Análisis de la resolución de requerimientos no funcionales con Globus Toolkit 4

Lennis Torres Clavijo

Departamento de Ingeniería de Sistemas y Computación

Universidad de Los Andes

Martes 06 de Junio de 2006

Contenido

INDICE DE IMAGENES

INDICE DE TABLAS

INTRODUCCIÓN

El objetivo de este trabajo es establecer pautas para evaluar el aseguramiento de los requerimientos no funcionales de servicios de datos por parte de una tecnología de Grid llamada Globus y específicamente se analiza la herramienta Globus Toolkit 4. A partir de esta evaluación, se establecen criterios de medición y aceptación de la resolución a los requerimientos no funcionales seleccionados mediante la aplicación de experimentos enmarcados en un contexto DataGrid.

Grid se ha centrado en el desarrollo de servicios y herramientas en general que resuelvan los requerimientos no funcionales que aparecen cuando se construyen organizaciones virtuales escalables, tales como la seguridad, la cual requiere el manejo de credenciales para los recursos nuevos, la escalabilidad en hardware, necesitando el manejo de recursos y servicios que soporten el acceso remoto al grid; la escalabilidad en datos, necesitando mecanismos de consulta de los datos y del estado de estos en los recursos, además del manejo del volumen de datos entre recursos de almacenamiento. Estos y otros requerimientos se escogieron para hacer este análisis.

Este documento se divide en tres partes:

La primera es un estudio de la tecnología grid computing desde su conceptualización pasando por su clasificación hasta su aplicabilidad en proyectos actuales.

En la segunda parte se estudia concretamente los tipos de Grid DataGrid, se estudian las tecnologías que en esta categoría son más relevantes, tales como Oracle 10g, Globus Toolkit, OGSA y SQL Server 2000 de Microsoft, también se estudian el middleware más significativos en este tipo de Grid y finalmente, se contrasta el alcance de este documento con beckmarking DataGrid.

Finalmente, se presenta el análisis de los requerimientos no funcionales en la herramienta Globus y se establece inicialmente un planteamiento metodológico que permite de manera científica y controlada la generación y aplicación de experimentos que permiten hacer esta evaluación. Adicionalmente, se hace explícita la aplicabilidad este planeamiento metodológico y se exponen los resultados alcanzados.

1.1 Motivación

La idea de compartir recursos de datos heterogéneos bajo un ambiente controlado no distribuido en donde cada recurso suministra sus datos mediante servicios. Esta idea, es muy común en requerimientos funcionales en donde consultar varias fuentes resulta ser un reto, en esta tecnología es un beneficio que ofrece.

Otra motivación, es el hecho de poder agrupar bajo la figura de grid, recursos computacionales que trabajen conjuntamente para procesos específicos, aunque el alcance del documento no lo cubre, mediante el estudio de la tecnología se puede conocer el esquema que desarrolla.

1.2 Problema Analizado

El aseguramiento del cumplimiento de los requerimientos no funcionales por parte de las tecnologías de DataGrid para las aplicaciones que se desarrollen sobre la herramienta Globus ToolKit 4 específicamente.

1.3 Objetivo Específicos

El objetivo de este trabajo es establecer pautas para evaluar el aseguramiento de los requerimientos no funcionales por parte de las tecnologías de Grid. Se trabaja con las propuestas de tecnología Globus. A partir de esta evaluación, se ofrecen criterios de medición y aceptación de dicha tecnología.

Hacer un planteamiento metodológico que permite hacer de forma científica y controlada la aplicación de experimentos y evaluar la forma en la que la herramienta da solución a los RNF establecidos. A partir de esta evaluación, se ofrecen criterios de medición y aceptación de dicha tecnología.

1.4 Alcance General

- Estudio de los servicios ofrecidos por Globus Toolkit 4 para desarrollar o implementar RNF
- Estudio de las algunas aplicaciones disponibles para la tecnología Grid y particularmente DataGrid.
- Análisis y evaluación de experimentos que miden los servicios Grid para la solución de RNF.

- Planteamientos metodológicos que permitan la aplicación experimentos a la tecnología DataGrid.

1.5 Características de los participantes

- Propiedad por organizaciones e individuos mutuamente desconfiados. En un grid los participantes son los recursos, pero estos recursos no son de un sólo dueño y por ende cada propietario es ''celoso" con sus recursos o con el contenido de estos.

- Requerimientos de seguridad diferentes y políticas requeridas. Cada dueño de recursos establece sus propios requerimientos de seguridad de acuerdo a las políticasque hacia el interior de su organización maneja.

- Recursos potencialmente defectuosos. Los recursos muchas veces resulta no sólo no dar respuesta a los requerimientos sino además ser de tecnologías muy viejas resultando ser obsoletas sin embargo para la tecnología Grid su potencia de computo aporta en pro del Grid mismo.

- Recursos heterogéneos. El grid integra diferentes tecnologías y diversos tipos de recursos, siempre que estos proporcionen servicios a los cliente del grid,

- Geográficamente separados. Un grid se caracteriza por unir recursos que no están en un mismo sitio sino que geográficamente se encuentran distantes entre ellos, valiéndose de la red como medio de encadenamiento,

- Políticas de administracion para los diferentes recursos. Cada dueño de recursosestablece sus propias políticas y es el grid quien debe dar permite el ambiente para que estas diferentes políticas convivan en pro de la compartición de recursos e incluso por cada recurso es posible que un mismo dueño maneje diferentes políticas.

2 ANTECEDENTES Y CONTEXTO

En este capítulo se hace un estudio de la tecnología grid, permitiendo conocer y entender que es la tecnología grid, de donde surge, que ofrece y que existe. Adicionalmente se estudian la taxonomía grid a partir de los tipos de grid.

2.1 Tecnología Grid Computing

El término Grid Computing apareció a mediados de los años 90 en los trabajos de Ian Foster y Carl Kesselman, ellos describen la Grid como: "El grid es una infraestructura de software que permite flexibilizar, asegurar y compartir coordinadamente recursos entre colecciones dinámicas de individuos, instituciones y recursos." [1]. Grid se enfoca en el acceso remoto a recursos computacionales y pretende ser un paradigma de desarrollo sin centrarse en una tecnología concreta, aunque enmarcado dentro de la tecnología de computación distribuida, para manejar la carga de trabajo y la asignación de los recursos disponibles de una organización con el fin de lograr poder de computo. Por lo anterior se establece que Grid engloba conceptos como sistemas operativos distribuidos, programación multiprocesador, redes de computadores, etc.

La definición que Ian Foster y Carl Kesselam dan en el capítulo Computational Grids que aparece en su libro "The Grid Blueprint for a Future Computing Infraestructura" es: Un grid computacional es una infraestructura compuesta por hardware y software que suministra al que la utiliza acceso seguro (en inglés, dependable), consistente (en inglés, consistent), penetrante o (en inglés, pervasive) y a bajo costo (en inglés, inexpensive), a unas elevadas capacidades computacionales [2].

Analizando las características anteriores se puede establecer que grid es un conjunto de recursos (entendiendo recursos como ciclos de CPU, datos, sensores, etc.). Y todos estos recursos necesitan una interconexión física o hardware y un control o monitoreo (usualmente un software) para que estén conectadas en un grid, de aquí que se utilice el termino infraestructura. Esta infraestructura completa, debe proporcionar a los usuarios un servicio seguro a todos los niveles a los cuales accede, tales como capacidad de cómputo, integridad de datos, seguridad de acceso, etc. Además de dar un servicio seguro, este debe ser consistente y debe estar basado en estándares, que permita el acceso y las operaciones sobre el grid de forma tal que no se caiga en heterogeneidades. Ahora, la posibilidad de acceder a cualquier recurso del grid en cualquier sitio, asegura que una vez conectado desde cualquier punto es posible extraer del grid toda la potencia que requiera. Por último el acceso y uso del grid debe tener un bajo costo de manera que sea atractivo para que las organizaciones y que su utilización sea aceptada.

El término Grid Computing describe una arquitectura de red que suministra servicios computacionales, orientados a la entrega de servicios de información, semejante a "on-demand" dado que a través de una red de sistemas de procesadores y almacenamiento, se proporciona el acceso y el procesamiento seguro de los servicios computacionales de la misma manera que los estándares Web proporcionan el acceso universal y transparente a documentos [4].

Grid se puede comparar con P2P (peer to peer) dado que el modelo P2P en especial el modelo Napster permitía a los usuarios acceder y compartir música entre PC, por su parte, los sistemas distribuidos utiliza un modelo similar para reunir los recursos disponibles en clusters separados de PCs o servidores. La tecnología Grid lleva estos clusters al siguiente nivel al conectarlos con otros múltiples clusters ubicados en áreas geográficas dispersas, logrando, de este modo, compartir los recursos y además logra una colaboración mejorada llegando nuevamente a los requerimientos mencionados anteriormente.

El problema central en que se enfoca Grid es poder compartir los recursos de forma coordinada y aportar una solución al problema en organizaciones virtuales multi- institucionales. Este compartir recurso se refiere al acceso directo a procesadores, programas, datos y otros recursos como sean requeridos en la solución de un problema, a través de altos controles a proveedores de recursos y consumidores definiendo de forma clara y cuidadosa qué recursos se compartieron, quién está habilitado a compartir y las condiciones bajo las cuales se efectuará. El conjunto de individuos y/o instituciones definidas por las reglas de compartir recursos forman lo que llamamos organizaciones virtuales o OV. [3]

Las OV se componen de aquellas instituciones, individuos y organizaciones que comparten un objetivo común. Para lograr este objetivo recurren a compartir sus recursos lo cual significa el acceso directo a computadoras, programas, archivos, datos y demás, pero el acceso debe estar controlado, ser seguro, flexible y a menudo limitado en un cierto tiempo.

En principio, Grid Computing puede hacer todo lo que se puede hacer con un supercomputador dado que grid busca ser equivalente ha este, de modo que abre la posibilidad de acceder a estos recursos tan costosos por parte de empresas que no cuentan con la infraestructura o los recursos económicos para adquirirla, dejando de ser limitante y convirtiéndose en ventaja dado que permite que diferentes áreas adquieran motivaciones para invertir o adaptarse a ella, tales como:

- Organizaciones Gubernamentales: Para planificación y seguimiento a problemas de orden público (como el terrorismo), planificación de acciones y seguimiento a desastres naturales, y todo proyecto que maneje información que permita hacer análisis para la elaboración de contingencias o estrategias que permitan controlar o erradicar los diferentes problemas.

- Organizaciones Médicas y de Salud: En el área de la salud se genera una gran cantidad y variedad de información generada a partir de las diferentes visitas de los pacientes a los diferentes servicios de salud, se generan datos acerca del estado de salud de cada uno (signos vitales, antecedentes personales, antecedentes familiares, etc.) esto conlleva a un plan diagnóstico tal como laboratorios, radiografías, etc., esto genera más información aun, dado pie a la necesidad de requerimientos como transaccionalidad de datos, fiabilidad y

confidencialidad de la información, eficiencia en la consulta y recuperación de la misma y la facilidad de análisis de datos distribuidos dada su naturaleza dispersa y diferente de ésta con el fin de encontrar casos similares o hacer un diagnostico más exacto.

- Organizaciones Educativas. Internet se consolida como herramienta educativa dado que facilita, beneficia y amplia la consulta de diferentes temas que refuerzan los procesos de aprendizaje y consulta en general. Sin embargo integrar en un Grid las Bibliotecas Electrónicas y los centros de e-Educación facilitarían el acceso a datos dispersos y la creación de aulas virtuales con estudiantes y profesores distribuidos de modo que se mejorará enormemente la situación actual.
- Empresas y Multinacionales. Un grid en estos ámbitos permitiría la administración, seguimiento y control del personal y los recursos distribuidos en pro de la realización de proyectos que permitan o implique su desarrollo de manera simultanea desde sus diferentes y distantes puntos.
- Comunidad Científica: esta área es la que más ha trabajado sobre el tema dado los avances o aportes que a dado a la tecnología de Grid Computing es esta comunidad, la cual requiere de recursos extras que les permitan analizar la gran cantidad de datos recopilado, la mayoría de los proyectos existentes bajo esta tecnología son desarrollados en esta área.

2.1.1 Características de Grid

- Números os recursos. En este tipo de tecnología prevalece el concepto de gran cantidad de recursos para ser compartidos.
- Propiedad por organizaciones e individuos mutuamente desconfiados. En un grid los participantes son los recursos, pero estos recursos no son de un sólo dueño y por ende cada propietario es ''celoso" con sus recursos o con el contenido de estos.
- Requerimientos de seguridad diferentes y políticas requeridas. Cada dueño de recursos establece sus propios requerimientos de seguridad de acuerdo a las políticas que hacia el interior de su organización maneja.
- Recursos potencialmente defectuosos. Los recursos muchas veces resulta no sólo no dar respuesta a los requerimientos sino además ser de tecnologías muy viejas resultando ser obsoletas sin embargo para la tecnología Grid su potencia de computo aporta en pro del Grid mismo.

- Recursos heterogéneos. El grid integra diferentes tecnologías y diversos tipos de recursos, siempre que estos proporcionen servicios a los cliente del grid,

- Geográficamente separados. Un grid se caracteriza por unir recursos que no están en un mismo sitio sino que geográficamente se encuentran distantes entre ellos, valiéndose de la red como medio de encadenamiento,

- Políticas de administracion para los diferentes recursos. Cada dueño de recursos establece sus propias políticas y es el grid quien debe dar permite el ambiente para que estas diferentes políticas convivan en pro de la compartición de recursos e incluso por cada recurso es posible que un mismo dueño maneje diferentes políticas.

- Conectados por redes multi-nivel heterogéneos. Cada dueño de recursos puede tener diferentes arquitecturas de red y estas son conectadas mediante el grid mismo.

2.1.2 Ventajas y Desventajas de Grid

Lo anterior conduce a establecer los siguientes beneficios, que hacen llamativa y alternativa la tecnología Grid Computing:

- Brinda una solución a problemas que requieren de un gran número de ciclos de procesamiento o acceso a una gran cantidad de datos.

- Permite a las organizaciones integrar recursos heterogéneos (recursos antiguos y nuevos) a su infraestructura tecnológica sin importar en donde estén localizados, brindando utilidades que controlan costos, seguridad, disponibilidad y aprovechamiento de los recursos controlando la subutilización de estos. Adicionalmente, permite crear una infraestructura más robusta y resistente, capaz de responder a requerimientos de alta disponibilidad,

- Permite a las organizaciones contar con un conjunto de capacidades de integración horizontal que gestiona de forma efectiva los recursos de toda la empresa, e incluso extienden la solución entre múltiples organizaciones.

- Permite a las organizaciones mejorar la calidad de los productos y servicios ofrecidos y mejorar su tiempo de respuesta, reduciendo costos y proporcionando una ventaja competitiva.

- Permite a las organizaciones construir una infraestructura de información de coste efectivo que asegure la completa utilización de la tecnología existente, cuya potencia computacional sumada sea considerable, eso permitiría generar sistemas distribuidos de muy bajo costo y gran potencia computacional.

- Permite a las organizaciones acceder y compartir bases de datos remotas. Esto permite mantener enormes cantidades de informaciónes dispersos abriendo la posibilidad de analizarlas continuamente.
- Permite a las organizaciones adoptar las nociones de flexibilidad y libertad de elección de estándares de seguridad y compartición de recursos, siendo capaces de descubrir dinámicamente y ajustarse a los entornos cambiantes y fluctuantes de lastecnologíasde la información.
- Facilita la posibilidad de compartir, acceder y gestionar información, mediante la colaboración y la flexibilidad operacional, pese a la diversidad de recursos (equipos, personas y hasta aptitudes).
- Permite a las organizaciones aumentar la fiabilidad de la infraestructura ante la recuperación de desastres o calamidades, los departamentos de tecnología de la información pueden mejorar la fiabilidad y disponibilidad de sus infraestructuras tecnológicas.

Hasta este punto Grid computing se presenta como la solución a muchos problemas, pero para que esta tecnología exponga todas las bondades anteriores necesita de diferentes servicioscomo Internet, conexiones 7 x 24 los 365 días, con banda ancha, VPN, firewalls, encriptación, comunicaciones seguras, políticas de seguridad, etc.

2.1.3 Interés de Grid Computing

Las necesidades de cálculo, espacio para almacenamiento de los datos y tiempo de respuesta lleva a definir cinco grandes áreas de trabajo determinadas:

2.1.3.1 Supercomputación distribuida.[2]

Dentro de esta área se encuentran aquellas aplicaciones cuyas necesidades es imposible satisfacer en un único nodo o punto. Estas necesidades se producen en instantes de tiempo determinados y consumen muchos recursos, por lo que se dice que son puntuales e intensivas. Clásicos ejemplos de este tipo de aplicaciones son las simulaciones, las herramientas de cálculo numérico, los procesos de análisis de datos, la extracción de conocimiento de almacenesde datos, etc.

2.1.3.2 Sistemas distribuidos de alto desempeño. [2]

En estas aplicaciones están consideradas todas aquellas que generan un flujo de datos a alta velocidad que debe ser analizado y procesado en tiempo real. Ejemplo de las aplicaciones

experimentales de física de alta energía, control remoto de equipos médicos de alta precisión y precio, todos los procesos de la denominada e-Medicine, el tratamiento de imágenes para la visión artificial, etc.

2.1.3.3 Proceso intensivo de datos. [2]

En esta área se encuentran aquellas aplicaciones que hacen un uso intensivo del espacio de almacenamiento. Las necesidades de almacenamiento de estas aplicaciones desbordan la capacidad de almacenamiento de un único nodo y los datos son distribuidos por todo el grid. Además de los beneficios por el incremento de espacio, la distribución de los datos a lo largo del grid permite el acceso a los mismos de forma distribuida. Por ejemplo los sistemas gestores de bases de datos distribuidas.

2.1.3.4 Servicios por Demanda.[2]

El concepto de potencia de cálculo y capacidad de almacenamiento a considerar algunos recursos de una organización como no necesarios. La tecnología grid ofrece a la organización esos recursos sin que la organización deba desarrollarlos por si misma. Ejemplos de este tipo de aplicaciones son aquellas aplicaciones que permiten acceder a hardware muy específico (equipos costosos de medida o de análisis de muestras) para la realizar operaciones a distancia.

2.1.3.5 Computación Colaborativa.[2]

Está directamente relacionado con el concepto de interacción humano a humano, de manera que se utilizan los enormes recursos computacionales del grid y su naturaleza distribuida para generar entornos virtuales 3D distribuidos.

2.1.4 Tecnología Inherente a un Grid

Desde el momento en el que los PC comenzaron a conectarse a Internet, surgió la idea de unir la potencia desperdiciada de cada uno para afrontar problemas a los que sólo podían enfrentarse las supercomputadoras que están al alcance de grande organizaciones o entidades educativas superiores o gubernamentales. La tecnología Grid como ya se menciono se basa en el aprovechamiento de los ciclos de procesamiento no utilizados por los millones de computadores conectados a Internet.

Gracias a esta conexión se consigue que puedan resolver tareas que son demasiado pesadas para ser resueltas por una única máquina. Básicamente la idea es contribuir a este esfuerzo, entonces una aplicación detecta cuando un PC está inactivo y en ese instante activa los procesos de cálculo que permiten poner en marcha las unidades de trabajo. El sistema emplea un servidor

que administra las unidades de trabajo y gestiona el control de las tareas que dichas unidades tienen asignadas en cada momento.

Partiendo de las ventajas y facilidad sobre las que Internet surge proporcionando la búsqueda de información existe, mediante la combinación de palabras claves, con este tipo de búsquedas tarde o temprano se encontrará lo que se necesita, pero de lo que realmente se trata es de encontrar la información en el menor tiempo posible. Además, dentro de un mismo buscador, el resultado puede variar dependiendo del refinamiento la consulta a la base de datos [9] [10] [11] [12] [5]. Un motor de búsqueda o mecanismo de búsqueda (search engine) es un programa que realiza búsquedas dentro de una base de datos que en el caso de Internet, la base de datos es de recursos web. Un robot, según el WWW Robots FAQ, es un programa que de manera automática atraviesa la estructura de documentos Web extrayendo un documento y a partir de éste extrayendo recursivamente todos los documentos que está referenciados por enlaces. Los documentos son almacenados en una base de datos e indexados con el fin de permitir su localización por un mecanismo de búsqueda. Un índice o directorio es una recopilación manual de documentos, que pueden mantenerse como directorio o bien ser introducidos también en una base de datos para permitir que se realicen búsquedas.

Estos robots surgen con el fin de medir el tamaño del WWW, sin embargo se convirtieron en herramientas muy útiles para localizar documentos. El criterio para seleccionar las páginas que visita un robot depende de cada robot, es decir, parten de una lista de servidores inicial, y a partir de ahí va visitando los diferentes enlaces de cada página hasta un nivel arbitrario respecto al inicial. Cuando un robot entra en un nuevo servidor, busca un fichero que se llama robots.txt, en el que se le indican los directorios permitidos y los prohibidos. Si este fichero no existe, considera todos permitidos. A los robots se les puede solicitar direcciones de páginas para que sean visitadas e incluidas en la base de datos mediante el diligenciamiento de un formulario (submission form). La manera en que cada robot indexa el contenido de las páginas que visita varía, algunos robots indexan los títulos de páginas HTML, los primeros párrafos o el contenido entero del documento, etc.

Los motores de búsqueda realizarán búsquedas dentro de una base de datos de documentos, recibe la consulta del usuario, que esta conformada por una o varias palabras, realiza la búsqueda en la base de datos, y extrae una lista ordenada de documentos que cumplen completa o parcialmente con la consulta. El orden de la lista depende de una puntuación que asocia el programa a cada documento cuando realiza la búsqueda, y éste también varia en cada caso.

Las técnicas de recuperación de información empleadas por los motores de búsqueda en Internet, provienen de las técnicas empleadas en el campo de los Sistemas de recuperación de Información y surgiendo grandes problemas en momento que se realizan operaciones de recuperación de

información con ellas, dado que el entorno de trabajo no es el mismo y las características intrínsecas de los datos almacenados en los mismos son considerablemente diferentes.

