An Introduction to Grid Computing Using EGEE

John Walsh, Brian Coghlan, and Stephen Childs

Abstract Grid is an evolving and maturing architecture based on several well-established services, including amongst others, distributed computing, role and group management, distributed data management and Public Key Encryption systems Currently the largest scientific grid infrastructure is Enabling Grids for e-Science (EGEE), comprised of approximately ∼250 sites, ∼50,000 CPUs and tens of petabytes of storage. Moreover, EGEE covers a large variety of scientific disciplines including Astrophysics. The scope of this work is to provide the keen astrophysicist with an introductory overview of the motivations for using Grid, and of the core production EGEE services and its supporting software and/or middleware (known by the name gLite). We present an overview of the available set of commands, tools and portals as used within these Grid communities. In addition, we present the current scheme for supporting MPI programs on these Grids.

1 Introduction

Grid is an evolving and maturing architecture based on several well-established services, including amongst others, distributed computing, role and group management, distributed data management and Public Key Encryption systems [1]. Currently the largest scientific grid infrastructure is Enabling Grids for e-Science (EGEE) [2], comprised of ∼250 sites, ∼50,000 CPUs and tens of petabytes of storage. Moreover, EGEE covers a large variety of scientific disciplines, including High

J. Walsh (✉)
School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland,
John.Walsh@cs.tcd.ie

B. Coghlan
School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland,
coghlan@cs.tcd.ie

S. Childs
School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland,
Stephen.Childs@cs.tcd.ie

Walsh, J. et al.: *An Introduction to Grid Computing Using EGEE*. Lect. Notes Phys. **791**, 47–80 (2009)
DOI 10.1007/978-3-642-03370-4_2

Energy Physics (HEP), Geophysics, Astrophysics and Biomedical Computing and these disciplines all have significantly different resource and security requirements.

The authors, as members of the EGEE, Int.EU.Grid [3] and Grid-Ireland [4] communities, present a basic introduction to Grid Computing with an EGEE-centric coverage of the current architecture, middleware and services in use. The scope of this work is to provide the keen astrophysicist with an introductory overview of the motivations for using Grid, and of the core production EGEE services and its supporting software and/or middleware (known by the name gLite [5]). We present an overview of the available set of commands, tools and portals as used within these Grid communities. In addition, we present the current scheme for supporting MPI programs on these Grids. For a comprehensive treatment on using the EGEE Grid, the reader is referred to the gLite Users Guide [6].

1.1 Basic Grid Concepts

Formally, a (computational/data) Grid is a system that is concerned with the integration, virtualisation and management of services and resources in a distributed, heterogeneous environment that supports collections of users and resources (virtual organisations) across traditional administrative and organisational domains (real organisations) [7]. Grids are conceptually based on the model of the provision and consumption of utility services, such as electricity, water and gas supplies [8]. The consumers/users are mostly concerned about the ease of access to and the quality of the provided services. Only very rarely do they want to know about the gory details of how these services are provided to them via the underlying infrastructures, such as water pipes and drainage. In the case of Grid Computing, the infrastructure is often referred to as an e-infrastructure [9]. Grids must be flexible, use open protocols, be secure and must allow co-ordination of resource sharing amongst dynamic collections of individuals, institutions and resources.

1.2 Virtual Organisation Motivations

Academic work, by its nature, often involves rather complex collaborations with people and resources coming from distinct administrative domains. The partners may bring a different set of skills, technology or compute and data resources into the collaboration in order to make it a success. These collaborations may be large, but may be localised, national or may even span international boundaries. Organisation and management of these collaborations give rise to several challenges:

- How do I share my data and resources with someone who does not work locally?
- In large collaborations, it is unlikely that everyone knows each other personally. How do we verify that someone who presents themselves as a member of the collaboration is bone fide?

- How do we ensure that, as a resource provider, we can limit our resources to a well-defined set of known and trusted users?
- Can we scale this in an efficient way, across institutional and, potentially, national boundaries?

Although some of these requirements can be addressed using simple technologies such as secure login shells, secure web servers (e.g. Apache [10] over the https protocol) and secure file transfer with password protection and encryption, it is clear that these do not scale very well. One may manage a small collection of individuals across a small number of domains in such a way, but consider the problem of adding a new member of a virtual organisation to 50 different administrative domains! How do we define or control the access rights that a person or group of people may have on the collection of domain resources? Other issues such as password synchronisation, group management also arise. Grids can address these problems by providing:

- Internationally recognised and secure identity credentials which can be renewed annually or revoked quickly.
- Support for virtual organisations via software and services which respect domain and grid policies.
- Middleware support providing controlled access to resources with additional support for roles and groups, secure access to resources (such as batch systems and storage), subject to the resource providers' policies and single sign-on support (login once, access all matching resources).

