GRID CON PARÁMETROS DTN
CASO DE ESTUDIO: COMPUTADORES PARA EDUCAR – UNIVERSIDAD DEL CAUCA

Monografía para optar al titulo de Magíster en Ingeniería, área Telemática

Ing. Juan Carlos Arteaga Córdoba

Director: Msc. Ing. Héctor Fabio Jaramillo Ordóñez

Universidad Del Cauca
Instituto de Postgrados en Electrónica y Telecomunicaciones
Facultad de Electrónica Y Telecomunicaciones
Departamento De Telemática
Grupo de Ingeniería Telemática
Línea de Investigación en Servicios Avanzados de Telecomunicaciones

Popayán, Julio de 2009

Primordialmente a Dios por su infinito amor y compañía en todo momento y lugar.
A mis padres por su gran amor, apoyo y confianza en este largo proceso.
A mis hermanos Giovanny y Rolando por su respaldo, cariño y comprensión.
A Lina Maria por su hermosa compañía y ternura.

Agradecimiento

De manera muy especial agradezco a mi tutor Héctor Fabio Jaramillo Ordóñez, por sus valiosas contribuciones, aportes y orientaciones en todo el proceso de formación en la maestría, a todos los amigos y compañeros del programa Computadores para Educar en especial a Jorge J. Moreno y Estivenson Armero por su apoyo, confianza y comprensión y a los compañeros del grupo de Ingeniería Telemática y del IPET.

Igualmente quiero extender mis agradecimientos a los miembros del Laboratorio de Sistemas Complejos de la Universidad de Buenos Aires especialmente a Guillermo Marshall quien me ha brindado su confianza y apoyo en este y otros procesos fundamentales para mi crecimiento profesional, por último a mis compañeros y amigos de la universidad, especialmente a Cristhian Figueroa, Juan Pablo Amaya, Wilmar Campo, Víctor Mondragón, Armando Ordóñez, Franco Urbano, Francisco Martínez, Jorge Figueroa, Javier Hurtado, Natalia Maya, y Oscar Caicedo.

Tabla de Contenido

1 Capítulo 1. Introducción 1
1.1 Justificación 1
1.2 Propuesta 1
1.3 Contenido del documento 2
2 Capítulo 2. Proyectos TIC 3
2.1 TIC en zonas rurales. 3
2.2 Proyectos Regionales 3
2.3 Las TIC en Colombia 3
2.3.1 Compartel 3
2.3.2 CERES 4
2.3.3 Computadores para Educar 4
2.3.4 Necesidades relacionadas con los programas gubernamentales 4
3 Capítulo 3. Tecnologías y Herramientas de Soporte 5
3.1 La computación paralela y distribuida 5
3.2 Tecnología Cluster 5
3.2.1 Concepto 5
3.2.2 Composición de un Cluster 5
3.2.3 Clasificación de los *Cluster* 6
3.2.4 Virtualización 7
3.3 Herramientas *Cluster* 8
3.3.1 Mosix 8
3.3.2 OpenMosix 9
3.3.3 Otras herramientas Cluster 9
3.3.4 Herramientas Cluster Seleccionadas 11
3.4 Tecnología Grid 12
3.4.1 Concepto 12
3.4.2 Aplicación de Grid 12
3.4.3 Características de Grid 13
3.5 Herramientas Grid 13
3.5.1 Unicore 13
3.5.2 Legion 13
3.5.3 Gridbus 14
3.5.4 Globus Toolkit 14
3.5.5 Comparación de herramientas Grid 20
3.6 Tecnología DTN (Delay/ Disrupt Tolerant Networking/ Red Tolerante a Retardos/ Interrupciones) 22
3.6.1 Disponibilidad y confiabilidad en redes de telecomunicaciones 22
3.6.2 Confiabilidad y disponibilidad en capas superiores 23
3.6.3 Casos de Utilización de Conceptos DTN 23
3.7 Módulos de Globus Toolkit adaptables para operación basada en DTN 24
3.7.1 Servicio RFT (*Reliable File Transfer*) 24
3.7.2 GRAM (*Grid Resource Allocation and Management*) 24
3.8 Tecnología Servicios Web 25
3.8.1 Composición de Servicios Web 25
3.8.2 Orquestación de Servicios Web 25
3.9 Herramientas para Servicios Web 25
3.9.1 Herramientas para crear y desplegar los servicios Web sobre la *Grid*. 25
3.9.2 Herramientas para ejecutar los servicios Web sobre la *Grid*. 26
3.9.3 Herramientas para consumir los servicios Web. 26
3.10 Proyectos relacionados y casos de implementación. 28

3.10.1 Casos de implementación 30
4 Capítulo 4. Diseño del *Cluster* y la Grid – DTN con el contenedor de servicios Web 32
4.1 Diseño de un *Cluster* de computadores 32
4.1.1 Componentes hardware para un *Cluster* 32
4.1.2 Hardware del nodo del *Cluster* 32
4.1.3 Tecnologías de red 33
4.1.4 Ubicación del *Cluster* 33
4.1.5 Refrigeración del sistema 33
4.1.6 Fuentes de alimentación 34
4.2 Diseño y construcción de una *Grid* 35
4.3 Arquitectura del Sistema 37
5 Capítulo 5. Ambientación Experimental 40
5.1 Planteamiento Experimental 40
5.2 Descripción de la capa *Cluster* experimental 40
5.2.1 Software Empleado 42
5.3 Descripción de la Capa *Grid* experimental 42
5.3.1 Infraestructura de la capa Grid 43
5.4 Pruebas ejecución 46
5.4.1 Diagrama de Casos de Uso Proyectos Institucionales 47
5.4.2 Diagrama Entidad Relación 49
5.4.3 Pruebas de acceso a la Base de Datos 49
5.5 Propuesta de Arquitectura DTN 50
5.5.1 Adaptaciones en Globus 50
5.5.2 Almacenamiento y reenvío de Unidades de Datos 51
5.5.3 Nodos, Identificadores y Custodia 51
5.5.4 Encaminamiento 52
6 Capítulo 6. Conclusiones y Recomendaciones 53
6.1 Conclusiones 53
6.2 Trabajo futuro 53
7 Referencias 55

Índice De Tablas

Tabla 3-1 Comparación de middleware Grid.....21
Tabla 5-1 Caso de Uso iniciarSesion 47
Tabla 5-2 Caso de Uso gestionarUsuarios 48
Tabla 5-3 Caso de Uso gestionarProyectos 48
Tabla 5-4 Caso de Uso gestionarContenidoProyectos.....48
Tabla 5-5 Caso de Uso verProyectos 48
Tabla 5-6 Prueba de consulta a la Base de Datos de Proyectos Institucionales con y sin *Cluster*.....50
Tabla 5-7 Prueba de escritura a la Base de Datos de Proyectos Institucionales con y sin *Cluster* 50

Índice De Figuras

Figura 3-1 Componentes Globus Toolkit 4.0 15
Figura 3-2 Componentes de Seguridad 16
Figura 3-3 Gestión de datos 17
Figura 3-4 Ejemplo de asociación entre un nombre de archivo lógico y tres réplicas en diferentes. 17
Figura 3-5 Gestión de ejecución 18
Figura 3-6 Servicio de información 19
Figura 3-7 Componente común de tiempo de ejecución 19
Figura 3-8 Comparación de Unicore, Globus, Legion y Gridbus 20
Figura 3-9 Prueba del servicio Web. 27
Figura 3-10 Prueba del servicio Web. 27
Figura 3-11 Vista de la aplicación en un móvil 28
Figura 4-1 Configuraciones típicas de Grid 35
Figura 4-2 Estructura en niveles de un entorno de *Grid* 36
Figura 4-3 Capa *Cluster* 37
Figura 4-4 Arquitectura por capas 39
Figura 5-1 Sistema doble capa (*Cluster* y *Grid*) 40
Figura 5-2 Equipos utilizados en la implementación del Cluster 41
Figura 5-3 Enrutador 42
Figura 5-4 Infraestructura para el prototipo de prueba 43
Figura 5-5 Emulador Sony Ericsson P900 invocando el servicio Web de Proyectos Institucionales 45
Figura 5-6.Vistas de la aplicación en un Sony Ericsson P1i 46
Figura 5-7 Diagrama de Casos de Uso Proyectos Institucionales 47
Figura 5-8 Diagrama entidad relación 49

1 Capítulo 1. Introducción

1.1 Justificación

La evolución de la tecnología, las redes de telecomunicaciones y la Internet han impulsado el desarrollo de aplicaciones multimedia, multiparte, distribuidas e interactivas que demandan una gran capacidad de procesamiento.

Muchos equipos existentes pueden quedar prematuramente obsoletos y sin cumplir su ciclo de vida útil, generando ineficiencias en costos y desperdicios con impacto ambiental negativo.

La ejecución de aplicaciones educativas complejas, software y sistemas de información multimedia institucional pueden realizarse sobre los equipos de bajas prestaciones mediante la distribución de requerimientos de procesamiento en una red de computadores de acuerdo a las capacidades de procesamiento de los equipos que la componen.

Las deficiencias en la conectividad de las localidades objeto del programa, determinadas por el tiempo medio entre fallas (*Mean Time Between Failures* MTBF), la rata de errores (*Bit Error Rate* BER) y la variación de retardo en las conexiones limitan la confiabilidad de las aplicaciones. Por lo tanto adicionalmente a la distribución del procesamiento se requiere de mecanismos que permitan incrementar la confiabilidad de las aplicaciones en condiciones de baja disponibilidad de las interconexiones.

Las aplicaciones y el middleware que las soporta deben contar con características de gestión ubicuas dada la escasez de personal técnico calificado y las condiciones de orden público que limitan los desplazamientos hacia las localidades rurales.

El presente proyecto propone una solución para facilitar y mejorar la provisión de servicios en computadores de bajas prestaciones y conectividad limitada, de tal forma que sea posible el uso de aplicaciones para el intercambio, publicación y gestión de información educativa que requieren mayor capacidad de procesamiento y almacenamiento, manteniendo el nivel de calidad de servicio a pesar de las limitadas condiciones del equipo y de conectividad a redes.

Adicionalmente a la reutilización de los equipos existentes, sin incurrir en costos de actualización o reemplazo, se impulsa la posibilidad de implementar nuevos y mejores servicios Web.

1.2 Propuesta

La propuesta está construida en la integración de las tecnologías *Cluster* (Yañez, 2004), mallas de computadores (*Grid*) (Mondardini, 2009), redes tolerantes a fallas e interrupciones DTN (*Delay/Disrupt Tolerant Networking*) (DTNRG, 2008) y servicios Web (Lin, 2006). Estas tecnologías y los conceptos de computación distribuida, computación paralela permiten aprovechar de manera eficiente los recursos computacionales existentes.

Las tecnologías *Cluster* y *Grid* se utilizan para ejecutar procesos con demanda intensiva de procesamiento. DTN permiten mejorar la confiabilidad de aplicaciones en entornos de red de baja disponibilidad. Los Servicios Web constituyen componentes software que extienden la operación de la Web desde una fuente de información distribuida a una fuente de servicios distribuidos (Lin, 2006), fundamentada en la interoperabilidad de aplicaciones Web (W3C, 2005).

La integración se plantea sobre una arquitectura de doble capa donde los computadores de un centro informático estarán organizados en *Cluster*, para la optimización de recursos. Los *Cluster* a su vez estarán interconectados entre sí mediante una *Grid* computacional. De esta forma las tecnologías *Cluster*, *Grid*, DTN y Servicios Web facilitan el óptimo uso de los recursos de cómputo, la comunicación con tolerancia a retardos, la compartición de aplicaciones y creación de servicios Web confiables.

La metodología a seguir para el desarrollo del proyecto está basada en el "Modelo Integral para un Profesional en Ingeniería" (Serrano, 2003) enfocada especialmente en dos de sus componentes:

- Modelo de Investigación Documental, para la generación de la base conceptual.
- Modelo para Construcción de Soluciones, para el proceso de desarrollo software.

1.3 Contenido del documento

La organización del documento es la siguiente:

En el capítulo 1 titulado *Introducción* contiene una breve introducción del trabajo desarrollado.

En el capítulo 2 titulado *Proyectos TIC* nombra los proyectos TIC y el fin de los mismos en diferentes entornos.

En el capítulo 3 titulado *Tecnologías y Herramientas Soporte* contiene la descripción y estado del arte de Tecnologías *Cluster*, *Grid*, DTN y Servicios Web, sus herramientas y los proyectos relacionados y casos de implementación.

En el capítulo 4 titulado *Diseño del cluster y la Grid-DTN con el contenedor de servicios Web* contiene los principales pasos que se deben tener en cuenta en el diseño y construcción de un *Cluster*, una *Grid* y un servicio Web y la descripción de la arquitectura del sistema.

En el capítulo 5 titulado *Ambientación Experimental* contiene a descripción del proceso de diseño para las diferentes capas, las pruebas de ejecución y la propuesta de la arquitectura DTN.

En el capítulo 6 titulado *Conclusiones y Recomendaciones* contiene el análisis de los resultados del trabajo realizado y se genera un conjunto de recomendaciones importantes para el desarrollo de trabajos futuros.

Anexo A. Ofrece una descripción detallada del proceso de Instalación y puesta a punto de la herramienta de creación de *Cluster* computacionales más utilizados por la comunidad de desarrolladores denominada openMosix.

Anexo B. Ofrece una descripción detallada del proceso de Instalación y puesta a punto de una herramienta de virtualización para Linux adaptado al sistema coLinux permitiendo así que openMosix se ejecute en nodos que tienen instalado el sistema operativo Windows, dicha herramienta es denominada cosMos.

Anexo C. Ofrece una descripción detallada del proceso de Instalación y puesta a punto de Globus Toolkit, herramienta para la creación de la capa *Grid*.

Anexo D. Ofrece una descripción del manejo de la herramienta *Introduce* mediante un ejemplo, para la creación y despliegue de servicios Web en *Grid*.

2 Capítulo 2. Proyectos TIC

2.1 TIC en zonas rurales.

Las Tecnologías de la Información y las Comunicaciones (TIC), son el conjunto de herramientas, equipos, programas informáticos, aplicaciones, redes y medios, que permiten la compilación, procesamiento, almacenamiento y transmisión de información como: voz, datos, texto, video e imágenes (Plan-Tic, 2008).

Uno de los beneficios más destacados de la introducción de las TIC es la reducción en la desigualdad y la mejora en la calidad de vida. Estas tecnologías tienen un importante impacto en la competitividad y en el desarrollo económico y social de los países. Estos beneficios sólo pueden convertirse en resultados concretos a medida que la sociedad se apropie de estas tecnologías y las haga parte de su desempeño cotidiano (Shakeel, 2001).

2.2 Proyectos Regionales

Existen también iniciativas internacionales que fomentan implementaciones de infraestructura en áreas rurales y remotas, tal es el caso del proyecto EHAS (Enlace Hispanoamericano de Salud) (EHAS, 2009) y el proyecto LINK ALL (*Local Communities Insertion* para América Latina) (LINK-ALL, 2009) ambos fundados por el programa @lis (Portal, 2008) de la Unión Europea. EHAS es un proyecto cuyo objetivo es mejorar el sistema público del cuidado de la salud en áreas rurales de Latinoamérica a través de tecnologías de bajo costo y servicios de acceso a información. El proyecto LINK ALL de inclusión digital promovió en la comunidad rural de Latinoamérica el desarrollo sostenible, el acceso al mercado global y fomentó su colaboración e integración con las TIC.

E-LANE (*European Latin American New Education*) (E-LANE, 2008) fue otro proyecto fundado por la Unión Europea. Su principal objetivo fue promover un entorno de aprendizaje integrado, ambos con un nivel de entrenamiento académico y no académico. El proyecto creó una plataforma *e-learning* de software abierto para uso e integración de aplicaciones existentes sólidas, diseñando una metodología de enseñanza novedosa orientada hacia esta plataforma e integrando contenido de importantes instituciones educativas de Europa y Latinoamérica, con el propósito de proveer un excelente material a bajo costo.

2.3 Las TIC en Colombia

El acceso a la información y el conocimiento a través de las TIC se ha constituido en uno de los principales factores para medir el nivel de desarrollo de una comunidad y en este sentido el Gobierno Nacional Colombiano ha emprendido diferentes iniciativas entre las cuales se destacan los programas Compartel (Compartel, 2009), CERES (CERES, 2009), y Computadores para Educar (CPE, 2004). Los objetivos de estas iniciativas son: el impulso a la educación, la atención de la salud, el desarrollo de contenidos culturales, la preservación de lenguas locales, y la promoción de iniciativas económicas para un desarrollo sostenible.

2.3.1 Compartel

Desde 1999, el gobierno nacional ha implementado programas de servicios y acceso universal enfocados en el desarrollo de infraestructura de comunicaciones para la provisión de telefonía y servicios Internet. Entre estas iniciativas se tienen el programa Compartel (Compartel, 2009) el cual tiene cinco subprogramas:

- **Telefonía rural:** provee servicios de telefonía en áreas donde ésta no se encuentra.
- **Tele-centros:** pone a disposición servicios de computadores, acceso a Internet y telefonía en diferentes municipios del país.

- **Conectividad:** provee acceso a Internet para instituciones públicas tales como colegios, bibliotecas, hospitales, alcaldías y organizaciones del gobierno en zonas rurales y de difícil acceso.
- **Expansión y reemplazo:** se enfoca en el mejoramiento e incremento de la infraestructura en áreas con altos niveles de servicios básicos insatisfechos.
- **Programas comunitarios:** fomenta la creación de estaciones de radio con contenido de interés público que soporte estrategias de comunicación y fortalecimiento de la cultura indígena.

Se espera que 22.406 instituciones públicas tengan acceso a Internet para el 2010, también se espera diseñar diferentes estrategias para mejorar la conectividad de 8.886 colegios públicos y transformar estas instituciones en centros de acceso comunitario. Así el país tendría 10.000 centros de acceso comunitario a Internet.

2.3.2 CERES

Los CERES (Centros Regionales de Educación Superior) (CERES, 2009) apoyan la educación superior en áreas rurales o remotas, o áreas urbanas menos favorecidas, y fomentan el desarrollo humano, económico y social. Cada CERES es el resultado de una alianza estratégica entre el gobierno nacional y regional, la sociedad civil, el sector privado e instituciones de educación superior. Ofrecen programas pertinentes y necesarios de alta calidad y que puedan ser tomados a distancia. Para lograr este objetivo, el gobierno nacional provee recursos para financiar y crear salas de computadores con acceso a Internet, suministrando todos los equipos requeridos.

2.3.3 Computadores para Educar

Uno de los programas auto sostenibles implementados por el gobierno colombiano, es el Programa Computadores para Educar (CPE, 2004), este nació de un esfuerzo conjunto entre el gobierno nacional y la empresa privada. Este programa es liderado por la Presidencia de la República, con la participación del Ministerio de Comunicaciones, el Ministerio de Educación, el Gobierno de Canadá, el SENA y varios socios de la empresa privada.

Es un programa de reciclaje tecnológico, cuyo objetivo es brindar acceso a las tecnologías de información y comunicaciones a instituciones educativas públicas del país, mediante el reacondicionamiento, ensamble y mantenimiento de equipos, y promover su uso y aprovechamiento significativo en los procesos educativos, a través de la implementación de estrategias de acompañamiento educativo y apropiación de TIC. Se espera beneficiar con el programas el 45% de escuelas públicas del país a finales del 2010, y tener un computador por cada 20 estudiantes, ya que actualmente se tiene una relación de un computador por cada 48 estudiantes.

Las iniciativas existentes en Colombia pueden no ser suficientes para terminar con la brecha digital, pero, si se optimizan sus procesos de funcionamiento, éstas iniciativas lograrán su objetivo en menos tiempo.

2.3.4 Necesidades relacionadas con los programas gubernamentales

Los recursos de cómputo y conectividad en las localidades objeto de los programas CPE y Compartel son limitados, y una actualización de equipos resultaría muy costosa y compleja, especialmente por cuestiones de compatibilidad de interfaces y discontinuidad comercial de componentes. De igual forma la introducción de nuevos enlaces de transmisión terrestre resulta económicamente inviable.

El éxito en la implementación de una tecnología depende del escenario y de los casos de uso de los que sea objeto. Específicamente las tecnologías *Cluster*, *Grid*, DTN y servicios Web se ajustan muy bien al contexto de países en desarrollo, donde existen recursos insuficientes y circunstancias adversas que limitan la implementación de otro tipo de estrategias y tecnologías (computadores nuevos, equipos de red de última tecnología, grandes anchos de banda) para que las TIC cumplan su objetivo.

3 Capítulo 3. Tecnologías y Herramientas de Soporte

3.1 La computación paralela y distribuida

La computación paralela se orienta esencialmente a resolver una tarea empleando múltiples procesadores de forma simultánea reduciendo el tiempo de ejecución (Universidad, 2008). La computación distribuida por otro lado se basa en la cooperación y compartición de recursos de computadoras interconectadas entre sí. La computación distribuida busca resolver grandes problemas dividiéndolos en pequeñas partes y entregando éstas a muchos computadores para obtener una solución combinando los resultados (Pearson, 2009).

Unas de las grandes motivaciones para la computación paralela y distribuida son el incremento del rendimiento en el procesamiento mediante la utilización de recursos compartidos y el balanceo de carga. Esto puede lograrse mediante las tecnologías *Cluster* y *Grid computing.*

La diferencia entre *Cluster* y *Grid* está en la forma en que se administran los recursos. En el caso de los *Cluster*, la ubicación de los recursos se hace mediante un administrador de recursos centralizado y todos los nodos trabajan juntos de manera cooperativa como un único recurso. En el caso de las *Grid*, cada nodo tiene su propio administrador de recursos y no pretende presentarse como una vista de un sistema único. A continuación se verá con más detalle cada una de estas tecnologías mencionadas.

3.2 Tecnología Cluster

3.2.1 Concepto

Un Cluster "es una arquitectura de computación paralela basada en la unión de máquinas independientes cooperativas integradas por medio de redes de interconexión para proveer un sistema coordinado, capaz de procesar una carga". El funcionamiento de los *Cluster* tiene que ver con el uso del poder de procesamiento de las máquinas que se encuentren ociosas (Catalán, 2004).

Un *Cluster* se divide en dos partes: el hardware de interconexión de los elementos del *Cluster* y el software, este último compuesto por un sistema operativo adaptado (por ejemplo GNU-Linux con un *kernel* modificado), compiladores especiales y aplicaciones.

Para programar un *Cluster* se puede utilizar un lenguaje de programación secuencial con una biblioteca de programación en paralelo. Se encuentran disponibles para un conjunto limitado de lenguajes de programación, como C, C++ y *Fortran*. Para que todos estos tipos de *Cluster* funcionen correctamente deben ser diseñados para entornos de programación y ejecución específicos.

Los *Cluster* han ido sustituyendo a los supercomputadores multiprocesamiento en múltiples entornos: centros de investigación, servidores Web, comercio electrónico, bases de datos de alto, entre otros.

3.2.2 Composición de un Cluster

Un Cluster está compuesto por (Enriquez, 2007):

- **Nodos:** La unidad fundamental de un Cluster es una computadora simple, también denominada nodo. Los Cluster constan de dos o más nodos, estos pueden ser simples computadores, sistemas multiprocesador o estaciones de trabajo, el Cluster puede estar conformado por nodos dedicados o por nodos no dedicados. Los Cluster pueden crecer en tamaño añadiendo más máquinas. Un Cluster como un todo puede ser más potente

que la más veloz de las máquinas con las que cuenta, factor que está a la velocidad de conexión con la que se han conectado los nodos del Cluster (Catalán, 2004).

- **Conexiones de Red:** los nodos de un *Cluster* están conectados entre sí por al menos un canal de comunicación, un *Cluster* sólo puede ser un grupo de computadoras personales estándar interconectadas a través de tecnologías *Ethernet*, *Myrinet*, *Infiniband*, SCI (*Scalable Coherent Interface*), entre otras. El medio de comunicación o red permite la coordinación y el intercambio de resultados entre los procesos que intervienen en una tarea, en particular en algunos tipos de clusters, permite la migración de procesos.

- **Sistema Operativo:** cada nodo debe poseer un sistema de control especializado, sistemas multiproceso, multiusuario, y otras características para facilitar la comunicación y acceso. Entre los sistemas operativos se encuentran herramientas específicas para el montaje de *Cluster*, a continuación se listan unos de ellos:

  - GNU-Linux.
  - Unix: Solaris; HP-UX.
  - Windows: NT; 2000; 2003 Server.
  - Mac OS X
  - FreeBSD.

- **Middleware:** esta es una capa de abstracción entre el usuario o aplicaciones de usuario y los sistemas operativos con la finalidad de proveer:

  - Una interfaz única de acceso al sistema donde el usuario tendrá la sensación de utilizar un único ordenador muy potente;
  - Herramientas para la optimización y mantenimiento del sistema tales como migración de procesos, puntos de control, balanceo de carga, tolerancia a fallos y priorización de trabajos;
  - Escalabilidad para la detección y utilización de nuevos nodos; entre otros.

Dentro de los *middleware* más comunes se tiene a Mosix (Barak, 1999), openMosix, Condor, OpenSSI, entre otros.

3.2.3 Clasificación de los *Cluster*

Los *Cluster* pueden clasificarse con base en sus características o en los servicios que soportan. Se pueden tener *Cluster* de alto rendimiento (HPC – *High Performance Computing Cluster*), *Cluster* de alta disponibilidad (HA – *High Availability*), *Cluster* SSI *(Cluster Single System Image – Imagen de Sistema Único)* y *Cluster* de balanceo de carga (Yañez, 2004).

3.2.3.1 Los Cluster de alto rendimiento (HPC)

Han sido creados para compartir el tiempo de proceso para operaciones que requieran alta capacidad de procesamiento, siempre que consten de un algoritmo paralelizable. Su mayor uso se da en aplicaciones de modelamiento. Una de sus características es la alta escalabilidad.