La mayoría de los motores de búsqueda emplean el Modelo Conceptual del Espacio Vectorial y muestran sus resultados ordenados según un algoritmo de ranking, sin embargo tienen características que los diferencian, particularmente el motor de búsqueda Google [11] desarrollado en la Universidad de Stanford en California, el cual se focaliza en realizar un uso más eficiente del espacio de almacenamiento y cuidar que el índice no se convierta en un elemento contraproducente para el rendimiento de respuesta del motor.

Google surge rompiendo el concepto que un índice completo de un motor de búsqueda permitiría encontrar cualquier documento de forma fácil, siempre y cuando todos los documentos estén incluidos en la base de datos. Sin embargo, el entorno Web más que la localización de documentos es la calidad de estos frente la consulta, dado que, como se mencionó anteriormente, puede retornar documentos que no hacen realmente referencia al tema consultado, resultando difícil encontrar entre los primeros documentos recuperados. Es por eso que el objetivo principal del diseño de Google es mejorar los índices de precisión en la recuperación de la información y mejorar la presentación de los documentos recuperados de manera que, los primeros resultados correspondan a los relacionados con las consulta solicitadas por los usuarios.

Google se caracteriza usar la conectividad de Internet proporciona para calcular un grado de calidad de cada página, este proceso de asignar el grado se denomina PageRank Adicionalmente, Google utiliza esta capacidad de conexión de los documentos web para mejorar los resultados de búsqueda. Pagerank asume que el número de enlaces que una página proporciona tiene que ver con la calidad de la página. PageRank puede verse como un modelo del comportamiento del usuario, basandose en los enlaces de las páginas visitas pero que pretende representar la forma de trabajar de los usuarios.

2.1.4.1 Proyectos de Grid Computing

- SETI@home: es el proyecto pionero del concepto de Grid Computing, establecido por el Instituto de Búsqueda de Inteligencia Extraterrestre (SETI). En este trabajo los datos recogidos en el observatorio de Arecibo (Puerto Rico) son procesados por más de dos millones de computadores de todo el mundo. Este proyecto es muy conocido mundialmente y una de las razones por las que tuvo tanto éxito es el objetivo científico del proyecto, la búsqueda de vida inteligente en el espacio. Para alcanzar ese objetivo se necesitaba una enorme cantidad de cómputo y ésta se conseguía cuando los propietarios de los equipos donaban sus ciclos de procesamiento inactivos para ayudar a buscar signos de vida extraterrestre procedentes del espacio. El propietario de un computador sólo tenia

que descargar de Internet un protector de pantalla al que un ordenador central envía pequeñas partes de los grandes procesos de cálculo. Una vez solventados por cada máquina, cuando su usuario no la está utilizando, eran enviados de nuevo al computador central. Esta iniciativa no sólo interconectó millones de computadores alrededor del mundo, permitiendo tener una capacidad de cómputo muy superior a la de un supercomputador, sino que tambien trajo a los directores del proyecto ahorros importantes en recursos y en capital necesitado. SETI@home se aseguró de tener la capacidad necesaria para analizar Terabytes de datos previamente recopilados por un radiotelescopio, sin contar con el gasto de un supercomputador[28].

- Globus Project: El Proyecto Globus es una iniciativa multi-institucional para la investigación y el desarrollo de tecnologías fundamentales para Grids, con la activa participación de la empresa IBM, cuya intención principal es crear una plataforma completa donde compartir aplicaciones y recursos informáticos en Internet. El Proyecto Globus[12] tiene su sede central en el Laboratorio Nacional Argonne y la Universidad del Instituto de Ciencias de Información de California del Sur. El proyecto permitirá llevar las redes Grid más allá de las habituales aplicaciones técnicas y científicas para que pueda ser de utilidad en aplicaciones reales de negocio, conectando muchos superordenadores dispersos geográficamente mediante Internet y unos protocolos específicos de código abierto creados por la organización internacional Globus. [5] . Uno de los primeros productos desarrollados por el Proyecto Globus[5] es el Globus Toolkit, que está siendo utilizado en varios proyectos de aplicación y despliegue de Grid en los Estados Unidos, Europa y el resto del mundo, se trata de una arquitectura abierta, un sistema de protocolos, servicios y herramientas que permiten una Grid segura y distribuida.

- Globus Alliance : El impacto tecnológico que ha generado el concepto de Grid ha removido esa estructura, actualmente se habla de la Globus Alliance [16], y ha generado multitud de agrupaciones, como por ejemplo el Global Grid Forum que agrupa el Grid Forum [25], es una comunidad en foro de investigadores individuales y usuarios que se enfocan en la promoción y el desarrollo de las tecnologías Grid y las aplicaciones mediante el desarrollo y la documentación de los avances, los mejores resultados, guías de implementación y estándares con énfasis en el consenso.

- Data Grid Project fundado por la Unión Europea, este proyecto se concentra en proveer la tecnología Grid necesaria en la investigación científica de próxima generación que requiere enormes cantidades de poder de procesamiento, análisis de datos y el tratamiento de millones de Gigabytes, a lo largo de diversas comunidades científicas dispersas geográficamente.

Otras empresas se han centrado más en el desarrollo de supercomputadores mediante la conexión de ordenadores en red, es el caso de HP y son muy conocidas las interconexiones de sus supercomputadores a grids de investigación, como la conexión del supercomputador de HP (9.2-teraflop) al Grid Científico del DOE (Department of Energy's Pacific NorthWest National Laboratory). Otras empresas han desarrollado nuevas versiones de sus productos utilizando las ideas de esta tecnología, por ejemplo, Oracle con su producto 10g, es una evolución de 9i a la tecnología Grid.

- GridSystems (www.gridsystems.com) es una compañía española fundada en Febrero de 2000 que ha desarrollado el productor InnerGrid para la aplicación de la tecnología grid en entornos empresariales, académicos y de investigación. Este software divide datos y procesos en pequeñas unidades que se ajustan dinámicamente a los recursos conectados a la red. El sistema es multiplataforma (Windows y todos los sistemas UNIX). Esta empresa ha desarrollado trabajos para diversas empresas pero su desarrollo más conocido es la red grid de la Universidad Politécnica de Valencia. En esta red se integra el servidor de la serie ALTIX 3000 de Silicon Graphics, con 48 procesadores Itanium 2, el cluster de IBM con 64 nodos, biprocesador Xeon, y, las 3000 CPU's con las que cuenta la universidad en aulas y departamentos. Este proyecto sitúa a la UPV a la cabeza de España en lo que se refiere a capacidad de cálculo disponible.

- ENTROPIA (www.entropia.com) es una empresa centrada en el PC Grid Computing, uno de sus trabajos más interesante es el que ha llevado a cabo en el centro de supercomputación de San Diego. El producto que desarrolla se denomina DCGrid Platform que permite utilizar una red de CPUs para aprovechar los ciclos libres de CPU aumentando la capacidad de cómputo global con un bajo coste. De esta manera los ciclos libres son utilizados por aplicaciones que consumen muchos recursos y necesitarían de un supercomputador o por trabajos puntuales e intensivos en una fracción de tiempo. La instalación del producto en una empresa permite la utilización de máquinas que han quedado obsoletas y la instalación de nuevas máquinas sin que se interfiera en el trabajo de la compañía, ya que el software es fácilmente escalable.

- Spitfire es un proyecto de Paquete de Trabajo dentro del Proyecto DataGrid Europeo(EDG). Ofrece un servlet en Java que acepta el pedido de la base de datos utilizando protocolos HTTP/HTTP y demuestran el resultado en un XML. TrustManager es un software de EDG utilizado para las autenticaciones de Spitfire; este software analiza los derechos de los usuarios para ejecutar basando las preguntas en su certificado de usuario (el grid). En TrustManager la autenticación va de la mano con el protocolo SSL y la Seguridad de Capa de Transporte (TLS). El servidor y el cliente se envían sus certificados X. 509 y los mensajes cifrados por sus llaves privadas que son relacionadas a las llaves

públicas incluidas en los certificados X. 509. De esta manera el servidor y el cliente se autentican a sí mismo como dueños del certificado [2].

2.2 Taxonomía Grid según Tipos de Grid

Existen diferentes tipos de grids dada la naturaleza de los requerimientos funcionales, como grid de investigación, grid de utilidad, entre otros, cada uno con necesidades especificas, sin embargo tienen una característica común proveer recursos optimizados para permitir diferentes funciones. Existen tres tipos de grids [9]:

2.2.1 Grid de Recursos

Provee mecanismos para el uso coordinado de recursos como ordenadores, archivos de datos, servicios e instrumentos de laboratorio. En este tipo de Grid los usuarios anónimos no pueden acceder a este sin las credenciales necesarias a las facilidades y ventajas otorgadas por del Grid de Recursos. Sólo los usuarios autorizados y previamente registrados pueden utilizarlo. La idea principal de este tipo de Grid es el proveer accesos sencillos, transparentes y eficientes a cualquier recurso independientemente de su localización. Estos recursos pueden ser desde poder de procesamiento, almacenamiento de información, ancho de banda, etc.

En esta categoría se encuentran los Grids Computacionales, que permiten el acceso a superordenadores distribuidos para realizar tareas que consumen mucho tiempo. La mayoría se basan en la herramienta Globus Toolkit. Hoy en día hay diversos prototipos de Grids dado los temas esecíficos que abarque tales como química, astrofísica, simulación del clima, geología, etc. También están los Grids de Datos o DataGrids, los cuales proveen mecanismos para el almacenamiento seguro y redundante en sitios esparcidos geográficamente. Igualmente los Grid Gubernamentales caen en esta categoría.

2.2.1.1 Grids Computacionales [9]

Los grid Computacionales hacen uso de lo ciclo de cómputo inactivos en pro de la ejecución de aplicaciones de cómputo intensivo propios en las comunidades científicas. Este tipo de grid permite la creación de ambientes que proveen niveles de desempeño de supercomputadoras. Dentro de este tipo de grid se encuentra:

- El proyecto Seti@Home usado para buscar vida extraterrestre y vinculo millones de computadores alrededor del mundo.

- El proyecto e-Science Grid de la Universidad de Oxford dirigido al acceso a grande colecciones de datos, al accedo a recurso de gran escala de procesamiento y alto desempeño para usuarios científicos individuales.
- El proyecto TeraGrid, centrado en las redes WAN de alta velocidad, soportado por un red que opera a 40 Gbps, y almacenamiento distribuido que incluye 20 teraflops de poder de cómputo y un petabyte para datos.

2.2.1.2 Grids de Datos o Data Grids [9]

Debido a los desafíos ocasionados al almacenar y procesar cantidades de Petabytes de datos en diferentes localizaciones, se convierte en un tema extendidamente demandante, dado requerimientos tales como la replicación, obtención, catalogación y la coordinación de estos datos deben ser aun refinados. Los Data Grids proveen recursos de cómputo permitiendo el análisis de datos compartidos en bases de datos dispersas. El mayor problema que los data grids ayuda a solucionar es la integración de datos. Las empresas a menudo necesitan para el análisis de desempeño en tiempo real de diversidad de datos provenientes de una variedad de fuentes externas e internas. Muchos vendedores grid ofrecen ambiente metadata que son capaces de tomar fotos o estados del momento pero de diferentes bases de datos y presentar esta fotos en aplicaciones bajo un formato que permite su exploración. Algunos proyectos conocidos que están bajo este tipo son:

- El proyecto Crossgrid, se trata del modelado y simulación de regiones susceptibles de inundaciones para predecir inundaciones y proporcionar datos a equipos de crisis en caso de inundación.
- Spitfire permite acceder a bases de datos relacionales desde el Grid, su objetivo es proporcionar un acceso transparente y seguro a bases de datos de metadata para aplicaciones y otro middleware Grid. Esta diseñado para metadata de las aplicaciones, no para sus datos.

2.2.1.3 Grids Gubernamentales [9]

Las entidades gubernamentales hacen uso de la computación distribuida para propósitos de defensa e inteligencia. Son usadas para reducir costos de operación para hacer mejor uso de los recursos y estimular las investigaciones y descubrimientos científicos.

2.2.2 Grids de Servicios

Proporciona una visión de las tecnologías de la información centrada en servicios, que se encuentra sujeta a unos acuerdos en los niveles de servicio. Sus principales características ser

transparente al usuario y estar siempre disponible. Proporciona recursos de manera balanceada a las necesidades del usuario. Se evalúa con base al uso realizado del servicio y persigue una reducción en los costos. La tecnología Grid es una herramienta que permitirá al Grid de Servicios ofrecer servicios de forma global a Comunidades Virtuales, rompiendo barreras organizativas y liberando las capacidades de los mismos. Este tipo de grid se encarga de hacer entrega de servicios y aplicaciones sin importar la ubicación geográfica, implementación o plataforma de hardware. Los servicios son montados en los recursos concretos disponibles en el Grid de Recursos.

Una de las mayores diferencias entre los grid de recursos y los grid de Servicios es que el Grid de Servicios provee servicios abstractos sin importar su localización, mientras que el Grid de Recursos facilita accesos a recursos concretos ofrecidos en un sitio en particular.

Grid es por tanto un impulsor de la empresa adaptada al cambio en donde los servicios son concebidos como servicios potentes a la vez que flexibles, con la agilidad de cambiar en línea con las prioridades de la organización. Permiten suministrar las aplicaciones y asignar las capacidades necesarias entre grupos de trabajo geográficamente dispersos según cambien sus necesidades, el beneficio claro es un uso más eficiente de los recursos, un mejor manejo en los picos de la demanda de los mismos, así como la colaboración global segura entre grupos extendidos que crucen las fronteras organizativas. Dentro de este tipo caen los tipos de grid:

2.2.2.1 Grids de Optimización Empresarial [9]

Los Grids Empresariales son usados para responder mejor a los cambios en la demanda del entorno, puede ser usado para optimizar el desempeño de los sistemas de información y almacenamiento. Se caracteriza por ofrecer:

- Reducción de costos de administración de sistemas permitiendo un acceso transparente y provee la oportunidad de consolidar servidores y almacenamiento.
- Reducir costos de capital y operación, de personal y software.
- Rápido desarrollo y publicación de aplicaciones
- Crea una plataforma de común para encadenar aplicaciones legacy a ambiente Web

2.2.2.2 Grids Colaborativas [8]

Este tipo de grid es usada para dar y procesar datos. Estos datos son presentados en forma visual a menudo para localizaciones geográficamente dispersas. Un vendedor involucrado en el diseño y

desarrollo de este tipo de grid es Silicon Graphic Inc. (SGI). Existen números os ejemplos de este tipo:

- Compañías Aeroespaciales. Lo usa para reducir los sistemas de información y los costos de almacenamiento.
- Compañías Automotrices. Es usada para compartir información de productos diseñados, los diseñadores usan sistemas locales y los ingenieros analizan los resultados de lo diseñado desde diferentes puntos de vista.

2.2.3 Grid de Información

Partiendo del crecimiento exponencial de Internet creó una infraestructura de red pública y de facil acceso, que permite la entrega de información de cualquier tipo a cualquier lugar en el mundo. Esta información se puede obtener al conectar cualquier computador a una red telefónica pública. Se caracteriza este tipo de grid por brindar servicios de compartición de archivos como lo hace Napster, Gnutella Network, E-Donkey forman parte del Grid de Información actual. A diferencia de Internet, los datos compartidos no se encuentran respaldados por una organización o dueño de algún sitio Web, sino que el servicio para compartir archivos es dispuesto por personas que desean intercambiar archivos tales como música, videos o software. El servicio de intercambio se mantiene gracias a los participantes, no hay un repartidor central involucrado. Es un ambiente distribuido, dinámico y altamente flexible.

Para solucionar los problemas que surgen al interconectar las diferentes Grids para formar una Grid global, se están aplicando conceptos y tecnologías provenientes del mundo de Internet como pueden ser Web Services; la problemática de seguridad de acceso, disponibilidad, descubrimiento y registro de nuevos servicios (funcionalidades proporcionadas por cada Grid) se están resolviendo utilizando las mismas tecnologías que en los servicios Web distribuidos. Entre estas tecnologías está la de acceso vía portales. Estos portales permiten acceder a los recursos de computación desde cualquier dispositivo dotado de un navegador para lanzar tareas, comprobar su estado y recoger los resultados.

2.2.4 Arquitectura del Grid

Las aplicaciones, los toolkits, las APIs, los SDK, etc. que se definen para la computación en Grid deben de cumplir una arquitectura general que fue presentada por Ian Foster, Carl Kesselman y Steve Tuecke en su artículo "Anatomy of the Grid: Enabling Scalable Organizations"[1]. Esta arquitectura general se articula en cinco niveles: la infraestructura, la conectividad, la gestión del recurso, la gestión de varios recursos y el nivel de aplicación.[3].

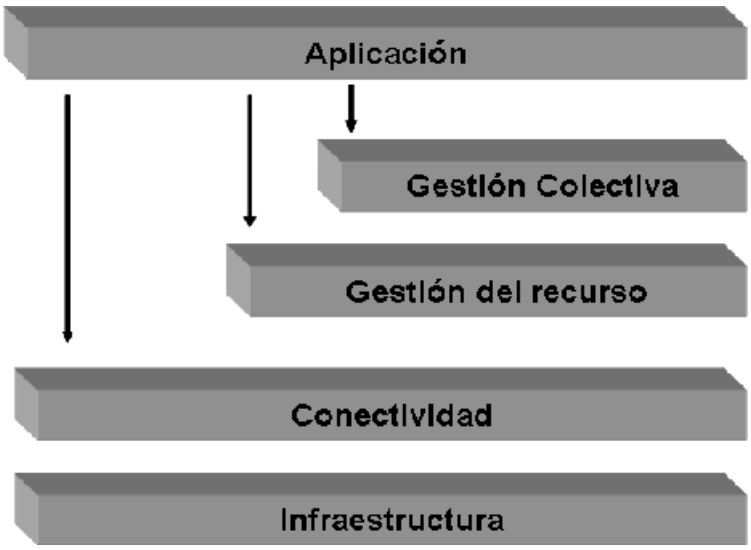

Figura 1 Arquitectura presentada por Ian Foster, Carl Kesselman y Steve Tuecke en su artículo "Anatomy of the Grid: Enabling Scalable Organizations"[1].

Esta arquitectura propuesta es una arquitectura de protocolos que definen los mecanismos básicos que permiten a los usuarios y a los recursos negociar, establecer, gestionar y explotar el hecho de compartir los recursos. Un recurso puede ser una entidad lógica, un sistema de archivo distribuido, un cluster de computadoras o un pool de computadoras distribuido, en esos casos la implementación del recurso puede involucrar protocolos internos. Una arquitectura abierta basada en un estándar facilita la extensibilidad, la interoperatibilidad, la portabilidad y el compartir código. De esta manera la estandarización de los protocolos permite estandarizar los servicios y mejorar las capacidades del grid[1].

En la terminología Grid la infraestructura se denomina la Fábrica y suministra los componentes que serán compartidos, es decir, todo los recursos que se compartiran en el grid, tales como cluster, supercomputadoras, bases de datos, etc..En este nivel se encuentran recursos específicos, tales como [1]:

- Recursos computacionales: mecanismos requeridos para iniciar programas, controlar y monitorear la ejecución de procesos resultantes. El manejo de mecanismos que permite el control sobre los recursos alojados a procesos son usados como mecanismos avanzados de reserva. Se necesitan algunas funciones de evaluación para la determinación de características de hardware y software así como información de estado relevante.
- Recursos alojados: se requieren algunos mecanismos para el guardado y recuperación de archivos. Hay mecanismos para la lectura y escritura de subconjuntos de archivos y/o ejecución remota de una selección de datos o reducción de funciones. El manejo de estos mecanismos permiten el control sobre los recursos alojados, para transferencia de datos y otras funciones específicas.

- Recursos de red: mecanismos de manejo que provean control sobre los recursos y la transferencia en la red, funciones que provean las características de red y carga.
- Repositorios de recursos: es una forma especializada en almacenamiento de recursos que requiere mecanismos de manejo de versiones de fuentes y código de objetos.
- Catálogos: es la forma especializada de almacenamiento que requiere mecanismos para implementar consultas y actualización de operaciones, como las base de datos relacionales.

El nivel de conectividad define el centro de las comunicaciones y protocolos de autenticación requeridos para las transacciones en la red , es decir, incluye los protocolos de comunicación y seguridad que permiten a los recursos de la capa de Infraestructura comunicarse entre si brindando mecanismos seguros para la verificación e identificación e usuarios y recursos. Entre estos protocolos se encuentran: la pila de protocolos TCP/IP, el protocolo SSL, Certificados X.509. También se incluyen los nuevos protocolos que se encuentran en fase de estudio y que permitirán mejorar el rendimiento en las redes de alta velocidad. La seguridad es un punto muy importante de la computación en grid por su propia naturaleza distribuida ya que se comparten recursos entre distintas organizaciones que pueden tener distintas políticas de seguridad. Las soluciones de Grid de seguridad deben ser capaces de interoperar con varias soluciones locales. Además, Usuarios basados en las relaciones confiables: en el cual el sistema de seguridad no debe requerir que cada proveedor de recursos coopere o interactúe con cada una de las configuraciones de los entornos de seguridad [1].

El nivel de recurso construye sobre la capa de conectividad protocolos de Comunicación y Autenticación, se centra en la gestión de un único recurso y permite tener información y control sobre el mismo. En este nivel se encuentran los protocolos que permiten obtener la información de un recurso: las características técnicas, la carga actual, el precio, etc. También se encuentran los protocolos que permiten el control del recurso: el acceso al mismo, el arranque de procesos, la gestión, la parada, la monitorización, la contabilidad de uso y la auditoria del recurso [1].

Las implementaciones de estos protocolos llaman a funciones de capa Fabric para acceder y controlar recursos locales, englobando todos los servicios que permiten gestionar un conjunto de recursos. Los protocolos de esta capa se refieren a recursos individuales y por ende ignoran resultados de estado global y acciones atómicas a lo largo de colecciones distribuidas, éstos son de consideración para la capa colección. Existen dos tipos de protocolos en esta capa [1]:

- Protocolo de información: para obtener información sobre la estructura y estado de los recursos.

