Universidad de Granada

Departamento de Ciencias de la Computación e Inteligencia Artificial

Aplicación de la tecnología Grid al diseño y desarrollo de un portal de recursos computacionales

José Ruedas Sánchez

Tesis Doctoral

Editor: Editorial de la Universidad de Granada
Autor: José Ruedas Sánchez
D.L.: Gr.2104 - 2006
ISBN: 84-338-4133-5

Aplicación de la tecnología Grid al diseño y desarrollo de un portal de recursos computacionales

Memoria que presenta

José Ruedas Sánchez

para optar al grado de Doctor

Septiembre de 2006

Director

José Luis Verdegay Galdeano

Departamento de Ciencias de la Computación e Inteligencia Artificial

La memoria titulada ***Aplicación de la tecnología Grid al diseño y desarrollo de un portal de recursos computacionales***, que presenta D. José Ruedas Sánchez para optar al grado de Doctor, ha sido realizada en el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada, bajo la dirección del Dr. D. José Luis Verdegay Galdeano, catedrático del referido departamento.

Granada, septiembre de 2006

El doctorando

Fdo. José Ruedas Sánchez

El director

Fdo. José Luis Verdegay Galdeano

Agradecimientos

Ante todo, deseo expresar mi agradecimiento a todas aquellas personas e instituciones que, con su ayuda, han contribuido de forma especial a la realización de esta tesis doctoral.

En primer lugar quiero agradecer al Dr. José Luis Verdegay, director de esta tesis, el constante apoyo y estímulo que me ha dispensado, así como su desinteresada disposición para la elaboración de esta memoria.

No menos influyente ha sido el Prof. Rafael Rodrigo Montero, quien depositó en mí su confianza para dirigir el Centro de Cálculo del Instituto de Astrofísica de Andalucía y propuso el tema de esta tesis. Mi más sincero agradecimiento.

Así mismo, quisiera expresar mi gratitud a todo el personal del Instituto de Astrofísica de Andalucía, y en su nombre a su director, Dr. José Carlos del Toro Iniesta. Gracias por la ayuda que me han dispensado, tanto humana como material. Destacar, a la Dra. Josefa Masegosa Gallego por aportar de su proyecto de investigación los recursos necesarios para el desarrollo de este trabajo y al Dr. Jaime Perea Duarte por su interés y asesoramiento. Mi gratitud a ambos.

Indudablemente, este trabajo no podría haberse llevado a cabo sin el apoyo incondicional de todo el equipo de IRISGrid. Agradezco efusivamente su valiosa colaboración, en especial a Javier Masa Marín, Antonio Fuentes Bermejo, Ignacio Martín Llorente, Rubén S. Montero, Eduardo Huedo y Fco. Javier Sánchez Martínez.

Mi más profundo agradecimiento y respeto es para mi madre. Su ilusión siempre me ha acompañado. A mi familia y amigos quiero darles las gracias por haber estado siempre apoyándome, y por supuesto, a mi mujer por haber estado a mi lado en todo momento respetando mi decisión.

A mi hijo Raúl, por todos los motivos imaginables.

Prólogo

La necesidad de evolucionar a tecnologías de computación basadas en la integración de recursos geográficamente distribuidos es un hecho incuestionable, que trataremos de hacer patente a lo largo de la memoria que aquí comienza.

Los proyectos científicos abordan objetivos cada vez más ambiciosos que requieren la resolución de problemas computacionales complejos, tanto por el volumen de los cálculos a realizar como por el tamaño y complejidad de las bases de datos utilizadas. También tenemos que tener en cuenta que los equipos científicos son en muchos casos el fruto de colaboraciones internacionales con miembros distribuidos por todo el planeta.

Proyectos multidisciplinares como la simulación y análisis de datos en Física de Partículas, Industria Aeronáutica y Automovilística, Biología Computacional (Proteómica y Genómica), análisis de imágenes de diagnóstico en Medicina, modelos predictivos en Ciencias Medio Ambientales, análisis observacional en Astronomía, etc. requieren obtener resultados en un periodo de tiempo razonable y adecuado. Es decir, estos proyectos necesitan una tecnología que permita la ejecución de una aplicación en menos tiempo (*Alto Rendimiento – High Performance Computing*), y que permita la ejecución de un número mucho mayor de aplicaciones (*Alta Productividad – High Troughput Computing*)

La necesidad de aprovechar los recursos disponibles en los sistemas informáticos conectados a Internet simplificando su utilización, ha dado lugar a una nueva disciplina en tecnología de la información conocida como Grid Computing. El concepto es análogo a las redes de suministro eléctrico: ofrecen un único punto de acceso a un conjunto de recursos distribuidos geográficamente en diferentes dominios de administración (supercomputadores, clusters, almacenamiento, fuentes de información, instrumentos, personal, bases de datos, etc.). De este modo, los sistemas distribuidos se pueden emplear como un único sistema virtual en aplicaciones intensivas en datos o con gran demanda computacional.

Esta tecnología marca un punto de inflexión con los proyectos de supercomputación previos en el entorno profesional científico, donde los recursos experimentales y humanos tienen en general un costo elevado y deben ser optimizados. La previsión es que con esta tecnología se podrán afrontar proyectos de investigación y retos tecnológicos a gran escala.

La idea básica de la tecnología Grid es, en resumen, aprovechar de modo óptimo los recursos proporcionados por equipos de computación distribuidos, mediante el uso de un software adecuado para planificar su utilización que tenga en cuenta las prioridades y diferentes periodos de demanda de los usuarios. La evolución de las redes de comunicaciones de alta velocidad dedicadas a la investigación han creando un escenario idóneo para el despliegue de esta tecnología, y aunque en nuestro país no está muy desarrollada, dado el creciente interés por aplicarla a una gran variedad de áreas temáticas, cualquier desarrollo científico-tecnológico asociado a la misma está totalmente justificado.

El principal objetivo, es que los grupos de investigación nacionales puedan, sin necesidad de realizar una gran inversión económica en recursos propios, explotar los recursos que ofrecen los centros e instituciones afiliadas a la red académica y de investigación nacional (RedIRIS), mediante la aplicación de la tecnología Grid. Este objetivo se puede lograr gracias a la iniciativa IRISGrid, en la que el autor participa como coordinador de área dentro de los proyectos de investigación del Instituto de Astrofísica de Andalucía (CSIC).

La iniciativa IRISGrid, surge con el objetivo de coordinar a nivel académico y científico a los grupos de investigación interesados en esta tecnología, tanto en su desarrollo, implantación y aplicaciones. Además de esta coordinación, IRISGrid tiene como objetivo crear la infraestructura Grid nacional que permita el uso de esta tecnología tanto a nivel de aplicabilidad en diferentes ámbitos, como a nivel de desarrollo e innovación en este campo. Actualmente, en la iniciativa IRISGrid participan diferentes grupos del ámbito científico y académico, y está reconocida como la iniciativa nacional para la implantación, desarrollo e investigación en entornos Grid.

Por todo esto el objetivo de esta Tesis Doctoral es la aplicación de la tecnología Grid, utilizando recursos de la red académica y de investigación nacional (RedIRIS), mediante el diseño y desarrollo de un portal Web, con objeto de proporcionar a la comunidad científica las herramientas necesarias para el acceso a los recursos computacionales, de forma eficiente, transparente y segura. Para ello la memoria se ha redactado en tres capítulos que describimos a continuación.

En el primero se realiza un breve análisis de las diferentes tecnologías de computación, haciendo especial énfasis en la tecnología de computación basada en la integración de recursos geográficamente distribuidos (Grid). Esta tecnología coordina recursos que no están sujetos a un control centralizado, usando protocolos e interfaces basados en estándares abiertos y de propósito general: permite interconectar recursos en diferentes dominios de administración respetando sus políticas internas de seguridad. También se realiza un análisis exhaustivo de la tecnología de computación Grid; se analiza la situación actual de esta tecnología, citando muchos de los proyectos e iniciativas relacionadas con el desarrollo de la misma. El análisis se efectúa en base a su arquitectura, es decir en el establecimiento de protocolos para la definición de servicios, interfaces de programación de aplicaciones y herramientas de desarrollo de software. En este capítulo, también se relacionan las diferentes aplicaciones y características que debe tener un sistema de computación Grid.

El segundo capítulo se dedica al proyecto Globus. La computación Grid necesita de importantes avances en mecanismos de comunicación, técnicas y herramientas; el proyecto Globus, nació con el objetivo de acelerar estos avances y de resolver los problemas de configuración y optimización de los entornos de computación Grid. La transferencia de tecnología entre proyectos, ha permitido definir Globus como el estándar para el desarrollo de aplicaciones (*middleware*) en entorno Grid. Se realiza el estudio y análisis de Globus toolkit en base a sus componentes y sus correspondencias con los diferentes servicios de alto nivel, y se proporciona un método genérico para la instalación y configuración de Globus toolkit, en el entorno de computación Grid LHC del CERN (LCG). Este proceso consiste básicamente en realizar de forma eficiente las principales tareas de instalación, configuración y

gestión de infraestructuras Grid basada en Globus, en relación con la ejecución de trabajos, entidades de certificación, sistemas de información y transferencias de archivos. Finalmente nos centramos en la instalación y configuración, del Interfaz de Usuario como punto de acceso a los servicios Grid y como base para el diseño y desarrollo del portal de recursos computacionales.

El tercer capítulo corresponde al diseño y desarrollo del portal. El control de acceso a la información, se realiza en base a la configuración de directivas y las variables de entorno del protocolo SSL; el portal proporciona autenticación y privacidad de la información mediante el uso de criptografía, utilizando los protocolos Secure Sockets Layer (SSL) y Transport Layer Security (TLS). La versión actual del middleware, utiliza el sistema operativo Scientific Linux 3 (SL3). En este capítulo se analiza la configuración del servidor Web Apache, incluido en la distribución SL3, así como también una serie de módulos diseñados para mejorar su funcionalidad; se describen los parámetros de configuración para los diferentes tipos de nodos que se pueden instalar en una infraestructura Grid basada en servicios Globus y la distribución de los ficheros correspondientes al diseño y programación de las páginas Web del portal y los módulos desarrollados para la integración con los servicios Globus. Las páginas Web se han diseñado con hojas de estilos CSS.

El desarrollo del portal se divide en tres fases: la asociada al acceso autorizado, la relativa a los servicios proporcionados por el portal y la referida a la integración de los servicios Globus. Estas tres fases se estudian y describen en profundidad en sucesivas y respectivas secciones. También se describen los distintos atributos que se pueden definir para la descripción de un trabajo; estos atributos representan información específica, necesaria para poder planificar y ejecutar el trabajo.

Para terminar se presentan las conclusiones y líneas actuales y futuras de trabajo, describiendo y analizando la posible estructura de un programa de *e-Ciencia*, en base a su arquitectura, organización y los requerimientos necesarios para crear un centro de *e-Ciencia*. Finalmente, la memoria concluye con la recopilación de la bibliografía consultada para su elaboración, adjuntándose dos anexos técnicos y un glosario de términos. Sobre este aspecto, consideramos necesario llamar la atención

del lector para explicar el uso repetido del inglés para referir los términos técnicos. Igual que es obvio que el autor no desconoce sus correspondientes traducciones al español, también lo es que si se redactara la tesis con esas traducciones, lo más probable es que cualquier experto en la materia no entendiera a que protocolo, algoritmo, etc. nos estábamos refiriendo. Por eso adoptamos la decisión de mantener las denominaciones originales, que por otra parte son las que comúnmente se emplean en el área, tanto en el texto del trabajo, como en algunas figuras que lo ilustran.

Indice

Prólogo i

1. Introducción

1.1. Tecnologías de computación 1
1.2. Antecedentes y perspectivas de la computación Grid 3
1.3. Arquitectura Grid 8
1.3.1. *Fabric layer* 9
1.3.2. *Connectivity layer* 11
1.3.3. *Resource layer* 12
1.3.4. *Collective layer* 14
1.3.5. *Aplications layer* 17
1.4. Protocolos InterGrid 18
1.5. Aplicaciones de computación Grid 19
1.6. Características de un sistema de computación Grid 21
1.7. Proyecto Globus 22

2. Globus toolkit

2.1. Componentes 25
2.1.1. *Localización y asignación de recursos* 25
2.1.2. *Comunicaciones* 26
2.1.3. *Servicios de información de recursos* 28
2.1.4. *Creación de procesos* 29
2.1.5. *Interfaz de autentificación* 29
2.1.6. *Servicios de acceso a datos* 33
2.1.7. *Servicios de gestión de recursos y aplicaciones* 33
2.1.8. *Servicios de alto nivel* 35

2.2. Instalación y configuración 37
2.2.1. *Tipos de nodos* 39
2.2.2. *Parámetros de configuración* 39
2.2.3. *Método de instalación* 45
2.2.4. *Método de configuración* 46
2.2.4. *Interfaz de usuario* 47

3. Diseño y desarrollo del portal
3.1. Configuración 49
3.1.1. *Secure Sockets Layer* 49
3.1.2. *Directivas SSL* 53
3.1.3. *Variables de entorno SSL* 59
3.1.4. *Configuración de directivas* 61
3.1.5. *Configuración de directorios* 64
3.2. Hojas de estilos CSS 67
3.3. Acceso autorizado 71
3.3.1. *Certificado de usuario* 73
3.3.2. *Registro en una Organización Virtual* 76
3.3.3. *Solicitud de cuenta de usuario* 78
3.4. Servicios proporcionados por el portal 80
3.4.1. *Relación de servicios* 80
3.4.2. *Certificado proxy* 82
3.4.3. *Recursos computacionales* 83
3.4.4. *Enviar trabajos* 86
3.4.5. *Estado de los trabajos* 88
3.4.6. *Información de los trabajos* 92
3.4.7. *Monitorización de los trabajos* 94
3.4.8. *Finalizar la sesión* 95
3.5. Integración de servicios Globus 96
3.6. Lenguaje de descripción de trabajos 164

Conclusiones 173
Bibliografía 181

Anexos

I. Atributos para la designación de recursos

II. Referencia de comandos Globus

Glosario de términos

Figuras

1.2. Evolución innovaciones tecnológicas 4

1.3. Arquitectura Grid (*Layers*) 9

1.3.4. Arquitectura Grid (*Protocolos*) 16

1.3.5. Arquitectura Grid (*APIs y SDKs*) 18

3.3. Acceso autorizado 72

3.3.1a. Solicitud de certificado de usuario X.509 73

3.3.1b. Solicitud de certificado de usuario X.509 a pkIRISGridCA 74

3.3.2a. Registro en una Organización Virtual (VO) 76

3.3.2b. Selección de una Organización Virtual (VO) 77

3.3.3. Solicitud de cuenta de usuario 78

3.4.1. Relación de servicios proporcionados por el portal 80

3.4.2. Cambio de validez del certificado proxy 82

3.4.3a. Información de los recursos computacionales disponibles 84

3.4.3b. Información de los recursos de almacenamiento disponibles 85

3.4.4. Enviar un trabajo para su ejecución en el Grid 87

3.4.5a. Consultar el estado de los trabajos enviados al Grid 88

3.4.5b. Información del estado de los trabajos enviados al Grid 89

3.4.5c. Diagrama de transición de estados 91

3.4.6a. Información del trabajo seleccionado (*Retrieve*) 92

3.4.6b. Información del trabajo seleccionado (*Output*) 93

3.4.7. Monitorización del trabajo seleccionado 94

3.4.8. Confirmación para finalizar la sesión 95

3.6. Diagrama de dependencias entre trabajos 164

1

Introducción

1.1. Tecnologías de computación

Entre las tecnologías de computación que a continuación vamos a considerar, destaca en primer lugar la "***Supercomputación***", es decir, la computación centralizada basada en servidor. Los problemas que surgen con esta tecnología son: falta de escalabilidad; mantenimiento y equipos muy caros, que una vez adquiridos pasan mucho tiempo desaprovechados; las demandas de cálculo son puntuales.

En paralelo al desarrollo de supercomputadores con un número creciente de procesadores, una de las soluciones con más éxito en los últimos años ha sido la construcción de clusters de ordenadores individuales (granjas), consistente en un conjunto de estaciones o computadores personales homogéneo dedicado a computación paralela. Con esta tecnología, denominada "***Cluster Computing***" se obtiene una mejor relación coste/rendimiento (3-10 veces) a la adquisición de un sistema multiprocesador. Sin embargo el modelo de programación es bastante más complejo así como su mantenimiento. Esta tecnología supone una buena solución para aplicaciones HTC (*Computación de Alta Productividad*).

Por otro lado, la "***Intranet Computing***" se basa en la utilización de los equipos de una red departamental para ejecutar trabajos secuenciales o paralelos por medio de una herramienta de gestión de carga, realizando una explotación de potencia computacional distribuida. Con esta tecnología se aumenta el aprovechamiento de los recursos informáticos, se mejora la escalabilidad y la fiabilidad, facilidad de administración y de sustitución de equipos obsoletos. Sin embargo no pueden gestionar recursos fuera del dominio de administración (algunas herramientas permiten colaboración interdepartamental asumiendo la misma estructura administrativa). Por tanto, la escalabilidad está limitada a la organización en picos de demanda y no se puede compartir recursos con otras organizaciones.

La "***Grid Computing***" es una nueva tecnología cuyo objetivo es la compartición de recursos en Internet de forma uniforme, transparente, segura, eficiente y fiable. La tecnología Grid nace dentro de la comunidad de supercomputación y de nuevo está basada en las dos acciones, agregar y compartir, junto a la evolución de la red. El concepto es análogo a las redes de suministro eléctrico: ofrecen un único punto de acceso a un conjunto de recursos distribuidos geográficamente en diferentes dominios de administración (supercomputadores, clusters, almacenamiento, fuentes de información, instrumentos, personal, bases de datos…).

Esta tecnología es complementaria a las anteriores en el sentido que permite interconectar recursos en diferentes dominios de administración respetando sus políticas internas de seguridad y su software de gestión de recursos en la Intranet. Coordina recursos que no están sujetos a un control centralizado, usando protocolos e interfaces basados en estándares abiertos y de propósito general. Transformando el desafío que supone la organización de los correspondientes recursos de computación, la tecnología Grid propone agregar y compartir recursos de computación distribuidos entre diferentes organizaciones e instituciones, a través de redes de alta velocidad, de modo que el acceso a los mismos sea sencillo, flexible y fiable.

La computación utilizando recursos distribuidos a través de la red no es una cuestión trivial. Las aplicaciones del denominado entorno de *High Throughput Computing* (HTC), son más sencillas de adaptar a una ejecución distribuida, mediante su división en múltiples partes. Por ejemplo, la simulación de cien millones de colisiones en un experimento de Física de Partículas puede realizarse de modo distribuido en cien máquinas cada una de las cuales realiza de manera independiente la simulación de un millón de sucesos. Por el contrario, en un entorno de *High Performance Computing* (HPC), se requiere una respuesta inmediata global del sistema para que la aplicación progrese y el cálculo no puede distribuirse de manera independiente.

Estas fronteras se difuminan tanto más en cuanto la capacidad de la red en tiempo de respuesta y de transferencia de datos mejora en comparación con el tiempo de

ejecución de cada paso en la aplicación, cuando esta contiene componentes paralelizables. Un caso típico es el entrenamiento de una red neuronal distribuida, donde el error en cada paso sucesivo se puede calcular de modo independiente distribuyendo la muestra de entrenamiento entre los diferentes nodos, y el error global se obtiene al final de cada paso agregando los resultados de los mismos. Las técnicas de paralelismo aplicables a sistemas multiprocesador, como las basadas en el uso de MPI pasan a poder aplicarse a nodos conectados en red.

Con esta tecnología, se obtienen los siguientes *beneficios*:

- Amortización de recursos propios.
- Gran potencia de cálculo a bajo precio sin necesidad de adquirir equipamiento.
- Mayor colaboración y compartición de recursos entre varios centros.
- Creación de organizaciones virtuales (VOs).
- Negocios basados en proveer recursos.

Los *desafíos técnicos* a los que se enfrenta esta nueva tecnología son:

- Recursos heterogéneos.
- Desarrollo de técnicas y herramientas para el descubrimiento, selección, reserva, asignación, gestión y monitorización de recursos.
- Desarrollo de aplicaciones (middleware) y de modelos eficientes de uso.
- Comunicaciones y seguridad.

Los *desafíos socioeconómicos* a los que se enfrenta son:

- Económicos: precio de los recursos, oferta/demanda …
- Organizativos: política de seguridad, dominios de administración, modelo de explotación y costes …

1.2. Antecedentes y perspectivas de la computación Grid

En 1965 Gordon Moore pronosticó el rápido ritmo de las innovaciones tecnológicas: (i) El rendimiento de un procesador se duplica cada 18 meses; la potencia de cálculo de los supercomputadores sólo crece linealmente.

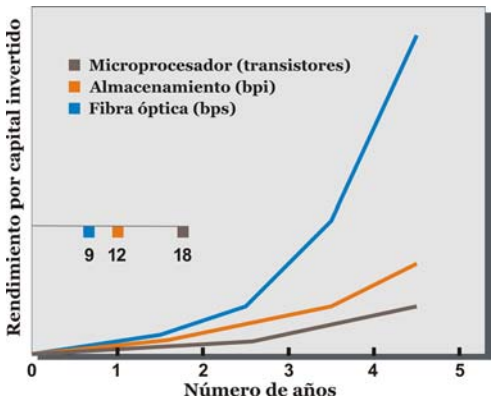


Fig 1.2. *Evolución innovaciones tecnológicas*

(ii) La capacidad de almacenamiento se duplica cada 12 meses; un único sistema no será capaz de analizar los datos que almacenen sus discos; un único centro no podrá analizar el volumen de información generado. (iii) El ancho de banda de red, se duplica cada 9 meses; la red permitirá de forma eficiente usar recursos distribuidos.

En 1969, Leonard Kleinrock, uno de los padres fundadores de Internet, apuntaba que probablemente veríamos la propagación de los servicios de computación que, como actualmente los servicios de suministro eléctrico, llegarían hasta todas las casas y oficinas. Kleinrock ya asemejaba estos servicios computacionales a los servicios de suministro eléctrico, de ahí el término Grid, con el que se denomina en inglés la red eléctrica. Entre un Grid eléctrico y uno computacional, no obstante, existen algunas diferencias: mientras que la electricidad es producida más económicamente centralizada en una gran central, se caracteriza por una métrica muy simple y se accede a través de una interfaz de conexión estandarizada, los recursos computacionales son producidos más económicamente de forma distribuida con servidores o clusters, se caracterizan por una métrica altamente compleja y se acceden mediante interfaces de software en proceso de estandarización.

Tras el desarrollo inicial de Internet durante los últimos treinta años y la revolución que ha supuesto el World Wide Web, se desarrolla el concepto de Grid de cálculo con el fin de compartir y utilizar recursos de computación; este concepto se amplía con la utilización de recursos de almacenamiento masivo distribuido, dando lugar a Grid de datos. En 1995 durante el congreso SuperComputing ’95, la experiencia I-WAY demostró la posibilidad de ejecutar aplicaciones distribuidas de diferentes áreas científicas en una red de 17 centros de USA conectados con una red de alta velocidad (155 Mbps). Este fue el punto de partida de varios proyectos en diferentes áreas, con un denominador común: compartir recursos distribuidos de computación.

Años más tarde, en 1998, esta idea empezaba a materializarse con el inicio de la Globus Alliance, creada para organizar la investigación y el desarrollo para crear tecnologías básicas para el Grid y con el libro "The Grid: Blueprint for a Future Computing Infrastructure", editado por Ian Foster y Carl Kesselmann. En el prólogo, Foster y Kesselmann plantean la analogía con la "*electrical power grid*": el usuario debe tener acceso a los recursos computacionales en condiciones similares a las que tiene para utilizar la energía eléctrica: desde cualquier sitio (*pervasive*), con un interfaz uniforme (*consistent*), pudiendo confiar en su funcionamiento (*dependable*), y a un coste asequible (*inexpensive*).

El siguiente paso fue la extensión a recursos conectados a través de Internet: la red Entropía agregó en dos años 30.000 ordenadores, logrando por ejemplo calcular el mayor número primo conocido; del mismo modo, el sistema SETI funciona en más de medio millón de PCs analizando los datos del radio telescopio Arecibo para la búsqueda de señales de inteligencia extraterrestre. La computación en Física de Partículas es un ejemplo claro de esta evolución: los experimentos del anterior acelerador LEP del CERN (Centro Europeo de Física de Partículas, Ginebra) pasaron de usar ordenadores Cray e IBM a clusters de ordenadores en la primera parte de la década de los 90. Un ejemplo de las posibilidades de compartir estos recursos, lo proporcionó la simulación en sólo un fin de semana utilizando los recursos completos del CERN de más de cinco millones de colisiones e+e-, para mejorar los resultados de la búsqueda del bosón de Higgs del experimento DELPHI. La solución para los experimentos del próximo acelerador LHC, viene de la mano del problema: agregar y compartir los recursos proporcionados por las instituciones participantes en los experimentos del LHC distribuidas geográficamente por todo el mundo.

En el año 2000, entre los primeros proyectos Grid surge la Information Power Grid de la NASA, la iniciativa de la Nacional Science Foundation (NSF) con los centros de supercomputación de NCSA y SDC, y la Advanced Strategic Computing Initiative (DOE). También cabe citar los proyectos Grid científicos de PDG y GriPhyN (Física de Partículas), DOE ScienceGrid, Earth System Grid (meteorología), Fusion Collaboratory (fusión nuclear), NEESGRID (simulación para el estudio de los terremotos) así como un centro de soporte Grid del NSF, el International Virtual

Data Grid Laboratory (iVDGL) y el proyecto TeraGrid, que unirá cuatro centros USA de supercomputación a 40 Gbps, son dos de los mas relevantes.

En Europa, en el año 2000, el programa comunitario IST lanza el proyecto European DataGrid (EDG) coordinado por el CERN, con el objetivo de "construir la próxima generación de infraestructura de computación que permita cálculo intensivo y análisis de bases de datos compartidas a gran escala, desde cientos de Terabytes a Petabytes, entre comunidades científicas ampliamente distribuidas". El proyecto utiliza Globus como software básico, y desarrolla nuevo "*middleware*" para construir aplicaciones que manejan un gran volumen de datos, como las citadas de Física de Partículas (CERN, LHC), de Bioinformática, y de la ESA (Observación de la Tierra). Con un importe de 10M€, el proyecto cuenta como socios principales con entidades de investigación nacionales con instalaciones significativas de cálculo, como el INFN (Italia), PPARC (UK), CNRS (Francia) o NIKHEF (Holanda). Su interconexión esta basada en la nueva red Gigabit europea Gèant, en funcionamiento desde finales del año 2001. En paralelo, el programa IST aprueba dos proyectos orientados a la industria: DAMIEN (Distributed Applications and Middleware for Industrial Use of European Networks), y EUROGRID, basado en el sistema UNICORE.

En el año 2001 se lanzan varios proyectos IST en el área científica, como GridLab, EGSO, DataTag y CrossGrid; este último, cuyo objetivo es el desarrollo de aplicaciones interactivas en el entorno Grid, y la extensión del testbed del proyecto DataGrid. La configuración actual, definida por el proyecto DataGrid, esta basada en PCs con sistema operativo Linux: Computing Elements, Storage Elements, Resource Brokers, etc. En el testbed de producción los centros pueden aportar los recursos disponibles incrementando el número de Computing y Storage Elements.

En Europa destaca la iniciativa de e-Science en el Reino Unido. Este programa multidisciplinar, con una dotación de mas de 200M €, incluye la creación de un centro nacional de e-Science (NeSC, en Edimburgo) así como varios centros regionales, y apuesta por áreas de interés comercial como las bases de datos distribuidas en colaboración con empresas como Oracle o IBM.

En paralelo se apoyará y promoverá la participación ante las convocatorias del VI Programa Marco en esta área, y en particular de cara a proyectos de infraestructura Grid, de aplicaciones en e-Health o en Grids for Complex Problem Solving.

Entre las Expresiones de Interés (EoI) para el Programa Marco, la propuesta de un proyecto integrado denominado EGEE (Enabling Grids for e-Science and Industry in Europe) coordinada por el CERN ha contado con el respaldo de gran número de instituciones. Este proyecto proporciona una infraestructura común europea sobre la evolución de la red Gèant, y a su vez proporcionaría el marco para las redes de excelencia o proyectos orientados a las diferentes aplicaciones en *e-Ciencia*.

El próximo acelerador LHC, que entrará en funcionamiento en el año 2007, requerirá el almacenamiento y procesamiento de varios Petabytes de datos cada año. Los recursos necesarios se estiman en un orden de magnitud por encima de los mayores supercomputadores actuales, y con claras dificultades técnicas, operativas y de financiación.

El interés se ha extendido rápidamente a la aplicación de esta tecnología en el entorno de grandes compañías. IBM considera "*Grid Computing*" como la opción para integrar múltiples ordenadores distribuidos sobre una red (en particular Internet) de modo que actúen como un gran supercomputador. Esta perspectiva ofrece una línea complementaria de trabajo a la tecnología "*Web Services*" que está desembarcando en el área comercial de la mano de IBM y de Microsoft.

Gracias a la evolución de las tecnologías en las redes de comunicación (RedIRIS2, Gèant2), se creó la iniciativa IRISGrid, como la primera propuesta nacional para el desarrollo de esta tecnología, en la que participan numerosos grupos de investigación con experiencia en tecnología Grid. Esta iniciativa, reforzada tras la reciente acreditación de RedIRIS como autoridad de certificación EUGridPMA, propone un entorno colaborativo (*e-Ciencia*) entre los diferentes grupos de investigación con su posible participación en el VI Programa Marco.

1.3. Arquitectura Grid

La idea básica de la tecnología Grid es, aprovechar de modo óptimo los recursos proporcionados por equipos de computación distribuidos, conectados a una red de banda ancha, mediante el uso de un software adecuado para planificar su utilización ("*scheduler*" y "*resource broker*") que tenga en cuenta las prioridades y diferentes periodos de demanda de los usuarios. Debe incluir mecanismos de autentificación y autorización basados en certificados digitales, y contar con una gestión basada en una instalación automatizada, flexible y dinámica, acompañada de monitorización en tiempo real de todos los recursos (tanto de red como de equipos de cálculo).

Los servicios e interfaces de programación de aplicaciones y herramientas de desarrollo de software son clasificados de acuerdo con su función en la capacidad de compartir recursos; se establecen los requerimientos que deben satisfacer y se plantea la importancia de definir un conjunto concreto de protocolos "*intergrid*" para habilitar la interoperatibilidad a través de diferentes sistemas Grid. El establecimiento, gestión y explotación de recursos y transferencia de resultados entre diferentes VOs, requiere de una nueva tecnología. En términos de arquitectura Grid, se identifican los componentes fundamentales del sistema, se especifica el propósito y la función de estos componentes, y cómo interactúa un componente con el resto de componentes.

En un entorno de red, la interoperatibilidad significa protocolos comunes; por tanto, una arquitectura Grid es ante todo una arquitectura de protocolos, que definen los mecanismos básicos con los que los usuarios de VO y los recursos son administrados. Una arquitectura abierta, facilita la interoperatibilidad, extensión, portabilidad y compartición de código; los protocolos estándar facilitan la definición de servicios estándar que proporcionan capacidades mejoradas.

Para crear un Grid operativo, se construyen interfaces de programación de aplicaciones (API) y kits de desarrollo de software (SDK) para proporcionar las reglas de programación necesarias.

Esta tecnología y arquitectura constituye lo que se denomina en Grid "*middleware*" (servicios requeridos para soportar un conjunto común de aplicaciones en un entorno de red distribuido). El concepto de interoperatividad es fundamental, por la necesidad de transferir resultados entre VO, a través de diferentes plataformas, lenguajes y entornos de programación, políticas de actuación, y tipos de recursos.

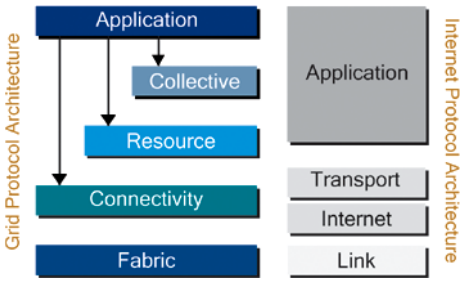


Fig 1.3. *Arquitectura Grid (Layers)*

Puesto que la arquitectura de protocolos de Internet se extiende desde la red a la aplicación, existe una correspondencia directa entre las capas (*layers*) de la arquitectura Grid y las de Internet. En la figura 2.2 se establece la relación entre la arquitectura Grid y la arquitectura de protocolos de Internet.

A continuación, se analizarán las capas (*layers*) de la arquitectura Grid, estableciendo en cada capa la correspondencia con la herramienta ***Globus Toolkit*** que será analizada con más detalle en el siguiente capítulo.

1.3.1. Fabric layer

Esta capa se corresponde con los interfaces de control local. Proporciona los recursos compartidos que se accederán a través de los protocolos Grid: recursos computacionales, sistemas de almacenamiento, catálogos, recursos de red, etc. Un recurso puede ser una entidad lógica, como un sistema de ficheros distribuidos, ordenador, cluster, o conjunto de ordenadores distribuidos; en tales casos, una implementación de recursos, puede involucrar protocolos internos que no son propios de la arquitectura Grid.

La experiencia sugiere que como mínimo, los recursos deberían implementar por un lado, mecanismos de consulta que permitan obtener su estructura, estado y capacidades, y por otro, mecanismos de gestión de recursos que proporcionen algún control de la calidad del servicio proporcionada.

- ***Recursos computacionales***

  Mecanismos requeridos para ejecutar programas y para monitorizar y controlar la ejecución de los procesos resultantes. Suponen mecanismos avanzados de reserva de recursos. Se necesitan funciones de consulta para determinar las características hardware y software así como información relevante del estado actual de carga y estado de las colas de procesamiento en el caso de recursos gestionados por colas.

- ***Recursos de almacenamiento***

  Mecanismos requeridos para transferir ficheros, que permitan el control de los recursos para la transferencia de datos (espacio, ancho de banda de los discos, ancho de banda de la red, CPU) así como los mecanismos avanzados de reserva de recursos. Las funciones de consulta se requieren para determinar las características hardware y software así como información relevante de carga como espacio disponible y utilización de ancho de banda.

- ***Recursos de red***

  Mecanismos de gestión que proporcionen control sobre los recursos localizados para la transferencia de datos en la red (priorización, reserva). Se deberían proporcionar funciones de consulta para determinar las características de la red y la carga de la red.

- ***Colecciones de Código***

  Esta forma especializada de almacenamiento de recursos, requiere mecanismos para la gestión de versiones de código fuente y código objeto. Sistema de control como CVS.

- ***Catálogos***

  Esta forma especializada de almacenamiento de recursos, requiere mecanismos para la implementación de operaciones de consulta y actualización de catálogos. Por ejemplo, una base de datos relacional.

Globus Toolkit ha sido diseñado principalmente para usar los componentes existentes en el Fabric layer, incluyendo protocolos e interfaces de otros fabricantes. En este sentido, incluirá los mecanismos necesarios que no estén proporcionados por el fabricante para proporcionar la funcionalidad de los recursos.

1.3.2. Connectivity layer

Define los protocolos de comunicación y autentificación necesarios para las transacciones de red específicas del Grid (comunicación de forma fácil y segura). Los protocolos de comunicación permiten el intercambio de datos entre los recursos del *Fabric layer*. Los protocolos de autentificación, construidos sobre los servicios de comunicación proporcionan mecanismos de seguridad encriptada para verificar la identidad de usuarios y recursos. Los requerimientos de comunicación, incluyen el transporte, enrutado y resolución de nombres. Consideramos que estos mecanismos están incluidos en el protocolo TCP/IP, en concreto las capas Internet (IP y ICMP), transporte (TCP, UDP) y aplicaciones (DNS, OSPF, RSVP, etc.). Esto no quiere decir que en un futuro, las comunicaciones Grid no demanden nuevos protocolos de red.

La clave pública basada en los protocolos de infraestructura de seguridad del Grid (GSI) se utiliza para la autentificación, protección de la comunicación, y autorización. GSI es una extensión de los protocolos de la capa de transporte de seguridad (TLS). El control de la autorización está soportado a través de una herramienta de autorización que permite integrar recursos a través del interfaz de control de acceso y autorización genérica (GAA). Con respecto a los aspectos de seguridad de *Connectivity layer*, al igual que con los protocolos de comunicación, cualquier estándar de seguridad desarrollado dentro del contexto del protocolo Internet es aplicable. Las soluciones de autentificación para entornos de VO deberían tener las siguientes características:

- Delegación de un conjunto de derechos de acceso de un recurso a otro.
- Integración con varias soluciones de seguridad locales.
- Entorno de seguridad común para el uso de recursos en otros Grid.
- Única autentificación para el acceso a todos los recursos Grid definidos en la capa *Fabric layer*.

Respecto a ***Globus Toolkit***, los protocolos Internet se utilizan para las comunicaciones. Las soluciones de seguridad Grid, también deberían proporcionar un soporte flexible para la protección de la comunicación y habilitar el control sobre las decisiones de autorización, incluyendo la posibilidad de restringir la delegación de derechos.

1.3.3. Resource layer

Define sobre los protocolos de comunicación y autentificación de la *Connectivity layer*, protocolos, interfaces de programación de aplicaciones (APIs) y herramientas de desarrollo de software (SDKs) para la negociación segura, iniciación, monitorización, control, registro de acceso y coste de las operaciones de compartir recursos individuales. En la implementación de estos protocolos, se realizan peticiones a las funciones de *Fabric layer* para acceder y tener control local de los recursos. Los protocolos de *Resource layer* se dedican exclusivamente a recursos individuales y, por tanto, ignoran cualquier petición de estado global y acciones dirigidas a colecciones distribuidas de recursos. Estas peticiones las gestionará *Collective layer*. Podemos distinguir dos clases primarias de protocolos:

- ***Protocolos de Información***
  Se utilizan para obtener información sobre la estructura y el estado de un recurso, (configuración, carga actual, política de uso, etc.)

- ***Protocolos de Gestión***
  Se utilizan para negociar el acceso a un recurso compartido, especificando requerimientos de recursos (incluyendo reserva avanzada y calidad de servicio) y la operación u operaciones a ser realizadas, tales como creación de procesos o acceso a datos.

Puesto que los protocolos de gestión son los responsables de iniciar las relaciones de compartir recursos, deben proporcionar un "punto de referencia en la política de aplicación", asegurando que las operaciones de protocolo requeridas son consistentes con la política del recurso a ser compartido.

Otras cuestiones que deben ser consideradas, incluyen el registro de accesos y el coste. Un protocolo de gestión, debe también soportar la monitorización del estado del recurso, de una operación y del control de dicha operación (p.e. terminación). Estos protocolos deben ser implementados para obtener los mecanismos fundamentales de compartir recursos a través de diferentes sistemas de gestión de recursos locales. La funcionalidad proporcionada por *Fabric layer*, resume las características principales requeridas por los protocolos del *Resource layer*. Se añade además la necesidad de reportar de forma fiable los errores, indicando cuando falla una determinada operación. Respecto a ***Globus Toolkit*** se adopta un pequeño conjunto de protocolos estándar:

- ***Lightweight Directory Access Protocol*** (LDAP)
  Protocolo de acceso a catálogos.

- ***Grid Resource Information Protocol*** (GRIP)
  Actualmente basado en LDAP, es utilizado para definir un protocolo estándar de información del recurso y el modelo de información asociado. Un protocolo asociado de registro de software del recurso, Grid Resource Registration Protocol (GRRP), utilizado para registrar recursos con Grid Index Information Servers (GIIS).

- ***Grid Resource Access and Management*** (GRAM)
  Protocolo basado en HTTP, utilizado para localizar recursos computacionales y para monitorizar y realizar un control computacional en dichos recursos.

- ***GridFTP***
  Versión extendida de File Transfer Protocol (FTP), es un protocolo de gestión para acceso a datos. Incluye el uso de protocolos de seguridad de *Connectivity layer*, acceso parcial a ficheros, y gestión de paralelización para transferencias de alta velocidad.

Globus Toolkit define para cada uno de estos protocolos para el usuario, los interfaces de programación de aplicaciones (APIs) y herramientas de desarrollo de software (SDKs) para los lenguajes C y Java.

Para los servidores (SDKs) también proporciona para cada protocolo, la facilidad de integración de varios recursos (computacionales, almacenamiento, red) dentro del Grid. Por ejemplo, GRIS implementa para el servidor la funcionalidad LDAP, con peticiones que permiten la publicación de información de recursos arbitrarios.

Un elemento importante de ***Globus Toolkit*** es el "*gatekeeper*", que proporciona a través del protocolo GRAM la planificación de varias operaciones locales, autentificadas con el protocolo GSI. El protocolo Generic Security Services (GSS) API se utiliza para adquirir, remitir, y verificar las credenciales de autentificación y para proporciona la integridad y privacidad en la capa de transporte dentro de SDKs, habilitando la substitución de servicios alternativos de seguridad de *Connectivity layer*.

1.3.4. Collective layer

Esta capa se encarga de la coordinación de múltiples recursos. Mientras que *Resource layer* se centra en las interacciones con recursos simples, la siguiente capa en la arquitectura Grid contiene los protocolos y servicios (y APIs y SDKs) que no están asociados con recursos específicos pero que registran las interacciones a través de colecciones de recursos: *Collective layer*.

Puesto que los componentes de *Collective layer* se construyen a partir de *Resource* y *Connectivity layer*, se pueden implementar una amplia variedad de servicios de compartición de recursos sin la necesidad de establecer nuevos requerimientos en los recursos que se van a compartir:

- ***Servicios de directorio***
  Permiten a los participantes de VO, descubrir la existencia y/o propiedades de recursos de una VO. Un servicio de directorio, permite a los usuarios realizar consultas de recursos por nombre y/o por atributos como tipo, disponibilidad, o carga. Los protocolos del nivel de recursos GRRP y GRIP se utilizan para la construcción de directorios.

- ***Servicios y agentes de localización y planificación***
  Permiten a los participantes de VO realizar peticiones de localización de uno o más recursos para un determinado propósito y las tareas de planificación en los recursos apropiados. Por ejemplo, AppLeS, Condor-G, Nimrod-G, y agente DRM.

- ***Servicios de replicación de datos***
  Dan soporte de gestión de recursos de almacenamiento de VO para maximizar el rendimiento en el acceso a los datos con respecto a métricas como tiempo de respuesta, fiabilidad y coste.

- ***Servicios de monitorización y diagnóstico***
  Permiten la monitorización de recursos de una VO por fallo, detección de intrusos, sobrecarga, etc.

- ***Sistemas de programación Grid***
  Permiten habilitar modelos de programación que se puedan integrar en entornos Grid, usando servicios Grid para la detección de recursos, seguridad, localización de recursos, y otros conceptos. Por ejemplo, Message Passing Interface (MPI).

- ***Gestión de carga de trabajo y colaboración de entornos de trabajo***
  También conocido como el problema de resolución de entornos (PSEs), proporcionado para la descripción, uso y gestión de flujos asíncronos de trabajo con múltiples componentes.

- ***Servicios de detección de Software***
  Descubren y seleccionan la mejor implementación software y plataforma de ejecución, basándose en los parámetros del problema a resolver. Por ejemplo NetSolve y Ninf.

- ***Servidores de autorización de VO***
  Este servicio asegura que se cumplen las políticas comunitarias que gobiernan el acceso a los recursos, generando capacidades que los miembros de la comunidad

pueden utilizar para el acceso de los recursos VO. Estos servidores proporcionan un servicio global para el cumplimiento de la política de acceso a los recursos en base a la información de recursos, y los protocolos de gestión de recursos dentro de *Resource layer*, y protocolos de seguridad dentro de *Connectivity layer*.

- ***Servicios de contabilidad y pago VO***
  Reúnen la información de utilización del recurso con el objeto de contabilizar, establecer la cuota, y/o limitar el uso del recurso por los miembros de la comunidad.

- ***Servicios de colaboración***
  Permiten el intercambio coordinado de información dentro de grandes comunidades de usuarios, bien de forma síncrona o de forma asíncrona. Por ejemplo, CAVERNsoft, Access Grid.

Los protocolos, APIs, SDKs y servicios de Collective y Resource layer, pueden combinarse de diferentes formas para proporcionar funcionalidad a las aplicaciones. Mientras que los protocolos de *Resource layer* deben ser en general ampliamente desarrollados, los protocolos *Collective layer* entran dentro de un propósito de aplicación o dominio más específico, incluso dentro de una determinada VO.

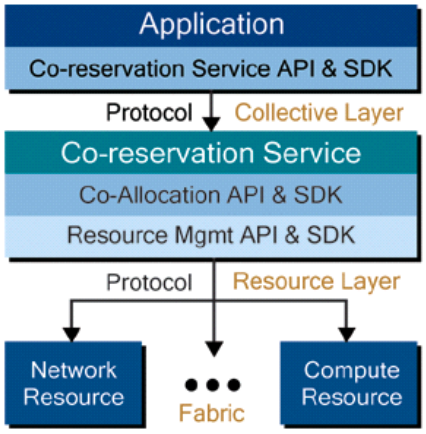


Fig 1.3.4. *Arquitectura Grid (Protocolos)*

Las funciones dentro de *Collective layer* pueden ser implementadas como servicios permanentes con protocolos asociados, o como SDKs (con APIs asociados) diseñados para enlazar con aplicaciones. En ambos casos, su implementación puede realizarse en base a protocolos y APIs de *Resource layer*. Los componentes de *Collective layer*, se pueden dirigir a los requerimientos de una comunidad específica de usuarios VO, o dominio de aplicación.

Por ejemplo, un SDK que implemente un protocolo coherente para una aplicación específica, o un servicio de reserva para un conjunto específico de recursos de red. Otros componentes de *Collective layer* pueden ser de un propósito más general, por ejemplo, un servicio de replicación que gestione una colección internacional de sistemas de almacenamiento para múltiples comunidades, o un servicio de directorio diseñado para habilitar el descubrimiento de VOs. En este sentido, lo más importante es que todos los componentes de los protocolos e interfaces de programación de aplicaciones (APIs) de *Collective layer* estén basados en estándares.

En ***Globus Toolkit***, además de los ejemplos de servicios mencionados anteriormente en esta sección, el Meta Directory Service (MDS) introduce el concepto de Grid Information Index Servers (GIISs) para soportar consultas arbitrarias en subconjuntos de recursos, con la información del protocolo LDAP utilizada para acceder a recursos específicos, GRISs para obtener el estado de los recursos y GRRP utilizado para el registro de recursos. También la replicación de catálogos y los servicios de gestión de réplica se utilizan para soportar la gestión de réplicas de conjuntos de datos en entorno Grid.

1.3.5. Applications layer

La última capa de la arquitectura Grid, está formada por las aplicaciones de usuario que operan dentro de un entorno de VO. Las aplicaciones están construidas en términos de servicios definidos en cualquier capa de la arquitectura Grid.

En cada capa, los protocolos formalmente definidos y establecidos, proporcionan el acceso a servicios útiles de gestión de recursos, acceso a datos, detección de recursos, y mucho más. En cada capa, los interfaces de programación de aplicaciones (APIs) también pueden estar implementados a partir de protocolos de intercambio de mensajes con los servicios apropiados para realizar las acciones deseadas. Los APIs se implementan con herramientas de desarrollo software (SDKs), que utilizan protocolos para interactuar con servicios de red que proporcionan las capacidades de operación al usuario final.