Existen *Cluster* que pueden ser denominados de alto rendimiento tanto a nivel de sistema como a nivel de aplicación. A nivel de sistema se tiene openMosix, mientras que a nivel de aplicación se encuentran otros como MPI (*Message Passing Interface* – Interfaz de Paso de Mensaje) (Argonne, 2009), PVM (*Parallel Virtual Machine* – Maquina Virtual Paralela), *Beowulf*[1] entre otros. En cualquier caso, estos *Cluster* hacen uso de la capacidad de procesamiento que pueden tener varias máquinas (Scyld, 2008).

1*Beowulf:* Sistema *Cluster* computacional en S.O. Linux, fue construido por Donald Becker y Thomas Sterling en 1994 para la NASA. Fue construido con 16 computadores personales con procesadores Intel DX4 de 200 MHz, que estaban conectados a través de un switch Ethernet. El rendimiento teórico era de 3.2 GFlops.

3.2.3.2 Los Cluster de alta disponibilidad (HAC)

También conocidos como *Cluster failover*, su objetivo es permitir que un servicio se encuentren activo durante el máximo periodo de tiempo posible. La monitorización mutua es parte de la colaboración entre los nodos del *Cluster*, los cuales ejecutan *scripts* cuando detectan fallas. Los *scripts* se utilizan para manejar fallas en las conexiones, direcciones IP, discos sistema de archivos así como aplicaciones y reinicio de aplicaciones.

Entre los Cluster HAC se tiene *ServiceGuard*[2], *LifeKeeper*[3], *FailSafe*[4] y *Heartbeat*[5] .

Existe una variante de alta confiabilidad, útil en escenarios donde existe independencia de datos entre las tareas individuales y se necesita saber cómo va a comportarse el sistema (respuesta en tiempo real). En este caso si se presenta una falla los servicios no deben reinicializarse, en su lugar se debe mantener el estado de las aplicaciones, en lugar de utilizar *checkpoints* para relanzar el servicio. Esta funcionalidad requiere de hardware especializado (Yañez, 2004).

3.2.3.3 Los Cluster de Balanceo de Carga

Son utilizados principalmente en servicios de telecomunicaciones y servicios basados en Internet. Ofrecen características de alta escalabilidad pero están limitados a aplicaciones en las cuales no exista interdependencia de estados entre los procesos paralelos.

En la práctica el balanceo se realiza mediante equipamiento dedicado (load balancer) previo a la entrega de las solicitudes a los servidores del *Cluster.*

El balanceo de carga puede hacerse a nivel de aplicación o a nivel IP. El LVS (*Linux Virtual Server*) utiliza balanceo a nivel IP. El proyecto LVS ofrece aplicaciones de mantenimiento y gestión que permiten construir un *Cluster* de servidores que implementa alta disponibilidad y balanceo de carga sobre el sistema operativo GNU-Linux (Yañez, 2004).

Un Cluster puede ser homogéneo si los nodos pueden tener la misma configuración de hardware y sistema operativo; semi-homogeneo si poseen diferente rendimiento pero con arquitecturas y sistemas operativos similares; o heterogéneo si poseen diferente hardware y/o sistema operativo.

3.2.3.4 Los Cluster SSI

En un *Clúster* SSI el conjunto de procesadores actúa como si fuesen un único recurso de computación. Dado que se comporta como un solo sistema, un Cluster ofrece características aptas para diversos entornos tal como e-business, investigación, educación, servicios Web, entre otros (IBM, 2009).

Actualmente los dos proyectos más importantes sobre *Cluster* SSI son openMosix (openmosix, 1999) y OpenSSI (OpenSSI, 2006) de los que se hablará más adelante.

3.2.4 Virtualización

Los *Cluster* y *Grid* son implementaciones de virtualización de recursos, que permiten la unión de recursos para conformar un sistema que brinde mejores prestaciones a determinada aplicación (Fernandez, 2007). La

2*ServiceGuard*: http://www.hp.com/products1/unix/highavailability/ar/mcserviceguard/
3 *LifeKeeper*: http://www.steeleye.com
4 *FailSafe*: http://oss.sgi.com/projects/failsafe1
5 *Heartbeat*: http://linux-ha.org

virtualización permite representar el hardware de una máquina mediante un software que lo emule de manera total o parcial, habilitando la separación de los recursos de un computador en varios ambientes de ejecución.

La virtualización se logra mediante el particionamiento, simulación y emulación de elementos hardware y software, y el manejo de ejecución de tiempo compartido (Fernandez, 2007). La virtualización puede ser de dos tipos: virtualización de recursos y virtualización de plataforma.

La virtualización de recursos se refiere a la agrupación de recursos individuales comunes de un sistema de cómputo, para crear la idea de un solo recurso con el fin de optimizar la utilización, disponibilidad, costo y fiabilidad de recursos; un ejemplo de este tipo de virtualización son los discos RAID (*Redundant Array of Inexpensive Disks* - Arreglo Redundante de Discos Económicos).

La Virtualización de plataforma se refiere a la creación de un ambiente virtual utilizando los elementos hardware y software de un computador; este ambiente virtual es la emulación de un hardware de cómputo completo donde se puede instalar un sistema operativo que se ejecuta como si estuviera en una plataforma de hardware autónoma. Un ejemplo son Las Máquinas Virtuales que constituyen una aplicación software que representa un computador virtual ejecutándose sobre un computador real (Fernandez, 2007). De este modo las máquinas pueden ejecutar un sistema operativo completo (Linux, Windows, entre otros) y ciertas aplicaciones o procesos (por ejemplo, aplicaciones Java). Es posible crear múltiples máquinas en un solo computador dependiendo de las características hardware del equipo.

Las siguientes herramientas software permiten crear maquinas virtuales, tanto en sistemas Linux como en Windows:

- Bochs (Boch, 2001).
- CoLinux (coLinux, 2009).
- Microsoft Virtual PC (Virtual-PC, 2009) y Microsoft Virtual Server (Virtual-Server, 2009).
- Parallels Workstation (Parallels, 2009).
- QEMU (Fabrice, 2005).
- Simics (Virtutech, 2000).
- Virtual Iron (Virtual-Iron, 2009).
- VMware (ESX Server, Fusion, Virtual Server, Workstation, Player y ACE) (VMware, 2009).
- Xen (University, 2008).

3.3 Herramientas *Cluster*

Dentro de las iniciativas de *Cluster* más relevantes se pueden mencionar las siguientes:

3.3.1 Mosix

Mosix actúa a nivel del sistema operativo, es decir, sus módulos son parte del *kernel.* Las aplicaciones se montan sobre ésta capa y se ejecutan de manera distribuida sobre varios nodos sin necesidad de hacer cambios en ellas.

Mosix ofrece una capa de distribución automática de procesos que efectúa un balance de carga mediante la derivación de nuevos procesos hacia la computadora con menos carga. De igual forma permite la migración de procesos desde nodos lentos a nodos más rápidos.

Una de las grandes ventajas de Mosix es que no requiere la confección especial de software. El núcleo de Mosix está formado por algoritmos que monitorizan y dan repuesta a las actividades requeridas del *Cluster.* Estos algoritmos están diseñados manteniendo los principios de facilidad de uso, optimización y escalabilidad. Los algoritmos de Mosix usan la migración de procesos para evitar problemas en la asignación y utilización de memoria (*swapping* o *thrashing)*.

Existe una alternativa de libre distribución, openMosix, desde febrero de 2002.

3.3.2 OpenMosix

OpenMosix modifica el kernel de Linux para trabajar como un nodo del *Cluster*. El algoritmo interno de balanceo de carga se ocupa de migrar manual o automáticamente y de forma transparente los procesos entre los demás nodos del *Cluster*.

Es uno de los proyectos de creación de *Cluster* computacionales más utilizados por la comunidad de desarrolladores, la razón es que es fácil de usar gracia a sus herramientas que facilitan su monitoreo y ejecución.

Esta migración se realiza de acuerdo a la velocidad de la Unidad Central de Proceso (CPU) de cada nodo, a su carga de procesos actual y a la conexión de red que lo une a los demás nodos. No migra procesos que utilicen memoria compartida, que utilicen múltiples hilos "threads" .

Los nodos pueden estar en diferentes subredes y utilizar diferentes enlaces (E*thernet,* PPP, entre otros) (Cortizo, 2007). Los nodos pueden añadirse o quitarse de la red sin que los procesos en ejecución se vean afectados. La red que interconecta a los nodos debe ser una red con suficiente ancho de banda.

openMosix utiliza un sistema de ficheros llamado oMFS (openMosix *File System*) que proporciona *cache*, *time stamp* y consistencia de enlace.

OpenMosix crea *Cluster* computacionales de imagen simple de sistema. Su última versión (openMosix *kernel* 2.6.15) fue liberada en enero 31 de 2006 y funciona en *kernel* 2.4 de Linux. Aunque está diseñado para ejecutarse sobre sistemas Linux con *Kernel* 2.4 se puede ejecutar en sistemas operativos como Windows XP gracias a la virtualización.

Para la implementación de openMoxis sobre sistemas Windows se consideraron las siguientes herramientas de integración Windows-Linux sobre una misma máquina: CoMoxis y CosMos.

3.3.2.1 CoMosix (Cooperative Linux + OpenMoxis)

CoLinux (Cooperative Linux) (Alón, 2004) es una herramienta que permite que los SO Linux y Windows se ejecuten simultáneamente sobre un mismo equipo. CoMosix resulta de la integración de coLinux y openMosix, y permite ejecutar openMosix sobre Windows. CoMosix incluye algunas herramientas de Linux para crear de manera fácil soluciones de *Cluster* heterogéneas, es decir un *Cluster* que posea nodos tanto Linux como Windows (CoMosix, 2007).

3.3.2.2 CosMos (Chaos-OS on Microsoft-OS)

Permite crear *Cluster*s de gran escala, robustez de manera rápida utilizando openMosix. Inicialmente se conoció como CHAOS y fue adaptado al sistema coLinux permitiendo así que openMosix se ejecute en nodos que tienen instalado el sistema operativo Windows y se distribuyó bajo la licencia GPL (Latter, 2004).

3.3.3 Otras herramientas Cluster

Existen otros *Cluster* computacionales que no modifican la forma en que funciona el *kernel*. Éstos utilizan otros medios para ejecutar tareas y mostrar información sobre ellas, y permiten construir, usar y administrar *Cluster*, permitiendo que se utilicen diferentes versiones de Linux (Muñoz, 2006). Estos incluyen: *Cluster*Knoppix, Oscar (Oscar, 2000), Rocks (NFS, 2006), Scyld *Beowulf* (Scyld, 2008).

Otro tipo de software utilizado comúnmente en *Cluster* son los despachadores de tareas, que permiten recibir varias tareas en el *Cluster* para que estas se vayan ejecutando en la medida que existan recursos disponibles en el

Cluster. Las tareas se encolan y los usuarios pueden seguir el progreso de las tareas en la cola o pueden eliminar una tarea determinada. Un administrador puede establecer prioridades y gestionar la cola. Entre este tipo de soluciones están: *Condor y PBS (Portable Batch System).*

Estas herramientas se describen brevemente más adelante en este capítulo.

3.3.3.1 OpenSSI

OpenSSI es un sistema *Cluster* de imagen simple de sistema que permite reunir un conjunto de computadores para que se comporten como si fueran un único equipo. Permite la creación de *Cluster* mediante un entorno de alta disponibilidad SSI para Linux. Sus ventajas son la disponibilidad, escalabilidad y las facilidades de administración de *Cluster*.

Su funcionamiento se basa en la migración automática de procesos de un nodo a otro para balancear el uso de los recursos entre los nodos del *Cluster*. OpenSSI está diseñado para ejecutarse en algunas distribuciones del sistema operativo Linux.

El proyecto paralelo OpenSSI Web View está enfocado en la interfaz para funciones de monitoreo y administración de los *Cluster* OpenSSI (Kilian, 2004).

OpenSSI permite balancear la carga de un *Cluster* dinámicamente, mediante el uso de migración de procesos, característica heredada de Mosix. El módulo de migración de procesos en OpenSSI utiliza un mecanismo de acceso a recursos remotos mediante la gestión de las interfaces de red, el sistema de archivos (*Cluster*CFS, *Cluster File System*) y sobre algunas llamadas al sistema (Cortizo, 2007).

El proyecto OpenSSI es una solución de *Clustering* que ofrece un entorno SSI completo y altamente disponible para Linux. OpenSSI ofrece una solución modular permitiendo que cada sub-componente se pueda configurar de manera individual y además que se pueda sustituir o evolucionar.

Las principales características de OpenSSI son:

- Disponibilidad, escalabilidad y facilidad de administración.
- Nombrado de entidades consistente a lo largo de los nodos.
- Los archivos de acceso a todos los nodos son completos y públicos.
- Está disponible un sistema de archivos del *Cluster* con un *failover* público.
- La migración de procesos se puede realizar de forma completa incluyendo las llamadas al sistema.
- El balanceo de carga de un proceso se realiza en tiempo de ejecución.
- Monitorización y reinicio del proceso.
- Servicio automático de *failover y* Sistema de archivos automáticos del *failover.*
- Arranque por red para nodos sin discos.

3.3.3.2 *Cluster*Knoppix

Es un sistema operativo (*bootable*), con un *kernel* openMosix ejecutable sin instalación. Actualmente se ha desarrollado una versión de Knoppix conocida como *Cluster*Knoppix la cual trae preinstalada una versión de openMosix y se puede instalar o ejecutar directamente desde el CD de instalación, lo que se conoce como Live CD, sin la necesidad de instalar la aplicación sobre los nodos del *Cluster*. Otra opción importante que ofrece es el

arranque de los equipos del *Cluster* a través de la red, esto se hace teniendo en cuenta que se debe configurar el computador servidor del *Cluster* como un servidor DHCP y además configurar las tarjetas de red como PXE[6].

3.3.3.3 Oscar

OSCAR (*Open Source Cluster Application Resources*) permite la instalación y administración de un *Cluster* de Alto Desempeño HPC tipo *Beowulf*, contiene un conjunto de aplicaciones preempaquetadas mediante las cuales se puede instalar y configurar un *Cluster* de manera sencilla, permite la creación de paquetes personalizados para cualquier clase de aplicaciones o utilidades distribuidas. Oscar crea imágenes de discos virtuales para proveer a los nodos todo el software cliente y herramientas administrativas para su uso inmediato, además incluye una arquitectura de prueba robusta y extensible que facilita la configuración del *Cluster*. El uso más general de un *Cluster* Oscar es la computación científica utilizando una implementación de la interfaz de paso de mensajes MPI. Una de las fortalezas de Oscar es la posibilidad de instalar múltiples implementaciones MPI sobre un solo *Cluster*, sin embargo, sólo se puede instalar sobre algunas distribuciones de Linux (Oscar, 2000).

3.3.3.4 Condor

Condor permite la creación de un *Cluster* de alta tasa de salida HTC, es decir un sistema de administración y monitorización de recursos. Puede ser usado para trabajos secuenciales o paralelos y es compatible con entornos de *Grid*. Permite personalizar las políticas de orden de ejecución y añadir tolerancia a fallos (Condor, 2009)

Condor pone en una cola tareas seriales o paralelas, elige cuando y donde ejecutar las tareas basado en unas políticas, monitorea su progreso y finalmente informa al usuario acerca de su estado. Se puede instalar tanto en Windows como en algunas versiones de Linux. (Condor, 2009).

3.3.3.5 PBS (Portable Batch System)

Opera en redes de entornos Unix multiplataforma, usa el modelo cliente-servidor y está organizado como un conjunto de comandos a nivel de usuario que interactúan con servicios a nivel de sistema, estos servicios gestionan la ejecución y reciben las peticiones de usuario.

3.3.4 Herramientas Cluster Seleccionadas

En el presente proyecto se busca generar *Cluster* computacionales que comprendan los equipos de diferentes centros informáticos, los cuales en su gran mayoría tienen preinstalado el sistema operativo Windows, por tal razón es necesario instalar un sistema de *Cluster* que permita utilizar estos equipos sin necesidad de cambiar su sistema operativo.

OpenMoxis está disponible para Windows mediante el uso de herramientas de virtualización. CosMos ofrece esta facilidad con algunas ventajas sobre otras herramientas como entra las cuales se destaca que no requiere de una imagen del sistema operativo ubicada en un sistema de archivos remoto NFS, dado que posee su propia imagen del sistema operativo y por lo tanto se ejecuta más rápido y más fácilmente, ocupa sin embargo más espacio en la máquina anfitrión.

Adicionalmente, OpenMoxis ofrece varias ventajas respecto a otro tipo de herramientas, entre estas están: manejo de múltiples dominios y múltiples sistemas de archivos en el nodo principal del cluster, un manejo de hasta 650 nodos, tamaño ilimitado de archivos, número ilimitado de archivos y direcciones. Adicionalmente provee una creación y copia dinámica de archivos y un control de flujo rápido.

---

6 PXE: *Pre boot eXecution Enviroment,* hace referencia al entorno de ejecución de pre-arranque, el cual permite arrancar e instalar un sistema operativo en computadores desde una red.

Por estas razones se seleccionó para el presente proyecto openMosix como herramienta para la creación del *Cluster* tanto en el servidor como en los nodos y para los equipos con sistema operativo Windows se seleccionó la virtualización de Linux CosMos.

3.4 Tecnología Grid

3.4.1 Concepto

Grid es una infraestructura que involucra el uso integrado y colaborativo de computadores, redes, bases de datos e instrumentos administrados por múltiples organizaciones (Asadzadeh, 2004). "*Grid* es un tipo de sistema paralelo y distribuido que permite compartir, seleccionar y agregar recursos autónomos y distribuidos geográficamente de manera dinámica en tiempo de ejecución dependiendo de su disponibilidad, capacidad, rendimiento, costo y los requerimientos de calidad de servicio de los usuarios" (Buyya, 2009).

Grid es un concepto análogo a las redes de suministro eléctrico, el cual ofrece un único punto de acceso a un conjunto de recursos distribuidos geográficamente en diferentes dominios de administración, dichos recursos pueden ser hardware, instrumentos, bases de datos, programas, computadoras, cluster y dispositivos móviles entre otros. Así los sistemas distribuidos se pueden emplear como un único sistema virtual en aplicaciones intensivas de datos o con gran demanda computacional

Esta tecnología ha evolucionado en la comunidad científica desde aproximadamente el inicio de la década de los 90 (Di Stefano, 2005), (Jacob, 2003), (Zaglakas, 2004). El primer proyecto que diferenció la *Grid* de otras redes distribuidas reenfocando la distribución geográfica para la gestión e integración de software fue I-WAY (*Information Wide Area Year* - Años de la Información de Gran Área), dirigido por Ian Foster. Con aquel proyecto nació Globus (Globus, 2008), una infraestructura *middleware* introducida en 1996 que permite a aplicaciones ser empleadas en una red de recursos heterogéneos y distribuidos. Un elemento central de Globus es el Globus Toolkit el cual define los servicios y capacidades requeridas básicas para construir una *Grid* computacional. Globus constituye la tecnología base para la gran parte de proyectos *Grid* actualmente (Rumiany, 2007). En el trabajo "*The physiology of the Grid*" (Kesselman, 2002) se propone una arquitectura donde definen un conjunto de comportamientos y capacidades centrales claves para sistemas *Grid*. La OGSA (*Open Grid Services Architecture* - Servicios de Arquitectura *Grid* Abierto) desarrollada por la comunidad GGF (*Global Grid Forum* - Foro de *Grid* Global*)* hoy conocida como OGF (*Open Grid Forum* - Foro de *Grid* Abierto) (Open, 2008), representan una evolución hacia la arquitectura de sistemas *Grid* basados en conceptos y tecnologías de servicios Web (Berman, 2003).

3.4.2 Aplicación de Grid

Grid tiene una amplia difusión y utilización en proyectos enfocados en: herramientas software, servicios, aplicaciones científicas y optimización de redes existentes. Un proyecto puede cubrir más de un área de investigación (Mondardini, 2009), por ejemplo en el Reino Unido el programa e-Science (E-Science, 2007) contiene una amplia variedad de proyectos *Grid* en varios campos incluyendo la medicina, la genética y ciencias biológicas, la astronomía y física de partículas, la ciencia medioambiental, el diseño e ingeniería, la química y ciencias de los materiales, las ciencias sociales y proyectos que involucran la participación conjunta de la industria y la academia.

En Estados Unidos el proyecto TeraGrid (Catlett, 2007), un proyecto de infraestructura está dedicado a: la investigación científica abierta, operar, mejorar y ampliar la capacidad de una cyber-infraestructura distribuida para más de 100 disciplinas y satisfacer las crecientes necesidades de la ciencia y la ingeniería de la comunidad. En Europa el EGEE (*Enabling Grids for E-SciencE*) (EGEE, 2008) su alcance es proveer a investigadores en la academia y la industria de 30 países europeos acceso a los mejores recursos de computación sin importar la ubicación geográfica, constituyendo una plataforma donde aplicaciones de gran importancia en informática, biomédica, biología, física de altas energías y meteorología funcionen a su máximo rendimiento. El centro que dirige el proyecto es el Laboratorio Europeo de Física de Partículas (CERN). Estos y otros proyectos actualmente existentes se pueden encontrar citados en la página del centro de información *Grid computing* (Buyya, 2009).

La *Grid* permite que instituciones de menos recursos tengan acceso a poder computacional de forma remota, o que diversas instituciones puedan unir sus recursos computacionales para obtener uno más poderoso. (Lamadrid, 2008) La implementación de la tecnología *Grid* proporciona múltiples beneficios en el campo de la investigación y la educación (Ferreira, 2005), entre estos beneficios se pueden mencionar: la eficiencia y reducción de costos, la explotación de recursos inutilizados, la creación de trabajo colaborativo, la fiabilidad/disponibilidad, el acceso a recursos adicionales, el balanceo de recursos, la reducción de tiempo de resultados, los Recursos Virtuales y las Organizaciones Virtuales (VO - *Virtual Organizations*), entre otros.

3.4.3 Características de Grid

La naturaleza dinámica de la *Grid* implica que exista inherentemente un nivel de tolerancia a fallas para los códigos, o miles de tareas, procesos o aplicaciones que pueden soportarse en miles de unidades de procesamiento altamente distribuidos. Como el número de recursos involucrados se incremente, así lo hará también la probabilidad que alguno de estos recursos falle durante la computación. Las aplicaciones *Grid* deben ser capaces de controlar en tiempo real fallas de comunicación o recursos de cómputo y proveer acciones para reactivar o recuperar las fallas.

Las herramientas de la *Grid* pueden asegurar un mínimo nivel de fiabilidad de cómputo en la presencia de fallas implementando mecanismos en tiempo real que adicionan alguna forma de fiabilidad de operaciones (Lee, 2003), para ello se toma en cuenta los siguientes condiciones (IBM-Corp, 1994):

- **Seguridad:** para habilitar el uso compartido de recursos, tienen que adoptarse medidas de seguridad para proteger los datos confidenciales o información, también para evitar daños eléctricos en los recursos computacionales.
- **Gestión de la carga de trabajo:** puesto que los computadores pueden ser añadidos, removidos, o cambiados frecuentemente, es necesario la detección continua de la disponibilidad de recursos y su estado actual.
- **Planificación de trabajo:** todas las tareas requeridas por la *Grid* deben ser hechas de la mejor manera teniendo en cuenta prioridades. Esto es conocido técnicamente como “balanceo de carga de trabajo”.
- **Gestión de datos:** todos los trabajos llevados a cabo necesitan ciertos datos que deben ser vistos y recuperados cuando sea necesario. Por eso la gestión de datos implica búsqueda y almacenamiento de datos de una manera fiable y segura.
- **Interfaces de usuario:** en la misma dirección un visor Web provee un rápido y fácil acceso para la información disponible, se requiere una interfaz de usuario amigable y de fácil manejo a través de la cual el usuario pueda consultar la *Grid*.

3.5 Herramientas Grid

3.5.1 Unicore

Unicore (*UNiform Interface to COmputer REsources* – Interfaz Uniforme para Recursos Computacionales) (Unicore, 2009), (Almond, 1999). Proporciona una interfaz uniforme para la preparación de trabajos, y el acceso seguro sin restricciones a los recursos *Grid*. Es transparente para los usuarios a fin de facilitar el desarrollo de aplicaciones distribuidas, que dentro de Unicore se definen como aplicaciones fragmentadas que pueden funcionar en diversos sistemas informáticos asincrónicamente o pueden ser sincronizadas secuencialmente. La última versión implementa varios estándares abiertos para lograr la interoperabilidad e integra fuerte seguridad y capacidad de flujos de trabajo.

3.5.2 Legion

Legion (Legion, 2009), (Grismshaw, 1997) es un meta-sistema basado en objetos, desarrollado en la universidad de Virginia. Legion proporciona la infraestructura software de modo que un sistema de máquinas heterogéneas de alto rendimiento, geográficamente distribuidas, puedan ejecutar tareas recíprocamente sin restricciones. Legion intenta proveer a los usuarios, en sus sitios de trabajo, una sola máquina, coherente y virtual.

En el sistema Legion se aplican los siguientes conceptos:

- Todo es un objeto, los objetos representan todos los componentes de soporte físico y software. Cada objeto es un proceso activo que responde a las invocaciones de método de otros objetos dentro del sistema. Legion define una API para la interacción de objetos, pero no el lenguaje de programación o el protocolo de comunicación.
- Los usuarios pueden definir sus propias clases, al igual que en sistemas orientados a objetos, se puede eliminar o redefinir la funcionalidad de una clase. Esta característica permite que la funcionalidad sea agregada o removida para cubrir las necesidades de un usuario.

3.5.3 Gridbus

Gridbus (*GRID computing and BUSiness* - computación y negocio *Grid*) se dedica al diseño y al desarrollo de tecnologías *Cluster* y *middleware Grid* para la computación orientada al servicio (Gridbus, 2009). Proporciona servicios de extremo a extremo para agregar o prestar servicios de recursos distribuidos dependiendo de la disponibilidad, capacidad, funcionamiento, costo, y requisitos de calidad del servicio de los usuarios. El objetivo clave del proyecto Gridbus es desarrollar *Cluster* de nueva generación y tecnologías *Grid* que apoyen la computación de uso general.

3.5.4 Globus Toolkit

Globus (Globus, 2008), (Foster, 1997) proporciona una infraestructura software que permite a usuarios gestionar recursos de computación heterogéneos y distribuidos como una sola máquina virtual. El proyecto Globus es un esfuerzo de investigación multi-institucional de los E.E.U.U. que proporciona los servicios básicos y las capacidades que se requieren para construir una *Grid*.