- Protocolos de manejo: usados para negociar el acceso a un recurso compartido, especificando por ejemplo sus requerimientos (incluyendo reservas avanzadas ó calidad de servicio), y operaciones para optimizar estos recursos, como ser la creación de procesos o acceso a datos. A partir de que el manejo de protocolos es responsable para la inicialización de las relaciones del compartir (idea base del grid), deben servir como puntos de aplicación de política, asegurando que las operaciones requeridas por el protocolo sean consistentes con las políticas bajo las cuales el recurso es compartido.

En la capa de recursos se encuentran los servicios de directorio, que permiten localizar los recursos que son de nuestro interés; los schedulers distribuidos, que permiten asignar las tareas a cada recurso; la monitorización y diagnóstico de la ejecución de las distintas tareas en que se distribuye la ejecución de una aplicación; la contabilidad, que permite calcular el coste de la utilización de varios recursos heterogéneos; y, el acceso a datos distribuidos, que gestiona la replicación de datos. El servicio de scheduler distribuido es una de las aplicaciones más complejas de un desarrollo grid ya que existen tres scheduler distintos [1]:

- El planificador de trabajos (job scheduler) que intenta maximizar la cantidad de trabajo realizado (trabajos por unidad de tiempo), priman la eficiencia del sistema grid.
- El planificador de recursos que intenta maximizar el uso de los recursos, priman la eficiencia del sistema grid
- El planificador de la aplicación que divide la aplicación en tareas, asigna los recursos para su ejecución y vigila el desarrollo de los mismos, prima la eficiencia de la aplicación.

La capa Colectiva contiene los protocolos y servicios – API's Y SDK, que no están asociadas con un recurso específico sino que es global y captura de interacciones a través de colecciones de recursos de ahí su nombre de colectiva. Dado que los componentes de capa colectiva se construyen sobre capa de Recursos y capa de Conectividad pueden implementar una amplia variedad de comportamientos para el compartir recursos sin la localización de nuevos requerimientos sobre los recursos almacenados. Por ejemplo [1]:

- Servicios de directorio: permite a participantes de las OV descubrir la existencia y/o propiedades de los recursos de OV. Permite a sus usuarios hacer consultas para recursos por el nombre y/o por atributos, como ser tipo, disponibilidad o carga.
- Co-Allocation, Scheduling y servicios brokering permiten a recursos de participantes de las OV averiguar sobre más recursos para propósitos específicos y para la programación de tareas sobre los recursos apropiados.

- Monitoreo y diagnóstico de servicios: soportan el monitoreo de los recursos de las OV para fallas, ataques de adversarios, detección de intrusos, sobrecarga y más.
- Servicios de replicación de datos: soporta el manejo de almacenamiento de los recursos pertenecientes a las OV para maximizar el rendimiento en los accesos a datos con las respectivas métricas como el tiempo de respuesta, costo, etc.
- Sistemas de programación GRID-ENABLED: permite usar en los entornos Grid modelos de programación familiar usando varios servicios Grid para localizar recursos, seguridad, etc.
- Sistemas de manejo de Workload y Collaboration Frameworks: provee para la descripción, uso y manejo de multi-steps, asíncronos, flujos de trabajo multi-componentes, etc.
- Sistemas de ubicación de software: descubre y selecciona las mejores implementaciones de software y ejecución de plataformas basadas en parámetros del problema a resolver. Ejemplo: NetSolve y Ninf.
- Servidor de autorización de la comunidad: refuerza el gobierno de las políticas de acceso a recursos generando capacidades que los miembros de la comunidad pueden usar para el acceso a recursos comunitarios.
- Servicios de cuentas de la comunidad y pagos: junta información acerca del uso de información para el propósito de manejo de cuentas, pagos, y/o limitación de usos de recursos a miembros de la comunidad.
- Servicios de colaboración: soporta el intercambio coordinado de información dentro de comunidades de usuarios potencialmente grandes. Mientras los protocolos de capa Recursos deben ser generales en naturaleza y desarrollados ampliamente, los protocolos de capa Collective expanden su espectro desde propósitos generales a aplicaciones altas o dominios específicos sólo entre OV específicas.

Las funciones de esta capa pueden implementarse como servicios persistentes con protocolos asociados o SDK`s designados para enlazarse con ciertas aplicaciones, en ambos casos sus implementaciones pueden construir en capa Recurso protocolos y API´s. Los servicios colectivos (collective services) se basan en protocolos: protocolos de información que obtienen datos sobre la estructura y estado de los recursos, y protocolos de manejo que negocian el acceso a recursos de una forma uniforme [1].

El último nivel, es el de aplicación, se centra en la definición de protocolos que permitan a las aplicaciones el acceso a la infraestructura del grid a través de las distintas capas. Según el tipo de aplicación puede ser necesario conectarse a las distintas capas o acceder directamente a una de

ellas, incluso directamente a la infraestructura. Cada una de estas capas tiene protocolos bien definidos que proveen acceso al uso de servicios: manejo de recursos, acceso a datos, etc.

Si se considera que una aplicación de usuario necesita analizar datos contenidos en archivos independientes, tendrá que realizar entonces las siguientes tareas básicas:

- Obtener la credencial necesaria de autenticación para abrir los archivos (recursos y protocolos de conectividad)
- Consultar el sistema de información y réplica de catálogos para determinar dónde pueden encontrarse las copias de los archivos en Grid así como también dónde se hallan los recursos más convenientes para hacer el análisis de datos (Collective Services)
- Pasar los pedidos a Fabric, la computadora apropiada, sistema de almacenamiento y redes, para extraer los datos, iniciar los procesos, y proveer los resultados (recursos y protocolos de conectividad)
- Monitorear el progreso de varios procesos y transferencia de datos, notificando al usuario cuándo el análisis se completa y también detectando y respondiendo ante situaciones de fallas (collective services).

3 SERVICIOS GRID

Uno de los campos de mayor innovación en el uso del grid computing, fuera de los conceptos de supercomputación, es el desarrollo de un estándar para definir los Grid Services frente a los actuales Web Services. Los Web Services es una tecnología de computación de sistemas distribuidos que permite la creación de aplicaciones basándose en el modelo cliente/servidor mediante una plataforma independiente del lenguaje y utilizando protocolos abiertos como HTTP. Son aplicaciones basadas en Internet que interactúan de forma dinámica con otras aplicaciones basadas en Web utilizando estándares como XML, UDDI y SOAP. Tanto IBM, como Microsoft y Sun están decididas a unir la tecnología grid con los Web Services. Mediante la versión de Globus Toolkit se pueden encontrar grid services en la red. También posibilita la creación de una interfaz para el servicio y permite invocar una instancia de un grid service. Con estas capacidades el desarrollo de los Grid Services es sólo cuestión de tiempo[5].

IBM en el campo de los Grid Services esta haciendo el intento de adaptación de su software WebSphere SDK for Web Services, a las nuevas ideas del Grid (en particular en lo referido a seguridad). En IBM se pueden encontrar ejemplos de aplicación real de la accesibilidad a través de Web Services que utilizan como sustento la tecnología grid. A modo de ejemplo se puede acceder

a su sistema de base de datos DB2 a través del Open Grid Service Architecture - Data Access and Integration (OGSA-DAI) que es un entorno que permite la integración de recursos de datos, como bases de datos relacionales y basadas en XML, en un entorno grid. Básicamente la aplicación soporta el protocolo SOAP para los mensajes entre un cliente y un servidor que cumple las especificaciones de OGSA-DAI.

En esta versión se introduce el concepto de grid service que es una ampliación de los Web Services y cuya arquitectura está especificada por el Global Grid Forum (OGSA, Open Grid Services Architecture y OGSI, Open Grid Services Infraestructure). La idea era buscar una tecnología de objetos distribuidos que se adaptase a las necesidades de una aplicación grid, y se pensó en los Web Services, aunque estos presentaban algunas limitaciones que se superaron [22]:

- Los Web Services no mantienen el estado de una invocación a otra, los grid services si.
- Los Web services no son transientes, es decir no se pueden crear varias instancias de un mismo servicio según se necesita y destruirlas cuando ya no son necesarias, en los grid services si se puede.
- Los Web Services no incluyen servicios de apoyo que han sido incluidos en los grid services como son las notificaciones, el servicio de persistencia, la gestión del ciclo de vida, etc.

Los grid services utilizan un enfoque de Factorías de Objetos de manera que en lugar de tener un único servicio compartido por todos los usuarios (como el Web Service) se tiene un servicio-factoría que crea instancias individuales del servicio. Cuando se invoca a una operación del servicio se accede a la instancia y no a la factoría. Además se puede crear una instancia por cliente, o varias por cliente o una para varios clientes. Por último la destrucción de la instancia puede correr a cargo del cliente o de la factoría. A los Servicios Grid se les incorporan una serie de mecanismos de la plataforma OGSI. Estos mecanismos adicionales se pueden agrupar en cuatro áreas [23]:

- Servicio de Nombres (Naming), que asegura la existencia de un nombre único para cada instancia de Servicio Grid y permite la localización de servicios Grid (discovering) mediante nombres.
- Servicio de Datos, que gestiona los conjuntos de datos asociados a la ejecución de un servicio Grid
- Notificación, es decir el conjunto de interfaces para registrar y suministrar notificaciones y subscripciones. Estos son los mecanismos usados para la comunicación entre los componentes de una aplicación Grid.

- Ciclo de Vida, mecanismos para la creación y destrucción de instancias de Servicios Grid

4 DATAGRIDS

A nivel de información, Internet también ha gestado la necesidad de tecnología Grid, en pro de un mejoramiento en las consultas realizadas por los motores de búsqueda. Antes de establecer cómo los requerimientos no funcionales impactan en la necesidad de una tecnología que proporcione una solución o por lo menos pautas de solución, es necesario para el análisis, remontarse al pasado a revisar como la carga de trabajo se va incrementando dada la potencia de procesamiento que la tecnología brinda y el avance mismo de la tecnología, la cual, ha permitido como consecuencia, el rápido crecimiento del volumen de información disponible en ámbitos como Intranet e Internet, haciéndose necesario mejorar los mecanismos de búsqueda de información y aprovechar al máximo las posibilidades que nos ofrece la propia Internet.

Primero surgió el concepto de Grid Computing con el fin de compartir recursos de cálculo a gran escala, permitir un acceso transparente para el usuario, flexibilidad y seguridad. El Data Grid extiende los recursos a compartir a los datos, compartir servidores, grandes bases de datos distribuidas, "Data mining" sobre bases de datos heterogéneas. Hoy en día los proyectos Grid también contemplan compartir otros recursos como instrumentos de medida, tales como telescopios, termómetros, etc.

La industria que estudia acerca de la proliferación de la computación distribuida concluye que la Infraestructura Tecnológica esta utilizando sólo una fracción de su capacidad de procesamiento instalado. Grid Computing es la solución de Oracle (http://www.oracle.com/technologies/grid/) para mejorar la utilización de la capacidad creando grupos de servidores industriales (cluster) conectados en grupos de forma modular, por ejemplo, formando el sistema de almacenamiento [22]. Los nuevos productos de Oracle (http://www.oracle.com/technologies/grid/): Oracle Database 10g, Oracle Application Server 10g, Oracle Enterprise Manager 10g y Oracle 10g Development Tools son compuestos de una sola imagen de la base de datos manejada y procesada a través de varios nodos que componen el grid. Ese grupo de nodos son configurables desde el Administrador de Control del Grid de Oracle que es una consola de monitoreo que da a los administradores la habilidad de planificar y desplegar los recursos de procesamiento de base de datos de un nodo a otro.

Estas nuevas versiones se centran en la aplicación de las ideas de Grid Computing al almacenamiento de datos. La nueva versión parte de la arquitectura actual en la que se encuentran la mayoría de los sistemas de información. Cada una de las capas es de uso dedicado a cada

sistema de información. Evidentemente, el fallo de alguno de los componentes de los tres niveles (Application Server, Database y Storage) supondría una perdida del servicio.

Oracle entrega esta funcionalidad de grid a través de un legacy de discos compartidos agrupando tecnología, el RAC de Oracle (el Grupo de Aplicaciones Reales). RAC se instala en cada nodo del grid para permitir que los nodos ejecuten o realicen pedidos de procesamiento a la base de datos. Esos pedidos se pueden publicar desde un computador cliente, un servidor de aplicaciones middleware, o por una aplicación que los procesa en el nodo él mismo.

Combinando grupos de procesamiento que corren el kernel de DBMS con las herramientas DBA se puede asignar dinámicamente tareas de procesamiento a nodos del grupo, permitiendo:

- Responder a demandas crecientes de recursos computacionales asignando carga de procesamiento de una máquina a otra.
- Agregar nuevo nodos de procesamiento y unidades de almacenamiento al grupo estrictamente cuando sea necesitado y cuando la capacidad acumulada del ambiente comienza a requerir un esfuerzo.

La evaluación de la necesidad de Oracle para mayor eficiencia en ambientes computacionales tiene mérito, y la Base de datos 10g del Oracle ofrece nuevas características para responder al desafío de la utilización de la capacidad dentro de la empresa. Sin embargo, la arquitectura de la Base de datos 10g del Oracle no proporciona una solución "fuera de la caja" para aplicar, lo que la Alianza de Globus llama: Los problemas verdaderos y específicos que subyacen el concepto de grid son coordinando los recursos compartidos y resolviendo problemas en organizaciones dinámicas y multi-institucional virtuales.

En la actual versión, toda intercomunicación entre nodos sucede dentro de aplicaciones de Oracle. Su modelo de seguridad carece de transparencia lo que requiere extender el Grid más allá de la plataforma del Oracle, y el apoyo de 10g para el descubrimiento de recursos es una característica crítica del estándar de Grid ya que es limitado a un director manejar las políticas basado en el manejado de la carga de trabajo.

Finalmente, Oracle propuso la solución de grid computing asumiendo que hay siempre máquinas sobrantes disponibles y sus cargas del trabajo son previsibles y ocurren en tiempos diferentes para sistemas diferentes. Por supuesto, esto es a menudo distante de la realidad en ambientes comerciales. Así que en vez de tener servidores con capacidad de procesamiento que en pocas ocasiones es completamente utilizada, los clientes que adoptan el modelo de Oracle puede acabar con los servidores adicionales que son subutilizados.

Oracle con la nueva arquitectura basada en grid computing pretende crear "granjas/factorías de almacenamiento" y servidores estándares y modulares, a las que se puede añadir servidores rápidamente.

Las características de esta arquitectura [6]:

- Capacidad de balanceo de sistemas [6]. No hay necesidad de calcular la capacidad de los sistemas en función de los picos de trabajo, ya que la capacidad se puede reasignar desde la granja de recursos a donde se necesite. Oracle 10g ha mejorado su arquitectura RAC (Real Application Clusters) y permite gestionar clusters con sus dos nuevas funcionalidades:
- Automatic Service Provisioning [5]. Permite la asignación y reasignación de los servidores a las cargas de trabajo (servicios). Los clientes hacen autenticación a los servicios y son automáticamente encaminados al servidor apropiado según el momento.
- Integrated Clusterware [6]. Permite la gestión (configuración, instalación, etc) de clusters.
- Alta disponibilidad. Con la nueva funcionalidad (Automatic Service Provisioning), si un servidor falla, Oracle reasigna los servicios en los servidores restantes.
- Reducción de costes [6]. Con esta arquitectura los servicios son gestionados por "granjas de recursos". No siendo necesario disponer de "grandes servidores" y de este modo se puede hacer uso de componentes de bajo coste. Cada sistema puede ser configurado siguiendo el mismo patrón.
- Facilidades de administración. Oracle 10g reduce el número de parámetros de configuración. Además, se ha introducido el Automatic Workload Repository para almacenar información sobre diagnósticos de rendimiento y recomendaciones de tuning que Oracle realiza automáticamente. Para el análisis de esta información ha introducido un nuevo motor de diagnóstico llamado Automatic Database Diagnostics Motor.

La utilización de esta versión permite a una empresa mejorar sus prestaciones sin renunciar al comportamiento tradicional de los Sistemas de Información ya que Oracle 10g también es operativo en un entorno con un único servidor (arquitectura tradicional). Además el gestor de la base de datos de la versión 10g es un 28% más rápido que el de la versión 9i, y se incorporan versiones mejoradas de alguna herramientas que fueron introducidas con la versión 9i como son: Flashback Recovery (una herramienta pensada para errores de usuario, con ella podemos "rebobinar" la actividad de la base de datos hasta el momento del error) y Data Guard (aporta nuevas funcionalidades para las copias de seguridad).

Todas estas ventajas son adquiridas por las empresas que están interesadas en incursionar en esta tecnología, sin embargo, la situación económica de muchas otras no permiten hacer inversiones tan costosas como es el licenciamiento de Oracle, pero esto no es un impedimento para la tecnología Grid dado que apoyados en su principio de compartir recursos, se permitirá el acceso a los recursos de forma que se explote en poder de computo del grid a favor de estas empresas.

4.1 Data Grid vs. Grid Computing

Grid surge como una infraestructura que permite compartir, seleccionar y agregar recursos, el Data Grid extiende los recursos a compartir a los datos:

- Compartir servidores: cinta, disco.
- Grandes bases de datos distribuidas.
- "Data mining" sobre bases de datos heterogéneas.

El papel de las base de datos dentro de la tecnología grid es como:

- Base de datos de las aplicaciones, conservando los datos en si.
- Metadata de las aplicaciones, manejando catálogos de recursos de datos o dando el servicio que permite seleccionar los recursos de datos que se quieren usar.
- Metadata del "middleware", para actividades como Monitorización y Réplicas

4.2 Tipos de Servicios Data Grid

Los servicios fundamentales del DataGrid son el servicio de la administración de metadata y el servicio de acceso a los datos. Existen también servicios de alto nivel de la arquitectura del DataGrid, tal como el servicio de administración de replica y servicio de selección de réplicas.

El servicio de administracion de metadata se especializa en tres tipos de metadata:

- Metadata de aplicaciones la cual define la estructura lógica o semántica de una instancia o grupo de instancias de esta metadata, esta metadata es más pequeña que la instancia de datos que para efectos de entendimiento hacia delante, cada instancia se llamara archivo de datos sin especificar si se trata de una base de datos o un archivo plano por ejemplo,

adicionalmente es más eficiente la consulta a través del catalogo de la metadata que en una colección de archivos de datos.

- Metadata de replica: esta maneja las copias de los archivos de datos y ofrece información del mapeo de la referencia lógica de un archivo a la localización física de la replica de un archivo en el DataGrid.
- Finalmente, la metadata de configuración del sistema que describe la información física del sistema de almacenamiento y del DataGrid, tal como el tráfico de la red, la capacidad y el uso de políticas.

Los servicios de metadata se focalizan en la metadata de aplicaciones o al mecanismo de acceso y administración de la información a través de los datos compartidos en el DataGrid. La metadata de replica es usada en los servicios de administracion de réplicas mientras que la metadata de configuración de metadata es usada en los servicios de acceso y selección de réplicas.

Los servicios de acceso a datos incluyen acceso, administración e invocación de transferencia de archivos de terceros en el sistema de almacenamiento, el cual como componente del DataGrid proporciona la funcionalidad de creación, escritura, lectura, manipulación y eliminación de instancias de archivos. Estas funciones de accesos a datos deben ser coherentes con los requerimientos de seguridad de acceso remoto a los sitios que mantiene los archivos de datos. La capacidad de conservación y aislamiento en los sistemas de almacenamiento impactan en la solidez del DataGrid. Por esto es que la aplicación de acceso debe proporcionarle al sistema de almacenamiento las información acerca de las pautas de acceso y la carga de trabajo de la red para que el sistema de almacenamiento pueda optimizar su comportamiento conforme a esta información. Así mismo, el sistema de almacenamiento debe proveerle esta información de desempeño a la aplicación de acceso , de modo que optimice su comportamiento e informe errores a las aplicaciones.

El administrador de réplicas es un servicio de alto nivel del DataGrid y es el encargado de la creación, eliminación y descubrimiento de copias de archivos de datos en el DataGrid. Este servicio mantiene un repositorio o catalogo de réplicas, donde cada entrada es un nombre de archivo físico asociado con uno o más localizaciones físicas permitiendo a los usuarios escoger acceder a archivos de sistemas de almacenamiento con mejor desempeño, adicionalmente, le permite al mismo DataGrid jerarquizar de modo que un nombre de archivo lógico pueda ser punto de entrada para otro nivel en el catalogo.

El servicio de selección de replica, permite escoger una copia de archivos de datos en particular del servicio de catalogo de réplicas basado en un grupo de criterios de selección, incluyendo accesibilidad, desempeño o velocidad, costo y seguridad.

4.3 Algunos Proyectos en desarrollo bajo el sistema Data Grid

- TeraGrid: Es un proyecto estadounidense, llevado adelante por la Fundación Nacional de Ciencias (NSF). Tiene el objetivo de interconectar instalaciones y centros de investigación académica en puntos geográficamente distantes, está considerado como una de las infraestructuras más grandes y más rápidas del mundo. [27] . Mediante este proyecto, los científicos tendrán la capacidad para simular actividades sísmicas a fin de diseñar edificios y puentes más seguros, los astrónomos podrán compartir datos desde sus telescopios y los investigadores médicos tendrán la posibilidad de compartir ideas y datos para quizá curar una enfermedad, lo que asegura que las oportunidades que se ofrecen a los científicos son incalculables.

- CrossGrid: proyecto de la Unión Europea nacido en 2001, es un middleware adaptado para ejecutar aplicaciones interactivas en un entorno Grid. En este proyecto se han definido cuatro aplicaciones que utilizarán desarrollos Grid comunes, que son: 1) Simulación interactiva y visualización de un sistema biomédico; 2) Sistema de apoyo a un equipo de crisis por inundaciones; 3) Análisis de datos distribuidos en Física de Altas Energías y 4) Previsión meteorológica y modelización de la contaminación atmosférica.