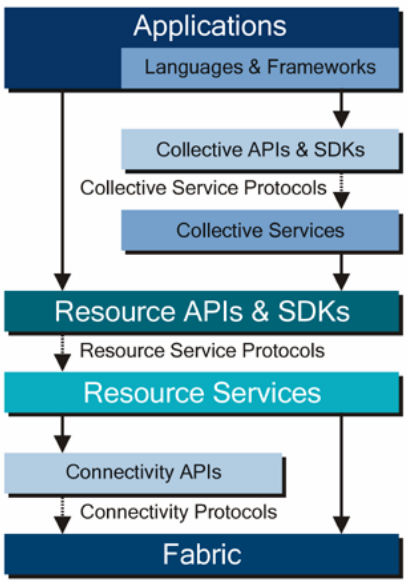


Fig 1.3.5. *Arquitectura Grid (APIs y SDKs)*

SDKs de más alto nivel, pueden proporcionar funcionalidad que no se corresponde con ningún protocolo específico, pero que combina operaciones de protocolos con referencias a APIs para implementar funcionalidad local. Las líneas sólidas representan llamadas directas y las líneas punteadas representan interacción con protocolos. En la práctica, el concepto "aplicación" puede aplicarse dentro de sofisticados entornos de trabajo y librerías (Common Component Arquitecture, SciRun, CORBA, Cactus).

Estos entornos de trabajo, pueden a su vez definir protocolos, servicios y/o APIs (p.e. Simple Workflow Access Protocol). Por tanto, el ámbito de este concepto está más allá del definido en *Applications layer*, que sólo describe la mayoría de los servicios y protocolos requeridos en la arquitectura Grid.

1.4. Protocolos InterGrid

La arquitectura Grid establece los requisitos que deben cumplir los protocolos y APIs para permitir compartir recursos, servicios y código. En ningún caso, obliga a establecer las normas que pueden utilizarse para implementar estos protocolos y APIs. Por ejemplo, podemos construir protocolos basados en Kerberos y protocolos basados en PKI en *Connectivity Layer* y acceder a estos mecanismos de seguridad a través del mismo API, gracias a GSS. De esta forma, no se puede interoperar con Grids construidos con diferentes protocolos, ya que no se pueden compartir servicios esenciales. Por esta razón, el término de computación Grid requiere que se seleccionen un conjunto de protocolos bastante difundidos en *Connectivity* y *Resource layers*, y un poco menos difundidos, en *Collective layer*. Así como los protocolos Internet permiten que diferentes redes de ordenadores puedan interconectarse e intercambiar información, los protocolos InterGrid permitirán

interconectar diferentes organizaciones para intercambiar información y compartir recursos. Los APIs estándar también son extremadamente útiles si el código puede compartirse entre las diferentes organizaciones. En la actualidad, Globus Toolkit representa la mejor aproximación para la identificación de estos APIs y protocolos InterGrid.

1.5. Aplicaciones de computación Grid

Una metacomputadora denota una red virtual de supercomputadores, construida dinámicamente a través de recursos distribuidos geográficamente y conectados a través de redes de alta velocidad. La existencia de estas redes virtuales de computación puede proveer del potencial necesario para enfrentarse a la resolución de nuevos y complejos problemas. La computación Grid, como la mayoría de las aplicaciones de computación distribuida, está motivada por la necesidad de acceder a recursos no localizados dentro de la propia computadora, generalmente por motivos económicos (las supercomputadoras son excesivamente caras), o por la propia naturaleza del entorno del trabajo, como por ejemplo en ingeniería donde es necesario conectar sistemas de realidad virtual, con bases de datos y supercomputadoras.

La computación Grid tiene muchos puntos comunes tanto con sistemas distribuidos como paralelos, pero también difieren de estas arquitecturas en varios aspectos. Como en los sistemas distribuidos, debe integrar dispositivos de muchos y diferentes tipos conectados por redes (no siempre fiables) y a menudo localizados en diferentes dominios administrativos. Sin embargo, las necesidades de las aplicaciones pueden requerir de modelos e interfaces radicalmente diferentes de los usados en sistemas distribuidos. Por otro lado, al igual que en un sistema paralelo, las aplicaciones de computación Grid necesitan de una gestión de las comunicaciones entre los distintos dispositivos. Pero, por el contrario la naturaleza heterogénea y dinámica de los sistemas de computación Grid limita la aplicación de las actuales herramientas y técnicas de paralelización. Estas consideraciones muestran que mientras la computación Grid puede construirse sobre tecnologías tanto distribuidas como paralelas, necesita además de importantes avances en mecanismos de comunicación,

técnicas y herramientas. Tanto científicos como ingenieros sólo acaban de comenzar a explorar las nuevas aplicaciones desarrolladas para redes de supercomputación. El experimento I-WAY considera cuatro tipos de aplicaciones:

- ***De escritorio*** (*Desktop supercomputing*)
  Estas aplicaciones conectan la capacidad de visualización de datos con supercomputadoras y /o bases de datos remotas, lo que permite una mejor explotación por parte de los usuarios de los recursos computacionales, al mismo tiempo que consigue una independencia de la distancia entre recursos, desarrolladores y usuarios.

- ***De instrumentación*** (*Smart instrument*)
  Aplicaciones que permiten la conexión entre instrumentos que pueden ir desde microscopios hasta satélites, y supercomputadoras remotas. Esta relación puede permitir procesado e interacción de datos en prácticamente tiempo real.

- ***Entornos colaborativos***
  Este tercer tipo de aplicaciones acopla múltiples entornos virtuales para que usuarios en diferentes localizaciones geográficas puedan interactuar con recursos de otras organizaciones.

- ***Supercomputación distribuida***
  Las aplicaciones encargadas de acoplar múltiples computadoras para que resuelvan un mismo problema que escapa de la capacidad de una sola máquina, así como diferentes componentes de un mismo problema en diferentes arquitecturas. Se pueden distinguir dos modos de operación: *programados* o *no programados*. En el modo *programado*, los recursos, una vez adquiridos se dedican a esa aplicación. Por el contrario en el modo *no programado*, se lanza la aplicación y esta misma utiliza recursos libres según vaya necesitando. Condor es un sistema que soporta este último modo. En general, el modo programado se utiliza para simulaciones que necesitan de un estrecho acoplamiento entre ellas, mientras que las no programadas es preferible para aplicaciones mas independientes entre ellas y que pueden adaptarse a recursos que varíen con el tiempo.

1.6. Características de un sistema de computación Grid

Los testbeds realizados hasta la fecha han permitido atisbar cuales van a ser las principales características de un sistema de computación Grid:

- ***Escalabilidad y selección***
  La idea es que las redes virtuales estén constituidas por un gran número de recursos (computadores, disco, etc.), y que puedan seleccionarse cuales y cuantos de ellos se van a encargar de una determinada aplicación o tipo de aplicación en función de criterios como conectividad, costo, seguridad y estabilidad.

- ***Heterogeneidad a múltiples niveles***
  Los miembros de una red virtual serán tremendamente heterogéneos, desde los dispositivos físicos hasta el software instalado, del gestor de colas y las políticas de uso.

- ***Estructura impredecible***
  Las aplicaciones de computación Grid se ejecutaran en un muy variable rango de entornos, construidos dinámicamente a partir de los recursos disponibles. La distribución geográfica y la complejidad son otros factores que dificultaran la determinación a priori de características del sistema tales como el ancho de banda y la latencia.

- ***Comportamiento dinámico y de comportamiento impredecible***
  En un entrono de computación Grid, los recursos, y especialmente la red, estarán muy probablemente compartidos. Como consecuencia de esto el comportamiento y el rendimiento del sistema puede variar con el tiempo, que pueden llegar a ser dramáticos, como en el caso de caídas de red y recursos. En general, no es posible garantizar unos mínimos que aseguren la calidad del servicio.

- ***Múltiples dominios administrativos***
  Los recursos utilizados en un sistema de computación Grid pertenecerán, en general, a diferentes dominios administrativos. Este hecho complica el ya por si complejo problema de la seguridad, ya que cada dominio utilizará sus propios sistemas de autentificación y políticas de acceso.

La principal conclusión que se puede obtener de los puntos reseñados anteriormente es la de la necesidad de contar con mecanismos que permitan a las aplicaciones obtener información en tiempo real sobre el estado y la estructura del sistema en el que se están ejecutando y usar esta información para tomar decisiones en la configuración. Esta información puede incluir desde actividad de red, disponibilidad de interfaces de red, características el procesador y mecanismos de autentificación.

1.7. Proyecto Globus

El Proyecto Globus es un esfuerzo multi-institucional de investigación y desarrollo que se dedica a la creación de tecnologías básicas para Grids de computación. La Open Grid Services Architecture (OGSA) es una evolución propuesta del actual Globus Toolkit hacia una arquitectura de sistemas Grid basada en una integración de servicios, conceptos y tecnologías Grid. Además de la información disponible a través de páginas Web, los ingenieros e investigadores necesitan un acceso adecuado a elevados recursos de computación, tanto de cálculo como de información almacenada en grandes bases de datos. Su interés parte de la propia filosofía no solo de compartir recursos distribuidos, y por tanto optimizar los mismos, sino de garantizar además un acceso transparente de los mismos a los usuarios. El software intermedio (*middleware*), entre la infraestructura básica y las aplicaciones, es el encargado de lograr este objetivo, y por lo tanto una componente fundamental del Grid. En esta dirección, el proyecto Globus ha sido el primer foco de atención y base de los primeros proyectos Grid que aparecieron en Estados Unidos.

En una primera fase, se ha desarrollado un "*toolkit*" para la infraestructura de computación Grid que provee de las capacidades e interfaces básicos en áreas tales como comunicación, información, localización y gestión de recursos, autentificación y acceso a datos. Junto con dicho toolkit se ha desarrollado un "*metacomputing abstracts machine*" en la cual pueden construirse un rango de infraestructuras, servicios y aplicaciones alternativas. Además se está construyendo herramientas para programación paralela, de localización de dispositivos y de gestión de servicios. El objetivo a largo plazo de Globus es resolver los problemas de configuración y optimización de los entornos de computación Grid.

Estos retos son bastante ambiciosos principalmente debido a la complejidad inherente en los sistemas de computación Grid, a que muchos recursos solo se identifican durante el proceso de ejecución, y a la propia naturaleza de las características de dichos recursos. Se está investigando el diseño de servicios que convivan en las capas de más alto nivel de Globus y que permitan la construcción de aplicación adaptativas; estos servicios conformarán la llamada "*Adaptative Wide Area Resource Environment*", o AWARE.

Existe toda una serie de hitos pioneros en el campo de la computación Grid: desde los paquetes PVM (Parallel Virtual Machine) y MPI (Message Passing Interface) que proveen de una capa de comunicación independiente de la arquitectura, hasta Condor que permite obtener una vista uniforme de un conjunto de procesadores, pasando por AFS que logra lo mismo con un conjunto de discos. Cada uno de estos sistemas ha resultado exitoso en su aplicación en sistemas de gran escala. Globus no permite competir con estos, sino ofrecer una infraestructura básica que pueda ser utilizada para la construcción de implementaciones de alto rendimiento y portabilidad en todo un rango de este tipo de servicios.

Básicamente se pretende desarrollar mecanismos de bajo nivel que puedan utilizarse para implementar servicios de alto nivel y construir técnicas que permitan a su vez que estos servicios puedan observar y guiar a los mecanismos de bajo nivel. Esto puede reducir la complejidad y mejorar la calidad del software de computación Grid introduciendo una única infraestructura de bajo nivel que pueda utilizarse con muchos y variados propósitos, y resolviendo muchos de los problemas de configuración de la actual computación Grid. En este sentido la librería de comunicación Nexus ha sido ya utilizada para construir implementaciones de alto rendimiento de diversos interfaces de programación paralela.

Se ha mencionado ya el experimento I-WAY, cuyos componentes incluyen la librería de comunicación Nexus, autentificación basada en Kerberos, mecanismos de creación de procesos, y una gestión centralizada de los recursos computacionales. Salvo en algunos aspectos (p.e. no escalabilidad, no gestión de recursos de red, etc.), el experimento demostró las ventajas de Globus como proveedor de mecanismos

básicos. Las aplicaciones I-WAY fueron desarrolladas con una gran variedad de herramientas de paralelización (MPI, CC++, CAVEcomm, etc.), que fueron portadas al entorno I-WAY adaptando estas herramientas a los mecanismos de Globus de autentificación, creación de procesos, y comunicación. Otro testbed es el Globus Ubiquitous Supercomputing Testbed (GUSTO), cuyo objetivo principal es el desarrollo y evaluación de mecanismos básicos de autentificación, gestión, comunicación e información. Para ello se ha conformado una comunidad de prueba formada por quince centros y con recursos de computación que van desde IBM, SG, etc., y redes ethernet y ATM OC3. La autentificación se basa en el método explicado en la sección anterior, y el servicio de información es ofrecido por un servidor dedicado que mantiene la información sobre la configuración de los recursos y las características de la red. Esta información se actualiza dinámicamente para tener una visión en tiempo casi real del estado del sistema.

El proyecto Globus aborda el problema de la computación Grid desde abajo, desarrollando mecanismos básicos que pueden utilizarse para implementar una gran variedad de servicios de alto nivel. Comunicación, localización de recursos, información, autentificación, acceso a datos, y otros servicios están actualmente desarrollándose. A su vez, este proyecto también considera el problema de la configuración en un sistema de computación Grid, con el objetivo principal de crear un *Adaptative Wide Area Resource Environment* (AWARE) que soporte la construcción de servicios y aplicaciones adaptativas a un entorno tan dinámico como el de la computación Grid. Para esto se han introducido en Globus los mecanismos de selección, información y notificación cuyo objetivo es servir de herramientas para la consecución de una configuración dinámica.

Resumiendo, el proyecto Globus ha logrado importantes avances en la definición de un núcleo para la arquitectura de un sistema de computación Grid a partir del cual pueden construirse una gran variedad de entornos de computación Grid, en el desarrollo de un framework que permite que las aplicaciones respondan al comportamiento dinámico subyacente en cualquier entorno de computación Grid y en la demostración en testbeds como I-WAY y GUSTO de que se pueden construir herramientas de alto nivel basados en los interfaces incluidos en el toolkit Globus.

2

Globus toolkit

2.1. Componentes

Globus toolkit, un proyecto "*open-source*" desarrollado por el equipo del Argonne Nacional Laboratory dirigido por Ian Foster en colaboración con el grupo de Carl Kesselman en la University of Southern California, incorpora los protocolos y servicios básicos necesarios para construir aplicaciones Grid. Básicamente consiste en un conjunto de módulos; cada módulo define un interfaz que es utilizado por los servicios de alto nivel para invocar los mecanismos propios de dicho módulo y proveer lo que se denomina una implementación, la cual, a su vez, utiliza las adecuadas operaciones de bajo nivel para implementar estos mecanismos en diferentes entornos.

Se van a describir con más detalle los servicios Globus. En cada caso se vera como los componentes establecen las correspondencia de diferentes implementaciones en diferentes servicios de alto nivel y como apoyo al desarrollo de servicios y aplicaciones del tipo AWARE. Se utiliza el termino "*Adaptative Wide Area Resource Environment*" (AWARE) para denotar un conjunto de interfaces de aplicación, servicios de alto nivel, y políticas de adaptación cuyo objetivo es explotar el entorno de computación Grid lo más eficientemente posible. Actualmente se están desarrollando componentes AWARE para varias aplicaciones. En conjunto, todos los módulos del Globus toolkit pueden considerarse como una máquina virtual para computación Grid.

2.1.1. Localización y asignación de recursos

Este componente ofrece los mecanismos necesarios para que una aplicación muestre sus requerimientos, identificar que recursos cumplen estos requerimientos y gestionarlos una vez han sido asignados para dicha aplicación.

Estos mecanismos de localización de recursos son necesarios porque en general las aplicaciones no tienen porque conocer la exacta localización de cada recurso, máxime cuando esta la localización de recursos disponibles puede cambiar continuamente. La asignación del recurso implica inicializar y gestionar dicho recurso para la interacción con la aplicación. La arquitectura de gestión de recursos (GRAM — Globus Resource Allocation Manager) de Globus permite el acceso transparente, unificado y seguro a los distintos gestores de recursos locales de cada organización virtual (PBS, Condor, LSF, SGE,…).

Los principales componentes de esta arquitectura son el lenguaje de especificación de recursos (RSL), el gestor de asignación de recursos (GRAM) y la asignación múltiple de recursos (DUROC).

El acceso a cada recurso se controla mediante una serie de asignaciones entre el *subject* (DN) del certificado de un usuario del Grid que puede usar el recurso, y un usuario local. Cada una de las organizaciones virtuales, atendiendo a su propia política de administración, deberá decidir qué usuarios del Grid tienen acceso a cada recurso, y si estos se asignan al mismo usuario local, o por el contrario si se crea una cuenta local distinta para cada usuario del Grid.

2.1.2. Comunicaciones

Este componente se encarga de ofrecer los servicios básicos de comunicación, permitiendo la implementación eficiente de un gran rango de métodos de comunicación, incluyendo paso de mensajes, llamadas a procedimientos remotos, memoria distribuida, multicast, etc. Los mecanismos deben ser conscientes de la calidad de servicio de la red (latencia, ancho de banda, etc.). El módulo de comunicación de Globus está basado en la librería de comunicación Nexus. Nexus define cinco conceptos básicos: nodos, contexto, hebra, links y peticiones de servicio remotas. Las funciones de Nexus que manipulan estos conceptos constituyen el interfaz de comunicación de Globus. El interfaz es utilizado ampliamente por otros módulos de Globus y también han sido utilizados para construir varios servicios de alto nivel, incluyendo herramientas de programación paralela.

Nexus une un punto de comunicación inicial (*startpoint*) con un punto final (*endpoint*) estableciendo un *link* de comunicación. Si se establecen múltiples puntos iniciales contra un mismo punto final, las comunicaciones entrantes son interpoladas de la misma manera que se hace con los mensajes cuando son enviados a un mismo nodo en un sistema de paso de mensajes. Si es al revés, y un mismo punto inicial se enfrenta a varios puntos finales la comunicación se opera en modo *multicast*.

Un punto inicial puede ser copiado entre procesadores, causando nuevos *links* de comunicación replicas de los links del punto inicial original. Esta capacidad de "copiado" implica que los puntos iniciales pueden ser utilizados como "nombres globales" para objetos, y por tanto puede ser utilizados en cualquier punto de un sistema distribuido.

Un *link* de comunicación soporta una única operación de comunicación: una petición asíncrona de servicio remoto (RSR – *remote service request*). Un RSR es aplicado a un punto inicial asignándole un nombre de proceso y *buffer* de datos. Para cada punto final *linkado* a este punto inicial, el RSR transfiere el *buffer* de datos al espacio de direcciones en el que se localiza dicho punto final e invoca remotamente el proceso especificado, pasando el punto final y el *buffer* como argumentos. Una dirección local puede estar asociada a un punto final, en cuyo caso los puntos iniciales asociados a dicho punto final pueden ser tratados como "punteros globales" a esa dirección.

El interfaz Nexus y la implementación de selección basada en reglas de los métodos – como el protocolo, el método de compresión, y la calidad de servicio – son utilizados para ejecutar la comunicación. Diferentes métodos de comunicación pueden asociarse con diferentes *links* de comunicación, en los cuales las reglas de selección determinan que método debería ser utilizado cuando un nuevo *link* se establece. Estos mecanismos han sido diseñados para soportar numerosos protocolos de comunicación y un uso selectivo de comunicación segura en entornos heterogéneos.

2.1.3. Servicios de información de recursos

Es un mecanismo uniforme para la obtención en tiempo real de información sobre la estructura y el estado del sistema Grid. Este mecanismo debe permitir al resto de componentes tanto dar como recibir información. Los entornos de computación Grid dependen críticamente de cómo se accede a la información sobre el sistema de red de supercomputación subyacente.

Entre la información requerida puede incluirse la siguiente: detalles de configuración sobre recursos tales como la cantidad de memoria, la velocidad de CPU, el numero de nodos en el caso de un cluster, o el número y tipo de interfaces de red; información de ejecución instantánea, como latencia de red, el ancho de banda disponible, y la carga de CPU; información específica para la aplicación, como requerimientos de memoria o estructuras de programación que se han encontrado efectivas en ejecuciones previas.

Diferentes datos tendrán diferentes objetivos y diferentes requerimientos de seguridad, pero alguna información puede ser al menos potencialmente ser necesitada de manera global por cualquier componente del sistema. La información puede ser obtenida de diferentes fuentes: por ejemplo, información de servicios estándares como puedan ser NIS, SNMP, etc; o de servicios más especializados que puede ir desde servicios de tiempo meteorológico; o de servicios exteriores como otra aplicación. Globus define un sencillo y unificado mecanismo de acceso para este gran rango de información, llamado *Servicio de directorio (MDS — Metacomputing Directory Service)*.

Basado en LDAP, MDS define un *framework* en el cual puede representarse información de interés en aplicaciones de computación distribuida. La información es estructurada como un conjunto de entradas, donde cada entrada incluye o bien ninguna o bien varios pares de atributos. El tipo de una entrada, llamado su *clase objeto*, especifica que atributos son obligatorios y cuales opcionales. La información MDS puede mantenerse en un servidor(es) LDAP. Sin embargo, los requerimientos de rendimiento y funcionalidad de las aplicaciones de computación Grid motivan un

número de extensiones. Un mecanismo de proxy soporta la integración de otros servicios de datos, como SNMP o NIS, además de permitir acceso remoto. Actualmente se está investigando conseguir mecanismos de caché y replicación, y en la definición de un interfaz de notificación que permitirá a los servicios y aplicaciones de más alto nivel que requieran notificaciones cuando determinadas condiciones se cumplan.

2.1.4. Creación de procesos

Este componente es usado para iniciar la computación en un recurso una vez ha sido localizado y asignado. Este task incluye configuración de ejecutables, creación del entorno de ejecución, comienzo del proceso, paso de argumentos, integración del proceso dentro del resto de la computación, y gestión y terminación del mismo.

2.1.5. Interfaz de autentificación

Provee de los mecanismos básicos de autentificación que se utilizan para validar tanto usuarios como recursos. Estos mecanismos son los ladrillos básicos sobre los que asentar métodos de seguridad de mayor nivel. Uno de los componentes clave de Globus es el protocolo *Grid Security Infrastructure* (GSI), que permite una autenticación única del usuario para todos los recursos mediante certificación digital basada en PKI y X.509. La versión inicial del módulo de autentificación de Globus soporta password, rsh y SSL. Con el objeto de aumentar el grado de abstracción del interfaz, se esta dirigiendo hacia el uso de *Generic Security System* (GSS).

GSS define un procedimiento estándar y un API para la obtención de credenciales (cliente y servidor), y para encriptación y desencriptación orientada a mensajes. GSS es independiente de cualquier mecanismo de seguridad particular y puede residir en capas superiores de diferentes métodos de seguridad, como Kerberos y SSL. GSS debe modificarse y extenderse para cumplir los requerimientos propios de la computación Grid. En general se sistema de computación Grid puede utilizar diferentes mecanismos de autentificación en diferentes situaciones y para diferentes propósitos, por tanto debe implementarse en GSS soporte para un variado rango de diferentes mecanismos de seguridad.

Además, funciones de búsqueda y selección son necesarias para que los servicios de alto nivel puedan implementar políticas de seguridad específicas seleccionadas de mecanismos de seguridad de bajo nivel.

Certificado digital

Un certificado digital es un documento digital mediante el cual un tercero confiable (una autoridad de certificación o CA) garantiza la vinculación entre la identidad de un sujeto o entidad y su clave pública. Si bien existen varios formatos de certificado digital, los más comúnmente empleados se rigen por el estándar UIT-T X.509v3. El certificado contiene usualmente el nombre de la entidad certificada, un número serie, fecha de expiración, una copia de la clave pública del titular del certificado (utilizada para la verificación de su firma digital), y la firma digital de la autoridad emisora del certificado de forma que el receptor pueda verificar que esta última ha establecido realmente la asociación. X.509 es un estándar ITU-T para infraestructura de claves pública (public key infrastructure o PKI). X.509 especifica, entre otras cosas, formatos estándar para certificados de claves públicas y un algoritmo de validación de ruta de certificación. X.509 es la pieza central de la infraestructura PKI, y es la estructura de datos que enlaza la clave pública con los datos que permiten identificar al titular. Su sintaxis, se define empleando el lenguaje ASN.1 (Abstract Syntax Notation One), y los formatos de codificación más comunes son DER (Distinguish Encoding Rules) o PEM (Privacy Enhanced Mail).

Estructura de un certificado X.509

Siguiendo la notación de ASN.1, un certificado contiene diversos campos, agrupados en tres grandes grupos:

- El primer campo, es el subject, que contiene los datos que identifican al titular. Estos datos están expresados en notación DN (Distinguished Name), donde un DN se compone a su vez de diversos campos, siendo los más frecuentes los siguientes; CN (Common Name), OU (Organizational Unit), O (Organization) y C (Country).

Además del nombre del titular (subject), el certificado, también contiene datos asociados al propio certificado digital, como la versión del certificado, su identificador (serialNumber), la CA firmante (issuer), el tiempo de validez (validity), etc. La versión X.509.v3 también permite utilizar campos opcionales (nombres alternativos, usos permitidos para la clave, ubicación de la CRL y de la CA, etc.).

- En segundo lugar, el certificado contiene la clave pública, que expresada en notación ASN.1, consta de dos campos, en primer lugar, el que muestra el algoritmo utilizado para crear la clave (p.e. RSA), y en segundo lugar, la propia clave pública.

- Por último, la CA, ha añadido la secuencia de campos que identifican la firma de los campos previos. Esta secuencia contiene tres atributos, el algoritmo de firma utilizado, el hash de la firma, y la propia firma digital.

Autoridad de certificación

Es la entidad de confianza, responsable de emitir y revocar los certificados digitales. La Autoridad de Certificación (CA) es quien da validez a los certificados mediante la firma digital de éstos con la clave privada de la CA, es decir, legitima ante terceras partes la relación entre la identidad de un usuario y su clave pública. La confianza de los usuarios en la CA es fundamental para el buen funcionamiento del servicio. Un certificado que se revoca es un certificado que se invalida definitivamente antes de que venza su período de validez (p.e. porque se sospecha que la clave privada correspondiente haya sido comprometida). Un certificado revocado no puede utilizarse más.

El mecanismo habitual de solicitud de un certificado a una CA consiste en que la entidad que lo solicita envíe una petición PKCS#10 a la CA, donde PKCS corresponde a un conjunto de especificaciones que son estándares de facto denominadas Public-Key Cryptography Standards. La petición incluye la información del solicitante, su clave pública y para que la CA pueda validar el par de

claves, un pequeño fragmento firmado con su clave privada. Si se requiere una comprobación adicional de los datos de la entidad, puede intervenir una autoridad de registro (RA) en el proceso de solicitud de certificación como intermediario. Las CA disponen de sus propios certificados públicos, cuyas claves privadas asociadas son empleadas por las CA para firmar los certificados solicitados. Sin embargo, un certificado puede estar auto-firmado cuando no hay ninguna CA que lo firme. Este es el caso de los certificados de CA raíz, el elemento inicial de cualquier jerarquía de certificación. Las jerarquías de certificación consisten en una estructura jerárquica de CA en la que se parte de una CA auto-firmada, y en cada nivel, se puede firmar certificados de entidades o bien certificados de otras CA subordinadas si se tiene permiso para ello.

El modo en el que se establece la confianza en las CA consiste en la instalación en los sistemas de cada entidad, de los certificados de las autoridades clasificadas como CA de confianza. De esta forma, cualquier certificado firmado por una CA de la lista se podrá validar, ya que se dispone de la clave pública con la que verificar la firma que lleva el certificado. Cuando el modelo de CA incluye una jerarquía, hará falta confiar explícitamente en los certificados de todas las cadenas de certificación en las que se confíe. Para ello, se puede localizar sus certificados mediante distintos medios de publicación en internet, pero también es posible que un certificado contenga toda la cadena de certificación necesaria para ser instalado con confianza.

Finalmente, las CA también se encargan de la gestión de los certificados firmados. Esto incluye las tareas de revocación de certificados mediante la solicitud de la entidad. Como se ha comentado, la lista denominada CRL contiene los certificados que entran en esta categoría, por lo que es responsabilidad de la CA publicarla y actualizarla debidamente. Por otra parte, otra tarea que debe realizar una CA es la de renovación de certificados por caducidad o revocación, por ejemplo. Una CA puede ser o bien publica o bien privada. Los certificados de CA de las CAs públicas suelen estar instalados en los navegadores y son reconocidos como entidades confiables. Las CAs públicas emiten los certificados para la población en general (aunque a veces están focalizadas hacia algún colectivo en concreto) y además firman CAs de otras organizaciones.

2.1.6. Servicios de acceso a datos

Es el responsable de ofrecer el acceso remoto a los datos almacenados en los sistemas de almacenamiento. Para el acceso con base de datos, Globus ofrece una total integración con CORBA. Las aplicaciones de computación Grid en general necesitarán de acceso a datos localizados en múltiples dominios administrativos.

Sistemas de ficheros distribuidos como NFS y DFS no están diseñados para aplicaciones de alto rendimiento. Sistemas de ficheros paralelos y librerías I/O han sido diseñados para dar alto rendimiento, pero no para ejecución distribuida. Para encauzar estos problemas, el módulo de Globus para el acceso de datos define una serie de primitivas que proveen de acceso remoto a los sistemas de ficheros paralelos. Este interfaz de I/O remoto (RIO) esta basado en el concepto de interfaz de dispositivo I/O (ADIO), que no es más que un interfaz para apertura, cierre, lectura y escritura paralela de ficheros. No define semántica para *caching* ni replicación de ficheros. Varios sistemas I/O muy extendidos han sido implementados eficientemente en ADIO. RIO extiende la capacidad de ADIO añadiendo acceso remoto y "nombres globales" usando el concepto de URL. RIO utiliza mecanismos propios de Nexus.

2.1.7. Servicios de gestión de recursos y aplicaciones

Las aplicaciones de computación Grid necesitan muy a menudo ejecutarse en entornos de red y en recursos de computación a un nivel muy cercano al máximo de rendimiento. Por este motivo, los entornos de computación Grid deben permitir a los programadores observar las diferencias existentes entre las características de los recursos del sistema y de igual modo guiar dichos recursos para que sean utilizados por servicios de más alto nivel. Alcanzar ambos objetivos sin comprometer la portabilidad es uno de los retos más importantes en el diseño de la computación Grid. Como ilustración de esto podemos fijarnos en el módulo Globus de comunicación. Este módulo debe seleccionar, en cada llamada a sus funciones de comunicación, uno de los varios mecanismos de bajo nivel posibles.

Por ejemplo, en una LAN, la comunicación puede basarse en TCP/IP mientras que un cluster de computación se utiliza protocolos especializados que ofrecen mayores anchos de banda y latencias más bajas. Por el contrario a nivel WAN, protocolos del tipo ATM pueden resultar más eficientes. Además la necesidad de poder manejar parámetros del protocolo, como tamaño del paquete (TCP), calidad de servicio, etc., complican el problema. La elección de un mecanismo de bajo nivel para que sea utilizado para una comunicación particular es por tanto un problema no trivial que pueden tener importantes implicaciones para el rendimiento de las aplicaciones. El módulo de Globus resuelve este problema proporcionando interfaces que permiten la selección de procesos para ser expuestos y guiados por las herramientas de alto nivel y aplicaciones. Estos interfaces ofrecen tres mecanismos: *de selección basada en reglas*, *de búsqueda del recurso adecuado*, y *de notificación*.

- ***Selección basada en reglas***

  Los módulos de Globus pueden elegir entre distintas alternativas (recursos, parámetros, etc.) en base a unas reglas ofrecidas por el usuario (p.e. "usa el tamaño de paquete TCP xx", "usa TCP sobre ATM"). A su vez estas reglas pueden ser sustituidas por otras alternativas por los servicios de más alto nivel (p.e. "usa tamaño de paquete TCP yy", "usa ATM exclusivamente").

- ***Búsqueda del recurso apropiado***

  La información que ofrece el servicio de información unificada (ver sección mas adelante) puede utilizarse como guía de selección tanto para los módulos de Globus como para las aplicaciones que utilizan dichos módulos. Por ejemplo, un usuario puede exigir una regla del tipo "usa ATM si la carga es baja, si no, utiliza Internet", y que el módulo utilice la información disponible sobre la carga de red para buscar el recurso adecuado.

- ***Notificación***

  Un mecanismo de notificación permite a los servicios de mas alto nivel y a las aplicaciones especificar cual es la calidad de servicio ofrecida por los servicios Globus y a su vez llamar a una función "*call-back*" cada vez que esta no supere un cierto valor. Este mecanismo puede ser utilizado, por ejemplo, para pasar a otra red cuando la actual sobrepasa un límite de carga.

2.1.8. Servicios de alto nivel

En la sección anterior, se describió el núcleo de los servicios e interfaces que componen el toolkit Globus. Estos interfaces no están planeados para uso como aplicaciones, sino como herramientas para la construcción de componentes de más alto nivel. Estos componentes pueden servir como funciones de middleware, sobre los cuales se construyen las aplicaciones a nivel de interfaz. Los servicios de alto nivel y las aplicaciones puede utilizar tanto mecanismos de selección, búsqueda y notificación para configurar la computación eficientemente para los recursos disponibles y/o adaptar su comportamiento cuando la cantidad y/o calidad de los recursos cambie durante la ejecución. Por ejemplo, consideremos una aplicación que corre en una computadora y transfiere los datos sobre la red para visualización remota. En el momento de comienzo de la ejecución la aplicación puede determinar el poder computacional y la capacidad de red disponible y configurar sus estructuras de computación y comunicación adecuadamente (p.e. puede decidir comprimir un tipo de datos y otros no). Durante la ejecución los mecanismos de notificación permiten que esta se adapte a los posibles cambios en la calidad de servicio de la red.

Interfaces de programación paralela

Numerosos interfaces de programación paralela han sido adaptados para utilizar los módulos propios de Globus, permitiendo a los programadores desarrollar aplicaciones de computación Grid utilizando herramientas familiares. Estos interfaces incluyen una completa implementación de MPI (y de todas las herramientas conviviendo en capas superiores de MPI, como HPF - high performance fortran -); extensiones de C++ para programación paralela; Fortran M, un fortran para ejecución paralela; nPerl, una versión extendida de Perl con mecanismos de ejecución remota, y Nexus-Java, una librería de clases Java que soporta procedimientos remotos. En concreto, sobre la implementación de Globus en MPI, un “dispositivo Nexus” mantiene el concepto de dispositivo de interfaz utilizado dentro de la implementación MPICH de MPI. Este dispositivo utiliza Nexus RSRs para implementar las operaciones de transferencia de datos requeridas por MPI. Además de Nexus, la implementación Globus de MPI conlleva el uso de mecanismos Globus de autentificación y arranque de procesos.

Estas componentes están perfectamente integradas en I-WAY, y el usuario puede asignar una colección heterogénea de recursos y arrancar un programa con solo teclear "*impirun*", encargándose Globus de seleccionar los adecuados mecanismos de autentificación, creación de procesos y comunicación.

Autentificación basada en certificados

El interfaz de autentificación Globus puede utilizarse para implementar todo un rango de diferentes políticas de seguridad. Se está desarrollando una política que defina un espacio global de autentificación, basados en clave pública, para todos los usuarios y dispositivos. La idea es definir una autoridad centralizada que crearía "cuentas" tanto para usuarios como para recursos. Estas cuentas permiten a una aplicación usar un único "identificador de usuario" y un "password" para todos los recursos, permitiendo la verificación de los recursos empleados. Hay que reseñar que esto no significa que los recursos no utilicen sus mecanismos usuales para determinar que usuarios pueden acceder y cuales no a ellos. Este mecanismo tiene la importante ventaja de que puede ser fácilmente implementada con los actuales protocolos de autentificación basados en certificación, tal como SSL. Es importante también destacar que mientras los certificados son dados por la autoridad de certificación central, esta no interactúa para nada en el propio proceso de autorizar a un usuario o un recurso.

En un plazo largo, los procesos de autentificación y autorización deberán adaptarse para grandes, heterogéneas y dinámicas comunidades de usuarios, con relaciones expandiéndose por muchos y variados dominios administrativos, relaciones que serán irregulares y selectivas. Probablemente se estimulara la formación de subcomunidades con sus propios criterios y políticas de uso, y donde muchos miembros delegaran en otros las labores de extender estas relaciones a otras subcomunidades. La política de certificados de Globus puede extenderse a un ilimitado número de formas de delegación de confianza. Los certificados Globus de recursos pueden ofrecerse a centros con múltiples recursos (o usuarios), que a su vez pueden generar certificaciones locales basadas en una relación de confianza con la certificación Globus.

2.2. Instalación y configuración

A continuación se proporciona un método genérico para instalar y configurar Globus toolkit, en el entorno de computación Grid LHC del CERN (LCG). LCG es el resultado del desarrollo de proyectos Grid, como DataGrid, DataTag, Globus, GriPhyN, iVDGL, y EGEE (Enabling Grids for E-sciencE). El middleware LCG-2 es el que se utiliza en todas las instituciones participantes en el proyecto EGEE. La versión actual del middleware, utiliza el sistema operativo *Scientific Linux 3* (SL3), bajo arquitecturas IA32 y IA64. Scientific Linux es una distribución basada en Red Hat Enterprise Linux con algunas modificaciones, aportadas principalmente por FNAL y el CERN, y tiene como objetivo establecer una base común de instalación para los diferentes experimentos. La dirección correspondiente a esta distribución es *http://www.scientificlinux.org*

Globus consta de tres componentes fundamentales: *Gestión de recursos* (GRAM - Globus Resource Allocation Manager), *Servicio de información* (MDS - Metacomputing Directory Service), *Gestión de datos* (GASS - Global Access to Secondary Storage). Cada componente proporciona un servicio básico como autenticación, asignación de recursos, información, comunicación, detección de fallos y acceso remoto a datos. Todos ellos usan el protocolo de seguridad GSI (Globus Security Infrastructure) para la comunicación y autentificación, y ya sea de forma independiente o conjunta, 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: descubrimiento, selección y preparación de recursos y envío, monitorización, migración y finalización de trabajos.

La configuración por defecto de Globus asigna al servicio GRAM (gatekeeper) al puerto 2119. El acceso a cada recurso se controla mediante una serie de asignaciones entre el subject (DN) del certificado de un usuario del Grid que puede usar el recurso, y un usuario local. Cada una de las organizaciones virtuales, atendiendo a su propia política de administración, deberá decidir qué usuarios del Grid tienen acceso a cada recurso; y si estos se asignan al mismo usuario local, o por el contrario

si se crea una cuenta local distinta para cada usuario del Grid. El sistema de información de Globus es el Metacomputing Directory Service (MDS), que usa el protocolo LDAP para la consulta uniforme de la información referente a los sistemas en el Grid. En particular, mediante el Grid Resource Information Service (GRIS), se puede consultar el estado, la configuración, y las prestaciones de cada recurso del Grid. La información suministrada por cada GRIS se agrupa en el Grid Index Information Service (GIIS), que ofrece una imagen conjunta y coherente de los recursos del Grid. Por defecto, el servidor LDAP (slapd), tanto para el GRIS como para el GIIS, se asigna al puerto 2135. Las consultas a los servidores GRIS y GIIS se realizarán de forma anónima.

La monitorización de la información sigue una estructura jerárquica. Se configurará un host como nodo raíz para el acceso superior GIIS, que a su vez preguntará a los servidores GIIS de cada VO con el nombre de la VO. Con el fin de simplificar su gestión, todos los grupos, están al mismo nivel independientemente de si pertenecen a la misma institución o ubicación geográfica; internamente las diferentes VOs podrán estar a su vez organizadas jerárquicamente. El servidor GridFTP es un protocolo de transferencia de ficheros, seguro y de alto rendimiento, basado en el protocolo FTP y de gran importancia en Grids de datos. El servicio GridFTP se asigna por defecto al puerto 2811.

La gestión de la seguridad es la componente más importante de un Grid. La infraestructura de seguridad seguida por Globus está basada en PKI (Public Key Infrastructure) por medio del uso de criptografía asimétrica con claves públicas y privadas. La información encriptada por medio de una clave privada sólo puede ser desencriptada por su clave pública y viceversa. La clave privada sólo es accesible a los usuarios y a las máquinas que desean autenticarse mientras que la clave pública es accesible para todos. La clave pública se distribuye por medio de un certificado X509 que lleva un identificador único o DN (Distinguished Name) asociado a un usuario, máquina o servicio, y que está firmado por una CA (Certification Authority). De este modo, cuando un usuario se autentica envía su certificado X509 y su identidad es reconocida si está firmado por una CA de confianza (todo esto se realiza de forma transparente para el usuario por medio del protocolo SSL). Mientras que la

autenticación implica demostrar quién se asegura ser, autentificar implica definir el acceso del usuario al sistema. Globus implementa la autorización por medio de un fichero de mapeo que asigna DNs a cuentas UNIX locales. Es importante resaltar que el hecho de que un usuario tenga un certificado X509 firmado por una CA válida, no implica que esté autorizado a acceder al sistema. Esta decisión es interna a cada VO que es responsable de la gestión de sus propios recursos.

2.2.1. Tipos de nodos

La relación de los diferentes tipos de nodos que se pueden instalar y configurar en una infraestructura Grid basada en servicios Globus, es la siguiente:

Type	Package	Description
BDII	lcg-BDII	Berkeley Database Information Index
CE	lcg-CE	Computing Element with no LRMS
CE_torque	lcg-CE_torque	CE with Torque LRMS client & server
LFC_mysql	lcg-LFC_mysql	mysql based LCG File Catalog server
LFC_oracle	lcg-LFC_oracle	oracle based LCG File Catalog server
MON	lcg-MON	RGMA monitoring system collector server
PX	lcg-PX	Proxy Server
RB	lcg-RB	Resource Broker
SE_classic	lcg-SE_classic	Storage Element on local disk
SE_dpm_disk	lcg-SE_dpm_disk	Disk server for SE Disk Pool Manager
SE_dpm_mysql	lcg-SE_dpm_mysql	DPM with SRM interface & mysql backend
SE_dpm_oracle	lcg-SE_dpm_oracle	DPM with SRM interface & oracle backend
SE_dcache	lcg-SE_dcache	Storage Element interfaced with dCache
dcache_gdbm	lcg-SE_dcache_gdbm	SE interfaced to dCache with gdbm
TAR_UI	lcg-TAR_UI	Re-locatable distribution for UI
TAR_WN	lcg-TAR_WN	Re-locatable distribution for Worker Node
UI	lcg-UI	User Interface
VOBOX	lcg-VOBOX	Machine to run VO agents
VOMS_mysql	lcg-VOMS_mysql	VO Management Service with mysql
VOMS_oracle	lcg-VOMS_oracle	VO Management Service with oracle
WN	lcg-WN	Worker Node with no LRMS
WN_torque	lcg-WN_torque	WN with Torque LRMS client

En el proceso de instalación y configuración, el parámetro `package` identifica al tipo de nodo correspondiente.

2.2.2. Parámetros de configuración

En esta sección se describen los parámetros de configuración para los diferentes tipos de nodos que se pueden instalar en una infraestructura Grid basada en servicios Globus.

– Base de datos de contraseñas
  `APEL_DB_PASSWORD`
– Directorio de registro de mensajes para el sistema de procesamiento por lotes
  `BATCH_LOG_DIR`
– Dirección del nodo que elige las direcciones de los recursos
  `BDII_FCR`
– Nombre del nodo correspondiente al servicio de información de recursos
  `BDII_HOST`
– Dirección del fichero de configuración del nodo `BDII`
  `BDII_HTTP_URL`
– Tipos de nodos que publican información en el nodo `BDII`
  `BDII_REGIONS`
– Dirección del nodo que genera la información del recurso computacional (CE)
  `BDII_CE_URL`
– Dirección del nodo que genera la información del recurso de almacenamiento (SE)
  `BDII_SE_URL`
– APT con Autoridades de Certificación
  `CA_REPOSITORY`
– Sistema de procesamiento por lotes definido en el CE
  `CE_BATCH_SYS`
– Modelo de CPU del elemento computacional (WN)
  `CE_CPU_MODEL`
– Frecuencia de reloj en MHz de la CPU del WN
  `CE_CPU_SPEED`
– Fabricante de la CPU del WN
  `CE_CPU_VENDOR`

– Area de datos compartida por los WNs de un CE

`CE_DATADIR`

– Nombre del nodo correspondiente al recurso computacional

`CE_HOST`

– Habilitar (`TRUE`) o deshabilitar (`FALSE`) la conectividad hacia el CE

`CE_INBOUNDIP`

– Memoria RAM en Kbytes del WN

`CE_MINPHYSMEM`

– Memoria Virtual en Kbytes del WN

`CE_MINVIRTMEM`

– Sistema Operativo instalado en el WN

`CE_OS`

– Versión del Sistema Operativo instalado en el WN

`CE_OS_RELEASE`

– Habilitar (`TRUE`) o deshabilitar (`FALSE`) la conectividad desde el CE

`CE_OUTBOUNDIP`

– Software disponible en el CE

`CE_RUNTIMEENV`

– Indice de rendimiento SpecFloat 2000 del WN

`CE_SF00`

– Indice de rendimiento SpecInt 2000 del WN

`CE_SI00`

– Número de CPUs para multiprocesamiento simétrico (SMP) en el WN

`CE_SMPSIZE`

– Nombre del nodo correspondiente al recurso de almacenamiento

`CLASSIC_HOST`

– Directorio raíz definido para el `CLASSIC_HOST`

`CLASSIC_STORAGE_DIR`

– Directorio de registro de mensajes para los trabajos ejecutados periódicamente

`CRON_DIR`

– Nombre del nodo correspondiente al servicio que gestiona conjuntos de nodos

`DCACHE_ADMIN`

– Lista de conjuntos de nodos gestionados por `DCACHE_ADMIN`

`DCACHE_POOLS`

– Rango de puertos de comunicaciones para el servicio `DCACHE` (`20000-25000`)

`DCACHE_PORT_RANGE`

– Directorio definido para el almacenamiento de datos

`DPMDATA`

– Tamaño máximo de fichero que se puede procesar

`DPMFSIZE`

– Base de datos de cuentas de usuario para el servicio `DPM`

`DPMMGR`

– Lista de los conjuntos de nodos registrados en el servicio `DPM` (`hostname:/path`)

`DPMPOOL`

– Conjunto de nodos correspondiente a `DPMPOOL`

`DPMPOOL_NODES`

– Contraseña para la base de datos de cuentas de usuario

`DPMUSER_PWD`

– Nodo correspondiente al servicio *Disk Pool Manager* (DPM)

`DPM_HOST`

– Rango de puertos de comunicaciones para el servicio DPM (`20000-25000`)

`DPM_PORT_RANGE`

– Directorio de almacenamiento para el servicio *WorkLoad* (WL)

`EDG_WL_SCRATCH`

– Dirección correspondiente al servicio de transferencia de ficheros (FTS)

`FTS_SERVER_URL`

– Directorio correspondiente a las funciones del programa de instalación *yaim*

`FUNCTIONS_DIR`

– Rango de puertos de comunicaciones para Globus

`GLOBUS_TCP_PORT_RANGE`

– Nombre de nodo del servicio de monitorización *GridIce*

`GRIDICE_SERVER_HOST`

– Lista de servidores LDAP para la autentificación de usuarios

`GRIDMAP_AUTH`

– Lista de DNs de certificados de RBs válidos para el nodo Proxy

`GRID_TRUSTED_BROKERS`

– Grupos VOMS y su correspondencia con los grupos locales

`GROUPS_CONF`

– Habilitar (`yes`) o deshabilitar (`no`) los servicios genéricos de seguridad

`GSSKLOG`

– Nombre de nodo correspondiente a la autentificación AFS

`GSSKLOG_SERVER`

– Directorio definido para la instalación y configuración de Globus toolkit

`INSTALL_ROOT`

– Directorio correspondiente a la instalación de Java VM

`JAVA_LOCATION`

– Nombre del gestor de trabajos utilizado por el *gatekeeper*

`JOB_MANAGER`

– Colección de software LCG

`LCG_REPOSITORY`

– Catálogo de ficheros

`LFC_CENTRAL`

– Base de datos de contraseñas para los usuarios del servicio LFC

`LFC_DB_PASSWORD`

– Nombre del nodo correspondiente al servicio LFC

`LFC_HOST`

– Lista de Organizaciones Virtuales (VOs) asociadas al servicio LFC

`LFC_LOCAL`

– Nombre del nodo correspondiente al servicio de monitorización (MON)

`MON_HOST`

– Contraseña para la base de datos MYSQL

`MYSQL_PASSWORD`

– Dominio de la Organización (*Full Qualified Domain Name*)

`MY_DOMAIN`

– Directorio por defecto para el registro del resultado de la ejecución de los trabajos

`OUTPUT_STORAGE`

– Nombre del nodo Proxy

`PX_HOST`

– Colas de procesamiento de trabajos definidas en el CE

`QUEUES`

– Nombre del nodo del *Resource Broker*

`RB_HOST`

– Nombre del nodo correspondiente al servicio RGMA

`REG_HOST`

– Habilitar (`yes`) el servicio dCache

`RESET_DCACHE_CONFIG`

– Arquitectura del recurso de almacenamiento (`multidisk`, `disk`, `tape`, `other`)

`SE_ARCH`

– Lista de nodos SE disponibles en la organización

`SE_LIST`

– Dirección de correo electrónico correspondiente a la organización

`SITE_EMAIL`

– Latitud donde se encuentra ubicada la organización

`SITE_LAT`

– Ciudad o provincia donde se encuentra ubicada la organización

`SITE_LOC`

– Longitud donde se encuentra ubicada la organización

`SITE_LONG`

– Nombre de la organización registrado en el servicio GIIS

`SITE_NAME`

– Identificador para la organización en la base de datos GOC

`SITE_SUPPORT_SITE`

– Identificador para la organización

`SITE_TIER`

– Página Web de la organización

`SITE_WEB`

– Definición del LRMS *Torque*

`TORQUE_SERVER`

– Fichero de configuración con la lista de usuarios

`USERS_CONF`

– Nombre del nodo correspondiente al servicio VOBOX

`VOBOX_HOST`

– Puerto de comunicaciones asignado al servicio VOBOX

`VOBOX_PORT`

– Lista de VOs autorizadas

`VOS`

– Directorio para la instalación de software

`VO_SW_DIR`

– Lista de *Worker Nodes* (WNs)

`WN_LIST`

– Versión de *yaim* que debe utilizarse para procesar el fichero de configuración

`YAIM_VERSION`

Para cada VO se definen los siguientes parámetros:

- Relación de colas para la ejecución de trabajos que se pueden solicitar en un CE
  `VO_name_QUEUES`
- Recurso de almacenamiento por defecto, que puede utilizar una VO
  `VO_name_SE`
- Directorio LDAP con la lista de gestores del software de la VO
  `VO_name_SGM`
- Area de datos del SE definido para la VO
  `VO_name_STORAGE_DIR`
- Area de datos del WN definida para la instalación de software
  `VO_name_SW_DIR`
- Directorio LDAP con la lista de usuarios de la VO
  `VO_name_USERS`
- Lista de registros VOMSES de ficheros para la VO
  `VO_name_VOMSES`
- Ficheros adicionales de correspondencia
  `VO_name_VOMS_EXTRA_MAPS`
- Lista de servidores VOMS para la VO
  `VO_name_VOMS_SERVERS`

2.2.3. Método de instalación

El método de instalación está basado en las herramientas `apt-get` y `yaim`. La herramienta `apt-get` está incluida en la distribución SL3.