Grids based on the gLite middleware supplement the standard Globus-based middleware [11] by providing a way to enumerate and publish information about the resources available (e.g. batch systems), storage usage and storage access interfaces, location, basic site details and the access controls and rights that are applied to site resources. This allows a form of service and resource discovery, permitting a grid user to have resources automatically selected for him/her based on a simple job specification language.

1.3 The International Dimension

Apart from the gLite-based Grids, there are many examples of other Grids with large-scale operations:

- Open Science Grid [12] (USA), an equivalent to EGEE
- TeraGrid [13] (USA), a supercomputing grid
- NAREGI [14] (Japan), a national grid
- National Grid Service [15] (UK), a national grid
- DEISA [16] (EU), a supercomputing grid
- NorduGrid [17] (Nordic Countries), a supra-national regional grid
- Int.EU.Grid [3] (EU), a grid for near-interactive jobs.

As Grid middleware was evolving, the lack of a standardisation body led to divergence in the implementations of some key architectural elements (such as grid storage, the types and formats of information published about each site). The importance of standardisation and interoperability has now been widely recognised and accepted by all of these Grids, and a great deal of effort is being made under the Open Grid Forum's [18] "Grid Interoperability Now" (GIN) [19] initiative.

1.4 The Role of Certificate Authorities

Access to grid resources is based on the establishment of trust relationships between the users and the resource providers. In order for this trust relationship to work in any scalable manner, a key requirement is the establishment of a set of trust authorities (identity providers/certificate authorities) who are mutually recognised by one another. It has been agreed that these identity providers must uphold a common set of agreed standards on establishing the bone fide identity of someone who presents themselves as requiring a Grid identity. The identity providers accept on good faith and with a good degree of confidence that the other identity providers have upheld all the required standards.

Conceptually, the Grid identity providers are similar to passport offices. When a person applies for a passport, he/she must fill out an application form and provide all required details. Additionally, he/she must provide some proof of identity, such as a drivers licence, birth certificate, photographs and signatures. After all the required proof of identity has been provided, the passport office will scrutinise the supplied materials and then accept or reject the application.

In the grid world, application and approval is a multi-staged process:

- The applicant needs to identify their certificate authority from a list of recognised identity providers.[1]
- The identity provider's web site/application form will include a list of registration authorities, where the applicant can physically present themselves, complete with relevant institutional ID.
- The application is typically completed via a web browser form, where the identity provider uses a public key encryption methodology to allow the user to sign their application with a secret key and encrypt the transactions. With the exception of DEISA (and this is being remedied), these grid infrastructures use a common public key encryption system based on the Globus toolkit [20].
- The selected registration authority will require that the applicant presents themselves physically and brings proof of identity with them.
- If the registration authority accepts the identity as bona fide, then the application can be electronically signed and sealed.
- The approved applicant is reminded of the duty of care that is placed upon them in protecting their electronic identity.

[1] http://www.eugridpma.org/members/worldmap/

- An electronic credential is then issued, where the applicant's identity cannot be repudiated.
- The credential must be converted by the successful applicant into the X.509 format as used on the Grid.

A great deal of effort is being made by the network and grid communities to simplify the process of acquiring access to the Grid by delegating and federating the identity management to the user's local institution or domain. An interesting example of this is the federated identity management as developed by the Swiss national grid [21].

1.5 The EGEE Grid

The Enabling Grids for e-Science Grid infrastructure is one of the largest and most complex grid infrastructures built to date. It is funded by the European Commission and the partner institutions. EGEE (phase I and II) evolved from the European Data Grid [22] (EDG) and the Large Hadron Collider Grid [23] (LCG) projects. EGEE was comprised of 92 partner institutions from 27 countries and had affiliates in ∼20 other countries. It had over 600 people involved in the overall day-to-day running of the infrastructure. EGEE phase III has now started and is of the same scale. The most salient aims are to

- Provide compute and data storage processing for e-Science;
- Support a large range of scientific communities, each with different requirements, and to encourage new application areas and new user communities;
- Run at production level 24 hours per day, 365 days per year;
- Provide a production-quality infrastructure;
- Develop and maintain innovative grid middleware (gLite) to ease user access and to increase their productivity;
- Run extra testing, certification and pre-production services to ensure overall quality of service;
- Establish grid interoperability with other leading grid projects;
- Encourage the adoption of grid technology in business and commercial domains,
- Provide and ensure top quality administrative and technical management.

At the time of writing, EGEE is composed of ∼250 sites in over 50 different countries, where these countries are partitioned into a set of 10 federations. The current production Grid handles ∼150,000 user jobs per day. The federal structure accommodates the ability to scale the infrastructure management. Each federation has an operations management team which has special responsibility for resolving operational problems of production sites in that region, providing regional helpdesks, and helping new sites to join the Grid by ensuring that candidate sites fulfil a set of well-defined requirements and pass a set of well-defined tests in a

pre-production environment. Once a candidate site has satisfied the requirements, it will be allowed to join the production environment.