- e-Ciencia: este proyecto del Reino Unido une los proyectos científicos relacionados con áreas como el análisis de la física de partículas, la biología computacional, la medicina, las ciencias medioambientales y la astrofísica, la cual sólo puede ser comprendida con el avance de la tecnología Grid o de computación distribuida. Precisamente, entorno a esta tecnología y a la e-Ciencia han surgido números proyectos y múltiples centros de investigación se han centrado en su desarrollo. [IFIC]

- UK e-Science: Un equipo de científicos del Reino Unido dio a conocer en una conferencia de gran magnitud, un elemento clave de la computación Grid que facilitará a los investigadores aprovechar enormes recursos informáticos de todo el mundo para afrontar los desafíos científicos clave en campos como el genoma humano y la física de partículas. Los responsables de la iniciativa elaboraron un conjunto de procedimientos que permitirán que los científicos que utilicen Grid accedan a las bases de datos de los resultados de investigación procedentes de sistemas que se encuentren en cualquier parte del mundo. [29]

- EGEE (Enabling Grids for e-Science in Europe): El proyecto utilizará la tecnología Grid para interconectar recursos computacionales de veintisiete países europeos, con el objeto primordial de unir los recursos de los equipos informáticos de las instituciones participantes y crear de este modo un supercomputador virtual, aprovechando la infraestructura de

comunicación de banda ancha proporcionados por la Red Europea de Investigación Géant. [30]

- Spitfire permite acceder a bases de datos relacionales desde el Grid. Su objetivo es proporcionar un acceso transparente y seguro a bases de datos de metadata para aplicaciones y otro middleware Grid.
- OGSA-DAI: es un "middleware" que permite el acceso e integración de fuentes de datos distribuidas en el Grid.

4.4 Tecnologías Subyacentes

4.4.1 SOAP (Simple Object Acess Protocol

Lenguaje de mensajera basado en XML, que especifica todas las reglas necesarias para ubicar los servicios Web, y establece la comunicación.

Con SOAP se puede cambiar el conjunto de caracteres a utilizar: US-ASCII, UTF-8 y UTF-16.

4.4.2 UDDI (Universal Description, Discovery and Integration)

Servicio de directorios donde los usuarios se pueden registrar y buscar servicios Web.

Almacena información de servicios Web en un directorio, mostrando las interfaces que fueron definidas en WSDL.

4.4.3 WSDL (Web Services Definition Language)

Documento escrito en XML en el que se describen las operaciones que un servicio ofrece. Utiliza los elementos:

<types>

<binding>

<message>

<PortType>

4.5 Tecnologías Relevantes

Conforme a los tipos de Grid se puede establecer que existe en el entorno. En cuanto a lostipos de grid de recursos son un poco más difíciles de implementar ya que los recursos son costososy no pueden ser entregados al público en general sin costo alguno. Sin embargo existen Grids Computacionales, que permiten el acceso a supercomputadores distribuidos para realizartareas que consumen mucho tiempo. La mayoría se basan en la herramienta que se ha convertido en el estándar en esta área, el Globus Toolkit.

También están los Grid de Datos, los cuales proveen mecanismos para el almacenamiento seguro y redundante en sitios esparcidos geográficamente. Debido a los desafíos ocasionados al almacenar y procesar cantidades de Petabytes de datos en diferentes localizaciones, se convierte en un tema extendidamente demandante. Temas como la replicación, obtención, catalogación y la coordinación de estos datos deben ser aun refinados.

Por su parte los grid de servicios disponibles como las máquinas de búsqueda, portales, páginas de servidor activas y diverso contenido dinámico. Normalmente son gratuitosdebido a patrocinioso publicidad. Servicios de email y autorización como Passport, GMX y Hotmail recaen en esta categoría. Con los web services y el Open Grid Service Architecture OGSA están diseñadospara proveer interoperabilidad entre los servicios sin importar la implementación, localización geográfica o plataforma de ejecución.

La tecnología Grid se ha convertido en uno de los mayores éxitos en la tecnología y la comunicación, es usada por un gran número de la población mundial para acceder a información actual. Una de las razones de su éxito es el concepto de hipervínculo, una referencia hacia otras webs que es muy fácil de usar, seguir los hipervínculos es comúnmente la manera más rápida para encontrar información sin tener que teclear la información. Otra causa para el éxito de esta red ha sido la facilidad con la que se puede actualizar la información, de esta forma ésta se mantiene al día.

4.5.1 Oracle 10 g

A nivel de datos, Oracle presenta Oracle10g, un software orientado específicamente a infraestructura Grid Computing, el cual se adapta a los cambios que el negocio necesita. El producto de Oracle 10g no da un gran aporte en términos de nuevas capacidad de procesamiento para sistemas distribuidos. El soporte de Microsoft a las arquitecturas orientadas a servicios mediante Web Services y .Net Remoting combinados con los beneficios de escalabilidad que ofrece Windows Server, es el cual es actualmente ofrecido como una solución para el procesamiento distribuido.

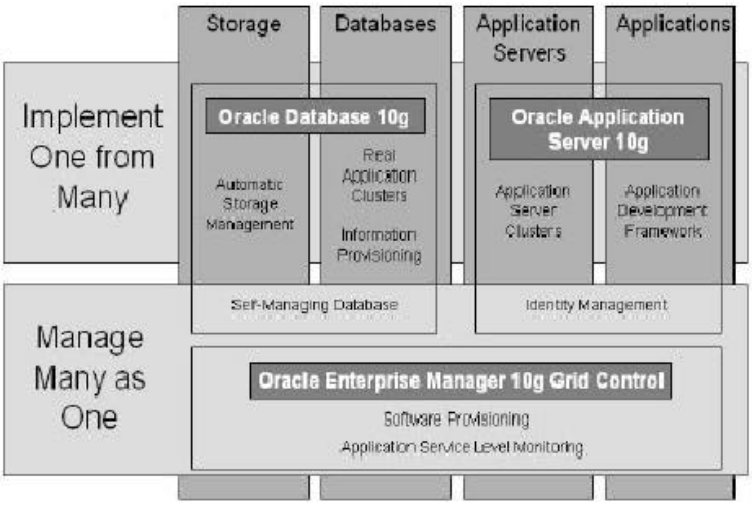

Figura 2 Arquiterura Oracle 10g [5]

4.5.1.1 Real Application Clusters

Grupo de Aplicaciones Reales (Real Application Clusters – RAC) han aumentado para manejar las características de carga de trabajo compartida en una base de datos Oracle 10g. Mientras que Oracle ha realizado notables avances en la estabilidad de RAC en 10g. En muchas situaciones, los usuarios pueden percibir los mismos niveles de escalabilidad y disponibilidad entre Oracle y SQL Server en un ambiente SMP diferenciándose únicamente en costo y complejidad.

4.5.1.2 Inteligencia de Negocio

Los productos OLAP y Data Mining de la base de datos Oracle 10g son diseñados para proveer funcionalidad de Inteligencia de Negocio para los clientes a escala empresarial de Oracle. Sin embargo, esto viene con un costo adicional tanto de licenciamiento como en integración. Por su parte Microsoft ofrece como un sólo servicio de Inteligencia de Negocio a ambos productos, fuera del SQL Server, sin embargo su integración no es tan costosa como en Oracle.

4.5.1.3 Administrabilidad

El fácil manejo es un factor importante a la hora de adoptar un Servidor. Oracle ha dado un gran paso en incorporar herramientas y wizard a 10g para simplificar la administración de tareas, con la presencia de un administrador de base de datos que las ejecute. La administración de recursos de SQL Server es más automática dado que hace una asignación dinámica de los recursos basado en la carga de trabajo y en la disponibilidad actual del recurso sin la necesidad de intervención humana. Oracle ha respondido muy bien frente a la queja de la complejidad de su DBMS y a la dificultad de manejabilidad que se presenta en los usuarios con las herramientas en 10g que son similares a las de SQL Server versión 7.0

4.5.1.4 Costo Total de Propiedad

Oracle es claramente superior en TCO (Total Cost of Ownership) con respecto a la plataforma Windows, solamente proporcionando la estrategia “Enterprise Grid” el cual reclama un costo-beneficio en la optimización de la utilización de recursos, este beneficio derivado de la mejora en estabilidad, la alta disponibilidad y la administrabilidad.

4.5.2 Microsoft en Grid Computing

Microsoft ha sido un miembro pionero en la mayor industria trabajando los grupos que se enfoca en la tecnología de grid. Microsoft Research es un patrocinador de la Alianza de Globus y Microsoft recibió el julio 2000 Taller del Foro de Grid. La Corporación Oracle llegó a ser un nuevo miembro del Foro de Grid en 2003.

Microsoft reconoció un principio de Grid Computing cuando este proporciono software desarrollados con herramientas para crear en Internet basándose en servicios computacionales. En el 2000, el apoyo para Web Services se identificó como un componente clave en el framework de .NET. Casi cuatro años más tarde, la plataforma .NET y las herramientas de desarrollo para construir las aplicaciones para arquitecturas de orientadas a servicios continúan madurando. Algunos evidencia de la viabilidad de este enfoque está ya disponible al público tal como "Web Services for the Virtual Observatory", una cooperativa entre la Universidad de Johnes Hopkins y Microsoft Research.

4.5.2.1 Comparación de la solución Grid de Oracle con la solución Microsoft

Oracle parte del principio que la escalabilidad se logra por la habilidad de agregar servidores al grid y realizando balanceo de cargas en los servidores existentes. Sin embargo, no tiene en cuenta la complejidad de la administración introducida y que las aplicaciones que no se construyen para ser manejadas en grupos(clusters) no puede funcionar en esta arquitectura. La escalabilidad de las aplicaciones que corren en plataformas de Windows es soportada por el Comité de Microsoft para asegurar que los productos Sistemas Servidor de Windows continúen beneficiándose de los avances de hardware hechos por servidores basados en SMP. Esto en línea con la demanda de los clientes para desarrollar menos servidores con sistemas nuevos y para consolidar las granjas existentes de servidores en sistemas más manejables con menos máquinas.

La tecnología Grid Computing es una evolución que permite dar solución a los requerimientos no funcionales administrados de manera independiente. La administrabilidad de las herramientas actuales permite hacer una extracción de su potencialidad, de modo que los requerimientos puedan ser atacados de manera transversal a los requerimientos funcionales sin ver afectados estos últimos.

A nivel de confiabilidad, ninguno de los motores tienen problemas con esto, eso lo demuestra la acogida en empresas comerciales de dichos productos, pero si se puede establecer bajo que ambientes hay disponibilidad de la información, dado que existe infraestructura como por ejemplo en Microsoft y COM+ cuya disponibilidad depende no sólo del desarrollo de estos sino de la infraestructura en la que este funcionado.

A nivel de eficiencia y tiempo de respuesta, es interesante establecer como se comporta un modelo con una carga considerable que permita tomar decisión de volumen de los recursos compartidos entendiendo recursos como la información almacenada.

4.5.3 Globus Toolkit

Se basa en tecnologías estándar como XML, SOAP, WSDL, Servicios Web y está implementado íntegramente en Java. Su arquitectura se estructura en varias capas. Los servicios Grid, en la implementación GT3.2, son básicamente Web Services, pero construidos siguiendo un conjunto de especificaciones definidas por OGSI. Dentro de GT3.2 se contempla el soporte de versiones anteriores del toolkit (compatibilidad hacia atrás) mientras se produce la migración de sus usuarios a GT3.2.

Cambiando de proveedor y retomando el tema de la herramienta, cabe menciona la herramienta principal generada por Globus Alliance es el Globus Toolkit El Globus Toolkit es una colección de componentes software que ofrecen la infraestructura básica necesaria para la creación y ejecución de aplicaciones distribuidas, así como para la construcción de Grids. Actualmente, Globus se ha convertido en el estándar de facto para la computación en Grid. Consta de tres componentes: [15]

- Gestión o manejo de recursos (GRAM)
- Servicios de información (MDS)
- Gestión o manejo de datos (GridFTP)

Globus, utiliza en cada uno de los componentes el protocolo de seguridad GSI para la comunicación y autentificación, considerando la Seguridad como un componente.

Estos componentes facilitan el acceso transparente y seguro a recursos distribuidos geográficamente en diferentes dominios de administración, además de servir como herramientas básicas para implementar las fases de la planificación de trabajos en Grids tales como: descubrimiento, selección y preparación de recursos; y envío, monitorización, migración y finalización de trabajos.

La versión actual del Globus Toolkit , GT4 (a Abril de 2005), y cuenta ya con la implementación a gran escala de Open Grid Services Architecture (OGSA), que es la arquitectura Grid definida por la organización [22]. La arquitectura de GT4 es la siguiente:

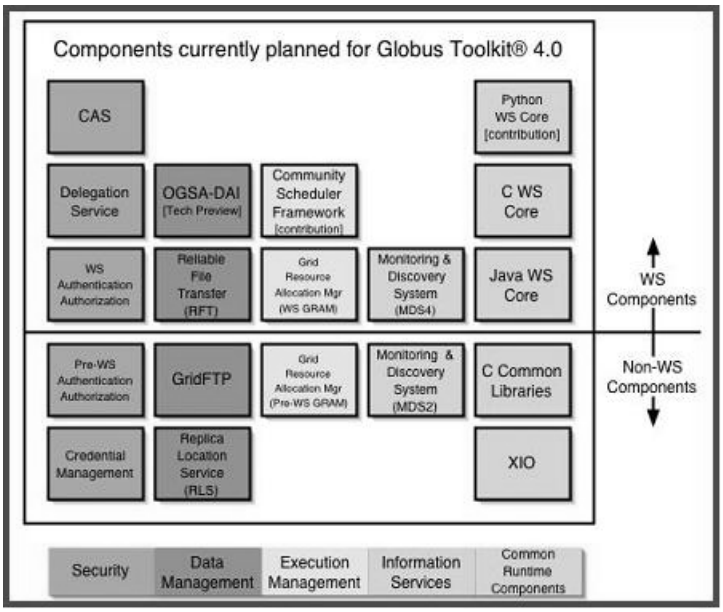

Figura 3. Arquitectura Globus Toolkit 4 [24]

- WS_Seguridad

GSI realiza cuatro funciones básicas: Protección de mensajes, autenticación, delegación y autorización, basándose en cuatro estándares:

1. TLS(a nivel de transporte) o WS-Security y WS-SecureConversation(a nivel de mensaje)
2. X.509 End Entity Certificates o Usuario-Contraseña para autenticación
3. X.509 Proxy Certificates y WS-Trust para delegación
4. SAML para autorización

Autenticación y Autorización: controla el acceso a los servicios y recursos, permitiendo el uso de métodos implementados por el usuario.

Delegación: servicio que delega credenciales a un contenedor

Autorización comunitaria: el Servicio de Autorización Comunitaria (CAS) permite administrar políticas en Organizaciones Virtuales sobre sus recursos.

Manejo de Credenciales: simpleCA y MyProxy–GSI-OpenSSH: parche del OpenSSH para aceptar autenticación con certificados.

En esta parte de los servicios de globus se estará haciendo alusión más adelante dado que atiende requerimientos de seguridad.

- WS Manejo de Datos :

Servicios de descubrimiento, transferencia ya acceso a grandes recursos de datos. Mediante los siguientes servicios

1. GridFTP: cliente y servidor FTP optimizado para transferencias de grandes cantidades de datos.
2. Reliable File Transfer (RFT): servicio de transferencia confiable, a prueba de interrupciones.
3. Replica Location (RLS): servicio de ubicación de réplicas
4. Data Replication (DRS): Servicio de replicación de datos
5. OGSA-DAI: acceso e integración de datos, permite integrar conjuntos de datos en formatos diferentes.

En esta parte de los servicios de globus también se estará haciendo alusión más adelante dado que atiende requerimientos de transaccionalidad, escalabilidad y rendimiento.

- WS Control de Ejecución:

Ejecución, calendarizacion y monitoreo de trabajos. Mediante los siguientes servicios:

1. WS-GRAM: ejecuta y monitorea trabajos
2. Community Scheduler Framework (CSF): interfaz unificada de diferentes calendarizadores (PBS, Condor, LSF, SGE).
3. Workspace Management: permite crear y administrar espacios de trabajo en hosts remotos.
4. Grid Telecontrol Protocol: servicio WSRF para control remoto de instrumentos.

- WS Servicios de Información:

Permite monitorear y descubrir recursos en el grid. Mediante el servicio MDS, el cual esta compuesto por los siguientes servicios:

1. Servicio de indexado: permite agregar recursos de interés a la grid.

2. Trigger Service: colecta datos de los recursos y realiza acciones basado en esa información.

3. WebMDS: proporciona una vista apta para un navegador de los datos colectados por el servicio de indexado.

- Otros Servicios Grid: otros servicios no GT4, que son desarrollados sobre los API en JAVA, C y PERL disponibles dentro del Common Runtime Components.

Los componentes de Globus Toolkit 4 son los siguientes:

- Globus Resource Allocation Manager (GRAM):

Gestiona peticiones y asignaciones de recursos para ejecución de aplicaciones.

- Monitoring and Discovery Service (MDS):

Sistema basado en LDAP para obtener información de distintos componentes de la Grid (capacidad de proceso, ancho de banda, tipo de almacenamiento, etc.). MDS permite recolectar información de los elementos para uso en la Grid. Permite construir un espacio de nombres uniforme entre varias organizaciones.

- Grid Security Infrastructure (GSI):

Proporciona autenticación y comunicación segura en Grid. Además, proporciona seguridad a través de fronteras de organizaciones y login único (single sign-on) para los usuarios de la Grid, incluyendo delegación de credenciales. Se basa en certificados con clave pública X.509 y SSL (secure sockets layer), con extensiones para aplicaciones específicas de Grid.

- Grid Resource Information Service (GRIS):

Proporciona un medio uniforme de preguntar por los recursos de la Grid, como la configuración actual, capacidades y estado. Tales recursos podrían incluir servidores, almacenamiento, nodos en la red, bases de datos y redes de enlace.

- GridFTP:

Mecanismo de transferencia de datos seguro y robusto para Grid. Basado en FTP (file transfer protocol) con extensiones para requisitos específicos de Grid. Incluye características adicionales como control de terceras partes sobre transferencias, transferencia paralela y transferencia parcial de archivos.

- Reliable File Transfer (RFT):

Servicio basado en OGSA que provee interfases para controlar y monitorear transferencia de archivos de terceros usando GridFTP Servers.

- Replica Location (RLS):

Mantiene y provee acceso para mapear información, desde nombres lógicos de ítems de datos a nombres de destino. Estos nombres de destino pueden representar localizaciones físicas de ítems de datos, o una entrada en la RLS que puede mapear a otro nivel de nombres lógicos de ítems de datos.

- OGSA-DAI:

Acceso e integración de datos, permite integrar conjuntos de datos en formatos diferentes.

4.5.4 Open Grid Services Architecture (OGSA)

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 resistente, 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. Este conjunto de especificaciones OGSA completa los estándares XML, WDSL y SOAP con los estándares desarrollados por Globus para tecnologías de redes Grid, utilizados para localizar, planificar y asegurar recursos informáticos.

OGSA cuenta con el apoyo de empresas de diferentes industrias, incluyendo AVAKI, proveedor de soluciones comerciales de software Grid; Entropía, proveedor de informática de redes Grid distribuida basada en PC; Microsoft; y Platform Computing, proveedor de software de informática distribuida. IBM tiene como objetivo la implantación de OGSA como punto clave en su "Proyecto eLiza". El proyecto eLiza es la iniciativa de informática autonómica de IBM para construir un servidor de infraestructura autogestionable, abierto y heterogéneo para el comercio electrónico y la puesta en práctica de Grids comerciales.

OGSA define el esqueleto de lo que debe ser una arquitectura grid, así como el modelo de programación de los servicios grid. Suministra instrucciones de cómo construir un servicio grid, con que componentes y la forma de ensamblarlos[23].

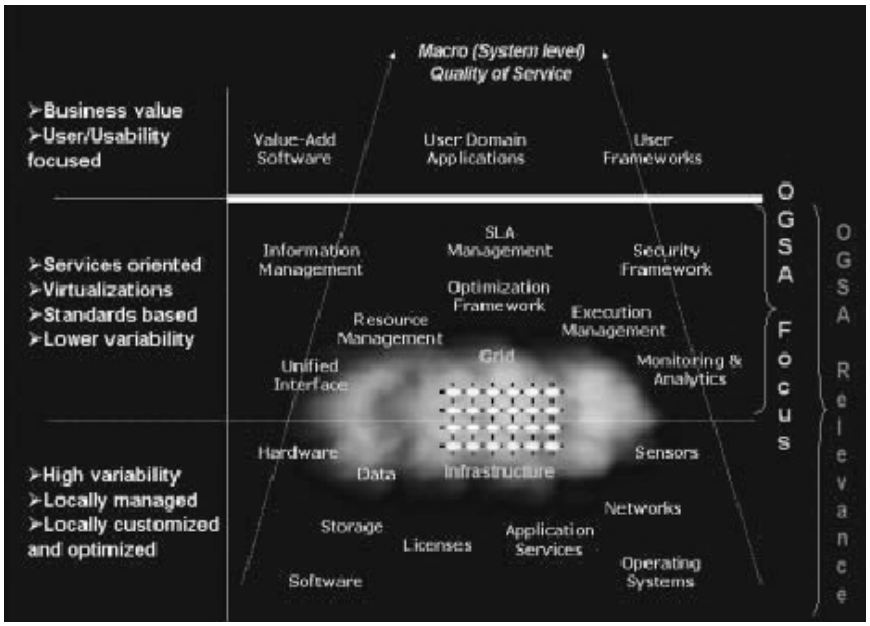

Figura 4. Concepto de la Infraestructura Grid [20]

GGF define OGSA de esta manera: un marco (framework) de amplio espectro de aplicación para la integración de Sistemas Distribuidos. Este framework define un núcleo interfaces, comportamientos, modelos de recursos, enlaces, etc que constituyen la Plataforma OGSA. El grupo de trabajo OGSA GGF define la Plataforma OGSA mediante siguientes pasos:

a) Definir el alcance de los servicios que se requiere para dar un soporte adecuado para las aplicaciones de Grid (e-science, e-business, etc)

b) Identificar un subconjunto básico de tales servicios que sean esenciales para los sistemas Grid y las aplicaciones que se ejecuten sobre ellos, es decir, definir un conjunto base (core) de servicios.

c) Especificar las funcionalidades de altonivel (alto nivel de abstracción) de estos servicios base (core) y sus interrelaciones.