Antes de instalar el middleware, es totalmente imprescindible tener instalado el paquete software `j2sdk` (java sdk), incluido en la distribución SL3. También es muy importante configurar el protocolo NTP para la correcta sincronización entre el nodo y el resto de nodos de la infraestructura Grid.

Los ficheros de configuración asociados a la herramienta `apt-get`, se encuentran ubicados en el directorio `/etc/apt/sources.list.d`

- SL3_repository (`sl.list`)

```
rpm ftp://ftp.scientificlinux.org//linux/scientific/
305/i386/apt-rpm os updates
```

- CA_repository (`lcg-ca.list`)

```
rpm http://grid-deployment.web.cern.ch/grid-deployment/gis
apt/LCG_CA/en/i386 lcg
```

- LCG_repository (`lcg.list`)

```
rpm http://grid-deployment.web.cern.ch/grid-deployment/gis
apt/LCG-2_7_0/sl3/en/i386 lcg_sl3 lcg_sl3.updates lcg_sl3.security
```

La herramienta `yaim` se puede obtener en la siguiente dirección:
http://www.cern.ch/grid-deployment/gis/yaim/lcg-yaim-x.x.x-x.noarch.rpm

- Los ficheros de configuración, se instalan en el directorio `/opt/lcg/yaim/examples`
- Los parámetros de configuración se especifican en el fichero `site-info.def`

Para instalar el middleware, debemos ejecutar el siguiente comando:

```
/opt/lcg/yaim/scripts/install_node <site-configuration-file> <packages>
```

2.2.4. Método de configuración

El método de configuración está basado en la herramienta `yaim`

Para configurar el middleware, debemos ejecutar el siguiente comando:

```
/opt/lcg/yaim/scripts/configure_node <site-configuration-file> <types>
```

Para obtener la información relativa a las autoridades de certificación (CA), ubicada en el directorio `/etc/grid-security` debemos ejecutar el comando:

```
apt-get update && apt-get –y install lcg-CA
```

Los certificados de hosts, deben solicitarse a pkIRISGridCA. Una vez obtenido un certificado válido, los ficheros correspondientes a la clave pública (`hostcert.pem`) y a la clave privada (`hostkey.pem`), deben ubicarse en el directorio `/etc/grid-security`

Estos certificados se requieren para la configuración de los nodos Computing Element (**CE**), LCG File Catalog (**LFC**), Monitoring System Collector (**MON**), Proxy Server (**PX**), Resource Broker (**RB**), Storage Element (**SE**) y VO agents (**VOBOX**).

Para mantener actualizada la información relativa a las autoridades de certificación (CA), es imprescindible programar un procedimiento de actualización periódica en cada nodo instalado:

/etc/cron.d/edg-fetch-crl

```
PATH=/sbin:/bin:/usr/sbin:/usr/bin
27 12,18,0,6 * * * root /opt/edg/etc/cron/edg-fetch-crl-cron >>
/var/log/edg-fetch-crl-cron.log 2>&1
```

2.2.5. Interfaz de usuario

El punto de acceso a los servicios Grid es el *User Interface* (UI). El UI es un nodo donde usuarios registrados en LCG, tienen una cuenta personal con su correspondiente certificado de usuario instalado. Desde el UI, un usuario puede ser autenticado y autorizado para utilizar los recursos Grid. Este es el componente que permite a los usuarios acceder a las funcionalidades ofrecidas por los el sistema de información (*Information*), el sistema de carga de trabajos (*WorkLoad*) y el sistema de gestión de datos (*Data Management*).

LCG-2 está organizado en Organizaciones Virtuales (VO). Cada VO agrupa usuarios, instituciones y recursos dentro del mismo dominio de administración. El objetivo es instalar y configurar un nodo UI, que permita al usuario:

- Listar todos los recursos Grid disponibles dentro de su VO
- Enviar trabajos para su ejecución
- Monitorizar el estado de los trabajos enviados
- Cancelar trabajos
- Obtener el resultado de los trabajos finalizados
- Solicitar información de los trabajos
- Copiar, replicar y borrar ficheros del Grid

El siguiente ejemplo es el utilizado para configurar nuestro nodo como User Interface, dentro de la Organización Virtual **swetest**

/opt/lcg/yaim/examples/IAA-info.def

```
SITE_NAME=IAA
MY_DOMAIN=iaa.es
CE_HOST=ce00.$MY_DOMAIN
BDII_HOST=lcg2bdii.ific.uv.es
MON_HOST=lcg2mon.ific.uv.es
PX_HOST=myproxy.cern.ch
RB_HOST=lcg2rb.ific.uv.es
REG_HOST=lcgic01.gridpp.rl.ac.uk
VOS="swetest"
LCG_REPOSITORY="rpm http://grid-deployment.web.cern.ch/grid-deployment/gis
apt/LCG-2_7_0/sl3/en/i386 lcg_sl3 lcg_sl3.updates lcg_sl3.security"
CA_REPOSITORY="rpm http://grid-deployment.web.cern.ch/grid-deployment/gis
apt/LCG_CA/en/i386 lcg"
YAIM_VERSION=2.7.0-2
JAVA_LOCATION= "/usr/java/j2sdk1.4.2_08"
INSTALL_ROOT=/opt
```

Para instalar el middleware correspondiente al UI, ejecutamos el comando:

```
/opt/lcg/yaim/scripts/install_node IAA-info.def lcg-UI
```

Para configurar el UI, ejecutamos el comando:

```
/opt/lcg/yaim/scripts/configure_node IAA-info.def UI
```

La instalación y configuración del Interfaz de Usuario, nos sirve como base para el diseño y desarrollo de un portal de recursos computacionales. Las herramientas utilizadas para el diseño y desarrollo del portal Web, son las siguientes:

- Para el diseño gráfico se ha utilizado *Adobe Photoshop*
- Para la elaboración de las páginas Web se ha utilizado *Macromedia Dreamweaver*
- Las páginas Web se han diseñado con hojas de estilos CSS
- La integración de los servicios Globus, se ha realizado con la programación de módulos en *Perl*, utilizando funciones *JavaScript*.

3

Diseño y desarrollo del portal

3.1. Configuración

En esta sección se analiza la configuración del servidor Web Apache HTTP versión 2.0, incluido en la distribución SL3, así como también una serie de módulos diseñados para mejorar su funcionalidad. Para el acceso al portal se utilizará el protocolo **https**. Este protocolo utiliza un canal cifrado basado en Secure Sockets Layer (SSL). La dirección Web correspondiente al portal es *https://irisgrid.iaa.es*

3.1.1. Secure Sockets Layer

El portal proporciona autenticación y privacidad de la información mediante el uso de criptografía, utilizando los protocolos Secure Sockets Layer (SSL) y Transport Layer Security (TLS). Normalmente, solo el servidor es autenticado (es decir, se garantiza su identidad) mientras que el cliente se mantiene sin autenticar; la autenticación mutua requiere un despliegue de infraestructura de claves públicas (o PKI) para los clientes. Los protocolos permiten a las aplicaciones cliente-servidor comunicarse de una forma segura. El protocolo SSL supone una serie de fases básicas: (i) Negociar entre las partes el algoritmo que se usará en la comunicación. (ii) Intercambio de claves públicas y autenticación basada en certificados digitales. (iii) Encriptación del tráfico basado en cifrado simétrico. Durante la primera fase, el cliente y el servidor negocian qué algoritmos criptográficos se van a usar. Las implementaciones actuales proporcionan las siguientes opciones: (i) *Criptografía de clave pública* RSA, Diffie-Hellman, DSA (Digital Signature Algorithm) o Fortezza. (ii) *Cifrado simétrico* RC2, RC4, IDEA (International Data Encryption Algorithm), DES (Data Encryption Standard), Triple DES o AES (Advanced Encryption Standard). (iii) *Funciones hash* para identificar archivos, registros de bases de datos, integridad en los archivos MD5 (Message-Digest Algorithm 5), o de la familia SHA (Secure Hash Algorithm).

El protocolo SSL intercambia registros; opcionalmente, cada registro puede ser comprimido, encriptado y empaquetado con un código de autenticación del mensaje (MAC). Cada registro tiene un campo de *content_type* que especifica el protocolo de nivel superior que se está usando. Cuando se inicia la conexión, el nivel de registro encapsula otro protocolo, el protocolo *handshake*, que tiene el *content_type 22*.

El cliente envía y recibe varias estructuras *handshake*: (i) Envía un mensaje *ClientHello* especificando una lista de conjunto de cifrados, métodos de compresión y la versión del protocolo más alta permitida; éste también envía bytes aleatorios que serán usados más tarde. (ii) Después, recibe un registro *ServerHello*, en el que el servidor elige los parámetros de conexión a partir de las opciones ofertadas con anterioridad por el cliente. (iii) Cuando los parámetros de la conexión son conocidos, cliente y servidor intercambian certificados (dependiendo de las claves públicas de cifrado seleccionadas); estos certificados son actualmente X.509, pero hay también un borrador especificando el uso de certificados basados en OpenPGP. (iv) El servidor puede requerir un certificado al cliente, para que la conexión sea mutuamente autenticada. (v) Cliente y servidor negocian una clave secreta común llamada *master secret*, posiblemente usando el resultado de un intercambio Diffie-Hellman, o simplemente encriptando una clave secreta con una clave pública que es desencriptada con la clave privada de cada uno. Todos los datos de claves restantes son derivados a partir de este *master secret* (y los valores aleatorios generados en el cliente y el servidor), que son pasados a través una función pseudo aleatoria cuidadosamente elegida.

TLS/SSL poseen una variedad de medidas de seguridad: (i) Enumerando todos los registros y usando el número de secuencia en el MAC. (ii) Usando un resumen de mensaje mejorado con una clave (de forma que solo con dicha clave se pueda comprobar el MAC). Esto se especifica en el RFC 2104. (iii) Protección contra varios ataques conocidos (incluidos ataques *man in the middle attack*), como los que implican un degradado del protocolo a versiones previas (por tanto, menos seguras), o conjuntos de cifrados más débiles. (iv) La función pseudo aleatoria divide los datos de entrada en 2 mitades y las procesa con algoritmos hash diferentes (MD5 y SHA), después realiza sobre ellos una operación XOR. De esta forma se protege a sí mismo

de la eventualidad de que alguno de estos algoritmos se revelen vulnerables en el futuro. (v) El mensaje que finaliza el protocolo handshake (Finished) envía un hash de todos los datos intercambiados y vistos por ambas partes.

SSL se ejecuta en una capa entre los protocolos de aplicación como HTTP, SMTP, NNTP y sobre el protocolo de transporte TCP, que forma parte de la familia de protocolos TCP/IP. Aunque pueda proporcionar seguridad a cualquier protocolo que use conexiones de confianza (tal como TCP), se usa en la mayoría de los casos junto a HTTP para formar HTTPS.

HTTPS es usado para asegurar páginas World Wide Web para aplicaciones de comercio electrónico, utilizando certificados de clave pública para verificar la identidad de los extremos. Aunque un número creciente de productos clientes y servidores pueden proporcionar SSL de forma nativa, muchos aún no lo permiten. En estos casos, un usuario podría querer usar una aplicación SSL independiente como Stunnel para proporcionar encriptación. No obstante, el Internet Engineering Task Force recomendó en 1997 que los protocolos de aplicación ofrecieran un forma de actualizar a TLS a partir de una conexión sin encriptación (*plaintext*), en vez de usar un puerto diferente para encriptar las comunicaciones (esto evitaría el uso de wrappers como Stunnel). SSL también puede ser usado para crear una red privada virtual (VPN), como en el caso de OpenVPN.

Desarrollado por Netscape, SSL versión 3.0 se publicó en 1996, que más tarde sirvió como base para desarrollar TLS versión 1.0, un estándar protocolo IETF definido por primera vez en el RFC 2246. Visa, MasterCard, American Express y muchas de las principales instituciones financieras han aprobado SSL para el comercio sobre Internet. SSL opera de una manera modular: sus autores lo diseñaron extensible, con soporte para compatibilidad hacia delante y hacia atrás, y negociación entre las partes (peer-to-peer). Algunas primeras implementaciones de SSL podían usar claves simétricas con un máximo de sólo 40-bit debido a las restricciones del gobierno de los Estados Unidos sobre la exportación de tecnología criptográfica. El gobierno de los Estados Unidos explícitamente impuso una clave de 40-bit lo suficientemente pequeña para ser “rota” por un ataque de fuerza bruta por las

agencias de seguridad nacional que desearan leer el tráfico encriptado, a la vez que representaban un obstáculo para atacantes con menos medios. Una limitación similar se aplicó a Lotus Notes en versiones para la exportación. Después de varios años de controversia pública, una serie de pleitos, y el reconocimiento del gobierno de Estados Unidos de cambios en la disponibilidad en el mercado de "mejores" productos criptográficos producidos fuera del país, las autoridades relajaron algunos aspectos de las restricciones de exportación. La limitación de claves de 40-bit en su mayoría ha desaparecido. Las implementaciones modernas usan claves de 128-bit (o más) para claves de cifrado simétricas.

La primera definición de TLS apareció en el **RFC 2246**: "The TLS Protocol Version 1.0" Otros RFC posteriores extendieron TLS:

- **RFC 2712**: "Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)". Las familias de cifrados de 40-bit definidas aparecen sólo para propósitos de documentación del hecho de que esas familias de códigos de cifrado han sido ya asignadas.
- **RFC 2817**: "Upgrading to TLS Within HTTP/1.1", explica cómo usar el mecanismo de actualización en HTTP/1.1 para iniciar TLS sobre una conexión TCP existente. Esto permite al tráfico HTTP inseguro y seguro compartir el mismo puerto conocido (en este caso, http: en el 80 en vez de https: en el 443).
- **RFC 2818**: "HTTP Over TLS", diferencia tráfico seguro de tráfico inseguro mediante el uso de un "puerto de servidor" diferente.
- **RFC 3268**: "AES Ciphersuites for TLS". Añade la familia de cifrado AES a los cifrados simétricos previamente existentes.
- **RFC 3546**: "Transport Layer Security (TLS) Extensions", añade un mecanismo para negociar extensiones de protocolos durante la inicialización de sesión y define algunas extensiones.
- **RFC 4279**: "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", añade tres conjuntos de nuevas familias de cifrados para que el protocolo TLS permita la autenticación basada en claves previamente compartidas.

TLS 1.1 es la nueva generación del protocolo TLS. TLS 1.1 clarifica algunas ambigüedades y añade cierto número de recomendaciones. La principal razón de esta nueva versión es un formato modificado para encriptación RSA anterior al uso de "*master secret*", que es parte del mensaje de intercambio de claves del cliente (si se usa RSA), para usar PKCS#1 versión 2.1, en detrimento de PKCS#1 versión 1.5 en TLS 1.0. La razón de dicho cambio es para protegerse contra ataques descubiertos por Daniel Bleichenbacher que podían lanzarse contra servidores TLS 1.0, usando PKCS#1 versión 1.5, que podrían fallar de diferentes formas dependiendo de si el formato desencriptado fuera correcto o no. Éste también incluye recomendaciones para evitar ataques remotos programados.

3.1.2. Directivas SSL

El módulo SSL (*mod_ssl*) proporciona soporte SSL v2/v3 y TLS v1 para el servidor http Apache. Este módulo utiliza OpenSSL para la criptografía y ha sido desarrollado por Ralf S. Engeschall a partir del trabajo realizado por Ben Laurie. Las diferentes directivas que se pueden configurar son las siguientes:

- ***SSLCACertificateFile***

  Especifica el fichero que contiene los certificados, codificados en formato PEM, de las Autoridades de Certificación (CA) válidas para la autentificación del cliente.

- ***SSLCACertificatePath***

  Especifica el directorio que contiene los certificados, codificados en formato PEM, de las Autoridades de Certificación (CA) válidas para la autentificación del cliente.

- ***SSLCARevocationFile***

  Especifica el fichero que contiene las listas de certificados revocados (CRLs), codificados en formato PEM, de las Autoridades de Certificación (CA) válidas para la autentificación del cliente.

- ***SSLCARevocationPath***

  Especifica el directorio que contiene las listas de certificados revocados (CRLs), codificados en formato PEM, de las Autoridades de Certificación (CA) válidas para la autentificación del cliente.

- ***SSLCertificateChainFile***

  Especifica el fichero que contiene los certificados, codificados en formato PEM, de las Autoridades de Certificación (CA) que forman la cadena del certificado del servidor.

- ***SSLCertificateFile***

  Especifica el fichero que contiene el certificado, codificado en formato PEM, del servidor.

- ***SSLCertificateKeyFile***

  Especifica el fichero que contiene la clave privada, codificada en formato PEM, del servidor. Si la clave privada (RSA o DSA) está encriptada, es necesario proporcionar la contraseña (*passphrase*) para iniciar el servidor.

- ***SSLCipherSuite***

  Especifica el algoritmo utilizado para el protocolo *handshake* cuando se inicia la conexión SSL: (i) RSA or Diffie-Hellman variants para *Key Exchange Algorithm*, (ii) RSA, Diffie-Hellman, DSS or none para *Authentication Algorithm*, (iii) DES, Triple-DES, RC4, RC2, IDEA or none para *Cipher/Encryption Algorithm*, y finalmente (iv) MD5, SHA or SHA1 para *MAC Digest Algorithm*

- ***SSLEngine***

  Habilita o deshabilita el protocolo SSL/TLS. Por defecto, el protocolo SSL/TLS está deshabilitado.

- ***SSLMutex***

  Especifica el método de sincronización de los procesos SSL del servidor Apache.

- ***SSLOptions***

  Especifica las diferentes opciones que se pueden aplicar a un determinado directorio. Cada opción puede ir precedida del símbolo (+) o del símbolo (-). Para añadir una opción, se utiliza el símbolo (+) y para suprimir una opción se utiliza el símbolo (-).

  – ***StdEnvVars***

  Esta opción se utiliza para proporcionar el conjunto de variables de entorno CGI/SSI, correspondientes al protocolo SSL

  – ***CompatEnvVars***

  Esta opción se utiliza para proporcionar el conjunto de variables de entorno CGI/SSI, compatibles con otras implementaciones SSL

  – ***ExportCertData***

  Esta opción se utiliza para proporcionar el conjunto de variables de entorno CGI/SSI, correspondientes a los certificados del servidor `SSL_SERVER_CERT` y del cliente `SSL_CLIENT_CERT` y `SSL_CLIENT_CERT_CHAIN`**n**

  – ***FakeBasicAuth***

  Cuando se especifica esta opción, el Subject Distinguished Name (DN) del certificado del cliente se utiliza como nombre de usuario para el proceso de autorización. Como contraseña, se utiliza `xxj31ZMTZzkVA` que corresponde a la encriptación DES de la palabra password.

  – ***StrictRequire***

  Establece restricciones de acceso en base a las directivas ***SSLRequireSSL*** o ***SSLRequire***

  – ***OptRenegotiate***

  Permite optimizar el proceso de renegociación para la conexión SSL.

- ***PassPhraseDialog***

  Normalmente, por razones de seguridad, el fichero que contiene la clave privada del certificado del servidor está encriptado. Para iniciar el servidor, se requiere la contraseña (*passphrase*) para desencriptar este archivo. Esta contraseña, se puede proporcionar de forma interactiva `builtin` o a través de la ejecución de un programa `exec:/path/to/program`

- ***SSLProtocol***

  Especifica el protocolo de conexión SSL que pueden utilizar los clientes: SSLv2, SSLv3, TLSv1, All

- ***SSLProxyCACertificateFile***

  Especifica el fichero que contiene los certificados, codificados en formato PEM, de las Autoridades de Certificación (CA) válidas para la autentificación del servidor remoto.

- ***SSLProxyCACertificatePath***

  Especifica el directorio que contiene los certificados, codificados en formato PEM, de las Autoridades de Certificación (CA) válidas para la autentificación del servidor remoto. Verificar el certificado del servidor remoto en el proceso de autentificación.

- ***SSLProxyCARevocationFile***

  Especifica el fichero que contiene las listas de certificados revocados (CRLs), codificados en formato PEM, de las Autoridades de Certificación (CA) válidas para la autentificación del servidor remoto.

- ***SSLProxyCARevocationPath***

  Especifica el directorio que contiene las listas de certificados revocados (CRLs), codificados en formato PEM, de las Autoridades de Certificación (CA) válidas para la autentificación del servidor remoto.

- ***SSLProxyCipherSuite***
  Especifica el algoritmo utilizado para el protocolo *handshake* cuando se inicia la conexión proxy SSL

- ***SSLProxyEngine***
  Habilita o deshabilita el protocolo SSL/TLS para la conexión proxy. Por defecto, el protocolo SSL/TLS para la conexión proxy está deshabilitado.

- ***SSLProxyMachineCertificateFile***
  Especifica el fichero que contiene los certificados y claves privadas de los clientes, codificados en formato PEM, utilizados por el servidor proxy en el proceso de autentificación con servidores remotos.

- ***SSLProxyMachineCertificatePath***
  Especifica el directorio que contiene los certificados y claves privadas de los clientes, codificados en formato PEM, utilizados por el servidor proxy en el proceso de autentificación con servidores remotos.

- ***SSLProxyProtocol***
  Especifica el protocolo SSL que pueden utilizar los clientes en una conexión proxy: SSLv2, SSLv3, TLSv1, All

- ***SSLProxyVerify***
  Establece el nivel de verificación de certificado para la autentificación de servidor remoto. Para el nivel `none` no se requiere el certificado del servidor remoto; en el nivel `optional` el servidor remoto puede presentar un certificado válido; en el nivel `require` el servidor remoto debe de presentar un certificado válido, y en el nivel `optional_no_ca` el servidor remoto puede presentar un certificado no emitido por una autoridad de certificación.

- ***SSLProxyVerifyDepth***
  Establece el número máximo de Autoridades de Certificación (CA) emisoras para la verificación del certificado del servidor remoto. El valor o corresponde a un certificado no emitido por una Autoridad de Certificación (*self-signed*).

- ***SSLRandomSeed***

  Requerido por OpenSSL, configura una o más fuentes/semillas para la generación de números pseudo-aleatorios (PRNG), cuando se inicia el servidor, o cuando se establece una conexión.

- ***SSLRequire***

  Esta directiva establece los requerimientos de acceso, principalmente en base al análisis de las variables de entorno CGI/SSL

- ***SSLRequireSSL***

  Deniega el acceso a todas las peticiones que no utilicen SSL (https)

- ***SSLSessionCache***

  Permite acelerar el procesamiento en paralelo de varias peticiones SSL. Se pueden definir dos técnicas para la sincronización de las memorias caché de los procesos del servidor: la técnica `dbm` utiliza el disco local y la técnica `shm` utiliza un segmento de memoria RAM compartida; `none` es el valor por defecto y equivale a no aplicar ninguna técnica.

- ***SSLSessionCacheTimeout***

  Establece el tiempo de validez de las memorias caché.

- ***SSLVerifyClient***

  Establece el nivel de verificación de certificado para la autentificación del cliente. Para el nivel `none` no se requiere el certificado del cliente. En el nivel `optional` el cliente puede presentar un certificado válido; en el nivel `require` el cliente debe de presentar un certificado válido, y en el nivel `optional_no_ca` el cliente puede presentar un certificado no emitido por una autoridad de certificación.

- ***SSLVerifyDepth***

  Establece el número máximo de Autoridades de Certificación (CA) emisoras para la verificación del certificado del cliente. El valor 0 corresponde a un certificado no emitido por una Autoridad de Certificación (*self-signed*).

3.1.3. Variables de entorno SSL

En este apartado se describen las posibles variables de entorno que se pueden utilizar para procesar las peticiones SSL

– Petición realizada utilizando el protocolo SSL
`HTTPS`
– Versión del protocolo SSL (SSLv2, SSLv3, TLSv1)
`SSL_PROTOCOL`
– Identificador, en hexadecimal, de la sesión SSL
`SSL_SESSION_ID`
– Especificación del cifrado
`SSL_CIPHER`
– Cifrado de exportación (`True`, `False`)
`SSL_CIPHER_EXPORT`
– Número de bits de cifrado utilizados
`SSL_CIPHER_USEKEYSIZE`
– Número de bits de cifrado posibles
`SSL_CIPHER_ALGKEYSIZE`
– Versión del módulo mod_ssl
`SSL_VERSION_INTERFACE`
– Versión de la librería OpenSSL
`SSL_VERSION_LIBRARY`
– Versión del certificado del cliente
`SSL_CLIENT_M_VERSION`
– Número de serie del certificado del cliente
`SSL_CLIENT_M_SERIAL`
– Subject DN (*Distinguished Name*) del certificado del cliente
`SSL_CLIENT_S_DN`
– Componente **`x509`** del Subject DN del certificado del cliente
`SSL_CLIENT_S_DN_`**`x509`**
– DN del emisor del certificado del cliente
`SSL_CLIENT_I_DN`
– Componente **`x509`** del DN del emisor del certificado del cliente
`SSL_CLIENT_I_DN_`**`x509`**

– Fecha de inicio de la validez del certificado del cliente

`SSL_CLIENT_V_START`

– Fecha de finalización de la validez del certificado del cliente

`SSL_CLIENT_V_END`

– Algoritmo utilizado para la firma del certificado del cliente

`SSL_CLIENT_A_SIG`

– Algoritmo utilizado para la generación de la clave pública del certificado del cliente

`SSL_CLIENT_A_KEY`

– Certificado del cliente, codificado en formato PEM

`SSL_CLIENT_CERT`

– Certificado en formato PEM, dentro de la cadena del certificado del cliente

`SSL_CLIENT_CERT_CHAIN`**`n`**

– Verificación del certificado del cliente (`NONE`, `SUCCESS`, `FAILED`: razón)

`SSL_CLIENT_VERIFY`

– Versión del certificado del servidor

`SSL_SERVER_M_VERSION`

– Número de serie del certificado del servidor

`SSL_SERVER_M_SERIAL`

– Subject DN (*Distinguished Name*) del certificado del servidor

`SSL_SERVER_S_DN`

– Componente **`x509`** del Subject DN del certificado del servidor

`SSL_SERVER_S_DN_`**`x509`**

– DN del emisor del certificado del servidor

`SSL_SERVER_I_DN`

– Componente **`x509`** del DN del emisor del certificado del servidor

`SSL_SERVER_I_DN_`**`x509`**

– Fecha de inicio de la validez del certificado del servidor

`SSL_SERVER_V_START`

– Fecha de finalización de la validez del certificado del servidor

`SSL_SERVER_V_END`

– Algoritmo utilizado para la firma del certificado del servidor

`SSL_SERVER_A_SIG`

– Algoritmo para la generación de la clave pública del certificado del servidor

`SSL_SERVER_A_KEY`

– Certificado del servidor, codificado en formato PEM

`SSL_SERVER_CERT`

Los componentes `x509` pueden ser C, ST, L, O, OU, CN, UID, Email: Country (C), State/Province (ST), Locality (L), Organization (O), Organizational Unit (OU), Common Name (CN), User ID (UID), Email

3.1.4. Configuración de directivas

El acceso a la información de las páginas Web y de los módulos programados, se realiza en base a la configuración de directivas del servidor Web. Dentro del conjunto de directivas definidas en los ficheros de configuración del servidor, se han configurado las siguientes directivas:

Directivas de configuración global

- Directorio de nivel superior, donde están ubicados los ficheros de configuración
  `ServerRoot "/etc/httpd"`
- Tiempo límite de respuesta del servidor para una petición (en segundos)
  `Timeout 300`
- Incluir los ficheros de configuración correspondientes a los módulos SSL y Perl
  `Include conf.d/*.conf`
- Dirección de correo electrónico del administrador del servidor Web
  `ServerAdmin manager@iaa.es`
- Nombre del servidor (*Full Qualified Domain Name*)
  `ServerName irisgrid.iaa.es`
- Identificar las peticiones utilizando el valor especificado en `ServerName`
  `UseCanonicalName On`
- Directorio de nivel superior, donde están ubicadas las páginas Web
  `DocumentRoot "/home/web/https"`
- Directorio correspondiente a las aplicaciones que puede ejecutar el servidor Web
  `ScriptAlias /globus/ "/home/web/globus/"`
- Archivo con la información de control de acceso al directorio donde esta ubicado
  `AccessFileName .htaccess`
- Deshabilitar el acceso a las áreas de usuario
  `UserDir disable`

- Procesar las directivas `include` definidas dentro de las páginas Web

  ```
  XBitHack on
  ```

- Definir parámetros restrictivos para el acceso al directorio raíz del sistema

  ```
  <Directory />
  ```

  – Ninguna opción adicional disponible para el servidor Web

  ```
  Options None
  ```

  – Ignorar cualquier opción definida en los ficheros `.htaccess`

  ```
  AllowOverride None
  </Directory>
  ```

- Definir los parámetros de acceso al directorio definido por `DocumentRoot`

  ```
  <Directory /home/web/https>
  ```

  – Permitir el uso de directivas `include` dentro de las páginas Web

  ```
  Options Includes
  ```

  – Ignorar cualquier opción definida en los ficheros `.htaccess`

  ```
  AllowOverride None
  ```

  – Orden en el que se evalúan las directivas `allow`, `deny`

  ```
  Order allow,deny
  ```

  – Permitir el acceso a cualquier petición

  ```
  Allow from all
  </Directory>
  ```

- Definir los parámetros de acceso al directorio definido por `ScriptAlias`

  ```
  <Directory "/home/web/globus">
  ```

  – Ninguna opción adicional disponible para el servidor Web

  ```
  Options None
  ```

  – Ignorar cualquier opción definida en los ficheros `.htaccess`

  ```
  AllowOverride None
  ```

  – Orden en el que se evalúan las directivas `allow, deny`

  ```
  Order allow,deny
  ```

  – Permitir el acceso a cualquier petición

  ```
  Allow from all
  </Directory>
  ```

Directivas de configuración SSL

- Cargar el módulo Dynamic Shared Object (DSO) correspondiente

  ```
  LoadModule ssl_module modules/mod_ssl.so
  ```

- Definir el puerto de comunicaciones para la aceptación de peticiones SSL

  ```
  Listen 443
  ```

- Definir el Host Virtual para el protocolo https

  ```
  <VirtualHost _default_:443>
  ```

  – Activar el módulo SSL

  ```
  SSLEngine on
  ```

  – Fichero que contiene el certificado, codificado en formato PEM, del servidor

  ```
  SSLCertificateFile /etc/httpd/conf/ssl.crt/server.crt
  ```

  – Fichero que contiene la clave privada del servidor (codificada en formato PEM)

  ```
  SSLCertificateKeyFile /etc/httpd/conf/ssl.pem/server.pem
  ```

  – Por defecto no se requiere el certificado para la autentificación del cliente

  ```
  SSLVerifyClient none
  ```

  – Proporcionar el conjunto de variables de entorno SSL para `ScriptAlias`

  ```
  <Directory /home/web/globus>
  SSLOptions +StdEnvVars
  </Directory>
  ```

- Directorio donde se requiere un certificado válido para la autentificación del cliente.

  ```
  <Directory /home/web/globus/required>
  ```

  – La directiva `SSLVerifyClient` establece el nivel de acreditación del usuario.

  ```
  SSLVerifyClient optional
  ```

  – Se ha configurado con el valor `optional`, para que el usuario pueda presentar un certificado válido cuando le sea requerido. Si el usuario no tiene instalado en su navegador un certificado válido, no podrá tener acceso a este directorio.

  ```
  SSLVerifyDepth 1
  SSLCACertificateFile /etc/httpd/conf/ssl.crt/ca.crt
  </Directory>
  </VirtualHost>
  ```

El proceso para la generación del certificado para el servidor Web es el siguiente:

– Generación de la clave privada

```
openssl genrsa -des3 -out server.key 2048
```

– Eliminar la contraseña (*passphrase*) de la clave privada RSA

```
openssl rsa -in server.key -out server.pem
```

– Generar el CSR (*Certificate Signing Request*)

```
openssl req -new -key server.key -out server.csr
```

– Firmar el certificado (*Self-Signed Certificate*)

```
openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

– Instalar la clave privada en el directorio

```
/etc/http/conf/ssl.key
```

– Instalar el certificado en el directorio

```
/etc/http/conf/ssl.crt
```

Directivas de configuración Perl

- Cargar el módulo Dynamic Shared Object (DSO) correspondiente

```
LoadModule perl_module modules/mod_perl.so
```

3.1.5. Configuración de directorios

La distribución de los ficheros correspondientes al diseño y programación de las páginas Web del portal y los módulos desarrollados para la integración con los servicios Globus es la siguiente:

- ***/home/web/https***

  Directorio donde se ubican las páginas Web correspondientes al portal. Los ficheros `index.html` contienen el código html de las páginas y los ficheros `index.gif` se corresponden con el título de las páginas. La página principal corresponde a las instrucciones de acceso al portal.

  – ***certificate***

  En este subdirectorio se ubican los ficheros de la página correspondiente a las instrucciones de solicitud de certificado de usuario.

– ***register***

En este subdirectorio se ubican los ficheros de las páginas correspondientes a las instrucciones de solicitud de registro en una Organización Virtual.

– ***include***

En este subdirectorio se ubican los ficheros requeridos por el servidor Web para la creación de las páginas Web. El diseño de las páginas se simplifica con el uso de la directiva SSI (Server Side Include), utilizando para todas las páginas las siguientes directivas:

```
<!--#include virtual="/include/header.html" -->
<!--#include virtual="/include/footer.html" -->
```

El fichero `header.html` contiene el código html correspondiente al diseño de la cabecera de todas las páginas. En este fichero se hace referencia al fichero `style.css`, en el que se define la hoja de estilos que utilizan todas las página y el fichero `scripts.js`, correspondiente al código de las funciones JavaScript que utilizan los módulos programados en Perl. El fichero `footer.html` contiene el código html correspondiente al pié de página.

– ***include/img***

Subdirectorio de imágenes correspondientes al diseño del marco de información de la página Web (`frame`), a los títulos (`labels`) y logotipos (`logos`) que aparecen en las cabeceras de las páginas Web y a los menús de opciones (`options`), ubicados en el margen izquierdo de la página.

- ***/home/web/globus***

Directorio donde se ubican los módulos programados que pueden ser requeridos por el usuario.

– ***required***

Para acceder a los módulos ubicados en este subdirectorio, es necesario tener instalado en el navegador un certificado de usuario firmado por una autoridad de certificación válida.

- ***/home/web/task***

  Directorio donde se ubican los módulos programados correspondientes a los servicios Globus. El acceso a estos módulos sólo puede realizarse a través del módulo principal `task` en base a las peticiones de usuario. Los módulos se ejecutan utilizando las credenciales del usuario.

- ***/home/web/certs***

  Directorio donde se ubican los certificados de usuario. El fichero `cdb` corresponde a la base de datos de certificados de usuario válidos, y el fichero `rdb` corresponde a la base de datos de solicitudes de creación de cuenta de usuario. Cuando un usuario se registra en el sistema (UI), no se crea explícitamente una cuenta de usuario, sino una correspondencia entre las credenciales de usuario y un identificador asociado a dichas credenciales. La información correspondiente a los registros de la bases de datos de certificados es la siguiente:

  ```
  uid:Certificate Distinguished Name:email
  ```

  – ***request***

  Subdirectorio donde se registran las solicitudes de creación de cuenta de usuario y de renovación de certificado.

  – ***uid***

  Para cada identificador de usuario, se crea el subdirectorio correspondiente, donde se registran las credenciales de usuario.

- ***/home/web/jobs***

  Para cada identificador de usuario, se crea el correspondiente subdirectorio donde se registran los trabajos enviados al Grid por el usuario. Cuando un usuario solicita enviar de un trabajo al Grid, el sistema comprueba que hay recursos disponibles para la ejecución del trabajo y en su caso, lo registra asignándole un identificador único que el usuario puede utilizar para consultar su estado, monitorizar su ejecución, cancelar su ejecución y obtener el resultado de su ejecución.

Dentro del área de usuario, se crea un subdirectorio para cada trabajo utilizando este identificador, donde se registran el fichero `submit.jdl` que corresponde a la descripción del trabajo (**J**ob **D**escription **L**anguage), el fichero `stdout` relativo a los datos obtenidos de la ejecución del trabajo y el fichero `stderr` correspondiente a la información de error resultante de la ejecución del trabajo.

- ***/tmp/log***
  Directorio correspondiente a la información proporcionada por los servicios Globus. Cuando un usuario realiza una petición de servicio Globus, se crea el fichero de reporte de información correspondiente. Este fichero, asociado al identificador de usuario, lo procesa el sistema para mostrar al usuario la información resultante de la petición.

- ***/tmp/jobInput***
  En este directorio, se registra la información del trabajo proporcionada por el usuario, para su análisis antes de ser enviado al Grid.

- ***/tmp/jobOutput***
  En este directorio se almacenan, antes de ser registrados en el área de usuario, los ficheros correspondientes a la solicitud del resultado de la ejecución de un trabajo.

3.2. Hojas de estilos CSS

Para el diseño de las páginas Web se han utilizado hojas de estilos en cascada (Cascading Style Sheets, CSS). Las hojas de estilos CSS son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML).

El W3C (World Wide Web Consortium) es el encargado de formular la especificación de las hojas de estilo que servirá de estándar para los navegadores. El objetivo es separar la estructura de un documento de su presentación. La información de estilo se puede adjuntar como un documento separado o en el mismo documento HTML.

En este último podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo `style`.

Las ventajas de utilizar hojas de estilo CSS son las siguientes: (i) Control centralizado de la presentación de un sitio Web completo con lo que se agiliza de forma considerable la actualización del mismo. (ii) Los Navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio Web remoto, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces. (iii) Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario (p.e. impresora, sintetizador de voz, dispositivo móvil ...). (iv) El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

Hay varias versiones de hojas de estilo CSS: CSS1 y CSS2, con CSS3 en desarrollo por el World Wide Web Consortium (W3C). Los navegadores modernos implementan CSS1 bastante bien, aunque existen pequeñas diferencias de implementación según marcas y versiones de los navegadores. CSS2, sin embargo, está solo parcialmente implementado en los más recientes. La maquetación CSS consiste en utilizar capas (*layers*) que en HTML se definen con las marcas `<div></div>` para estructurar la aplicación Web, como alternativa a las tablas y los marcos (frames). Así, podemos nombrar cada capa con el atributo `id` de manera que podremos definir las propiedades de dicha capa en el archivo CSS correspondiente.

Las clases definidas en la hoja de estilos `style.css` utilizadas para la generación de las páginas Web del portal son las siguientes:

- Estilo por defecto para el contenido de la página

```
font-family: verdana,tahoma,arial;
background-color: #003333;
overflow: hidden;
margin: 0 0 0 0;
color: #333333;
```

- Estilo definido para un párrafo, una lista, y una fila/columna de una tabla

```
font-family: verdana,tahoma,arial;
list-style-position: inside;
line-height: 18px;
font-size: 12px;
margin: 0;
```

- Estilo definido para el salto de línea

```
line-height: 8px
```

- Estilo definido para la línea horizontal

```
height: 2px
```

- Capa para la visualización de información con barras de desplazamiento

```
div.scroll { height: 242px; width: 580px; clear: both; overflow: auto }
```

- Estilos definidos para los enlaces

```
a:link    { color: #006666; font-weight: bold; text-decoration: none }
a:visited { color: #006666; font-weight: bold; text-decoration: none }
a:hover { color: #000000; font-weight: bold; text-decoration: underline }
```

- Clase para visualizar texto intermitente

```
.blink { text-decoration: blink; font-weight: bold }
```

- Clase para el título de una sección

```
.label {
  padding: 3px 15px 4px 15px;
  background-color: #003333;
  font-weight: bold;
  color: #cccc99;
}
```

- Clase para el campo de formulario de tipo `button`

```
.button {
  padding: 0px 12px 2px 12px;
  background-color: #cccc99;
  font-weight: bold;
  color: #003333;
}
```

- Clase para el campo de formulario de tipo `input`

```
.input {
  padding: 1px 5px 1px 5px;
  border: 1px solid #999966;
  background-color: #ffffcc;
  color: #333333;
  height: 20px;
}
```

- Clase para el campo de formulario de tipo `select`

```
.select {
  border: 1px solid #999966;
  background-color: #ffffcc;
  color: #333333;
}
```

- Clase para visualizar el tiempo restante para la realización de una nueva petición

```
.clock {
  position: absolute; top: 140; left: 810;
  padding: 3px 5px 4px 0px;
  background-color: #003333;
  font-weight: bold;
  color: #cccc99;
  direction: rtl;
  border: 0px;
  width: 30px;
}
```

- Clase para ocultar el estilo por defecto del campo de formulario de tipo `file`

```
.hide {
  opacity: 0; -moz-opacity: 0; filter: alpha(opacity=0);
  position: absolute;
  font-size: 14px;
}
```

- Clase para visualizar el mensaje de solicitud de servicio `info`

```
.pop { position: absolute; top: 85; left: 400 }
```

- Clase para ocultar el mensaje de solicitud de servicio `info`

```
.off { position: absolute; top: -1000; left: -1000 }
```

- Clase para ocultar el campo asociado al identificador del trabajo `jobId`

```
.off { position: absolute; top: -1000; left: -1000 }
```

3.3. Acceso autorizado

Esta sección corresponde a la primera fase de desarrollo del portal; en esta fase se desarrollan las páginas Web correspondientes a los requerimientos de acceso a los recursos Grid. El objetivo de esta fase es proporcionar al usuario toda la información necesaria para el acceso a los servicios Globus proporcionados por el portal.

En la página principal del portal *https://irisgrid.iaa.es* se informa al usuario de las instrucciones de acceso a los servicios proporcionados:

- ***Access Requirements***
  Información correspondiente a los requerimientos de acceso a los recursos de IRISGrid. Para utilizar los recursos, el usuario debe obtener un certificado digital de la Autoridad de Certificación pkIRISGridCA, registrarse en una Organización Virtual (VO) y obtener una cuenta en el Interfaz de usuario.

- ***Authorized Access***
  El acceso a los servicios Grid sólo es posible si el usuario tiene instalado en su navegador el certificado firmado por la autoridad de certificación pkIRISGridCA. Para completar el acceso, el usuario debe seleccionar una Organización Virtual (en la que previamente se haya registrado), y debe proporcionar la contraseña (*passphrase*) de la clave privada de su certificado.

El usuario puede acceder a la información específica de cada requerimiento, a través del menú de opciones que se ofrece en esta página:

- ***Certificate***
  Esta opción nos proporciona la información necesaria para la solicitud de certificado de usuario X.509 a la Autoridad de Certificación pkIRISGridCA.

- ***Register***
  Esta opción nos proporciona la información necesaria para el registro en una Organización Virtual (VO).

- ***Account***

  Esta opción nos proporciona la información necesaria para la solicitud de cuenta de usuario en el Interfaz de Usuario.

- ***Access***

  Esta opción nos permite regresar a la página de acceso.

Fig 3.3. *Acceso autorizado*

3.3.1. Certificado de usuario

Para autenticarse con los recursos Grid, el usuario necesita tener un certificado X.509 firmado por una Autoridad de Certificación (CA) reconocida por LCG.

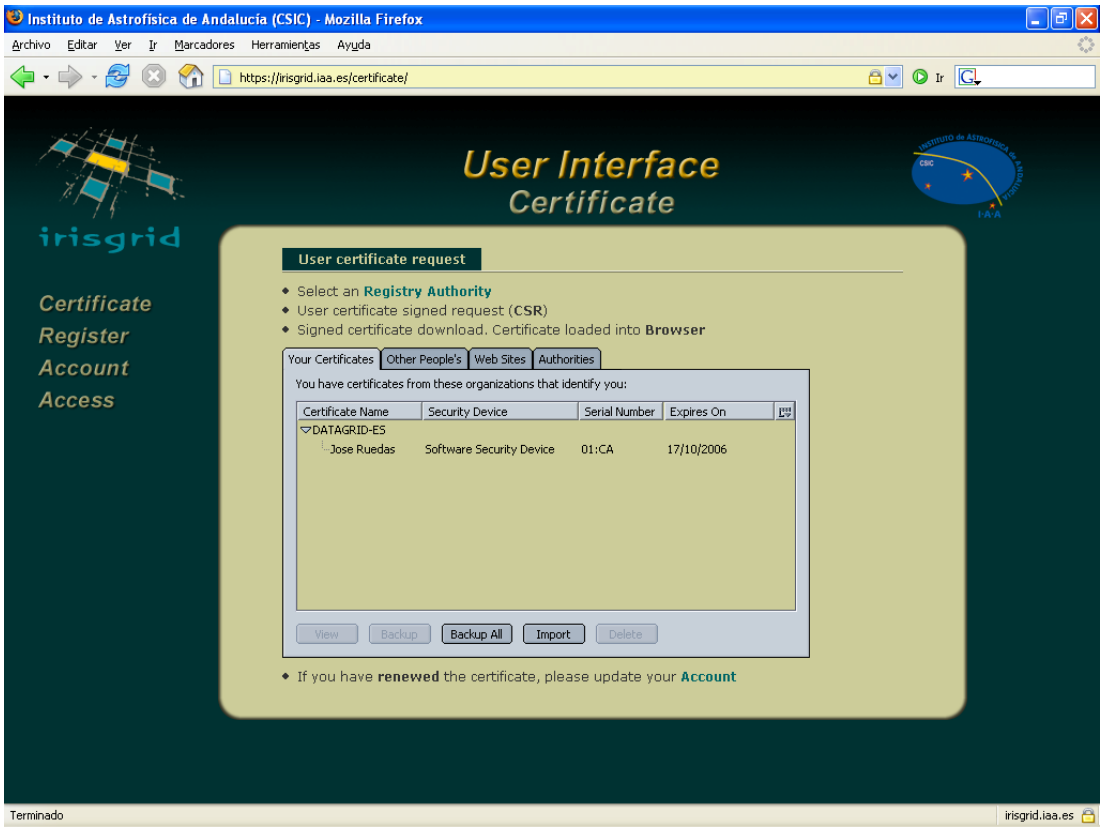


Fig 3.3.1a. *Solicitud de certificado de usuario X.509*

Los pasos necesarios para la obtención del certificado de usuario son los siguientes:

- ***Elección de la Autoridad de Registro*** (RA)
  Debe elegir la RA correspondiente a su organización.

- ***Solicitud de certificado*** (CSR)
  – El identificador IRISGrid, identifica a una persona dentro de IRISGrid. Está formado por un código de usuario y un código de organización, separados por el carácter "@".

– El usuario deberá proporcionar su información personal (Nombre y Apellidos)

– La Clave de Usuario le servirá para verificar su identidad ante una petición de su Autoridad de Registro (RA).

– Si desea incluir su email en el certificado, debe indicarlo en la opción correspondiente a privacidad. En este caso dicha dirección será incluida en el campo X509v3 Subject Alternative Name. Con la inclusión de la dirección de correo electrónico en el certificado estamos haciendo público nuestro mail y podemos ser objetivo del SPAM.

– El PIN corresponde a la contraseña (*passphrase*)que se utilizará para generar el certificado. El PIN le permitirá posteriormente renovar o revocar el certificado.

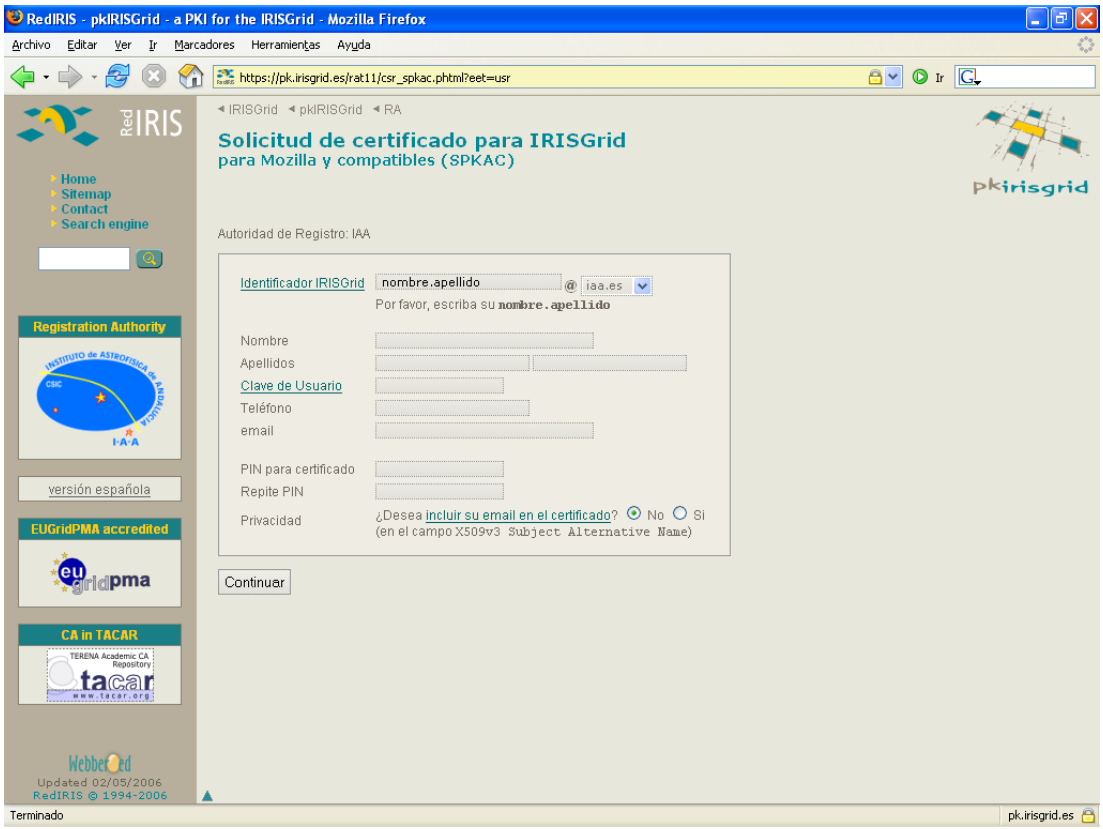


Fig 3.3.1b. *Solicitud de certificado de usuario X.509 a pkIRISGridCA*

- ***Descarga del certificado solicitado*** (una vez firmado por la CA)
  Una vez realizada la solicitud, el administrador de la Autoridad de Registro debe de confirmar los datos proporcionados por el usuario y autorizar la firma del certificado. Una vez firmado el certificado pkIRISGrid notificará por correo electrónico al usuario la disponibilidad del mismo.