El proyecto Globus provee un conjunto de herramientas de software libre que se pueden utilizar para construir *Grid* y aplicaciones basadas en *Grid*. Permite la distribución del poder de computación, de bases de datos, y de otras herramientas con seguridad en línea a través de límites corporativos, institucionales y geográficos sin sacrificar la autonomía local. Los servicios, las interfaces y los protocolos de base en Globus permiten que los usuarios tengan acceso a recursos alejados sin restricciones mientras que simultáneamente ofrece el control local sobre quién y cuando puede utilizar los recursos.

El conjunto de herramientas de Globus es modular, y una aplicación puede explorar características de Globus, tales como localización o gestión de recursos, seguridad, infraestructura de información, comunicación, tolerancia a fallas e interoperabilidad entre diferentes entornos entre otras.

Globus esta integrado por diversos componentes software esenciales para su correcto funcionamiento. Como puede observarse en la Figura 3-1 (Sotomayor, 2005), dichos componentes pueden organizarse en la arquitectura de Globus y distribuirse en cinco categorías específicas:

- Seguridad (*Security*).
- Gestión de datos (*Data Management*).
- Gestión de ejecución (*Execution Management*).
- Servicios de información (*Information Services*).
- Tiempo de ejecución común (*Common Runtime*).

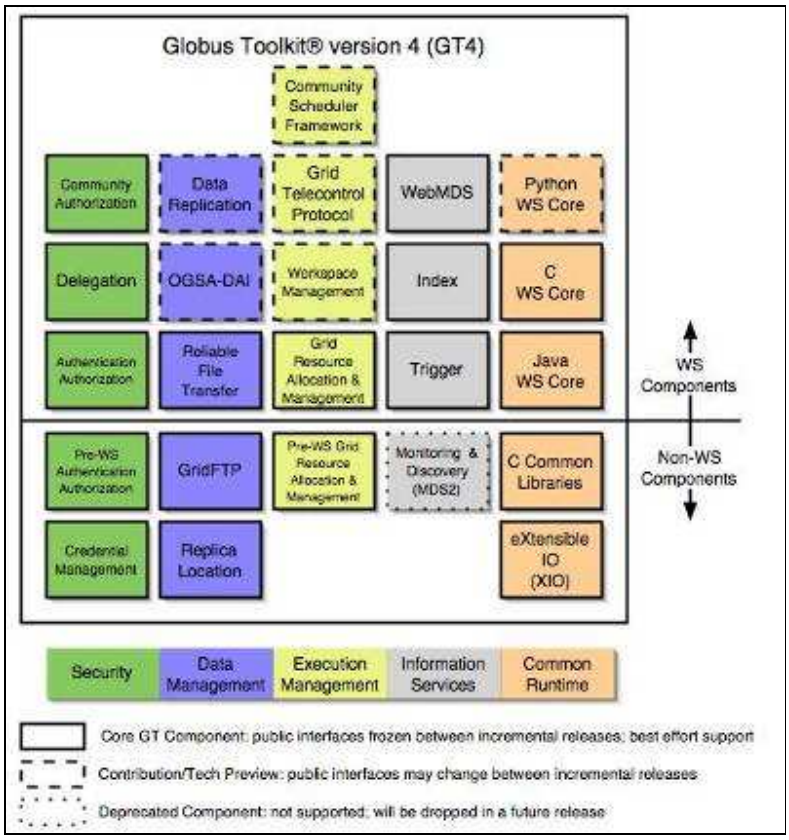

Figura 3-1 Componentes Globus Toolkit 4.0

Para la explicación de estos componentes se parte del hecho que GT4 se enfoca en servicios Web (WS) y que el juego de herramientas también incluye componentes que no están implementados sobre componentes de servicios Web. Por ejemplo, el componente de GridFTP (*File Transfer Protocol*) utiliza un protocolo no-WS que comenzó como protocolo *ad hoc* de Globus, pero se convirtió más adelante en una especificación de *Global Grid Forum* (Sotomayor, 2005).

El juego de herramientas de Globus incluye componentes que pueden ayudar a construir sistemas *Grid*. Una de estas herramientas es el componente núcleo de Java WS porque es la base para la mayor parte de los componentes de WS. Un punto importante es que no es necesario tener profundos conocimientos sobre la base de Java WS para utilizar muchos componentes del GT4 como GRAM (*Grid Resource Allocation and Management*), MDS (*Monitoring and Discovery System*) y entre otros. Sin embargo, si se desea construir un sistema *Grid* que integre todos los componentes de Java WS con servicios personalizados, se requiere tener más conocimientos sobre el núcleo de Java WS (Sotomayor, 2005).

A continuación se describen uno a uno los componentes de cada categoría que se observan en la Figura 3-1

3.5.4.1 Seguridad

Los componentes de seguridad de Globus (Ver Figura 3-2) permiten hacer comunicaciones seguras a través de la infraestructura de Seguridad *Grid* (GSI), sus componentes son: (GT4FactSheet, 2005).

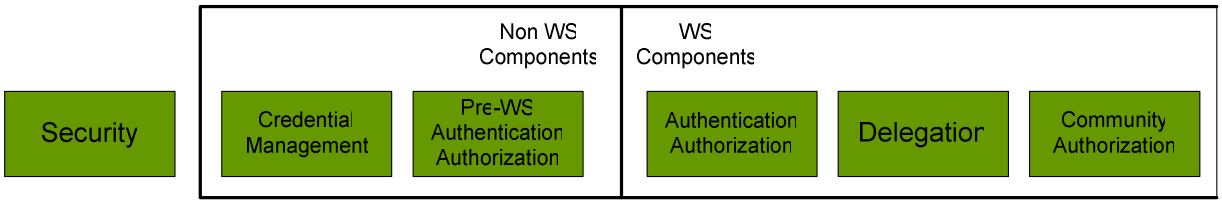

Figura 3-2 Componentes de Seguridad

- **Gestión de Credenciales (*Credential Management*) este componente comprende dos sub-componentes**:
  - SimpleCA: es un paquete que proporciona una autoridad simplificada de certificación con el fin de publicar las credenciales a los usuarios y a los servicios del juego de herramientas de Globus.
  - MyProxy: es un depósito credencial en línea, que permite almacenar las credenciales proxy X.509[7] mediante una contraseña. Gracias a estos se elimina la necesidad de copiar manualmente la clave privada y los archivos de certificado entre las máquinas. *MyProxy* también se puede utilizar para la autenticación de los portales *Grid* y la renovación credenciales con los gestores del trabajo.

- **Autorización y Autenticación Pre-WS (*Pre WS Authentication Authorization*)**: estos componentes proporcionan APIs y herramientas para la autenticación, la autorización y la gestión de certificados. La API de autenticación se construye usando la tecnología de infraestructura de llave pública (PKI), por ejemplo certificados X.509 y TLS. Además de la autenticación ofrece un mecanismo de delegación basado sobre certificados *Proxy* X.509.

- **Autorización Autenticación (*Authentication Authorization*)**: este componente es comprendido por dos sub-componentes:
  - Seguridad de nivel de transporte y mensajes: la parte de servicios Web de GT4 utiliza SOAP sobre HTTP para la comunicación de mensajes. La autenticación WS y la autorización de seguridad del nivel de mensaje ejecuta la seguridad estándar de servicios Web y la especificación de conversación segura para Servicios Web y así proporcionar la protección de mensaje para los mensajes SOAP. Las características incluyen la autenticación del remitente, la encriptación del mensaje, la protección de la integridad del mensaje y la protección de la respuesta. La autenticación de servicios Web y la autorización de seguridad del Nivel-Transporte, para el transporte de mensajes, la proporciona un canal seguro usando HTTP sobre SSL/TLS (HTTPS).
  - Marco de Autorización: este componente proporciona un marco para la autorización del nivel de contenedor. Permite que las cadenas de los módulos de autorización sean asociadas a las diferentes entidades en el contenedor como el caso de los servicios.

- **Delegación (*Delegation*)**: este servicio proporciona una interfaz para la delegación de credenciales en un ambiente de alojamiento (*hosting*), es decir que una sola credencial delegada se puede compartir a través de invocaciones múltiples de servicios en ese ambiente de recibimiento.

- **Comunidad de Autorización CAS (*Community Authorization*)**: permite que una organización virtual exprese políticas respecto a los recursos distribuidos a través de un número de sitios. Un servidor CAS publica aserciones a los usuarios virtuales de la organización, concediéndoles derechos de acceso de granuralidad-fina a los recursos. Los servidores reconocen y hacen cumplir las aserciones. CAS se diseña para ser extensible a los múltiples servicios y es apoyado actualmente por el servidor de GridFTP.

7 X.509 Proxy: Han sido propuestas para usarse en Infraestructuras de Seguridad Grid y permitir una dinámica delegación y simples firmas conforme a la ley para usuarios finales.

3.5.4.2 Gestión de Datos (Data Management)

Corresponde a los componentes que permiten gestionar un gran conjunto de datos en la organización virtual Figura 3-3, estos son (GT4FactSheet, 2005):

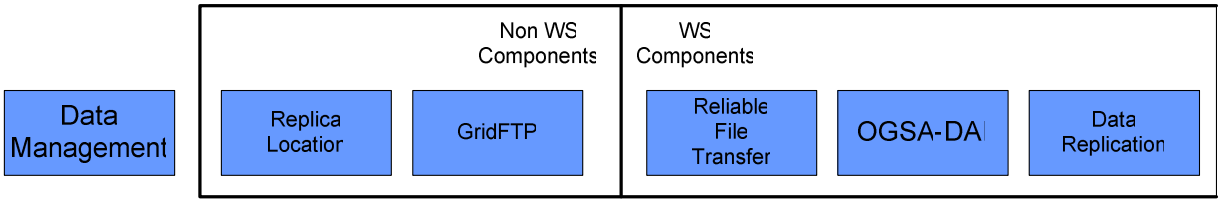

Figura 3-3 Gestión de datos

- **Servicio de localización de replicación (RLS) (*Replica Location*)**: permite el registro y el descubrimiento de replicaciones. Las réplicas son copias de archivos en sistemas de almacenamiento. Para entender mejor el funcionamiento del RLS es necesario definir los siguientes términos:
  - Nombre de archivo lógico: en un identificador único para el contenido de un archivo.
  - Nombre de archivo Físico: es la ubicación de una copia del archivo en un sistema de almacenamiento.

  Estos términos son ilustrados en la Figura 3-4, la tarea del RLS es mantener la asociación entre el nombre de archivo lógico y uno o más nombres de archivos físicos de réplicas. Los usurarios pueden proveer un nombre de archivo lógico al servidor RLS y pedir todos los nombres de archivo físico de las replicas registradas a él, además de asociar atributos o información descriptiva a los diferentes nombres registrados en el RSL.

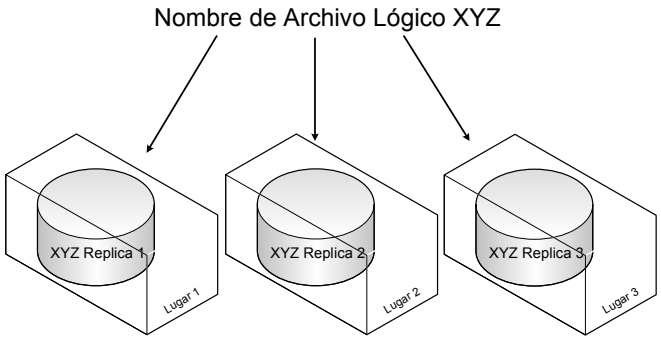

Figura 3-4 Ejemplo de asociación entre un nombre de archivo lógico y tres réplicas en diferentes.

GridFTP: es un protocolo de alto rendimiento, seguro, confiable de transferencia de datos optimizado para el ancho de banda en redes. El protocolo GridFTP se basa en FTP altamente popular en Internet.

- **Servicio de Transferencia Confiable (RFT) (*Reliable File Transfer*)**: la implementación del servicio RFT en GT 4.0 utiliza mensajes estándar SOAP sobre HTTP para someter y manejar un sistema de transferencias de terceras partes, de cancelación de archivos y de directorios utilizando GridFTP. El servicio también proporciona una interfaz de múltiples parámetros del canal de control de transferencia como, por ejemplo, el tamaño del almacenador (*buffer*) intermediario TCP y flujos paralelos entre otras. El usuario crea un recurso de RFT haciendo una petición de transferencia (que consiste en un sistema de transferencias de terceras partes GridFTP) al servicio RFT principal. El recurso se crea, se autoriza, se autentica y finalmente expone sus operaciones para controlar y para manejar las transferencias.

- **OGSA-DAI**: proporciona un marco puro del servicio de datos de Java para el acceso e integración de recursos de datos tales como archivos, bases de datos XML y relacionadas sobre la *Grid*.

- **Servicio de Réplica de Datos (DRS) (*Data Replication*)**: la funcionalidad primaria del componente es permitir que los usuarios identifiquen un sistema de archivos deseados que existen en su ambiente *Grid*, hacer las replicas locales de esos archivos de datos transfiriendo archivos de una o más localizaciones fuente, y colocar las nuevas reproducciones en un Servicio de Localización de Replicas. El DRS se ajusta a la especificación de WSRF y expone un Recurso-WS (llamado un recurso "Replicador") el cuál representa el estado actual de la actividad pedida de la réplica y permite que los usuarios se suscriban a las varias características del recurso para supervisar el estado del recurso. El Núcleo de Java WS utiliza Globus RLS para localizar y registrar las replicas y Globus RFT para transferir archivos.

3.5.4.3 Gestión de Ejecución (Execution Management)

Los componentes de gestión de ejecución están relacionados con la iniciación, monitoreo, gestión, planeación y coordinación de programas ejecutables usualmente llamados trabajos dentro de la *Grid* Figura 3-5 (GT4FactSheet, 2005).

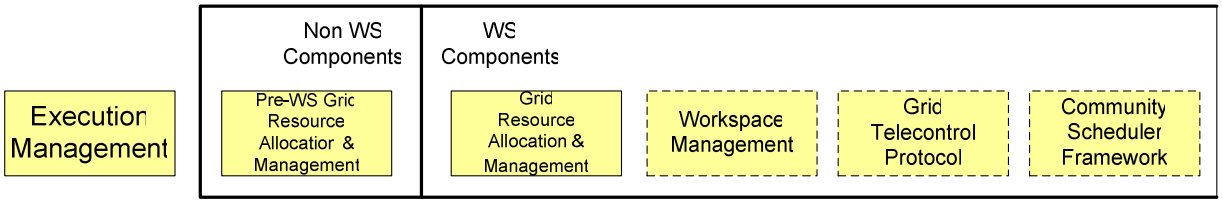

Figura 3-5 Gestión de ejecución

- **Asignación y Gestión de Recursos *Grid* (GRAM) Pre-WS**: proporciona una sola interfaz para solicitar y utilizar recursos de un sistema remoto para la ejecución de "trabajos".

- **Asignación y Gestión de Recursos *Grid* de Servicios Web (GRAM de WS)**: abarca un sistema de servicios Web para localizar, someter, supervisar, y cancelar trabajos en recursos de computación de la *Grid*. El GRAM WS no es un planificador de trabajo, sino un sistema de servicios y clientes para comunicar con una gama de diversos planificadores de trabajos *batch/Cluster* usando un protocolo común.

- **Cuentas Dinámicas (DA) denominado anteriormente como (*Workspace Management Services*)**: permiten que un cliente de la *Grid* cree y maneje dinámicamente un espacio de trabajo en un sitio lejano. La infraestructura se compone de un servicio de fábrica que permite a un cliente autorizado de la *Grid* crear cuentas individuales o grupos de cuentas, y se compone también de un servicio de cuenta que permite que a un cliente autorizado de la *Grid* manejar características de la cuenta individual, tales como políticas o Tiempo de Vida *Time to Live* (TTL) del acceso de la cuenta. Estos conceptos se representan como servicios de WSRF y se ejecutan usando la puesta en práctica de WSRF GT4.

- **Protocolo de control de Tele-operaciones Globus (GTCP)**: es una interfaz del servicio para el telecontrol, generalmente utilizado para controlar, por ejemplo, las simulaciones físicas y de cómputo heterogéneas unidas en experimentos distribuidos geográficamente en la ingeniería de terremotos. También se ha utilizado para controlar los sistemas de adquisición de datos (que accionan la recopilación de datos) y las cámaras de alta resolución (que accionan la adquisición de imágenes) durante experimentos de ingeniería, y, en un grado inferior, para controlar la colocación, la longitud focal, entre otras, de microscopios electrónicos en el campo de la neurología. GTCP expone dos interfaces: una interfaz WSRF del servicio usado por los clientes para controlar los instrumentos remotos y las simulaciones, y un módulo de interconexión para facilitar la integración de los nuevos programas de computo (plataformas físicas o de cómputo de simulación) al servidor de GTCP.

- **El Marco del Planificador de la Comunidad 4.0 (CSF4.0)**: proporciona una interfaz y herramientas para que los usuarios de la *Grid* desplieguen trabajos, creen reservaciones avanzadas y definan diversas políticas de previsión en el nivel *Grid*.

3.5.4.4 Servicios de Información (Information Services)

Los servicios de información Figura 3-6, estos componentes están desactualizados y seguramente desaparecerán en futuras publicaciones del juego de herramientas de Globus (GT4FactSheet, 2005).

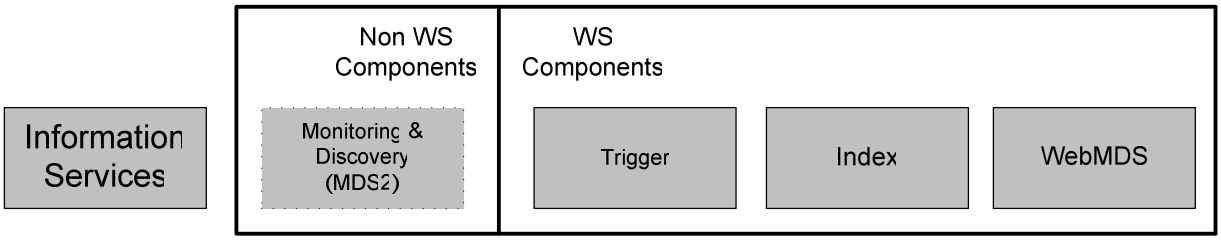

Figura 3-6 Servicio de información

- **El Sistema de Descubrimiento y Monitoreo MDS PRE-WS (*Monitoring & Discovery* (MDS2))**: proporciona un mecanismo estándar para publicar y descubrir el estado de los recursos y la información de configuración.

- **Servicio Disparador (*Trigger*)**: recoge datos de recursos en la *Grid* y de acuerdo a ellos tomar una acción; un ejemplo de esto puede ser el envío de un *e-mail* cuando la longitud de la cola de un recurso de cálculo pasa de un valor umbral.

- **Servicio de índice (*Index*)**: recoge información de supervisión y descubrimiento de recursos *Grid*, y la publica en una sola localización.

- **WebMDS**: permite a usuarios finales ver la información de la supervisión a través de una interfaz estándar de un navegador Web, sin la necesidad de instalar algún software adicional en el PC cliente. WebMDS se ejecuta como un *servlet* que utiliza una interfaz que recopila la información de supervisión (o cualquier otra información en formato XML), y la presenta al usuario en una forma legible.

3.5.4.5 Componente común de tiempo de ejecución (Common Runtime)

El componente común de tiempo de ejecución proporciona un conjunto de bibliotecas y herramientas fundamentales que son necesarias para construir servicios Web Figura 3-7 (GT4FactSheet, 2005).

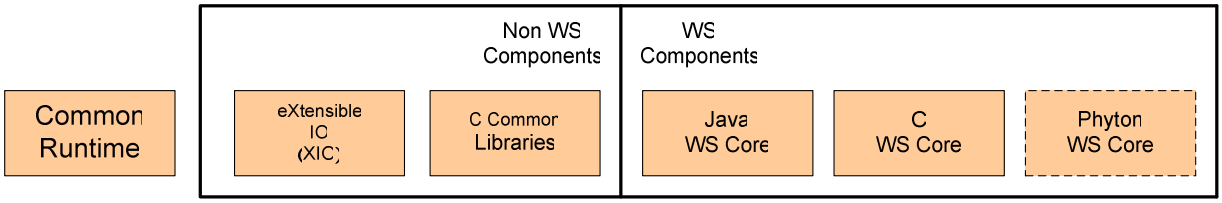

Figura 3-7 Componente común de tiempo de ejecución

- **Globus XIO (*eXtensible IO*)**: es una biblioteca extensible que posee interfaces de entrada y salida escrita en el lenguaje de programación C que se utiliza junto con las herramientas de Globus. Provee una sola API (abierto/cerrado/lectura/escritura) que soporta múltiples protocolos, con aplicaciones encapsuladas como controladores. Los controladores de XIO distribuidos con GT4.0 incluyen el TCP (*Transmission Control Protocol*),

UDP (*User Datagram Protocol*), HTTP (*Hypertext Transfer Protocol*), GSI (*Grid Security Infraestructure*), GSSAPI_FTP (*Generic Security Service Application Program Interface*), telnet[8] y colas.

- **Bibliotecas comunes de C (*C Common Libraries*)**: proporcionan una capa de abstracción para tipos de datos, llamadas de sistema libc, y estructuras de datos usadas a través del juego de herramientas de Globus por las aplicaciones.

- **Núcleo Java WS (*Java WS Core*)**: es una implementación de la familia de estándares de la Estructura de Recursos de Servicios Web (WSRF) y la Notificación de Servicios Web (WSN). Ésta provee APIs y herramientas para la construcción de servicios Web *stateful*[9].

- **Núcleo C WS (*C WS Core*)**: provee un conjunto de herramientas básico en C para la creación de servicios Web y clientes WSRF conforme a las especificaciones del Recurso-WS y Notificación-WS.

- **Núcleo Python WS (*Python WS Core*)**: provee un conjunto de herramientas *python* básicas para la creación de servicios Web WSRF-permitidos con interoperabilidad para Java WSRF. El desempeño es la principal motivación y preocupación. El soporte WSRF incluye Vida de Recursos-WS, Propiedades de Recursos-WS y Notificaciones de Recursos-WS.

3.5.5 Comparación de herramientas Grid

En la Figura 3-8 (Asadzadeh, 2004) se hace una comparación de las herramientas *middleware* vistas anteriormente con base en los servicios provistos a través de la arquitectura *Grid*.

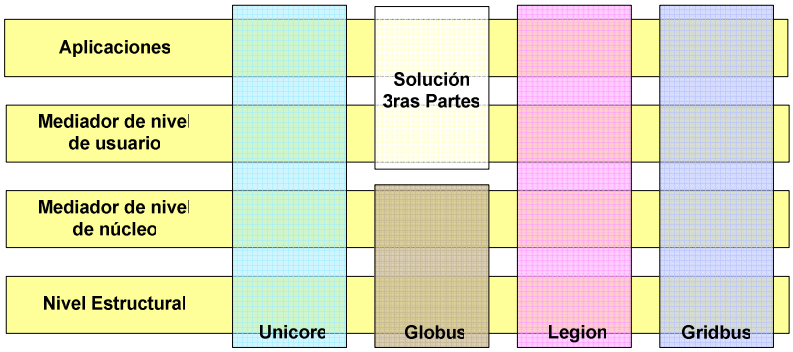

Figura 3-8 Comparación de Unicore, Globus, Legion y Gridbus

Unicore y legion están verticalmente integrados y firmemente unidos. Proporcionan componentes cliente y servidor. Globus sigue un método denominado "conjunto-de-servicios" y proporciona un extenso conjunto de herramientas básicas que se pueden utilizar selectivamente para construir sistemas *Grid* con diversos servicios como detección y control de fallas. Por ejemplo, aunque Gridbus tenga su propio *middleware* independiente de bajo nivel, su gestor se ha diseñado para funcionar con Globus. El conjunto de componentes Gridbus, se extienden a través del apilado, pero no se integran tan firmemente como Unicore y Legion y se pueden utilizar de manera independiente uno de otro.

En la Tabla 3-1 (Asadzadeh, 2004) se hace una comparación de las diversas herramientas *Grid*.

---

8 Telnet: (*TELecommunication NETwork*) es el nombre de un protocolo de red (y del programa informático que implementa el cliente), que sirve para acceder mediante una red a otra máquina, para manejarla remotamente como si se estuviera sentado delante de ella.
9 Los servicios *stateful* son aquellos que mantienen almacenada la información en las variables de la sesión, es decir que pueden mantener ciertos datos almacenados durante el tiempo en que el servicio esté disponible en el servidor de servicios Web.