OGSA define:

- Establecimiento de identidad y negociación de autenticaciones
- Definición de normativas

- Descubrimiento, monitorización y gestión de servicios
- Negociación y monitorización de Niveles de Servicio
- Comunicación y Gestión de la Virtualización de miembros de la red Grid.
- Uso jerárquico de servicios Grid

4.6 Middleware para gestion de DataGrid - OGSA-DAI

OGSA-DAI es un middleware que permite acceder recursos de datos tales como bases de datos de XML o relacionales, mediante servicios Web. Un servicio Web de OGSA-DAI permite consultar, actualizar, transformar y entregar datos, de modo que para el usuario se ofrecen servicios de integración de datos.

Los servicios Web de OGSA-DAI se pueden desplegar dentro de un ambiente Grid, convirtiéndose en un medio para usuarios Grid de acceder sus recursos de datos.

Entre los beneficios de OGSA-DAI se encuentran:

- Permite manejar diferentes tipos de recursos de datos, tales como bases de datos relacionales, XML y archivos, para ser usados mediante Web Services.
- Proporciona mecanismos de consulta, actualización, transformación y entrega de datos.
- Permite un acceso consistente a los datos
- Permite la generación de metadata de los datos y de los recursos o fuentes de datos en donde están almacenados estos datos, para accederlos.
- Soporta la integración de datos provenientes de diferentes fuentes de datos.
- Permite la combinación de servicios para proveer servicios web de alto nivel que soporte el procesamiento de consultas distribuidas.

OGSA-DAI puede soportar los siguientes recursos:

- Diferentes tipos de recursos de datos, tales como bases de datos relacionales, XML y archivos.
- Los datos de estos diferentes tipos de datos pueden ser consultados y actualizados, tales como Archivo plano, Streams, Sistemas Administradores de Bases de Datos y Catálogos.

- Los datos pueden ser transformados mediante XSLT y la compresión/descompresión de los datos mediante ZIP y GZIP.
- Permite la entrega de datos mediante los mismos servicios web, URL, ServidoresFTP o GRIDFTP o archivos

Los requerimientos a servicios web de OGSA-DAI tienen un formato uniforme independientemente del recurso de datos expuesto por el servicio (aunque las acciones especificadas dentro de cada requerimiento pueden ser los datos de recurso específicos). El proceso que se realiza en OGSA-DAI para invocación de servicios se ilustra a continuación:

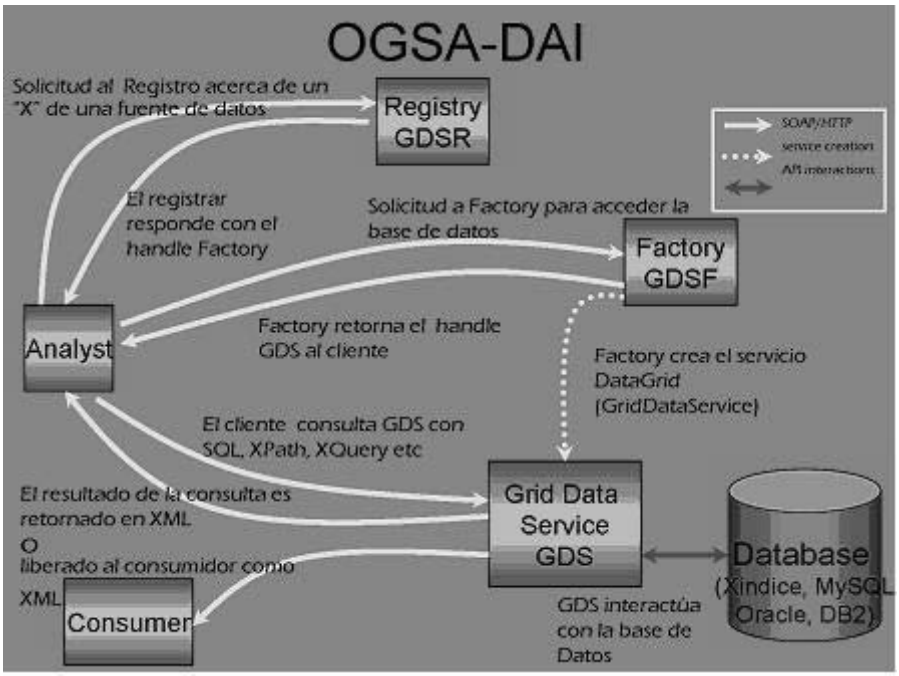

Figura 5. Flujo de invocación de servicios en OGSA-DAI

Proporcione información acerca de los recursos de datos expuestos por un servicio web de OGSA-DAI y la funcionalidad ofrecida por el servicio a clientes. Los servicios de datos expuestos por OGSA-DAI tienen como objetivos:

- Mover los datos a donde se necesitan
- Gestionar las réplicas
- Ejecutar queries y actualizaciones
- Transformación entre distintos formatos

- Gestión de metadata

4.6.1.1 Arquitectura de OGSA-DAI

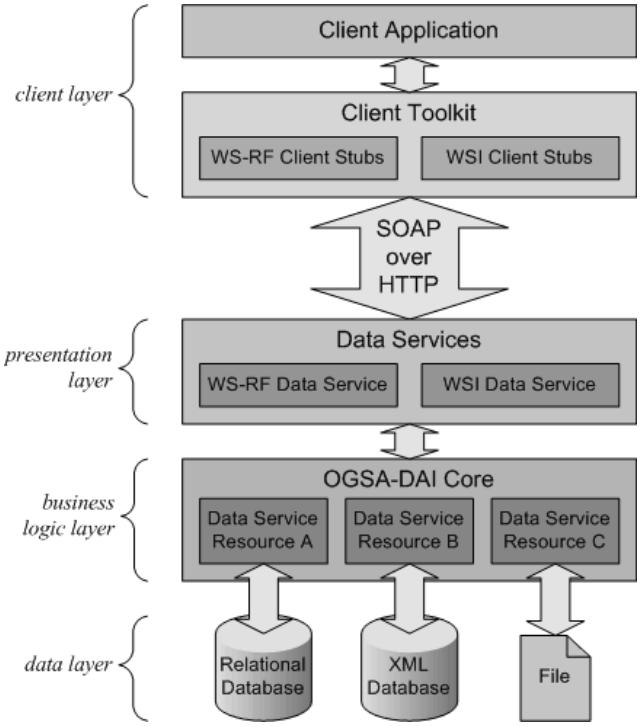

Figura 6. Arquitectura de OGSA-DAI tomado de http://www.ogsa-dai.ac.uk/documentation/ogsadai-wsrf-2.2/doc/background/architecture.html

4.7 Bechmarking existentes similares

En este capítulo se analizará y evaluaran otros trabajos relacionados con la tecnología Grid, los cuales hacen un análisis desde diferentes puntos de vista de modo que permita no sólo enmarcar el aporte que el presente documento proporciona y adicionalmente vislumbra aspectos que este documento no alcanza a abarcar.

El primer documento que se revisará es el desarrollado por UK Engineering Task Force (ETF) el cual evalúa a Globus Toolkit 4 como middleware para el desarrollo de un proyecto futuro llamado UK e-Science Projects. El segundo documento fue desarrollado en la Universidad de Edimburgo habla de un análisis de desempeño realizado a OGSA-DAI y el ultimo documento, es desarrollado por El departamento de Ciencia de Computación de la Universidad de Victoria, la escuela de ciencias de información de salud en donde se presentan un ejemplo de aplicación de la tecnología grid sobre un problema real en el área medica, en donde se realizan evaluaciones a nivel de algunos de los RNF seleccionados en este documento y se llama Can GRID Services Provide Answers to the Challenges of National Health Information Sharing?.

4.7.1 UK Engineering Task Force Globus Toolkit Version 4 Middleware Evaluation[31]

En este documento se evalúa el paquete integral liberado Globus Toolkit versión 4 en adelante GT4, resaltando características como la estabilidad del producto liberado y la mejoría en el desempeño con respecto a las versiones anteriores. Lo que realmente es interesante del documentos es que toma detalles adicionales a los requerimientos no funcionales hace críticas sobre su nivel de desarrollo tales como la documentación y el soporte y su evolución con cada liberación.

El documentos hace un análisis de cómo maneja dos requerimientos no funcionales seguridad y escalabilidad, el aporte que este documentos brinda ahonda en el como mostrando los pasos, pero además, experimenta sobre estos requerimientos no funcionales entre otros para medir el grado de efectividad en la resolución de estos.

4.7.2 Performance Analysis of the OGSA-DAI Software[32]

Evalúa el desempeño de OGSA-DAI mediante la utilización de herramientas y servicios DataGrid. Realiza experimentos netamente transacciones de selección y específicamente de tipos de datos String para establecer el desempeño, a estos experimentos le agrega tipos de seguridad para demostrar que no se deteriora el desempeño.

4.7.3 Can GRID Services Provide Answers to the Challenges of National Health Information Sharing?[33]

Planteamiento de un middleware orientado al área medica, sobre el cual se pretende dar solución a la necesidad de integración de fuente, analizando aspecto de administracion, seguridad y escalabilidad a nivel de recursos de datos.

5 ANALISIS DE LA RESOLUCION DE GLOBUS A REQUERIMIENTOS NO FUNCIONALES

En este capítulo se establecen las pautas para evaluar el aseguramiento por parte de la herramienta Globus Toolkit 4 de los requerimientos no funcionales de un servicio. A partir de esta evaluación, se ofrecen criterios de medición y aceptación de dicha tecnología.

Se hace un planteamiento metodológico que permite hacer de forma científica y controlada la aplicación de experimentos en entornos de experimentación controlados permitiendo medir y evaluar la forma en la que la herramienta da solución a los requerimientos no funcionales establecidos (RNF).

En la primera parte se expone la metodología aplicada para la creación de experimentos que permiten hacer el análisis de esta tecnología. más adelante se revisa cada experimento creado a partir de la metodología y los resultados alcanzados, así mismo los inconvenientes que en la aplicación del estos experimento se presentaron.

5.1 Estrategia metodología

Este planteamiento metodológico establece una serie de etapas que enmarcan los requerimientos no funcionales dentro de un contexto delimitado, este contexto es definido, luego es instalado y sobre este se diseñan los experimentos que se quieren aplicar a los requerimientos no funcionales termina siendo evaluado en términos de medida y criterios de aceptación. Las etapas que componen este planteamiento metodológico son:

6. Entendimiento: se establecen los requerimientos no funcionales que se quieren evaluar, con que propósito y en que ambiente. En esta etapa se hace explícita la tecnología o parte de la tecnología se evalúa, qué se mide o qué características se evalúan y la plataforma en la que se experimenta, es importante en este ultimo punto tener claridad en la versión, dado que existen objetos de estudio que no funcionan de la misma manera de una versión a otra o de un ambiente a otro.

7. Definición de modelo de datos: se hace la de definición del tipo recurso de datos y el modelo de datos que se utiliza para experimentación.

8. Preparación de entornos de evaluación: con el modelo de datos diseñado se crea físicamente y se dispone la infraestructura necesaria para recrear lo diseñado.

9. Diseño de experimentos: definición de cada experimento estableciendo el objetivo que se busca alcanzar, el resultado que espera, los eventos modificadores. Esta etapa arranca con el establecimiento de los eventos, las medidas y los criterios de aceptación que se aplican a cada uno de los requerimientos no funcionales.

10. Ejecución o aplicación de eventos iterativamente: etapa en la que se describe cada uno de los pasos realizado y los resultado presentados frente a cada uno de los eventos provocados.

Cada etapa conlleva a actividades propias sobre los cuales se genera documentación o evidencia de cada una de ellas. El siguiente es un esquema en el que se como las etapas fluye en el planteamiento metodológico de manera secuencia y la necesidad de ejecutarlo en ese orden garantiza el óptimo resultado que se quiere conseguir.

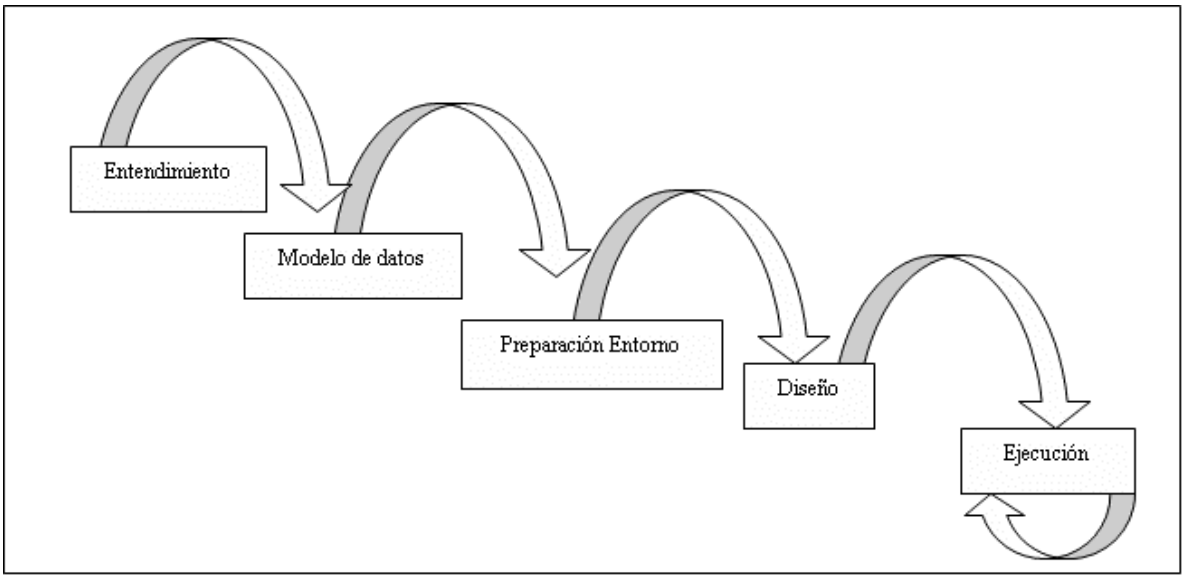

Figura 7 – Planteamiento Metodológico

Cabe anotar que la etapa de ejecución es la que permite ir aplicando eventos propios de cada Requerimiento No Funcional y mediante el resultado o reacción al evento se mide la tolerancia de la tecnología a dichos requerimientos.

5.2 Solución Propuesta

A partir de este punto se revisa, analiza y evalúa la herramienta GT4 bajo es esquema planteado anteriormente.

5.2.1 Etapa de entendimiento

El propósito es experimentar con la herramienta de tecnología Grid, GlobusToolkit 4, con el fin de medir su efectividad frente a los siguientes requerimientos no funcionales:

Transaccionalidad: en este aspecto se revisa que las transacciones sean ejecutadas a nivel de Grid, cumpliendo con la característica ACID, es decir que sea atómica, que la información quede en un estado consistente después de la ejecución de transacciones, así mismo que sean aisladas una de la otra para no atentar contra la característica anteriormente mencionada y finalmente establecer que cualquier cambio perdura en el recurso.

La medida a utilizar en este requerimiento no funcional es el cumplimiento de esta característica y dependiendo del servicio se da su aceptación, es decir, por cada experimento se define el criterio de aceptación conservando siempre la medida.

Seguridad: en cuanto a seguridad sólo se revisan los aspectos de autenticación y autorización, las medidas aplicadas son los mecanismos de comunicación segura que estos dos aspectos involucran y la autorización de uso en ambientes homogéneos pero ubicaciones diferentes. El criterio de aceptación es que se debe dar solución a ambos aspectos.

Administrabilidad: Se revisa las facilidades que presente la herramienta para administrar aspectos como la seguridad, la integración o desvinculación de recursos, la integración o desvinculación de nodos. En este requerimiento se mide con el costo operativo que implica administrar un grid. El criterio de aceptación es si existe la forma o no de administrar los otros requerimientos no funcionales.

Escalabilidad: Se analiza desde dos puntos de vista, el primero es la facultad de agregar y/o retirar recursos dado que las fuentes pueden ser dentro de una misma máquina o diferentes máquinas. El segundo punto es la vinculación de nuevos usuarios que se pueda realmente enlazar y hacer uso del grid entre diferentes máquinas. La medida en ambos casos es la cantidad de recursos y de usuario respectivamente, los criterios de aceptación son los mecanismos que permiten la escalabilidad.

Desempeño: Este último aspecto, establece el tiempo de respuesta que un servicio tiene frente a la solicitud de un cliente y los criterios de aceptación dependen del servicio y del ambiente del servicio.

Cada uno de esos requerimientos, usualmente se encuentran resueltos en las tecnologías existentes en el manejo de información, tanto a nivel de bases de datos, como en tecnología de

componentes distribuidos. El objetivo es evidenciar como la tecnología Grid los abarca en determinados entornos, dichos entornos son:

A. Entorno donde se es propietario de los recursos, es decir, se tiene propiedad y control sobre los equipos, fuentes de datos y servicios.

B. Entorno donde se podrían crear y compartir recursos o aplicaciones de tipo general que pudiesen ser utilizadas por distintas organizaciones virtuales existentes en el grid.

C. Entorno donde no se tienen control sobre los recursos.

Los entornos citas en los numerales A y B se dividen de acuerdo al número de recursos involucrados en cada experimento, es decir, si se tiene una fuente de datos y dos clientes o viceversa, o si sólo es una fuente y un cliente o el cliente y la fuente esta en el mismo equipo.

Entorno de evaluación / RNF	Propietario de los Recursos		Sin recursos	Recursos Compartidos	
	Fuente y Cliente Iguales	Una fuente y un cliente independientes	Una fuente y un cliente independientes	Fuente y Cliente Iguales	Una fuente y un cliente independientes
Transaccionalidad	X	X	X	X	
Seguridad	X	X	X	X	
Desempeño		X		X	
Administrabilidad	X	X	X	X	
Escalabilidad		X		X	

Tabla 1. Requerimiento no funcionales vs. Entornos de evaluación

5.2.1.1 Entorno de Realización

La plataforma o sistema operativo sobre el cual se instalo es Linux, en la distribución Ubuntu. La versión de Globus Toolkit es 4.0 y Postgres 8.0.3.

Figura 8 – Planteamiento Metodológico

5.2.2 Etapa de definición de modelo de datos

El recurso de datos definido es una base de datos en Postgres 8.0.3, los scripts de creación se relacionan en el anexo 1 anexos: Creación de tablas en Postgres y el modelo es el siguiente:

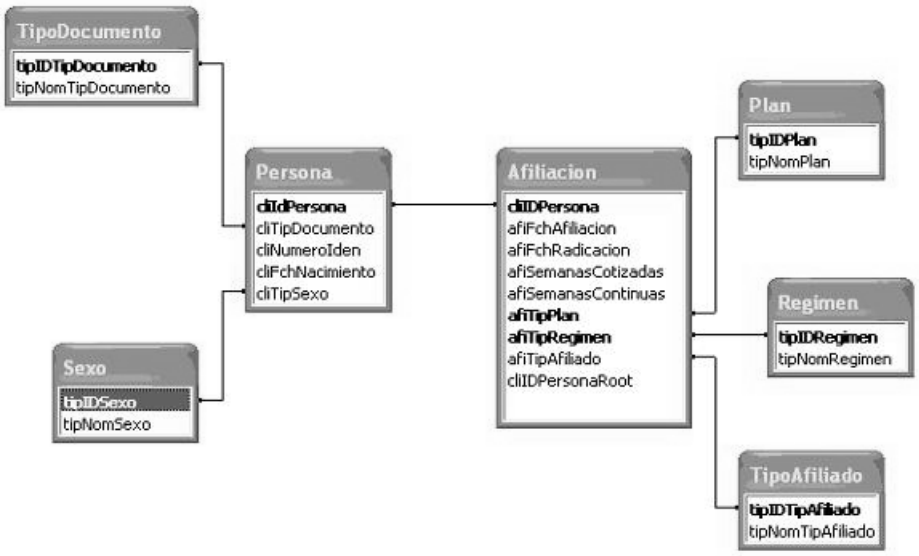

Figura 9 – Modelo de datos

El modelo se basa en un persona que tiene tipo de documentos y sexo, estas dosultimastambién son tablas que están asociadas por foreing key a persona y una persona puede tener una o muchas afiliaciones con planes diferentes en régimenes diferentes y como un tipo de afiliado diferente, este modelo se tomo porque era en su momento una necesidad de negocio conocida.

5.2.3 Etapa de preparación de entornos de evaluación

Inicialmente, la infraestructura sobre la cual se creo todo el ambiente es el laboratorio de redes, con dos máquinas se realizo el proceso de instalación y preparación correspondiente a esta etapa.

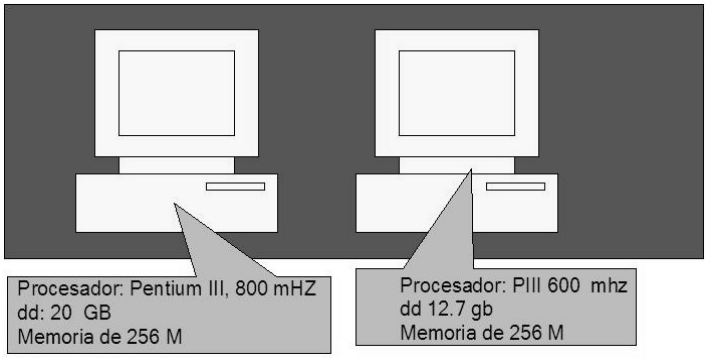

Figura 10 – Descripción de maquinas del laboratorio

En esta etapa es primordial seguir con gran cuidado los pasos de instalación que se relaciona en la página de globus: http://www.globus.org/toolkit/docs/4.0/index.html, en el link Installing GT 4.0 (System Administrator's Guide) porque cualquier cambio o alteración tiene impacto en la usabilidad de Globus.