```
Subject: pkIRISGrid CA: Certificado disponible para su descarga
From: pkirisgrid@rediris.es

To: ruedas@iaa.es

Ha sido generado un certificado para el identificador IRISGrid: jose.ruedas@iaa.es

Puede descargarlo desde:
https://pk.irisgrid.es/rat11/crt_get.phtml
```

  Cuando el usuario solicita el certificado, el navegador genera la clave privada del mismo. Por tanto, para descargar el certificado, el usuario debe de utilizar el mismo navegador desde el que solicitó el certificado (suponiendo que no ha sido reinstalado). Para la descarga del certificado, el usuario deberá indicar su identificador IRISGrid.

- ***Revocar certificado***
  Si por causas de seguridad desea anular el certificado, debe indicar su identificador IRISGrid y el correspondiente PIN del certificado.

3.3.2. Registro en una Organización Virtual

LCG-2 está organizado en Organizaciones Virtuales (VO). Cada VO agrupa usuarios, instituciones y recursos dentro del mismo dominio de administración.

El registro en una VO, sólo es posible si el usuario tiene instalado en su navegador el certificado firmado por la autoridad de certificación pkIRISGridCA.

Antes de que los recursos LCG puedan ser utilizados, el usuario debe aceptar la normativa de uso de LCG, y debe registrarse en una VO a través del correspondiente servicio de registro. La solicitud de registro en una VO, será evaluada por el manager de la VO, notificando al usuario si se acepta o no la solicitud.

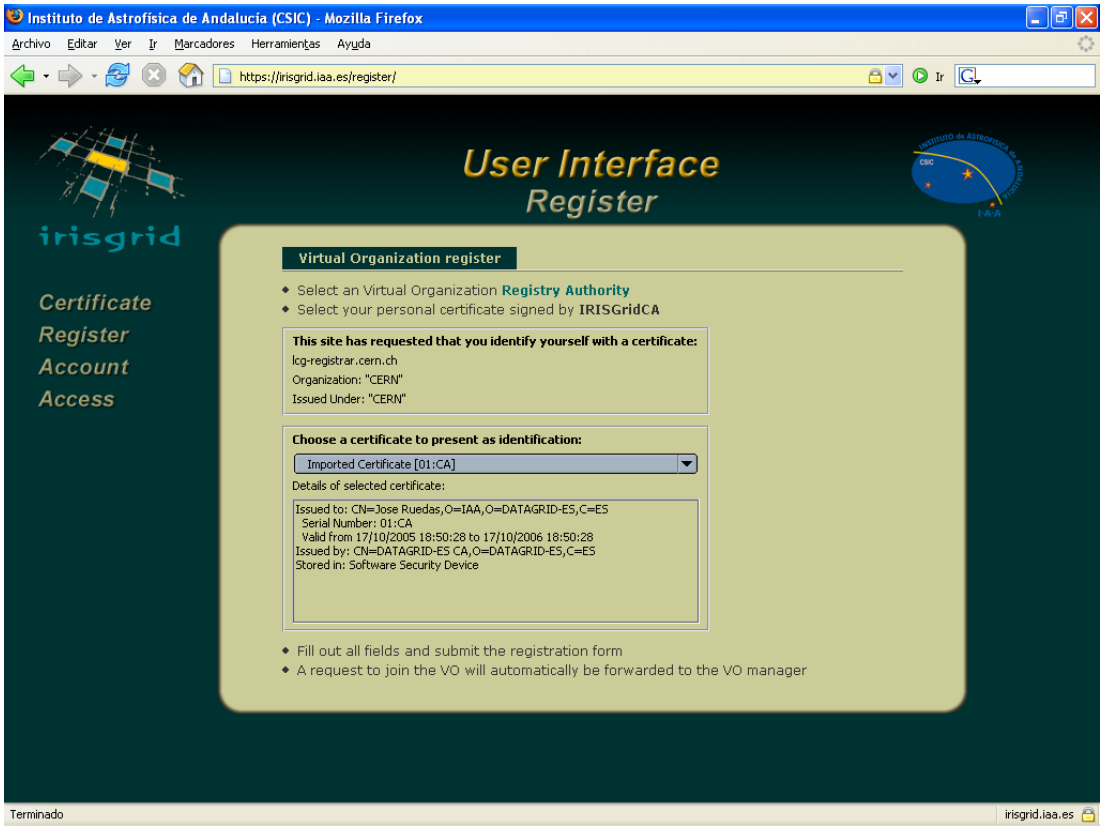


Fig 3.3.2a. *Registro en una Organización Virtual (VO)*

Una vez aceptada la solicitud, la autorización de un usuario en un recurso específico del Grid, se puede realizar de dos formas diferentes:

- La primera es la más simple, y se basa en el mecanismo de ***grid-mapfile***

  Cada recurso Grid tiene un grid-mapfile local que asocia los certificados de usuario a cuentas locales del sistema. Cuando un usuario realiza una petición de servicio a un recurso Grid, el subject del certificado del usuario (proporcionado por el certificado proxy) se comprueba que está en el grid-mapfile local y se obtiene en su caso la cuenta local asociada. Con esta cuenta se realiza la petición de servicio.

- La segunda se basa en el servicio VOMS y en el mecanismo LCAS/LCMAPS, que permite una definición más detallada de los privilegios de usuario.

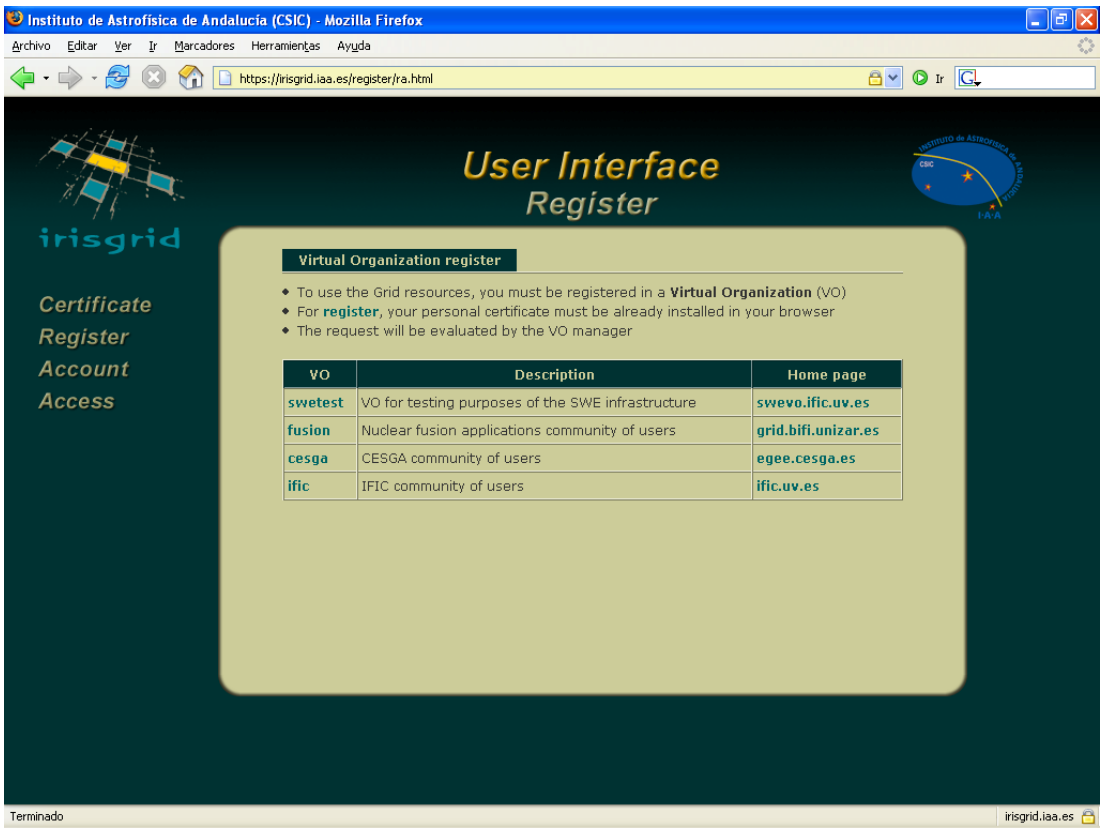


Fig 3.3.2b. *Selección de una Organización Virtual (VO)*

3.3.3. Solicitud de cuenta de usuario

El último paso es la solicitud de una cuenta de usuario en el User Interface. La solicitud sólo se puede realizar si el usuario tiene instalado en su navegador el certificado firmado por la autoridad de certificación pkIRISGridCA.

A través de esta opción, los usuarios registrados en el UI pueden actualizar sus credenciales (renovación del certificado).

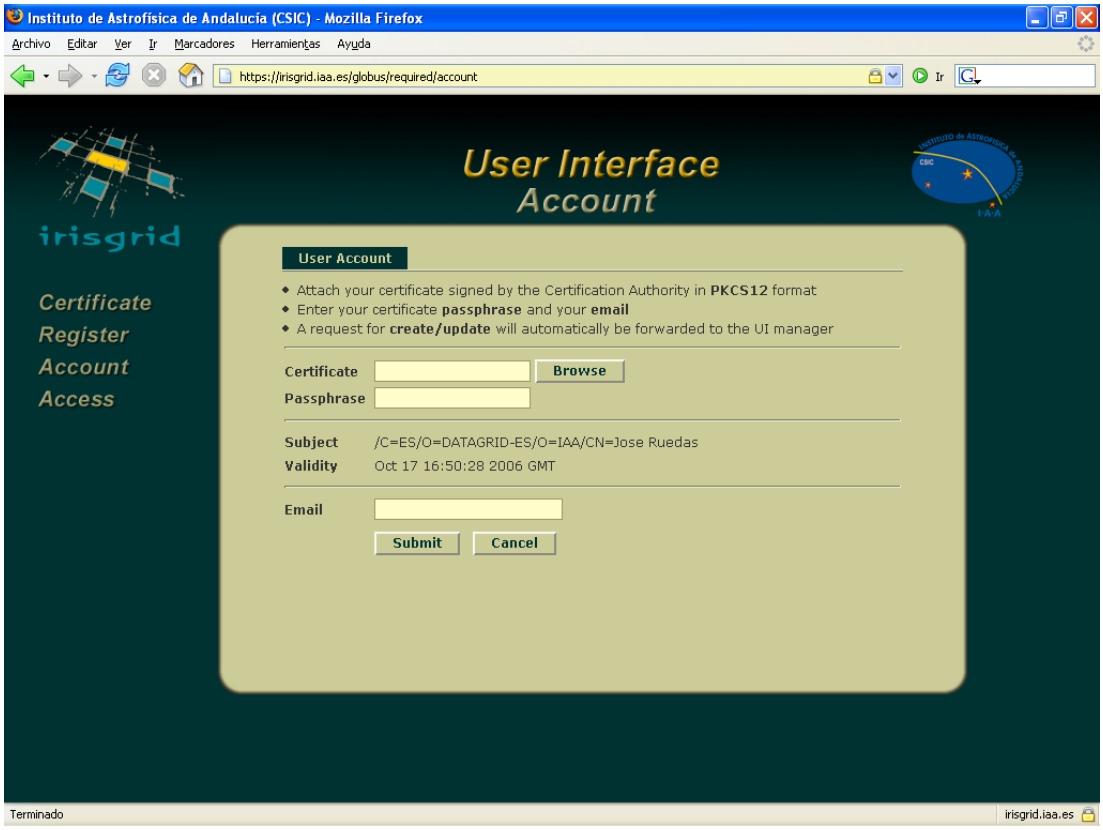


Fig 3.3.3. *Solicitud de cuenta de usuario*

Para solicitar una cuenta en el UI o para renovar sus credenciales, el usuario debe:

- Adjuntar, en formato PKCS12, su certificado firmado por la autoridad de certificación pkIRISGridCA. El navegador permite al usuario exportar los certificados en formato PKCS12.

- Especificar la contraseña correspondiente a la clave privada de su certificado. Algunos navegadores requieren una contraseña para exportar el certificado. En este caso, el usuario debe de especificar la misma contraseña que utilizó para generar la clave privada.

- Especificar su dirección de correo electrónico.

La solicitud se registra en el UI y se le notifica al usuario por correo electrónico. Para activar la cuenta, el usuario deberá confirmar la petición conectándose a la dirección especificada en el mensaje. De esta forma el sistema verifica la validez de la dirección de correo proporcionada por el usuario.

```
Subject: [UI] Account Request
From: manager@iaa.es

To: ruedas@iaa.es

A request for an user account on UI irisgrid.iaa.es has been made using this email address.

  Subject...: /DC=es/DC=irisgrid/O=iaa/CN=jose.ruedas
  Validity..: Mar  1 12:54:37 2007 GMT

Please click on the following URL to confirm this request:
https://irisgrid.iaa.es/globus/required/register?request=b111e482-605b-441a-ae2c-636aa2b9806a

If you have not made this request, please contact with the UI manager as soon as possible.

Thank you,
manager@iaa.es
```

Una vez confirmada la petición, se registran los archivos correspondientes al certificado de usuario y su clave privada, notificando al usuario que dispone de acceso autorizado.

```
Subject: [UI] Authorized Access
From: manager@iaa.es

To: ruedas@iaa.es

Your user account on UI irisgrid.iaa.es has been created.

Please keep the following information for future reference
Request: a08d15df-b4f5-467e-88e9-4753876304b9

Thank you,
manager@iaa.es
```

3.4. Servicios proporcionados por el portal

Esta sección corresponde a la segunda fase de desarrollo del portal. En esta fase se desarrollan las páginas Web relativas a los servicios Globus proporcionados por el portal.

3.4.1. Relación de servicios

Una vez realizado el acceso autorizado, el UI con las credenciales de usuario, genera y firma un certificado temporal, denominado ***proxy***, para autentificar de forma segura al usuario en cada interacción con los recursos Grid.

Si el proceso de creación del certificado proxy ha sido correcto, el usuario podrá comprobar a pié de página la validez del certificado.

Fig 3.4.1. *Relación de servicios proporcionados por el portal*

A través del menú de opciones, el usuario puede solicitar los siguientes servicios:

- ***Certificate*** **-** Esta opción permite al usuario establecer el periodo de validez del certificado proxy (***Change***).

- ***Resources*** **-** Esta opción permite al usuario obtener una relación de todos los recursos computacionales disponibles en el Grid (***Computer***). Dentro de esta opción, el usuario puede solicitar la relación de todos los recursos de almacenamiento disponibles en el Grid (***Storage***).

- ***Submit*** **-** Esta opción permite al usuario enviar al Grid un trabajo, para su ejecución en un recurso computacional disponible.

- ***Status*** **-** Esta opción permite al usuario obtener el estado de los trabajos enviados al Grid, en el periodo de tiempo solicitado (***Query***). Dentro de esta opción, el usuario puede:

  – Solicitar la información correspondiente a la ejecución de un trabajo (***Monitor***)
  – Cancelar la ejecución de un trabajo (***Cancel***)
  – Solicitar el resultado de la ejecución del trabajo (***Retrieve***)

- ***Logout*** **-** Esta opción permite al usuario finalizar la sesión, revocando el certificado proxy creado al inicio de la misma.

3.4.2. Certificado proxy

La opción ***Certificate*** permite al usuario establecer el periodo de validez del certificado proxy. Por defecto, el UI genera un certificado proxy con una validez de 24 h. Esta validez se puede modificar para aquellos trabajos que requieran un tiempo de ejecución superior.

Para modificar la validez del certificado proxy, el usuario debe de proporcionar el periodo de validez en términos de días (***days***) y horas (***hours***) y la contraseña de su certificado (***passphrase***)

Si el proceso de generación del nuevo certificado proxy ha sido correcto, el usuario podrá comprobar, a pié de página, la nueva validez del certificado.

Fig 3.4.2. *Cambio de validez del certificado proxy*

3.4.3. Recursos computacionales

Cuando el usuario selecciona la opción ***Resources***, se genera una relación de todos los recursos computacionales disponibles en el Grid.

Para los recursos computacionales se proporciona la información correspondiente al identificador de recurso, nombre de recurso, gestor de trabajos, número de CPUs total y disponibles, y número de trabajos aceptados y en ejecución. La información obtenida la proporciona el Information Service (IS):

- ***Computer Element (CE)***
  Se construye a partir de un conjunto de nodos denominados Worker Nodes (WN), un Local Resource Management System (LRMS), y un nodo que actúa como Grid Gate (GG) o Gatekeeper, que es el nodo visible para el Grid.

  El nodo Grid Gate debe ser accesible desde el Grid. Este nodo es el responsable de aceptar trabajos y distribuirlos para su ejecución en los WNs. El nodo GG proporciona un interfaz uniforme a los recursos computacionales que gestiona.

  En todos los nodos WNs, están disponibles los comandos y librerías necesarias para realizar acciones en los recursos Grid. Cada organización LCG-2, tiene al menos un CE y un conjunto de WNs.

- ***Job Manager***
  En LCG-2, los LRMS (***Job Manager***) soportados son Portable Batch System (PBS), Load Sharing Facility (LSF), Torque y Condor.

- ***CPUs, Free, Jobs, Run***

  – ***CPUs*** número de Worker Nodes (WNs) que forman el CE
  – ***Free*** número de WNs disponibles para ejecutar trabajos
  – ***Jobs*** número de trabajos enviados al CE pendientes de ser ejecutados
  – ***Run*** número de trabajos en ejecución.

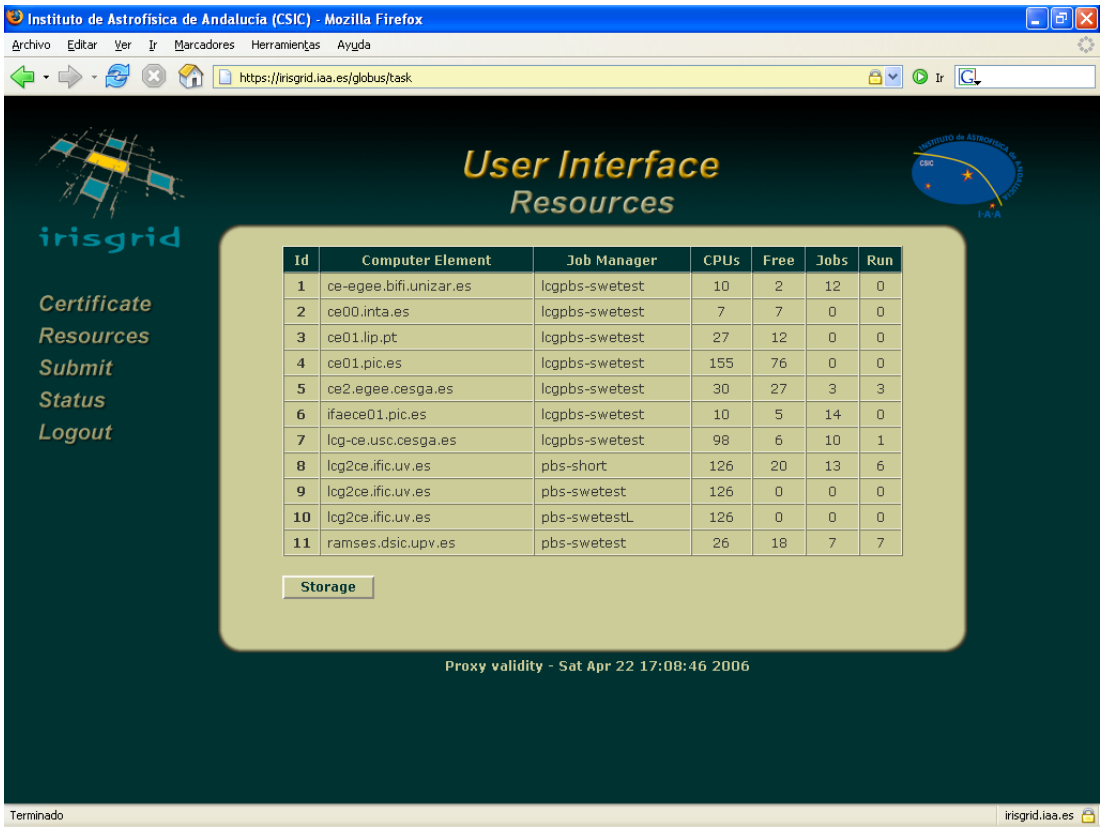


Fig 3.4.3a. *Información de los recursos computacionales disponibles*

Dentro de esta opción, el usuario puede solicitar la información correspondiente a los recursos de almacenamiento disponibles en el Grid. Cuando el usuario solicita esta información, se genera una relación de todos los recursos de almacenamiento disponibles en el Grid.

Para los recursos de almacenamiento se proporciona la información correspondiente al identificador de recurso, nombre de recurso, capacidad utilizada y capacidad disponible. La información obtenida la proporciona el Information Service (IS):

- ***Storage Element (SE)***
  Nombre del recurso de almacenamiento. El SE proporciona acceso uniforme a recursos de almacenamiento. Cada organización LCG-2 proporciona uno o más SEs.

- ***Type***

  El SE puede controlar servidores de disco, arrays de discos o Mass Storage Systems (MSS). Los SEs pueden soportar diferentes interfaces y protocolos de acceso. ***GSIFTP*** es el protocolo para la transferencia de ficheros y ***RFIO*** es utilizado para el acceso a ficheros. Algunos recursos de almacenamiento, están gestionados por Storage Resource Manager (***SRM***), que permite gestionar los contenidos de los recursos de almacenamiento.

- ***Available (GB)***

  Este parámetro indica, en GigaBytes, la capacidad de almacenamiento disponible en el recurso SE.

- ***Used (GB)***

  Este parámetro indica, en GigaBytes, el almacenamiento del recurso SE utilizado.

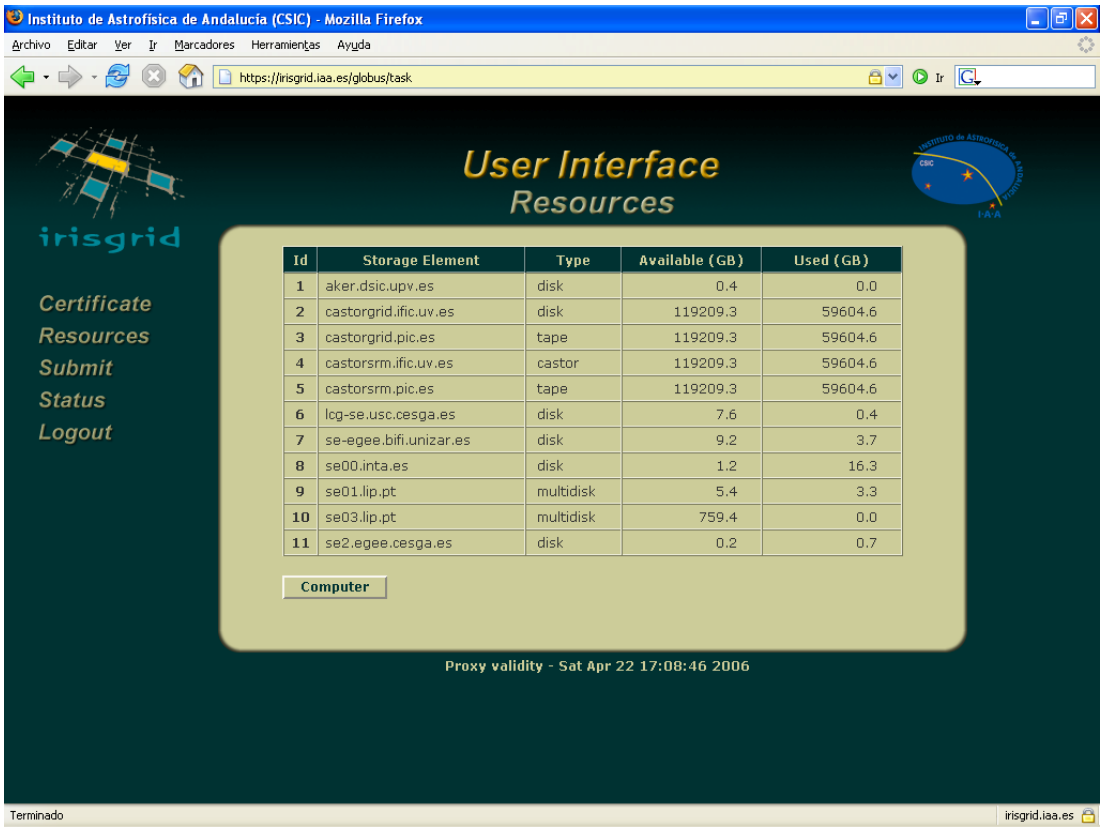


Fig 3.4.3b. *Información de los recursos de almacenamiento disponibles*

3.4.4. Enviar trabajos

La opción ***Submit*** permite al usuario enviar al Grid un trabajo, para su ejecución en un recurso computacional disponible.

Para enviar un trabajo al Grid, el usuario debe:

- Especificar el tipo de trabajo (***JobType***). El tipo de trabajo por defecto es ***Normal***. Si el trabajo está programado en entorno MPI, el usuario debe de especificar ***MPICH*** como tipo de trabajo.
- Adjuntar el fichero ejecutable correspondiente al trabajo (***Executable***). Este fichero puede ser una secuencia de comandos que ejecuten uno o más programas registrados en el CE.
- En caso necesario, especificar los argumentos (***Arguments***) y adjuntar los ficheros de datos (***StdInput***) requeridos por el trabajo.
- Para definir requerimientos específicos, el usuario debe adjuntar el fichero de descripción del trabajo (***JobJDL***). En el Anexo I, se especifican los diferentes atributos que se pueden definir dentro de un fichero de descripción de trabajo (JDL).

Los servicios del *Workload Management System* (WMS) son los responsables de aceptar trabajos y de enviarlos al apropiado CE para su ejecución, dependiendo de los requerimientos del trabajo y de los recursos disponibles. WMS obtiene esta información del *Information Service* (IS) y del *File Catalog* (FC). El Resource Broker (RB) es el nodo donde se ejecutan los servicios WMS.

Al enviar un trabajo al Grid, se requiere una autenticación GSI entre el UI y el RB y entre el RB y el CE. El usuario envía el trabajo desde el UI al nodo RB. En base al fichero de descripción del trabajo (JDL), el fichero ejecutable (***Executable***) y los ficheros de datos (***StdInput***) son transferidos desde el UI al nodo RB. Este conjunto de ficheros se denominan ***Input Sandbox***

El conjunto de ficheros correspondiente al resultado de la ejecución del programa (***StdOutput***) y de registro de errores (***StdError***), se denomina ***Output Sandbox***

El mecanismo Input/Output Sandbox se utiliza sólo cuando se transfieren al Grid ficheros de datos de poco tamaño. Los ficheros de gran tamaño, deben manejarse desde SEs y registrarse en File Catalog, pudiendo ser replicados.

Puede suceder que el certificado proxy de usuario expire antes de la finalización del trabajo. Para que este trabajo pueda continuar y evitar que se produzcan errores en la petición de servicios por acceso no autorizado, el servicio Workload Management Service permite la renovación del certificado proxy si el trabajo lo requiere. El ***proxy server*** (PX) es el componente que permite esta funcionalidad.

Fig 3.4.4. *Enviar un trabajo para su ejecución en el Grid*

3.4.5. Estado de los trabajos

La opción ***Status*** permite al usuario obtener el estado de los trabajos enviados al Grid, en el periodo de tiempo solicitado.

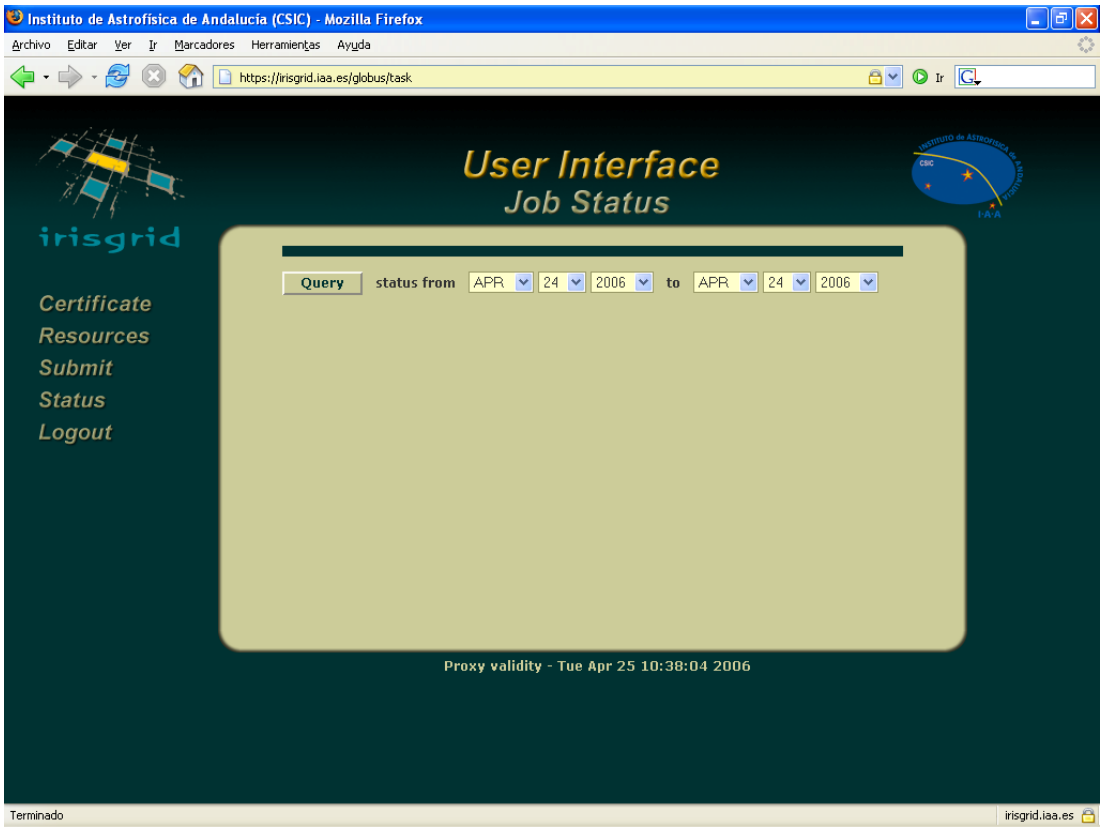


Fig 3.4.5a. *Consultar el estado de los trabajos enviados al Grid*

Una vez realizada la solicitud, se genera una relación de todos los trabajos enviados al Grid (***RegJob***), dentro del periodo de tiempo especificado. Si el periodo de consulta coincide con la fecha actual, el sistema repite la petición cada 30 segundos.

Todas las operaciones se registran en el *Logging and Bookeeping* (LB). Las consultas de estado se realizan desde el UI a la base de datos del LB.

La información proporcionada por la base de datos del LB es la siguiente:

- ***Job***

  Nombre del trabajo. Este nombre lo proporciona el usuario cuando envía el trabajo para su ejecución.

- ***Status***

  Estado del trabajo en el momento de la consulta.

- ***Destination***

  Recurso computacional (CE) que está procesando o ha procesado el trabajo.

- ***Reached on***

  Fecha de registro para el correspondiente estado.

Fig 3.4.5b. *Información del estado de los trabajos enviados al Grid*

Los diferentes estados que puede registrar un trabajo son:

- ***Submitted***
  El usuario envía el trabajo desde el UI al nodo RB. En base al fichero de descripción del trabajo, uno o más ficheros son transferidos desde el UI al nodo RB. Este conjunto de ficheros se denominan ***Input Sandbox***.

- ***Waiting***
  El WMS localiza el mejor CE disponible para la ejecución del trabajo. Para ello, consulta el BDII para obtener el estado de los recursos y en su caso, el File Catalog para localizar los datos requeridos.

- ***Ready***
  El Resource Broker envía el trabajo al CE seleccionado.

- ***Scheduled***
  El CE recibe la petición y envía el trabajo para su ejecución al Local Resource Management System (LRMS).

- ***Running***
  El LRMS encarga la ejecución del trabajo a uno de los nodos WN locales disponibles. Los ficheros del usuario se transfieren del RB al WN donde se va a ejecutar el trabajo.

  El trabajo puede acceder a los ficheros SE de su propia organización, utilizando el protocolo *rfio* o *gsidcap*, o a los ficheros de SEs de otras organizaciones después de copiarlos al sistema de archivos local del WN con las herramientas del servicio *Data Management*. El trabajo puede producir una salida de datos, que puede copiar en un SE y registrarla en el File Catalog para que esté disponible para otros usuarios. Este proceso se realiza utilizando las herramientas del servicio Data Management.

- ***Done***
  Si el trabajo finaliza sin errores, la salida (***Output Sandbox***) es transferida al nodo RB.

- ***Cleared***

  El usuario ha solicitado la información correspondiente a la ejecución del trabajo desde el UI, utilizando WMS.

- ***Aborted***

  Si el CE seleccionado es incapaz de aceptar el trabajo, automáticamente es reenviado a otro CE que cumpla los requerimientos establecidos por el usuario. Después de un número máximo de reenvíos, el trabajo se registra como *Aborted*.

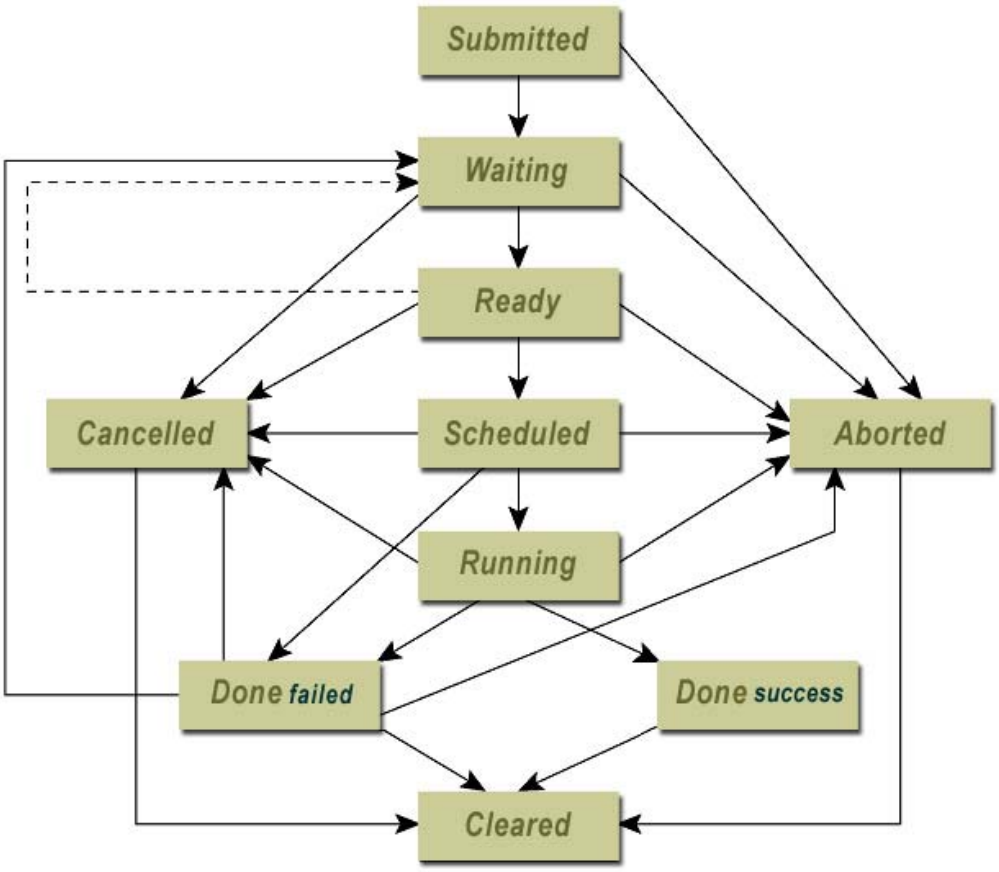


Fig 3.4.5c. *Diagrama de transición de estados*

3.4.6. Información de los trabajos

Dentro de la relación de trabajos obtenida previamente en la consulta de estado, el usuario puede acceder a la información específica de cada trabajo.

Fig 3.4.6a. *Información del trabajo (Retrieve)*

Esta página está dividida en tres secciones:

- ***Job description***

  Información correspondiente a la descripción del trabajo (parámetros definidos por el usuario para la ejecución del trabajo).

- ***Job status***

  Información correspondiente al estado actual del trabajo. Se ofrece al usuario la posibilidad de monitorizar o supervisar dicho estado (***Monitor***).

- ***Job output***

  Si el trabajo ha finalizado correctamente, se ofrece al usuario la posibilidad de solicitar el resultado de la ejecución del trabajo (***Retrieve***).

  Seleccionando esta opción, el UI hace la correspondiente petición al RB y la salida (***Output Sandbox***) es transferida del CE al UI. Si la solicitud se ha realizado correctamente, el trabajo se registra como ***Cleared*** y se ofrece al usuario la posibilidad de descargar los datos obtenidos de la ejecución (***stdout***), y en su caso los errores obtenidos de la ejecución (***stderr***).

  Por último, se ofrece al usuario la posibilidad de regresar a la página correspondiente a la solicitud de consulta de estado (***Back***)

Fig 3.4.6b. *Información del trabajo seleccionado (Output)*

3.4.7. Monitorización de los trabajos

La opción ***Monitor*** permite al usuario monitorizar el proceso de la ejecución de un trabajo y supervisar su estado.

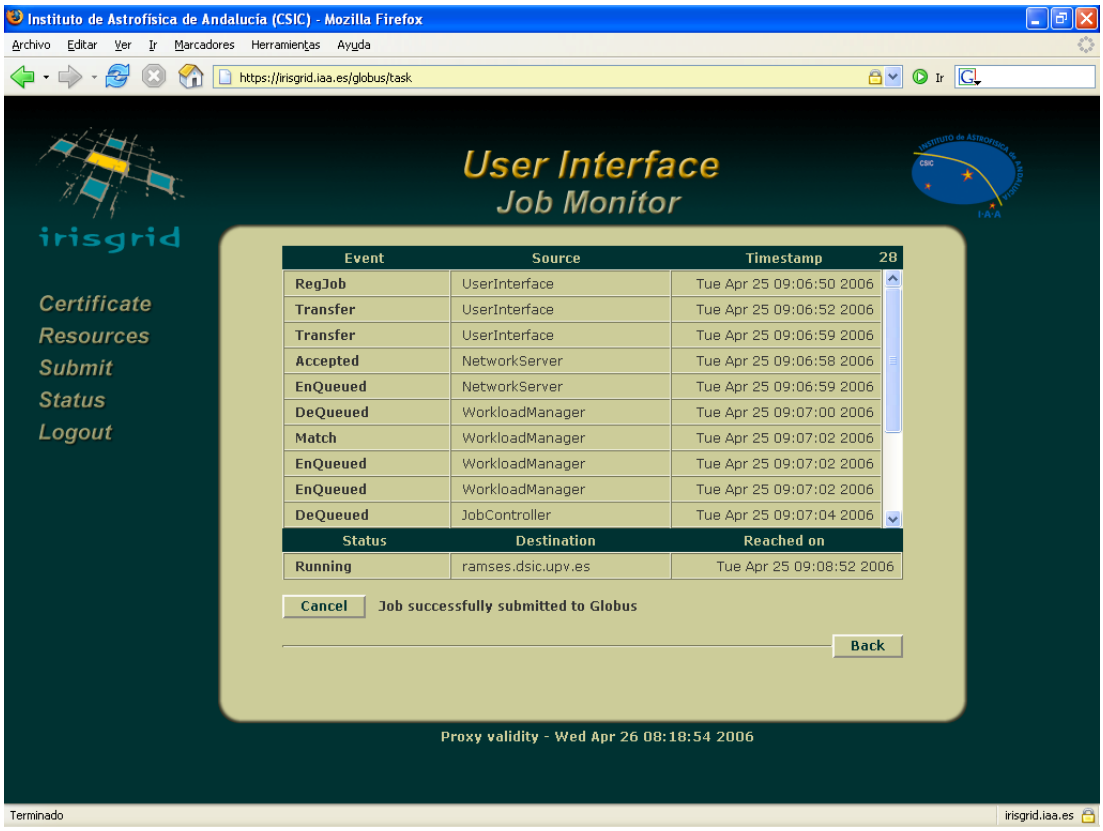


Fig 3.4.7. *Monitorización del trabajo seleccionado*

La información correspondiente a la monitorización de un trabajo, está dividida en dos secciones:

- ***Logging***
  Información correspondiente al proceso de ejecución del trabajo seleccionado

- ***Status***
  Información correspondiente al estado actual del trabajo. Si el trabajo está en proceso de ejecución, el sistema repite la petición cada 30 segundos y se ofrece al usuario la posibilidad de cancelar la ejecución del trabajo (***Cancel***).

Seleccionando esta opción, el UI hace la correspondiente petición al RB. Si se ha podido cancelar el trabajo, éste se registra como ***Cancelled***

También se ofrece al usuario la posibilidad de regresar a la página correspondiente a la solicitud de información del trabajo (***Back***).

3.4.8. Finalizar la sesión

La opción ***Logout*** permite al usuario finalizar la sesión, revocando el certificado proxy creado al inicio de la misma. El UI solicita al usuario la confirmación para finalizar la sesión.

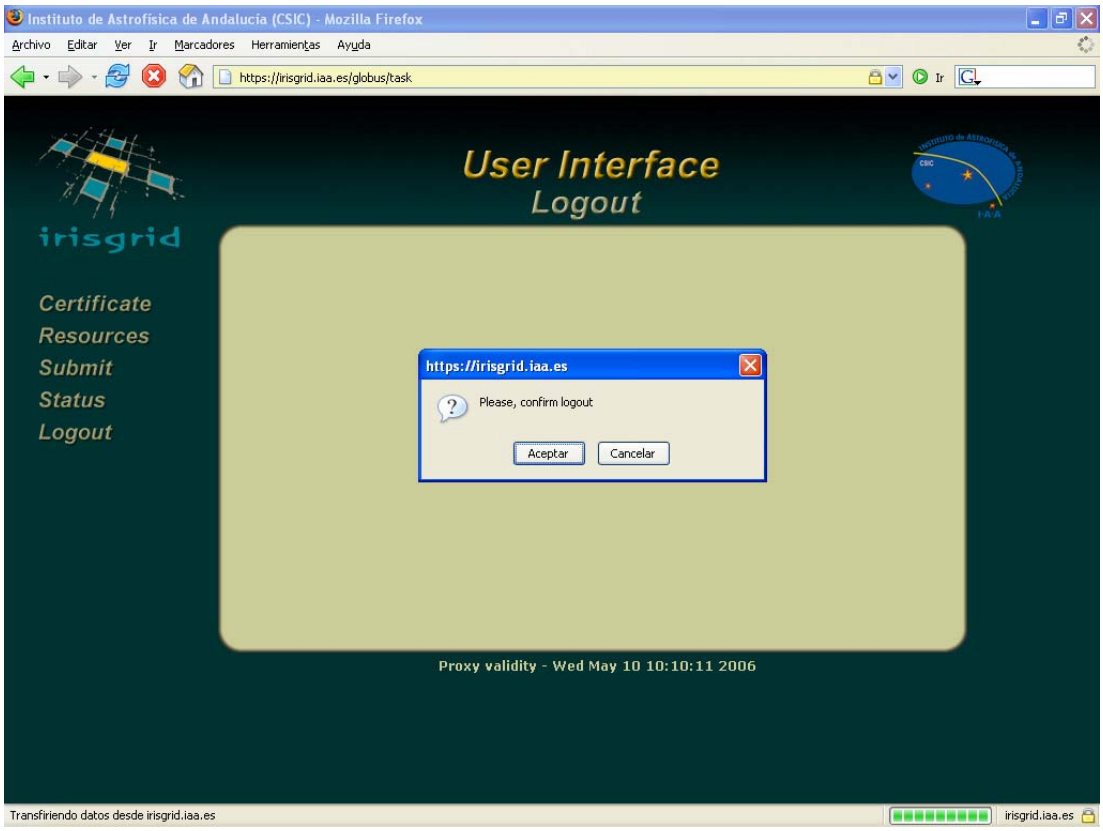


Fig 3.4.8. *Confirmación para finalizar la sesión*

3.5. Integración de servicios Globus

Esta sección corresponde a la tercera y última fase de desarrollo del portal. En ella se analizan los diferentes módulos programados y su correspondencia con los servicios Globus proporcionados por el portal.

La integración de estos servicios se realiza mediante la programación de módulos en lenguaje *Perl* y *JavaScript*.

Los módulos programados están ubicados en tres directorios:

- ***/home/web/globus/required***
  Para acceder a estos módulos, el usuario debe proporcionar sus credenciales a través de un certificado firmado por una CA válida.

  Si el usuario no tiene instalado en su navegador un certificado válido, no podrá tener acceso a este directorio. El control de acceso a los módulos programados, se realiza a través de la comprobación de las variables de entorno y de los datos proporcionados por el usuario.

- ***/home/web/globus***
  Para acceder a estos módulos, no es necesario presentar credenciales de usuario. El control de acceso a los módulos programados, se realiza a través de la comprobación de las variables de entorno y de los datos proporcionados por el usuario.

- ***/home/web/task***
  El acceso a estos módulos está restringido. El usuario no puede hacer una petición dentro de esta área; por tanto, no es necesario realizar un control de acceso a los módulos programados.

A continuación se analizan los diferentes módulos programados, en base al código principal del módulo y de las subrutinas que lo componen.

3.5.1. exec

Todos los comandos Globus se ejecutan a través de este módulo, ubicado en el directorio `/home/web/task`

Es un ***script***, que se ejecuta con el intérprete de comandos ***sh***, en el que se definen las variables de entorno necesarias para la ejecución de los comandos Globus.

El parámetro `$1` corresponde al identificador de usuario (***uid***)

