/local/openca/OpenCA/etc# ./configure_etc.sh
```

Se agrega la CA al inicio del sistema.

```
openca:/usr/local/openca/OpenCA/etc# cp openca_rc /etc/init.d/
openca:/usr/local/openca/OpenCA/etc# update-rc.d openca_rc defaults
```

Es necesario configurar apache para el entregar el certificado de la CA instalada, el primer paso es generando un certificado raiz con openssl para la CA.

```
openca:/usr/local/openca/OpenCA/etc# mkdir /etc/apache2/ssl
openca:/usr/local/openca/OpenCA/etc# cd /etc/apache2/ssl
openca:/etc/apache2/ssl# openssl req -x509 -newkey rsa:2048 -keyout
cakey.pem -days 3650 -out cacert.pem -nodes
```

Los archivos creados(cakey.pem y cacert.pem) contienen la llave privada y pública de la CA instalada. Es necesario configurar apache cargar los certificados correspondientes a la CA cuando se acceda vía web a la OpenCA.

```
openca:/etc/apache2/ssl# a2enmod ssl
openca:/etc/apache2/ssl# vim ../sites-enabled/ssl
```

El archivo ssl contiene la información del sitio web con soporte para SSL *(Ver Anexo 12.7).*

Se reinicia apache y se puede acceder a OpenCA por medio de la dirección web en un navegador:

- Para la Certificate Authority https://ipopenca/ca
- Para la Registrate Authority https://ipopenca/ra
- Acceso Público https://ipopenca/pub

Funcionamiento OpenCA

Se accede a https://ipopenca/ca con el login y password por omisión, se realizan los siguientes pasos:

```
General -> Initialization -> Initialize the Certification Authority
Initialize Database.
```

Genera el mensaje: *"The database was successfully initialized."*

```
Generate new CA secret key -> escoger preferencias -> OK
```

Se digita el passphrase para la llave secreta de la CA, después de generada se muestra la llave secreta.

```
Generate new CA Certificate Request (use generated secret key)->
```
se editan

los campos -> OK -> confirmar la siguiente página OK -> se el passphrase de la llave secreta de la CA.

Self Signed CA Certificate (from altready generated request) -> se escoge el tiempo de validez -> Muestra el certificado digital firmado.

Rebuild CA Chain

12.7. /etc/apache2/sites-enabled/openca

```
NameVirtualHost *:443
<VirtualHost *:443>
        SSLEngine On
        SSLOptions +StdEnvVars
        SSLCertificateFile /etc/apache2/ssl/cacert.pem
        SSLCertificateKeyFile /etc/apache2/ssl/cakey.pem
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/
        <Directory />
                Options FollowSymLinks
                AllowOverride None
        </Directory>
        <Directory /var/www/>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
                RedirectMatch ^/$ /apache2-default/
        </Directory>
       ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
        <Directory "/usr/lib/cgi-bin">
                AllowOverride None
                Options ExecCGI -MultiViews +SymLinksIfOwnerMatch
                Order allow,deny
                Allow from all
        </Directory>

        ErrorLog /var/log/apache2/error.log
       LogLevel warn
       CustomLog /var/log/apache2/access.log combined
        ServerSignature On
    Alias /doc/ "/usr/share/doc/"
    <Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 127.0.0.0/255.0.0.0 ::1/128
    </Directory>
</VirtualHost>
```

12.8. glite-voms-server.cfg.xml

```
<config xmlns:xi="http://www.w3.org/2001/XInclude">
<xi:include href="/opt/glite/etc/config/vo-list.cfg.xml" xpointer="" />
        <instance service="voms" iterate="volist">
                <parameters/>
       </instance>
       <parameters>
               <voms.db.type value="mysql"/>
                <voms.db.host value="localhost"/>
                <voms.admin.smtp.host value="voms-grid.uis.edu.co"/>
               <voms.mysql.admin.password value="-------"/>
                <glite.installer.verbose value="true"/>
               <glite.installer.checkcerts value="true"/>
<voms.proxy.timeout value="86400"/>
               <voms.admin.install value="true"/>
                <voms.admin.requestScheduler.disable value="true"/>
                <voms.admin.webRegistration.disable value="false"/>
<voms.mysql.admin.name value="root"/>
               <voms.db.mysql.port value="3306"/>
               <set.mysql.root.password value="true"/>
               <mysql.root.password value="---------"/>
                <voms.db.mysql.maxConnections value="500"/>
                <voms.db.oracle.port value="1521"/>
                <voms.db.min.connections value="1"/>
                <voms.db.max.connections value="20"/>
                <voms.db.startup.connections value="10"/>
               <voms.admin.oracle.connection.string value=""/>
                <voms.logrotate.period value="daily"/>
                <voms.logrotate.logNumber value="90"/>
<voms.db.mysql.library value="{GLITE_LOCATION}/lib/libvomsmysql.so"/>
<voms.db.oracle.library value="${GLITE_LOCATION}/lib/libvomsoracle.so"/>
<voms.db.oracle.instantclient.location
value="/usr/lib/oracle/10.2.0.1/client/"/>
        </parameters>
</config>
```

12.9. vo-list.cfg.xml