El tamaño de la base de datos Postgres es distribuidas en la siguiente cantidad de registros:

- Tipo de Documento: 2 registros
- Sexo: 2 registros
- PLAN: 2 registros
- Regimen: 2 registros
- TipAfiliado: 2 registros
- Persona: 30000 registros

A continuación se explica la descarga e instalación de la plataforma:

5.2.3.1 Descarga

1. Java: j2sdk-1_4_2_08-linux-i586-rpm de Sun
2. Ant : apache-ant-1.6.5-bin.tar
3. Bison: bison_1.35-7_i386.deb

```
4. zlib : zlib-1.2.3.tar
5. sudo: sudo-1.6.8p9.tar
6. Globus Toolkit: globus-4.0.1
7. PostgreSQL : postgresql-base-8.0.3.tar
```

Previo a la instalación se debe asegurar que se cuenta con los prerrequisitos y la distribución de globus y postgres que se necesita. En el capítulo 3 de la documentación de globusse detallan los prerrequisitos (http://www.globus.org/toolkit/docs/4.0/admin/docbook/ch03.html), en la página existe sobre cada uno de estos prerrequisitos los link a cada uno de los sitios de donde se puede descargar, el orden en que se numeran es el orden en el que se instalan, salvo postgres que es aconsejable como se menciona durante la ejecución de los pasos de instalación.

5.2.3.2 Instalación de la plataforma

```
1. $ adduser Globus
2. $ tar xfj gt4.0.0-all-source-installer.tar.bz2
3. $ export PATH=/grid/JAVA/1.5.0_01/bin:/grid/ANT/1.6.2/bin:$PATH
4. $ export GLOBUS_LOCATION=/grid/GLOBUS/4.0
5. $ mkdir $GLOBUS_LOCATION
6. $ chown Globus:grid $GLOBUS_LOCATION
7. $ cd gt4.0.0-all-source-installer
8. $ ./configure --prefix=$GLOBUS_LOCATION
9. $ make
10. $ make install
11. $ su postgres
12. $ createuser Globus
13. $ createdb  Globus rftDatabase
14. $ psql -d rftDatabase \ -f
    $GLOBUS_LOCATION/share/Globus_wsrf_rft/rft_schema.sql
15. $ vi /usr/local/pgsql/bin/ pg_hba.conf
16. $ postmaster –i /usr/local/pgsql/data
17. $ vi $GLOBUS_LOCATION/etc/Globus_wsrf_rft/jndi-config.xml
```

1. Crear el usuario con el que realiza la instalación de GT4 y a este se le dan permisos sobre los archivos de GT4.

2. Descomprimir los fuentes de gt4

3. Añadir al PATH los programas o aplicaciones mencionados anteriormente, java y ant.

4. Añadir la variable de entorno GLOBUS_LOCATION para que todo sea más fácilmente comprensible

5. Crear el directorio donde se requiere instalarlo y se le dio a dicho directorio los permisos necesarios para el usuario de Globus. En el script esta consistido por los comandos 5 y 6.

7. Localiza la carpeta de globus con la variable de forma que se comprueba la validez de esta previo a la instalación.

8. Instalar GT4 (este proceso depende de la potencia de la máquina donde se realice la instalación y puede llegar a durar más de 1 hora). En el script esta consistido por los comandos 8 al 10.

11. Configurar de Reliable File Transfer (RFT), asegurarse inicialmente de tener instalado PostgreSQL, dado que se presenta el error mencionado anteriormente, enseguida se crea un usuario con permisos para que pueda acceder a una base de datos que de acuerdo con las instrucciones es llamada rftDatabase.

12 Bajo el usuario postgres (administrador de la base de datos) crear el usuario globus en postgres con privilegio para crear bases de datos. En el script esta consistido por loscomandos12 y 13.

14. Crear las tablas necesarias en la base de datos con el usuario Globus

15 Modificar el archivo pg_hba.conf que por defecto se encuentra en `/usr/local/pgsql` /bin y se añade la siguiente línea:

host rftDatabase "usuario_Globus" "host_ip" 255.255.255.255 md5

16 Reiniciar potgresql para que los cambios tengan efecto.

17 Modificar el archivo $GLOBUS_LOCATION/etc/Globus_wsrf_rft/jndi-config.xml y se sustituye en la sección connectionString el username y password por el que se puso.

Recomendaciones

En el punto 15, hay que tener en cuenta que la línea que sugiere el manual se cambio un poco dado que la línea se tuvo que dejar es de la siguiente forma:

```
host rftDatabase " Globus" "R11.uniandes.edu.co"  trust
```

En el punto 17 también hay que tener en cuenta que la cadena de conexión debe llevar el nombre de la máquina y no la IP de lo contrario no lo encuentra la base de datos y la consecuencia es que el contenedor no sube.

Es recomendable hacer la instalación de Postgres al momento de instalar globus de acuerdo con los pasos y no al momento de instalarse el sistema operativo dado que por seguir la instalación se puede hacer doble instalación lo cual genera errores de IODBC, como se muestra en la figura siguiente:

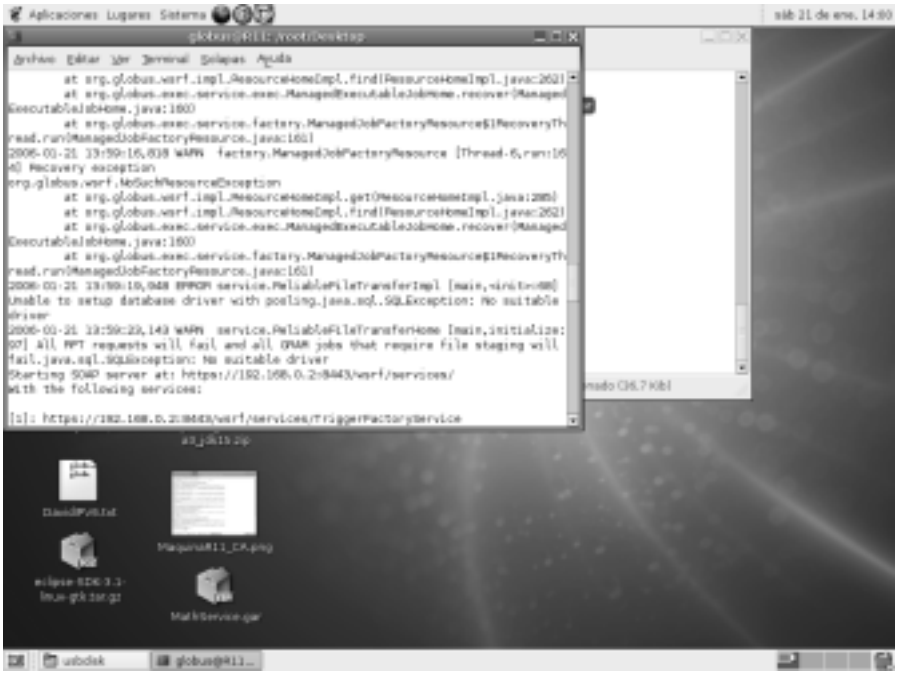

Figura 11 – Muestra de error en el ambiente: No reconoce iodbc

La solución es en primera instancia, validar que no exista una instalación previa de postgres revisando en la variable de entorno POSTGRES_HOME si esta la instalación y de no existir esta variable igualmente buscar si esta instalado, parece un paso obvio pero por su misma simplicidad se olvida.

En segunda instancia si ya se instalo una segunda versión de postgre, la solución es eliminar una estas instalaciones, asignar a la variable en la variable de entorno POSTGRES_HOME la ruta de la instalación que queda y continuar la instalación de Globus en el punto en el que lo sugiere el manual de Globus, es decir en el momento en el que se configura el RFT.

5.2.3.3 Instalación de la base de datos

```
1. $ createdb  DataGrid
2. $ psql –l
```

```
3.  $ psql  DataGrid  -f /users/local/Scritp/script.sql
4.  $ psql  DataGrid  -f /users/local/Scritp/script1.sql
5.  $ psql  DataGrid  -f /users/local/Scritp/script2.sql
6.  $ psql –l
7.  $ psql  DataGrid  -f /users/local/Scritp/insertar.sql
8.  vi /usr/local/pgsql/bin/pg_hba.conf
```

En este momento ya se tienen bases creadas propias del ambiente Globus, ahora se hace el mismo proceso pero para el recurso de datos que se va a compartir en el ambiente Grid, con los siguientes pasos:

1. Crear bajo el usuario Globus, la base de datos.

2. Visualice el catalogo de base de datos para validar que si fue creada, como se muestra en la figura:

Figura 12 – Creación de Base

3. Crear las tablas necesarias en la base de datos con el usuario Globus mediante el siguiente comando y a partir de los script previamente creados y relacionados en el anexo 1. Este script esta conformado por los comando del 2 al 5.

6. Validar la creación de tablas sobre la base de datos.

Figura 13 – Listado de tablas

7. Cargar o insertar los datos sobre las tablas, mediante el script de insertar, relacionado en el anexo 2

Figura 14 –Tablas con datos

8. Modificar el archivo pg_hba.conf que por defecto se encuentra en /usr/local/pgsql /bin y se añade la siguiente línea:

```
host DataGrid " Globus" "R11.uniandes.edu.co"  trust
```

Recomendaciones

Para que el resultado del primer paso del script sea CREATE DATABASE, el usuario Globus debe ser creado, esto se asegura dado que en la parte de configuración de RTF se le dieron los permisos de creación de base de datos

Una vez lista la creación de la base de datos, en la documentación se realiza la instalación de iodbc, sin embargo, Ubuntu cuenta con un iodbc propio, impidiendo la instalación del recomendado por Globus, por lo tanto este paso sugerido en la documentación de Globus debe ser obviado para la plataforma seleccionada.

5.2.3.4 Instalación de la certificadora

Se instala entidad certificadora en donde los servidores y usuarios adquieren su certificados para poder ingresar, en este caso se instalo SimpleCA que provee Globus, siguiendo los siguientes pasos:

```
1. $GLOBUS_LOCATION/setup/globus/setup-simple-ca
2. $GLOBUS_LOCATION/setup/globus_simple_ca_790b429b_setup/setup-gsi –default
3. $GLOBUS_LOCATION/bin/grid-cert-request -host R11.uniande.edu.co
4. $GLOBUS_LOCATION/setup/globus/grid-ca-sign -in /etc/grid-
   security/hostcert_request.pem -out hostsigned.pem
5. $ cp /home/globus/hostsigned.pem /etc/grid-security/
6. $ mv /etc/grid-security/hostsigned.pem /etc/grid-security/hostcert.pem
```

1. Instala la entidad certificadora nativa de Globus, ubicado en la carpeta setup/Globus dentro del path registrado en el $GLOBUS_LOCATION, bajo el usuario Globus

2. Se ejecuta el setup que genero el comando anterior dentro de la carpeta con el hash aplicado a ese servidor, para que se creen las carpetas que manejan el esquema de seguridad, /etc/grid-security, /etc/grid-security/certificates y ejecute las politicas de seguridad. Dentro de estas carpetas se crean automáticamente:

/etc/grid-security

A. hostcert.pem: certificado usado por el servidor en autenticación mutua.

B. hostkey.pem: clave privada correspondiente al certificado del servidor(read-only by root)

C. grid-mapfile: mapea nombres de usuario de Grid a cuentas de usuarios locales

/etc/grid-security/certificates

A. CA certificates: certs en los que se confía cuando se valida certs, por lo que no deben ser verificados

B. ca-signing-policy.conf: define los nombres de usuarios que pueden ser firmadospor cada AC

3 Solicitar un certificado para el equipo 'host', enviando el nombre del equipo, para que se generen los archivos hostcert.pem , hostkey.pem y hostcert_request.pem.

4. Firmar los certificados para que sena validos como usuario globus.

5. Copiar el 'hostsigned.pem' firmado a la carpeta /etc/grid-security y renombrarlo a hostcert.pem reemplazando el archivo existente.

Recomendaciones

Si ya hay una entidad certificadora diferente a la simpleza, es recomendable usarla ya que la SimpleCA no ofrece garantías de seguridad.

5.2.3.5 Certificados para los usuarios

Este proceso se realiza cada vez que se ingresa un usuario al grid.

```
1. $grid-cert-request
2. $grid-ca-sign -in /tmp/usercert_request.pem -out /tmp/signed.pem
3. $cp /tmp/signed.pem ~/.globus/
4. $mv ~/.globus/signed.pem ~/.globus/usercert.pem
```

1 Solicita el certificado de host con el usuario que se registra no globus ni root

2 Firmar el certificado para que sea valido pero no en la ubicación ~/.globus/ y cambie el nombre del archivo de salida a signed.pem como usuario Globus

3 Envíe el certificado a la ruta del cliente ~/.globus/ con el nombre usercert.pem como el usuario

Archivos importantes en este proceso son:

$HOME/.globus

A. usercert.pem: Certificado de usuario

B. userkey.pem: Clave privada de usuario

5.2.3.6 Publicación de servicios

```
1. $ /GLOBUS_LOCATION/bin/globus-deploy-gar
   /home/globus/HelloWorldServices.gar
2. $ /GLOBUS_LOCATION/bin/Globus-start-container
```

1 Se hace deploy o publicación de un servicio en el contenedor.

2 Subir el contenedor

5.2.4 Etapa de diseño de experimentos

A continuación se expondrán los experimentos básicos que se aplican sobre la tecnología grid en un entorno propio sobre cada RNF.

5.2.5 Transaccionalidad

5.2.5.1 Experimento1: Propiedad ACID de la transacción en réplicas

Objetivo: Validar la atomicidad, consistencia, durabilidad y aislamiento del DataGrid.

Detalle: En un ambiente replicado, es decir, un ambiente compuesto por lo menos con dos recursos de datos, uno replica del otro, se realiza un invocación a un servicio de datos en el que se efectua alteración en los datos (inserción, eliminación o actualización) desde un cliente autenticado.

Precondición: Un servicios de datos que modifiquen datos están disponibles en el contenedor y existe autorización sobre los recursos de datos

Criterio de aceptación: El DataGrid mantiene las propiedades de ACID entre las réplicas existentes cuando se aplican transacciones de modificación sobre los datos.

Medición: La transacción debe cumplir el ACID.

5.2.5.2 Experimento 2: Propiedad ACID con concurrencia en réplicas

Objetivo: Validar en un ambiente concurrente que la atomicidad, consistencia, durabilidad y aislamiento en el DataGrid se mantiene.

Detalle: En un ambiente replicado se realiza una invocación de múltiples servicio de datos en el que se efectúan operaciones diferentes sobre la base de datos (inserción, eliminación, consulta y

actualización), no es relevante la cantidad de clientes sino la cantidad de servicios que recurren a los recursos de datos.

Precondición: Los servicios de datos con las cuatro operaciones están disponibles en el contenedor para interactuar con los recursos de datos y autorización sobre los mismos.

Criterio de aceptación: El DataGrid mantiene las propiedades de ACID entre las réplicas existentes.

Medición: Número de procesos concurrentes vs número de réplicas.

5.2.6 Seguridad

5.2.6.1 Experimento 3: Autenticación del usuario vs Usuarios creados en la plataforma

Objetivo: Validar que la autenticación del grid esta acorde con la autenticación a la plataforma.

Detalle : En el ambiente grid esta el sistema operativo Ubuntu en donde se crea el usuario que va hacer uso del grid, este usuario se certifica con la entidad certificadora del grid, pero si el usuario no esta creado en la plataforma no debe permitirse su ingreso a los recursos de grid.

Precondición: El usuario esta certificado pero no creado en la plataforma.

Criterio de aceptación: El DataGrid mantiene la coherencia de seguridad con la plataforma.

Medición: Autenticación del usuario en el grid.

5.2.6.2 Experimento 4: Autenticación del usuario vs Recursos de datos

Objetivo: Establecer si es necesaria la creación de usuario en dada uno de los recursos de datos en los que un usuario del grid desea ingresar o se aplica la autenticación única.

Detalle: En la preparación del entorno esta el script que permite hacer la certificación de los usuarios que ingresan al grid, proceso que debe realizarse siempre que un nuevo usuario ingresa, lo que se busca demostrar o evidenciar es que con este certificado el usuario tiene permisos sobre cualquier recurso y no es necesaria su creación en cada uno de estos.

Precondición: En el recurso de datos no se ha creado el usuario y no se le ha dado permiso sobre el recurso.

Criterio de aceptación: El usuario recibe respuesta de los servicios de datos de los diferentes recursos de DataGrid.

Medición: Autenticación del usuario en los recursos de datos.

5.2.6.3 Experimento 5: Autorización del usuario sobre los recursos de datos con la certificación

Objetivo: Establecer si es necesaria la autorización en cada uno de los recursos de datos a los que un usuario del grid desea ingresar.

Detalle: En este experimento se busca dimensionar el alcance del certificado en la medida en el que con este puede llegar hasta la información, pese a que cada recurso no lo tenga especificado localmente.

Precondición: El usuario no existe en los recursos de datos ni tiene permisos sobre ellos

Criterio de aceptación: El usuario certificado no tiene autorización sobre los recursos de datos del DataGrid.

Medición: Nivel de Autorización del usuario en el DataGrid.

5.2.6.4 Experimento 6: Autorización del usuario certificado sobre los servicios de datos

Objetivo: Establecer si es necesaria la autorización en los servicios de datos si se le da autorización sobre los recursos.

Detalle: En este experimento se busca dimensionar el alcance del certificado en la medida en que el contenedor se va restringiendo de acuerdo a su nivel de autorización, pese a que con la simple expedición del certificación no se determino dicho nivel y su alcance con los recursos de datos

Precondición: El usuario no existe en los servicios de datos

Criterio de aceptación: El usuario certificado no tiene autorización sobre los recursos de datos del DataGrid.

Medición: Nivel de Autorización del usuario en el DataGrid.

5.2.6.5 Experimento 7: Disponibilidad y Tolerancia a Fallos

Objetivo: Validar en un ambiente replicado de varios tipos de recursos, si al momento de no encontrar disponible un recurso de datos reconoce la replica correspondiente.

Detalle: En este experimento se busca ver la disponibilidad que maneja el ambiente data grid si sus recursos de datos pierden capacidad de procesamiento por carga de trabajo. Este experimento va

ligado con el manejo de carga de trabajo que maneja y el esquema de asignación de trabajo que los ambientes grid tienen innato.

Precondición: Ambiente DataGrid

Criterio de aceptación: El DataGrid sigue proporcionando los servicios.

Medición: Asignación de carga por recurso disponible y tiempo de disponibilidad.

5.2.7 Administrabilidad

5.2.7.1 Experimento 8: Administración de usuario y sincronización de usuario en el DataGrid

Objetivo: Establecer si el DataGrid ofrece mecanismo para hacer administración de usuario.

Detalle: En el script de certificación de usuario se realizan operaciones sobre el usuario creado en la plataforma, pero qué tan costosa es la certificación si para cada usuario que se vincula se requiere realizar este proceso y la sincronización, si se realiza una actualización en el usuario tal como eliminarlo.

Precondición: El usuario no existe en el DataGrid

Criterio de aceptación: El usuario es autenticado y autorizado en el DataGrid y los mecanismos de administración de usuario.

Medición: Costo de administración de usuario en el DataGrid.

5.2.7.2 Experimento 9: Administración de servicios de datos

Objetivo: Establecer si el DataGrid ofrece mecanismo para hacer administración de servicios de datos.

Detalle: En el script de certificación de usuario se realizan operaciones sobre el usuario creado en la plataforma, pero qué tan costosa es la certificación si para cada usuario que se vincula se requiere realizar este proceso y la sincronización, si se realiza una actualización en el usuario.

Precondición: Sólo existen los servicios propios de globus

Criterio de aceptación: Los mecanismos de administración de servicios ofrecidos en el DataGrid.

Medición: Costo de administración de servicios de datos en el DataGrid.

5.2.7.3 Experimento 10: Administración de recursos de datos

Objetivo: Establecer si el DataGrid maneja mecanismo de inclusión de recursos de datos sin importar su naturaleza (bases de datos y archivos indistintamente).

Detalle: Establecer si tiene restricciones el DataGrid en el tipo de recursos pese a que el concepto de grid del cual proviene dice que no hay limitantes en este sentido y revisar el mecanismo que emplea el DataGrid para hacer esta administración.

Precondición: No hay sino la base del contenedor en el DataGrid

Criterio de aceptación: Los mecanismos de administración de recursos ofrecidos en el DataGrid.

Medición: Costo de administración de servicios de datos en el DataGrid.

5.2.8 Escalabilidad

5.2.8.1 Experimento 11: Incremento de Recursos de datos

Objetivo: Establecer el limite de inclusión de recursos de datos

Detalle: Establecer inicialmente si es posible agregar recursos y, de ser posible, explorar si tiene límite o restricciones el DataGrid en la cantidad de recursos pese a que el concepto de grid no presenta limitaciones ni en cantidad ni en número, dado que la finalidad del concepto grid es la vinculación de todo tipo de recursos que aporte servicio a cualquier capa del grid. Adicionalmente, revisar el mecanismo que emplea el data grid para dar solución a esto.

Precondición: No hay sino la base del contenedor en el DataGrid. Adicionalmente, el experimento se realiza sobre recursos ubicados en una misma red.

Criterio de aceptación: Los mecanismos de administración de recursos ofrecidos en el DataGrid.

Medición: Número de recursos adicionados y Costo de administración de recursos de datos en el DataGrid.

5.2.8.2 Experimento 12: Incremento de Servicios de datos

Objetivo: Establecer el mecanismo de inclusión de por los menos 4 servicios nuevos de acceso a recursos de datos.

Detalle: Establecer si es posible agregar servicios y ,si es posible, extenderse a si tiene límite en la cantidad de servicios que es capaz de ofrecer y si tiene policitas que permitan estructurar el alcance de los servicios dentro del DataGrid.

Precondición: Sólo existen los servicios de globus

Criterio de aceptación: Los mecanismos de administración de recursos ofrecidos en el DataGrid.

Medición: Número de servicios adicionados y el Costo de administración de servicios de datosen el DataGrid.

5.2.8.3 Experimento 13: Incrementar de usuarios

Objetivo: Asignar credenciales a diferentes usuarios para establecer si hay límite de usuarioso de emisión de certificaciones

Detalle: Este experimento es derivado al de seguridad, al tener el mecanismo de certificación lo que resta en este punto es establecer si tiene límite en la cantidad de usuario y si tiene metodología o procedimiento que permitan estructurar el manejo de usuarios dentro del DataGrid.

Precondición: Sólo existen el usuario globus y postgres que son propios de la instalación y configuración de GT4.

Criterio de aceptación: Los mecanismos de administración de usuario ofrecidos más que el mecanismo de creación de usuarios.