```
# Certificate
export X509_USER_PROXY=/tmp/x509up_$1

# Information Service
export LCG_LOCATION=/opt/lcg
export LCG_GFAL_INFOSYS=lcg2bdii.ific.uv.es:2170

# Workload Service
export EDG_LOCATION=/opt/edg
export EDG_WL_LOCATION=/opt/edg

# Globus toolkit
export GLOBUS_LOCATION=/opt/globus

# Path
export PATH=/bin:/usr/bin:/opt/globus/bin:/opt/globus/sbin:/opt/edg/bin:
/opt/edg/sbin:/opt/lcg/bin:/opt/lcg/sbin
```

El resto de parámetros corresponden a los argumentos definidos para la ejecución del comando Globus

```
# Execute Command
$2 $3 $4 $5 $6 $7 $8 $9
```

En el Anexo II, puede consultarse la referencia de comandos Globus utilizados para la integración de servicios.

3.5.2. shared

En este módulo, ubicado en el directorio `/home/web/task`, se implementan las funciones requeridas por los módulos que procesan las peticiones de servicio, para la generación del contenido de las páginas Web.

La variable `opt` corresponde a la función requerida por el módulo que procesa la petición de servicio:

```
switch ($opt) {
```

- Cabecera de página (***header***)
  `case 'header' { &header }`
- Pié de página (***footer***)
  `case 'footer' { &footer }`
- Título de una sección (***label***)
  `case 'label'  { &label  }`
- Relación de servicios proporcionados por el portal (***info***)
  `case 'info'   { &info   }`
- Regresar a la anterior petición de servicio (***back***)
  `case 'back'   { &back   }`
- Fecha de consulta del estado de los trabajos enviados (***date***)
  `case 'date'   { &date   }`

```
else { exit }}
```

[header]

Esta función es requerida por todos los módulos que procesan las peticiones de servicio.

El código html correspondiente a la cabecera de la página Web, se genera en base al contenido del fichero `/home/web/task/header.html` y a los datos proporcionados por el módulo **access**.

En el fichero `/home/web/task/header.html` se definen:

- Directivas de control, utilizadas por el navegador, para revisar el contenido de las página Web

```
<meta http-equiv="Pragma" content="no-cache">
<meta http-equiv="Expires" content="-1">
```

- La ubicación de la hoja de estilos en cascada *style.css*

```
<link href="/include/style.css" rel="stylesheet" type="text/css">
```

- La ubicación del módulo *scripts.js* con la implementación de funciones JavaScript

```
<script type="text/javascript" src="/include/scripts.js"></script>
```

- El formulario correspondiente al menú de opciones

```
<form name="task" action="/globus/task" enctype="multipart/form-data">
```

A través del menú de opciones, el usuario puede solicitar los siguientes servicios:

- ***Certificate***

  Establecer el periodo de validez del certificado proxy

```
<input name="proxy" type="image">
```

- ***Resources***

  Obtener la relación de todos los recursos computacionales disponibles

```
<input name="element" type="image">
```

- ***Submit***

  Enviar un trabajo al Grid, para su ejecución en un CE disponible

```
<input name="submit" type="image">
```

- ***Status***

  Obtener el estado de los trabajos enviados al Grid, en el periodo de tiempo solicitado

```
<input name="status" type="image">
```

- ***Logout***

  Finalizar la sesión, revocando el certificado proxy creado al inicio de la misma

```
<input name="logout" type="image">
```

El módulo **access** proporciona los datos correspondientes al identificador de usuario (`uid`) y su Organización Virtual (`vo`), requeridos para la ejecución de los comandos Globus. En la cabecera de la página se incluyen estos parámetros, necesarios para identificar la petición de servicio:

- Identificador de usuario

```
<input name="uid" type="hidden" value="$uid">
```

- Organización Virtual

```
<input name="vo" type="hidden" value="$vo">
```

Todos los datos proporcionados por este formulario son procesados por el módulo **task**

[footer]

Esta función es requerida por todos los módulos que procesan las peticiones de servicio. El código html correspondiente al pié de página, se genera en base al contenido del fichero `/home/web/task/footer.html` y a los datos proporcionados por el módulo **access**

En el fichero `/home/web/task/footer.html` se define el mensaje que el usuario puede visualizar durante la petición de servicio. Esta información se visualiza, asignando la clase `pop` a la capa (*layer*) definida con el identificador `info`

La variable `proxy` corresponde a la validez del certificado proxy registrada en el fichero `/tmp/log/uid.px` generado en el módulo **access**

```
# Proxy validity
$proxy = `cat /tmp/log/$uid.px`;
```

Se comprueba la validez del certificado. La información resultante de la comprobación se incorpora en el pié de página

```
$info = $proxy ? "Proxy validity - $proxy" : "Proxy certificate not valid";
```

[info]

Esta función genera el código html correspondiente a la información de los servicios ofrecidos por el Interfaz de Usuario. Esta información se visualiza cuando se realiza el acceso autorizado y cuando el usuario modifica la validez del certificado proxy.

[label]

Esta función genera el código html correspondiente al título de una sección dentro de la página Web.

[back]

Esta función genera el código html que permite al usuario regresar a la anterior petición de servicio. Esta solicitud se realiza a través de la variable `request`

```
<input name="request" type="submit" value="Back">
```

[date]

A través de esta función se genera el código html correspondiente al periodo de consulta del estado de los trabajos enviados al Grid. La fecha de inicio de la consulta (*día, mes, año*) se define a través de las variables `fdd fmm faa` y la fecha de fin de consulta (*día, mes, año*) se define a través de las variables `tdd tmm taa`

```
<select name="fdd" class="select">
<select name="fmm" class="select">
<select name="faa" class="select">
<select name="tdd" class="select">
<select name="tmm" class="select">
<select name="taa" class="select">
```

3.5.3. scripts.js

En este módulo se implementan las funciones ***JavaScript*** requeridas por las páginas Web. Las funciones que se implementa en este módulo son las siguientes:

- Visualizar el título correspondiente a la página (***label***)
- Repetir la solicitud de información (***reload***)
- Identificación de un trabajo (***request***)
- Solicitud de confirmación para finalizar la sesión (***logout***)
- Adjuntar un conjunto de ficheros de datos (***upload***)

[label]

Esta función permite visualizar el título correspondiente a la página Web. El título de la página se define a través de la variable `label`

```
function label(title) {
document.images['label'].src="/include/img/labels/"+title+".gif";
}
```

[reload]

Esta función permite repetir la petición de información solicitada por el sistema. El intervalo de repetición se define a través de la variable `time`

```
function reload() {
  var dx = document.task.time.value;
  if (dx > 0) {
    dx-- ;
    document.task.time.value = dx ;
    setTimeout ("reload()",1000);
  }
  else {
    info.className = "pop";
    document.task.submit();
  }
}
```

[request]

Esta función permite identificar el trabajo seleccionado por el usuario. El identificador correspondiente al trabajo seleccionado se define a través de la variable `jobId`

```
function request(job) {
  document.task.jobId[job].checked = true;
  info.className = "pop";
  document.task.submit();
}
```

[logout]

Esta función solicita al usuario la confirmación para finalizar la sesión

```
function logout() {
  if (confirm ("Please, confirm logout")) { document.task.submit() }
  else { window.location="javascript:history.back()" }
}
```

[upload]

Esta función permite adjuntar un conjunto de ficheros de datos dentro del formulario definido en la página Web.

Para utilizar esta función, es necesario definir la variable `input` y una capa (***layer***) con el identificador `files`

```
<input type="file" id="input">
<div id="files"></div>
```

La variable `max` corresponde al número máximo de ficheros que se pueden adjuntar

```
function upload(max) {
var multi_selector = new MultiSelector( document.getElementById( 'files' ),
max ); multi_selector.addElement( document.getElementById( 'input' ) );
}
```

La función ***MultiSelector*** permite definir, dentro del formulario correspondiente, el conjunto de ficheros de datos que se desean adjuntar

```
function MultiSelector( list_target, max ) {
this.list_target = list_target;
this.max = max;
this.count = 0;
this.id = 0;
```

Cuando se adjunta un fichero de datos, se crea un nuevo elemento dentro del formulario. La variable asignada a este elemento, se especifica con la palabra `data` seguida del valor correspondiente a la variable `id`

```
this.addElement = function( element ) {
  element.name = 'data' + this.id++;
  element.multi_selector = this;

  element.onchange = function() {
    var new_element = document.createElement( 'input' );
    new_element.className = 'hide';
    new_element.type = 'file';
    this.parentNode.insertBefore( new_element, this );
    this.multi_selector.addElement( new_element );
    this.multi_selector.addListRow( this );
    this.style.position = 'absolute';
    this.style.left = '-1000px';
  };
```

Si no se ha superado el valor establecido por la variable `max`, el usuario puede adjuntar un nuevo fichero de datos.

```
  if( this.count++ == this.max ){ element.disabled = true };
  this.current_element = element;
};
```

El nombre del fichero se incorpora a la lista de ficheros de datos que se desean adjuntar.

```
this.addListRow = function( element ) {
  var new_row = document.createElement( 'div' );
  new_row.element = element;

  var new_row_button = document.createElement( 'input' );
  new_row_button.type = 'radio';

  var new_row_data = document.createTextNode( element.value );
```

Al seleccionar un fichero, dentro de la lista de ficheros de datos que se desean adjuntar, se actualiza la lista eliminando el elemento del formulario correspondiente.

```
  new_row_button.onclick= function() {
    this.parentNode.element.parentNode.removeChild(this.parentNode.element);
    this.parentNode.parentNode.removeChild( this.parentNode );
    this.parentNode.element.multi_selector.count--;
    this.parentNode.element.multi_selector.current_element.disabled = false;
    return false;
  };

  this.list_target.appendChild( new_row );
  new_row.appendChild( new_row_button );
  new_row.appendChild( new_row_data );

  };
};
```

3.5.4. account

En este módulo, ubicado en el directorio `/home/web/globus/required`, se implementan las funciones correspondientes a la solicitud de cuenta de usuario.

[check_data]

En esta función se programan las instrucciones necesarias para mantener la integridad de acceso al módulo mediante la comprobación de los datos proporcionados por el usuario.

En primer lugar se obtienen los datos asociados a las variables de entorno SSL

```
@ssl=($ENV{'SSL_CLIENT_VERIFY'},$ENV{'SSL_CLIENT_S_DN'},$ENV{'SSL_CLIENT_V_END'});
```

La variable `SSL_CLIENT_S_DN` corresponde al Subject Distinguished Name (DN) del certificado de usuario X509, y la variable `SSL_CLIENT_V_END` corresponde a la validez del certificado.

Se comprueba que el usuario tiene registrado en su navegador un certificado válido, firmado por la autoridad de certificación pkIRISGridCA. Esta comprobación se realiza a través de la variable de entorno `SSL_CLIENT_VERIFY`. Si el usuario no tiene registrado en su navegador un certificado válido, se le notifica.

```
# Check certificate
if ($ssl[0] ne 'SUCCESS') { exit_code(0) }
```

A continuación se comprueba la petición de acceso al módulo, a través de la variable de entorno `HTTP_REFERER`

```
# Request referer
$url = $ENV{'HTTP_REFERER'};
```

Si el acceso al módulo no se ha realizado desde el propio portal Web, se notifica al usuario que la petición no está autorizada.

```
# Check request referer
if (index($url,'https://irisgrid.iaa.es') != 0) { exit_code(1) }
```

[account_request]

Una vez comprobada la identidad del usuario, se genera la página Web con la definición de todos los parámetros necesarios para procesar las peticiones de solicitud de cuenta de usuario o de renovación de credenciales.

En el formulario `account` se definen todos los parámetros requeridos

```
<form name="account" action="/globus/request" enctype="multipart/form-data">
```

Los datos proporcionados por este formulario son procesados por el módulo **request**

- Certificado de usuario en formato PKCS12

  ```
  <input name="cert" type="file">
  ```

- Contraseña correspondiente a la clave privada del certificado de usuario

  ```
  <input name="pwd" type="password">
  ```

- Dirección de correo electrónico

  ```
  <input name="email" type="text">
  ```

- Subject Distinguished Name (DN) del certificado de usuario X509

  ```
  <input name="subject"  type="hidden" value="$ssl[1]">
  ```

- Validez del certificado de usuario X509

  ```
  <input name="validity" type="hidden" value="$ssl[2]">
  ```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se produce una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

```
0 - Valid certificate required
1 - Request not allowed
```

3.5.5. request

En este módulo, ubicado en el directorio `/home/web/globus`, se implementan las funciones correspondientes al registro de solicitudes de cuenta de usuario, en base a los datos proporcionados por el módulo **account**

[check_data]

En esta función se programan las instrucciones necesarias para mantener la integridad de acceso al sistema mediante la comprobación de los datos proporcionados por el usuario.

En primer lugar se comprueba la petición de acceso al módulo, a través de la variable de entorno `HTTP_REFERER`

```
# Request referer
$url = $ENV{'HTTP_REFERER'};
```

Si el acceso al módulo no se ha realizado desde el propio portal Web, se notifica al usuario que la petición no está autorizada.

```
# Check request referer
if (index($url,'https://irisgrid.iaa.es') != 0) { exit_code(0) }
```

A continuación se comprueban los datos correspondientes al formulario de solicitud. El usuario debe proporcionar todos los datos correspondientes al formulario: *Certificate*, *Passphrase* y *Email*.

```
# Required
$cert  = $cgi->param('cert');
$pwd   = $cgi->param('pwd');
$email = $cgi->param('email');
```

Si el usuario no ha proporcionado el fichero correspondiente al certificado en formato PKCS12 (*certificate*), se le notifica.

```
if (!$cert)  { exit_code(1) }
```

Para poder generar el certificado, el usuario debe proporcionar la contraseña (*passphrase*) correspondiente a la clave privada. Si el usuario no la ha proporcionado, se le notifica.

```
if (!$pwd)    { exit_code(2) }
```

Para poder notificar los datos de la solicitud, es necesario que el usuario especifique su dirección de correo electrónico (*email*). Si el usuario no la ha proporcionado, se le notifica.

```
if (!$email) { exit_code(3) }
```

A continuación, se definen las variables de entorno correspondientes al certificado de usuario, que se utilizarán para registrar la solicitud y notificar al usuario los datos de la solicitud

```
# SSL client
$ssl[1] = $cgi->param('subject');
$ssl[2] = $cgi->param('validity');
```

Se define el directorio correspondiente al registro de certificados

```
$path = "/home/web/certs";
```

Finalmente se comprueba la solicitud. Si la solicitud ya ha sido registrada, se le notifica al usuario.

```
# Check request
if (`grep ':$ssl[1]:' $path/passwd`) { exit_code (4) }
```

[check_cert]

En esta función se genera el certificado de usuario y su correspondiente clave privada, a partir del fichero que el usuario proporciona en el formulario de solicitud.

- Las solicitudes se registran en el archivo `/home/web/certs/rdb`
- Las correspondencias entre el identificador de usuario y su certificado, se registran en el archivo `/home/web/certs/cdb`

Para identificar la solicitud de cuenta de usuario, se genera un número aleatorio (Universally Unique Identifier) con el comando `uuidgen`

```
# Request identifier
chomp ($request = `uuidgen`);
```

Este identificador se utiliza para crear el directorio en el que se almacenarán temporalmente el certificado PKCS12, el certificado de usuario y su correspondiente clave privada.

```
# Request directory
$db = "$path/request/$request";

# Make request directory
`mkdir -m 700 $db`;
```

El fichero proporcionado por el usuario en el formulario de solicitud, se almacena en el directorio anteriormente definido

```
# Upload certificate
open (DB,">$db/usercert.p12");
while (read($cert,$data,2048)) { print DB $data }
close (DB);
```

En primer lugar, se genera el certificado de usuario en formato PEM (`usercert.pem`), utilizando la contraseña (*passphrase*) proporcionada por el usuario. Se ejecuta el comando `openssl` para la generación del certificado. Si el certificado no se ha generado correctamente (verificación MAC), se elimina el directorio que identificaba a la solicitud, y se le notifica al usuario.

```
# Arguments
$args[0] = "pkcs12";
$args[1] = "-clcerts";
$args[2] = "-nokeys";
$args[3] = "-in $db/usercert.p12";
$args[4] = "-out $db/usercert.pem";
$args[5] = "-password pass:$pwd";
$args[6] = "-passin pass:$pwd";
```

```
# User certificate
if (index(`openssl @args 2>&1`,'OK') < 0) { `rm -rf $db`; exit_code(5) }
```

En segundo lugar, se genera la clave privada en formato PEM (`userkey.pem`), utilizando la contraseña (*passphrase*) proporcionada por el usuario. Se ejecuta el comando `openssl` para la generación de la clave privada. Si la clave privada no se ha generado correctamente (verificación MAC), se elimina el directorio que identificaba a la solicitud y se le notifica al usuario.

```
# Arguments
$args[0] = "pkcs12";
$args[1] = "-nocerts";
$args[2] = "-in $db/usercert.p12";
$args[3] = "-out $db/userkey.pem";
$args[4] = "-password pass:$pwd";
$args[5] = "-passin pass:$pwd";
$args[6] = "-passout pass:$pwd";

# User private key
if (index(`openssl @args 2>&1`,'OK') < 0) { `rm -rf $db`; exit_code(5) }
```

Se ejecuta el comando `openssl` para obtener el Subject Distinguished Name (DN) del certificado generado.

```
# Certificate subject
chomp ($subject = `openssl x509 -noout -subject -in $db/usercert.pem`);
substr($subject,0,index($subject,'/'),'');
```

El resultado se comprueba con el Subject Distinguished Name (DN) proporcionado por el módulo **account**. Si el certificado proporcionado por el usuario no es válido, se elimina el directorio que identificaba a la solicitud, y se le notifica al usuario.

```
# Check subject
if ($subject ne $ssl[1]) { `rm -rf $db`; exit_code(6) }
```

Una vez comprobada la validez del certificado, se establecen los permisos de acceso adecuados para mantener la privacidad del certificado:

```
# Certificate access permissions
`chmod 0400 $db/usercert.p12 $db/userkey.pem; chmod 0444 $db/usercert.pem`;
```

Se comprueba si la solicitud corresponde a la renovación del certificado de usuario. En este caso, se actualiza el certificado de usuario y su correspondiente clave privada, y se le notifica al usuario.

```
# Check account
($uid,@data) = split (':',`grep ':$ssl[1]:' $path/cdb`);
if ($uid) { `mv -f $db/* $path/$uid; rm -rf $db`; exit_code (7) }
```

En caso contrario, se registra la solicitud con los datos correspondientes al identificador de solicitud, el Subject Distinguished Name (DN) del certificado de usuario X509, y la dirección de correo electrónico.

```
# Register request
`echo '$request:$ssl[1]:$email' >> $path/rdb`;
```

[send_mail]

Los datos correspondientes a la solicitud, se envían por correo electrónico a la dirección proporcionada por el usuario.

```
open (MAIL,"|/usr/sbin/sendmail -t");
close (MAIL);
```

Para activar la cuenta, el usuario deberá confirmar la petición conectándose a la dirección Web especificada en el mensaje. De esta forma el sistema verifica la validez de la dirección de correo proporcionada por el usuario.

En la dirección Web se proporciona el correspondiente identificador de solicitud
`https://irisgrid.iaa.es/globus/required/register?request=`**`uuid`**

[success]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar una vez registrada la solicitud.

```
Your request has been registered.
The UI manager will contact you to confirm account creation.
Thank you.
```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se produce una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

```
0 - Request not allowed
1 - PKCS12 certificate required
2 - Passphrase required
3 - Email required
4 - Request already registered
5 - Bad passphrase
6 - Valid PKCS12 certificate required
7 - Certificate renewed
```

3.5.6. register

En este módulo, ubicado en el directorio `/home/web/globus/required`, se implementan las funciones correspondientes al registro de cuenta de usuario.

El registro se realiza una vez que el usuario confirma su solicitud, conectándose a la dirección especificada en el mensaje de notificación de solicitud. Es imprescindible que el usuario utilice el navegador donde tiene instalado el certificado proporcionado para la solicitud.

[check_data]

En esta función se programan las instrucciones necesarias para mantener la integridad de acceso al módulo mediante la comprobación de los datos proporcionados por el usuario.

En primer lugar se obtienen los datos asociados a las variables de entorno SSL

```
@ssl=($ENV{'SSL_CLIENT_VERIFY'},$ENV{'SSL_CLIENT_S_DN'},$ENV{'SSL_CLIENT_V_END'});
```

La variable `SSL_CLIENT_S_DN` corresponde al Subject Distinguished Name (DN) del certificado de usuario X509, y la variable `SSL_CLIENT_V_END` corresponde a la validez del certificado.

Se comprueba que el usuario tiene registrado en su navegador un certificado válido, firmado por la autoridad de certificación pkIRISGridCA. Esta comprobación se realiza a través de la variable de entorno `SSL_CLIENT_VERIFY`. Si el usuario no tiene registrado en su navegador un certificado válido, se le notifica.

```
# Check certificate
if ($ssl[0] ne 'SUCCESS') { exit_code(0) }
```

A continuación se comprueba el identificador de la solicitud. Cada petición debe de estar asociada a un identificador de solicitud.

```
# Required
$request = $cgi->param('request');
```

Si no se ha proporcionado el identificador de la solicitud, se notifica al usuario que ha realizado una petición no autorizada.

```
# Check required
if (!$request) { exit_code(1) }
```

Con los datos correspondientes a la solicitud, se consulta la base de datos de solicitudes `/home/web/certs/rdb`

```
# Request data
($request,$subject,$email)= split(':',`grep '$request:$ssl[1]:' $path/rdb`);
```

Si la solicitud no existe o no está pendiente de confirmación, se le notifica al usuario

```
# Check request data
if (!$request) { exit_code (2) }
```

[register_request]

Una vez identificada la solicitud, se procede a la creación de la correspondiente cuenta de usuario. Las correspondencias entre el identificador de usuario y su certificado, se registran en el archivo `/home/web/certs/cdb`

Se consulta la base de datos `/home/web/certs/cdb` para calcular el identificador de usuario (`uid`). Para definir el nombre de usuario se utiliza la letra `u` seguida del identificador calculado.

```
# Next uid
chomp ($data = `tail -1 $path/cdb`);
$uid = ($data ? substr($data,1,4) : 1000);
$uid++;
```

Se registra la correspondencia entre el identificador de usuario y su certificado. Los datos registrados corresponden al nombre de usuario, el Subject Distinguished Name (DN) del certificado de usuario X509, y su dirección de correo electrónico.

```
# Update certificates database
`echo 'u$uid:$subject:$email' >> $path/cdb`;
```

Una vez establecida la correspondencia entre el identificador de usuario y su solicitud, se registran en el directorio de usuario el certificado de usuario (`usercert.pem`) y su correspondiente clave privada (`userkey.pem`)

```
# Register certificate
`mv -f $path/request/$request $path/u$uid`;
```

Para completar el registro, se elimina la solicitud de la base de datos de registro de solicitudes `/home/web/certs/rdb`

```
# Update request database
`grep -v '$request:$subject:$email' $path/rdb > $path/.rdb;
mv -f $path/.rdb $path/rdb`;
```

[send_mail]

Una vez completado el registro de la cuenta de usuario, se notifica por correo electrónico al usuario que dispone de acceso autorizado.

```
open (MAIL,"|/usr/sbin/sendmail -t");
close (MAIL);
```

[success]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar una vez registrada la cuenta de usuario.

```
Your user account has been created.
Now, you have authorized access.
Thank you.
```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

0 - Valid certificate required
1 - Request not allowed
2 - Valid request required

3.5.7. access

En este módulo, ubicado en el directorio `/home/web/globus/required`, se implementan las funciones correspondientes al acceso autorizado.

[check_data]

En esta función se programan las instrucciones necesarias para mantener la integridad de acceso al módulo mediante la comprobación de los datos proporcionados por el usuario.

En primer lugar se obtienen los datos asociados a las variables de entorno SSL

```
@ssl=($ENV{'SSL_CLIENT_VERIFY'},$ENV{'SSL_CLIENT_S_DN'},$ENV{'SSL_CLIENT_V_END'});
```

La variable `SSL_CLIENT_S_DN` corresponde al Subject Distinguished Name (DN) del certificado de usuario X509, y la variable `SSL_CLIENT_V_END` corresponde a la validez del certificado.

Se comprueba que el usuario tiene registrado en su navegador un certificado válido, firmado por la autoridad de certificación pkIRISGridCA. Esta comprobación se realiza a través de la variable de entorno `SSL_CLIENT_VERIFY`. Si el usuario no tiene registrado en su navegador un certificado válido, se le notifica.

```
# Check certificate
if ($ssl[0] ne 'SUCCESS') { exit_code(0) }
```

A continuación se comprueba la petición de acceso al módulo, a través de la variable de entorno `HTTP_REFERER`

```
# Request referer
$url = $ENV{'HTTP_REFERER'};
```

Si el acceso al módulo no se ha realizado desde el propio portal Web, se notifica al usuario que la petición no está autorizada.

```
# Check request referer
if (index($url,'https://irisgrid.iaa.es') != 0) { exit_code(1) }
```

A continuación se comprueban los datos correspondientes al formulario de acceso. El usuario debe proporcionar todos los datos correspondientes al formulario de acceso: *VO, passphrase*

```
# Required
$vo  = $cgi->param('vo');
$pwd = $cgi->param('pwd');
```

Para poder generar el certificado proxy, el usuario debe proporcionar la contraseña (*passphrase*) correspondiente a la clave privada. Si el usuario no la ha proporcionado, se le notifica.

```
if (¡$pwd) { exit_code(2) }
```

Se comprueba, a través de la variable de entorno `SSL_CLIENT_S_DN`, si la cuenta de usuario está registrada en la base de datos `/home/web/certs/cdb`

```
# Account data
($uid,$subject,$email) = split (':',`grep ':$ssl[1]:' $path/cdb`);
```

Si la cuenta de usuario no está registrada, se le notifica al usuario.

```
# Check account
if (!$uid) { exit_code(3) }
```

Antes de iniciar la sesión, se crean los directorios correspondientes al registro de:

- Información proporcionada por los servicios Globus (`/tmp/log`)
- Ficheros de los trabajos proporcionados por el usuario (`/tmp/jobInput`)
- Solicitudes de resultado de la ejecución de los trabajos (`/tmp/jobOutput`)

```
# Required directories
`mkdir –m 700 /tmp/log /tmp/jobInput /tmp/jobOutput`;
```

[proxy_init]

Esta función corresponde a la generación del certificado proxy de usuario. El comando Globus correspondiente a este servicio es `grid-proxy-init`. Para la ejecución de este comando, es necesario definir la variable de entorno

`X509_USER_PROXY` que identifica al certificado proxy de usuario, y el directorio donde está registrado el certificado de usuario y su correspondiente clave privada.

```
# Proxy certificate
export X509_USER_PROXY=/tmp/x509up_uid

# User certificate directory
$db = "/home/web/certs/$uid";
```

La validez por defecto del certificado proxy es de 24 h. Esta validez se puede modificar dentro de la opción ***Certificate***. Una vez establecidos todos los parámetros, se ejecuta el comando `grid-proxy-init`

```
# Arguments
$args[0] = "/opt/globus/bin/grid-proxy-init";
$args[1] = "-pwstdin";
$args[2] = "-valid 24:00";
$args[3] = "-cert $db/usercert.pem";
$args[4] = "-key $db/userkey.pem";
$args[5] = "-certdir /etc/grid-security/certificates";
$args[6] = "-out /tmp/x509up_$uid";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/uid`

```
# Proxy certificate
system ("(echo '$pwd') | $exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `grid-proxy-init`

- `ERROR: Couldn't read user key: Bad passphrase`
- `ERROR: Couldn't find valid credentials to generate a proxy.`
- `Warning: your certificate and proxy will expire Tue Oct 17 18:50:28 2006 which is within the requested lifetime of the proxy`

Se comprueba que el certificado proxy se ha generado correctamente. En caso contrario, se le notifica al usuario.

```
# Check proxy
chomp (($data,$px[0]) = split ('until:',`grep 'until:' /tmp/log/$uid`));
chomp (($data,$px[1]) = split ('expire',`grep 'expire' /tmp/log/$uid`));
```

```
if    ($px[0]) { `echo '$px[0]' > /tmp/log/$uid.px` }
elsif ($px[1]) { `echo '$px[1]' > /tmp/log/$uid.px` }
else { exit_code(4) }
```

El fichero `/tmp/log/`**`uid`**`.px` registra la validez del certificado

```
Your identity: /C=ES/O=DATAGRID-ES/O=IAA/CN=Jose Ruedas
Creating proxy ..................................... Done
Your proxy is valid until: Sat Mar 11 04:22:06 2006
```

[task_info]

Una vez comprobada la validez del certificado proxy, se genera la página Web con la definición de todos los parámetros necesarios para procesar las peticiones de servicio solicitadas por el usuario. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.
  ```
  system ("$exec header services $uid $vo");
  ```
- Se genera el código correspondiente a la información de los servicios ofrecidos por el interfaz de usuario que pueden ser solicitados por el usuario.
  ```
  system ("$exec info");
  ```
- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy
  ```
  system ("$exec footer $uid");
  ```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

```
0 - Valid certificate required
1 - Request not allowed
2 - Passphrase required
3 - User account required
4 - Bad passphrase
```

3.5.8. task

Este módulo, ubicado en el directorio `/home/web/globus`, coordina todas las peticiones de servicio solicitadas por el usuario.

[check_data]

En esta función se programan las instrucciones necesarias para mantener la integridad de acceso al módulo mediante la comprobación de los datos proporcionados por el usuario.

En primer lugar se comprueba la petición de acceso al módulo, a través de la variable de entorno `HTTP_REFERER`

```
# Request referer
$url = $ENV{'HTTP_REFERER'};
```

Si el acceso al módulo no se ha realizado desde el propio portal Web, se notifica al usuario que la petición no está autorizada.

```
# Check request referer
if (index($url,'https://irisgrid.iaa.es') != 0) { exit_code(0) }
```

Cada petición de servicio, está asociada a un identificador de usuario (`uid`) y su correspondiente Organización Virtual (`vo`). Estos datos, requeridos para la ejecución de los comandos Globus, los proporciona el módulo **access**

```
# Required
$uid = $cgi->param('uid');
$vo  = $cgi->param('vo');
```

Los módulos correspondientes a la petición de servicio, se encuentran ubicados en el directorio `/home/web/task`

```
# Task directory
$task = "/home/web/task";
```

Se comprueba si el usuario ha realizado la petición a través del menú de opciones.

```
# Task options
$proxy   = $cgi->param('proxy.x');
$element = $cgi->param('element.x');
$submit  = $cgi->param('submit.x');
$status  = $cgi->param('status.x');
$logout  = $cgi->param('logout.x') ;
```

Si el usuario ha solicitado un servicio a través del menú de opciones, se ejecuta el módulo correspondiente:

- Formulario para cambiar la validez del certificado proxy (***Certificate***)
  ```
  if ($proxy) { system ("$task/proxy Request $uid $vo") }
  ```
- Listado de recursos computacionales disponibles (***Resources***)
  ```
  elsif ($element) { system ("$task/element Computer $uid $vo") }
  ```
- Formulario para enviar un trabajo (***Submit***)
  ```
  elsif ($submit) { system ("$task/submit Request $uid $vo") }
  ```
- Formulario para consultar el estado de los trabajos (***Status***)
  ```
  elsif ($status) { system ("$task/status Request $uid $vo") }
  ```
- Petición para finalizar la sesión (***Logout***)
  ```
  elsif ($logout) { system ("$task/logout Request $uid $vo") }
  ```

Si el usuario no ha realizado la petición de servicio a través del menú de opciones, se comprueba la procedencia de la petición.

```
# Request options
else { &task_request }
```

[task_request]

La variable `request` identifica la procedencia de las peticiones realizadas dentro de cada uno de los servicios proporcionados por el menú de opciones.

```
# Required
$request = $cgi->param('request');
```

El usuario puede solicitar los siguientes servicios:

- Cambio de validez del certificado proxy (***Change***)
- Listado de recursos computacionales disponibles (***Computer***)
- Listado de recursos de almacenamiento disponibles (***Storage***)
- Enviar un trabajo para su ejecución (***Submit***)
- Consultar el estado de los trabajos enviados (***Query***)
- Monitorizar la ejecución de un trabajo (***Monitor***)
- Cancelar la ejecución de un trabajo (***Cancel***)
- Solicitar el resultado de la ejecución de un trabajo (***Retrieve***)
- Descargar los datos obtenidos de la ejecución de un trabajo (***stdout***)
- Descargar los errores obtenidos de la ejecución de un trabajo (***stderr***)
- Regresar a la anterior solicitud de servicio (***Back***)
- Finalizar la sesión (***Logout***)

Se comprueba la variable `request` y se ejecuta la función correspondiente al servicio solicitado por el usuario.

```
switch ($request) {
  case 'Change'   { &job_proxy   }
  case 'Computer' { &job_element }
  case 'Storage'  { &job_element }
  case 'Submit'   { &job_submit  }
  case 'Query'    { &job_status  }
  case 'Monitor'  { &job_monitor }
  case 'Cancel'   { &job_cancel  }
  case 'Retrieve' { &job_info    }
  case 'stdout'   { &job_info    }
  case 'stderr'   { &job_info    }
  case 'Back'     { &job_back    }
  case 'Logout'   { system ("$task/logout Logout $uid $vo") }
```

La petición de servicio solicitada por el usuario, puede corresponder a:

- Consultar el estado de los trabajos enviados durante la sesión (***Query***)
- Monitorizar el estado de un trabajo en proceso de ejecución (***Monitor***)

En este caso, el sistema actualiza de forma automática los datos correspondientes a la solicitud, realizando una petición de servicio cada 30 segundos. Si la petición de servicio ha sido generada por el propio sistema, se comprueba la procedencia de la petición.

```
else { &task_reload }}
```

[task_reload]

La variable `reload` identifica la procedencia de las peticiones realizadas por el propio sistema.

```
# Required
$reload = $cgi->param('reload');
```

El sistema puede generar de forma automática una petición para los siguientes servicios:

- Consultar el estado de los trabajos enviados (***Query***)
- Monitorizar la ejecución de un trabajo (***Monitor***)

```
switch ($reload) {
  case 'Query'   { &job_status  }
  case 'Monitor' { &job_monitor }
  else { exit }}
```

[job_proxy]

Esta función corresponde a la solicitud de cambio de validez del certificado proxy (***Change***). Esta petición se realiza a través de la ejecución del módulo **proxy**. Para poder cambiar la validez del certificado proxy, el usuario debe proporcionar la contraseña (*passphrase*) correspondiente a la clave privada.

```
# Required
$pwd = $cgi->param('pwd');
```

Si el usuario no la ha proporcionado, se le notifica.

```
# Check required
if (!$pwd) { exit_code(1) }
```

La validez del certificado proxy, expresada en horas, se calcula en base a los datos proporcionados por el usuario (*days*, *hours*)

```
# Arguments
$dd = $cgi->param('dd');
$hh = $cgi->param('hh');

# Validity
$hh += $dd*24;
```

Un vez establecidos todos los parámetros, se ejecuta el módulo **proxy**

```
# Proxy init
system ("$task/proxy Change $uid $vo $pwd $hh");
```

[job_element]

Esta función corresponde al listado de los recursos computacionales (***Computer***) y de almacenamiento (***Storage***) disponibles. Esta petición se realiza a través de la ejecución del módulo **element**

```
# Resources info
system ("$task/element $request $uid $vo");
```

[job_submit]

En esta función se procesan todos los datos requeridos para enviar un trabajo para su ejecución (***Submit***). Esta petición se realiza a través de la ejecución del módulo **submit**. El usuario, a partir del formulario de envío de trabajo, puede proporcionar la siguiente información:

- Fichero de descripción del trabajo (*JobJDL*)
  ```
  $job[0] = $cgi->param('jobJDL');
  ```
- Tipo de trabajo (*JobType*)
  ```
  $job[1] = $cgi->param('jobType');
  ```
- Fichero ejecutable correspondiente al trabajo (*Executable*)
  ```
  $job[2] = $cgi->param('jobExecutable');
  ```
- Argumentos requeridos por el trabajo (*Arguments*)
  ```
  $job[3] = $cgi->param('jobArguments');
  ```

- Ficheros de datos requeridos por el trabajo (*StdInput*)

```
@input = $cgi->param('data');
```

Se define el directorio donde se almacenará y procesará toda la información proporcionada por el usuario:

```
# Submit directory
$db = "/tmp/jobInput/$uid";

# Make submit directory
`rm –rf $db; mkdir –m 700 –p $db`;
```

A continuación, se obtiene la información correspondiente al fichero de descripción de trabajo (*JobJDL*)

```
# Upload jobJDL
open (DB,">$db/dconf");
while (read($job[0],$data,2048)) { print DB $data }
close (DB);
```

Se obtiene la información correspondiente al fichero ejecutable (*Executable*)

```
# Upload jobExecutable
open (DB,">$db/dexec");
while (read($job[2],$data,2048)) { print DB $data }
close (DB);
```

Se obtiene la información correspondiente a los ficheros de datos requeridos por el trabajo (*JobInput*)

```
# Upload jobInput
foreach ($cgi->param) {
if ((index($_,'data') eq 0) && ($cgi->param($_)))
{ push(@input,$cgi->param($_)) }}

for ($i=0; $i < scalar(@input); $i++) {
  open (DB,">$db/data$i");
  while (read($input[$i],$data,2048)) { print DB $data }
  close (DB);
}
```

Para poder enviar el trabajo, es necesario que el usuario proporcione la información correspondiente al fichero ejecutable (*Executable*). Si el usuario no ha proporcionado esta información, se le notifica.

```
# Check filesizes
if (-s "$db/dexec") { `mv -f $db/dexec $db/$job[1]` } else { exit_code(2) }
```

Si el usuario no ha proporcionado la información correspondiente al fichero de descripción del trabajo, se definen los atributos necesarios para la ejecución del trabajo

```
if (-s "$db/dconf") {`mv -f $db/dconf $db/$job[0]`} else {&job_description}
```

Un vez establecidos todos los parámetros, se ejecuta el módulo **submit**

```
# Job submit
system ("$task/submit Submit $uid $vo $job[0]");
```

[job_description]

En esta función se definen los atributos requeridos para la ejecución del trabajo. Esta información corresponde al fichero de descripción del trabajo.

- ***JobType***

  Tipo de trabajo. Sólo es necesario definirlo si es un trabajo de tipo MPICH.

  ```
  $info[0] = ($job[1] eq "MPICH") ? "JobType = \"MPICH\";\n" : '';
  ```

- ***Executable***

  Este atributo es obligatorio y corresponde al fichero ejecutable.

  ```
  $info[1] = "Executable = \"$job[2]\";\n";
  ```

- ***Arguments***

  Argumentos requeridos por el fichero ejecutable.

  ```
  $info[2] = $job[3] ? "Arguments = \"$job[3]\";\n" : '';
  ```

- ***StdInput***

  Conjunto de ficheros de datos requeridos por el fichero ejecutable.

  ```
  $info[3] = $input[-1] ? "StdInput = \"$input[-1]\";\n" : '';
  ```

- ***StdOutput***

  Fichero de salida de datos. Valor por defecto `stdout`

  ```
  $info[4] = "StdOutput = \"stdout\";\n";
  ```

- ***StdError***

  Fichero de registro de errores. Valor por defecto `stderr`

  ```
  $info[5] = "StdError = \"stderr\";\n";
  ```

- ***InputSandbox***

  Conjunto de ficheros que se envían para la ejecución del trabajo.

  ```
  $info[6] = "InputSandbox = {\"$job[2]\"$data};\n";
  ```

- ***OutputSandbox***

  Conjunto de ficheros que se obtienen de la ejecución del trabajo.

  ```
  $info[7] = "OutputSandbox = {\"stdout\",\"stderr\"};\n";
  ```

- ***NodeNumber***

  Número de nodos necesarios para la ejecución de un trabajo MPICH

  ```
  $info[8] = ($job[1] eq "MPICH") ? "NodeNumber = 64;\n" : '';
  ```

Una vez definidos todos los parámetros, se registra el fichero de descripción del trabajo correspondiente:

```
# Job description
open (DB,">$db/$job[0]");
print DB @info;
close (DB);
```

En la siguiente sección, se describen todos los atributos que se pueden especificar para la descripción de un trabajo.

[job_status]

Esta función corresponde a la consulta del estado de los trabajos enviados (***Query***), a partir de la selección de un periodo de tiempo. Esta petición se realiza a través de la ejecución del módulo **status**

El periodo de tiempo se define a través de las fechas de inicio y de fin de consulta proporcionadas por el usuario.