GRID		Herramientas			
		Unicore	***Globus Toolkit***	***Legion***	***Gridbus***
Características	**Enfoque**	Modelos de programación de alto nivel	Servicios de bajo nivel	Modelos de programación de alto nivel	Abstracción y modelos de mercado
	Categoría	Tareas uniformes de presentación y monitoreo	Computación genérica	Computación genérica	Computación genérica
	Arquitectura	Sistema multi-niveles Vertical	En capasy Toolkit modular	Sistema integrado verticalmente	Componentes en capas y modelo de utilidad
	Modelo de Implementación	Objeto de trabajo abstracto	Modelo de reloj de arena con nivel de sistema	Metasistema orientado a objeto	Modelo de reloj de arena con nivel de usuario
	Tecnologías de implementación	Java	C y Java	C++	C, Java, C# y Perl
	Plataforma de ejecución	Unix	Unix	Unix	Unix y Windows con .NET (Alchemi)
	Entornos de programación	Entorno de flujo de trabajo	Librerías de sustitución para Unix & librerías C. librería MPI especial (MPICH-G), CoG (Commodity *Grid*) kits en Java, Pitón, CORBA, Matlab, Java Server Pages, Perl y Servicios Web.	Interfaces de programación de Aplicación (API) Legion, Utilidades de línea de comandos.	API Java gestor basado en XML lenguaje de parámetros de barrido, modelos de hilo *Grid* vía Alchemi.
	Algunos usuarios y aplicaciones	EuroGrid (EuroGrid, 2009), Proyecto de Interoperabilidad Grid (GRIP) (Menday, 2003), OpenMolGrid (Bala, 2002), y NAREGI Japones (NAREGI, 2003).	AppLeS (Berman, 1997), Ninf (Sato, 1997), Nimrod-G (Abramson, 2000), NASA IPG (Johnston, 1999), Condor-G (Frey, 2001), Gridbus Broker (Venugopal, 2004), Proyecto eScience de UK (Hey, 2002), GriPhyN (GriPhyN, 2006), y EU Data Grid (Hoschek, 2000).	NPACI Testbed (Grismshaw, 2006) y NCBioGrid (NC-BioGrid, 2006). Adicionalmente, este ha sido usado en el estudio del flujo continuo simetrico axial (Beekwilder, 1998), y aplicaciones de proteinas plegables (Natrajan, 2001).	Portal ePhysics (Beeson, 2009), Belle Análisis Data Grid (BADG, 2005), NeuroGrid (Buyya, 2003), Natural Language Engineering (Hughes, 2004).
	Modelo de distribución	Código abierto	Código abierto	No código abierto, disponible en versión comercial	Código abierto

Tabla 3-1 Comparación de middleware Grid

Globus es uno de los middleware Grid de bajo nivel más utilizado actualmente. Proporciona servicios clave tales como acceso de recursos, gestión de datos e infraestructura de seguridad. Unicore es un sistema Grid utilizado en java usado en diversos proyectos incluyendo EuroGrid y el Proyecto de Interoperabilidad Grid (GRIP). Globus y Legion proporcionan servicios gestores mientras que Unicore y Gridbus no proporcionan gestor de peticiones de usuario. La seguridad de Unicore se basa en el protocolo de SSL (Secure Socket Layer) y el tipo certificados X.509V3. Globus proporciona servicios de seguridad con GSI, que se basa otra vez en los certificados SSL y X.509. En Gridbus, los servicios de seguridad son reforzados por el conjunto de herramientas de Microsoft .NET[10]. Otros componentes de Gridbus refuerzan ampliamente los servicios de seguridad proporcionados por Globus GSI.

Globus contiene implementaciones de WSRF (*Web Service Resource Framework* – Estructura de Recursos para Servicios Web) en Java y C, muchas de las herramientas de Globus han sido reprogramadas para utilizar WSRF. Respecto a los servicios Web es importante tener claro que éstos generalmente no tienen estado, es decir, no mantiene ninguna información entre sucesivas invocaciones. Esta falta de estado limita el número de cosas que se pueden hacer con los servicios Web, aunque existen soluciones, como por ejemplo hacer que el servicio lea el

10 http://www.microsoft.com/NET/

estado de una base de datos, o hacer que recupere el estado de la sesión a través de *cookies* o sesiones del servicio Web.

WSRF proporciona un conjunto de operaciones que los servicios Web compatibles pueden implementar para convertirse en servicios Web con estado; los clientes de estos servicios Web se comunican con servicios WSRF que representan a recursos y que permiten almacenar y recuperar información. Los clientes invocarán el servicio añadiendo como parámetro el identificador del recurso que será utilizado durante la petición, codificado en una referencia que cumpla con direccionamiento de servicios Web. Esta referencia puede ser simplemente una URI (*Uniform Resource Identifier*), o puede ser tan compleja como un XML que identifique o incluso describa totalmente el recurso en cuestión.

Globus es la herramienta de mayor divulgación y uso a nivel mundial y además sobre ella se desarrollan aplicaciones por terceros, ella ofrece servicios como el RFT y GRAM que permiten características de tolerancia a fallas limitada, permiten la creación de servicios Web confiables *stateful* gracias al WSRF, permiten localización y gestión de recursos, seguridad, infraestructura de información, comunicación e interoperabilidad entre diferentes entornos, además existe una documentación muy extensa actualizada en su sitio Web. Por lo anterior y muchas ventajas más es la herramienta que se utilizará en el presente trabajo, de tal manera que la capa Grid estará a cargo de la herramienta Globus toolkit.

3.6 Tecnología DTN (Delay/ Disrupt Tolerant Networking/ Red Tolerante a Retardos/ Interrupciones)

3.6.1 Disponibilidad y confiabilidad en redes de telecomunicaciones

El objetivo de las redes de telecomunicaciones es proveer comunicación continua. De acuerdo al desarrollo de tecnologías y el surgimiento de nuevas aplicaciones, estas redes han tenido ciclos de especialización y convergencia.

Las redes de telefonía, las cuales tienen enfoque síncrono para servicios de voz, son tolerantes a errores pero sensibles a retardos. Para estas redes, las capas de transporte se basan en canales de velocidad constante, y se soportan sobre tecnologías de transporte sincrónico, en especial SDH (Synchronous Digital Hierarchy). Las redes de datos por su parte tienen un enfoque asíncrono, son sensibles a errores pero tolerantes a retardos de baja escala. Las tecnologías de red que las soportan son generalmente IP y Ethernet.

Actualmente, el crecimiento de las redes IP y la sustitución de las redes conmutadas han impulsado la convergencia de servicios de voz y datos sobre las primeras. Esto ha impuesto requerimientos de gestión de calidad de servicio sobre las redes de datos subyacentes. Las tecnologías de telecomunicaciones han evolucionado para soportar todo tipo de servicios sobre redes IP en configuraciones de alta disponibilidad, manteniendo los retardos y la pérdida de paquetes en límites adecuados para cada tipo de servicio.

Las configuraciones de alta disponibilidad en redes de telecomunicaciones se implementan en varias capas. En la capas de transporte y transmisión se logran mediante la diversidad de trayectos y configuraciones en anillos redundantes. En las capas de servicio y aplicación se logra mediante configuraciones de redundancia N+1 en servidores.

Sin embargo, el despliegue de infraestructura de telecomunicaciones para alcanzar sitios apartados (conmutación, datos y transmisión) es muy limitado y su actualización o reposición puede resultar económicamente inviable.

En los enlaces de este tipo de localidades la tasa de errores llega a ser muy alta, los retardos y la variación de retardos muy grande y las interrupciones generan una baja disponibilidad. Estas condiciones se presentan especialmente sobre enlaces de radio de largo alcance y acceso satelital.

Las funcionalidades y configuraciones de las tecnologías de red pueden aportar a mejorar las condiciones de conectividad hasta cierto límite. Cuando los retardos o los periodos de indisponibilidad de los enlaces son muy grandes las tecnologías de la capa de transporte y de la capa de red no operan correctamente. De igual forma los programas de la capa de aplicación ligados a los servicios de la capa de transporte y red pueden quedar in-operativos.

3.6.2 Confiabilidad y disponibilidad en capas superiores

En condiciones de conectividad intermitente, retardo variable o retardo imprevisto de las comunicaciones, tasas de error altas y variables, los protocolos de Internet (TCP, UDP) no logran su cometido. Para asegurar el correcto funcionamiento de aplicaciones en estas condiciones se debe tomar en cuenta dos premisas:

- Las aplicaciones deben estar diseñadas para estos escenarios:
    - Deben implementar los conceptos de estados y transacciones.
    - Deben minimizar los intercambios extremo a extremo y las interacciones de tipo diálogo.
    - Deben estar adaptadas a reinicios en medio de transacciones.
    - Deben proveer e intercambiar información acerca de la prioridad y tiempo de vida de la información que se intercambia.
- Debe existir una capa intermedia entre la aplicación y la capa de transporte para proveer funciones y condiciones estándares para el intercambio asíncrono confiable de información y gestione los estados de conectividad.

El middleware se basará en un esquema de "almacenamiento y reenvío" que permita aprovechar las oportunidades aleatorias o programadas de comunicación entre los componentes de la red. Esta comunicación puede estar basada en o sujeta a la movilidad de los equipos de la red (Warthman, 2003), (Fall, 2003), (Jones, 2007), (Beck, 2007)

Las aplicaciones desarrolladas para usar con DTN deben implementar los conceptos de estados y transacciones y el middleware debe estar en capacidad de manejar funcionalidades de retraso, almacenamiento y reenvío.

3.6.3 Casos de Utilización de Conceptos DTN

DTN está enfocada en soluciones para ambientes hostiles de comunicación que incluyen: comunicaciones en regiones remotas de difícil acceso, comunicaciones en escenarios de guerra, comunicaciones espaciales, comunicaciones submarinas, entre otros. Los proyectos basados en conceptos DTN son de aplicación específica, algunos casos son:

- ZebraNet (Juang, 2002) nodo sensores inalámbricos (adheridos a los animales) colectan datos de ubicación y de forma oportuna reportan su historial cuando ellos se acercan a una estación base con rango de alcance. En
- *Data Mules* nodo sensores de baja potencia pueden ahorrar energía si estos periódicamente son visitados por una "mula" que viaja entre ellos y proporciona un servicio guardar – reenviar el mensaje no interactivo. Esto mismo utiliza
- DarkNet (Pentland, 2004) donde las mulas de datos son motocicletas que viajan regularmente a regiones apartadas.
- Interconexión de nodos con gran cantidad de información y largos retardos (Ott, 2006)
- "Distribución de contenidos, sincronización y seguimiento de actividades estudiantiles en un entorno de aprendizaje desconectado para el proyecto E-Lane" el cual fue realizado dentro de la Universidad del Cauca (Bravo, 2006).

Las aplicaciones objeto de adaptación pueden ser aquellas que requieran el intercambio de información de gran tamaño sin requerimientos de urgencia o tiempo real. Ejemplos de aplicaciones que pueden operar de manera

asíncrona son mensajería (correo, voz, multimedia), acceso a bases de datos (ingreso y consulta de información), carga y descarga de información de aplicación específica (telemetría, teletrabajo, tele-educación, telemedicina), entre otras.

3.7 Módulos de Globus Toolkit adaptables para operación basada en DTN

La herramienta Globus implementa una característica de tolerancia a fallas y confiabilidad que le da la capacidad al usuario Grid de controlar en tiempo de ejecución las fallas en la comunicación y los recursos computacionales para proveer acciones de recuperación o reacción (Thain, 2004).

Para esto Globus cuenta con servicios para la detección y manipulación de fallas en los módulos RFT (*Reliable File Transfer*) y GRAM (*Grid Resource Allocation and Management*).

3.7.1 Servicio RFT (*Reliable File Transfer*)

El servicio RFT se encarga de proveer la característica de tolerancia a fallas y recuperación. Esto lo hace utilizando PostgreSQL[11] donde almacena el estado de transferencia de puntos de control y marcas de fallas de transferencias transitorias, y usa mecanismos de reintentos con retroceso exponencial[12] durante la transferencia.

RFT se ha probado en la recuperación de datos durante transferencias en el caso en que existan bloqueos de servidores de origen y destino, fallas en el contenedor (cuando la maquina que ejecute el contenedor colapse) y fallas en el sistema de archivos entre otros.

A continuación se muestra un conjunto de posibles fallas que pueden ser tratadas utilizando RFT (RFT, 2009):

- **Fallas en la fuente o destino de GridFTP**: en este caso RFT reintenta la transferencia utilizando la última marca de reinicio almacenadas en la base de datos para reanudar las transferencias desde puntos previos a la falla. Una falla está asociada a un servidor, y esto significa que todas las demás transferencias hacia o desde el servidor, serán objeto de reintento. Esto permite prevenir futuras fallas de transferencia utilizando la información disponible en la base de datos (RFT, 2009).
- **Fallos de la red:** esta falla ocurre cuando existe alto tráfico en la red o por cualquier otra razón en la que se pierden paquetes o se producen desconexiones. Esta falla se considera transitoria y el RFT revisa la transferencia en cuestión con retroceso exponencial (RFT, 2009).
- **Fallas del contenedor:** éste tipo de fallas ocurre cuando la máquina que ejecuta el contenedor colapsa o si el contenedor se reinicia cuando hay transferencias activas (RFT, 2009).

RFT no maneja fallas de espacio en disco para la base de datos.

3.7.2 GRAM (*Grid Resource Allocation and Management*)

En Globus la ejecución de tareas remotas es soportada por el servicio de Administración de asignación de recursos Grid (GRAM), el cual define mecanismos para atender peticiones de ejecución de tareas, monitoreo y control de resultados de las tareas ejecutadas (Feller, 2007).

GRAM provee características de tolerancia a fallas mediante el requerimiento que impone sobre los clientes para que estos entreguen un "job contact" de tal forma que el servicio GRAM pueda utilizarlo para reconectarse con la tarea. GRAM también ofrece una opción autónoma para hacer que la información de contacto de la tarea persista y

11 PostgreSQL: www.postgresql.org

12 Retroceso exponencial: Es un algoritmo usado para espaciar o repetir retransmisiones de algunos bloques de datos, de esta forma reduce la carga e incrementa la probabilidad de éxito en la comunicación.

de esta forma pueda monitorear y controlar todos las tareas que ha creado, sin la intervención de los clientes (WS-GRAM, 2009).

3.8 Tecnología Servicios Web

Un servicio Web es un componente software, independiente del lenguaje de programación utilizado para su desarrollo, que busca la interoperabilidad entre aplicaciones utilizando estándares Web. Los componentes de un servicio Web son: UDDI (*Universal Description, Discovery and Integration*) (Bellwood, 2002) para la publicación y descubrimiento; SOAP (*Simple Object Access Protocol*) (Gudin, 2007) que brinda una forma de acceso a través de la definición de un protocolo de mensajes XML (*Extensible Markup Language*) (Bray, 2008) para el servicio básico de interoperabilidad, y un descriptor WSDL (*Servicios Web Description Language*) (Cristensen, 2001) que constituye un lenguaje común para la descripción de Servicios Web (Curbera, 2003). Con el fin de explotar la funcionalidad que estas plataformas ofrecen, actualmente se puede contar con propuestas como las que Sun Microsystems lidera (JSR 172) (J2ME, 2004), cuyo objetivo es garantizar la interoperabilidad entre servicios Web y dispositivos móviles. Dicha organización se preocupa por esta iniciativa proponiendo a este API como forma de aprovechar y unir la principal característica de los servicios Web (la integración de sistemas heterogéneos con dominios tales como *business to business* y *business to consumer*) con la movilidad que ofrece un dispositivo JME13 (*Java Micro Edition Sun Microsystems*).

3.8.1 Composición de Servicios Web

La composición de servicios Web se refiere a servicios Web complejos que internamente invocan a otros. Un Servicio Web complejo se implementa como combinación de otros servicios. Este concepto es importante porque es una buena herramienta que admite la orquestación de servicios Web publicados por las partes que intervienen en un proceso de negocio.

3.8.2 Orquestación de Servicios Web

La orquestación de Servicios Web se refiere al control secuencial de un grupo de servicios Web a través de la composición de éstos dentro de un proceso de negocio (Cor, 2006). Otra definición se encuentra en (Virdell, 2003) donde se dice que la orquestación impone orden y cronometraje sobre un grupo de servicios Web. Un ejemplo de un orquestador de WS es un motor BPEL (*Business Process Execution Language* - Lenguaje de Ejecución de Procesos de Negocio).

3.9 Herramientas para Servicios Web

3.9.1 Herramientas para crear y desplegar los servicios Web sobre la *Grid*.

- GT4IDE (Globus Toolkit 4 *Integrated Development Environment*): Una de las primeras herramientas que permitió realizar el proceso de implementación de servicios Web en una *Grid*. Se presentó como un componente para el entorno de desarrollo Eclipse[14], ofreciendo una abstracción del proyecto Globus Toolkit (GT4) para los programadores. Este proyecto se puede encontrar en (GT4IDE, 2005).

- GDT (*Grid Development Tools*): Es un paquete de componentes útiles para el desarrollo de servicios y aplicaciones, creación de flujos de trabajo y administración de la *Grid*, empaquetados como un módulo para el entorno de desarrollo integrado Eclipse y MAGE (*Marburg Ad-hoc Grid Enviroment*). GDT se divide en (MAGE-DT, 2009):

14 Eclipse: Plataforma de desarrollo de código abierto basada en Java. Es un desarrollo de IBM cuyo código fuente fue puesto a disposición de los usuarios. En sí mismo Eclipse es un marco y un conjunto de servicios para construir un entorno de desarrollo a partir de componentes conectados (plug-in).

- Generador de servicios GDT.
- Orquestador Visual de la *Grid*.
- Parte de las herramientas para la administración, despliegue y otras relacionadas con la *Grid*.

- Incubator/*Introduce*: es un conjunto de herramientas y sistemas mediadores que se han creado para soportar el desarrollo de aplicaciones en el marco de los servicios *Grid*. Permite a los desarrolladores utilizar tecnologías *Grid* sin necesidad de conocer detalles de bajo nivel (caGrid, 2009).

En este proyecto se utilizó *Introduce* como herramienta para la etapa de creación y despliegue de nuevos servicios. Esta herramienta permite crear un servicio para posteriormente utilizar un editor cualquiera para codificar la implementación de los servicios. Para la edición del código fuente existen varias herramientas, sin embargo la recomendada por *Introduce*, es Eclipse. Esto es por que *Introduce* al guardar el proyecto de servicios Web lo hace compatible con Eclipse, es decir el proyecto queda guardado en el mismo formato de un proyecto de Eclipse. De esta manera *Introduce* se presenta como una herramienta poderosa que despliega el servicio de manera muy simple sobre el contenedor de servicios Web de la *Grid*.

3.9.2 Herramientas para ejecutar los servicios Web sobre la *Grid*.

Para la ejecución se usa uno de los componentes comunes de tiempo de ejecución el cual proporciona un conjunto de bibliotecas y herramientas fundamentales que son necesarias para construir y ejecutar servicios Web (GT4FactSheet, 2005). Este componente es el Núcleo Java WS (Java WS Core): es una implementación de la familia de estándares de la Estructura de Recursos de Servicios Web (WSRF) y la Notificación de Servicios Web (WSN). Ésta provee APIs y herramientas para la construcción de servicios Web stateful[15].

3.9.3 Herramientas para consumir los servicios Web.

Los clientes requieren herramientas que les permitan consumir los servicios desplegados sobre el contenedor de servicios de la *Grid*. La gran ventaja que tienen los servicios Web es que para ser consumidos no están ligados a ninguna tecnología en particular, ni siquiera al lenguaje en el que fueron escritos, esto es por que utilizan tecnologías XML estándares como su descripción WSDL y transportes de mensajes SOAP. En el presente proyecto se realizaron pruebas con clientes Web, móviles y aplicaciones de escritorio desarrolladas en dos plataformas diferentes Java y C#[16].

A continuación se describen los diferentes clientes potenciales para los servicios Web desplegados en la *Grid*.

3.9.3.1 Cliente Aplicación de Escritorio

Una de las grandes ventajas que ofrecen los servicios Web es que los clientes como aplicaciones de escritorio pueden estar escritos en cualquier lenguaje de programación como por ejemplo C# y Java.

- C#: en este caso se utilizó visual Studio 2005 como entorno de desarrollo integrado el cual posee una herramienta que permite invocar servicios Web de manera fácil y rápida, solo basta con agregar la dirección del archivo descriptor del servicio (WSDL).

15 Los servicios *stateful* son aquellos que mantienen almacenada la información en las variables de la sesión, es decir que pueden mantener ciertos datos almacenados durante el tiempo en que el servicio esté disponible en el servidor de servicios Web.
16 C#: Visual Studio http://msdn.microsoft.com/

- Java: en este caso se utilizó NetBeans[17], este IDE (*Integrated Development Environment*) es de libre distribución y permite crear tanto aplicaciones de escritorio como aplicaciones móviles y Web. Al igual que el anterior IDE, NetBeans posee una herramienta que permite invocar servicios Web.

- Cliente Web: el cliente Web se realizó utilizando NetBeans, este cliente Web consiste de una página Web con los formularios y botones respectivos para invocar el servicio Web. Las páginas Web se despliegan sobre un servidor de aplicaciones que puede ser GlassFish[18] o cualquier otro Figura 3-9, Figura 3-10.

Figura 3-9 Prueba del servicio Web.

Figura 3-10 Prueba del servicio Web.

17 NetBeans: Es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación.

18 GlassFish: Es un servidor de aplicaciones que implementa las tecnologías definidas en la plataforma *Java Enterpise Edition* y permite ejecutar aplicaciones que siguen esta especificación. Es gratuito y de código libre.

- Cliente Móvil: Para el caso de los clientes móviles una invocación directa del servicio Web es compleja, por estar razón fue necesario implementar un servicio Web en Java y desplegarlo en GlassFish el cual invoca al servicio Web de la *Grid* y pasa la información obtenida a los clientes, es decir que se tiene un servidor de aplicaciones intermediario. Los clientes móviles invocan al servicio Web intermediario y este a su vez invoca al servicio Web de la *Grid*. Para los clientes móviles se utilizó el lenguaje de programación Java en su versión micro JME, la cual es la versión adaptada para dispositivos con bajas capacidades tanto en procesamiento cómo gráficas ver Figura 3-11 JME es la abreviatura de *Java Micro Edition*, el cual es en su mayoría un "subconjunto" del lenguaje de programación Java en su versión estándar JSE (*Java Standard Edition*). En la prueba se utilizó este lenguaje y una herramienta llamada *Stubs Generator* del *Wireless Toolkit* (WTK). Esta última se utilizó en el proceso de generación del código que permite invocar los servicios Web.

Figura 3-11 Vista de la aplicación en un móvil

3.10 Proyectos relacionados y casos de implementación.

Alrededor del campo de la educación y la investigación existen varias iniciativas que hacen uso de *Grid* para mejorar el aprendizaje en dichos campos. Estas iniciativas apuntan en la misma dirección para los fines de este proyecto, ya que todas soportan el desarrollo de sus objetivos, en el uso de la tecnología *Grid*.

Básicamente se encuentran trabajos e iniciativas para el soporte de *e-learning*[19], *t-learning*[20] (López, 2005), (Li, 2007), (Gaeta, 2005), (Capuano, 2005), donde problemas tradicionales en *e-learning/t-learning* pueden encontrar soluciones adecuadas de los desarrollos adelantados alrededor de *Grid* y viceversa, entre ellos se pueden mencionar los siguientes proyectos:

- ELeGI (*European Learning Grid Infrastructure* - Infraestructura Grid para el Aprendizaje Europeo) (Stefano, 2005), (ELeGI, 2008) que busca avanzar radicalmente en el uso eficaz de la tecnología para mejorar el aprendizaje en Europa a través del diseño, implementación y validación de una arquitectura software

19 E-learning: Educación mediada por las TIC para apoyar procesos educativos.
20 T-learning: Aprendizaje a través de TV interactiva o un dispositivo similar basada en pantalla que maneja contenidos ricos en video entregados a través de una o varias plataformas (satélites, cable, etc).

orientada al servicio, conducida a la pedagogía basada en tecnologías *Grid* para soporte de aprendizaje ubicuo, colaborativo, experimental y personalizado.

- IEEE LOM (*Learning Object Metadata* - Metadatos para Objetos de Aprendizaje) (Neumann, 2005) para metadatos descriptivos, permitiendo búsquedas por contenido.

- IMS GLC (*IMS Global Learning Consortium* - Consorcio de Aprendizaje Global IMS) (IMS, 2008) con contenido empaquetado para intercambio de LOs (*Learning Objects* - Objetos de Aprendizaje) entre herramientas y entornos de aprendizaje.

- AICC (*Aviation Industry CBT Committe* - Comité CBT de Industria de Aviación) (AICC, 2008) para la comunicación en tiempo real entre el contenido y el entorno de aprendizaje.

- ADL (*Advanced Distributed Learning* - Aprendizaje Distribuido Avanzado) (ADL, 2007) iniciativa en colaboración entre el gobierno, la industria y la academia de los Estados Unidos para especificar un entorno que permita la interoperabilidad de herramientas, contenidos y cursos de aprendizaje.

- SCORM (*Sharable Content Object Reference Mode* - Modelo de Referencia de Objeto de Contenido Compartible) como un modelo de referencia para integrar las especificaciones citadas anteriormente, SCORM es una colección de especificaciones adaptadas de múltiples fuentes para proveer un conjunto completo de capacidades *e-learning* que permitan interoperabilidad, accesibilidad y reusabilidad de contenido *e-learning* basado en Web.

Todas estas iniciativas, entre otras, apuntan al mejoramiento en el aprendizaje de un sector socioeconómico determinado como lo son las Universidades, Institutos y Centros de Investigación, donde se cuenta con ciertas condiciones de infraestructura adecuadas para llevar a cabo estos proyectos. La realidad es que no siempre se cuenta con este tipo de infraestructuras TIC, generalmente estas deficiencias en infraestructura y condiciones en torno a las TIC son problemas reales que se viven en los países en desarrollo, es por ese motivo que este proyecto permite abordar necesidades específicas diariamente vividas en estos contextos, particularmente con los computadores de bajos recursos informáticos y los deficientes accesos a Internet.

En Colombia, ya existen iniciativas *Grid*, tal es el caso de *Grid* Colombia (Grid-Colombia, 2006) la cual es una iniciativa de investigación que intenta unir esfuerzos para crear una *Grid* con propósitos académicos. *Grid* Colombia une esfuerzos con RENATA (Red Nacional Académica de Tecnología Avanzada) (RENATA, 2009) la cual es una red avanzada a nivel nacional que conecta centros de investigación y universidades, ésta a su vez está conectada a sí misma con CLARA (Cooperación Latino Americana de Redes Avanzadas) (CLARA, 2009) que de igual forma es una red de centros de investigación.

Otro proyecto importante para el trabajo en *Grid Computing* adelantado actualmente en Latino América es el EELA-2 (*E-science grid facility for Europe and Latin America* – Facilidades *Grid* en E-science para Europa y América Latina) (EELA-2, 2009) que tiene por objetivo la construcción y producción de infraestructura *Grid* escalable de alta capacidad y calidad, proporcionando acceso las 24 horas del día y a nivel mundial a computación distribuida, almacenamiento y recursos de red necesarios para un amplio espectro de aplicaciones científicas de Europa y América Latina, garantizando su sostenibilidad a largo plazo. Este proyecto tuvo su primera fase con el nombre de EELA (*E-infrastructure Shared between Europe and Latin America* - E-infraestructura compartida entre Europa y Latino América) (EELA, 2008), en el cual participaron 10 países con sus redes académicas y diferentes instituciones entre universidades y centros de investigación, su objetivo fue llevar las e-Infraestructuras de los países latinoamericanos al nivel de explotación de los países Europeos. La EELA se beneficia del estado maduro del proyecto ALICE (ALICE, 2009) (América Latina Interconectada Con Europa) y de la red CLARA.