1.6 Initial Grid Applications

As, historically, the EGEE Grid evolved from the EDG and the LCG Grids, its initial application domains were geared towards supporting the goals and ambitions of the LHC High Energy Physics (HEP) experiments [24] at CERN. EGEE-I was aimed at establishing the move from the experimental and developmental nature of these Grids and to move them into providing a production environment for supporting generic e-Science applications across as wide a range of disciplines as possible. Initially the requirements of the biomedical research community was regarded as sufficiently juxtapose from those of HEP that it was prioritised as a scientific domain that would bring out the best in the provision of grid services. EGEE-I was more than successful in achieving this goal; many other research domains started to use the Grid for production work. Under EGEE-II the applications and scientific domains were widened further to include Astronomy, Astrophysics, Earth Sciences, Computational Chemistry and Nuclear Fusion amongst others.

The EGEE Grid is not intended to simply provide endless raw processing power and storage resources for large scientific applications. Experiments, or virtual organisations, wishing to use the Grid are expected, to some reasonable degree, to contribute resources back to the grid community. Indeed, as the EGEE is not a resource provider per se (the resources are provided by the constituent sites), the VOs may wish to negotiate with the sites to allow them to have access to the sites' resources. The Grid helps to accommodate access to a larger pool of available resources and to accommodate the management of virtual organisations and their data. In principle, the virtual organisations should contribute as much as they use, with the overall effect being a zero-sum game.

2 The EGEE Grid Architecture and Implementation

The EGEE Grid attempts to support the needs of the general scientific community. Its scale requires a significant degree of management at both the project level and at the site and service levels. We shall concentrate here on how the day-to-day services are managed.

At the service level, the EGEE is rather complicated. We have chosen to present a simplified overview of its structure by partitioning it into a set of "Top" or "Higher" level services which help manage the grid infrastructure and its associated VOs and a set of lower level services, such as those provided by the sites to enable grid users to access typical site-level resources such as batch systems and storage systems. See Fig. 1.

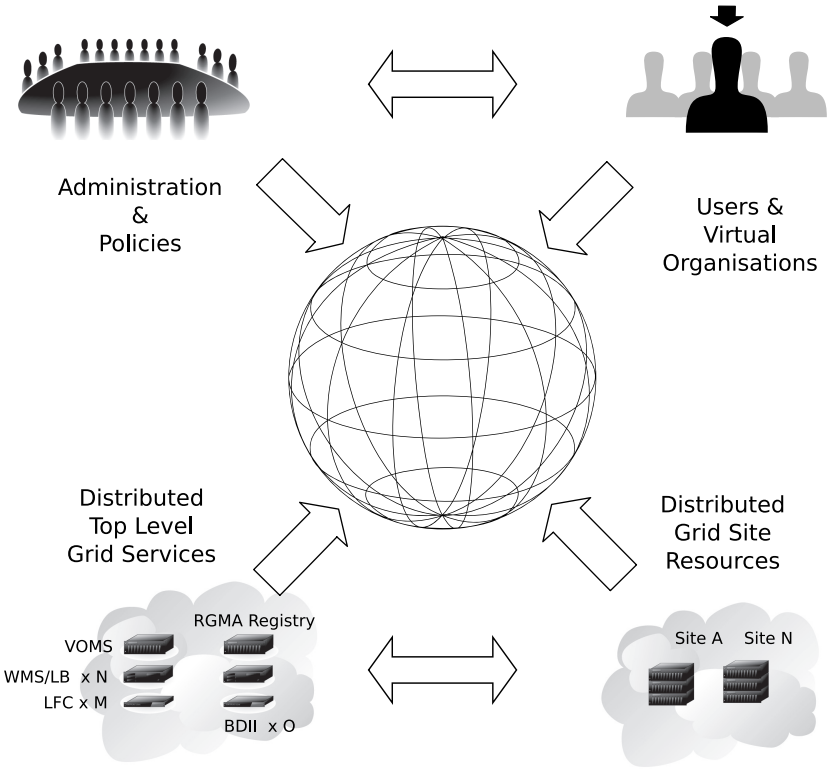

Fig. 1 EGEE actors

2.1 Grid Services

The top-level grid services, which may be centralised or distributed, may be viewed as the set of critical services which enable the Grid to function as an ensemble of

- Administrative Infrastructure
  - Global Grid User Support (GGUS)
  - Grid Operations Centre Database (GOCDB)
  - Service Availabilty Monitoring Service (SAMS) testing
  - Core Infrastructure Centre (CIC)
  - Certificate Authorities (CA)
  - Grid Training
- Core Services
  - Information services
  - Authentication and authorisation
  - Workload management
  - Data management

- Site Services
  - Job management – gatekeeper services, batch systems
  - Storage management – storage services
  - Information – resource and site-BDIIs, R-GMA monitoring
- Access Services
  - User interfaces (UIs)
  - Grid portals

2.2 Accessing the Grid

A grid site may optionally, and almost without exception does, include a grid user interface (UI) ([6], Sect. 3.3.2) – a host that allows the user to interact with the Grid and its services. In addition, a number of web-based portals are also available. Architecturally, the main difference between a user interface and a portal is that a UI is sometimes considered a site service, whereas a portal is definitely a higher-level pan-Grid service and may even span several Grids.