```
# From date
$fmm = $cgi->param('fmm');
$fdd = $cgi->param('fdd');
$faa = $cgi->param('faa');

# To date
$tmm = $cgi->param('tmm');
$tdd = $cgi->param('tdd');
$taa = $cgi->param('taa');
```

Una vez obtenida la relación de trabajos enviados, registrados dentro del periodo de consulta, el usuario puede solicitar la información disponible correspondiente al trabajo seleccionado. Esta petición se realiza a través de la ejecución del módulo **info**

Cuando se selecciona un trabajo, se proporciona la información correspondiente al identificador del trabajo (`jobId`) y el periodo de consulta de estado en el que se solicita la información específica del trabajo (`jobSt`)

```
# Job arguments
$jobId = $cgi->param('jobId');

# Status query
$jobSt = "$fmm $fdd $faa $tmm $tdd $taa";
```

Si el usuario ha seleccionado un trabajo, se realiza la correspondiente solicitud de información; en este caso la petición se realiza a través de la ejecución del módulo **info**

```
if ($jobId) { system ("$task/info Request $uid $vo $jobId '$jobSt'") }
```

En caso contrario, el usuario ha solicitado el estado de los trabajos registrados dentro del periodo definido por las fechas de consulta; en este caso la petición se realiza a través de la ejecución del módulo **status**

```
else { system ("$task/status Query $uid $vo $jobSt") }
```

[job_monitor]

Esta función corresponde a la supervisión del estado o a la monitorización de la ejecución del trabajo seleccionado por el usuario (***Monitor***). Esta petición se realiza a través de la ejecución del módulo **monitor**. Para procesar esta petición, es necesario proporcionar la información correspondiente al identificador del trabajo (`jobId`) y al periodo de consulta de estado en el que se solicita la información específica del trabajo (`jobSt`)

```
# Arguments
$jobId = $cgi->param('jobId');
$jobSt = $cgi->param('jobSt');

# Job monitor
system ("$task/monitor $uid $vo $jobId '$jobSt'");
```

[job_cancel]

Durante el proceso de monitorización, el usuario puede solicitar cancelar la ejecución del trabajo seleccionado (***Cancel***). Esta petición se realiza a través de la ejecución del módulo **submit**. Para procesar esta petición, es necesario proporcionar la información correspondiente al identificador del trabajo (`jobId`)

```
# Arguments
$jobId = $cgi->param('jobId');

# Job cancel
system ("$task/submit Cancel $uid $vo $jobId");
```

[job_info]

A través de la selección de un trabajo, el usuario puede:

- Solicitar el resultado de la ejecución (***Retrieve***)
- Descargar los datos obtenidos de la ejecución (***stdout***)
- Descargar los errores obtenidos de la ejecución (***stderr***)

Estas peticiones identificadas por la variable `request`, se realizan a través de la ejecución del módulo **info**. Para procesar esta petición, es necesario proporcionar la

información correspondiente al identificador del trabajo (`jobId`) y al periodo de consulta de estado en el que se solicita la información específica del trabajo (`jobSt`)

```
# Arguments
$jobId = $cgi->param('jobId');
$jobSt = $cgi->param('jobSt');

# Job info
system ("$task/info $request $uid $vo $jobId '$jobSt'");
```

[job_back]

Esta función corresponde a la petición de regresar a la anterior solicitud de servicio (***Back***). Para poder procesar esta petición, se requiere la información correspondiente al identificador del trabajo (`jobId`) y al periodo de consulta de estado en el que se solicita la información específica del trabajo (`jobSt`)

```
# Arguments
$jobId = $cgi->param('jobId');
$jobSt = $cgi->param('jobSt');
```

A través de la variable `reload` se identifica la procedencia de esta petición

```
# Referer
$reload = $cgi->param('reload');
```

Durante el proceso de monitorización de la ejecución de un trabajo, el usuario puede regresar a la solicitud de información del trabajo. En este caso la petición se realiza a través de la ejecución del módulo **info**

```
if ($reload) { system ("$task/info Request $uid $vo $jobId '$jobSt'") }
```

Dentro de la solicitud de información del trabajo, el usuario puede regresar a la solicitud de estado de trabajos en el periodo de tiempo previamente especificado. En este caso la petición se realiza a través de la ejecución del módulo **status**

```
else { system ("$task/status Query $uid $vo $jobSt") }
```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

```
0 - Request not allowed
1 - Passphrase required
2 - Executable required
```

3.5.9. proxy

En este módulo, ubicado en el directorio `/home/web/task`, se procesan las siguientes solicitudes de servicio:

- Cambiar la validez del certificado proxy (***Change***).

Los parámetros para la ejecución de este módulo son los siguientes:

```
($opt,$uid,$vo,$pwd,$hh) = @ARGV;
```

- La variable `opt` corresponde a la petición que se debe procesar.
- Cada petición está asociada a un identificador de usuario (`uid`) y su correspondiente Organización Virtual (`vo`).
- Para poder generar el certificado proxy, el usuario debe proporcionar la contraseña (*passphrase*) correspondiente a la clave privada (`pwd`).
- La validez del certificado proxy, expresada en horas (`hh`), se calcula en base a los datos proporcionados por el usuario (*days*, *hours*)

Se pueden realizar las peticiones: ***Request*** y ***Change***.

```
switch ($opt) {
  case 'Request' { &proxy_request }
  case 'Change'  { &proxy_init    }
  else { exit }}
```

[proxy_request]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando selecciona la opción ***Certificate***. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.

  ```
  system ("$exec header certificate $uid $vo");
  ```

- Se genera el código correspondiente al título de la sección
  `system ("$exec label 'Proxy certificate'");`

- Se definen los parámetros necesarios para poder procesar la petición correspondiente al cambio de validez de certificado (***Change***).

  – Validez del certificado (*days*, *hours*)
  `<select name="dd"><select name="hh">`

  – Contraseña (*passphrase*) correspondiente a la clave privada
  `<input name="pwd" type="password">`

  – Cambiar la validez del certificado proxy (***Change***)
  `<input name="request" type="submit" value="Change">`

- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy
  `system ("$exec footer $uid");`

[proxy_init]

Esta función corresponde a la generación del certificado proxy, con la validez establecida por el usuario. El comando Globus correspondiente a este servicio es `grid-proxy-init`

Para la ejecución de este comando, es necesario definir la variable de entorno `X509_USER_PROXY` que identifica al certificado proxy de usuario, y el directorio donde está registrado el certificado de usuario y su correspondiente clave privada.

```
# Proxy certificate
export X509_USER_PROXY=/tmp/x509up_uid

# User certificate directory
$db = "/home/web/certs/$uid";
```

Una vez establecidos todos los parámetros, se ejecuta el comando `grid-proxy-init`

```
# Arguments
$args[0] = "/opt/globus/bin/grid-proxy-init";
$args[1] = "-pwstdin";
$args[2] = "-valid $hh:00";
$args[3] = "-cert $db/usercert.pem";
$args[4] = "-key $db/userkey.pem";
$args[5] = "-certdir /etc/grid-security/certificates";
$args[6] = "-out /tmp/x509up_$uid";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Proxy certificate
system ("(echo '$pwd') | $exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `grid-proxy-init`

- `ERROR: Couldn't read user key: Bad passphrase`
- `ERROR: Couldn't find valid credentials to generate a proxy.`
- `Warning: your certificate and proxy will expire Tue Oct 17 18:50:28 2006 which is within the requested lifetime of the proxy`

Se comprueba que el certificado proxy se ha generado correctamente. En caso contrario, se le notifica al usuario.

```
# Check proxy
chomp (($data,$px[0]) = split ('until:',`grep 'until:' /tmp/log/$uid`));
chomp (($data,$px[1]) = split ('expire',`grep 'expire' /tmp/log/$uid`));

if    ($px[0]) { `echo '$px[0]' > /tmp/log/$uid.px` }
elsif ($px[1]) { `echo '$px[1]' > /tmp/log/$uid.px` }
else { exit_code(0) }
```

El fichero `/tmp/log/`**`uid`**`.px` registra la nueva validez del certificado

```
Your identity: /C=ES/O=DATAGRID-ES/O=IAA/CN=Jose Ruedas
Creating proxy ..................................... Done
Your proxy is valid until: Sat Mar 11 04:22:06 2006
```

[proxy_init]

Una vez comprobada la validez del certificado proxy, se genera la página Web con la definición de todos los parámetros necesarios para procesar una nueva petición.

Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.

```
system ("$exec header services $uid $vo");
```

- Se genera el código correspondiente a la información de los servicios ofrecidos por el interfaz de usuario

```
system ("$exec info");
```

- Se genera el código correspondiente al pié de página, donde se puede comprobar la nueva validez del certificado proxy

```
system ("$exec footer $uid");
```

- Se notifica al usuario que el certificado proxy ha sido renovado

```
Proxy certificate renewed
```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

```
0 - Bad passphrase
```

3.5.10. element

En este módulo, ubicado en el directorio `/home/web/task`, se procesan las siguientes solicitudes de servicio:

- Listado de recursos computacionales disponibles (***Computer***)
- Listado de recursos de almacenamiento disponibles (***Storage***)

Los parámetros para la ejecución de este módulo son los siguientes:

```
($opt,$uid,$vo) = @ARGV;
```

La variable `opt` corresponde a la petición que se debe procesar; cada petición está asociada a un identificador de usuario (`uid`) y su correspondiente Organización Virtual (`vo`).

Se pueden realizar las peticiones: ***Computer*** y ***Storage***.

```
switch ($opt) {
  case 'Computer' { &computer_request }
  case 'Storage'  { &storage_request }
  else { exit }}
```

[computer_request]

Esta función corresponde a la generación del listado de recursos computacionales disponibles (***Computer***). El comando Globus correspondiente a este servicio es `lcg-info`. Para la ejecución de este comando, es necesario definir la variable de entorno `LCG_GFAL_INFOSYS` que identifica al nodo Database Information Index Server (BDII).

```
export LCG_GFAL_INFOSYS=lcg2bdii.ific.uv.es:2170
```

Se definen los argumentos requeridos para la ejecución del comando `lcg-info`

```
# Arguments
$args[0] = "/opt/lcg/bin/lcg-info";
$args[1] = "--vo $vo";
$args[2] = "--list-ce";
$args[3] = "--quiet";
$args[4] = "--sed";
```

En el Anexo I, se pueden consultar los diferentes atributos que se pueden especificar para los recursos computacionales (CE). Para simplificar la consulta, solamente se especifican los siguientes atributos

```
$args[5] = "--attrs TotalCPUs,FreeCPUs,TotalJobs,RunningJobs";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Computer Elements
system ("$exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `lcg-info`

- `lcg-info: error in contacting BDII lcg2bdii.ific.uv.es:2170`

Si el comando se ha ejecutado correctamente, se muestra al usuario el resultado de la consulta. En caso contrario, se le notifica al usuario.

```
# Check error
if (`grep 'lcg-info: error' /tmp/log/$uid`) { exit_code(0) }
```

[computer_info]

Una vez obtenido el listado de recursos computacionales disponibles, se genera la página Web con la definición de todos los parámetros necesarios para procesar una nueva petición. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.

  ```
  system ("$exec header resources $uid $vo");
  ```

- Se genera el código correspondiente a la información de los recursos computacionales disponibles. Esta información se visualiza a través de una capa (*layer*) definida en la hoja de estilos `style.css` por la clase `scroll`

  ```
  <div class="scroll">
  ```

- Se ofrece al usuario la posibilidad de solicitar el servicio correspondiente al listado de recursos de almacenamiento disponibles (***Storage***)

```
<input name="request" type="submit" value="Storage">
```

- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy

```
system ("$exec footer $uid");
```

[storage_request]

Esta función corresponde a la generación del listado de recursos de almacenamiento disponibles (***Storage***). El comando Globus correspondiente a este servicio es `lcg-info`. Para la ejecución de este comando, es necesario definir la variable de entorno `LCG_GFAL_INFOSYS` que identifica al nodo Database Information Index Server (BDII).

```
export LCG_GFAL_INFOSYS=lcg2bdii.ific.uv.es:2170
```

Se definen los argumentos requeridos para la ejecución del comando `lcg-info`

```
# Arguments
$args[0] = "--vo $vo";
$args[1] = "--list-se";
$args[2] = "--quiet";
$args[3] = "--sed";
```

En el Anexo I, se pueden consultar los diferentes atributos que se pueden especificar para los recursos de almacenamiento (SE). Para simplificar la consulta, solamente se especifican los siguientes atributos

```
$args[4] = "--attrs SEArch,AvailableSpace,UsedSpace";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Storage Elements
system ("$exec $uid /opt/lcg/bin/lcg-info @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `lcg-info`

- `lcg-info: error in contacting BDII lcg2bdii.ific.uv.es:2170`

Si el comando se ha ejecutado correctamente, se muestra al usuario el resultado de la consulta. En caso contrario, se le notifica al usuario.

```
# Check error
if (`grep 'lcg-info:' /tmp/log/$uid`) { exit_code(0) }
```

[storage_info]

Una vez obtenido el listado de recursos de almacenamiento disponibles, se genera la página Web con la definición de todos los parámetros necesarios para procesar una nueva petición. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.
  ```
  system ("$exec header resources $uid $vo");
  ```
- Se genera el código correspondiente a la información de los recursos de almacenamiento disponibles. Esta información se visualiza a través de una capa (*layer*) definida en la hoja de estilos `style.css` por la clase `scroll`
  ```
  <div class="scroll">
  ```
- Para ofrecer al usuario la posibilidad de solicitar el servicio correspondiente al listado de recursos computacionales disponibles (***Computer***), se define la variable `request`
  ```
  <input name="request" type="submit" value="Computer">
  ```
- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy
  ```
  system ("$exec footer $uid");
  ```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

```
0 - Unable to contact Information Index Server
```

3.5.11. submit

En este módulo, ubicado en el directorio `/home/web/task`, se procesan las siguientes solicitudes de servicio:

- Enviar un trabajo para su ejecución (***Submit***)
- Cancelar la ejecución de un trabajo (***Cancel***)

Los parámetros para la ejecución de este módulo son los siguientes:

```
($opt,$uid,$vo,$job) = @ARGV;
```

- La variable `opt` corresponde a la petición que se debe procesar; cada petición está asociada a un identificador de usuario (`uid`) y su correspondiente Organización Virtual (`vo`).

- Cuando se solicita enviar un trabajo para su ejecución, la variable `job` corresponde al fichero de descripción de trabajo; cuando se solicita cancelar la ejecución de un trabajo, la variable `job` corresponde al identificador del trabajo.

Se pueden realizar las peticiones: ***Request, Submit*** y ***Cancel***.

```
switch ($opt) {
  case 'Request' { &submit_request }
  case 'Submit'  { &job_submit }
  case 'Cancel'  { &job_cancel }
  else { exit }}
```

[submit_request]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando selecciona la opción ***Submit***. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.

  ```
  system ("$exec header submit $uid $vo");
  ```

- Se genera el código correspondiente al título de la sección

```
system ("$exec label 'Job description'");
```

- Se definen los parámetros necesarios para poder procesar la petición de enviar un trabajo para su ejecución (***Submit***)

  – Tipo de trabajo (*JobType*)

```
<select name="jobType">
```

  – Fichero ejecutable correspondiente al trabajo (*Executable*)

```
<input name="jobExecutable" type="file">
```

  – Argumentos (*Arguments*) requeridos por el trabajo

```
<input name="jobArguments" type="text">
```

  – Ficheros de datos (*StdInput*) requeridos por el trabajo

```
<input id="input" type="file">
<div id="files"></div>
<body onLoad="upload(4)">
```

  – Fichero de descripción del trabajo (*JobJDL*)

```
<input name="jobJDL" type="file">
```

  – Enviar el trabajo para su ejecución (*Submit*)

```
<input name="request" type="submit" value="Submit">
```

- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy

```
system ("$exec footer $uid");
```

[job_submit]

Esta función corresponde a la solicitud de ejecución de un trabajo. La información correspondiente al trabajo, previamente procesada por el módulo **task**, está registrada en el directorio de usuario `/tmp/jobInput/`**`uid`**

En este caso, la variable `job` corresponde al fichero de descripción de trabajo.

Antes de enviar el trabajo para su ejecución, se consulta la disponibilidad de recursos computacionales. Esta consulta se realiza ejecutando el comando Globus `edg-job-list-match`

```
# Arguments
$args[0] = "/opt/edg/bin/edg-job-list-match";
$args[1] = "--vo $vo";
$args[2] = "$job";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Check resources
system ("cd $path; $exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `edg-job-list-match`

- `Error: UI_PROXY_NOT_FOUND`
  `Proxy certificate not found.`
- `Error: UI_PROXY_DURATION`
  `Proxy certificate will expire within less then 00:20 hours.`
- `Error: API_NATIVE_ERROR`
  `Unable to connect to remote`
- `Error: UI_NO_NS_CONTACT`
  `Unable to contact any Network Server`
- `Error: UI_JDL_ERROR`
  `Invalid attributes definition`
- `Error: UI_NO_VOMS`
  `Unable to determine a valid user's VO`
- `edg-job-list-match failure`
  `Not found Computing Element for job requirements`

Se comprueba que el comando se ha ejecutado correctamente. Si se ha producido alguna excepción que impide que el trabajo pueda ser ejecutado, se le notifica al usuario. Una vez comprobada la disponibilidad de recursos computacionales, se ejecuta el comando Globus `edg-job-submit`

```
# Arguments
$args[0] = "/opt/edg/bin/edg-job-submit";
$args[1] = "--vo $vo";
$args[2] = "$job";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Job submit
system ("cd $path; $exec $uid @args &> /tmp/log/$uid");
```

La información obtenida corresponde al identificador asignado al trabajo (`jobId`).

```
# jobId
chomp(($data,$jobId) = split (':9000/',`grep 'https' /tmp/log/$uid`));
```

A través de este identificador, se crea el directorio asociado al trabajo (dentro del área de usuario). En este directorio se registra el fichero correspondiente a la descripción del trabajo.

```
# Job directory
$db = "/home/web/jobs/$uid/$jobId";

# Job register
`mkdir -m 700 -p $db; mv -f $path/$job $db/submit.jdl`;
```

Finalmente se notifica al usuario que el trabajo ha sido enviado satisfactoriamente.

```
Job successfully submitted
```

Una vez realizada la notificación, el usuario podrá visualizar de nuevo la página Web correspondiente a la opción de enviar un trabajo para su ejecución (***Submit***)

[job_cancel]

Esta función corresponde a la cancelación de la ejecución de un trabajo (***Cancel***). El comando Globus correspondiente a este servicio es `edg-job-cancel`. Se definen los argumentos requeridos para la ejecución del comando; en este caso, la variable `job` corresponde al identificador del trabajo.

```
# Arguments
$args[0] = "/opt/edg/bin/edg-job-cancel";
$args[1] = "--noint";
$args[2] = "--logfile /tmp/log/$uid";
$args[3] = "https://lcg2rb.ific.uv.es:9000/$job";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Job cancel
system ("$exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `edg-job-cancel`

- `Error: UI_PROXY_NOT_FOUND`
  `Proxy certificate not found.`
- `Error: UI_PROXY_DURATION`
  `Proxy certificate will expire within less then 00:20 hours.`
- `Error: API_NATIVE_ERROR`
  `Network is unreachable`
- `Error: UI_CANCEL_FAIL`
  `Unable to cancel any job`
- `Error: NS_JOB_CANCEL_NOT_ALLOWED`
  `Not allowed to cancel the job`

Se comprueba que el comando se ha ejecutado correctamente. Si se ha producido alguna excepción que impide que el trabajo pueda ser cancelado, se le notifica al usuario. En caso contrario, se notifica al usuario que la petición de cancelación ha sido enviada al RB; una vez realizada la notificación, el usuario podrá visualizar de nuevo la página Web correspondiente a la monitorización de la ejecución del trabajo (***Monitor***)

`Cancel request sent to Resource Broker`

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

0 - `Proxy certificate not valid`
1 - `Unable to contact Resource Broker`
2 - `Invalid Attributes`
3 - `Not found Computing Element for job requirements`
4 - `Job successfully submitted`
5 - `Not allowed to cancel the job`
6 - `Cancel request sent to Resource Broker`

3.5.12. status

En este módulo, ubicado en el directorio `/home/web/task`, se procesan las siguientes solicitudes de servicio:

- Consultar el estado de los trabajos enviados (***Query***)

Los parámetros para la ejecución de este módulo son los siguientes:

```
($opt,$uid,$vo,$fmm,$fdd,$faa,$tmm,$tdd,$taa) = @ARGV;
```

- La variable opt corresponde a la petición que se debe procesar; cada petición está asociada a un identificador de usuario (`uid`) y su correspondiente Organización Virtual (`vo`).
- La consulta está definida por el mes, día y año de inicio de consulta (`fmm`, `fdd`, `faa`) y por el mes, día y año de fin de consulta (`tmm`, `tdd`, `taa`).

Se pueden realizar las peticiones: ***Request*** y ***Query***.

```
switch ($opt) {
  case 'Request' { &status_request }
  case 'Query'   { &status_query }
  else { exit }}
```

[status_request]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando selecciona la opción ***Status***. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.

  ```
  system ("$exec header status $uid $vo");
  ```

- Se definen los parámetros necesarios para poder procesar la petición correspondiente a la consulta de estado de los trabajos (***Query***)

– Fecha de inicio de consulta. El valor por defecto corresponde a la fecha actual.

```
system ("$exec date fmm fdd faa $fmm $fdd $faa");
```

– Fecha de fin de consulta. El valor por defecto corresponde a la fecha actual.

```
system ("$exec date tmm tdd taa $tmm $tdd $taa");
```

– Consultar el estado de los trabajos enviados (***Query***)

```
<input name="request" type="hidden" value="Query">
<input type="submit" value="Query">
```

- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy

```
system ("$exec footer $uid");
```

[status_query]

Esta función corresponde a la consulta del estado de los trabajos enviados (***Query***), en el periodo establecido por el usuario. El comando Globus correspondiente a este servicio es `edg-job-status`

En primer lugar, se comprueba la validez de las fechas proporcionadas por el usuario.

```
local (@date) = (localtime(time))[0,1,2,3,4,5,6];
local ($aa,$mm,$dd) = ($date[5]+1900,$date[4]+1,$date[3]);

# From date
$from = sprintf ("%02d:%02d:00:00:%04d",$fmm,$fdd,$faa);

# To date
$to = sprintf ("%02d:%02d:23:59:%04d",$tmm,$tdd,$taa);

# Check date
$date[0] = sprintf ("%04d%02d%02d",$aa,$mm,$dd);
$date[1] = sprintf ("%04d%02d%02d",$faa,$fmm,$fdd);
$date[2] = sprintf ("%04d%02d%02d",$taa,$tmm,$tdd);
```

Se establece el periodo de consulta (`query`)

```
$query = ($date[2] ge $date[0]) ? "--from $from" : "--from $from --to $to";
```

Si el periodo de consulta coincide con la fecha actual, el sistema repite la petición cada 30 segundos (`check`)

```
$check = ($date[1] eq $date[0]) ? 1 : 0;
```

Una vez establecido el periodo de consulta, se definen los argumentos requeridos para la ejecución del comando `edg-job-status`

```
# Arguments
$args[0] = "/opt/edg/bin/edg-job-status";
$args[1] = "--vo $vo";
$args[2] = "--all";
$args[3] = "$query";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Query status
system ("$exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `edg-job-status`

- `Error: UI_PROXY_NOT_FOUND`
  `Proxy certificate not found.`
- `Error: UI_PROXY_DURATION`
  `Proxy certificate will expire within less then 00:20 hours.`
- `Error: API_NATIVE_ERROR`
  `Unable to retrieve status query information`
  `Network is unreachable`
- `Error: LB_API_OPEN_ALL`
  `Unable to open connection with any LB supplied`
- `Error: UI_NO_JOB_FOUND`
  `Unable to find any job matching the query`
- `Error: UI_ARG_OUT_OF_LIMIT`
  `Invalid query arguments`

Si el comando se ha ejecutado correctamente, se muestra al usuario el resultado de la consulta. Si se ha producido alguna excepción que impide que se pueda realizar la consulta, se le notifica al usuario.

[status_info]

Una vez obtenido el listado de trabajos registrados dentro del periodo de consulta, se genera la página Web con la definición de todos los parámetros necesarios para procesar una nueva petición. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.

```
system ("$exec header status $uid $vo");
```

- Se genera el código correspondiente a la información del estado de los trabajos. Esta información se visualiza a través de una capa (*layer*) definida en la hoja de estilos `style.css` por la clase `scroll`

```
<div class="scroll">
```

- Para ofrecer al usuario la posibilidad de solicitar la información específica de un trabajo, se define la variable correspondiente al identificador del trabajo `jobId`

```
<input name="jobId" type="radio" value="$job">
```

- Se definen los parámetros necesarios para poder procesar una nueva petición de consulta de estado de los trabajos (***Query***). El valor inicialmente establecido para el periodo de consulta, corresponde al periodo definido en la anterior consulta.

  – Fecha de inicio de consulta.

```
system ("$exec date fmm fdd faa $fmm $fdd $faa");
```

  – Fecha límite de consulta.

```
system ("$exec date tmm tdd taa $tmm $tdd $taa");
```

  – Consultar el estado de los trabajos enviados (***Query***)

```
<input name="request" type="hidden" value="Query">
<input type="submit" value="Query">
```

- Si el periodo de consulta coincide con la fecha actual, se definen los parámetros necesarios para generar de forma automática una petición cada 30 segundos. Esta petición se realiza a través de la variable `reload`

```
<input name="time" type="text" value="30" class="clock" readonly>
<input name="reload" type="hidden" value="Query">
<body onLoad="reload()">
```

- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy

```
system ("$exec footer $uid");
```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

```
0 - Proxy certificate not valid
1 - Unable to contact Logging & Bookeeping Server
2 - Unable to find any job matching the query
3 - Invalid query date
```

3.5.13. info

En este módulo, ubicado en el directorio `/home/web/task`, se procesan las siguientes solicitudes de servicio:

- Solicitar el resultado de la ejecución del trabajo (***Retrieve***)
- Descargar los datos obtenidos de la ejecución del trabajo (***stdout***)
- Descargar los errores obtenidos de la ejecución del trabajo (***stderr***)
- Regresar a la anterior solicitud de servicio (***Back***)

Los parámetros para la ejecución de este módulo son los siguientes:

```
($opt,$uid,$vo,$job,$stat) = @ARGV;
```

- La variable `opt` corresponde a la petición que se debe procesar; cada petición está asociada a un identificador de usuario (`uid`) y su correspondiente Organización Virtual (`vo`)

- La variable `job` corresponde al identificador del trabajo seleccionado por el usuario; la variable `stat` corresponde al periodo de consulta de estado en el que se solicita la información específica del trabajo.

Se pueden las peticiones: ***Request***, ***Retrieve***, ***stdout***, ***stderr*** y ***Back***.

```
switch ($opt) {
  case 'Request'  { &job_request }
  case 'Retrieve' { &job_output  }
  case 'stdout'   { &job_stdout  }
  case 'stderr'   { &job_stderr  }
  case 'Back'     { &job_request }
  else { exit }}
```

[job_request]

Esta función verifica el estado del trabajo seleccionado. El comando Globus correspondiente a este servicio es `edg-job-status`.

Se definen los argumentos requeridos para la ejecución del comando:

```
# Arguments
$args[0] = "/opt/edg/bin/edg-job-status";
$args[1] = "https://lcg2rb.ific.uv.es:9000/$job";
```

La variable `job` corresponde al identificador del trabajo. El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Status info
system ("$exec $uid @args &> /tmp/log/$uid");

# Status database
local (@info) = ('Current Status:','Status Reason:');

chomp(($data,$ds[0]) = split ($info[0],`grep '$info[0]' /tmp/log/$uid`));
chomp(($data,$ds[1]) = split ($info[1],`grep '$info[1]' /tmp/log/$uid`));
```

Se analiza el registro de errores correspondiente al comando `edg-job-status`

- `Error: UI_PROXY_NOT_FOUND`
  `Proxy certificate not found.`
- `Error: UI_PROXY_DURATION`
  `Proxy certificate will expire within less then 00:20 hours.`
- `Error: API_NATIVE_ERROR`
  `Unable to retrieve status query information`
  `Network is unreachable`
- `Error: LB_API_OPEN_ALL`
  `Unable to open connection with any LB supplied`

Si el comando se ha ejecutado correctamente, se comprueba el estado del trabajo. Si el trabajo ha finalizado correctamente (***Success***), se ofrece al usuario la posibilidad de solicitar el resultado de la ejecución del trabajo (***Retrieve***).

```
# Retrieve
$check = (index($ds[0],'Success') > 0) ? 1 : 0;
```

Si se ha producido alguna excepción que impide que se pueda realizar la consulta, se le notifica al usuario. En caso contrario, se muestra al usuario toda la información disponible del trabajo.

[job_info]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando selecciona un trabajo. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.

```
system ("$exec header info $uid $vo");
```

- Se definen los parámetros correspondientes al identificador del trabajo (`jobId`) y al periodo de consulta de estado (`stat`) en el que se solicita la información específica del trabajo.

```
<input name="jobId" type="hidden" value="$job">
<input name="jobSt" type="hidden" value="$stat">
```

- Se genera el código correspondiente a la sección ***Job description***

```
system ("$exec label 'Job description'");
```

  La información proporcionada corresponde a la descripción del trabajo, registrada en el fichero `submit.jdl`

```
open (DB,"$path/submit.jdl");
while (chomp($data=<DB>)) { print "<li>$data</li>" };
close (DB);
```

- Se genera el código correspondiente a la sección ***Job status***

```
system ("$exec label 'Job status'");
```

  La información proporcionada corresponde al estado actual del trabajo. Se ofrece al usuario la posibilidad de monitorizar o supervisar dicho estado (***Monitor***).

```
<input name="request" type="submit" value="Monitor">
```

- Se genera el código correspondiente a la sección ***Job output***

```
system ("$exec label 'Job output'");
```

– Si el trabajo ha finalizado correctamente, se ofrece al usuario la posibilidad de solicitar el resultado de la ejecución del trabajo (***Retrieve***)

```
<input name="request" type="submit" value="Retrieve">
```

– Si la solicitud de resultado se ha realizado correctamente, se ofrece al usuario la posibilidad de descargar los datos obtenidos de la ejecución (***stdout***), y en su caso los errores obtenidos de la ejecución (***stderr***)

```
<input name="request" type="submit" value="stdout">
<input name="request" type="submit" value="stderr">
```

- Por último, se ofrece al usuario la posibilidad de regresar a la anterior solicitud de consulta de estado (***Back***)

```
<input name="request" type="submit" value="Back">;
```

- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy

```
system ("$exec footer $uid");
```

[job_output]

Esta función corresponde a la solicitud del resultado de la ejecución de un trabajo (***Retrieve***). El comando correspondiente a este servicio es `edg-job-get-output`

Se define el directorio temporal donde se registran todas las solicitudes de resultado, y el directorio asociado al trabajo (dentro del área de usuario)

```
# Output directory
$path = "/tmp/jobOutput";

# Job directory
$db = "/home/web/jobs/$uid/$job";
```

Se ejecuta el comando `edg-job-get-output` para solicitar el resultado de la ejecución del trabajo. La solicitud se realiza a través del identificador de trabajo (`job`)

```
# Arguments
$args[0] = "/opt/edg/bin/edg-job-get-output";
$args[1] = "https://lcg2rb.ific.uv.es:9000/$job";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Retrieve output
system ("$exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `edg-job-get-output`

- `Error: UI_PROXY_NOT_FOUND`
  `Proxy certificate not found.`
- `Error: UI_PROXY_DURATION`
  `Proxy certificate will expire within less then 00:20 hours.`
- `Error: API_NATIVE_ERROR`
  `Network is unreachable`
- `Error: LB_API_OPEN_ALL`
  `Unable to open connection with any LB supplied`

Si la solicitud de resultado se ha realizado correctamente, se registran los ficheros obtenidos en el directorio asociado al trabajo (dentro del área de usuario).

```
# Register output files
`mv -f $path/jobOutput_$job/* $db`; `rm -rf $path/jobOutput_$job`;
```

Si se ha producido alguna excepción que impide que se pueda solicitar el resultado de la ejecución del trabajo, se le notifica al usuario. En caso contrario, se vuelve a mostrar al usuario toda la información disponible del trabajo.

```
# Check success
if (`grep 'JOB GET OUTPUT OUTCOME' /tmp/log/$uid`) { &job_request }
else { exit_code(2) }
```

[job_stdout]

Esta función permite al usuario descargar los datos obtenidos de la ejecución de un trabajo (***stdout***)

```
# Job directory
$path = "/home/web/jobs/$uid/$job";

print "Content-Type: application/force-download\n";
print "Content-Disposition: attachment; filename=stdout\n\n";
```

```
# StdOutput
system ("cat $path/stdout");
```

[job_stderr]

Esta función permite al usuario descargar los errores obtenidos de la ejecución de un trabajo (***stderr***)

```
# Job directory
$path = "/home/web/jobs/$uid/$job";

print "Content-Type: application/force-download\n";
print "Content-Disposition: attachment; filename=stderr\n\n";

# StdError
system ("cat $path/stderr");
```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

0 - Proxy certificate not valid
1 - Unable to contact Logging & Bookeeping Server
2 - Unable to retrieve job output

3.5.14. monitor

En este módulo, ubicado en el directorio `/home/web/task`, se procesan las siguientes solicitudes de servicio:

- Monitorizar la ejecución de un trabajo (***Monitor***)
- Regresar a la anterior solicitud de servicio (***Back***)

Los parámetros para la ejecución de este módulo son los siguientes:

```
($uid,$vo,$job,$stat) = @ARGV;
```

- Cada petición está asociada a un identificador de usuario (`uid`) y su correspondiente Organización Virtual (`vo`)
- La variable `job` corresponde al identificador del trabajo; la variable `stat` corresponde al periodo de consulta de estado en el que se solicita la información específica del trabajo.

[job_monitor]

Esta función permite monitorizar el proceso de ejecución de un trabajo, o supervisar el estado de un trabajo una vez finalizado el proceso de ejecución. En esta función, se obtiene la información correspondiente al registro de eventos producidos durante la ejecución del trabajo y al estado del trabajo.

En primer lugar, se obtiene la información correspondiente al registro de eventos producidos durante la ejecución del trabajo. El comando Globus correspondiente a este servicio es `edg-job-get-logging-info`. Se ejecuta este comando para obtener el registro de eventos. La variable `job` corresponde al identificador del trabajo.

```
# Arguments
$args[0] = "/opt/edg/bin/edg-job-get-logging-info";
$args[1] = "https://lcg2rb.ific.uv.es:9000/$job";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Logging info
system ("$exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores relativo al comando `edg-job-get-logging-info`

- `Error: UI_PROXY_NOT_FOUND`
  `Proxy certificate not found.`
- `Error: UI_PROXY_DURATION`
  `Proxy certificate will expire within less then 00:20 hours.`
- `Error: API_NATIVE_ERROR`
  `Network is unreachable`
- `Error: LB_API_OPEN_ALL`
  `Unable to open connection with any LB supplied`

Si se ha producido alguna excepción que impide que se pueda obtener el registro de eventos producidos durante la ejecución del trabajo, se le notifica al usuario. En caso contrario, se procesa la información obtenida.

En segundo lugar, se verifica el estado del trabajo. El comando Globus correspondiente a este servicio es `edg-job-status`. Se definen los argumentos requeridos para la ejecución del comando. La variable `job` corresponde al identificador del trabajo.

```
# Arguments
$args[0] = "/opt/edg/bin/edg-job-status";
$args[1] = "https://lcg2rb.ific.uv.es:9000/$job";
```

El resultado de la ejecución se almacena en el fichero `/tmp/log/`**`uid`**

```
# Status info
system ("$exec $uid @args &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `edg-job-status`

- `Error: UI_PROXY_NOT_FOUND`
  `Proxy certificate not found.`
- `Error: UI_PROXY_DURATION`
  `Proxy certificate will expire within less then 00:20 hours.`
- `Error: API_NATIVE_ERROR`
  `Unable to retrieve status query information`
  `Network is unreachable`
- `Error: LB_API_OPEN_ALL`
  `Unable to open connection with any LB supplied`

Si se ha producido alguna excepción que impide que se pueda realizar la consulta, se le notifica al usuario. En caso contrario, se procesa la información obtenida. Una vez procesada toda la información, se genera la página Web con la información del registro de eventos producidos durante la ejecución del trabajo y con la información correspondiente al estado del trabajo.

[logging_info]

En esta función se procesa la información correspondiente al registro de eventos producidos durante la ejecución del trabajo. Para cada evento, se obtiene el componente o servicio que lo genera y la fecha y hora de registro.

```
local (@info) = ('Event:','- source','- timestamp');

# Logging database
@ds0 = `grep -e '$info[0]' /tmp/log/$uid`;
@ds1 = `grep -e '$info[1]' /tmp/log/$uid`;
@ds2 = `grep -e '$info[2]' /tmp/log/$uid`;
```

[status_info]

En esta función se procesa la información correspondiente al estado del trabajo. Se obtiene el estado actual, la descripción del estado, el recurso computacional que lo ha ejecutado, y la fecha y hora de registro.

```
@info = ('Current Status:','Status Reason:','Destination:','reached on:');

# Status database
chomp(($data,$ds[0]) = split ($info[0],`grep '$info[0]' /tmp/log/$uid`));
chomp(($data,$ds[1]) = split ($info[1],`grep '$info[1]' /tmp/log/$uid`));
chomp(($data,$ds[2]) = split ($info[2],`grep '$info[2]' /tmp/log/$uid`));
chomp(($data,$ds[3]) = split ($info[3],`grep '$info[3]' /tmp/log/$uid`));
```

Si el trabajo está en proceso de ejecución (`check`), el sistema repite la petición cada 30 segundos y ofrece al usuario la posibilidad de cancelar la ejecución del trabajo (***Cancel***).

```
# Cancel allowed
$check=1;
```

```
# Check job status
local (@stat) = ('Aborted','Cancelled','Cleared','Success','Waiting');
for ($i=0; $i < scalar(@stat); $i++) { if (index($ds[0],$stat[$i]) > 0)
{ $check=0 }}
```

[job_info]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando solicita monitorizar la ejecución de un trabajo. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.
  ```
  system ("$exec header monitor $uid $vo");
  ```

- Se definen los parámetros correspondientes al identificador del trabajo y al periodo de consulta de estado en el que se solicita la información específica del trabajo.
  ```
  <input name="jobId" type="hidden" value="$job">
  <input name="jobSt" type="hidden" value="$stat">
  ```

- Se genera la información correspondiente al registro de eventos producidos durante la ejecución del trabajo. Esta información se visualiza a través de una capa (*layer*) definida en la hoja de estilos `style.css` por la clase `scroll`
  ```
  <div class="scroll">
  ```

- Se genera la información correspondiente al estado del trabajo. Si el trabajo está en proceso de ejecución (`check`):

  – Se ofrece al usuario la posibilidad de cancelar la ejecución del trabajo (***Cancel***).
  ```
  <input name="request" type="submit" value="Cancel">
  ```

  – Se definen los parámetros necesarios para generar de forma automática una petición cada 30 segundos. Esta petición se realiza a través de la variable `reload`
  ```
  <input name="time" type="text" value="30" class="clock" readonly>
  <body onLoad="reload()">
  ```

- Por último, se ofrece al usuario la posibilidad de regresar a la anterior solicitud de información del trabajo (***Back***)

```
<input name="request" type="submit" value="Back">;
```

- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy

```
system ("$exec footer $uid");
```

[exit_code]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando se genera una excepción. Los códigos de excepción correspondientes a este módulo, son los siguientes:

```
0 - Proxy certificate not valid
1 - Unable to contact Logging & Bookeeping Server
```

3.5.15. logout

En este módulo, ubicado en el directorio `/home/web/task`, se procesan las siguientes solicitudes de servicio:

- Finalizar la sesión (***Logout***)

Los parámetros para la ejecución de este módulo son los siguientes:

```
($opt,$uid,$vo) = @ARGV;
```

- La variable `opt` corresponde a la petición que se debe procesar.
- Cada petición está asociada a un identificador de usuario (`uid`) y su correspondiente Organización Virtual (`vo`).

Se pueden realizar las peticiones: ***Request*** y ***Logout***.

```
switch ($opt) {
  case 'Request' { &logout_request }
  case 'Logout'  { &logout }
  else { exit }}
```

[logout_request]

Esta función corresponde a la generación de la página Web que el usuario puede visualizar cuando selecciona la opción ***Logout***. Antes de finalizar la sesión, se le pide al usuario que confirme la petición. Para generar esta página, se utilizan las funciones proporcionadas por el módulo **shared**

- Se genera el código correspondiente a la cabecera de la página Web, en el que se define el identificador de usuario (`uid`), su Organización Virtual (`vo`) y los parámetros correspondientes al menú de opciones.

  ```
  system ("$exec header logout $uid $vo");
  ```

- Se definen los parámetros necesarios para que, en caso de confirmación, se pueda finalizar la sesión.

  ```
  <input name="request" type="hidden" value="Logout">
  <body onLoad="logout()">
  ```

- Se genera el código correspondiente al pié de página, donde se puede comprobar la validez del certificado proxy

```
system ("$exec footer $uid");
```

[logout]

Una vez que el usuario confirma la petición, el sistema finaliza la sesión revocando el certificado proxy creado al inicio de la misma. El comando Globus correspondiente a este servicio es `grid-proxy-destroy`. Para la ejecución de este, es necesario definir la variable de entorno `X509_USER_PROXY` que identifica al certificado proxy de usuario.

```
# Certificate
export X509_USER_PROXY=/tmp/x509up_uid
```

Una vez establecidos todos los parámetros, se ejecuta el programa correspondiente a la revocación del certificado proxy de usuario. El resultado de la ejecución se almacena en el fichero `/tmp/log/uid`

```
# Destroy proxy certificate
system ("$exec $uid /opt/globus/bin/grid-proxy-destroy &> /tmp/log/$uid");
```

Se analiza el registro de errores correspondiente al comando `grid-proxy-destroy`

- `ERROR: Proxy file doesn't exist or has bad permissions`
- `ERROR: Couldn't find a valid proxy.`

Se elimina el fichero que registra la validez del certificado proxy de usuario

```
`rm -f /tmp/log/$uid.px`;
```

Una vez finalizada la sesión, el usuario podrá visualizar de nuevo la página de acceso al Interfaz de Usuario.

3.6. Lenguaje de descripción de trabajos

En esta sección se describen los distintos atributos que se pueden definir para la descripción de un trabajo. Los atributos del lenguaje de descripción de trabajos (JDL) representan información específica del trabajo, y determinan las operaciones que deben ser realizadas por el *Workload Management System* (WMS) para planificar el trabajo. A partir de los datos proporcionados por el usuario, el UI crea el fichero de descripción con los atributos necesarios para la ejecución del trabajo.

[Type]

Este atributo es obligatorio y corresponde al tipo de petición descrita en el fichero JDL. Los posibles valores para este atributo son los siguientes:

- Job
- DAG (Direct Acyclic Graph)
- Reservation
- Co-allocation.

Valor por defecto: Type = "Job"

[Dependencies]

Este atributo es opcional y sólo se puede especificar para un diagrama de dependencias entre trabajos (DAG). El tipo DAG corresponde al diagrama de dependencias entre trabajos, donde los datos y la ejecución de uno o más trabajos, depende de la ejecución de otro o más trabajos.

```
Dependencies = {
        { jobA, jobB },
        { jobA, jobC },
        { jobA, jobD },
        { { jobB, jobC }, jobE }
        { { jobC, jobD }, jobF }
};
```

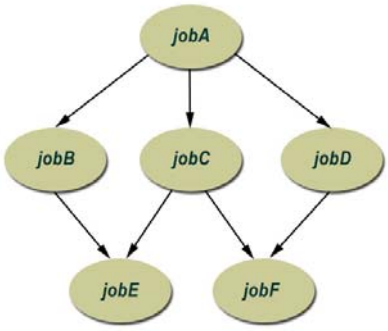


Fig 3.6. *Diagrama de dependencias entre trabajos*

[JobType]

Este atributo es obligatorio y corresponde al tipo de trabajo descrito en el fichero JDL. Los posibles valores para este atributo son los siguientes:

- Normal
- Interactive
- Checkpointable
- MPICH
- Partitionable
- Checkpointable, Interactive
- Checkpointable, MPICH

Valor por defecto: JobType = "Normal"

[Executable]

Este atributo es obligatorio y corresponde al nombre del fichero ejecutable. Este fichero puede ser una secuencia de comandos que ejecuten uno o más programas registrados en el CE remoto. En este caso, se deben especificar las ruta absolutas de los programas a ejecutar, así como las variables de entorno necesarias para su ejecución.

La otra posibilidad es proporcionar el nombre de un fichero ejecutable transferido al UI por el usuario. En este caso, el fichero ejecutable se transfiere del UI al CE para su ejecución y debe de especificarse en el atributo ***InputSandBox***

Executable = "exec_filename";
InputSandBox = {"exec_filename", ... };

[Arguments]

Este atributo es opcional y corresponde a los diferentes argumentos que se proporcionan al fichero ejecutable.

Arguments = "arg_1 arg_2 arg_3 ... arg_n";

[StdInput]

Este atributo es opcional y corresponde al fichero estándar de entrada de datos que se proporciona al fichero ejecutable. El fichero de datos se transfiere del UI al CE y debe de especificarse en el atributo ***InputSandBox***. Este atributo no puede ser especificado para trabajos interactivos.

StdInput = "stdin_filename";
InputSandBox = {"exec_filename", "stdin_filename" ... };

[StdOutput]

Este atributo es opcional y corresponde al fichero estándar de los datos que se obtienen de la ejecución del trabajo. Para solicitar posteriormente este fichero, una vez ejecutado el trabajo, debe de especificarse en el atributo ***OutputSandBox***. Este atributo no puede ser especificado para trabajos interactivos.

StdOutput = "stdout_filename";
OutputSandBox = {"stdout_filename", ... };

[StdError]

Este atributo es opcional y corresponde al fichero estándar de los errores que se obtienen de la ejecución del trabajo. Para solicitar posteriormente este fichero, una vez ejecutado el trabajo, debe de especificarse en el atributo ***OutputSandBox***. Este atributo no puede ser especificado para trabajos interactivos.

StdError = "stderr_filename";
OutputSandBox = {"stdout_filename", "stderr_filename", ... };

[InputSandBox]

Este atributo es opcional y corresponde al conjunto de ficheros de datos que se proporcionan al fichero ejecutable. Este conjunto de ficheros de datos se transfieren del UI al CE.

InputSandBox = { ... "file1", "file2", ... "filen" };

[OutputSandBox]

Este atributo es opcional y corresponde al conjunto de ficheros de datos que se obtienen de la ejecución de un trabajo. Este conjunto de ficheros se solicitan ejecutando el comando Globus ***edg-job-get-output***

OutputSandBox = { ... "file$_1$", "file$_2$", ... "file$_n$" };

[Environment]

Este atributo es opcional y corresponde al conjunto de variables de entorno necesarias para la ejecución del trabajo.

Environment = { "var$_1$=value$_1$", "var$_2$=value$_2$", ... "var$_n$=value$_n$" };

[InputData]

Este atributo es obligatorio si se especifica el atributo ***DataAccessProtocol*** y corresponde al conjunto de Logical File Names (LFN) o Grid Unique Identifiers (GUID) necesarios para la ejecución del trabajo. Estos ficheros están almacenados en SEs y publicados en los catálogos de réplica. El Resource Broker, utiliza los LFNs y GUIDs para localizar al CE más apropiado para la ejecución del trabajo.

InputData = {"lfn:LFN_name", ... "guid:GUID", ... };

[DataAccessProtocol]

Este atributo es obligatorio si se especifica el atributo ***InputData*** y corresponde al conjunto de protocolos de acceso al SE para la obtención de los ficheros especificados en el atributo ***InputData***. Los posibles valores para este atributo son: ***file***, ***gridftp*** y ***rfio***

DataAccessProtocol = {"protocol" ... };

[OutputData]

Este atributo es opcional y permite definir para cada fichero de datos, que se obtiene de la ejecución de un trabajo, el LFN con el que debe registrarse y el SE donde debe almacenarse (***copyAndRegister***).

En la definición de este atributo, se utiliza un conjunto de atributos que sólo pueden especificarse dentro del este atributo:

- ***OutputFile***

  Este atributo es obligatorio si se especifica el atributo ***OutputData*** y corresponde al fichero de datos que se obtiene de la ejecución del trabajo.

- ***StorageElement***

  Este atributo es opcional y corresponde al Uniform Resource Identifier (URI) del SE donde se almacenará el fichero de datos especificado por el atributo ***OutputFile***. Si no se especifica este atributo, se selecciona el SE asociado al CE.

- ***LogicalFileName***

  Este atributo es opcional y corresponde al LFN asociado al fichero de datos especificado por el atributo ***OutputFile***, para su registro en el Replica Catalogue. Si no se especifica este atributo, el WMS le asigna un GUID.