```
<config>
        <vo name="uis">
                <parameters>
                        <vo.name value="uis"/>

                        <voms.hostname value="voms-grid.uis.edu.co"/>

                        <voms.port.number value="15000"/>
```

```
                        <voms.cert.url value=""/>

                        <voms.cert.subj value=""/>

                        <voms.db.name value="voms_${vo.name}"/>

                        <voms.db.user.name value="mysql_uis"/>

                        <voms.db.user.password value="--------"/>

                        <vo.sgm.vo.role value=""/>

                        <pool.account.basename value=""/>

                        <pool.account.group value=""/>

                        <pool.account.number value=""/>

                        <voms.db.host value="localhost"/>

                        <voms.admin.smtp.host value="localhost"/>

                        <voms.admin.notification.e-mail
value="admin@ciencias.uis.edu.co"/>

                        <vo.sgm.user value="${vo.name}sgm"/>

                        <vo.sgm.group value="${vo.name}sgm"/>

                        <voms.admin.certificate
value="/root/adminvouiscert.pem"/>

                        <voms.admin.requestScheduler.disable
value="true"/>
 <voms.db.mysql.port value="3306"/>
                </parameters>
        </vo>
</config>
```

12.10. Resultado de la comprobación de gLite-VOMS

```
[root@voms-grid scripts]# ./glite-voms-server-config.py -c
Owner of file or directory /var/glite is root:root
Owner of file or directory /var/log/glite is root:root
Owner of file or directory /tmp/glite is root:root
....
....
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
             Main settings of gLite VOMS Server
```

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

  VOMS SERVER
     ------------------------------------------------------------
     [DB type]            mysql
     [VOMS-admin enabled] true
     [VOMS endpoint]      https://voms-grid.uis.edu.co:8443/vomses/
     ------------------------------------------------------------

  VOMS VO settings
     ------------------------------------------------------------

     [VO] uis
         ============================================================
         [VOMS VO endpoint]         https://voms-
grid.uis.edu.co:8443/voms/uis
         [VOMS Hostname]            voms-grid.uis.edu.co
         [VOMS Port]                15000
         [VOMS Proxy Timeout]       86400 s
         [VOMS logrotation period]  daily
         [VOMS logrotation number]  90
         ----------------------------------------------------
         [VOMS-admin SMTP]    localhost
         [VOMS-admin mail]    admin@grid.uis.edu.co
         [VOMS-admin cert]    /root/adminvouiscert.pem
         [VOMS-admin disable requestScheduler] true
         [VOMS-admin disable web registration] false
         ----------------------------------------------------
         [DB Name]            voms_uis
         [DB UserName]        mysql_uis
         [DB UserPassword]    ----------
         [DB Host]            localhost
         [DB AdminName]       root
         [DB AdminPassword]   ----------
         [DB Port]            3306
         ============================================================
```