Grid inicialmente fue creada para solucionar un problema particular en el desarrollo mundial (mejorar el procesamiento, almacenamiento y comunicación de información para la investigación científica), pero su función

también encaja perfectamente para ser empleada con el fin de mejorar el acceso y uso de las TIC en determinadas regiones del mundo donde las TIC están siendo subutilizadas.

3.10.1 Casos de implementación

- **unGrid** Proyecto supercomputador distribuido de la Universidad Nacional de Colombia (unGrid, 2009) unGrid está construida con base en la tecnología JavaSpaces, uno de los servicios provistos por Jini. Jini a su vez es un conjunto de especificaciones (APIs) y protocolos de red para la construcción de sistemas distribuidos orientados a servicios. Jini es también un modelo de programación basado en el registro, búsqueda y descubrimiento de servicios, transacciones y eventos remotos. JavaSpaces es un servicio de la plataforma Jini y se puede considerar como un espacio de memoria distribuida compartida.

El Proyecto unGrid busca construir un supercomputador a partir de la red de computadores con que cuenta la Universidad Nacional en su sede de Bogotá, esta idea es similar a la idea de integrar todos los centros informáticos de CPE y crear una gran malla para el intercambio de información y recursos computacionales, en unGrid usan una tecnología basada en Java Jini, una tecnología que usa principalmente JavaSpaces, eventos distribuidos, descubrimiento de servicios y multicast. JavaSpaces está creado con base en RMI (Remote Method Invocation), lo cual quiere decir que solo se puede utilizar para invocar métodos de programas escritos en Java y los clientes deben ser Java (Taylor, 2004), similar a esta tecnología se encuentra Globus Toolkit que ofrece almacenamiento distribuido, transferencia distribuida de archivos, descubrimiento de servicios e incorpora a Java como uno de los lenguajes dentro de la plataforma además de C o Python, está orientado hacia los servicios Web, quienes ofrecen interfaces estándares que permiten que sus operaciones sean invocadas a través de Internet desde cualquier aplicación escrita en cualquier lenguaje de programación. Por lo anterior Globus es usada en el presente trabajo pues es la herramienta de mayor aceptación en Grid, es catalogada como la herramienta usada por defecto para el desarrollo de Grid en Estados Unidos, además cuenta con una extensa documentación y soporte.

- **Implementación de un cluster homogéneo para la resolución de problemas de alta complejidad computacional** en la Universidad Nacional sede Medellín (Branch, 2008), proyecto desarrollado con la ayuda de 6 computadoras homogéneas Pentium 4 1.5 GHz y RAM 512MB, la herramienta para la creación del cluster es openMosix.

El desarrollo del Cluster computacional en este caso se fundamenta para la resolución de problemas de alta complejidad donde se usaron computadores homogéneos a diferencia del presente trabajo, donde se usa computadores heterogéneos para la solución con la función principal de optimización en la capacidad de procesamiento y almacenamiento de los centros informáticos de CPE, para la distribución de carga trabajo de aplicaciones educativas, redes educativas, información de proyectos y demás aplicaciones relacionadas con las instituciones adscritas al programa CPE. En ambos casos se utiliza como herramienta de creación del cluster a openMosix, herramienta que presta grandes beneficios en la gestión del cluster y distribución de procesos, entre otros.

- **Reacciun2** (Reacciun2, 2009): La nueva red avanzada que soporta su funcionamiento en una consolidada arquitectura de hardware Sun que incluye 88 estaciones de trabajo W2100z (AMD 2.6GHz, D.D 146GB, RAM 16GB), 18 servidores Sun Fire V40z (AMD 3GHz, D.D 300GB, RAM 64GB), 2 servidores Sun Fire V890 (AMD 2.1GHz, D.D. 146GHz, RAM 128GB) y una solución cluster grid que será utilizada como el nodo del Primer Grid Nacional de Venezuela.

  Esta red en su primera fase cuenta con un ancho de banda de 34 megas, y permitirá a sus usuarios realizar proyectos de investigación y desarrollo e intercambio de información conjuntamente con otras universidades que también cuentan con Redes Avanzadas en los Estados Unidos, Europa y América Latina.

Las investigaciones e intercambios de conocimientos que se servirán de Reacciun2 facilitarán de inmediato potenciar avances importantes en las áreas de Tele-Salud, Tele-Educación, Bibliotecas Digitales, Redes de Telecomunicaciones y Videoconferencias de Alta Calidad.

Como se puede observar los recursos computacionales de esta solución está conformada por equipos de computo de gama alta con muy buenas características tanto en almacenamiento como de procesamiento, con una infraestructura de red de extenso ancho de banda, a fin de brindar soporte en áreas relacionadas con la educación, al igual que el presente proyecto son la gran diferencia que en este caso se utilizaran recursos computacionales de más bajas prestaciones e infraestructura de red que presenta retardos e interrupciones.

- **SENIAT** (SENIAT, 2009) el Servicio Nacional Integrado de Administración Aduanera y Tributaria de Venezuela realizó una importante inversión de 2 millones de dólares para modernizar su plataforma tecnológica con el propósito de reducir sus costos operacionales y atender mejor las necesidades de sus usuarios al migrar sus antiguos sistemas Compaq/HP DS20, Alpha 8400, ML 370 bajo sistema operativo TRU64, a una renovada plataforma de infraestructura basada en los servidores Sun FIRE Cluster corriendo el sistema operativo Solaris y el motor de base de datos Oracle9i.

Esta solución como muchas otras existentes alrededor de cluster busca optimizar los costos operacionales relacionados con la capacidad de procesamiento de la Institución, algo similar es necesario solucionar con el presente proyecto, pues se necesita optimizar la capacidad de procesamiento en los centros informáticos de CPE pero a diferencia de esta solución se debe evitar incurrir en costosas actualizaciones o compras de equipos.

- **Sistema de información para el manejo de datos moleculares en café** (Rivera, 2008) en el centro de investigaciones de café CENICAFE de Colombia, proyecto desarrollado para el manejo adecuado de la gran cantidad de información que en la actualidad se obtiene en los proyectos de estudio de genomas, se utilizaron equipos tales como 4 IBM (AMD 2.4 GHz, RAM 4GB, 2 D.D. 74GB) 1 IBM (Intel 3.6 GHz, RAM 5GB, 6 D.D. 300GB), 1 IBM (2 Intel 3 GHZ, RAM 4GB, 6 D.D. 74 GB), 1 SunFire v240 (1.5 GHz, RAM 16GB, 2 D.D. 146 GB), 1 Power MAC (2 G5 2.7GHz, RAM 2GB, 2 D.D. 250GB), 1 Apple (Xeon 2GHz, RAM 4GB, D.D. 750 GB), 2 IBM Intellistation aPro, para la construcción del cluster HPC se usaron bibliotecas LAM-MPI y para el sistema Grid se uso Sun Grid Engine (SGE).

Esta solución muestra claramente una infraestructura bastante moderna para el montaje del cluster, posee características de procesamiento y almacenamiento altas caso contrario a la infraestructura de nuestro trabajo, además la solución para el sistema Grid se desarrollo bajo la herramienta SGE la cual es una herramienta comercial propietaria, a diferencia de la herramienta Globus Toolkit de código abierto usada en el presente trabajo.

4 Capítulo 4. Diseño del *Cluster* y la Grid – DTN con el contenedor de servicios Web

En este capítulo se describen los principales pasos en el diseño y construcción de un *Cluster*, una *Grid* y un servicio Web. Cabe resaltar que ciertos pasos de diseño se ejecutan de manera similar en los dos casos de *Cluster* y *Grid*.

4.1 Diseño de un *Cluster* de computadores

Después de la anterior descripción de los diferentes tipos de *Cluster* y herramientas utilizadas para su construcción, se relata el conjunto de pasos más relevantes y comunes en la construcción de un *Cluster* (Calicut, 2006).

- **Primer paso:** tener claro el objetivo general para el cual se va a construir el *Cluster*, es decir, las necesidades que busca solucionar, las funcionalidades que presta, la disponibilidad, las características de las aplicaciones que se ejecutarán sobre él, los usuarios que va a soportar en un determinado tiempo, y demás detalles que se vayan a tener en cuenta al momento de gestionarlo.
- **Segundo paso:** se decide el tipo de *Cluster* o arquitectura más adecuada para solucionar eficazmente las necesidades descritas en el primer paso.
- **Tercer paso:** selección del hardware a utilizar bien sea para adquirir o reutilizar recursos hardware. En éste último caso, el paso de selección de hardware adquiere especial importancia puesto que la reutilización de herramientas podría hacer parte de los objetivos de construcción de un *Cluster*, tal como ocurre en el presente trabajo.
- **Cuarto paso:** sistema operativo a utilizar, el software de montaje del *Cluster* y otras aplicaciones software.
- **Quinto paso:** adecuación del lugar donde será ubicado el *Cluster*, este paso no siempre es necesario, pero es muy importante ya que de esto depende el correcto funcionamiento de los equipos.

Si bien éste sería el orden lógico no siempre es el único, pues muchas veces el diseño debe regirse a cierto tipo de hardware existente o ciertas restricciones en sistemas operativos o software ya establecidos.

Una visión de dichas arquitecturas se encuentra en el capítulo 3 de este documento, por tanto la información necesaria para la toma de decisiones y diseño de los pasos uno, dos y cuatro, se detalla en dicho capítulo y la información necesaria para los pasos tres y cinco se describe a continuación.

4.1.1 Componentes hardware para un *Cluster*

La clave para construir un *Cluster* esta en el desempeño de los terminales que conforman los nodos y la red de interconexión para la transferencia de datos entre ellos. Se han logrado avances significativos entorno a ellos en las últimas décadas, los *Cluster* han sido casi siempre creados por usuarios o empleados al interior de las organizaciones, en los últimos años existen soluciones *Cluster* propietarias para ciertas especificaciones de usuario, desde la estructura hardware del *Cluster* hasta la estructura externa como torres, *racks*, sistemas de potencia y refrigeración.

4.1.2 Hardware del nodo del *Cluster*

Un nodo provee un sistema de cómputo y una capacidad de almacenamiento que integran varios subsistemas en una simple unidad. Ejemplo de estos subsistemas son:

- **Procesador:** La selección del procesador esta ligada a la selección de la tarjeta madre del nodo por compatibilidad y si se decide utilizar más de un procesador por nodo se debe tener en cuenta una arquitectura SMP y un acceso compartido a memoria que influye directamente en la programación empleada para el

Cluster. Se recomienda para este tipo de nodos utilizar dispositivos comerciales de compañías con experiencia en multiprocesadores (Sterling, 2000).

En ciertos casos como se comento anteriormente es probable que ya se cuente con la existencia de equipos lo cual conlleva a disponer de máquinas de diversas arquitecturas (AMD, Intel, entre otras), generaciones (486, Pentium, Pentium II, entre otras), fabricantes (Compaq, Dell, IBM, entre otras) o una mezcla de nodos multiprocesador y uniprocesador, de hecho existen diferentes configuraciones posibles; en estos casos lo más importante es conocer que tipo de aplicaciones software permiten aprovechar esas características particulares de cada sistema.

- **Memoria:** Lo más recomendable es que se use una memoria con la velocidad de acceso más rápida disponible en el mercado. Si el *Cluster* incluye nodos SMP.

- **Dispositivos de almacenamiento:** se puede construir un *Cluster* cuyos nodos obtengan la información de arranque por Red en lugar de almacenarla en disco duro, esto es posible si la placa base tiene la opción de arranque por red esto facilita además el mantenimiento (fallas hardware, reemplazo de discos, entre otros) y la disminución de consumo de potencia. (Muñoz, 2006).

- **Periféricos:** Existen diversidad de periféricos pero quizá los más usados para estos casos son el monitor, teclado, *mouse*, cámaras y unidades de almacenamiento extraíble. Este tipo de dispositivos deben ser conectados al nodo mediante puertos externos, por ejemplo puertos tales como USB, PS2, RS232, HDMI, entre otros. Si en determinado caso existen varios nodos contiguos se recomienda utilizar un solo monitor, teclado y *mouse*, para ello se utiliza una consola serial o un *switch* KVM (*KeyBoard* Video Mouse, Teclado Video Ratón), lo cual permite optimizar el espacio, organización, y consumo de potencia.

4.1.3 Tecnologías de red

La compatibilidad del hardware de red con otro hardware *Cluster* y sistemas software es el mayor factor en la selección del hardware de red.

Un amplio y creciente rango de posibilidades esta disponible para los diseñadores de *Cluster* cuando de tecnologías de interconexión se trata. De igual manera los costos relativos al hardware de red pueden variar haciendo que la selección de estos dispositivos impacte en el costo general del *Cluster*. Por esta razón se prefiere construir *Cluster* con productos económicos que típicamente se encuentran en redes de área local (Apon, 2001).

4.1.4 Ubicación del *Cluster*

La adecuación del *Cluster* depende de ciertos factores, entre ellos el número de computadores, el cableado, la refrigeración, la fuente de alimentación, el acceso físico, entre otros.

Hay que tener especial cuidado al manejar la relación espacio y accesibilidad, ya que muchas veces al tratar de ahorrar espacio se apilan computadores de tal manera que complican el acceso físico para reparaciones y obstruyen la disipación del calor (Joseph, 2004).

Comúnmente la organización de computadores se realiza con la ayuda de estantes prediseñados para estos propósitos, dichos estantes suelen denominarse *racks*.

4.1.5 Refrigeración del sistema

La evacuación del aire caliente en estos sistemas es de adelante hacia atrás, esta circulación es vital ya que los equipos producen gran cantidad de calor y es recomendable mantener los equipos por debajo de los 21°C.

La humedad también es un factor crítico, con una humedad alta, existe mayor probabilidad de condensación y con una humedad baja, existe mayor probabilidad de generarse electricidad estática. Las gamas recomendadas son alrededor del 40% y 60% (Joseph, 2004).

4.1.6 Fuentes de alimentación

Para estimar el consumo de potencia de un *Cluster* es ideal realizar un inventario e incluir todo lo que pueda consumir energía eléctrica. Es recomendable contar con una batería o UPS (*Uninterruptible Power System*) para evitar la pérdida de información cuando ocurran fallas eléctricas.

En general, cuando se desee realizar una correcta adecuación de los equipos de cómputo para un *Cluster*, se deben tener presente los siguientes puntos esenciales (Muñoz, 2006):

- Cuando se seleccione el lugar en el cual se ubicarán los equipos, debe quedar suficiente espacio para que una persona pueda realizar tareas de mantenimiento, actualizar hardware y trabajar con el *Cluster* (programarlo).
- Si el número de equipos lo amerita, adicionar un sistema de enfriamiento para evitar daños por calor.
- Asegurarse de que los cables de potencia estén bien instalados y no haya ningún peligro de cortos o sobrecargas en el sistema.
- Si es posible ubicar los nodos y equipos de red en *racks* o anaqueles (si se utilizan torres) para facilitar el acceso a los componentes.
- Hay que estar seguros de que el piso puede resistir el peso del número de nodos y todos los elementos que componen el *Cluster*. Esto es muy importante al usar *racks* porque todo el peso de los equipos se concentra en un pequeño espacio.
- Si se va a instalar una gran cantidad de nodos se puede considerar un piso falso, en el cual se introduzca la mayoría del cableado de potencia y de interconexión del *Cluster*, para una mayor organización y para evitar accidentes con los cables esparcidos.
- En general se debe tener en cuenta las especificaciones en las normas ANSI (ANSI, 2009) EIA/TIA (TIA, 2009) sobre cableado estructurado.

4.2 Diseño y construcción de una *Grid*

Al igual que el diseño y construcción de los *Cluster* las *Grid* siguen un conjunto de pasos muy similares, por esta razón esta sección detalla la arquitectura básica y las principales configuraciones de la *Grid* que se utilizará en el presente proyecto.

A diferencia del *Cluster* el cual se ubica básicamente dentro de una red de área local con equipos de igual o similares arquitecturas, la *Grid* lo hace en lugares regionalmente distantes, es decir mientras que en el *Cluster* la red es una red de área local, en una *Grid* la red es Internet, la cual presenta heterogeneidad de equipos como de dispositivos de red. Esto se convierte en un reto para una *Grid* que debe compartir recursos de manera uniforme, transparente, segura y eficiente. Para esto, una *Grid* debe cumplir con los siguientes tres puntos (Foster, 2002):

1. Coordinar recursos que no estén sujetos a un control centralizado.
2. Utilizar interfaces y protocolos de propósito general, abiertos y estándares.
3. Desarrollar calidades de servicios no triviales.

Además se debe tener en cuenta la ubicación geográfica ya que de acuerdo a ello existen varias configuraciones de *Grid* como se puede ver en la Figura 4-1. (Tarricone, 2004).

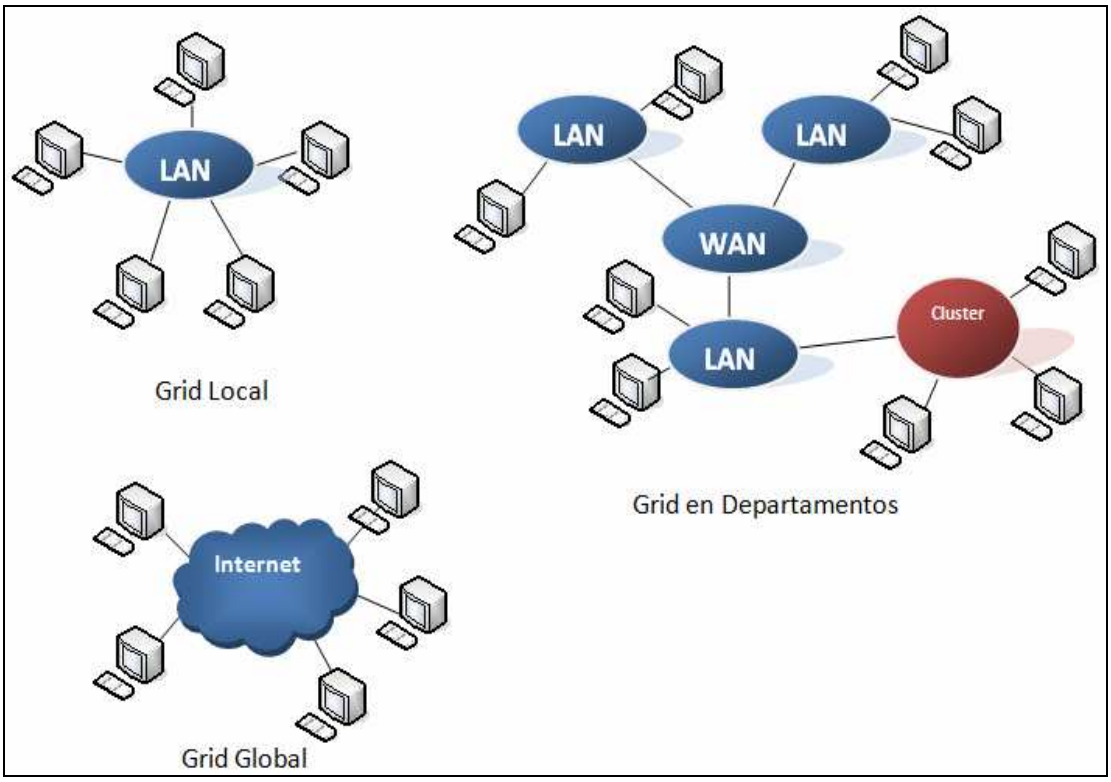

Figura 4-1 Configuraciones típicas de Grid

Otro aspecto importante es conocer la arquitectura general de la *Grid*, la cual está formada por cuatro niveles como se aprecia en la Figura 4-2.

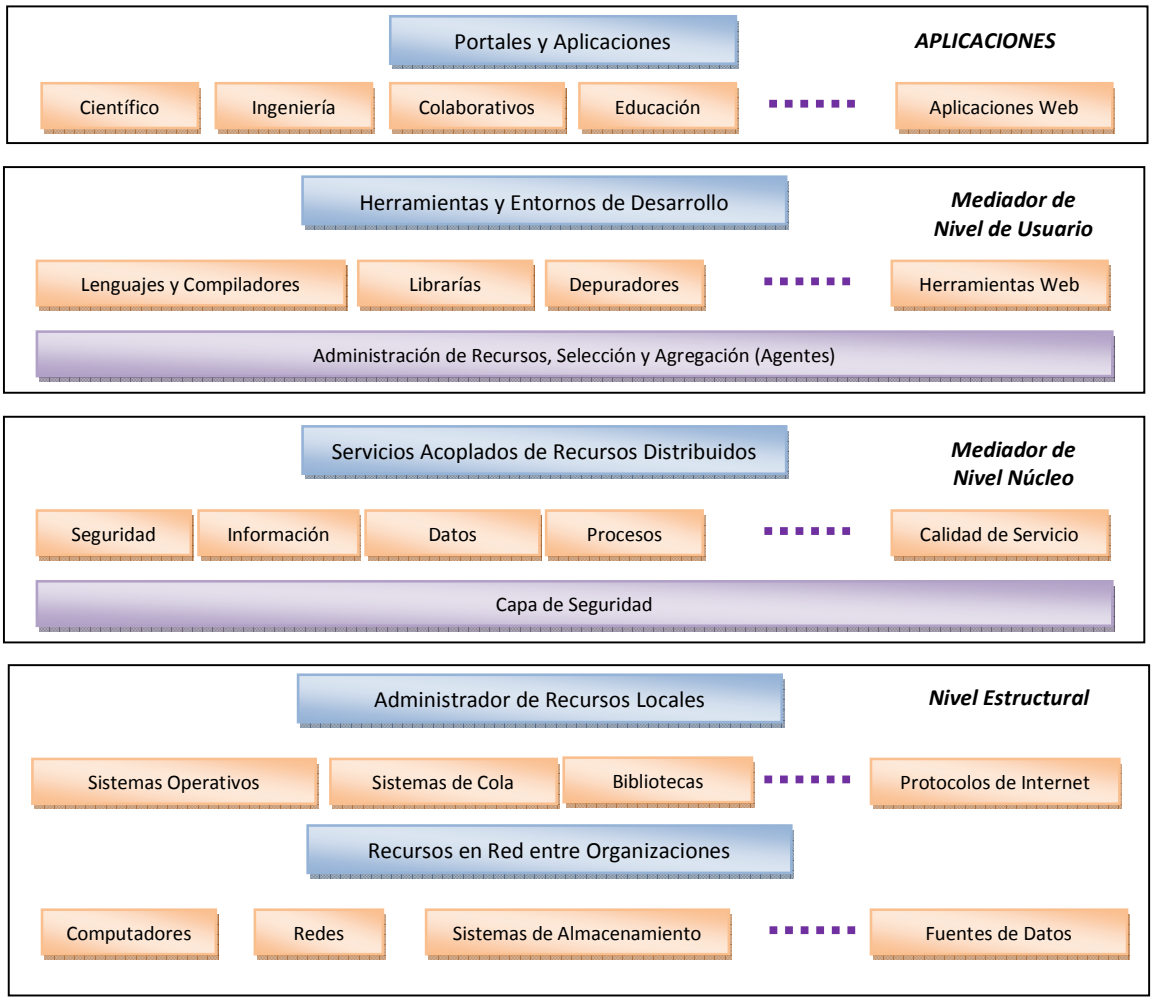

Figura 4-2 Estructura en niveles de un entorno de *Grid*

Se muestra la pila de hardware y software dentro de una arquitectura típica de *Grid* la cual consiste de cuatro niveles: Estructural, Mediador de nivel de núcleo, Mediador de nivel de usuario y el Nivel de portales y Aplicaciones de la *Grid*.

- **El nivel estructural** está constituido por recursos distribuidos tales como computadores, redes y dispositivos de almacenamiento. Los recursos computacionales representan múltiples módulos tales como *Cluster*, supercomputadores, servidores y PC's ordinarios, con sistemas operativos (variantes de Linux o Windows).

- **El nivel mediador de núcleo *Grid*** ofrece servicios tales como administración de procesos remotos, ubicación colaborativa de recursos, acceso de almacenamiento, registro de información, descubrimiento, seguridad y aspectos de calidad de servicio tales como reservación de recursos. Esos servicios abstraen la complejidad y heterogeneidad del nivel estructural ofreciendo un método consistente para acceder a los recursos distribuidos.

- **El nivel mediador de nivel de usuario *Grid*** utiliza las interfaces presentadas por el mediador de bajo nivel para ofrecer servicios y abstracción de mayor nivel, aquellos incluyen entornos de desarrollo de aplicaciones, herramientas de programación y agentes de recursos.

- **El nivel de portales y aplicaciones *Grid*** es el que interactúa directamente con los usuarios finales y se desarrolla típicamente utilizando lenguajes orientados a *Grid* y utilidades como HPC++[21] o MPI. Los portales *Grid* ofrecen servicios de aplicaciones sobre la Web, de esta manera los usuarios pueden solicitar y obtener resultados de sus tareas sobre recursos remotos (Asadzadeh, 2004).

4.3 Arquitectura del Sistema

La arquitectura del sistema se desarrolló bajo las circunstancias del problema central, donde se busca optimizar los recursos computacionales de algunos centros informáticos, estableciendo una comunicación entre ellos que tenga en cuenta los retardos e interrupciones en la conexión de la red.

Un punto importante para la construcción de un *Cluster* es la determinación del hardware disponible para la construcción del *Cluster*, en el caso del presente proyecto se encontró que en cada centro informático se cuenta con un número que oscila entre diez y veinte computadores heterogéneos con conexión a red, algunos con unidad de disco compacto y por lo menos tres de ellos con buenas características (procesador igual o superior a 1.2 Mhz, RAM 1GB, D.D. 30 GB).

Con base en lo anterior se plantea un sistema basado en una arquitectura de doble capa, cada una de las capas se denomina capa *Cluster* y capa *Grid* respectivamente. La capa *Cluster* es aquella donde los computadores de un centro informático están organizados en *Cluster*, para la optimización de recursos computacionales. Estos *Cluster* a su vez estarán interconectados a otros *Cluster* de centros informáticos mediante la denominada capa *Grid* que permite la comunicación para la adecuada distribución de información, aplicaciones y recursos computacionales con un alto grado de tolerancia entre *Cluster* geográficamente distantes cuya red de interconexión es Internet.