```
OutputData = {
        [
                OutputFile = “filename”;
                StorageElement = “SE_name”;
                LogicalFileName = “lfn:LFN_name”;
        ] ...
                };
```

[OutputSE]

Este atributo es opcional y corresponde al Uniform Resource Identifier (URI) del SE donde se almacenará los datos obtenidos de la ejecución del trabajo. Normalmente un URI consta de dos partes:

- Identificador del método de acceso (protocolo) al recurso
- Nombre del recurso

```
OutputSE = “SE_name”;
```

[VirtualOrganisation]

Este atributo es obligatorio y corresponde al nombre de la Organización Virtual en la que se realizan las peticiones de usuario a través de sus credenciales. El valor de este atributo, por orden de precedencia, lo determina:

- El certificado proxy de usuario (si contiene información VOMS)
- Los comandos Globus a través de los argumentos **`--vo`** o **`--config-vo`**
- La variable de entorno `EDG_WL_UI_CONFIG_VO`
- El atributo ***VirtualOrganisation***
- El parámetro `DefaultVO` definido en el fichero de configuración `$EDG_WL_LOCATION/etc/edg_wl_ui_cmd_var.conf`

VirtualOrganisation = “swetest”;

[RetryCount]

Este atributo es opcional y corresponde al máximo número de intentos que el WMS debe de realizar para asignar el trabajo a un CE. Para deshabilitar este mecanismo, se especifica `0` como valor de este atributo. El valor por defecto de este atributo lo determina el parámetro `RetryCount` definido en el fichero de configuración `$EDG_WL_LOCATION/etc/edg_wl_ui_cmd_var.conf`

RetryCount = 3;

[MyProxyServer]

Este atributo es opcional y corresponde a la dirección (<***host fqdn***>) del servidor donde el usuario tiene registrado su certificado proxy. Este registro se realiza a través del comando Globus ***myproxy-init***. El valor de este atributo, lo determina:

- El comando Globus ***myproxy-init*** a través del argumento ***–s***
- La variable de entorno `MYPROXY_SERVER`
- El atributo ***MyProxyServer***
- El parámetro `MyProxyServer` definido en el fichero de configuración `$EDG_WL_LOCATION/etc/<vo_name>/edg_wl_ui.conf`

La definición de este atributo, activa el mecanismo de renovación del certificado proxy para aquellos trabajos que requieran un tiempo de ejecución superior a la validez del certificado proxy.

MyProxyServer = "host_fqdn";

[HLRLocation]

Este atributo es obligatorio cuando el servicio WMS de contabilidad está activado. Corresponde al registro que utiliza el sistema WMS para notificar al usuario el coste de la ejecución del trabajo.

El valor por defecto de este atributo, lo determina el parámetro `HLRLocation` definido en el fichero de configuración `$EDG_WL_LOCATION/etc/<vo_name>/edg_wl_ui.conf`

HLRLocation = "<host fqdn>:<port>:<X509_contact>" ;

[NodeNumber]

Este atributo es obligatorio cuando el tipo de trabajo es MPICH. Corresponde al número de nodos que se necesitan para ejecutar el trabajo. El Resource Broker (RB) utiliza este atributo para seleccionar un CE con un número de WN igual o superior al número especificado.

NodeNumber = 8;

[JobSteps]

Este atributo es opcional y sólo se puede especificar cuando el tipo de trabajo es Checkpointable. Corresponde al número de etapas, o conjunto de etiquetas utilizadas para la revisión del trabajo. Cada vez que se revisa el trabajo, se registra su estado con los datos correspondientes de la ejecución, de forma que, en caso necesario, se pueda reasignar el trabajo para continuar su ejecución.

JobSteps = 1000;
JobSteps = {"label$_1$","label$_2$", … "label$_n$"};

[CurrentStep]

Este atributo es obligatorio cuando el tipo de trabajo es Checkpointable. Corresponde al índice inicial, correspondiente a la etapa o etiqueta, que debe de utilizarse para la revisión del trabajo.

Valor por defecto: CurrentStep = 0;

[ListenerPort]

Este atributo es opcional y sólo se puede especificar cuando el tipo de trabajo es Interactive. Corresponde al puerto de comunicaciones utilizado para la transmisión de datos.

ListenerPort = 44000;

[Requirements]

Este atributo es obligatorio y corresponde al conjunto de requerimientos específicos que debe cumplir un CE para la ejecución del trabajo. Este conjunto se expresa en base a los atributos CE registrados en el Information Service (IS).

Requirements = (other.GlueCEattribute1 == "value1") && ...
(other.GlueCEattributen == "valuen");

El valor por defecto de este atributo, se puede establecer en el fichero de configuración `$EDG_WL_LOCATION/etc/edg_wl_ui_cmd_var.conf`

Valor por defecto: Requirements = other.GlueCEStateStatus == "Production";

Dentro de este atributo, para definir el conjunto de requerimientos específicos que debe cumplir un SE asociado al CE, se utilizan los siguientes atributos:

- anyMatch()
- whichMatch()
- allMatch()

[Rank]

Este atributo es obligatorio y establece la preferencia, dentro del conjunto de requerimientos específicos definidos por el atributo ***Requirements***, que debe cumplir un CE para la ejecución del trabajo.

Rank = other.GlueCEattribute1 - ... other.GlueCEattributen;

El valor por defecto de este atributo, se puede establecer en el fichero de configuración `$EDG_WL_LOCATION/etc/edg_wl_ui_cmd_var.conf`

Valor por defecto: Rank = other.GlueCEStateFreeCPUs;

Conclusiones

La resolución de muchos problemas de cálculo científico sigue siendo un desafío, tanto por su dificultad técnica, como por la necesidad de recursos. Actualmente se están desarrollando nuevos tipos de aplicaciones de altas exigencias que no pueden ejecutarse en una única máquina; tales aplicaciones necesitan del desarrollo de redes virtuales de supercomputadores o de metacomputadores, de entornos de ejecución que contengan redes de alta velocidad, bases de datos, instrumentación científica, etc. distribuidos geográficamente.

La creación de una red de supercomputadores permite desarrollar entornos de supercomputación con un más asequible coste. La experiencia con este tipo de redes ha demostrado fehacientemente que existe un gran número de aplicaciones de considerable importancia científica y económica que pueden beneficiarse de las características de la computación Grid. La evolución de los recursos de red ha hecho factible esta posibilidad. La solución viene dada por dos acciones conjuntas: agregar y compartir; la red permitirá de forma eficiente usar recursos distribuidos.

Hoy en día, cálculo, almacenamiento e información, entre otros, constituyen los principales recursos utilizados y compartidos mediante la red. La evolución de las redes de comunicaciones de alta velocidad dedicadas a la investigación y de las tecnologías y aplicaciones colaborativas está creando un escenario idóneo para la interacción entre investigadores. La implementación de bancos de prueba, utilizando herramientas Grid, es totalmente necesaria para impulsar el desarrollo de la *e-Ciencia* en España. La *e-Ciencia* se entiende como el conjunto de actividades científicas desarrolladas mediante el uso de la tecnología Grid. El desarrollo de la *e-Ciencia*, en general, tendrá un impacto científico con la explotación eficiente de centros o recursos de excelencia y la existencia de nuevas formas de compartir conocimiento; un impacto tecnológico, ya que permitirá abrir nuevos mercados y nuevas formas de colaboración y desarrollo de proyectos; y un impacto social, pues proveerá de acceso para vencer la brecha tecnológica; demorar su desarrollo alejaría demasiado a España de los países que ya disponen de un programa de *e-Ciencia*.

La tecnología Grid se presenta como una de las más prometedoras posibilidades en el entorno de los recursos computacionales distribuidos; del mismo modo que el World Wide Web, desarrollado en el Laboratorio Europeo de Física de Partículas (CERN), ha proporcionado una vía de colaboración básica dentro de la red global Internet, será necesaria una infraestructura mucho más potente y compleja para soportar los proyectos de I+D de empresas y organizaciones científicas previstos en los próximos años.

El nuevo protocolo de Internet IPv6 permitirá trabajar con una Internet más rápida y accesible. Una de las ideas clave en la superación de las limitaciones actuales de Internet IPv4 es la aparición de nuevos niveles de servicio que harán uso de la nueva capacidad de la red para intercomunicar los ordenadores. Este avance en la comunicación permitirá el avance de las ideas de *Grid Computing* al utilizar como soporte la altísima conectividad de Internet. Es por ello que 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. La tecnología Grid, constituirá la próxima generación de Internet; por eso es nuestro propósito seguir trabajando en el desarrollo de un programa de *e-Ciencia*

Por último, como adelanto de nuestras líneas de trabajo futuras, analizamos la posible estructura de un programa de *e-Ciencia*, en base a su arquitectura, organización y los requerimientos necesarios para crear un centro de e-Ciencia. Los Grids computacionales definen una ***arquitectura*** "genérica" de construcción de aplicaciones junto con una estructura distribuida no solo de infraestructura sino también en desarrollo de middleware y aplicaciones. Las fronteras entre las diferentes capas de esta arquitectura, y de su distribución, son las que definen la estructura de un programa de e-Ciencia/Grid. Uno de los puntos básicos es la idea de una "infraestructura" Grid común. El punto de unión actual es la red académica nacional (RedIRIS) y europea (Gèant), conectando los recursos de computación; estos recursos ser multidisciplinares (p.e. los recursos de computación de un centro de Supercomputación, o del centro de cálculo de una Universidad, o de un cluster de un Centro de Investigación).

La existencia de un middleware común permitirá aprovechar de forma mucho más efectiva estos recursos, sobre los que se establecerán para su uso diferentes Organizaciones Virtuales (VO). Este proceso puede extenderse a niveles comunes a las aplicaciones, no definidos en el esquema anterior, con puntos como acceso a bases de datos, técnicas de data-mining, visualización, etc.

Las áreas de trabajo propuestas de acuerdo al esquema anterior son: (i) Puesta en marcha de infraestructura Grid y operación de la misma en testbeds y en modo producción. Disponibilidad de recursos existentes de computación en esquema Grid, incorporación de nuevos recursos, procedimientos de instalación automatizados, "*autonomic computing*". Autoridades de certificación. Soporte de VO. Acceso a los recursos de Red. Coordinación con los operadores de red. (ii) Desarrollo básico de tecnología común para proyectos Grid. Incluye aportaciones originales o refuerzo de líneas existentes en middleware básico y común para desarrollo de aplicaciones. (iii) Desarrollo de aplicaciones en el entorno Grid: librerías comunes y aplicaciones finales en cada área temática. (iv) Transferencia de tecnología, Cursos, Difusión y Proyectos con Empresas

Una ***organización*** a través de centros de e-Ciencia o similares tiene varias ventajas: (i) Posibilidad de co-financiación institucional/local/regional de los proyectos. (ii) Soporte local de usuarios y desarrolladores. (iii) Facilidad de difusión, contactos cercanos con recursos locales y empresas. (iv) Esquema similar al del soporte de Red. (v) Competencia saludable.

Las funciones de los centros son: (i) Aportar recursos computacionales en un entorno Grid, y mantener operativos estos recursos en el marco de un Grid nacional. (ii) Promover proyectos en áreas temáticas de interés regional y especialmente en colaboración con empresas. (iii) Difusión de la iniciativa Grid. (iv) Participar en la coordinación a nivel nacional.

La descripción de los centros de e-Ciencia se detalla mas adelante. Su financiación se realiza a través de los Equipos indicados a continuación, y está ligada a la participación en proyectos.

Equipo de Soporte de Infraestructura Grid. Este equipo estará distribuido entre los centros de e-Ciencia, pero funcionara de forma coordinada. La dotación a aportar desde fondos institucionales, locales o regionales será: (i) Complementar los recursos humanos de proyectos nacionales o internacionales para mantener operativo el equipo de soporte de infraestructura Grid. (ii) Complementar recursos de computación necesarios en un entorno Grid (p.e. servidor "*gatekeeper*", servidores de publicación de recursos, etc.)

La financiación puede realizarse mediante acciones especiales complementarias de los proyectos realizados en los centros de e-Ciencia correspondientes. Este equipo operará un testbed nacional. Existirá un centro responsable de Coordinación de Soporte que organizará un servicio de asistencia continua incluyendo un HelpDesk. Funcionará siguiendo el modelo definido por el proyecto europeo EGEE. Además este equipo coordinará los temas relacionados con la adquisición e instalación de infraestructura de computación, y relación con proveedores.

Equipo de Soporte de Red, Seguridad y Herramientas de Colaboración. Este equipo estará coordinado por RedIRIS, y contaría con una persona dedicada en cada centro de e-Ciencia. Entre las tareas de este equipo está la consideración de todos los aspectos relativos a la red, el establecimiento y operación de la Autoridad de Certificación, y la instalación y mantenimiento de herramientas colaborativas (desde listas de correo, Web, a videoconferencia, salas avanzadas, etc.).

Equipo de Desarrollo de la Arquitectura Grid y elaboración de middleware. Este equipo reunirá a profesionales del campo de computación distribuida, contando con la información del equipo de Aplicaciones. Se financiará mediante proyectos del programa TIC, coordinados en su preparación a través de este equipo. Las áreas básicas relacionadas abarcan desde middleware básico, herramientas de computación distribuida, acceso a bases de datos, etc.

Equipo de desarrollo e integración de Aplicaciones. Este equipo coordinará el desarrollo de aplicaciones Grid, promoviendo las mismas a través de proyectos adecuados bien de los distintos programas nacionales, o europeos.

El Equipo contará con un asesor por área, y coordinadores de integración del Equipo de Soporte de Infraestructura y de Middleware. Estará abierto a nuevas áreas siguiendo los criterios de programas nacionales y europeos. Cada área contará con una financiación específica para posibilitar la coordinación y en su caso la gestión de las VO correspondientes. En principio esta financiación se solicitará dentro de uno de los proyectos correspondientes en el área. El proyecto se asignará formalmente al centro de e-Ciencia correspondiente al coordinador del mismo.

Equipo de transferencia de tecnología. Este equipo promoverá la participación en proyectos con empresas. Coordinará las actividades de cada centro de e-Ciencia, su difusión y prospectiva en iniciativas internacionales. La coordinación entre todos los equipos anteriores se llevará a cabo mediante el comité ejecutivo de IRISGrid, que contará con presencia de los coordinadores de los centros de e-Ciencia, y de los equipos antes referidos. El comité ejecutivo estará asesorado por un comité técnico y un comité científico; el comité técnico actuará además en la revisión de los proyectos presentados y la evaluación de los realizados, e incluirá expertos de la comunidad Grid internacional; el comité científico asesorará en las áreas de aplicación, y recogerá la opinión de los responsables (gestores de programas, etc.).

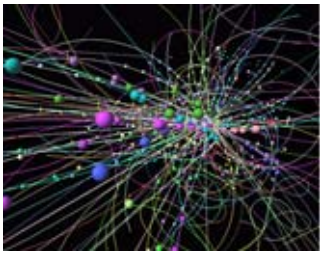
Los ***centros de e-Ciencia***, aún siendo virtuales, contarán con un coordinador y miembros adscritos. La adscripción será voluntaria y abierta, bien como usuario o con aportación de infraestructura.

Cada centro de e-Ciencia deberá tener actividad en infraestructura conectada en Grid, promoción de una línea temática de aplicaciones, desarrollo de proyectos con la participación significativa de empresas, administración de recursos y diseminación. Cada centro deberá identificar dentro del personal adscrito al mismo un coordinador, un responsable de infraestructura Grid en producción, un responsable de recursos de red, un promotor de proyectos con empresas y un responsable de diseminación. Todos los responsables participarán a su vez en la organización nacional dentro del comité ejecutivo, comités técnicos, foros industriales y de diseminación de IRISGrid.

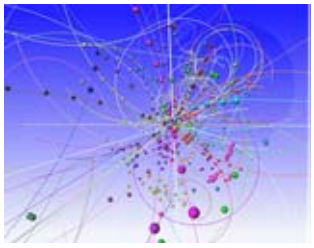
Algunas aplicaciones interesantes y estimulantes en las que ya tenemos experiencia y que animan a seguir en esta línea de trabajo son las siguientes:

Física de Altas Energías

***Análisis de datos a partir de grandes bases de datos distribuidas*.**



- El origen de la "masa" de todas las partículas, está asociado a una partícula fundamental enunciada pero todavía no descubierta: Higgs Boson
- A partir del acelerador de partículas – Large Hadron Collider (LHC), se aceleran protones a una energía suficiente para producir una partícula varios cientos de veces más pesada: Higgs Boson, la última pieza clave en el Modelo Estándar para comprender el origen de la "masa".

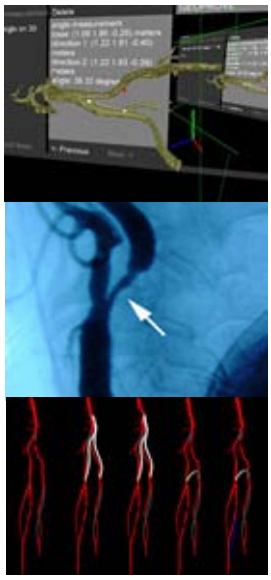


- El problema es que todas las colisiones son registradas por sofisticados detectores (CMS)
- La información es almacenada en bases de datos distribuidas con un volumen de datos de millones de Gb.
- Sólo una pequeña cantidad de estas complejas colisiones, produce un Higgs Boson.

- Se utilizan la tecnología Grid para usar técnicas de filtrado on-line junto con sofisticados algoritmos matemáticos, como redes neuronales, para realizar el correspondiente análisis.
- Se desarrolla un portal Web, para el interfaz gráfico de usuario (GUI).

Biomedicina

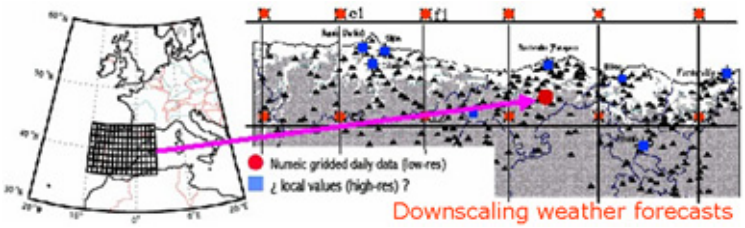
Simulación interactiva y visualización en cirugía vascular

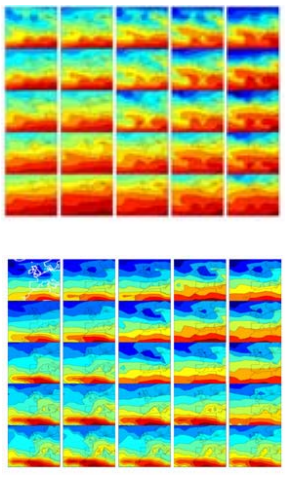


- Se obtienen los datos a partir de instrumentación médica (escáner) y se realiza un diagnóstico previo.
- Se genera el modelo arterial en 3D correspondiente. Sobre este modelo se confirma el diagnóstico previo.
- En base a este diagnóstico, se localizan las deficiencias en la estructura de las arterias, y se proponen varias alternativas para su modificación.
- Para cada alternativa y empleando la tecnología Grid, se realiza una simulación del flujo sanguíneo.
- Con los resultados obtenidos en tiempo real, se realiza el tratamiento e intervención adecuados.

Polución Atmosférica

Análisis de datos a partir de modelos de circulación atmosférica

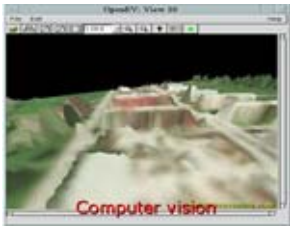


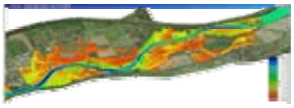


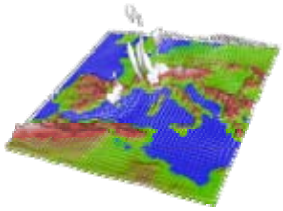
- A partir de patrones de circulación atmosférica, se realizan las predicciones correspondientes y se ajusta el modelo de polución atmosférica en la zona próxima a una central térmica.
- Se realiza data-mining en bases de datos obtenidas a partir de los modelos de circulación atmosférica, mediante tecnología Grid.
- Necesidad de respuesta en tiempo real para realizar diferentes estimaciones, de acuerdo con las predicciones meteorológicas.

Meteorología

Predicción de desastres por inundación







- Se obtienen los datos a partir de estaciones meteorológicas e hidrológicas, radares meteorológicos, y sistemas de adquisición y procesamiento de datos vía satélite.
- También se obtienen datos de otras fuentes externas de información (servicios hidrológicos de otros países, etc)
- A través de la infraestructura Grid, se realizan las simulaciones de los correspondientes modelos hidrológicos hidráulicos y meteorológicos analizando los resultados.
- Se contrastan las predicciones con la opinión de otros expertos, también a través de la infraestructura Grid.
- En caso necesario, se vuelven a realizar las simulaciones con nuevos parámetros para analizar el correspondiente impacto.
- Se toman las medidas necesarias.

Bibliografía

[1] **HAND: Highly Available Dynamic Deployment Infrastructure for Globus Toolkit 4**. L. Qi, H. Jin, I. Foster, J. Gawor, 2006.

[2] **Virtual Workspaces: Achieving Quality of Service and Quality of Life in the Grid**. K. Keahey, I. Foster, T. Freeman, and X. Zhang. *Scientific Programming Journal*, 2006.

[3] **A Multipolicy Authorization Framework for Grid Security.** Bo Lang, Ian Foster, Frank Siebenlist, Rachana Ananthakrishnan, Tim Freeman. Accepted by the IEEE NCA06 Workshop on Adaptive Grid Computing (*Proc. Fifth IEEE Symposium on Network Computing and Application*), Cambridge, USA, July 24-26, 2006.

[4] **Globus Toolkit Version 4: Software for Service-Oriented Systems**. I. Foster. *IFIP International Conference on Network and Parallel Computing*, Springer-Verlag LNCS 3779, pp 2-13, 2005.

[5] **The Open Grid Services Architecture, Version 1.0**. I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich. *Informational Document, Global Grid Forum (GGF)*, January 29, 2005.

[6] **MDS: The Gobus Monitoring and Discovery System.** Ben Clifford. *Cluster World's On the Grid column*, February 2005.

[7] **Web Services and the Grid.** Lee Liming. *Cluster World's On the Grid column*, March 2005

[8] **The Globus® Toolkit Replica Location Service.** Ann L. Chervenak. *Cluster World's On the Grid column*. April 2005

[9] **Service-Oriented Science**. I. Foster. *Science*, vol. 308, May 6, 2005.

[10] **Virtual Workspaces in the Grid**. K. Keahey, I. Foster, T. Freeman, X. Zhang, D. Galron. *Proceedings of Europar 2005*, Lisbon, Portugal, September, 2005.

[11] **State and Events for Web Services: A Comparison of Five WS-Resource Framework and WS-Notification Implementations**. M. Humphrey, G. Wasson, K. Jackson, J. Boverhof, M. Rodriguez, Joe Bester, J. Gawor, S. Lang, I. Foster, S. Meder, S. Pickles, and M. McKeown, *4th IEEE International Symposium on High Performance Distributed Computing (HPDC-14)*, Research Triangle Park, NC, 24-27 July 2005.

[12] **X.509 Proxy Certificates for Dynamic Delegation**. V. Welch, I. Foster, C. Kesselman, O. Mulmo, L. Pearlman, S. Tuecke, J. Gawor, S. Meder, F. Siebenlist. *3rd Annual PKI R&D Workshop*, 2004.

[13] **End-to-End Quality of Service for High-end Applications**. I. Foster, M. Fidler, A. Roy, V, Sander, L. Winkler. *Computer Communications*, 27(14):1375-1388, 2004.

[14] **The Grid.** Ian Foster. *Cluster World's On the Grid column*. January 2004

[15] **Security and Credential Management on the Grid.** Sam Lang and Sam Meder. *Cluster World's On the Grid column*. January 2004.

[16] **Modeling Stateful Resources with Web Services v. 1.1**. I. Foster (ed), J. Frey (ed), S. Graham (ed), S. Tuecke (ed), K. Czajkowski, D. Ferguson, F. Leymann, M. Nally, I. Sedukhin, D. Snelling, T. Storey, W. Vambenepe, S. Weerawarana, March 5, 2004.

[17] **The WS-Resource Framework**. K. Czajkowski, D. F. Ferguson, I. Foster, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, W. Vambenepe. March 5, 2004.

[18] **From Open Grid Services Infrastructure to WS-Resource Framework: Refactoring & Evolution**. K. Czajkowski, D. Ferguson, I. Foster, J. Frey, S. Graham, T. Maguire, D. Snelling, S. Tuecke. March 5, 2004.

[19] **Publish-Subscribe Notification for Web services**. S. Graham (ed), P. Niblett (ed), D. Chappell, A. Lewis, N. Nagaratnam, J. Parikh, S. Patil, S. Samdarshi, I. Sedukhin, D. Snelling, S. Tuecke, W. Vambenepe, B. Weihl, March 5, 2004.

[20] **Using the Globus Toolkit with Firewalls.** Olle Mulmo and Von Welch. *Cluster World's On the Grid column*. March 2004.

[21] **Standardizing the Grid.** Lee Liming, Tom Garritano, Steve Tuecke. *Cluster World's On the Grid column*. April 2004.

[22] **Performance Analysis of the Globus Toolkit Monitoring and Discovery Service, MDS2**. X. Zhang and J. Schopf. *Proceedings of the International Workshop on Middleware Performance (MP 2004), part of the 23rd International Performance Computing and Communications Workshop (IPCCC)*, April 2004.

[23] **The Globus™ eXtensible Input/Output (XIO) System.** William Allcock and John Bresnahan. *Cluster World's On the Grid column*. May 2004.

[24] **Reliable Data Transport: A Critical Service for the Grid**. W.E. Allcock, I. Foster, R. Madduri. *Building Service Based Grids Workshop, Global Grid Forum 11*, June 2004.

[25] **Performance and Scalability of a Replica Location Service**. A.L. Chervenak, N. Palavalli, S. Bharathi, C. Kesselman, R. Schwartzkopf. *Proceedings of the International IEEE Symposium on High Performance Distributed Computing (HPDC-13)*, June 2004.

[26] **Testing in the Grid Environment.** Charles A. Bacon. *Cluster World's On the Grid column*. June 2004

[27] **Grid Packaging Software.** Scott Gose. *Cluster World's On the Grid column*. July 2004

[28] **Agreement-Based Interactions for Experimental Science**, K. Keahey, T. Araki, and P. Lane. In Proceedings of *Europar*, August, 2004.

[29] **Monitoring Clusters and Grids.** Jennifer M. Schopf and Ben Clifford. *Cluster World's On the Grid column*. August 2004

[30] **Open Grid Services Architecture Use Cases**. I. Foster, D. Gannon, H. Kishimoto, J. Von Reich. *Information Document, Global Grid Forum (GGF)*, October 28, 2004.

[31] **MPICH-G2: An MPI for Grids.** Nicholas T. Karonis. *Cluster World's On the Grid column*. November 2004

[32] **From Sandbox to Playground: Dynamic Virtual Environments in the Grid**. K. Keahey, K. Doering, and I. Foster. *Proceedings of 5th International Workshop in Grid Computing (Grid 2004)*, Pittsburgh, PA, November 2004.

[33] **Descripción, análisis e integración de plataformas de cómputo distribuido.** J. Ruedas, 2003.

[34] **Transparent Grid Computing: A Knowledge-Based Approach**. J. Blythe, E. Deelman, Y. Gil, C. Kesselman. *IA AI 2003*, 2003.

[35] **The Role of Planning in Grid Computing**. J. Blythe, E. Deelman, Y. Gil, C. Kesselman, A. Agarwal, G. Mehta, K. Vahi. *ICAPS 2003*, 2003.

[36] **Grid Resource Management**. J. Nabrzyski, J.M. Schopf, J. Weglarz (Eds). *Kluwer Publishing*, Fall 2003.

[37] **The Community Authorization Service: Status and Future**. Ian Foster, Carl Kesselman, Laura Pearlman, Steven Tuecke, and Von Welch. *In Proceedings of Computing in High Energy Physics 03 (CHEP '03)*, 2003.

[38] **Fine-Grain Authorization Policies in the GRID: Design and Implementation**. K. Keahey, V. Welch, S. Lang, B. Liu, S. Meder. *1st International Workshop on Middleware for Grid Computing*, 2003.

[39] **Globus: An Infrastructure for Resource Sharing.** Tom Garritano. *Cluster World's On the Grid column*, 2003

[40] **Planning for Workflow Construction and Maintenance on the Grid**. J. Blythe, E. Deelman, Y. Gil. *ICAPS 2003 Workshop on Planning for Web Services*, 2003.

[41] **Mapping Abstract Complex Workflows onto Grid Environments**. E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, K. Blackburn, A. Lazzarini, A. Arbree, R. Cavanaugh, S. Koranda. *Journal of Grid Computing*, 1(1), 25-39, 2003.

[42] **MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface**. N. Karonis, B. Toonen, and I. Foster. *Journal of Parallel and Distributed Computing*, 2003.

[43] **Grid Database Access and Integration: Requirements and Functionalities**. M.P. Atkinson, V. Dialani, L. Guy, I. Narang, N.W. Paton, D. Pearson, T. Storey, and P. Watson. *Global Grid Forum Informational Document (GFD.13)*, March 2003.

[44] **GridFTP Protocol Specification** (*Global Grid Forum Recommendation GFD.20*). W. Allcock, editor. March 2003.

[45] **Security for Grid Services**. V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, S. Tuecke. *Twelfth International Symposium on High Performance Distributed Computing (HPDC-12)*, IEEE Press, June 2003.

[46] **Open Grid Services Infrastructure (OGSI) Version 1.0**. S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, P. Vanderbilt, D. Snelling; *Global Grid Forum Draft Recommendation*, June 2003.

[47] **Agreement-based Grid Service Management (OGSI-Agreement) (Draft 0)**. K. Czajkowski, A. Dan, J. Rofrano, S. Tuecke, and M. Xu. *Global Grid Forum, GRAAP-WG Author Contribution*, 12 June 2003.

[48] **A Performance Study of Monitoring and Information Services for Distributed Systems**. X. Zhang, J. Freschl, and J. Schopf. *Proceedings of HPDC*, August 2003.

[49] **Grid Resource Management: State of the Art and Future Trends.** Edited by Jarek Nabrzyski, Jennifer M. Schopf, Jan Weglarz. *Springer*; 1 edition (September, 2003)

[50] **Dynamic Creation and Management of Runtime Environments in the Grid**. K. Keahey, M. Ripeanu, and K. Doering. *Proceedings of Workshop on Designing and Building Web Services (GGF 9)*, Chicago, IL, October, 2003.

[51] **The Grid 2: Blueprint for a New Computing Infrastructure.** Edited by Ian Foster and Carl Kesselman. *Morgan Kaufmann*; 2 edition (November 18, 2003)

[52] **Conservative Scheduling: Using Predicted Variance to Improve Scheduling Decisions in Dynamic Environments**.L. Yang, J.M. Schopf, I. Foster. *Supercomputing 2003*, November 2003.

[53] **The Grid: A New Infrastructure for 21st Century Science**. I. Foster. *Physics Today*, 55(2):42-47, 2002.

[54] **Grids and Research Networks as Drivers and Enablers of Future Internet Architectures**. K. Baxevanidis, H. Davies, I. Foster, F. Gagliardi. *Computer Networks*, 2002.

[55] **Grid Services for Distributed System Integration**. I. Foster, C. Kesselman, J. Nick, S. Tuecke. *Computer*, 35(6), 2002.

[56] **A Community Authorization Service for Group Collaboration**. L. Pearlman, V. Welch, I. Foster, C. Kesselman, S. Tuecke. *Proceedings of the IEEE 3rd International Workshop on Policies for Distributed Systems and Networks*, 2002.

[57] **Fine-Grain Authorization for Resource Management in the Grid Environment**. K. Keahey, V. Welch. *Proceedings of Grid2002 Workshop*, 2002.

[58] **Toward a Framework for Preparing and Executing Adaptive Grid Programs**. D. Angulo, R. Aydt, F. Berman, A. Chien, K. Cooper, H. Dail, J. Dongarra, I. Foster, D. Gannon, L. Johnsson, K. Kennedy, C, Kesselman, M. Mazina, J. Mellor-Crummey, D. Reed, O. Sievert, L. Torczon, S. Vadhiyar, and R. Wolski. *IPDPS, 2002*.

[59] **SNAP: A Protocol for negotiating service level agreements and coordinating resource management in distributed systems**. K. Czajkowski, I. Foster, C. Kesselman, V. Sander, and S. Tuecke. *Lecture Notes in Computer Science*, 2537:153-183, 2002.

[60] **File and Object Replication in Data Grids**. H. Stockinger, A. Samar, B. Allcock, I. Foster, K. Holtman, and B. Tierney; *Journal of Cluster Computing*, 5(3)305-314, 2002.

[61] **Data Access and Integration Services on the Grid**. N. Paton, M.P. Atkinson, V. Dialani, D. Pearson, T. Storey, and P. Watson. *UK e-Science Programme Technical Report Series (UKeS-2002-03)*, February 2002.

[62] **Data Management and Transfer in High Performance Computational Grid Environments**. B. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal, S. Tuecke. *Parallel Computing Journal*, Vol. 28 (5), May 2002, pp. 749-771.

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

[64] **A Decentralized, Adaptive, Replica Location Service**. M. Ripeanu, I. Foster; *11th IEEE International Symposium on High Performance Distributed Computing (HPDC-11),* Edinburgh, Scotland, July 24-16, 2002.

[65] **Design and Evaluation of a Resource Selection Framework for Grid Applications**. D. Angulo, I. Foster, C. Liu, and L. Yang. *Proceedings of IEEE International Symposium on High Performance Distributed Computing (HPDC-11),* Edinburgh, Scotland, July 2002.

[66] **InfoGram: A Grid Service that Supports Both Information Queries and Job Execution**. G. von Laszewski, I. Foster, J. Gawor, A. Schreiber, C. Pena. *Proceedings of the 11th IEEE International Symposium on High-Performance Distributed Computing (HPDC-11),* IEEE Press, Edinburg, Scotland, July 2002.

[67] **What is the Grid? A Three Point Checklist**. I. Foster, *GRIDToday*, July 20, 2002.

[68] **Grids: Top Ten Questions**. J.M. Schopf and B. Nitzberg, *Scientific Programming, special issue on Grid Computing*, 10(2):103 - 111, August 2002.

[69] **Computational Grids in Action: The National Fusion Collaboratory**. K. Keahey, T. Fredian, Q. Peng, D.P. Schissel, M. Thompson, I. Foster, M. Greenwald, and D. McCune, *Future Generation Computer Systems*, 18:8, pg. 1005-1015, October 2002.

[70] **The Emergence of the Grid**. I. Foster. *Nature Yearbook of Science and Technolgy*, Nature Publishing Group, 2001.

[71] **The Anatomy of the Grid: Enabling Scalable Virtual Organizations**. I. Foster, C. Kesselman, S. Tuecke. *International J. Supercomputer Applications*, 15(3), 2001.

[72] **Grid Technologies & Applications: Architecture & Achievements**. I. Foster. *Intl Conference on Computing in High Energy and Nuclear Physics*, 2001.

[73] **Designing Grid-based Problem Solving Environments**. G. von Laszewski, I. Foster, J. Gawor, P. Lane, N. Rehn, M. Russell. *34th Hawai'i International Conference on System Science*, 2001.

[74] **An Online Credential Repository for the Grid: MyProxy**. J. Novotny, S. Tuecke, V. Welch. *Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10)*, IEEE Press, August 2001.

[75] **On Fully Decentralized Resource Discovery in Grid Environments**. A. Iamnitchi and I. Foster. *International Workshop on Grid Computing, Denver, CO, November 2001.*

[76] **Grid Information Services for Distributed Resource Sharing**. K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman. *Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10)*, IEEE Press, August 2001.

[77] **The Emerging Grid**. I. Foster, C. Kesselman. *Computational Aerosciences in the 21st Century*, 29-46, Kluwer Academic, 2000.

[78] **Grid Computing**. I. Foster. *Advance*, 51-56, 2000.

[79] **A Quality of Service Architecture that Combines Resource Reservation and Application Adaptation**. I. Foster, A. Roy, V. Sander. *8th International Workshop on Quality of Service*, 2000.

[80] **A National-Scale Authentication Infrastructure**. R. Butler, D. Engert, I. Foster, C. Kesselman, S. Tuecke, J. Volmer, V. Welch. *IEEE Computer*, 33(12):60-66, 2000.

[81] **Scheduling with Advanced Reservations**. W. Smith, I. Foster, V. Taylor. *Proceedings of the IPDPS Conference*, May 2000.

[82] **Exploiting Hierarchy in Parallel Computer Networks to Optimize Collective Operation Performance**. N. Karonis, B. de Supinski, I. Foster, W. Gropp, E. Lusk, J. Bresnahan. *Proceedings of the 14th International Parallel Distributed Processing Symposium (IPDPS '00)*, pp 377-84, Cancun, Mexico, May 2000.

[83] **Computational Grids.** I. Foster, C. Kesselman. *Chapter 2 of "The Grid: Blueprint for a New Computing Infrastructure"*, Morgan-Kaufman, 1999.

[84] **The Beta Grid: A National Infrastructure for Computer Systems Research**. I. Foster. *Proc. 1999 Extreme Linux Workshop*, also published in *Proc. NetStore Conference,* 1999.

[85] **Communication Services for Advanced Network Applications**. J. Bresnahan, I. Foster, J. Insley, S. Tuecke, B. Toonen. *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications 1999*, Las Vegas, Nevada, June 28-July1, 1999. Volume IV, pp1861-1867.

[86] **A Network Performance Tool for Grid Computations**. C. Lee, R. Wolski, I. Foster, C. Kesselman, J. Stepanek. *Supercomputing '99*, 1999.

[87] **A Fault Detection Service for Wide Area Distributed Computations**. P. Stelling, C. DeMatteis, I. Foster, C. Kesselman, C. Lee, G. von Laszewski. *Cluster Computing*, 2:117-128, 1999.

[88] **Resource Co-Allocation in Computational Grids**. K. Czajkowski, I. Foster, and C. Kesselman. *Proceedings of the Eighth IEEE International Symposium on High Performance Distributed Computing (HPDC-8)*, pp. 219-228, 1999.

[89] **A Distributed Resource Management Architecture that Supports Advance Reservations and Co-Allocation**. I. Foster, C. Kesselman, C. Lee, R. Lindell, K. Nahrstedt, A. Roy. *Intl Workshop on Quality of Service*, 1999.

[90] **The Globus Project: A Status Report**. I. Foster, C. Kesselman. *Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop*, pp. 4-18, 1998.

[91] **Managing Security in High-Performance Distributed Computing**. I. Foster, N. T. Karonis, C. Kesselman, S. Tuecke. *Cluster Computing*, 1(1):95-107, 1998.

[92] **A Security Architecture for Computational Grids**. I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. *Proc. 5th ACM Conference on Computer and Communications Security Conference*, pp. 83-92, 1998.

[93] **A Resource Management Architecture for Metacomputing Systems**. K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, S. Tuecke. *Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing*, pg. 62-82, 1998.

[94] **The Quality of Service Component for the Globus Metacomputing System**. C. Lee, C. Kesselman, J. Stepanek, R. Lindell, S. Hwang, B. Scott Michel, J. Bannister, I. Foster, A. Roy. *Proc. IWQoS '98*, pp. 140-142, 1998.

[95] **Application Experiences with the Globus Toolkit**. S. Brunett, K. Czajkowski, S. Fitzgerald, I. Foster, A. Johnson, C. Kesselman, J. Leigh, S. Tuecke. *Proceedings of 7th IEEE Symp. on High Performance Distributed Computing*, July 1998.

[96] **A Secure Communications Infrastructure for High-Performance Distributed Computing**. I. Foster, N. Karonis, C. Kesselman, G. Koenig, S. Tuecke. *6th IEEE Symp. on High-Performance Distributed Computing,* pp. 125-136, 1997.

[97] **Globus: A Metacomputing Infrastructure Toolkit**. I. Foster, C. Kesselman. *Intl J. Supercomputer Applications*, 11(2):115-128, 1997.

[98] **Managing Multiple Communication Methods in High-Performance Networked Computing Systems**. I. Foster, J. Geisler, C. Kesselman, S. Tuecke. *J. Parallel and Distributed Computing*, 40:35-48, 1997.

[99] **A Directory Service for Configuring High-Performance Distributed Computations**. S. Fitzgerald, I. Foster, C. Kesselman, G. von Laszewski, W. Smith, S. Tuecke. *Proc. 6th IEEE Symposium on High-Performance Distributed Computing*, pp. 365-375, 1997.

[100] **Software Infrastructure for the I-WAY High Performance Distributed Computing Experiment**. I. Foster, J. Geisler, W. Nickless, W. Smith, S. Tuecke. *Proc. 5th IEEE Symposium on High Performance Distributed Computing*, pp. 562-571, 1997.

[101] **Enabling Technologies for Web-Based Ubiquitous Supercomputing**. I. Foster, S. Tuecke. *Proc. 5th IEEE Symp. on High Performance Distributed Computing*, pp. 112-119, 1996.

[102] **The Nexus Task-Parallel Runtime System**. I. Foster, C. Kesselman, S. Tuecke. *Proc. 1st Int'l Workshop on Parallel Processing*, pp. 457-462, 1994.

[Supercomputing]

TOP500 Site: www.top500.org

[Cluster computing]

HispaCluster: www.hispacluster.org

IEE TFCC (*reports*): www.ieetfcc.org

Sun BluePrints: www.sun.com/blueprints

Mosix de la Universidad de Israel: www.mosix.cs.huji.ac.il

Beowulf Project en CESDIS : www.beowulf.org

Avalon: cnls.lanl.gov/Internal/Computing/Avalon

Sandia Labs Internacional Plant: www.cs.sandia.gov

Coral en ICASE: www.icase.edu

Babieca en el CAB: dalbe.inta.es/~LCASAT/trab/o_babieca.htm

[Intranet computing]

Sun Grid Engine de Sun Microsystems: www.sun.com/gridware

LSF de Platform Computing: www.platform.com

Condor de la Universidad de Wisconsin: www.cs.wisc.edu/condor

Message Passing Interface, MPI: www-unix.mcs.anl.gov/mpi

InnerGrid de Grid Systems: www.gridsystems.com

appLES de la Universidad de California: apples.ucsd.edu

Nimrod de la Universidad de Monash: www.csse.monash.edu.au/~rajkumar

Entropía: www.entropia.com

SETI: setiathome.ssl.berkeley.edu

[Grid computing]

Globus: www.globus.org

EDG/Datagrid: www.eu-datagrid.org

Geant : www.dante.net/geant

Damien : www.hlrs.de/organization/pds/projects/damien

EUROGRID : www.eurogrid.org

CrossGrid : www.eu-crossgrid.org

Global Grid Forum: www.globalgridforum.org

Open Grid Services Architecture: www.globus.org/ogsa

e-Science : www.research-councils.ac.uk/escience

Grid-based systems for Complex Problems Solving: www.cordis.lu/ist/grids

LHC Computing Grid: lcg.web.cern.ch/LCG

EGEE: www.eu-egee.org

Anexo I

Atributos para la designación de recursos

La información de los recursos Grid se obtiene a través de los atributos definidos dentro del esquema GLUE (*Grid Laboratory Uniform Environment*), utilizando el lenguaje de especificación UML (*Unified Modeling Language*).

1. Recursos computacionales (CE)

En esta sección se especifican los atributos asociados al recurso computacional (CE) y al conjunto de nodos (WNs) que lo forman.

- Base class for the CE information (objectclass GlueCETop)
  - No attributes
- Computing Element (objectclass GlueCE)
  - Unique identifier for the CE
    `GlueCEUniqueID`
  - Human-readable name of the service
    `GlueCEName`
- CE Policy (objectclass GlueCEPolicy)
  - Maximum wall clock time available to jobs submitted to the CE, in minutes
    `GlueCEPolicyMaxWallClockTime`
  - Maximum CPU time available to jobs submitted to the CE, in minutes
    `GlueCEPolicyMaxCPUTime`
  - Maximum allowed total number of jobs in the queue
    `GlueCEPolicyMaxTotalJobs`
  - Maximum allowed number of running jobs in the queue
    `GlueCEPolicyMaxRunningJobs`
  - Information about the service priority
    `GlueCEPolicyPriority`
  - Maximum number of single-processor jobs that can be running at a given time
    `GlueCEPolicyAssignedJobSlots`

- General Info for the queue associated to the CE (objectclass GlueCEInfo)
  - Name of the local batch system
    `GlueCEInfoLRMSType`
  - Version of the local batch system
    `GlueCEInfoLRMSVersion`
  - Version of GRAM
    `GlueCEInfoGRAMVersion`
  - Fully qualified name of the host where the gatekeeper runs
    `GlueCEInfoHostName`
  - Port number for the gatekeeper
    `GlueCEInfoGateKeeperPort`
  - Number of CPUs in the cluster associated to the CE
    `GlueCEInfoTotalCPUs`
  - Contact string for the service
    `GlueCEInfoContactString`
  - Job manager used by the gatekeeper
    `GlueCEInfoJobManager`
  - Path of the directory for application installation
    `GlueCEInfoApplicationDir`
  - Path a shared the directory for application data
    `GlueCEInfoDataDir`
  - Unique identifier of the default SE
    `GlueCEInfoDefaultSE`

- Network adapter (objectclass GlueHostNetworkAdapter)
  - Name of the network card
    `GlueHostNetworkAdapterName`
  - IP address of the network card
    `GlueHostNetworkAdapterIPAddress`
  - MTU size for the LAN to which the network card is attached
    `GlueHostNetworkAdapterMTU`
  - Permission for outbound connectivity
    `GlueHostNetworkAdapterOutboundIP`
  - Permission for inbound connectivity
    `GlueHostNetworkAdapterInboundIP`

- CE State (objectclass GlueCEState)
  - Queue status: queuing (jobs are accepted but not run), production (jobs are accepted and run), closed (jobs are neither accepted nor run), draining (jobs are not accepted but those in the queue are run)
    `GlueCEStateStatus`
  - Total number of jobs (running + waiting)
    `GlueCEStateTotalJobs`
  - Number of running jobs
    `GlueCEStateRunningJobs`
  - Number of jobs not running
    `GlueCEStateWaitingJobs`
  - Worst possible time between the submission of a job and the start of its execution, in seconds
    `GlueCEStateWorstResponseTime`
  - Estimated time between the submission of a job and the start of its execution, in seconds
    `GlueCEStateEstimatedResponseTime`
  - Number of CPUs available to the scheduler
    `GlueCEStateFreeCPUs`
  - Number of jobs that could start, given the current number of jobs submitted
    `GlueCEStateFreeJobSlots`

- Job (objectclass GlueCEJob)
  - Local user name of the job's owner
    `GlueCEJobLocalOwner`
  - GSI subject of the real job's owner
    `GlueCEJobGlobalOwner`
  - Local job identifier
    `GlueCEJobLocalID`
  - Global job identifier
    `GlueCEJobGlobalId`
  - Job Status: Submitted, Waiting, Ready, Scheduled, Running, Done, Aborted, Cleared, Checkpointed
    `GlueCEJobGlueCEJobStatus`
  - Any scheduler specific information
    `GlueCEJobSchedulerSpecific`

- Access control (objectclass GlueCEAccessControlBase)
  - A rule defining any access restrictions to the CE. Current semantic: VO = a VO name, DENY = an X.509 user subject

    `GlueCEAccessControlBaseRule`

- Host (objectclass GlueHost)
  - Unique identifier for the host

    `GlueHostUniqueId`
  - Human-readable name of the host

    `GlueHostName`

- VO View (objectclass GlueVOView)
  - Local ID for this VO view

    `GlueVOViewLocalID`

- Processor (objectclass GlueHostProcessor)
  - Name of the CPU vendor

    `GlueHostProcessorVendor`
  - Name of the CPU model

    `GlueHostProcessorModel`
  - Version of the CPU

    `GlueHostProcessorVersion`
  - Clock speed of the CPU

    `GlueHostProcessorClockSpeed`
  - Name of the CPU instruction set architecture

    `GlueHostProcessorInstructionSet`
  - Other description for the CPU

    `GlueHostProcessorOtherProcessorDescription`
  - Size of the unified L1 cache

    `GlueHostProcessorCacheL1`
  - Size of the instruction L1 cache

    `GlueHostProcessorCacheL1I`
  - Size of the data L1 cache

    `GlueHostProcessorCacheL1D`
  - Size of the unified L2 cache

    `GlueHostProcessorCacheL2`

- Cluster (objectclass GlueCluster)
  - Unique identifier for the cluster
    `GlueClusterUniqueID`
  - Human-readable name of the cluster
    `GlueClusterName`

- Subcluster (objectclass GlueSubCluster)
  - Unique identifier for the subcluster
    `GlueSubClusterUniqueID`
  - Human-readable name of the subcluster
    `GlueSubClusterName`
  - Path of temporary directory shared among worker nodes
    `GlueSubClusterTmpDir`
  - Path of temporary directory local to the worker nodes
    `GlueSubClusterWNTmpDir`
  - Total number of real CPUs in the subcluster
    `GlueSubClusterPhysicalCPUs`
  - Total number of logical CPUs (e.g. hyperthreading)
    `GlueSubClusterLogicalCPUs`

- Main memory (objectclass GlueHostMainMemory)
  - Physical RAM
    `GlueHostMainMemoryRAMSize`
  - Unallocated RAM
    `GlueHostMainMemoryRAMAvailable`
  - Size of the configured virtual memory
    `GlueHostMainMemoryVirtualSize`
  - Available virtual memory
    `GlueHostMainMemoryVirtualAvailable`

- Processor load (objectclass GlueHostProcessorLoad)
  - Average processor availability for a single node
    `GlueHostProcessorLoadLast1Min`
  - Average processor availability for a single node
    `GlueHostProcessorLoadLast5Min`
  - Average processor availability for a single node
    `GlueHostProcessorLoadLast15Min`

- Architecture (objectclass GlueHostArchitecture)
  - Platform description
    `GlueHostArchitecturePlatformType`
  - Number of CPUs in a SMP node
    `GlueHostArchitectureSMPSize`
- SMP load (objectclass GlueHostSMPLoad)
  - Average processor availability for a single node
    `GlueHostSMPLoadLast1Min`
  - Average processor availability for a single node
    `GlueHostSMPLoadLast5Min`
  - Average processor availability for a single node
    `GlueHostSMPLoadLast15Min`
- Operating system (objectclass GlueHostOperatingSystem)
  - OS name
    `GlueHostOperatingSystemOSName`
  - OS release
    `GlueHostOperatingSystemOSRelease`
  - OS or kernel version
    `GlueHostOperatingSystemOSVersion`
- Local file system (objectclass GlueHostLocalFileSystem)
  - Path name or other information defining the root of the file system
    `GlueHostLocalFileSystemRoot`
  - Size of the file system in bytes
    `GlueHostLocalFileSystemSize`
  - Amount of free space in bytes
    `GlueHostLocalFileSystemAvailableSpace`
  - True if the file system is read-only
    `GlueHostLocalFileSystemReadOnly`
  - File system type
    `GlueHostLocalFileSystemType`
  - The name for the file system
    `GlueHostLocalFileSystemName`
  - Host unique identifier of clients allowed to remotely access this file system
    `GlueHostLocalFileSystemClient`

- Remote file system (objectclass GlueHostRemoteFileSystem)
  - Path name or other information defining the root of the file system
    `GlueHostLRemoteFileSystemRoot`
  - Size of the file system in bytes
    `GlueHostRemoteFileSystemSize`
  - Amount of free space in bytes
    `GlueHostRemoteFileSystemAvailableSpace`
  - True if the file system is read-only
    `GlueHostRemoteFileSystemReadOnly`
  - File system type
    `GlueHostRemoteFileSystemType`
  - The name for the file system
    `GlueHostRemoteFileSystemName`
  - Host unique id of the server which provides access to the file system
    `GlueHostRemoteFileSystemServer`
- Storage device (objectclass GlueHostStorageDevice)
  - Name of the storage device
    `GlueHostStorageDeviceName`
  - Storage device type
    `GlueHostStorageDeviceType`
  - Maximum transfer rate for the device
    `GlueHostStorageDeviceTransferRate`
  - Size of the device
    `GlueHostStorageDeviceSize`
  - Amount of free space
    `GlueHostStorageDeviceAvailableSpace`
- Location (objectclass GlueLocation)
  - Local ID for the location
    `GlueLocationLocalID`
  - Location name
    `GlueLocationName`
  - Location path
    `GlueLocationPath`
  - Version
    `GlueLocationVersion`

- File (objectclass GlueHostFile)
  - Name for the file
    `GlueHostFileName`
  - File size in bytes
    `GlueHostFileSize`
  - File creation date and time
    `GlueHostFileCreationDate`
  - Date and time of the last modification of the file
    `GlueHostFileLastModified`
  - Date and time of the last access to the file
    `GlueHostFileLastAccessed`
  - Time taken to access the file, in seconds
    `GlueHostFileLatency`
  - Time for which the file will stay on the storage device
    `GlueHostFileLifeTime`
  - Name of the owner of the file
    `GlueHostFileOwner`

- Benchmark (objectclass GlueHostBenchmark)
  - SpecInt2000 benchmark
    `GlueHostBenchmarkSI00`
  - SpecFloat2000 benchmark
    `GlueHostBenchmarkSF00`

- Application software (objectclass GlueHostApplicationSoftware)
  - List of software installed on this host
    `GlueHostApplicationSoftwareRunTimeEnvironment`

2. Recursos de almacenamiento (SE)

Los atributos que se especifican a continuación, proporcionan la información relativa al recurso de almacenamiento (SE). Los atributos correspondientes a los elementos asociados al SE, (StorageService, StorageLibrary y StorageSpace), se definen dentro de sus respectivas clases (GlueSE, GlueSL y GlueSA).

- Base Class for the storage service (objectclass GlueSETop)
  - No attributes
- Base Class for the storage library (objectclass GlueSLTop)
  - No attributes
- Base Class for the storage space (objectclass GlueSATop)
  - No attributes
- Storage Service (objectclass GlueSE)
  - Unique identifier of the storage service (URI)
    `GlueSEUniqueId`
  - Human-readable name for the service
    `GlueSEName`
  - Port number that the service listens
    `GlueSEPort`
  - Unique identifier of the storage library hosting the service
    `GlueSEHostingSL`
  - The total size of the storage space managed by the service
    `GlueSESizeTotal`
  - The size of the storage capacity that is free for new areas for any VO/user
    `GlueSESizeFree`
  - Underlying architectural system category
    `GlueSEArchitecture`
- State (objectclass GlueSAState)
  - Total space available in the storage space (in kilobytes)
    `GlueSAStateAvailableSpace`
  - Used space in the storage space (in kilobytes)
    `GlueSAStateUsedSpace`

- Storage Service Access Protocol (objectclass GlueSEAccessProtocol)
  - Protocol type to access or transfer files
    `GlueSEAccessProtocolType`
  - Port number for the protocol
    `GlueSEAccessProtocolPort`
  - Protocol version
    `GlueSEAccessProtocolVersion`
  - Security features supported by the protocol
    `GlueSEAccessProtocolSupportedSecurity`
  - Time to access a file using this protocol
    `GlueSEAccessProtocolAccessTime`
  - Local identifier
    `GlueSEAccessProtocolLocalID`
  - Network endpoint for this protocol
    `GlueSEAccessProtocolEndpoint`
  - Function supported by this control protocol
    `GlueSEAccessProtocolCapability`
- Protocol details (objectclass GlueSEControlProtocol)
  - Protocol type (e.g. srmv1)
    `GlueSEControlProtocolType`
  - Protocol version
    `GlueSEControlProtocolVersion`
  - Local identifier
    `GlueSEControlProtocolLocalID`
  - Network endpoint for this protocol
    `GlueSEControlProtocolLocalID`
  - Function supported by this control protocol
    `GlueSEControlProtocolCapability`
- Storage Library (objectclass GlueSL)
  - Human-readable name of the storage library
    `GlueSLName`
  - Unique identifier of the machine providing the storage service
    `GlueSLUniqueID`
  - Unique identifier for the provided storage service
    `GlueSLService`

- Storage Service State (objectclass GlueSEState)
  - System load (for example, number of files in the queue)

    `GlueSEStateCurrentIOLoad`

- File Information (objectclass GlueSLFile)
  - File name

    `GlueSLFileName`
  - File size

    `GlueSLFileSize`
  - File creation date and time

    `GlueSLFileCreationDate`
  - Date and time of the last modification of the file

    `GlueSLFileLastModified`
  - Date and time of the last access to the file

    `GlueSLFileLastAccessed`
  - Time needed to access the file

    `GlueSLFileLatency`
  - File lifetime

    `GlueSLFileLifeTime`
  - File path

    `GlueSLFilePath`

- Local File system (objectclass GlueSLLocalFileSystem)
  - Path name (or other information) defining the root of the file system

    `GlueSLLocalFileSystemRoot`
  - Name of the file system

    `GlueSLLocalFileSystemName`
  - File system type (e.g. NFS, AFS, etc.)

    `GlueSLLocalFileSystemType`
  - True is the file system is read-only

    `GlueSLLocalFileSystemReadOnly`
  - Total space assigned to this file system

    `GlueSLLocalFileSystemSize`
  - Total free space in this file system

    `GlueSLLocalFileSystemAvailableSpace`
  - Unique identifiers of clients allowed to access the file system remotely

    `GlueSLLocalFileSystemClient`

- Remote File system (objectclass GlueSLRemoteFileSystem)
  - Path name (or other information) defining the root of the file system
    `GlueSLRemoteFileSystemRoot`
  - Total space assigned to this file system
    `GlueSLRemoteFileSystemSize`
  - Total free space in this file system
    `GlueSLRemoteFileSystemAvailableSpace`
  - True is the file system is read-only
    `GlueSLRemoteFileSystemReadOnly`
  - File system type (e.g. NFS, AFS, etc.)
    `GlueSLRemoteFileSystemType`
  - Name of the file system
    `GlueSLRemoteFileSystemName`
  - Unique identifier of the server exporting this file system
    `GlueSLRemoteFileSystemServer`
- Directory Information (objectclass GlueSLDirectory)
  - Directory name
    `GlueSLDirectoryName`
  - Directory size
    `GlueSLDirectorySize`
  - Directory creation date and time
    `GlueSLDirectoryCreationDate`
  - Date and time of the last modification of the directory
    `GlueSLDirectoryLastModified`
  - Date and time of the last access to the directory
    `GlueSLDirectoryLastAccessed`
  - Time needed to access the directory
    `GlueSLDirectoryLatency`
  - Directory lifetime
    `GlueSLDirectoryLifeTime`
  - Directory path
    `GlueSLDirectoryPath`
- Architecture (objectclass GlueSLArchitecture)
  - Type of storage hardware (i.e. disk, RAID, tape library)
    `GlueSLArchitectureType`

- Performance (objectclass GlueSLPerformance)
  - Maximum bandwidth between the service and the network

    `GlueSLPerformanceMaxIOCapacity`

- Storage Space (objectclass GlueSA)
  - Pathname of the directory containing the files of the storage space

    `GlueSARoot`
  - Local identifier

    `GlueSALocalID`
  - Root path of the area

    `GlueSAPath`
  - Lifetime for the storage area (permanent, durable, volatile, other)

    `GlueSAType`
  - Unique identifier

    `GlueSAUniqueID`

- Policy (objectclass GlueSAPolicy)
  - Maximum file size

    `GlueSAPolicyMaxFileSize`
  - Minimum file size

    `GlueSAPolicyMinFileSize`
  - Maximum allowed amount of data that a single job can store

    `GlueSAPolicyMaxData`
  - Maximum allowed number of files that a single job can store

    `GlueSAPolicyMaxNumFiles`
  - Maximum allowed lifetime for non-permanent files

    `GlueSAPolicyMaxPinDuration`
  - Total available space

    `GlueSAPolicyQuota`
  - Lifetime policy for the contained files

    `GlueSAPolicyFileLifeTime`

- Access Control Base (objectclass GlueSAAccessControlBase)
  - List of the access control rules

    `GlueSAAccessControlBase Rule`

3. Asociación de recursos (CE-SE)

Los atributos que se especifican a continuación, proporcionan la información relativa a la asociación entre un recurso computacional (CE) y uno o más recursos de almacenamiento (SE).

- Associations between an CE and one or more SEs (objectclass GlueCESEBindGroup)
  - Unique ID for the CE
    `GlueCESEBindGroupCEUniqueID`
  - Unique ID for the SE
    `GlueCESEBindGroupSEUniqueID`

- Associations between an SE and a CE (objectclass GlueCESEBind)
  - Unique ID for the CE
    `GlueCESEBindCEUniqueID`
  - Access point in the cluster from which CE can access a local SE
    `GlueCESEBindCEAccesspoint`
  - Unique ID for the SE
    `GlueCESEBindSEUniqueID`
  - Information about the name of the mount directory on the worker nodes of the CE and the exported directory from the SE
    `GlueCESEBindMountInfo`
  - It expresses a preference when multiple SEs are bound to a CE
    `GlueCESEBindWeight`

Anexo II

Referencia de comandos Globus

[grid-proxy-init]

Syntax