12.11. site-info.def

```
YAIM_LOGGING_LEVEL=INFO

CA_REPOSITORY="rpm http://linuxsoft.cern.ch/ LCG-CAs/current production"

MY_DOMAIN=uis.edu.co

INSTALL_ROOT=/opt

CE_HOST=ce-grid.$MY_DOMAIN

RB_HOST=wms-grid.$MY_DOMAIN

WMS_HOST=wms-grid.$MY_DOMAIN

LB_HOST="wms-grid.$MY_DOMAIN:9000"
```

```
PX_HOST=wms-grid.$MY_DOMAIN
MON_HOST=mon-grid.$MY_DOMAIN
REG_HOST=mon-grid.$MY_DOMAIN
FTS_HOST=ftm-grid.$MY_DOMAIN
LFC_HOST=lfc-grid.$MY_DOMAIN
BDII_HOST=bdii-grid.$MY_DOMAIN
SITE_BDII_HOST=bdii-grid.$MY_DOMAIN
GRID_TRUSTED_BROKERS="'/C=CO/O=UIS/OU=Internet/CN=wms-grid.uis.edu.co'"
USERS_CONF=/etc/glite/yaim/users.conf
GROUPS_CONF=/etc/glite/yaim/groups.conf
FUNCTIONS_DIR=/opt/glite/yaim/functions
BATCH_SERVER=$CE_HOST
JOB_MANAGER=lcgpbs
CE_BATCH_SYS=torque
BATCH_BIN_DIR=/usr/bin
BATCH_VERSION=torque-1.0.1b
BATCH_LOG_DIR=/var/spool/pbs
SITE_EMAIL=admin@grid.uis.edu.co
SITE_CRON_EMAIL=$SITE_EMAIL   # not yet used will appear in a later
release
SITE_SUPPORT_EMAIL=$SITE_EMAIL
SITE_NAME=griduis
SITE_LOC="Bucaramanga, Colombia"
SITE_LAT=7.08 # -90 to 90 degrees
SITE_LONG=73.08 # -180 to 180 degrees
SITE_WEB="http://grid.uis.edu.co"
SITE_TIER="TIER 2"
SITE_SUPPORT_SITE="grid.uis.edu.co
MYSQL_PASSWORD=”----------”
JAVA_LOCATION="/usr/java/jdk1.5.0_14"
DPM_HOST="se-grid.$MY_DOMAIN"
CLASSIC_STORAGE_DIR="/storage"
VOS="uis"
QUEUES=${VOS}
```

```
UIS_GROUP_ENABLE="uis"
VO_SW_DIR=/opt/exp_soft
EDG_WL_SCRATCH=""
VO_UIS_SW_DIR=$VO_SW_DIR/uis
VO_UIS_DEFAULT_SE=$DPM_HOST
VO_UIS_STORAGE_DIR=$CLASSIC_STORAGE_DIR/uis
VO_UIS_VOMS_SERVERS='vomss://voms-grid.uis.edu.co:8443/voms/uis?/uis/'
VO_UIS_VOMSES="'uis               voms-grid.uis.edu.co              15000
/C=CO/O=UIS/OU=Internet/CN=voms-grid.uis.edu.co'"
CE_DATADIR=""
CE_CPU_MODEL=PIV
CE_CPU_VENDOR=intel
CE_CPU_SPEED=3200
CE_OS="Scientific Linux CERN"
CE_OS_RELEASE=4.6
CE_OS_VERSION="SLC"
CE_OS_ARCH=i686
CE_MINPHYSMEM=256
CE_MINVIRTMEM=256
CE_PHYSCPU=1
CE_LOGCPU=1
CE_SMPSIZE=1
CE_SI00=381
CE_SF00=0
CE_OUTBOUNDIP=TRUE
CE_INBOUNDIP=FALSE
CE_RUNTIMEENV="
    LCG-2
   LCG-2_1_0
    LCG-2_1_1
    LCG-2_2_0
    LCG-2_3_0
    LCG-2_3_1
    LCG-2_4_0
```

```
    LCG-2_5_0
    LCG-2_6_0
    LCG-2_7_0
    GLITE-3_0_0
    GLITE-3_1_0
    R-GMA "
SE_LIST=DPM_HOST
APEL_DB_PASSWORD=--------
WN_LIST=/etc/glite/yaim/wn-list.conf
```

12.12. bdii.conf

```
BDII_PORT_READ=2170
BDII_PORTS_WRITE="2171 2172"
BDII_USER=edguser
BDII_BIND=mds-vo-name=resource,o=grid
BDII_PASSWD=bdii4322
BDII_SEARCH_FILTER='*'
BDII_SEARCH_TIMEOUT=180
BDII_BREATHE_TIME=60
BDII_AUTO_UPDATE=yes
BDII_AUTO_MODIFY=yes
BDII_MODIFY_DN=no
BDII_IS_CACHE=yes
BDII_UPDATE_URL=http://bdii-grid.uis.edu.co/bdii-update.conf
BDII_DIR=/opt/bdii
BDII_VAR_DIR=/opt/bdii/var
BDII_DEFAULT_LDIF=/opt/bdii/etc/default.ldif
SLAPD=/usr/sbin/slapd
SLAPD_DEBUG_LEVEL=4
SLAPADD=/usr/sbin/slapadd
```

12.13. Máquinas Virtualizadas en Equipos

Equipo Host	Equipos Guests	IP Interna	IP Externa
192.168.109.141	Workload Manager System	192.168.109.154	200.21.228.157
	Logging and Bookkeeping		
	Proxy Certificates		
192.168.109.142	Worker Node	192.168.109.158	-
192.168.109.143	BDII	192.168.109.155	200.21.228.155
192.168.109.144	Computing Element	192.168.109.156	200.21.228.156
	User Intreface	192.168.109.152	200.21.228.158
192.168.109.145	OpenCA	192.168.109.150	200.21.228.160
	Virtual Organization Management Service	192.168.109.151	200.21.228.159

Tabla 4: Equipos y direcciones ips.