La capa *Cluster*, es la capa encargada de manejar los computadores de las salas de cada centro informático. Como se menciona anteriormente cada centro contará con al menos tres computadores con características computacionales favorables, uno de los cuales será el equipo servidor del *Cluster* y el resto de equipos serán los nodos y clientes *Cluster* **Figura 4-3**.

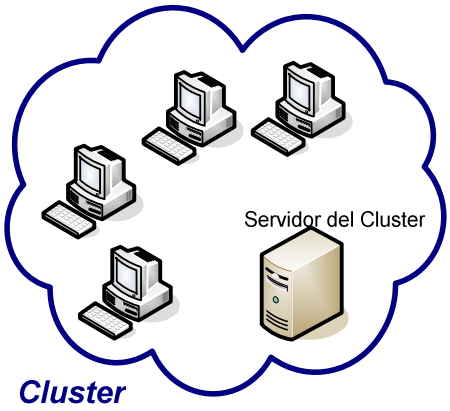

Figura 4-3 Capa *Cluster*

En esta capa se realizará la distribución de procesos dependiendo de la carga de trabajo que en cierto momento lleguen a tener los computadores que hagan parte del *Cluster*, esto es, si en cierto momento un computador que esta consumiendo muchos recursos tales como memoria RAM, procesador o disco duro, el sistema detecta esta

21 HPC++: *High Performance* C++ consiste de una biblioteca C++ desarrolladas por el consorcio HPC++ para soportar un modelo estándar para la programación portable y paralela en C++.

situación e inmediatamente redistribuye la carga, representada en procesos de cómputo, a otro computador que este ocioso o al menos con poca carga de trabajo.

Generalmente los *Cluster* requieren de arquitecturas hardware y software homogéneas, sin embargo actualmente existen herramientas que permiten la construcción de *Cluster* heterogéneos, por ejemplo se pueden tener nodos con sistema operativo Linux o Windows, en este último caso se instala una máquina virtual cooperativa, que contiene el sistema operativo Linux y las aplicaciones a utilizar. Es preferible que los servidores *Cluster* trabajen bajo el sistema operativo Linux por seguridad y estabilidad.

La implementación de esta capa se realiza utilizando el concepto de múltiples máquinas físicas que contribuyen con su potencia de procesamiento y se comportan como un solo sistema denominado *Cluster* SSI o Imagen Simple de Sistema. En otras palabras un usuario que esté conectado al *Cluster* tendrá la sensación de estar trabajando en un equipo de buenas capacidades. En esta capa, para el desarrollo del *Cluster*, se eligió un sistema conocido cómo openMosix (Ver Anexo A) y cosMos (Ver Anexo B) tras diversas pruebas y análisis con otras distribuciones tales como coMosix, coLinux (coLinux, 2009), Condor (Condor, 2009) y OpenSSI (OpenSSI, 2006). OpenMosix trabaja como una extensión del *kernel* de Linux y permite el perfecto agrupamiento en *Cluster*s y balanceo de carga de potencia de procesamiento entre sistemas interconectados en una red.

La capa *Grid* encargada de interconectar los *Cluster* distribuidos geográficamente, esta conformada por un servidor externo independiente a los *Cluster*, este servidor estará geográficamente ubicado en cualquier centro informático o Institución con mejores características de conectividad a Internet, a él tendrán acceso los servidores de la capa *Cluster* para soportar características de comunicación de una *Grid*, características de tolerancia a fallas y compartición de aplicaciones. En la Figura 4-4 se puede observar claramente esta arquitectura donde se representan por nubes rojas los *Cluster* que conforman la capa *Cluster* y por una nube azul los equipos que conforman la capa *Grid*, además se puede observar que solo un equipo de cada *Cluster* hará parte de la *Grid*.

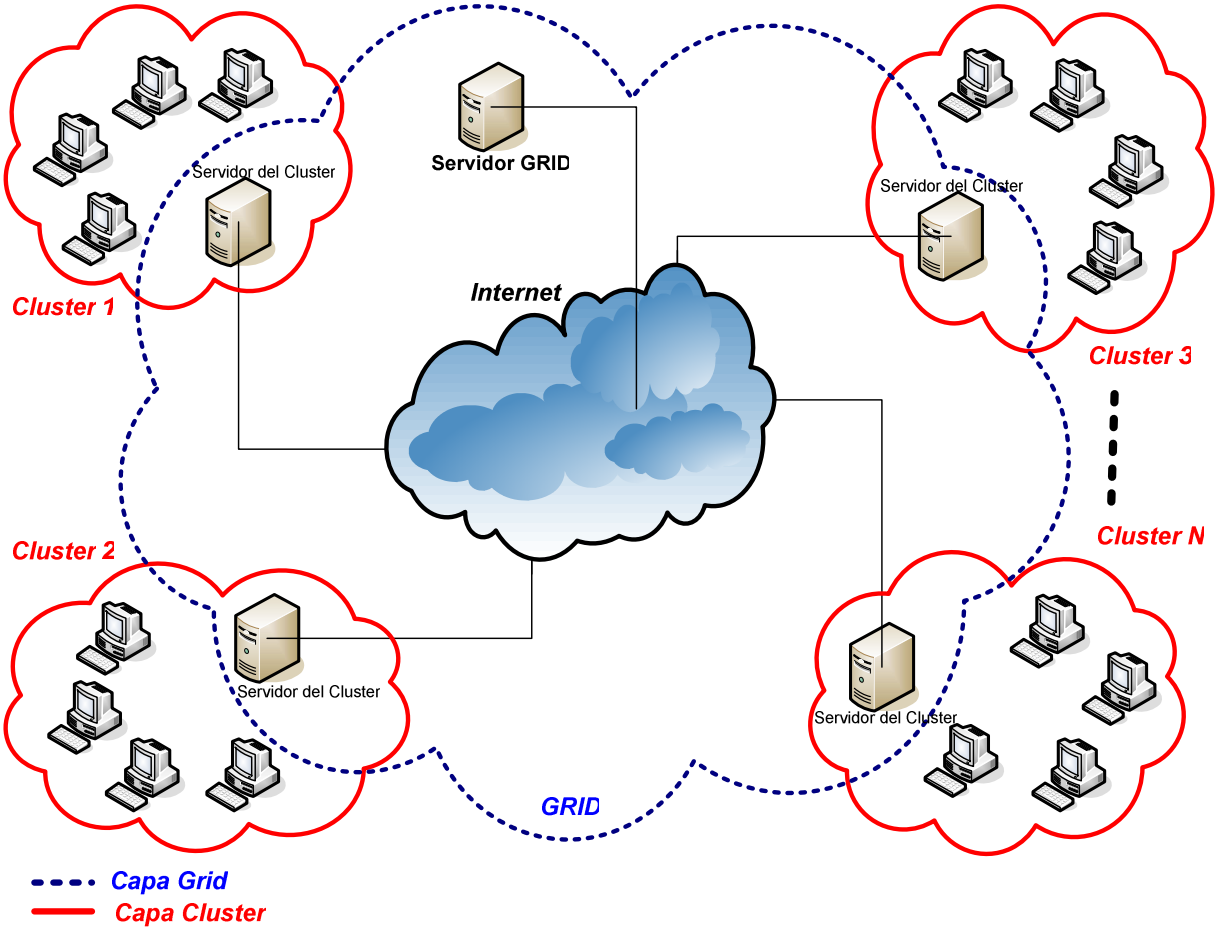

Figura 4-4 Arquitectura por capas

Como se mencionó anteriormente, entre ellos se tienen Globus Toolkit (Anexo C), Unicore, Gridbus, Legion, entre otros, para este caso se utiliza Globus Toolkit, una de las herramientas más aceptadas por la comunidad investigativa, puesto que Globus es un *middleware* de libre distribución, que permite a las aplicaciones ser empleadas en una red de recursos heterogéneos y distribuidos definiendo los servicios y capacidades requeridas básicas para construir una *Grid* computacional apoyando así a la creación de sesiones de trabajo colaborativo, *e-learning*, redes educativas, aplicaciones para el intercambio, publicación y gestión de información educativa que requieren mayor capacidad de procesamiento y almacenamiento, manteniendo el nivel de calidad de servicio a pesar de las limitadas condiciones del equipo y de conectividad a redes, entre muchas otras aplicaciones distribuidas.

Las tecnologías utilizadas para la implementación de cada capa son diferentes, sin embargo son transparentes para el usuario final. De esta manera un usuario puede utilizar cualquier computador dentro de un *Cluster* sin importar sus características computacionales a fin de ejecutar determinadas aplicaciones distribuidas y compartir información.

Cabe resaltar que las capas descritas anteriormente son independientes entre sí, esto quiere decir que los *Cluster* funcionarán aun sin la existencia de la *Grid* y la *Grid* funcionará sin necesitar que todos los nodos *Cluster* y clientes *Cluster* estén conectados, esto con el fin de ofrecer de manera fácil la adición y remoción de computadores al sistema.

5 Capítulo 5. Ambientación Experimental

Esta solución que integra tecnologías *Cluster, Grid*, DTN y servicios Web se monta en máquinas con características físicas heterogéneas, con software no comercial y sistemas operativos Linux y Windows. Antes de definir los requisitos del servicio Web es necesario definir el software para el *Cluster* y la *Grid*. En este capítulo se describe el *Cluster* configurado, la *Grid*, el software seleccionado y posteriormente los requerimientos definidos para el servicio Web, por último se describen las pruebas de ejecución para un servicio Web prototipo diseñado en el contexto de CPE-Unicauca, dicho prototipo define un repositorio de Proyectos Institucionales que manejan las instituciones educativas del programa CPE-Unicauca al cual se puede ingresar consumiendo un servicio Web mediante clientes de escritorio y móviles, dicha infraestructura es adaptable a redes educativas como moodle en Grid.

5.1 Planteamiento Experimental

La solución planteada está compuesta por un sistema de doble capa: capa *Cluster* y capa *Grid*. Estas capas se encuentran soportadas en dos redes, la capa *Grid* está sobre Internet, para cumplir con su objetivo de distribución geográfica de los equipos que la constituyen y la capa *Cluster* se encuentra soportada en redes de área local debido a que la distribución de procesamiento para la optimización de recursos computacionales requiere tiempos cortos de respuesta de la red. La distribución por capas del sistema se puede observar en la Figura 5-1.

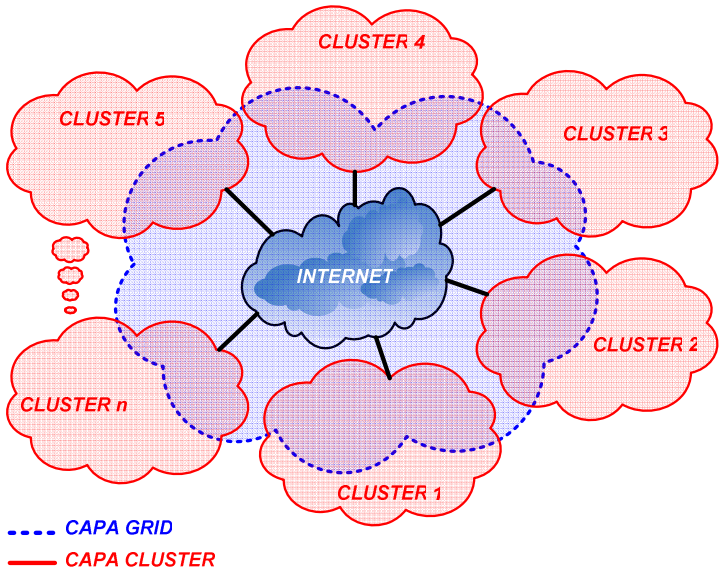

Figura 5-1 Sistema doble capa (*Cluster* y *Grid*)

Las nubes encerradas por contornos de color rojo representan las redes de área local en las que se encuentran ubicados los *Cluster*. La nube encerrada por el contorno de color azul representa a Internet que es la red en la cual se encuentra soportada la *Grid*. Nótese además, que parte de las nubes que representan al *Cluster* se encuentran también dentro de la nube de la *Grid*, esto es así debido a que el computador central de cada uno de los *Cluster* debe pertenecer también a la *Grid* para que los demás equipos del *Cluster* puedan acceder a los servicios de la *Grid* a través de él.

5.2 Descripción de la capa *Cluster* experimental

Se describe a continuación la configuración hardware de los equipos y después el software seleccionado para la implementación del *Cluster*. En esta capa es donde se realiza la distribución de procesos para la optimización de recursos computacionales.

El *Cluster* configurado consta de 3 equipos con Windows XP sin Globus y 3 equipos con Debian Sarge y Globus, dichos equipos tienen las siguientes características hardware, ver Figura 5-2:

TOSHIBA

Memoria: 512 Mega Bytes
Disco Duro: 10 Giga Bytes
Procesador: Pentium-III 1200Mhz
S.O.: Debian Sarge
Red: Cableada 100 Mbps

INTEL

Memoria: 512 Mega Bytes
Disco Duro: 10 Giga Bytes
Procesador: Pentium- IV 1.6 Ghz
S.O.: Debian Sarge
Red: Cableada 100 Mbps

DELL

Memoria: 128 Mega Bytes
Disco Duro: 10 Giga Bytes
Procesador: Pentium III 550 MHz
S.O.: Debian Sarge
Red: Cableada 100 Mbps

SONY

Memoria: 2 Giga Bytes
Disco Duro: 100 Giga Bytes
Procesador: Intel Core 2 Duo 1.8 GHz – 789 MHz
S.O.: Windows XP
Red: Inalámbrica 54 Mbps

HP

Memoria: 256 Mega Bytes
Disco Duro: 250 Giga Bytes
Procesador: Amd Turion X2 ultra dual core Mobile ZM80 – 2.1 GHz
S.O.: Windows XP
Red: Inalámbrica 54 Mbps

AMD

Memoria: 256 Mega Bytes
Disco Duro: 60 Giga Bytes
Procesador: Amd 64 Bits 3000+ 2.0 GHz
S.O.: Windows XP
Red: Cableada 100 Mbps

Figura 5-2 Equipos utilizados en la implementación del Cluster

Dichos equipos están interconectados entre sí y estos a su vez a la red de Internet mediante un enrutador ver Figura 5-3:

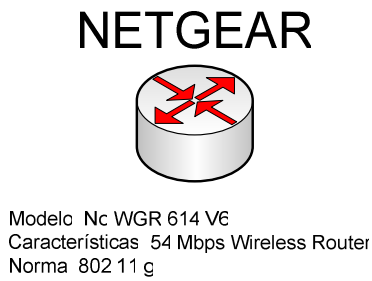

Figura 5-3 Enrutador

La conexión al enrutador es mixta, cableada e inalámbrica dependiendo del Terminal, la configuración de dicha red esta estructurada con una topología en estrella.

5.2.1 Software Empleado

Para el desarrollo de esta capa se seleccionó el sistema software openMosix instalado sobre un sistema operativo Linux Knoppix 3.6 actualizado a *kernel* 2.6.15 como herramienta para la creación del *Cluster* en el lado del servidor. Los clientes del *Cluster* utilizaron openMosix pero en este caso además de ser instalados sobre Linux también se instalaron en equipos con Sistema Operativo Windows XP, para este último caso se seleccionó la virtualización de Linux conocida como CosMos, la cual está basada en openMosix como se mencionó en el capitulo anterior.

OpenMosix brinda diversas herramientas gráficas por medio de las cuales se puede administrar el *Cluster* de manera comprensible y eficaz, por ejemplo, gracias a estas herramientas se puede:

- Gestionar la carga del *Cluster*.
- Determinar la disponibilidad de los nodos en el *Cluster*.
- Hacer balanceo de carga en tiempo real.
- Controlar la velocidad de procesamiento y memoria RAM de cada nodo y el *Cluster* en general.
- Distribuir los procesos de manera automática o manual con el Identificador de Procesos (PID - *Process Identifier*) y analizar la historia del comportamiento de la carga de procesos en un periodo de tiempo (ver Anexo A).

La instalación del *Cluster* openMosix puede hacerse de tres maneras distintas:

- Instalación desde los archivos fuente.
- Instalación desde los archivos RPM.
- Instalación desde un LiveCD de Knoppix (ver Anexo A).

En el Anexo A de instalación de OpenMosix se explica toda la configuración y puesta a punto del *Cluster*.

La instalación de la capa *Cluster* culmina una vez este funcionando el *Cluster*, de ahí en adelante se continúa con el montaje de la capa *Grid*.

5.3 Descripción de la Capa *Grid* experimental

La *Grid* que se implementa como solución prototipo, cuenta con dos nodos y un servidor central, el servidor central es el equipo Intel descrito en la Figura 5-2, los dos nodos están a cargo de los equipos Toshiba y Dell respectivamente conectados en red. En esta capa es donde se realiza la distribución información, tareas y aplicaciones con características de de tolerancia a fallas limitadas en la red a nivel geográfico, gracias a que *Grid* por su naturaleza permite este tipo de funciones o pueden ser adaptables a este entorno.

La implementación de la capa Grid se soporta sobre el juego de herramientas de Globus integrado por diversos componentes software esenciales disponibles para su correcto funcionamiento, ver la Figura 3-1 donde se muestran dichos componentes

5.3.1 Infraestructura de la capa Grid

La infraestructura utilizada para el prototipo de prueba se puede observar en la Figura 5-4. En ella se identifica el equipo servidor de la *Grid*, la base de datos y los diferentes clientes que consumen los servicios de la *Grid*.

Básicamente se puede observar del lado izquierdo de la Figura 5-4 el servidor de la *Grid* el cual contiene diversos módulos de servicios de los cuales solo se especifican los más relevantes, entre los módulos más destacados se encuentra el modulo de servicios RFT el cual brinda soporte al modulo de servicios GRAM y este a su vez al módulo Núcleo Java de servicios Web que contiene los servicios Web (Hakizumwami, 2005). Esto representa que gracias a RFT y GRAM, los servicios desplegados en la Grid adquieren características de tolerancia a fallos.

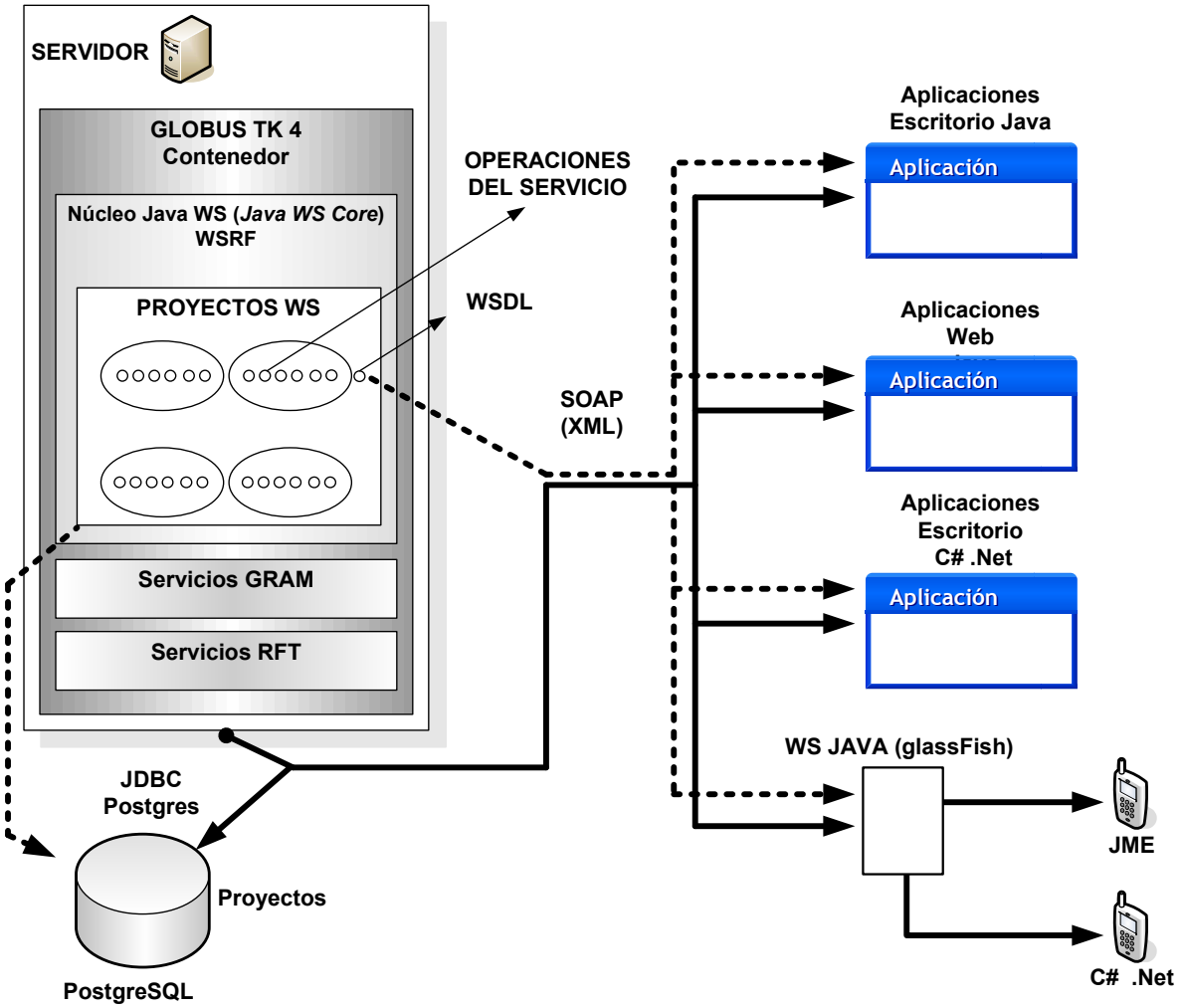

Figura 5-4 Infraestructura para el prototipo de prueba

El servidor de la *Grid* posee internamente el contenedor de Servicios Web de la *Grid* (WSRF) dentro de éste contenedor se pueden alojar diferentes Servicios Web que tienen acceso a la *Grid*, éstos servicios pueden tener una o más funciones que los clientes pueden consumir. Para que un cliente pueda consumir un servicio Web requiere

un descriptor de servicios, el cual es un archivo denominado WSDL que describe las operaciones, la manera de invocarlo y las principales interfaces del servicio Web. Para la comunicación entre el servidor que posee los servicios y el cliente usa un protocolo basado en XML llamado SOAP, estos mensajes tienen una cabecera y un contenido del mensaje, dichos mensajes si van desde el cliente al servidor llevan los parámetros necesarios para la invocación del servicio, y si es del servidor al cliente lleva las respuestas que el cliente necesite y que el servidor envíe.

Los pasos a seguir para crear un nuevo servicio Web en la *Grid* pueden resumirse de la siguiente manera:

- **Crear y desplegar los servicios Web**: Para crear el servicios Web se usó la herramienta *Introduce* (Ver Anexo D) como se menciono en el capítulo anterior. Al servicio creado se le agregan funciones u operaciones las cuales se implementan en Java, para ello se uso el entorno de desarrollo integrado Eclipse[22], el cual es compatible con *Introduce* ya que *Introduce* almacena los proyectos que contienen a los servicios Web como proyectos creados en Eclipse. Finalmente se debe desplegar el servicio sobre la *Grid*, para esto se uso nuevamente *Introduce*.

  La base de datos se despliega sobre un servidor de bases de datos denominado PostgreSQL, éste servidor se encuentra instalado en el equipo servidor de la *Grid*. El servidor de bases de datos no necesariamente debe estar instalado sobre el servidor de la *Grid* él podría instalarse en cualquier otro equipo perteneciente a la *Grid*. Se seleccionó Postgre como servidor de bases de datos ya que es de libre distribución y además se encuentra como uno de los requerimientos de instalación de Globus, por tanto es indispensable contar con él al momento de la instalación de Globus.

- **Consumir los servicios Web**: Para este prototipo experimental se utilizaron clientes desarrollados en dos diferentes lenguajes de programación Java y C#. Para ambos lenguajes de programación se desarrollaron aplicaciones Web de escritorio y aplicaciones móviles.

  Para las aplicaciones Web y de escritorio desarrolladas tanto en C# como en Java la invocación de los servicios se realizó directamente; por ejemplo en Java se utilizó un entorno de desarrollo integrado llamado NetBeans[23] el cual ofrece una herramienta que permite invocar un servicio Web con simplemente conocer la dirección URL (*Uniform Resource Locator*) del archivo descriptor del servicio (WSDL) y el entorno genera automáticamente el código de invocación del servicio. De manera similar se invoca un servicio en C# utilizando el entorno de desarrollo integrado llamado Visual Studio.

  Para el caso de las aplicaciones móviles tanto en C# como en Java se utilizó un servidor de servicios Web intermedio, el cual responde a las invocaciones de los clientes e invoca los servicios correspondientes de la *Grid* para entregar los resultados. El servidor utilizado en este caso es GlassFish[24] una herramienta gratuita y de mucha aceptación que posee grandes facilidades para el despliegue y prueba de servicios Web Java utilizando NetBeans. De esta manera un cliente móvil que desee invocar un servicio Web de la *Grid*, invocará un servicio Web desplegado en GlassFish, y será este servicio quien invoque el servicio de la *Grid*, esto hace que el procedimiento sea transparente para un cliente móvil quien obtiene la información que requiere desde el servidor de la *Grid* utilizando un intermediario que invoca directamente los servicios de la *Grid*. Figura 5-4.

  La creación de los clientes móviles en C# se realizó con Visual Studio 2005, el componente de aplicaciones móviles para Windows *Mobile* 5.0 y un *handheld* Xperia emulado. Para los clientes Java se usó JME (versión adaptada para dispositivos con bajas capacidades) y una herramienta llamada *Stubs Generator* del *Wireless Toolkit* (WTK). Para la Emulación de estos clientes se utilizó un *handheld* Sony Ericsson con sistema operativo SymbianP900 Figura 5-5 , la prueba real del cliente en java se realizó en un equipo Sony Ericsson P1i con

22 Eclipse: www.eclipse.org
23 NetBeans: www.netbeans.org
24 GlassFish: https://glassfish.dev.java.net

sistema operativo Symbian v9.1 UIQ 3.0, el cual accedió a través de una conexión inalámbrica WiFi a los servicios Web de la *Grid* , Figura 5-6.