```
grid-proxy-init-bin [-help][-pwstdin][-limited][-valid H:M] ...
```

Options

```
-help, -usage             Displays usage
-version                  Displays version
-debug                    Enables extra debug output
-q                        Quiet mode, minimal output
-verify                   Verifies certificate to make proxy for
-pwstdin                  Allows passphrase from stdin
-limited                  Creates a limited globus proxy
-independent              Creates a independent globus proxy
-old                      Creates a legacy globus proxy
-valid <h:m>              Proxy is valid for h hours and m
                          minutes (default:12:00)
-hours <hours>            Deprecated support of hours option
-bits  <bits>             Number of bits in key {512|1024|2048|4096}
-policy <policyfile>      File containing policy to store in the
                          ProxyCertInfo extension
-pl <oid>,                OID string for the policy language
-policy-language <oid>    used in the policy file
-path-length <l>          Allow a chain of at most l proxies to be
                          generated from this one
-cert     <certfile>      Non-standard location of user certificate
-key      <keyfile>       Non-standard location of user key
-certdir  <certdir>       Non-standard location of trusted cert dir
-out      <proxyfile>     Non-standard location of new proxy cert
```

[grid-proxy-info]

Syntax

```
grid-proxy-info [-help][-f proxyfile][-subject][...][-e [-h H][-b B]]
```

Options

```
-help, -usage             Displays usage
-version                  Displays version
-debug                    Displays debugging output
-file <proxyfile>  (-f)   Non-standard location of proxy
[printoptions]            Prints information about proxy
-exists [options]  (-e)   Returns 0 if valid proxy exists, 1 otherwise

[printoptions]
    -subject       (-s)   Distinguished name (DN) of subject
    -issuer        (-i)   DN of issuer (certificate signer)
    -identity             DN of the identity represented by the proxy
    -type                 Type of proxy (full or limited)
    -timeleft             Time (in seconds) until proxy expires
    -strength             Key size (in bits)
    -all                  All above options in a human readable format
    -text                 All of the certificate
    -path                 Pathname of proxy file

[options to -exists]      (if none are given, H = B = 0 are assumed)
    -valid H:M     (-v)   time requirement for proxy to be valid
    -hours H       (-h)   time requirement for proxy to be valid
                          (deprecated, use -valid instead)
    -bits  B       (-b)   strength requirement for proxy to be valid
```

[grid-proxy-destroy]

Syntax

```
grid-proxy-destroy [-help][-dryrun][-default][-all][--] [file1...]
```

Options

```
-help, -usage             Displays usage
-version                  Displays version
-debug                    Display debugging information
-dryrun                   Prints what files would have been destroyed
-default                  Destroys file at default proxy location
-all                      Destroys any user (default) and delegated
                          proxies that are found
--                        End processing of options
file1 file2 ...           Destroys files listed
```

[lcg-info]

Synopsis

```
lcg-info --list-ce [--bdii bdii] [--vo vo] [--sed] [--quiet]
        [--query query] [--attrs list]

lcg-info --list-se [--bdii bdii] [--vo vo] [--sed] [--quiet]
        [--query query] [--attrs list]

lcg-info --list-attrs

lcg-info --help
```

Options

`--help` Prints the manual page and exits.

`--list-attrs` Prints a list of the attributes that can be queried.

`--list-ce` Lists the CEs which satisfy a query, or all the CEs if no query is given.

`--list-se` Lists the SEs which satisfy a query, or all the SEs if no query is given.

`--query` Restricts the output to the CEs (SEs) which satisfy the given query.

`--bdii` Allows to specify a BDII in the form <hostname>:<port>. If not given, the value of the environmental variable `LCG_GFAL_INFOSYS` is used. If that is not defined, the command returns an error.

`--sed` Prints the output in a "sed-friendly" format: "%" separate the CE (SE) identifier and the printed attributes, "&" separate the values of multi-valued attributes.

`--quiet` Suppresses warning messages.

`--attrs` Specifies the attributes whose values should be printed.

`--vo` Restricts the output to CEs or SEs where the given VO is authorized. Mandatory when VO-dependent attributes are queried upon.

[edg-job-status]

Syntax

```
edg-job-status  [options]  <job Id(s)>
```

Options

```
--help
--version
--all
--input, -i          <file_path>
--verbosity, -v      <verbosity_value>
--from               [MM:DD:]hh:mm[:[CC]YY]
--to                 [MM:DD:]hh:mm[:[CC]YY]
--config, -c         <file_path>
--config-vo          <file_path>
--vo                 <vo_name>
--output, -o         <file_path>
--noint
--debug
--logfile            <file_path>
```

Options

```
--help
      displays command usage.

--version
      displays UI version.

--all
      displays status information about all job owned by the user
      submitting the command. This option can't be used either if one or
      more edg_jobIds have been specified or if the --input option has been
      specified. All LBs listed in the vo-specific UI configuration file
      EDG_WL_LOCATION/etc/<vo_name>/edg_wl_ui.conf are contacted to fulfil
      this request.

--input file_path
-i file_path
      displays bookkeeping info about dg_jobIds contained in the input
      file. When using this option the user is interrogated for choosing
      among all, one or a subset of the listed job identifiers. This option
      can't be used either if one or more edg_jobIds have been specified or
      if the --all option has been specified. When this option is used, the
      list of edg_jobIds contained in the file is displayed and the user is
      prompted for a choice. Single jobs can be selected specifying the
      numbers associated to the job identifiers separated by commas.
```

```
      E.g. 1,3,5 selects the first, the third and the fifth edg_jobId in
      the list. Ranges can also be selected specifying ends separated by a
      dash. E.g. 3-6 selects edg_jobIds in the list from third position
      (included) to sixth position (included). It is worth mentioning that
      it is possible to select at the same time ranges and single jobs.
      E.g. 1,3-5,8 selects the first job id in the list, the ids from the
      third to the fifth (ends included) and finally the eighth one.

--verbosity verb_level
-v verb_level
      sets the detail level of information about the job displayed to the
      user. Possible values for verb_level are 0,1 and 2.

--from [MM:DD:]hh:mm[:[CC]YY]
      makes the command query LB for jobs that have been submitted (more
      precisely     entered the "Submitted" status) after the specified
      date/time. If only hours and minutes are specified then the current
      day is taken into account. If the year is not specified then the
      current year is taken into account.

--to [MM:DD:]hh:mm[:[CC]YY]
      makes the command query LB for jobs that have been submitted (more
      precisely     entered the "Submitted" status) before the specified
      date/time. If only hours and minutes are specified then the current
      day is taken into account. If the year is not specified then the
      current year is taken into account.

--config file_path
-c file_path
      if the command is launched with this option, the configuration file
      pointed to by file_path is used instead of the standard configuration
      file.

--config-vo file_path
      if the command is launched with this option, the vo-specific
      configuration file pointed to by file_path is used instead of the
      standard vo-specific configuration file. This option is not allowed
      when one or more edg_jobIds specified as command arguments.

--vo vo_name
      this option allows the user to specify the Virtual Organisation
      she/he is currently working for. If the user proxy contains VOMS
      extensions then the VO specified through this option is overridden by
      the default VO contained in the proxy (i.e. this option is only
      useful when working with non-VOMS proxies). The following precedence
      rule is followed for determining the user's VO:

      -the default VO from the user proxy (if it contains VOMS extensions),
      -the VO specified through the --vo or --config-vo options,
```

```
       -the VO specified in the configuration file pointed by the
       EDG_WL_UI_CONFIG_VO  environment variable,

       -the default VO specified in the
       $EDG_WL_LOCATION/etc/edg_wl_ui_cmd_var.conf (DefaultVO field)
       configuration file.

       If none of the listed trials has success an error is returned and the
       submission is aborted. This option is not allowed when one or more
       edg_jobIds are specified as command arguments.

--output file_path
-o file_path
       writes the bookkeeping information in the file specified by file_path
       instead of the standard output. file_path can be either a simple name
       or an absolute path (on the submitting machine). In the former case
       the file file_path is created in the current working directory.

--noint
       if this option is specified every interactive question to the user is
       skipped and the operation is continued.
       All warning messages and errors (if any) are written to the file
       edg-job-status_<UID>_<PID>_<timestamp>.log under the /tmp directory.
       Location of log file is configurable.

--debug
       when this option is specified, information about the API functions
       called inside the command are displayed on the standard output and
       are written to the file edg-job-status_<UID>_<PID>_<timestamp>.log
       under the /tmp directory too. Location of log file is configurable.

--logfile file_path
       when this option is specified, the command log file is relocated to
       the location pointed by file_path

edg_jobId
       job identifier returned by edg-job-submit. Job identifiers must
       always be provided as last arguments of the command.
```

[edg-job-get-output]

Syntax

```
edg-job-get-output  [options]  <job Id(s)>
```

Options

```
--help
--version
--input, -i     <file_path>
--dir           <directory_path>
--config, -c    <file_path>
--noint
--debug
--logfile       <file_path>
```

Options

```
--help
      displays command usage.

--version
      displays UI version.

--input file_path
-i file_path
      this option makes the command return the OutputSandbox files for each
      edg_jobId contained in the file_path. This option can't be used if
      one (or more) edg_jobIds have been already specified. The format of
      the input file must be as follows: one edg_jobId for each line and
      comment lines must begin with a "#" or a "*" character. When this
      option is used, the list of edg_jobIds contained in the file is
      displayed and the user is prompted for a choice. Single jobs can be
      selected specifying the numbers associated to the job identifiers
      separated by commas. E.g. 1,3,5 selects the first, the third and the
      fifth edg_jobId in the list. Ranges can also be selected specifying
      ends separated by a dash. E.g. 3-6 selects edg_jobIds in the list
      from third position (included) to sixth position (included). It is
      worth mentioning that it is possible to select at the same time
      ranges and single jobs. E.g. 1,3-5,8 selects the first job id in the
      list, the ids from the third to the fifth (ends included) and finally
      the eighth one.

--dir directory_path
      retrieved files (previously listed by the user through the
      OutputSandbox attribute of the job description file) are stored in
      the location indicated by directory_path/<edg_jobId unique string>.
```

```
--config file_path
-c file_path
      if the command is launched with this option, the configuration file
      pointed to by file_path is used instead of the standard configuration
      file.

--noint
      if this option is specified every interactive question to the user is
      skipped and the operation is continued. All warning messages and
      errors (if occurred) are written to the file
      edg-job-get-output_<UID>_<PID>_<timestamp>.log under the /tmp
      directory. Location of log file is configurable.

--debug
      when this option is specified, information about parameters used for
      the API functions calls inside the command are displayed on the
      standard output and are written to
      edg-get_job_output_<UID>_<PID>_<timestamp>.log file under the /tmp
      directory too. Location of log file is configurable.

--logfile file_path
      when this option is specified, the command log file is relocated to
      the location pointed by file_path

edg_jobId
      job identifier returned by edg-job-submit. If a list of one or more
      job identifiers is specified, edg_jobIds have to be separated by a
      blank. Job identifiers must be last argument of the command.
```

[edg-job-get-logging-info]

Syntax

```
edg-job-get-logging-info  [options]  <job Id(s)>
```

Options

```
--help
--version
--input, -i     <file_path>
--verbosity, -v <verbosity_value>
--config, -c    <file_path>
--output, -o    <file_path>
--noint
--debug
--logfile       <file_path>
```

Options

```
--help
      displays command usage.

--version
      displays UI version.

--input file_path
-i file_path
      retrieves logging info for all edg_jobIds contained in the file_path.
      This option can't be used  if one or more edg_jobIds have been
      specified. When this option is used, the list of edg_jobIds contained
      in the file is displayed and the user is prompted for a choice.
      Single jobs can be selected specifying the numbers associated to the
      job identifiers separated by commas. E.g. 1,3,5 selects the first,
      the third and the fifth edg_jobId in the list. Ranges can also be
      selected specifying ends separated by a dash. E.g. 3-6 selects
      edg_jobIds in the list from third position (included) to sixth
      position (included).

--verbosity verb_level
--v verb_level
      sets the detail level of information about the job displayed to the
      user. Possible values for verb_level are 0,1 and 2.

--config file_path
-c file_path
      if the command is launched with this option, the configuration file
      pointed to by file_path is used instead of the standard configuration
      file.
```

```
--output file_path
-o file_path
      writes the logging information in the file specified by file_path
      instead of the standard output. file_path can be either a simple name
      or an absolute path (on the submitting machine). In the former case
      the file file_path is created in the current working directory.

--noint
      if this option is specified every interactive question to the user is
      skipped and the operation is continued. All warning messages and
      errors (if occurred) are written to the file
      edg-job-logging_<UID>_<PID>_<timestamp>.log under the /tmp
      directory. Location for log file is configurable.

--debug
      when this option is specified, information about the API functions
      called inside the command are displayed on the standard output and
      are written to the file edg-job-logging_<UID>_<PID>_<timestamp>.log
      under the /tmp directory too. Location for log file is configurable.

--logfile file_path
      when this option is specified, the command log file is relocated to
      the location pointed by file_path

edg_jobId
      job identifier returned by edg-job-submit. Job identifiers must
      always be provided as last arguments for this command.
```

[edg-job-submit]

Syntax

```
edg-job-submit  [options]  <jdl_file>
```

Options

```
--help
--version
--vo            <vo_name>
--input, -i     <file_path>
--resource, -r  <ce_id>
--config, -c    <file_path>
--config-vo     <file_path>
--output, -o    <file_path>
--chkpt         <file_path>
--nolisten
--nogui
--nomsg
--noint
--debug
--logfile       <file_path>
```

Options

```
--help
      displays command usage.

--version
      displays UI version.

--vo vo_name
      this option allows the user to specify the Virtual Organisation
      she/he is currently working for. If the user proxy contains VOMS
      extensions then the VO specified through this option is overridden by
      the default VO contained in the proxy (i.e. this option is only
      useful when working with non-VOMS proxies). The following precedence
      rule is followed for determining the user's VO: the default VO from
      the user proxy (if it contains VOMS extensions), the VO specified
      through the --vo or --config-vo options, the VO specified in the
      configuration file pointed by the EDG_WL_UI_CONFIG_VO environment
      variable, the VirtualOrganisation attribute in the JDL (if the user
      proxy contains VOMS extensions this value is overridden  as above),
      the default VO specified in the DefaultVO field configuration file
      $EDG_WL_LOCATION/etc/edg_wl_ui_cmd_var.conf. If none of the listed
      trials has success an error is returned and the submission is
      aborted.
```

```
--input file_path
-i file_path
      if this option is specified, the user will be asked to choose a CEId
      from a list of CEs contained in the file_path. Once a CEId has been
      selected the command behaves as explained for the --resource option.
      If this option is used together with the -noint one and the input
      file contains more than one CEId, then the first CEId in the list is
      taken into account for submitting the job.

--resource ce_id
-r ce_id
      if the command is launched with this option, the job-ad sent to the
      NS contains a line of the type SubmitTo = ce_id  and the job is
      submitted by the WMS to the resource identified by ce_id without
      going through the match-making process. Accepted format for the CEId
      is: <full hostname>:<port number>/jobmanager-<service>-<queue name>
      where <service> could be for example lsf, pbs, bqs, condor but can
      also be a different string as it is freely set by the site
      administrator when the queue is set-up. Note that when this option is
      used, the ".BrokerInfo" file is not generated.

--config file_path
-c file_path
      if the command is launched with this option, the configuration file
      pointed to by file_path is used instead of the standard configuration
      file.

--config-vo file_path
      if the command is launched with this option, the vo-specific
      configuration file pointed to by file_path is used instead of the
      standard vo-specific configuration file.

--output file_path
-o file_path
      writes the generated edg_jobId assigned to the submitted job in the
      file specified by out_file. out_file can be either a simple name or
      an absolute path (on the submitting machine). In the former case the
      file out_file is created in the current working directory.

--chkpt file_path
      This option can be used only for checkpointable jobs. The state
      specified as input is a checkpoint state generated by a previously
      submitted job.  This option makes the submitted job start running
      from the checkpoint state given in input and not from the very
      beginning. The initial checkpoint states to be used with this option
      can be retrieved by means of the edg-job-get-chkpt command.
```

--nolisten
This option can be used only for interactive jobs. It makes the command forward the job standard streams coming from the WN to named pipes on the UI machine whose names are returned to the user together with the OS id of the listener process. This allows the user to interact with the job through her/his own tools. It is important to note that when this option is specified, the UI has no more control over the launched listener process that has hence to be killed by the user (through the returned process id) once the job is finished.

--nogui
This option can be used only for interactive jobs. As the command for such jobs opens a X window, the user should make sure a X server is running on the local machine and if she/he is connected to the UI node from a remote machine (e.g. with ssh) enable secure X11 tunneling. If this is not possible, the user can specify the --nogui option that makes the command provide a simple standard non-graphical interaction with the running job.

--nomsg
this option makes the command print on the standard output only the edg_jobId generated for the job if submission was successful; the location of the log file containing massages and diagnostics is printed otherwise.

--noint
if this option is specified every interactive question to the user is skipped and the operation is continued. All warning messages and errors (if occurred) are written to the file edg-job-submit_<UID>_<PID_<timestamp>.log under the /tmp directory. Log file location is configurable.

--debug
when this option is specified, information about parameters used for the API functions calls inside the command are displayed on the standard output and are written to edg-job-submit_<UID>_<PID>_<timestamp>.log file under the /tmp directory too. Log file location is configurable.

--logfile file_path
when this option is specified, the command log file is relocated to the location pointed by file_path

jdl_file
this is the file containing the JDL describing the job to be submitted. It must be the last argument of the command.

[edg-job-list-match]

Syntax

```
edg-job-list-match  [options]  <jdl file>
```

Options

```
--help
--version
--verbose, -v
--rank
--config, -c    <file_path>
--config-vo     <file_path>
--vo            <vo_name>
--output, -o    <file_path>
--noint
--debug
--logfile       <file_path>
```

Options

```
--help
      displays command usage.

--version
      displays UI version.

--verbose
-v
      displays on the standard output the job class-ad that is sent to the
      Network Server generated from the job description file. This differs
      from the content of the job description file since the UI adds to it
      some attributes that cannot be directly inserted by the user (e.g.,
      defaults for Rank and Requirements if not provided,
      VirtualOrganisation etc).

--rank
      displays the "matching" CEIds and the associated ranking values.

--config file_path
-c file_path
      if the command is launched with this option, the configuration file
      pointed to by file_path is used instead of the standard configuration
      file.

--config-vo file_path
      if the command is launched with this option, the vo-specific
      configuration file pointed to by file_path is used instead of the
      standard vo-specific configuration file.
```

```
--vo vo_name
      this option allows the user to specify the Virtual Organisation
      she/he is currently working for. If the user proxy contains VOMS
      extensions then the VO specified through this option is overridden by
      the default VO contained in the proxy (i.e. this option is only
      useful when working with non-VOMS proxies).The following precedence
      rule is followed for determining the user's VO:

      -the default VO from the user proxy (if it contains VOMS extensions)
      -the VO specified through the --vo or --config-vo options
      -the VO specified in the configuration file pointed by the
      EDG_WL_UI_CONFIG_VO environment variable

      -the VirtualOrganisation attribute in the JDL (if the user proxy
      contains VOMS extensions this value is overridden as above)

      -the default VO specified in the DefaultVO field configuration file
      $EDG_WL_LOCATION/etc/edg_wl_ui_cmd_var.conf

      If none of the listed trials has success an error is returned and the
      submission is aborted.

--output file_path
-o file_path
      returns the CEIds list in the file specified by file_path. file_path
      can be either a simple name or an absolute path (on the submitting
      machine). In the former case the file file_path is created in the
      current working directory.

--noint
      if this option is specified every interactive question to the user is
      skipped and the operation is continued. All warning messages and
      errors (if any) are written to the file edg-job-list-match
      <UID>_<PID>_<timestamp>.log under the /tmp directory. Location of the
      log file is configurable.

--debug
      when this option is specified, information about the API functions
      called inside the command are displayed on the standard output and
      are written to the file edg-job-list-
      match_<UID>_<PID>_<timestamp>.log under the /tmp directory too.
      Location of the log file is configurable.

--logfile file_path
      when this option is specified, the command log file is relocated to
      the location pointed by file_path

jdl_file
      this is the file containing the classad describing the job to be
      submitted. It must be the last argument of the command.
```

[edg-job-cancel]

Syntax

```
edg-job-cancel  [options]  <job Id(s)>
```

Options

```
   --help
   --version
   --all
   --input, -i     <file_path>
   --config, -c    <file_path>
   --config-vo     <file_path>
   --vo            <vo_name>
   --output, -o    <file_path>
   --noint
   --debug
   --logfile       <file_path>
```

Options

```
--help
       displays command usage.

--version
       displays UI version.

--all
       cancels all job owned by the user submitting the command. This option
       can't be used either if one or more edg_jobIds have been specified
       explicitly or with the -input option.

--input file_path
-i file_path
       cancels edg_jobId contained in the file_path. This option can't be
       used neither if one or more edg_jobIds have been specified nor with
       the -all option. When this option is used, the list of edg_jobIds
       contained in the file is displayed and the user is prompted for a
       choice. Single jobs can be selected specifying the numbers associated
       to the job identifiers separated by commas. E.g. 1,3,5 selects the
       first, the third and the fifth edg_jobId in the list. Ranges can also
       be selected specifying ends separated by a dash. E.g. 3-6 selects
       edg_jobIds in the list from third position (included) to sixth
       position (included). It is worth mentioning that it is possible to
       select at the same time ranges and single jobs. E.g. 1,3-5,8 selects
       the first job id in the list, the ids from the third to the fifth
       (ends included) and finally the eighth one.
```

```
--config file_path
-c file_path
      if the command is launched with this option, the configuration file
      pointed to by file_path is used instead of the standard configuration
      file.

--config-vo file_path
      if the command is launched with this option, the vo-specific
      configuration file pointed to by file_path is used instead of the
      standard vo-specific configuration file. This option is allowed only
      when used together with the --all one.

--vo vo_name
      this option allows the user to specify the Virtual Organisation
      she/he is currently working for. If the user proxy contains VOMS
      extensions then the VO specified through this option is overridden by
      the default VO contained in the proxy (i.e. this option is only
      useful when working with non-VOMS proxies). The following precedence
      rule is followed for determining the user's VO:

      -the default VO from the user proxy (if it contains VOMS extensions)
      -the VO specified through the --vo or --config-vo options

      -the VO specified in the configuration file pointed by the
      EDG_WL_UI_CONFIG_VO environment variable

      -the default VO specified in the DefaultVO field configuration file
      $EDG_WL_LOCATION/etc/edg_wl_ui_cmd_var.conf

      If none of the listed trials has success an error is returned and the
      submission is aborted. This option is allowed only when used together
      with the --all one.

--output file_path
-o file_path
      writes the cancel results in the file specified by file_path instead
      of the standard output. file_path can be either a simple name or an
      absolute path (on the submitting machine). In the former case the
      file file_path is created in the current working directory.

--noint
      if this option is specified every interactive question to the user is
      skipped and the operation is continued. All warning messages and
      errors (if occurred) are written to the file
      edg-job-cancel_<UID>_<PID>_<timestamp>.log under the /tmp directory.
      Location of the log file is configurable.

--logfile file_path
      when this option is specified, the command log file is relocated to
      the location pointed by file_path
```

--debug
when this option is specified, information about the API functions called inside the command are displayed on the standard output and are written to the file edg-job-cancel_<UID>_<PID>_<timestamp>.log under the /tmp directory too. Location of the log file is configurable.

edg_jobId
job identifier returned by edg-job-submit. The job identifier list must be the last argument of this command.

Glosario de términos

A

AA	Application Area
AAA	Authentication Authorization Accounting
ACL	Access Control List
AFC	Administrative Federation Committee
AFM	Administrative Federation Meeting
AFS	Andrew File System
AGM	Architecture Group Member
AIDA	Abstract Interfaces for Data Analysis
ALICE	LHC physic experiment
AliEn	Grid for Alice experiment
AM	Activity Manager
APEL	EGEE/LCG Accounting Application
API	Application Programming Interface
ARDA	Architectural Roadmap Toward Distributed Analysis
ASIS	Application Software Installation Server
ATF	Architecture Task Force (replaced by PTF)
ATLAS	LHC physic experiment
AUP	Acceptable Use Policy
AWG	Application Working Group

B

BaBar	B and B-bar experiment
BAR	Bandwidth Allocation and Reservation
BDII	Berkeley Database Information Index
BQS	Batch Queue System

C

CA	Certification Authority
CA	Consortium Agreement
CAS	Community Authorization Service
CASTOR	CERN Advanced STORage Manager
CERN	European Organisation for Nuclear Research
CHEP	Computing in High Energy Physics

CIC	Core Infrastructure Centrer
CIS	Core Infrastructure Services
ClassAd	Classified Advertisement
CLI	Command Line Interface
CLHEP	Class Library for HEP
CM	Cluster Manager
CMMi	Capability Maturity Model Integration
CMS	LHC physic experiment
Compass	Physics experiment at CERN
Condor	High Throughput Computing Project
CP	Certification Policy
CPS	Certification Practice Statement
CRL	Certificate Revocation List
CSS	Cascading Style Sheets
CT	Certification and Testing Team
CVS	Concurrent Versions System

D

D0	D0 Experiment
DAGS	Directed Acyclic GraphS (work-flow software from Condor project)
DAIS	Data Access and Integration Services
DataGrid	European DataGrid
DB	Database
DC	Data Challenge
dcap	dCache Access Protocol
DCS	Dynamic connectivity Service
DEISA	Distributed European Infrastructure for Supercomputing Aplications
DES	Data Encryption Standard
diffserv	differentiated services
DIT	Directory Information Tree
DLI	Data Location Interface
DN	Distinguished Name
DoS	Denial of Service
DPM	Disk Pool Manager
Dteam	Deployment team

E

e-IRGSP	e-Infrastructure Reflection Group Support Project
EAC	External Advisory Committee

EC	European Commission
EDG	European DataGrid
EDMS	Engineering Data Management Service
EDT	European DataTag
EELA	Extending EGEE to Latin America
EGAAP	EGEE Generic Application Advisory Panel
EGEE	Enabling Grids for E-sciencE
EGO	European Grid Organisation
eIRG	e-Infrastructure Reflection group
EIS	Experiment Integration Structure
EMBnet	European Molecular Biology network
EMT	Engineering Management Team
ENOC	EGEE Network Operation Centre
EO	Earth observation
ERA	European Research Area
ESC	EGEE Support Committee
ESM	Experiment Software Manager
ESR	Earth Science research
ESUS	Experiment Specific User Support
ETICS	eInfrastructure for Testing, Integration and Configuration of Software
EU	European Union
EUGridPMA	European Grid Authentication policy management authority for e-science

F

FAQ	Frequently asked questions
FCR	Freedom of Choice for Resources
FNAL	Fermi National Accelerator Laboratory
FR	Federation representative
FTE	Full Time Equivalent
FTS	File Transfer System

G

GAA	Generic Authorization and Access
GAE	HEP software
GAG	Grid Application Group
Ganglia	High Performance Computing Monitoring System
GAP	Gender Action Plan
GASS	Global Access to Secondary Storage
GD	Grid Deployment

GDB	Grid Deployment Board
GE	Gigabit-Ethernet
GEANT	European Academic Network
GENSER	GENerator SERvices
GENIUS	Grid Enabled web eNvironment for site Independent User job Submission
GFAL	Grid File Access Library
GGF	Global Grid Forum
GGUS	Global Grid User Support
GigE	Gigabit Ethernet
GIIS	Grid Index Information Server
GILDA	Grid INFN Laboratory for dissemination activities
GIP	Generic Information Provider
GK	GateKeeper
gLite	Codename of the middleware software suite developed by JRA1
GLUE	Grid Laboratory for a Unified Environment
GMA	Grid Monitoring Architecture
GN2	Codename for GEANT-2 (the successor to the GEANT network)
GOC	Grid Operation Center
GOCDB	Grid Operations Centre Database
GRAM	Globus Resource Allocation Manager
Grid3	The Grid 3 Project in the USA
GridICE	Grid monitoring software
GRIP	Grid Resource Information Protocol
GRIS	Grid Resource Information Service
GRRP	Grid Resource Registration Protocol
GSI	Grid Security Infrastructure
GSC	Grid Support Center
GSS	Generic Security Service
GST	Grid Services Testing
Gstat	Grid Statistics monitoring tool
GT	Globus Toolkit
GUID	Grid User Identifier
GUMS	Grid User Management System
GUT	Grid Unit Testing

H

HEP	High Energy Physics
HepCAL	HEP Application Grid requirements
HPC	High Performance Computing

HSM	Hierarchical Storage Manager
HTC	High Throughput Computing
HTTP	HyperText Transfer Protocol
HTTPS	Secure Socket Layer HyperText Transfer Protocol

I

I3	Integrated Infrastructure Initiative
IA64	Instruction Architecture-64
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers (standardisation)
IS	Information Service
ISO 9001	International Organization for Standardization
IST	Information Society Technologies
ITIL	Information Technology Infrastructure Library

J

JCS	Job Control Service
JDL	Job Description Language
JDK	Java Development Kit
JRA	Joint Research Activity
JRA1	EGEE Middleware Re-engineering and Integration activity
JRA2	EGEE Quality Assurance activity
JRA3	EGEE Security activity
JRA4	EGEE Network Services development activity
JSPG	Joint security policy group

K

KCA	Kerberized Certificate Authority

L

LAN	Local Area Network
LB	Logging and Bookkeeping Service
LCAS	Local Centre Authorization Service
LCG-0/1/2	LHC Computing Grid Middleware 0/1/2
LCG	LHC Computing Grid
LCMAPS	Local Credential MAPping Service
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
LFC	LCG File Catalog

LFN	Logical File Name
LHC	Large Hadron Collider
LHcb	LHC physic experiment
LRC	Local Replica Catalog
LRMS	Local Resource Management System
LSF	Load Sharing Facility

M

MDS	Monitoring and Discovery Service
MDS	Metacomputing Directory Service
MoU	Memorandum of Understanding
MPI	Message Passing Interface
MSS	Mass Storage System
MW	Middleware
MyProxy	Provides a credential repository so that users can be authenticated
MySQL	Structured Query Language

N

NA	Networking Activity
NA1	EGEE Project Management activity
NA2	EGEE Dissemination and Outreach activity
NA3	EGEE User Training and Induction
NA4	EGEE application identification and support activity
NA5	EGEE International Cooperation activity
NE	Northern Europe Federation
NEG	Northern European Grid
NFS	Network File System
NGIS	National Grid Initiatives
NMI	NSF Middleware Initiative (National Science Foundation)
NOC	Network Operation Center
NPM	Network Performance Monitoring
NREN	National / Regional e-Network
NS	Network Server
NS	Name Server

O

OAG	Operation Advisory Group
OCC	Operations Coordination Centre
OGSA	Open Grid Services Architecture

OGSI	Open Grid Services Infrastructure
OMC	Operation Management Center
OMII	Open Middleware Infrastructure Institute
Oracle	Data base management system
OSCT	Operational Security Coordination Team
OSG	Open Science Grid
OSI	Open Source Initiative

P

PBS	Portable Batch System
PD	Project Director
PDG	Protein Design Group
PEB	Project Execution Board
Perl	Practical Extraction and Report Language
PFN	Physical File Name
PIC	Policy and International Cooperation
PID	Process IDentifier
PKCS	Public Key Cryptography Standards
PKCS12	File format used to store private keys with accompanying Public key certificates protected with a password-based symmetric key.
PKI	Public Key Infrastructure
PMA	Policy Management Authority
PMB	Project Management Board
PO	Project Office
POB	Project Overview Board
POOL	Pool of Persistent Objects for LHC
PPS	Pre-production Platform Service
PPT	Project Progress Tracking tool
PR	EU Periodic Reports
PR	Public Relations
PROOF	HEP software
PTD	Project Technical Director
PTF	Project Technical Forum
PX	ProXy Server

Q

QA	Quality Assurance
QAG	Quality Assurance Group
QAM	Quality Assurance Management

QAR	Quality Assurance Representative
QI	Quality Indicator
QoS	Quality Of Services
QR	EU Quarterly Report
QUATTOR	Optimisation Toolkit for Optimising Resources

R

R-GMA	Relational-Grid Monitoring Architecture
RA	Registration Authority
RAID	Redundant array of inexpensive disks
RB	Resource Broker
RBAC	Rules Based Access Control
RC	Resource Center
RDBMS	Relational Data Base Management System
Remedy	Problem tracking software for helpdesks
RFC	Request For Comments
RFIO	Remote File Input/Output
RGMA	Registry Grid Monitoring Architecture
RHEL	Red Hat Enterprise Linux
RLI	Replica Location Index
RLS	Replica Location Service
RM	Replica Manager
RMC	Replica Metadata Catalog
RMS	Replica Management System
ROC	Regional Operation Center
ROS	Replica Optimization Service
RPM	RPM package Manager
RSA	Public key cryptographic system
RSL	Resource Specification Language
RSS	Really Simple Syndication
RTAG	Requirements Technical Assessment Group

S

SA	Specific Service Activity
SA1	EGEE European Grid Support, Operation and Management activity
SA2	EGEE Network Resource Provision activity
SA3	Integration & Testing activity within EGEE II
SASL	Simple Authorization & Security Layer
SC2	Software Computing Committee

SCG	Security Coordination Group
SCM	Software Configuration Management
SCRAM	Software Configuration, Release And Management
SDK	Software Development Kit
SE	Storage Element
SEAL	HEP software
SEE	South East Europe
SEEGRID	South East Grid e-Infrastructure Development
SEEREN	South East European Research Networking
SFT	Site Functional Test
SGM	Security Group Member
SIMBA	CERN Mailing list management tool
SIPS	Site Integrated Proxy Services
SL	Scientific Linux
SLA	Service Level Agreement
SLOC	Software Lines Of Code (number of)
SLR	Service Level Request
SLS	Service Level Specification
SM	Software manager
SMP	Symmetric Multi Processor
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SPEC	Standard Performance Evaluation Corporation
SPI	Software Process Infrastructure
SRB	Storage Resource Broker
SRM	Storage Resource Manager
SSA	Special Support Action
SU	Support unit
SURL	Storage URL
SW	Software
SWE	South West Europe Federation

T

TA	Technical Annex
TAB	Technical Advisory Board
TCG	Technical Coordination Group
TD	Technical Director
TLS	Transport Layer Security
TNLC	Technical Network Liaison Committee

Torque	Open source resource manager
TPM	Ticket Process Management
TT	Trouble Ticket
TURL	Transport URL

U

UI	User Interface
UIG	User Information Group
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Universal Resource Locator
UUID	Universal Unique ID

V

VV	Verification and Validation activity
VDT	Virtual Data Toolkit
VO	Virtual Organization
VOMS	Virtual Organization Membership Service

W

WAN	Wide Area Network
WBS	Work breakdown structure
WCIT	World Congress on Information Technology
WLM	WorkLoad Management software
WMS	Workload Management System
WN	Worker Node
WSRF	Web Services Resource Framework

X

X.509	Public-Key Infrastructure
XML	eXtensible Markup Language
XSL	eXtensible Stylesheet Language

Y

YAIM	Yet Another Installation Method