12.14. glite_wms_wmproxy.gacl

```
<gacl version="0.0.1">
  <entry>
    <voms>
      <fqan>uis</fqan>
    </voms>
    <allow>
      <exec/>
    </allow>
  </entry>
</gacl>
```

12.15. wn-list.conf

```
wn-grid.uis.edu.co
wn.utp.ac.pa
wn.ula.ve
```

12.16. edg-pbs-knownhosts.conf

```
NODES = ce-grid.uis.edu.co wn-grid.uis.edu.co
PBSBIN = /usr/bin
KEYTYPES = rsa1,rsa,dsa
KNOWNHOSTS = /etc/ssh/ssh_known_hosts
```

12.17. edg-pbs-shostsequiv.conf

```
NODES       =  ce-grid.uis.edu.co wn-grid.uis.edu.co
PBSBIN      = /usr/bin
```

12.18. Programa1.m

```
clc;
clear all;
a=rand(1000,1000);
b=rand(1000,1000);
for j=1:5;tic;for i=1:3;c=a*b;end;t=toc;mf(j)=1999*3/t;end;disp(max(mf))
```

12.19. tiempos.jdl

```
# Type = "Job";
JobType = "Normal";
Executable = "/usr/bin/octave";
StdOutput = "tiempos.out"
StdError = "tiempos.err";
InputSandbox = {"tiempos.m"};
OutputSandbox = {"tiempos.err", "tiempos.out", "tiempos.txt"};
Arguments = "tiempos.m";
ShallowRetryCount = 1;
RetryCount = 0;
```

13. BIBLIOGRAFÍA

13.1. LIBROS Y ARTICULOS

1. BART Jacob, BROWN Michael, FUKUI Kentaro, TRIVEDI Nihar (Dec, 2005). Introduction to grid computing. 1era ed. Riverton, NJ, USA. IBM Corp. 2005. ISBN:0738494003.

2. FOSTER Ian, KESSELMAN Carl. Computational grids. San Francisco, CA, USA. 1999. ISBN: 1-55860-475-8.

3. FOSTER Ian, KESSELMAN Carl, TUECKE Steven. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. Thousand Oaks, CA, USA. 2001. ISSN: 1094-3420.

4. FOSTER Ian. What is the Grid? A Three Point Checklist. Argonne National Laboratory & University of Chicago. July 20, 2002.

5. KAMATH John-Paul. From web to grid, Cern's huge data demands forge a path for business. Computer Weekly,12. Retrieved April 7, 2008, from ProQuest Computing database. (Document ID: 1456394681).

6. MARKOFF John. Supercomputing's New Idea Is Old One. New York Times (Late Edition (east Coast)), p. C.1. Retrieved April 8, 2008, from Banking Information Source database. (Document ID: 378956161).

7. NOBSA Julián, RODRIGUEZ Iván, DÍAZ Gilberto. Estudio Comparativo de las Técnicas de Paravirtualización y Virtualización a nivel de Sistema Operativo. Bucaramanga, Colombia. Agosto 2008.

8. SAMPIERI Hernández; COLLADO C. Fernández; BAPTISTA Lucio, P. Metodología de la Investigación. 2da ed. Santa Fé de Bogotá, Colombia. Mc Graw Hill. 1998 (Cap.4 y cap5).

13.2. ENLACES

- ✔ CLARA - Cooperación Latino Americana de Redes Avanzadas (Mayo, 2008). Obtenido de http://www.redclara.net/

- EELA - E-Infrastructure shared between Europe and Latin America. (Abril, 2008). Obtenido de http://www.eu-eela.org/
- gLite 3.1 User Guide (Marzo, 2008). Obtenido de https://edms.cern.ch/file/722398//gLite-3-UserGuide.html
- Grid Colombia Web Site (Mayo, 2008). Obtenido de http://urania.udea.edu.co/grid-colombia/index.php
- GridPP - UK Computing for Particle Physics (Abril, 2008). Obtenido de http://www.gridpp.ac.uk/
- Introduction to Parallel Computing (Marzo, 2008). Obtenido de https://computing.llnl.gov/tutorials/parallel_comp
- List of distributed computing projects. (2008, Abril). En Wikipedia, The Free Encyclopedia. Obtenido de http://en.wikipedia.org/w/index.php?title=List_of_distributed_computing_projects&oldid=251705657
- OpenGrid Forum (Marzo, 2008). Obtenido de http://www.ggf.org/
- The globus alliance (Mayo, 2008). Obtenido de http://www.globus.org
- Virtualization and grid computing. (2008, Abril). En Virtualization and grid computing on distributed computing, VMs, Globus, Xen, Nimbus, and other technology. Obtenido de http://www.gridvm.org/
- Welcome to the Globus Toolkit Homepage (Abril,2008). Obtenido de http://www.globus.org/toolkit
- Welcome to Grid-Workshop (Junio,2008). en Setting up the various grid components with yaim 4.0. Obtenido de http://trac.healthgrid.org/grid-workshop/wiki