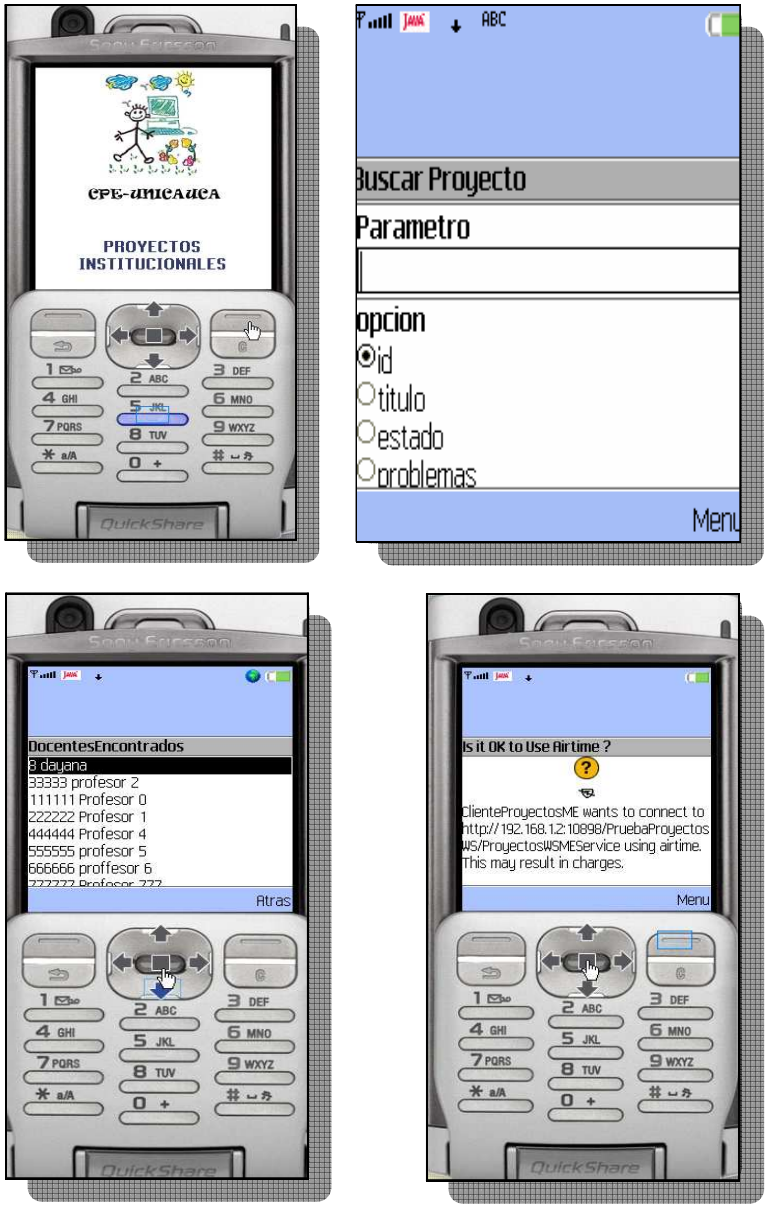

Figura 5-5 Emulador Sony Ericsson P900 invocando el servicio Web de Proyectos Institucionales

Figura 5-6.Vistas de la aplicación en un Sony Ericsson P1i

5.4 Pruebas ejecución

Se utilizaron como clientes aplicaciones escritas en lenguaje Java (Ver Anexo D). En la parte del servidor se utilizó el equipo Intel con el sistema operativo Linux Knoppix 3.6, el software openMosix para la capa *Cluster* y Globus Toolkit para la capa *Grid*.

La aplicación prototipo desarrollada se pensó para solucionar uno de los problemas más relevantes que vivencia el programa CPE-Unicauca, dicho problema se fundamenta en la imposibilidad de llevar un control general por cualquier usuario ya sea docente directivo o estudiante relativo a los proyectos institucionales que se manejan en cientos de instituciones que han sido beneficiadas con el programa, de tal forma que se acceda a la información de manera remota.

Esta aplicación denominada "Base de Datos de Proyectos Institucionales CPE-Unicauca" consiste en tener una Base de Datos de los Proyectos Institucionales desarrollados en cada Institución Educativa beneficiada por el programa. El acceso a esta base de datos se realiza consumiendo un servicio Web mediante clientes de escritorio o móviles.

El beneficio de montar esta base de datos en la *Grid* se justifica en que sobre la *Grid* se desarrollan aplicaciones por terceros, ella ofrece servicios como el RFT y GRAM que permiten características de tolerancia a fallas limitadas, permiten la creación de servicios Web confiables *stateful* gracias al WSRF, permiten localización y gestión de recursos, seguridad, infraestructura de información, comunicación e interoperabilidad entre diferentes.

Además la Arquitectura de Servicios Abiertos *Grid* (OGSA) presenta un conjunto de especificaciones y estándares que combina los beneficios de la informática *Grid* y los servicios Web. Así, los clientes pueden, por primera vez, compartir y acceder a los recursos informáticos que necesitan en Internet, contando con el soporte de una infraestructura muy robusta, con capacidad de autogestión y siempre disponible; pueden integrar aplicaciones y compartir datos y potencia de procesado, consiguiendo unos niveles de eficiencia muy altos, así como muy bajos costos.

Por último cabe resaltar que los servicios de OGSA, denominados servicios *Grid*, son un tipo especial de servicios Web que permiten a las organizaciones la provisión de acceso externo implícito a sus recursos de computación (y no solamente a sus datos, como sucede actualmente con los servidores Web) (Miguel, 2003).

5.4.1 Diagrama de Casos de Uso Proyectos Institucionales

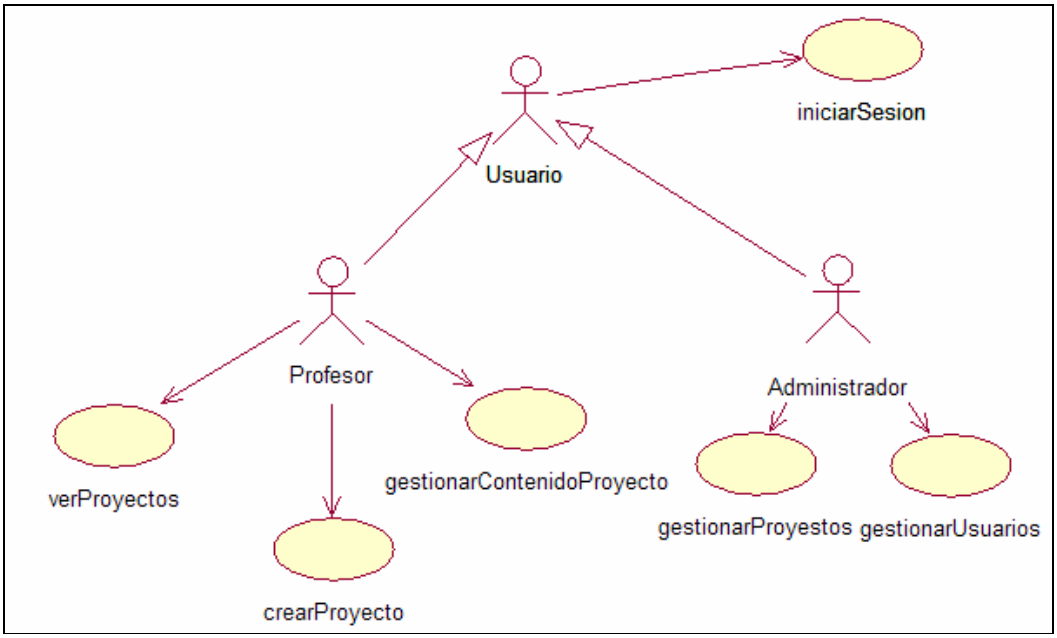

Figura 5-7 Diagrama de Casos de Uso Proyectos Institucionales

5.4.1.1 Descripción de los casos de uso

Actores:

- Usuario: es una clase madre de los actores Administrador y Profesor que corresponde a las tareas que realizan de igual manera los dos actores, como es el caso del inicio de sesión.
- Administrador: es el actor encargado de realizar las operaciones básicas de administración sobre los usuarios y proyectos almacenados en la base de datos de la *Grid.*
- Profesor: corresponde a los docentes que pueden ver proyectos institucionales de otros estamentos educativos y que a la vez pueden publicar sus propios proyectos.

Casos de Uso:

- Caso de Uso iniciarSesion:

Identificador	iniciarSesion
Actor	Usuario
Descripción	Permite distinguir los diferentes roles y usuarios válidos dentro del sistema, es básicamente una funcionalidad de seguridad.

Tabla 5-1 Caso de Uso iniciarSesion

- Caso de Uso gestionarUsuarios:

Identificador	gestionarUsuarios
Actor	Administrador
Descripción	Permite realizar las operaciones CRUD (*create, update, delete*) sobre los usuarios asignados al sistema y sus respectivos roles

Tabla 5-2 Caso de Uso gestionarUsuarios

- Caso de Uso gestionarProyectos:

Identificador	gestionarProyectos
Actor	Administrador
Descripción	Permite realizar las operaciones CRUD (*create, update, delete*) sobre los proyectos publicados por los profesores en la base de datos del sistema.

Tabla 5-3 Caso de Uso gestionarProyectos

- Caso de Uso gestionarContenidoProyecto:

Identificador	gestionarContenidoProyecto
Actor	Profesor
Descripción	Por medio de este caso de uso, el sistema les ofrece a los profesores la posibilidad editar y agregar contenido nuevo a los proyectos que han publicado en el sistema.

Tabla 5-4 Caso de Uso gestionarContenidoProyectos

- Caso de Uso crearProyecto:

Identificador	crearProyecto
Actor	Profesor
Descripción	Este caso de uso les permite a los profesores crear y publicar nuevos proyectos institucionales desarrollados al interior de su región educativa.

Tabla 4.5: Caso de Uso crearProyectos

- Caso de Uso verProyectos:

Identificador	verProyectos
Actor	Profesor
Descripción	Este caso de uso le permite a los profesores revisar los diferentes proyectos institucionales de diferentes regiones publicados en el sistema.

Tabla 5-5 Caso de Uso verProyectos

5.4.2 Diagrama Entidad Relación

El sistema contiene básicamente 3 tablas descritas a continuación:

- Proyecto: representa el proyecto institucional que se va a publicar en el sistema.
- Docente: contiene la información de los docentes involucrados en un cierto proyecto institucional, el campo que relaciona a la Tabla proyecto con la Tabla docente es el campo Id. el cual permite saber en que proyecto está involucrado un docente.
- Estudiante: contiene la información de los estudiantes que participan en un proyecto institucional.

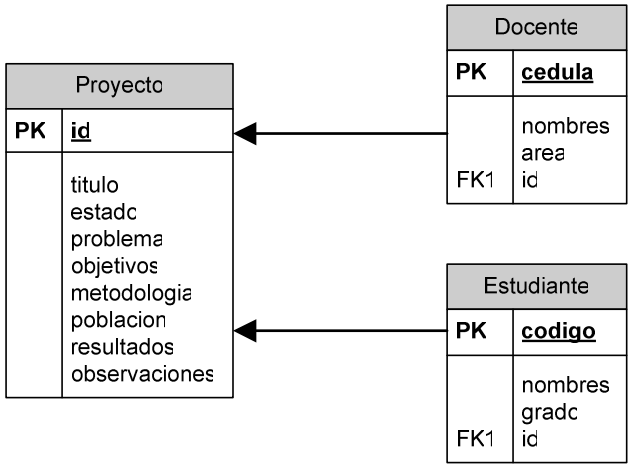

Figura 5-8 Diagrama entidad relación

5.4.3 Pruebas de acceso a la Base de Datos

La ejecución de las pruebas consiste en invocar un determinado número de veces el servicio Web de Proyectos Institucionales montados sobre la *Grid* desde la aplicación de escritorio, inicialmente en un entorno que utiliza la *Grid* sin la presencia del *Cluster* computacional. Posteriormente se realiza la misma prueba pero esta vez se agrega el equipo al *Cluster* desarrollado con openMosix y se ejecuta la misma prueba anterior, finalmente se comparan los resultados para observar la diferencia en los tiempos ejecución de los servicios Web con y sin *Cluster* computacional.

La prueba presenta ciertas limitaciones en la ejecución de servicios Web desarrollados en Java. Las aplicaciones de la máquina virtual de Java no se pueden distribuir. Una solución es desarrollar la aplicación en el lenguaje de programación C. Esto debe funcionar dado que las aplicaciones en C pueden utilizar funcionalidades nativas del sistema operativo, lo que no puede hacer Java al utilizar una máquina virtual.

Sin embargo, cuando un nodo se encuentra ocupado con múltiples procesos, el *Cluster* detecta los nodos que se encuentren ociosos y distribuye de manera automática o manual las tareas que sean compatibles con la distribución, así por ejemplo si se consume un servicio Web desarrollado en Java (el cual no se puede distribuir) que esté ocupando muchos recursos de procesamiento en el equipo servidor y que al mismo instante sobre el mismo equipo se estén ejecutando otras aplicaciones tal como grandes ciclos desarrollados en C, el *Cluster* toma los procesos que pueden distribuirse y los envía a los equipos ociosos para liberar así recursos en el equipo servidor.

Por tanto aunque los procesos de aplicaciones desarrolladas en Java no se pueden distribuir, el servidor *Cluster* realiza su mejor esfuerzo por balancear los nodos del *Cluster* distribuyendo los procesos que se pueden distribuir y así liberar recursos para los proceso de aplicaciones Java o procesos que no se pueden distribuir.

5.4.3.1 Pruebas de consulta a la Base de Datos

En esta prueba se realizó una invocación directa del servicio Web para la consulta a la base de datos de los proyectos institucionales utilizando inicialmente la capa *Grid* sin utilizar la capa *Cluster*, es decir que no se hace distribución de procesamiento local. Posteriormente se realiza la misma invocación pero esta vez con las dos capas, *Grid* y *Cluster*, ver Tabla 5-6.

PRUEBA DE CONSULTA A BASE DE DATOS								
DETALLE DE PRUEBA	SIN CLUSTER				CON CLUSTER			
	Items	Consultas	Tiempo Total Todas Consultas (ms)	Tiempo Promedio Una Consulta (ms)	Items	Consultas	Tiempo Total Todas Consultas (ms)	Tiempo Promedio Una Consulta (ms)
Consulta Base de Datos de Proyectos Institucionales	7,0	200,0	124125,0	620,6	7,0	200,0	103379,0	516,6

Tabla 5-6 Prueba de consulta a la Base de Datos de Proyectos Institucionales con y sin *Cluster*

5.4.3.2 Pruebas de escritura a la Base de Datos

En esta prueba se realizó una invocación directa del servicio Web para la escritura a la base de datos de los proyectos institucionales utilizando inicialmente la capa *Grid* sin utilizar la capa *Cluster*, es decir que no se hace distribución de procesamiento local. Posteriormente se realiza la misma invocación pero esta vez con las dos capas, *Grid* y *Cluster*, ver Tabla 5-7.

PRUEBA DE ESCRITURA A BASE DE DATOS								
DETALLE DE PRUEBA	SIN CLUSTER				CON CLUSTER			
	Items	Escritura	Tiempo Total Todas Escrituras (ms)	Tiempo Promedio Una Escritura (ms)	Items	Escritura	Tiempo Total Todas Escrituras (ms)	Tiempo Promedio Una Escritura (ms)
Escritura Base de Datos de Proyectos Institucionales	7,0	200,0	117455,0	587,2	7,0	200,0	115720,0	578,6

Tabla 5-7 Prueba de escritura a la Base de Datos de Proyectos Institucionales con y sin *Cluster*

En las dos pruebas tanto de consulta como de escritura sobre la base datos descritas anteriormente se puede observar que los tiempos de ejecución disminuyen cuando se tiene la presencia del *Cluster*, esto se debe a que sobre los equipos usados para la prueba se realiza una distribución de procesos para el balanceo de carga. Cabe resaltar que la diferencia de tiempos no es muy significativa debido a que los equipos en el momento de realizar las pruebas estaban ejecutando procesos normales de su sistema operativo y además la invocación directa del servicio Web para la consulta a la base de datos por su naturaleza no consume gran cantidad de recursos computacionales.

5.5 Propuesta de Arquitectura DTN

5.5.1 Adaptaciones en Globus

Para adaptar el funcionamiento de Globus a una DTN entre los nodos Grid se debe utilizar de manera sistemática la transferencia confiable de archivos y se debe realizar modificaciones al registro y control de tales transferencias.

Sobre la capa de transferencia confiable de archivos se debe construir una capa de administración DTN y se debe implementar una funcionalidad gateway que permita a una capa de aplicación la invocación de operaciones y reportar el éxito o falla de tales invocaciones.

Estas adaptaciones se describen a continuación.

5.5.2 Almacenamiento y reenvío de Unidades de Datos

En DTN, de acuerdo al RFC4838, las Unidades de Datos de Aplicación (ADU) que contienen información de aplicación e información de encaminamiento deben ser transformadas por la *capa bundle* en unidades de datos llamadas *"bundles".*

Las ADU deben incluir toda la información necesaria para que sea considerada una unidad de trabajo independiente y de esta forma se minimice las interacciones y el intercambio de información entre los nodos. Esto es importante dado que la red en cuestión está caracterizada por grandes y altamente variables retardos.

Es posible adaptar las aplicaciones para que entreguen las ADU a Globus en forma de archivos, los cuales pueden ser registrados y administrados en la base de datos SQL y puede ser gestionada su entrega a un nodo Globus mediante RFT.

De esta forma el archivo corresponde al *bundle*, el cual debe contener además de los identificadores de los nodos origen y destino la información útil de entrada o de respuesta para el proceso receptor en el destino. Son necesarias adaptaciones para permitir el almacenamiento de información de gestión del *bundle* en la base de SQL de Globus para registrar y controlar la transferencia confiable tramo a tramo con selección de enrutamiento a través de los nodos. Un *bundle* debe contener una marca de tiempo de origen, indicador de vida útil, una asignación de clase de servicio y la longitud.

Los *bundles* pueden ser almacenados de manera persistente en la base de datos de Globus (empotrado o mediante referencia al sistema de archivos) o en el sistema de archivos del servidor hasta que se den las condiciones de conectividad para su relevo. De esta forma los *bundles* están en condiciones de sobrevivir a reinicios del sistema y el almacenamiento estará disponible y distribuido homogéneamente a través de la red.

La estructura de los *bunldes* puede estar basada en una adaptación de Extensiones de Correo de Internet Multipropósito MIME (*Multipurpose Internet Mail Extensions*).

Los *bundles* pueden contener una o más unidades de datos llamados "*blocks*". Aunque una implementación inicial con Globus no se tenga la funcionalidad de ensamblar y desensamblar *blocks* en *bundles*, los *bundles* pueden ser divididos en fragmentos durante su transmisión con RFT.

Globus puede utilizar *bundles* para transferir código que finalmente se ejecute en el nodo destino para implementar migración de procesos.

5.5.3 Nodos, Identificadores y Custodia

Un *bundle* contiene identificadores para la fuente y uno o más destinos (EID, *EndPoind Identifiers*), en este caso los identificadores corresponden a nodos de la Grid. Los EID se expresan utilizando la sintaxis de las URI. En DTN es posible especificar un grupo de nodos mínimo a los cuales se debe entregar un *bundle* (MRG, *Minimun Reception Group*"). Globus ofrece la posibilidad de enviar tareas a múltiples nodos de destino que en este caso tendría una correspondencia directa con el MRG. La base de datos SQL debe implementar un registro de despacho para múltiples destinos.

En una DTN un concepto clave es el nodo custodia, el cual es un nodo que tiene en su poder el *bundle*. La custodia representa la responsabilidad de la entrega confiable de *bundles* entre los nodos. Dado que una DTN generalmente se hace transferencias tramo a tramo, únicamente cuando la transferencia al siguiente salto ha sido exitosa se puede transferir la custodia del *bundle*. Los mensajes de requerimiento de custodia al siguiente salto, así como los mensajes de aceptación de custodia deben estar incluidos en *bundles*. En este caso se asume que existe un camino de regreso.

La custodia hace uso de la funcionalidad de retransmisión de RFT, pero se requiere sin embargo un registro de estado de custodia que debe ser almacenado de manera persistente en la base de datos SQL de Globus, así como los estados de progreso y reportes de entrega exitosa.

Cada *bundle* convertido en archivo debe contener como mínimo la siguiente información, la cual también debe ser mapeada en la base de datos SQL de Globus:

- Marca de tiempo de creación
- Vida útil
- Banderas de clase de servicio
- Identificador EID de la Fuente
- Identificadores EID de los destinos
- Identificador EID del nodo al cual deben enviarse reportes de progreso y entrega.
- Identificador del EID custodia. Referencia al nodo al cual se debe notificar la aceptación de una custodia.

Una posible mejora para la operación automática de Globus puede ser la especificación de prioridad en la migración de procesos y transferencia de archivos, solicitados por a priori por la aplicación. En este caso sería posible contar con acuses del progreso y del estado de la migración de procesos y de la transferencia de archivos.

5.5.4 Encaminamiento

Aunque en una red DTN se podría implementar encaminamiento óptimo de acuerdo a la capacidad de cada nodo y el retardo estadístico entre nodos, este tipo de implementación es muy compleja y es objeto de investigación.

Los protocolos alrededor de Internet TCP-IP y los protocolos de enrutamiento no están diseñados para operar en condiciones de grandes retardos, por tanto una implementación estricta de DTN requeriría una adaptación de toda la capa de transporte y red.

Globus requiere una adaptación de sus módulos para que pueda utilizar capas de transporte diferente a TCP-IP.

6 Capítulo 6. Conclusiones y Recomendaciones

6.1 Conclusiones

La Grid creada con Globus Toolkit ofrece soporte de distribución dinámica y virtual de acceso remoto a aplicaciones, datos y recursos computacionales. Para distribuir las aplicaciones, incluyendo las de ofimática, se requiere que estén diseñadas para entornos Grid. Para solucionar el problema de carga de aplicaciones de ofimática no distribuibles se usa únicamente la capa cluster, donde éstas pueden alojarse en el servidor cluster y los terminales pueden ser terminales tontas con acceso mediante emulación. Otra solución es usar procesadores de texto, hojas de cálculo, y programas de presentación en línea explotando los beneficios de distribución de procesamiento de la Grid.

Los actuales problemas de manejo de información entorno a las Instituciones Educativas pueden abordarse eficientemente con tecnologías soportadas en contextos Grid, tal es el caso de los servicios Web implementados en este trabajo. Estos servicios sobre globus poseen la particularidad de seguridad, distribución, capacidad de autogestión, disponibilidad y conservación del estado, características muy importantes cuando se requiere utilizar servicios Web que se van a invocar por muchos usuarios y los cuales pueden contener información sensible consiguiendo altos niveles de eficiencia, así como muy bajos costos.

La arquitectura propuesta permitió construir un entorno distribuido heterogéneo de doble capa con diferentes niveles de acceso, con una instancia *Grid* que permite el acceso a diferentes recursos a través de Internet de manera segura y una instancia *Cluster*, el cual busca aprovechar al máximo el poder computacional de los equipos incluidos dentro de una red de área local (normalmente un solo centro informático) y distribuir procesos entre estos equipos.

Después de trabajar con el sistema Globus se identificó las adecuaciones que se deben desarrollar en este sistema para su implementación en un entorno que tenga en cuenta tipos de conectividad con grandes retardos e interrupciones y así ofrecer una solución para una Grid – DTN.

6.2 Trabajo futuro

En relación con la educación y las TIC que es el campo de acción del programa computadores para educar, queda planteada una arquitectura para el desarrollo de trabajos que se enfoquen en la creación de servicios soportados en entornos *Grid* y *Cluster* tales como:

- Registro ubicuo de personas sobre un centro educativo particular.
- Creación ubicua personalizada de currículos y sincronización de información.
- Laboratorios de instrumentación real y virtual en línea.
- Presentación de trabajos soportados en los recursos conectados a la *Grid* y *Cluster*.
- Clases en línea interactivas con expertos para incrementar el desempeño de las mismas.
- Evaluaciones en línea de diversos tipos por ejemplo selección múltiple con parámetros aleatorios.
- Acceso al análisis y evaluación de datos de diversos tipos.
- Acceso asíncrono, análisis de minería de datos e investigación.

De igual manera se plantea el uso de estas tecnologías para aplicarlas al entrenamiento y educación virtual interactiva, *e-learning* o *t-learning* y así contribuir a:

- Soportar el aprendizaje ubicuo, colaborativo, experimental y personalizado en un entorno que permita la interoperabilidad de herramientas, contenidos y cursos de aprendizaje.

- Apoyar las capacidades de *e-learning* que permitan la interoperabilidad, accesibilidad y reusabilidad de contenido *e-learning* basado en Web.
- Equilibrar el rápido y continuo cambio del plan de estudios y el incremento acelerado de la comunidad educativa.

Implementas las adecuaciones correspondientes de Globus para su operación en entorno DTN.