Medición: Número de usuarios adicionados y el Costo de administración de usuarios en el DataGrid.

5.2.9 Desempeño

5.2.9.1 Experimento 14: Invocación de servicios de datos

Objetivo: Medir el tiempo de respuesta de los servicios de datos desde que son invocados hasta que se recibe la respuesta.

Detalle: Establecer el tiempo de respuesta y los saltos entre servicios que implica hacer una solicitud de datos grande volúmenes de datos, esta solicitud no es que en pantalla queden todos los datos sino en un repositorio.

Precondición: Recursos de datos con grande volúmenes de datos.

Criterio de aceptación: Dependen del servicio de datos, si es búsqueda en grandes volúmeneso devoluciones de grandes volúmenes de datos.

Medición: Tiempo de respuesta en el DataGrid.

5.2.9.2 Experimento 15: Desempeño de los servicios de índices

Objetivo: Medir el tiempo de respuesta de los servicios de índices desde que son invocadoshasta que resuelven la solicitud.

Detalle: Establecer el tiempo de respuesta, los saltos entre servicios y el *delay* que el servicio de índices aplica al recorrer los diferentes recursos.

Precondición: Recursos de datos con grande volúmenes de datos

Criterio de aceptación: Dependen del servicio de datos, si es búsqueda en grandes volúmeneso devoluciones de grandes volúmenes de datos.

Medición: Tiempo de respuesta en el DataGrid

5.3 Ejecución de experimentos

Los servicios o recursos que se encontraron como los que dan respuesta a los RNF seleccionados son:

RNF	Servicio o Recurso
Tansaccionalidad	OGSA-DAI
Seguridad	GSI
Administrabilidad	Todos los servicios
Escalabilidad	RLS GSI Hierarchy.xml
Desempeño	RLS

Tabla 2. RNF vs Servicios y Recursos

Cada uno de estos servicios proporciona funciones que son finalmente las que me permiten medir si realmente para lo que esta destinado cada servicios, se esta haciendo o se esta efectivamente suministrando y de qué manera. Sin embargo y pese a tener identificado que serviciosson losque

hay que validar con los experimentos, a continuación se relacionan los experimentos y su estado de realización.

Tansaccionalidad	Ejecución
Propiedad ACID de la transacción en réplicas	No
Propiedad ACID con concurrencia en réplicas	No
Seguridad	
Autenticación del usuario vs Usuarios creados en la plataforma	Si
Autenticación del usuario vs Recursos de datos	Si
Autorización del usuario sobre los recursos de datos con la certificación	Si
Autorización del usuario certificado sobre los servicios de datos	Si
Disponibilidad y Tolerancia a Fallos	No
Administrabilidad	
Administración de usuario y sincronización de usuario en el datagrid	Si
Administración de servicios de datos	Si
Administración de recursos de datos	Si
Escalabilidad	
Incremento de Recursos de datos	No
Incremento de Servicios de datos	No
Incremento de usuarios	No
Desempeño	
Invocación de servicios de datos	Si
Desempeño de los servicios de índices	No

Tabla 3. Experimentos vs Estado de ejecución

El mecanismo de experimentación se relaciona a continuación, claro esta partiendo de la premisa que ya se ejecutaron los pasos relacionado en el script de instalación de GlobusToolkit 4 y ya esta instalada la base de datos

5.3.1 Seguridad – Autenticación

El primer experimento consiste en validar que no se maneja seguridad independiente del manejado en el sistema operativo Ubuntu, inicialmente se creo el usuario

```
1. $./adduser Lennis
2. crear credencial para usuarios. Ver punto 5.2.3.5
3. $./grid-proxy-init
4. $./globus-personal-gatekeeper – start
5. $./globusrun – o –r
   “R11.uniandes.edu.co:35659:/o=Grid/OU=GlobusTest/Ou=simpleCA-
   r11.uniandes.edu.co/OU=uniandes.edu.co/CU=Lennis Torres Clavijo”
   ‘&(executable=/bin/date)’
```

Después de esto el resulta obtenido se observa en la figura 15 del anexo 4, GRAM Job submmission failed becuase the gatekeeper contact cannot be parsed (error code 96), que eslo esperado, que no se puede ejecutar un servicio porque esta denegado el servicio.

En seguida, se agrega el usuario por la administracion de Usuarios de Ubuntu (figura 16 del anexo 4), este proceso no se puede manejar por scritp ya que es un proceso operativo sobre el sistema operativo.

Luego se realiza la ejecución de los pasos o el script anterior y el resultado es el observado en la figura 17, esto nos da como resulta que efectivamente la seguridad es coherente con la manejada por el sistema operativo.

Continuando con el proceso los experimentos validando si es necesario crear el usuario con permisos sobre los recursos para esto, se ejecuta el comando de inicializar la base de datos, al no tener permisos se crea el usuario por los servicios de postgres y se da permiso al usuario y con esto se valida que es coherente con la seguridad de los recursos.

5.3.2 Seguridad – Autorización

Básicamente para el ingreso a cada uno de los pasos anteriores, hayí uno que no relaciono y es la creación de la línea que corresponde al certificado del usuario en el gripmapfile, si el usuario no es

creado en este punto, simplemente no tiene acceso al recurso, no lo relaciono en la parte de autenticación porque es netamente el error 7 de permiso denegado y aquí el manejo del script depende del certificado que se adiciona, en el caso del experimento es:

```
"R11.uniandes.edu.co:35659:/o=Grid/OU=GlobusTest/Ou=simpleCA-
r11.uniandes.edu.co/OU=uniandes.edu.co/CU=Lennis Torres Clavijo" lennis
```

Cabe anotar que el uso de las comillas es importante para el archivo, ya que delimita desde donde hasta donde va el certificado.

5.3.3 Escalabilidad

Escalabilidad busca adicionar recursos, usuarios y servicios. Para este punto se tiene dos equipos disponibles, sin embargo el entorno ideal de ejecución es una red de más de 10 equipos en donde se puede ver la interacción por grid de estos equipos. El proceso aplicado a los tres experimentos es adicionar recursos, usuarios y servicios, dependiendo del caso, de manera incrementar, es decir iniciar con dos recursos, dos usuarios y dos servicios, validar el desempeño (tiempo de respuesta) e ir incrementando a cuatro, ocho, dieciséis, treinta y dos y el total de equipos disponibles, de modo que con cada incremento se vaya revisando si efectivamente se va incorporando y el tiempo de respuesta no se degrada.

Los recursos son adicionado tanto al servicio de rls como al hierarchy.xml para manejar al jerarquía entre ellos en <upstream> o en <downstream> de acuerdo a lo que corresponda.

```
<config>
<!-- <upstream> elements specify remote index services that the local index
    will be registered to.  Set an upstream entry for each VO index that you wish
to participate in.
-->
<!--
<upstream>https://R11.uniandes.edu.co:8443/wsrf/services/DefaultIndexService</ups
tream> -->
<!-- <downstream> elements specify remote index service which will be registered
into the local index. You do not need to configure <downstream> services unless
you are building your own VO. -->
<!--
<downstream>https://R11.uniandes.edu.co:8443/wsrf/services/DefaultIndexService</d
ownstream> -->
</config>
```

Para analizar escalabilidad de servicios es preciso valerse de los comando de *Deploy* como se muestra en el anexo 6, entonces siguiendo el esquema, se agregan dos servicios con *Deploy*,

luego se realiza el llamado a cada servicio desde la misma maquina pero en otra terminary desde el otro equipo.

```
1. $java –classpath ..build/stubs/classes/: $CLASSPATH
   org/globus/datgridHelloWordlServices/cliente/ClienteHello
   http:///192.168.0.2:8080/wrsf/services/datagrid.HelloWorldServices/HelloWor
   dlsServicesFactoryService Lennis
```

El script corresponde al servicio sobre el cual se probó pero esta cambia de nombre (http:///192.168.0.2:8080/wrsf/services/datagrid.HelloWorldServices/HelloWordlsServicesFactoryService) dependiendo del nombre del servicio que se sube y a partir de este punto, de acuerdo a los parámetros del servicio invocado. Por cada servicio invocado se debe crear un cliente que haga el llamado adecuado.

Después de incrementar servicios, se incrementan recurso en la misma cantidad y se vuelve hacer el llamado con el script de cliente correspondiente. A medida que se incrementa lo ideal es poder hacerlo simultaneo de modo que se pueda ver como se resuelve la concurrencia.

5.3.4 Administrabilidad

Básicamente en este punto se limita a la manipulación de los diferentes archivos ya nombrados, gripmapfile y hierarchy.xml a medida que se incorpora un usuario o un recurso respectivamente.

En la instalación de la base de datos rft de Globus, la modificación del archivo jndi-config.xml, en donde se define la cadena de conexión y el usuario que se conecta. La modificación que se hace es en el siguiente tag:

```
1.  - <parameter>
2.    <name>driverName</name>
3.    <value>org.postgresql.Driver</value>
4.    </parameter>
5.  - <parameter>
6.    <name>connectionString</name>
7.    <value>jdbc:postgresql://R11.uniandes.edu.co/rftDatabase</value>
8.    </parameter>
9.  - <parameter>
10.   <name>userName</name>
11.   <value>globus</value>
12.   </parameter>
13. - <parameter>
14.   <name>password</name>
15.   <value>globus</value>
```

```
16.   </parameter>
17. - <parameter>
```

Cada uno de los experimentos implica pasos manuales como los citados, el proceso de medición consiste en contar la cantidad de vez que se incurre en esta modificación a medida que se incrementan usuario, recursos y servicios de modo que se demuestre que el costo es proporcionar al crecimiento del grid.

5.3.5 Desempeño

El proceso que se sigue en el primer experimento es con respecto al tiempo de respuesta del contenedor frente a la invocación de un servicio. El script asociado es el siguiente:

```
18. $postmaster –i /usr/local/pgsql/data
19. $globus-start-container
20. $java           –classpath           ..build/stubs/classes/:           $CLASSPATH
    org/globus/datgridHelloWordlServices/cliente/ClienteHello
    http:///192.168.0.2:8080/wrsf/services/datagrid.HelloWorldServices/HelloWor
    dlsServicesFactoryServecr Lennis
```

Las dos primeras instrucciones se ejecutan en el equipo que tiene instalado los servicios y en si GT4 y la instrucción 3 se ejecuta en el mismo equipo desde otra Terminal inicialmente o desde el otro equipo con un usuario ya autenticado.

5.4 Resultados de experimentos

5.4.1 Transaccionalidad

El proceso de replicación permite reducir la carga de acceso a datos e incrementa la disponibilidad de los mismos, proporcionando un acceso más rápido. Adicionalmente, evita la existencia de puntos únicos de fallo ayudando a balancear cargas.

5.4.2 Seguridad - Autenticación

El proceso de autenticación se apoya en el GSI, dado que es este componte es el que soporta la creación del certificado x.509 con la firma de la entidad certificadora que hace parte de la infraestructura creada, con este certificado el usuario no tiene que volver a autenticarse, es decir, a dar su password.

La autenticación se reduce al intercambio de credenciales entre usuarios y servicios. Finalmente, el resultado fue el esperado, se puede concluir que se hace única autenticación, dado que se puede

estar accediendo un gran número de recursos, introducir una credencial cada vez que se cambia de recurso resulta incómodo y poco ventajoso frente a otras soluciones.

Se puede concluir que se haceun único proceso de acceso, dado que podemos estaraccediendo un gran número de recursos y si estamos hablando de un sólo sistema, introducir un credencial cada vez que se cambia de recurso resulta incomodo y poco ventajoso frente a otras soluciones. Con éste certificado es posible contactar con los servicios de recepción de trabajos, pero realmente lo que se realiza en el cliente es obtener un proxy a partir del certificado de usuario que será el que realmente se envíe en la petición del servicio. Básicamente en el servidor se obtiene información del usuario a partir de este proxy, su DN que es el que se utiliza para ver si el usuario está finalmente autorizado o no a utilizar el servicio. Si este servicio necesita la comunicación con otros se utilizará la delegación de este proxy como método para autenticarse en éstos nuevosservicios.

Se valida que la seguridad de Globus esta acorde con de la seguridad del sistema operativo por lo tanto se hace necesario que el administrador del sistema cree el usuario y le asigne credencial.

5.4.3 Seguridad - Autorización

El resultado en seguridad hacia los experimentos de autorización es que no es llevada a cabo directamente por GSI. Es decir una vez que una entidad se ha identificado como tal y se sabe quien es, el problema de autorización se basa en permitir o no el uso de estos recursos a esta entidad autenticada.

Se mantiene en cada recurso que se quiere autenticar un archivo gridmapfile que contiene los Distinguished Names de aquellos certificados que se aceptan, por lo que cada vez que se accede a un servicio de este recurso se comprueba si el proxy con el que se está accediendo está incluido, permitiendo el acceso si así es o rechazándolo en caso contrario

5.4.4 Administrabilidad

Globus requiere muchos pasos manuales, es decir que se tienen que hacer todo por comando para poder hacer las cosas que se requieren, pero eso puede ser ventajoso en la medida en que la tecnología permite que se maneje a criterio del usuario lo que necesite, pero este usuario tiene que tener un gran dominio sobre no sólo los comandos si no de los servicios que se prestan para alcanzar los resultados esperados. La ventaja que trae la administración por comando esque se pueden crear script a la medida de las necesidades del administrador logrando un control de los procedimientos.

El usuario puede agregar o eliminar del gripmapfile siempre que tenga permisos, esto significa que no es cualquier usuario sin o el administrador. Aunque Globus requiere muchos pasosmanuales,

es decir que se tienen que hacer todo por comando para poder hacer las cosas que se requieren, pero eso puede ser ventajoso en la medida en que la tecnología permite que se maneje a criterio del usuario lo que necesite, pero este usuario tiene que tener un gran dominio sobre no sólo los comandos si no de los servicios que se prestan para alcanzar los resultados esperados.

Por ejemplo para la instalación de la base de datos rft de Globus es necesario configurar el archivo jndi-config.xml, en donde se define la cadena de conexión y el usuario que se conecta, si este no es el mismo que se maneja durante el proceso de generación del certificado para los clientes. Si no se utiliza el mismo nombre y en su lugar se usa la IP asociada, cuando se trata de subir los servicios de Globus, el error que la base de datos y el mismo comando Globus-star-cointaner, es que el usuario Globus no puede acceder, la forma de corregir este problema es siempre manejar el nombre de la máquina. Ahora, no se puede pensar en que para usar la IP desde el principio se trabaje con esta dado que el certificado al generarse genera error e impide la autenticación si no se trabaja con los nombres.

Resulta costosa la administración si no hay un punto central de control, de lo contrario, la solución es dar permisos a todo nivel y eso va en contra de cualquier política de seguridad.

5.4.5 Escalabilidad

El RLS no brinda garantía de consistencia entre réplicas de datos o unicidad de servicio de ubicación de nombres de archivos registrados en el directorio. Para esto es necesario diseñar los servicios que provean esta funcionalidad y que utilicen el RLS.

Con el archivo hierarchy.xml se maneja la escalabilidad de servicios, en dos sentidos

- Configurando las URLs que contienen indices

<upstream>http://myresource.isi.edu:8080/wsrf/services/DefaultIndexService</upstream>

- Configurando las URLs de los recursos

<downstream>http://myvo.org:8080/wsrf/services/DefaultIndexService</downstream>

Detalle de experimento:

Se modificó el archivo hierarchy.xml para probar el manejo de creación de jerarquías, dado que en este archivo se maneja el índice de servicios registrados en el servidor, para esto se le quitan los comentarios a la línea:

<upstream>https://vo-host:8443/wsrf/services/DefaultIndexService</upstream>

Donde el nombre del host y el puerto deben ser coherente con los datos del host en donde se están hospedando los servicios. Sin embargo cuando estos se activa el contenedor arroja un error

Experimento Inclusión de Servicios

No tiene restricciones en el número de servicios que se publican y controla que no se duplique por nombre ni servicios. Adicionalmente, la publicación en Globus implica que un conjunto de servicios se ponen a disposición de los usuarios en el contenedor del servicio, ubicado en http: //localhost:8080/wsrf/services (si se utiliza –nosec al momento de iniciar el contenedor, de lo contrario el protocolo será "https" y el puerto será 8443). Los servicios individuales se identificarán dependiendo de lo que denomina ha sido designado a ellos en su archivo de descriptor de despliegue. Para conseguir acceso a estos servicios el URI del servicio se especificará en el código del cliente, y esto se utilizará para encontrar la dirección del recurso del servicio.

5.4.6 Rendimiento

El rendimiento sigue estando bajo la responsabilidad de la capacidad de los equipos, sólo se llega a ver realmente la ventaja si se cuenta con un planificador que distribuya la tarea y permita acortar el tiempo de respuesta. Sin embargo el hecho de consultar varias base de datos toma más tiempo dada la latencia de la red y esto implica que el barrido de los recursos se multiplica proporcionalmente, Es decir si el contenedor demora en contestar 18 segundo por un recurso, el tiempo para n recursos es n * 18, donde n es el número de recursos y 18 es el tiempo promedio de respuesta del contenedor.

Otro factor que resulta relevante a la hora de hablar de rendimiento es la máquina misma en la cual se este trabajando, dado que no es lo mismo tener un contenedor en una máquina con configuración PIII de 600 MHZ y 256 en RAM que un equipo PIII de 1.6 GHZ y 512 en RAM.

6 CONCLUSIONES

1. El objetivo es interconectar de forma débilmente acoplada recursos gestionados por diferentes tecnologías y políticas de seguridad, hoy en día incompatibles.

2. En cuanto a seguridad se aplica el concepto de autenticación única lo cual da la posibilidad de aprovechamiento de recursos. El problema latente esta al inicio, es decir en el momento en el que se monta el ambiente en donde la falta de control sobre la infraestructura, obstaculiza el proceso normal de instalación del ambiente, esto no quiere decir que los usuarios tienen permisos totales sino que se deben tener políticas de operación de los

usuarios de modo que no se vuelve una red pública sino que realmente se compartan los recursos.

3. En cuanto a administrabilidad desde el punto de vista de instalación, configuración y operación de grids, no es amigable, no es un wizard sino que requieren de conocimiento y sumo cuidado en la consecución de los pasos de instalación y configuración dado que puede generar fallos en el ambiente en impedir el correcto desenvolvimiento de servicio propios del ambiente.

4. En escalabilidad, la conclusión es que efectivamente el concepto de integrar muchas fuente es posible aunque cada proceso de escalamiento a nivel de Globus es tedioso, por eso es que OGSA-DAI ofrece los servicios que permiten hacer mediante un sólo comando estos procesos.

5. Datagrid hace un aprovechamiento completo de la ventaja que XML por ejemplo proporciona para su estandarización un proceso, dado que los archivos que se modifican en los procesos desarrollados todos estaban en esta estructura.

6. GlobusToolkit es útil para desarrollo de aplicaciones e integradores de sistemas pero no es una herramienta de usuario, ni una aplicación, ni un planificador. Mientras que OGSA-DAI es un middleware construido para asistir el acceso y la integración de datos desde diferentes fuentes a través del grid.

7. Globus Toolkit proporciona los servicios y recursos para dar respuesta a los RNF de seguridad, administrabilidad, transaccionalidad, escalabilidad y desempeño

7 TRABAJOS FUTUROS

Se espera que los experimentos puedan ser repetidos en otras herramientas de la tecnología Grid, siempre que se vaya a analizar los mismos requerimientos no funcionales, porque de lo contrario sólo se puede hacer replicación del planteamiento metodológico en otros requerimientos no funcionales.

Se pueden generar lineamientos sobre las consideraciones encontradas en cuanto a la apropiación de tecnología, usabilidad y recomendaciones sobre requerimientos no funcionales analizados durante un proceso de diseño de aplicaciones.

8 REFERENCIAS

[1] I. Foster, C. Kesselman, S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of Supercomputer Applications, 15(3), 2001. http://www.Globus.org/alliance/publications/papers.php.

[2] I. Foster, C. Kesselman. Chapter 2 of "The Grid: Blueprint for a New Computing Infrastructure", Morgan-Kaufman, 1999. http://www.Globus.org/alliance/publications/papers.php.

[3] Organizaciones virtuales. http://gridcafe.web.cern.ch/gridcafe/grid&you/VO.html

[4] The Globus project. Commodity Grid Kits. 2002. http://wwwunix.Globus.org/cog/

[5] Globus Project. www.Globus.org/research/papers.html

[6] First Look Oracle Database 10g A Preliminary Report on Oracle.s Latest Release. June 1, 2004

[7] Oracle Grid Computing. An Oracle Business White Paper January 2005. http://www.oracle.com/technologies/grid/

[8] Oracle Grid Computing Glossary. http://www.oracle.com/technologies/grid/

[9] Clabby Analytics. The Grid Report. 2004 Edition

[10] Anatomy of a search engine: http://www-db.stanford.edu/~backrub/google

[11] Sitio oficial de google: http://www.google.com/corporate/history.html

[12] Buyukkokten, O.; Garcia-Molina, H.; Paepcke, A.; Winograd, T.: Power Browser: Efficient Web Browsing for PDAs. http://dbpubs.stanford.edu:8090/pub/1999-32

[13] Goldman, R.; Widom, J.: WSQ/DSQ: A Practical Approach for Combined Querying of Databases and the Web. http://dbpubs.stanford.edu:8090/pub/1999-44

[14] Buyukkokten, O.; Garcia-Molina, H.; Paepcke, A.: Focused Web Searching with PDAs. http://dbpubs.stanford.edu:8090/pub/1999-29 09/02/04

[15] I. Foster, C. Kesselman, J. Nick, S. Tuecke. "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration." Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002.