7 Referencias

- Abramson D., Giddy J., and Kotler L., *High Performance Parametric Modeling with Nimrod/G: KillerApplication for the Global Grid?*, Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS 2000), May 1-5, 2000, Cancun, Mexico, IEEE CS Press, USA, 2000.
- ADL, Advanced Distributed Learning, 2007. [En línea]. Disponible: http://www.adlnet.gov/ [Consulta: marzo, 2009].
- AICC, Aviation Industry CBT Comité, 2008. [En línea]. Disponible: http://www.aicc.org [Consulta: marzo, 2009].
- ALICE, America Latina Interconectada Con Europa, 2009. [En línea]. Disponible: http://alice.dante.net/ [Consulta: marzo, 2009].
- Almond J., Snelling D., UNICORE: Uniform Access to Supercomputing as an Element of Electronic Commerce. *Future Generation Computer Systems.* 1999.
- Alón D.,"Cooperative Linux," *Proceedings of the Linux Symposium Volume One*, Ottawa, Ontario, Canada. 2004.
- ANSI, American National Standards Institute 2009 [En línea]. Disponible: http://www.ansi.org/ [Consulta: marzo, 2009].
- Apon a., Baker M., "Network Technologies;" University of Arkansas, USA, and, University of Portsmouth, UK. Sage Publications, Inc. Volume 15, 2001.
- Argonne National Laboratory, Mathematics and Computer Science Division, The Message Passing Interface (MPI) Standard 2009 [En línea]. Disponible: http://www.mcs.anl.gov/research/projects/mpi/ [Consulta: marzo, 2009].
- Asadzadeh P., Buyya R., Kei C., et al, "Global Grids and Software Toolkits: A Study of Four Grid *Middleware* Technologies", Grid Computing and Distributed Systems (GRIDS) Laboratory. University of Melbourne, Australia, 2004.
- BADG Project, *Belle Analysis Data Grid*, 2005 [En Línea]. Disponible: http://epp.ph.unimelb.edu.au/epp/grid/badg/ [Consulta: marzo, 2009]
- Bala P., Pytlinski J., Nazaruk M., "*BioGRID-An European grid for molecular biology*, Proceedings," 11th IEEE International Symposium on High Performance Distributed Computing (HPDC-11), Edinburgh, UK, 2002.
- Barak A. and Shiloh A, MOSIX *Cluster* and Multi-*Cluster* Management 1999 [En línea]. Disponible: http://www.mosix.org/ [Consulta: marzo, 2009].
- Beck R., Hinkel K., Peoples C., Parr G., et al, "GPSDTN: Predictive Velocity-Enabled Delay-Tolerant Networks for Arctic Research and Sustainability," accepted for presentation at the *IEEE First International Workshop on Tracking Computing Technologies*, Julio, 2007.
- Beekwilder N., Grimshaw A., "*Parallelization of an Axially Symmetric Steady Flow Program*," CS Technical Report CS-98-10, University of Virginia, USA, 1998.
- Beeson B., Melnikoff S., Venugopal S., and D. Barnes, *A Portal for Grid-enabled Physics*, Australian Virtual Observatory 2009 [En Línea]. Disponible: http://www.aus-vo.org/ [Consulta: marzo, 2009]
- Bellwood, T et al. UDDI Version 2.03 Data Structure Reference. UDDI Committee Specification, UDDI Spec TC. 2002.
- Berman F., Hey A., Fox G., *Grid Computing, Making the Global Infraestructure a Reality*, The Atrium, Southern Gate, Chichester, West Sussex, England: Wiley series in communications networking & distributed systems, 2003.
- Berman F. and Wolski R., *"The AppLeS Project: A Status Report,"* 8th NEC Research Symposium, Berlin, Germany, 1997.
- Boch, The Bochs Project, The cross plataform IA-32 Emulator, Bochs, 2001. [En línea]. Disponible: http://bochs.sourceforge.net/ [Consulta: marzo, 2009].
- Branch W., Mesa A. "Implementación de un cluster homogéneo para la resolución de problemas de alta complejidad computacional" Universidad Nacional sede Medellín, Revista Avances en Sistemas e Informática, ISSN 1657-7663, 2008]

- Bravo L, Valencia V., "Distribución de contenidos, sincronización y seguimiento de actividades estudiantiles en un entorno de aprendizaje desconectado para el proyecto e-lane," Universidad del Cauca, 2006.
- Bray. T, Paoli. J, McQueen. S, Maler. E, Yergeau. F. Extensible Markup Language (XML) 1.0. W3C Recommendaton. 2008.
- Brewer E., Demmer M., Du B., et al, "The Case for Technology in Developing Regions," *IEEE Innovative Technology for Computer Professionals,* University of California at Berkeley, Intel Research Berkeley, 2005. [En línea]. pp. 26-38. Disponible: http://tier.cs.berkeley.edu/docs/CFT-ieee.pdf [Consulta: Agosto, 2008].
- Buyya R., Date D., Mizuno-Matsumoto Y., Venugopal S., and Abramson D., "*Composition of Distributed Brain Activity Analysis and its On-Demand Deployment on Global Grids*," New Frontiers in High-Performance Computing: Proceedings of the 10th International Conference on High Performance Computing (HiPC 2003) Workshops, Hyderabad, India, 2003.
- Buyya R., *Grid* Computing Info Centre (*Grid* Infoware), 2009. [En línea]. Disponible: http://www.gridcomputing.com/ [Consulta: marzo, 2009].
- caGrid, Introduce 2009 [En Línea]. Disponible: http://www.cagrid.org/display/introduce/Home [Consulta: marzo, 2009]
- Calicut, Kerala, "Implementing a Linux *Cluster*," Deepak Lukose National Institute of Technology. 2006.
- Capuano N., et al, "How To Use Grid Technology for Building the Next Generation Learning Environments," *Towards the Learning Grid,* Vol. 127, U.S.A.:, IOSPress, pp. 182-192. 2005.
- Catalán M., "El manual para el *Clustering* con OpenMosix" Versión 1.0. Mikel a.ka.mc2 & Buytaert, 2004.
- Catlett C., et al, "NSF Extensible Terascale Facility," *TeraGrid Annual Report and Program Plan*, 2007.
- CERES, Centros Regionales de Educación Superior, Ministerio de Educación Nacional, República de Colombia, 2009. [En línea]. Disponible: http://www.mineducacion.gov.co/1621/article-85678.html [Consulta: marzo, 2009].
- Christensen. E, Curbera. F, Meredith. G, Weerawarana. S. Web Services Description Language (WSDL) 1.1. W3C Note 2001.
- CLARA Cooperación Latino Americana de Redes Avanzadas, 2009. [En línea]. Disponible: http://www.redclara.net/ [Consulta: marzo, 2009].
- coLinux, SoureceForge.net 2009. [En línea]. Disponible: http://www.colinux.org/ [Consulta: marzo, 2009].
- CoMosix, SourceForge.net, CoMosix Index Page 2007 [En línea]. Disponible: http://comosix.sourceforge.net/ [Consulta: marzo, 2009].
- Compartel, Ministerio de Comunicaciones, República de Colombia, 2009. [En línea]. Disponible: http://www.compartel.gov.co/ [Consulta: marzo, 2009].
- Condor Project University of Wisconsin, Condor High Throughput Computing, 2009 [En línea]. Disponible: http://www.cs.wisc.edu/condor/ [Consulta: marzo, 2009].
- Cor G., "10 Oportunidades arquitecturales para Workflow," Paradigma software Microsoft Regional Director Uruguay, Paraguay & Bolivia. 2006.
- Cortizo J., Quintana D., Rodriguez P."Comparativa de *Cluster*s SSI (A Comparative Study of SSI *Cluster*s)" Universidad Europea de Madrid, Villaviciosa de Odón, 2007.
- CPE, Computadores para Educar, 2004. [En línea]. Disponible: http://www.computadoresparaeducar.gov.co/ [Consulta: marzo, 2009].
- Curbera F., Goland Y., et al, "Business Process Execution Language for Web Services Version 1.1. 5 May 2003," BEA Systems, International Business Machines Corporation, Microsoft Corporation, SAP AG, Siebel Systems. 2003.
- Di Stefano M., *Distributed Data Management For Grid Computing.* Hoboken, New Jersey: John Wiley & Sons, Inc. 2005.
- DTNRG, Delay Tolerant Networking Research Group, Enero 2008. [En línea]. Disponible: http://www.dtnrg.org [Consulta: marzo, 2009].
- EELA, E-Infraestructure shared between Europe and Latin America, 2008. [En línea]. Disponible: http://www.eu-eela.org/eela_about.php [Consulta: Agosto, 2008].
- EELA-2, E-science grid facility for Europe and Latin America, 2009. [En línea]. Disponible: http://www.eu-eela.eu [Consulta: marzo, 2009].
- EGEE Enabling Grids for E-sciencE, 2008. [En línea]. Disponible: http://www.eu-egee.org/ [Consulta: marzo, 2009].

- EHAS, Enlace Hispano Americano de Salud, Ingeniería sin Fronteras 2009. [En línea]. Disponible: http://www.ehas.org [Consulta: marzo, 2009].
- E-LANE, European and Latin American New Education (E-LANE), 2008. [En línea]. Disponible: http://e-lane.org/ [Consulta: marzo, 2009].
- ELeGI, European Learning Grid Infrastructure, 2008. [En línea]. Disponible: http://www.ELeGI.org/ [Consulta: marzo, 2009].
- Enriquez S., Isaías H. "*Clustering* y Grid Computing" Sistemas Distribuidos, Escuela de Informática, Universidad de Trujillo, 2007.
- E-Science, 2007. Science & Technology Facilities Council. [En línea]. Disponible: http://www.e-science.clrc.ac.uk and http://www.escience-grid.org.uk/ [Consulta: marzo, 2008].
- EuroGrid, The EuroGrid Project 2009 [En Línea]. Disponible: http://www.eurogrid.org/ [Consulta: marzo, 2009].
- Fabrice Bellard, Quemu open source processor emulator 2005 [En línea]. Disponible: http://www.nongnu.org/qemu [Consulta: marzo, 2009].
- Fall K., "A Delay-tolerant network architecture for challenged internets", Intel Research, Berkeley, inédito, 2003.
- Feller M., Foster I., Martin S."GT4 GRAM: A Functionality and Performance Study," TeraGrid Conference, Madison, Wi, 2007.
- Fernández O., "Virtualización con Microsoft Virtual PC,"Primera Edición. 2007.
- Ferreira L, et al, *Grid Computing in Research and Education*, First Ed. U.S.A.: IBM Corp, Red Books, 2005.
- Foster I, Kesselman L., "Globus: A metacomputing infrastructure toolkit," *International Journal of Supercomputer Applications* 1997.
- Foster I., "What is the Grid? A Three Point Checklist," Argonne National Laboratory & University of Chicago, 2002.
- Frey J., et al, "*Condor-G: A Computation Management Agent forMulti-Institutional Grids*," Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC10) San Francisco, California, 2001.
- Gaeta M., Ritrovato P., and Salermo S., "Making e-Learning a Service Oriented Utility: The European Learning Grid Infrastructure Project," *Towards the Learning Grid,* Vol. 127, U.S.A.:, IOSPress, pp. 41-63. 2005.
- Globus, The Globus Alliance, 2008. [En línea]. Disponible: http://www.globus.org/ [Consulta: marzo, 2009].
- Gridbus, The Grid Computing and Distributed Systems (GRIDS) Laboratory, University of Melbourne, *"The Gridbus Project,"* Department of Computer Science and Software Engineering The University of Melbourne, Australia 2009 [En línea]. Disponible: http://www.gridbus.org/ [Consulta: marzo, 2009].
- *Grid*-Colombia, 2006. [En línea]. Disponible: http://urania.udea.edu.co/grid-colombia/index.php [Consulta: marzo, 2009].
- Grimshaw A., *A Worldwide Virtual Computer for an Advancing Legion of Applications*, NPACI. 2006 [En Línea]. Disponible: http://www.npaci.edu/enVision/v15.2/legion.html [Consulta: agosto, 2008].
- Grimshaw A., Wolf W., *"The Legion vision of a worldwide virtual computer,"*. Communications of the ACM 1997.
- GriPhyN, National Science Foundation *Grid Physics Network* (GriPhyN) 2006 [En Línea]. Disponible: http://www.griphyn.org/ [Consulta: marzo, 2009].
- GT4FactSheet, University of Chicago Globus Project and Globus Toolkit, Status and Plans for the Globus Toolkit 4.0 (GT4) 2005 [En Línea]. Disponible: http://www-unix.globus.org/toolkit/docs/4.0/GT4Facts/ [Consulta: marzo, 2009]
- GT4IDE, The Globus Toolkit, SourceForge.net, Mambo open source engine, Globus service build tools GT4IDE 2005 [En Línea]. Disponible: http://gsbt.sourceforge.net/ [Consulta: marzo, 2009]
- Gudin. M, Hadley.M, Mendelsohn. N, Moreau. J, Nielsen. H, Karmarkar. A, Lafon. Y. SOAP Version 1.2 Part 1: Messaging Framework. W3C Recommendation. 2007.
- Hakizumwami B., "Constructing Web Services with the Globus Toolkit Version 4," O'Reilly Media, Inc. 2005. Disponible: http://www.onjava.com/pub/a/onjava/2005/10/19/constructing-web-services-with-globus-toolkit.htm

- Hey T. and Trefethen A., "The UK e-Science Core Programme and the Grid," *Future Generation Compute Systems*, Volume 18, Issue 8, 2002.
- Hoschek W., et al, "*Data Management in an Internacional Data Grid Project*," Proceedings of the first IEEE/ACM International Workshop on Grid Computing, (Springer Verlag Press, Germany), India, 2000.
- Hughes B., Venugopal S., Buyya R., "*Grid-based Indexing of a Newswire Corpus,"* Technical Report, GRIDSTR- 2004-4, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, 2004.
- IBM, About *Cluster* Systems, United State, 2009 [En línea]. Disponible: http://www-03.ibm.com/systems/*Cluster*s/about/index.html [Consulta: marzo, 2009].
- IBM-Corp, New to *Grid* Computing, 1994. [En línea]. Disponible: http://www-128.ibm.com/developerworks/grid/newto/ [Consulta: marzo, 2009].
- IMS Global Innovations Adoption Learning IMS Global Learning Consortium, 2008. [En línea]. Disponible: http://www.imsglobal.org/learningdesign [Consulta: marzo, 2009].
- J2ME, Sun Microsystems, Inc. "Java 2 Platform, Micro Edition (J2ME) Web Services." A technical White Paper, 4150 Network Circle, Santa Clara, CA 95054 USA. 2004.
- Jacob B., et al, *Enabling Applications for Grid Computing with Globus*. First Ed. U.S.A.: IBM Corp, Red Books, 2003.
- Johnston W., Gannon D., and Nitzberg B., "*Grids as production computing environments: The engineering aspects of NASA's Information Power Grid*," In Proc. Eighth IEEE International Symposium on High Performance Distributed Computing, Redondo Beach, CA, USA, 1999.
- Jones E., Li L., Schmidtke, and Ward P. "Practical Routing in Delay-Tolerant Networks," *IEEE Trans. On Mobile Computing,* Vol. 6, No. 8, pp. 943-959. 2007.
- Joseph D. Sloan, "High Performance Linux *Cluster*s with OSCAR, Rocks, OpenMosix, and MPI," O´Relley editors, 2004.
- Juang P., Oki H., Wang Y., Maronosi M., Peh L., and Rubenstein D, "Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet," *Proc. ASPLOS*, 2002.
- Kesselman C., Foster I., Nick J., and Tuecke S., "The Physiology of the Grid: Open Grid Services Architecture for Distributed Systems Integration," *Proc. 4th Global Grid Forum (GGF4) Workshop*. 2002.
- Kilian Cavalotti, SourceForge.net, . OpenSSI wevView a cluster monitoring system 2004 [En línea]. Disponible: http://openssi-webview.sourceforge.net/ [Consulta: marzo, 2009].
- Lamadrid L., Diseño y desarrollo de un portal grid para la docencia universitaria, Universidad Autónma de Baja California, 2008
- Latter I., "Planet Series Project , Midnightcode," Project Chaos / CosMos (*The Great Systems*) 2004 [En línea]. Disponible: http://midnightcode.org/projects/chaos/ [Consulta: marzo, 2009].
- Lee C., Talia D., "Grid programming models: current tools, issues and directions" *The Aerospace Corporation, California, United States,*2*Universit´a della Calabria, Rende, Italy*. *Grid Computing, Making the Global Infraestructure a Reality*, The Atrium, Southern Gate, Chichester, West Sussex, England: Wiley series in communications networking & distributed systems, 2003.
- Lefévre L., and Gelas J., "Towards Interplanetary Grids," INRIA/LIP École Normale Supérieure de Lyon, 2006.
- Legion, Worldwide Virtual Computer 1993 [En línea]. Disponible: http://legion.virginia.edu/ [Consulta: marzo, 2009].
- Li Chun-lian, Sun Yu, and Zhuang Shu-fan, "Study on the e-learning grid construction for higher special education," *IFIP International Conference on Network and Parallel Computing - Workshop*, 2007.
- Lin L., Budak I., "Discovery of Semantic Relations between Web Services," icws,pp.357-364, ICWS'06. 2006.
- LINK-ALL, Local Communities Insertion Network para America Latina, 2009. [En línea]. Disponible: http://www.link-all.org [Consulta: marzo, 2009].
- López M., Nores, et al, "Solutions for personalized t-learning," *In 3rd European Conferen-ce on Interactive Television: User Centred ITV Systems, Programmes and Applications (EuroITV-05),* 2005.
- Madrid, Spain. 2004
- MAGE-DT, The Distributed Systems Group - University of Marburg, Germany, MAGE-GDT: Grid Development Tools (not only) for Eclipse http://mage.uni-marburg.de/trac/gdt/wiki - MAGE-

GDT:GridDevelopmentToolsnotonlyforEclipse#MAGE-GDT:GridDevelopmentToolsnotonlyforEclipse, 2009 [En Línea]. Disponible: http://mage.uni-marburg.de/trac/gdt/wiki [Consulta: marzo, 2009]

- Menday R., Wieder P., *GRIP: The Evolution of UNICORE towards a Service-Oriented Grid*, Cracow Grid Workshop, October 2003.
- Miguel L., et al, "Computación Grid e Ingeniería del Software Basada en Componentes en CSCL," ETSI de Telecomunicación, Universidad de Valladolid Camino Viejo del Cementerio s/n, 47011 Valladolid, Spain, 2003.
- Mondardini R., 2009. Grid Café. [En línea]. Disponible: http://gridcafe.web.cern.ch/gridcafe [Consulta: marzo, 2009].
- Muñoz E., Rozo L. "Plataforma en el Web para Computación Paralela," Universidad del Cauca, 2006.
- NAREGI, National Research Grid Initiative, NAREGI Project, Servian, 2003 [En Línea]. Disponible: http://www.naregi.org/ [Consulta: marzo, 2009].
- Natrajan A., et al, "*Studying Protein Folding on the Grid: Experiences using CHARMM on NPACI Resources under Legion*," Proceedings of the 10th International Symposium on High Performance Distributed Computing (HPDC-10), 2001.
- NC-BioGrid 2006 [En Línea]. Disponible: http://www.ncbiogrid.org/ [Consulta: marzo, 2009]
- Neumann F., Geys R., "SCORM and the Learning Grid," *Towards the Learning Grid,* Vol. 127, U.S.A.: IOSPress, pp. 88-98. 2005.
- NFS award, Rocks Core Development 2006 [En línea]. Disponible: http://www.rocks*Clusters*.org/wordpress [Consulta: marzo, 2009].
- Open *Grid* Forum, 2008. [En línea]. Disponible: http://www.gridforum.org/ [Consulta: marzo, 2008].
- openMosix Source forge, Inc. 1999 [En línea]. Disponible: http://sourceforge.net/projects/openmosix/ [Consulta: marzo, 2009].
- OpenSSI Source forge, Inc., (Single System Image) *Cluster*s for Linux 2006 [En línea]. Disponible: http://openssi.org [Consulta: marzo, 2009].
- Oscar Open*Cluster*Group.org, Open Source *Cluster* Aplication Resources OSCAR, 2000 [En línea]. Disponible: http://oscar.open*Cluster*group.org/ [Consulta: marzo, 2009].
- Ott J., and Pitkänen M., "Application-aware DTN Routing," sometido a publicación, 2006.
- Parallels Workstation 2.2 for Windows & Linux — Virtual Machine for Linux 2009. [En línea]. Disponible: http://www.parallels.com/products/workstation [Consulta: marzo, 2009].
- Pearson K., Distributed Computing Info, 2009 [En línea] Disponible: http://distributedcomputing.info/ [Consulta: marzo, 2009].
- Pentland A., Fletcher R., and Hasson A., "DarkNet: Rethinking Connectivity in Developing Nations," *Computer*, pp. 78-83. 2004.
- Plan-Tic, "Plan Nacional de Tecnologías de la Información y las Comunicaciones" Ministerio de Comunicaciones República de Colombia, Bogotá D.C. 2008.
- Portal de cooperación Europa Latinoamérica en materia de Sociedad de la Información, 2008. [En línea]. Disponible: http://www.alis-online.org/ [Consulta: marzo, 2009].
- Reacciun2, Sun Microsystem, 2009. [En línea]. Disponible: http://co.sun.com/sunnews/success/, http://co.sun.com/sunnews/success/pdf/Caso%20de%20Exito_CNTI_Reacciun2.pdf [Consulta: marzo, 2009].
- RENATA Red Nacional Académica de Tecnología Avanzada Colombia, 2009. [En línea]. Disponible: http://www.renata.edu.co/ [Consulta: marzo, 2009].
- RFT Admin Guide, University of Chicago Globus Project and Globus Toolkit Chapter 10 2009 [En Línea]. Disponible: http://www.globus.org/toolkit/docs/4.0/admin/docbook/ch10.html#id2560988 [Consulta: marzo, 2009]
- Rivera L. et al, "Sistema de información para el manejo de datos moleculares en café: 1. Desarrollo y uso de herramientas", Rev. Acad. Colomb. Cienc. 32(124): 317-324, 2008.
- Rumiany D, Maya N. "*Grid* Technology: Infraestructure for Development," *Working Paper ICT for Development dgComunity*, 2007.
- Salas J., Pérez F, et al, "WS-Replication: A Framework for Highly Available Web Services," School of Computer Science, Universidad Politecnica de Madrid

- Sato M., et al, "*Ninf: A Network Based Information Library for Global World-Wide Computing Infrastructure*," Proceedings of the Internacional Conference on High Performance Computing and Networking Europe (HPCN Europe), Vienna, Austria, 1997.
- Scyld Penguin Computing, Software, Beowulf.org 2008 [En línea]. Disponible: http://www.beowulf.org/ [Consulta: marzo, 2009].
- SENIAT, Sun Microsystem, 2009. [En línea]. Disponible: http://co.sun.com/sunnews/success/, http://co.sun.com/sunnews/success/pdf/Caso%20de%20Exito_SENIAT.pdf [Consulta: marzo, 2009].
- Serrano C, "Un Modelo Integral para un Profesional en Ingeniería," Universidad del Cauca, 2003.
- Shakeel H., Best M., Miller B., and Weber S., *Comparing Urban and Rural Telecenters Costs, MIT Media Laboratory, E-Development Group, Massachusetts Institute of Technology*: Cambridge, MA, USA, 2001.
- Sotomayor B, "The Globus Toolkit 4 Programmer's," Tutorial Borja Sotomayor University of Chicago Department of Computer Science, 2005.
- Stefano A., "An integrated view of Grid services, Agents and Human Learning," *Towards the Learning Grid,* Vol. 127, U.S.A.: IOSPress, pp. 63-79. 2005.
- Sterling T., "An Introduction to PC *Cluster*s for High Performance Computing," California Institute of Technology and NASA Jet Propulsion Laboratory, 2000.
- Tarricone L., Esposito A., "Grid Computing for Electromagnetics," Artech House Inc, 2004.
- Taylor I., From P2p To Web Services And Grids: Peers In A Clientserver World, Springer. 2004
- Thain D and Livny M., "Building Reliable Clients and Services," in *The Grid: Blueprint for a New Computing Infrastructure (2nd Edition)*: Morgan Kaufmann, 2004.
- TIA, Telecommunications Industry Association 2009 [En línea]. Disponible: http://www.tiaonline.org/ [Consulta: marzo, 2009].
- unGrid Universidad Nacional de Colombia, 2009. [En línea]. Disponible: http://ungrid.unal.edu.co/ungrid.htm [Consulta: marzo, 2009].
- UNICORE, SourceForge.net UNICORE (*Uniform Interface to Computing Resources*) 2009 [En línea]. Disponible: http://www.unicore.eu/ [Consulta: marzo, 2009].
- Universidad Nacional del Sur, Sistemas Distribuidos Modulo 1 –Computación Paralela y Distribuida, 2008
- University of Cambridge Computer Laboratory 2008 [En línea]. Disponible: http://www.cl.cam.ac.uk/research/srg/netos/xen/ [Consulta: marzo, 2009].
- Venugopal S., Buyya R. and Winton L., "A Grid Service Broker for Scheduling Distributed Data-Oriented Applications on Global Grids," Technical Report, *GRIDS-TR-2004-1*, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, 2004.
- Virdell M., IBM Developer Relations "Business processes and workflow in the Web services world," IBM. 2003.
- Virtual-Iron Software, Inc 2009 [En línea]. Disponible: http://www.virtualiron.com/ [Consulta: marzo, 2009].
- Virtual-PC, Microsoft Windows, Microsoft Virtual PC, 2009. [En línea]. Disponible: http://www.microsoft.com/windows/products/winfamily/virtualpc/default.mspx [Consulta: marzo, 2009].
- Virtual-Server, Microsoft Windows, Microsoft Virtual Server, 2009. [En línea]. Disponible: http://www.microsoft.com/windowsserversystem/virtualserver/default.aspx [Consulta: marzo, 2009].
- Virtutech, Simics.net 2000 [En línea]. Disponible: https://www.simics.net [Consulta: marzo, 2009].
- VMware, Inc. 2009 [En línea]. Disponible: http://www.vmware.com/ [Consulta: marzo, 2009].
- W3C, Guía Breve de Servicios Web. W3C 2005. [En linea]. Disponible: http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb [Consulta: marzo, 2009].
- Warthman F., *Delay-Tolerant Networks (DTNs) Tutorial*, Version 1.1 3/5/03, Warthman Associates, 2003.
- WS-GRAM University of Chicago Globus Project and Globus Toolkit GT 4.0 Component Fact Sheet: Web Service Grid Resource Allocation and Management 2009 [En Línea]. Disponible: http://www.globus.org/toolkit/docs/4.0/execution/wsgram/WSGRAMFacts.html [Consulta: marzo, 2009]
- Yañez R.M., "Introducción a las tecnologías de *Clustering* en GNU/Linux" Universidad de Sevilla, 2004.
- Zaglakas G., Grid computing: Meeting the challenges of an On Demand world, 2004. [En línea]. Disponible: http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/zaglakas/index.html [Consulta: marzo, 2009].