[16] Ian Baird. "Understanding Grid Computing" Daily News and Information for the Global Grid Community / July 1, 2002: vol. 1 no. 3

[17]. BRIN, S. and PAGE, L. The anatomy of a large-scale hypertextual Web search engine. Computer Netsorks and ISDN Systems, 30, 1998. http://www-db.stanford.edu/~backrub/google.html

[18] EU DataGrid Project http://web.DataGrid.cnr.it

[19] DOE Science Grid http://www.doesciencegrid.org

[20] I. Foster, C. Kesselman,. "Globus: A Metacomputing Infrastructure Toolkit". International Journal of Supercomputer Applications, vol .11 no.2, 1997

[21] Globus Project Open Grid Services Architecture Working Group https://forge.gridforum.org/projects/ogsa-wg

[22] Grid Service Development Tools Guide
http://wwwunix.Globus.org/toolkit/3.0/ogsa/docs/tools_guide.html
[23] Open Grid Services Infrastructure Working Group https://forge.gridforum.org/projects/ogsi-wg
[24] GT4 documentation http://www.Globus.org/toolkit/docs/4.0/.
[25] Grid Forum http://www.globalgridforum.com/
[26] SUN http://www.sun.com/solutions/landing/infrastructure/grid.xml
[27] Welcome to Teragrid, home-page: http://www.teragrid.org
[28] SETI@home: Search for Extraterrestrial Intelligence at home: setiathome.ssl.berkeley.edu
[29] EGEE: Enabling Grids for e-Science in Europe: http://www.eu-egee.org
[30] Forum global de Grid (GGF), 5 de agosto de 2002 CIENCIA DIGIT@L home-page: www.cienciadigital.net
[31] UK Engineering Task Force Globus Toolkit Version 4 Middleware Evaluation. http://www.nesc.ac.uk/technical_papers/UKeS-2005-03.pdf
[32] Performance Analysis of the OGSA-DAI Software http://www.allhands.org.uk/2004/proceedings/papers/142.pdf
[33] Can GRID Services Provide Answers to the Challenges of National Health Information Sharing? http://portal.acm.org/citation.cfm?id=961330&dl=ACM&coll=GUIDE

9 ANEXOS

9.1 Anexo 1: Creación de tablas para Postgres

```
CREATE TABLE TIPDOCUMENTO(

tipIDTipDocumento int primary key,

tipNomTipDocumento varchar(50)

);

CREATE TABLE SEXO (

tipIDSexo int primary key,

tipNomSexo varchar(50)

);

CREATE TABLE PERSONA(

cliIDPersona int primary key,

cliIDTipDocumento int references TIPDOCUMENTO (tipIDTipDocumento),

cliNúmeroIden varchar(20),

cliNombres varchar (80),

cliApellidos varchar (80),

cliFchNacimiento date,

cliIDSexo int references SEXO(tipIDSexo)
```

```
);

CREATE TABLE PLAN(
tipIDPlan int primary key,
tipNomPlan varchar(50)
);
CREATE TABLE REGIMEN (
tipIDRegimen int primary key,
tipNomRegimen varchar(50)
);

CREATE TABLE TIPAFILIADO (
tipIDTipAfiliado int primary key,
tipNomTipAfiliado varchar(50)
);
CREATE TABLE AFILIADO(
cliIDPersona int references PERSONA (cliIDPersona),
afiFchAfiliacion date,
afiFchRadicacion date,
afiSemanasCotizadas int,
```

```
afiSemanasContinuas int,
afiIDPlan int references PLAN (tipIDPlan),
afiIDRegimen int references REGIMEN(tipIDRegimen),
afiIdTipAfiliado int references TIPAFILIADO (tipIDTipAfiliado),
PRIMARY KEY(cliIDPersona,afiIDPlan, afiIDRegimen)
);
```

9.2 Anexos 2: Inserción de datos

```
insert into PLAN (tipIDPlan ,tipNomPlan )
values (1,'individual');
insert into PLAN (tipIDPlan ,tipNomPlan )
values (2,'colectivo');

insert into REGIMEN (tipIDRegimen, tipNomRegimen)
values (1, 'contributivo');
insert into REGIMEN (tipIDRegimen, tipNomRegimen)
values (2, 'subsidiado');

insert into  TIPAFILIADO (tipIDTipAfiliado , tipNomTipAfiliado)
values (1,'cotizante');
insert into  TIPAFILIADO (tipIDTipAfiliado , tipNomTipAfiliado)
```

values (2,'beneficiario');

9.3 Anexos 3: Servicio base experimental

```
import uk.org.ogsadai.client.toolkit.GenericServiceFetcher;

import uk.org.ogsadai.client.toolkit.Response;

import uk.org.ogsadai.client.toolkit.activity.ActivityRequest;

import uk.org.ogsadai.client.toolkit.activity.sql.SQLQuery;

import uk.org.ogsadai.client.toolkit.activity.sql.WebRowSet;

import uk.org.ogsadai.client.toolkit.service.DataService;

/**

 * A simple example of using the client toolkit to run an SQL

 * query and return the results in the response to the client

 * as WebRowSet.

 *

 * @author The OGSA-DAI team.

 */

public class SimpleExample {


    // Copyright statement

    private static final String COPYRIGHT_NOTICE =
```

"(c) International Business Machines Corporation, 2005. (c) University of Edinburgh 2002 - 2005.";

```
   public static void main(String[] args) throws Exception {
            String handle = "http://localhost:8080/wsrf/services/ogsadai/DataService";
            String id = "MySQLResource";
            DataService service = GenericServiceFetcher.getInstance().getDataService(handle, id);
            SQLQuery query =
                  new SQLQuery("select * from Plan where tipIDPlan ='1'");
      WebRowSet rowset = new WebRowSet( query.getOutput() );
      ActivityRequest request = new ActivityRequest();
      request.add( query );
      request.add( rowset );
            Response response = service.perform( request );
            System.out.println(response.getAsString());
   }
}
```

9.4 Anexos 4: Detalle experimento de seguridad

Se realizo el proceso de ejecución de un servicio mediante el globusrun, sin incluir el usuario Lennis, el resultado fue el esperado.

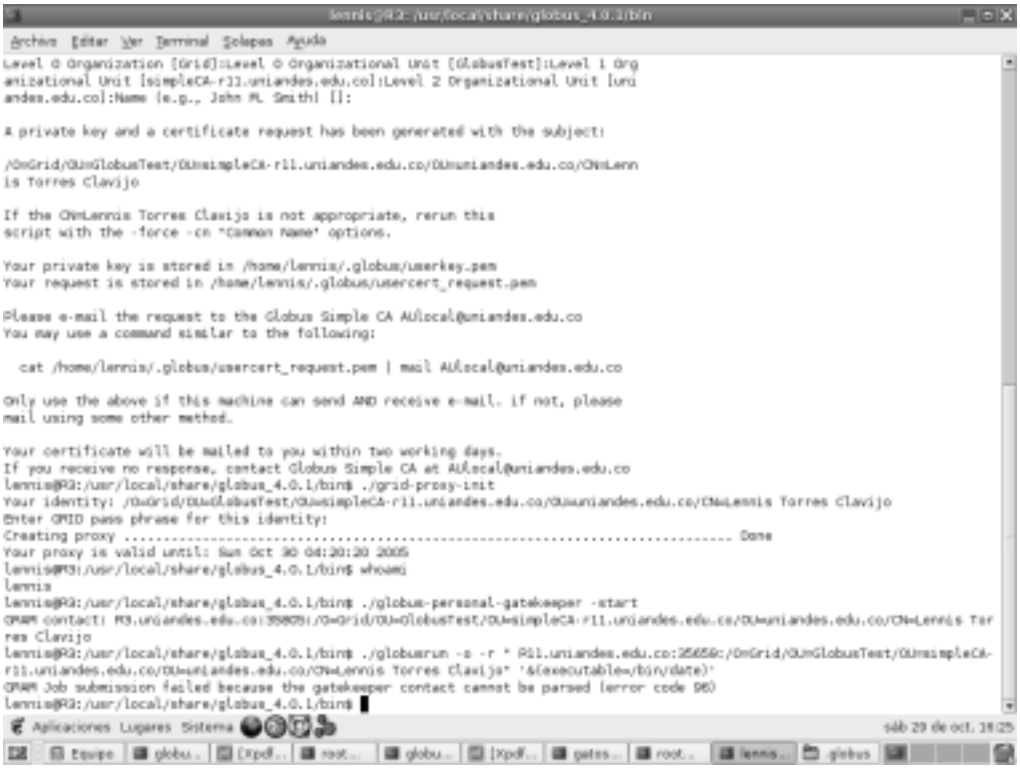

Figura 15 – Usuario lennis sin crear

Se incluyo el usuario Lennis y se reintento el comando globusrun

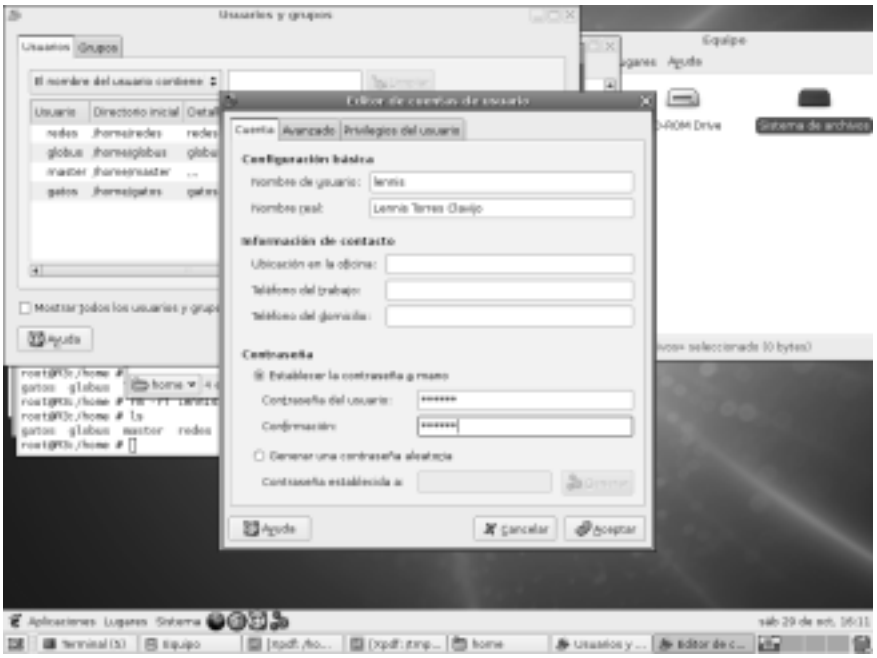

Figura 16 – Creación Usuario lennis

El resultado fue el esperado, se puede concluir que hacer único logeo, dado que podemosestar accediendo un gran número de recursos y si estamos hablando de un sólo sistema, introducirun credencial cada vez que se cambia de recurso resulta incomodo y poco ventajoso frente a otras soluciones.

Figura 17 – Ejecución globusrun

9.5 Anexos 5: Detalle experimento de desempeño

Con la instrucción globus-start-conteiner se levanta el contenedor de servicios de globus

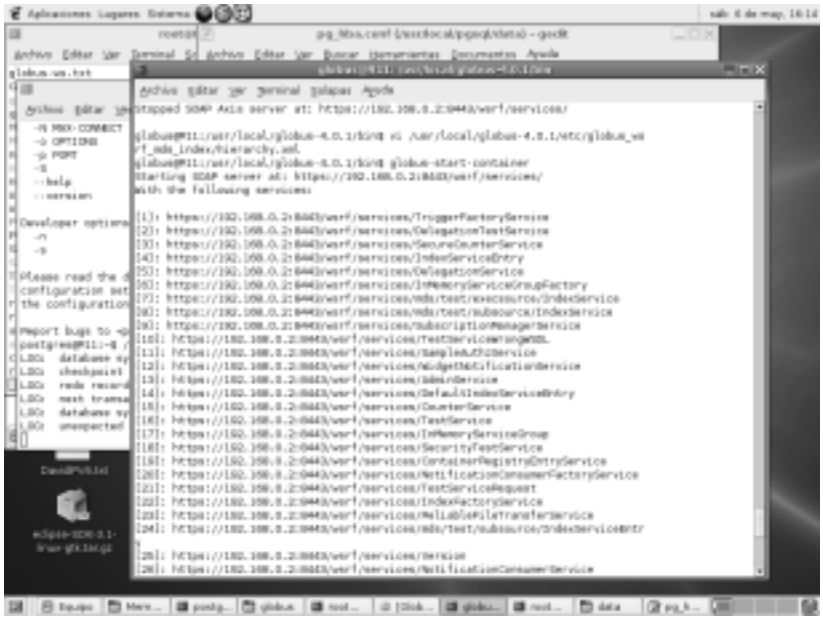

Figura 18 – Contenedor con seguridad

9.6 Anexos 6: Deploy de servicios

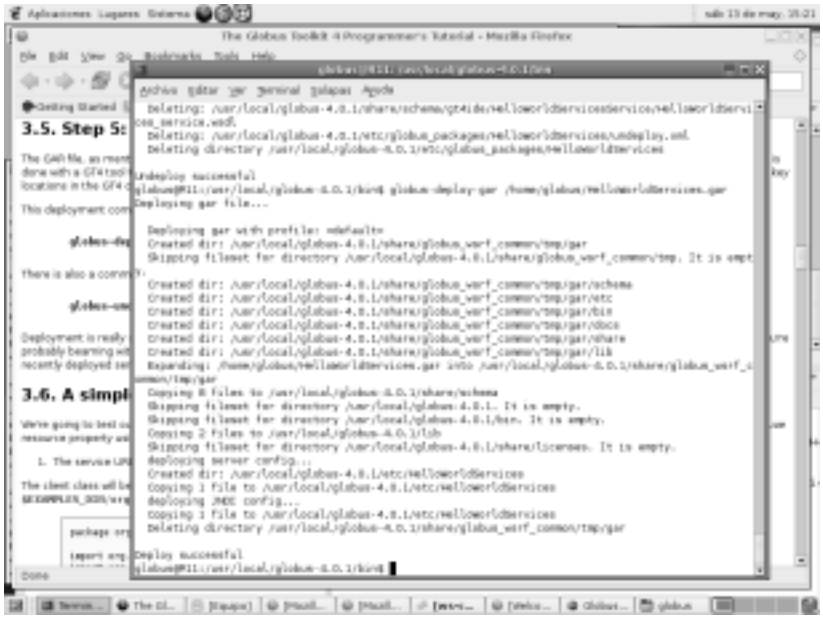

Figura 19 – Deploy de servicios

9.7 Anexos 7: jndi-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
- <jndiConfig xmlns="http://wsrf.globus.org/jndi/config">
- <service name="ReliableFileTransferFactoryService">
- <resource name="home" type="org.globus.wsrf.impl.ServiceResourceHome">
- <resourceParams>
- <parameter>
<name>factory</name>
<value>org.globus.wsrf.jndi.BeanFactory</value>
</parameter>
</resourceParams>
</resource>
- <resource name="mdsConfiguration"
type="org.globus.wsrf.impl.servicegroup.client.MDSConfiguration">
- <resourceParams>
- <parameter>
<name>reg</name>
<value>true</value>
</parameter>
```

<parameter>

<name>**factory**</name>

<value>**org.globus.wsrf.jndi.BeanFactory**</value>

</parameter>

</resourceParams>

</resource>

</service>

<service name="**ReliableFileTransferService**">

<resource name="**configuration**" type="**org.globus.transfer.reliable.service.RFTConfiguration**">

<resourceParams>

<parameter>

<name>**factory**</name>

<value>**org.globus.wsrf.jndi.BeanFactory**</value>

</parameter>

<parameter>

<name>**backOff**</name>

<value>**10000**</value>

</parameter>

<parameter>

<name>**maxActiveAllowed**</name>

<value>**100**</value>

</parameter>

</resourceParams>

</resource>

- <resource name="**dbConfiguration**" type="**org.globus.transfer.reliable.service.database.RFTDatabaseOptions**">

- <resourceParams>

- <parameter>

<name>**factory**</name>

<value>**org.globus.wsrf.jndi.BeanFactory**</value>

</parameter>

- <parameter>

<name>**driverName**</name>

<value>**org.postgresql.Driver**</value>

</parameter>

- <parameter>

<name>**connectionString**</name>

<value>**jdbc:postgresql://R11.uniandes.edu.co/rftDatabase**</value>

</parameter>

<parameter>
<name>**userName**</name>
<value>**globus**</value>
</parameter>
<parameter>
<name>**password**</name>
<value>**globus**</value>
</parameter>
<parameter>
<name>**maxActive**</name>
<value>**20**</value>
</parameter>
<parameter>
<name>**maxIdle**</name>
<value>**10**</value>
</parameter>
<parameter>
<name>**maxWait**</name>
<value>**-1**</value>

```
</parameter>
</resourceParams>
</resource>
- <resource name="home" type="org.globus.transfer.reliable.service.ReliableFileTransferHome">
- <resourceParams>
- <parameter>
<name>factory</name>
<value>org.globus.wsrf.jndi.BeanFactory</value>
</parameter>
- <parameter>
<name>resourceClass</name>
<value>org.globus.transfer.reliable.service.ReliableFileTransferResource</value>
</parameter>
- <parameter>
<name>resourceKeyName</name>
<value>{http://www.globus.org/namespaces/2004/10/rft}TransferKey</value>
</parameter>
- <parameter>
<name>resourceKeyType</name>
```

<value>**java.lang.String**</value>

</parameter>

- <parameter>

<name>**sweeperDelay**</name>

<value>**60000**</value>

</parameter>

</resourceParams>

</resource>

</service>

</jndiConfig>

9.8 Anexos 8: Mediciones

El proceso de medición se realizo en las sala Waira, en donde ya se encontraba instalado el ambiente, sólo restaba ejecutar experimentos. El experimento 3 que se reviso fue la autorización de los usuarios, en este caso, se creaba usuario y se certificaban a partir del proceso de certificación explicado en los script, sobre las maquinas que se incluyen en el grid, como lo muestra la figura siguiente:

Figura 20 – Creación de certificados

La creación de cada certificación, se debe realizar por cada usuario que se integra al grid, y en cada maquina que se vincula al grid, esto dado el experimento 5, la primera medida que se realizo fue:

Nro. Maquinas / Nro. Usuarios	1	2	3	Total
1	1	2	3	6
2	2	4	6	12
4	4	8	12	24
Total	7	14	21	

Tabla 4 Número de Maquina vs Número de usuarios

Para tener autorización a los recursos el usuario tiene que crearse tantas veces como recursos acceda, con estas mediciones se evidencia el desgaste administrativo que conlleva.

Adicionalmente, se tomaron medidas a partir de los tamaños de los archivos que se genera por cada certificación, esto para dar respuesta del experimento 13 de escalabilidad, en donde, el limite lo determina la capacidad de la maquina misma para almacenar los certificados de cada usuario. En la figura siguiente se muestras los tamaños para los dos usuarios que se tienen creados.

Figura 21 – Listado de archivos de certificación

En la tabla siguiente se muestras los datos al hacer el ingreso de usuarios como se ve reflejado en espacio en disco de cada uno de los clientes.

Numero de Usuarios	usercert.pem	usercert_request	userkey.pem	Total
1	2749	1463	963	5175
2	5498	2926	1926	10350
4	10996	5852	3852	20700
8	21992	11704	7704	41400
16	43984	23408	15408	82800
32	87968	46816	30816	165600
50	137450	73150	48150	258750

Tabla 5 Número de usuários vs Certificado

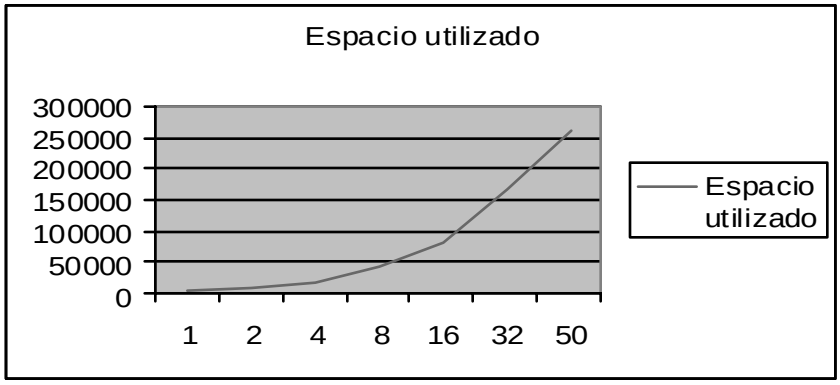

Figura 22 – Curva incremental de espacio por certificado

Finalmente, para tener acceso no sólo es necesaria la creación y certificación de usuario, sino también su vinculación en el gridmapfile, este es un proceso que adiciona peso a la carga administrativa que conlleva el manejo de los usuarios en el grid.

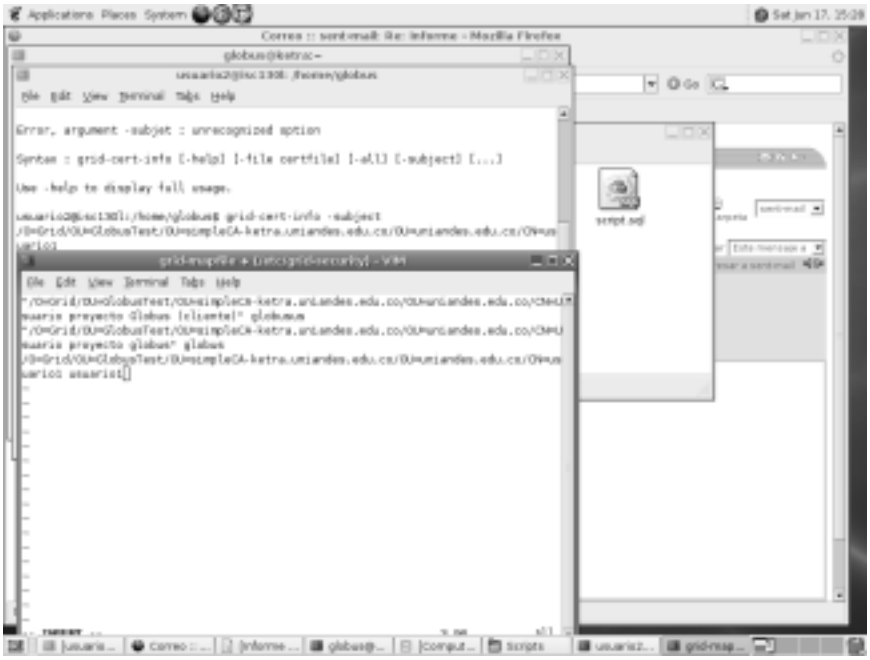

Figura 23 – Modificación de gridmapfile