2.2.1 The User Interface

The UI provides a powerful set of command line tools. It also includes a number of application libraries, which allows the grid users to develop their own grid applications. In general, a UI will enable the user to

- List resources suitable to execute a given job,
- Submit jobs for execution,
- Cancel submitted jobs,
- Retrieve the output of finished jobs,
- Show the status of submitted jobs,
- Retrieve extra logging and bookkeeping information for jobs,
- Copy, replicate and delete files from the Grid,
- Discover services and monitor status using the information system,
- Develop grid-enabled applications (APIs/Libraries).

2.2.2 Grid Portals

Portals help address the need for “pervasive access” to the Grid. Access to the Grid and grid services are presented via relatively intuitive and easy-to-use web-based interfaces. VOs are encouraged to actively develop portal applications and their associated web interfaces. Currently, there are a number of portals and portlet servers available, for example:

- Genius [25], based on EnginFrame [26]
- Migrating Desktop [27], from Int.EU.Grid
- P-Grade [28], based on GridSphere [29]

These portals hide the complexities of Grid access mechanisms. They provide certificate, job and data management. Genius is extensively used by EGEE's Gilda [30] test bed for training purposes. Migrating Desktop specifically supports visualisation for near-interactive jobs. P-Grade enables workflow creation, execution, import, export, archival and monitoring, with a built-in graphical workflow editor. P-Grade also provides a single interface to multiple Grids.

2.3 EGEE Sites

An EGEE site must agree to the provision and maintenance of a number of site services. The principal ones are discussed below.

2.3.1 The Gatekeeper

The Gatekeeper ([6], Sect. 3.3.3) is the bridge between the Grid and the resource provider's batch system. The ensemble of the Gatekeeper, batch system and its associated compute nodes is often referred to as a Compute Element (CE) ([6], Sect. 3.3.3). Currently there are a number of different flavours of compute elements supported by EGEE: PBS (OpenPBS [31] and Torque/Maui [32]), LSF [33], Condor [34] and SGE [35]. Compute Nodes (Worker Nodes (WN)) attach to the batch system to run grid jobs. The Site Manager installs middleware and sets up queues for grid access, as well as addtional policies to enforce fair-sharing. The CE publishes information about the status of the batch system, for example:

- Access policy to Grid Queues (which VOs supported at site)
- Number of CPUs available (per VO)
- Expected time to run a job for particular VOs

This information is used by the higher levels of grid middleware to target appropriate jobs to the CE. Once targeted, those jobs are authenticated via grid mechanisms and the grid user is mapped to a local account. An account from a pool of resusable accounts is obtained (this account is cleaned after the job's completion). Resource usage of the local batch system is logged and republished later. Jobs execute on the WNs (scheduled by the batch system), not on the gatekeeper. For better efficiency the VO may install VO-specific software at the site so that this software does not need to be conveyed through the Grid to the CE with each job.

2.3.2 The Storage Element

An EGEE site must provide one or more Storage Elements ([6], Sect. 3.3.4). The storage elements provide a set of standard grid-enabled access methods (protocols) allowing the grid user to access some of the site's storage facilities. Given the scale of the data sets encountered in Grids, data management has a high priority and sophisticated services are provided for creating, using and maintaining data sets as

collections (above filesystems) across pools of disks and tape robots. Just as for the CE, information is published about

- Type of Storage Element and protocols supported,
- Supported VOs and their Access Control,
- Available space for VO,
- Duration that data may be kept at site:
  - Temporary,
  - Pinned (retained until further notice) or
  - Permanent.

Site Managers are expected to ensure that data on a grid site is maintained according to agreed policies.

2.3.3 The Site-BDII Service

Each resource at a site, such as the CE and SE, collects and then publishes information about their state on a regular basis. The information is processed and republished in a well-defined format, known as the "Glue Schema" [36]. The role of a site-BDII ([6], Sect. 5.1.5) is to gather this information from a list of known supported services and resources and then republish this in an aggregated form, showing the overall state of the site.

2.3.4 The Site R-GMA Service

As R-GMA acts as a distributed "virtual database", a site's R-GMA server hosts some of that data and will deal with all R-GMA interaction from clients at the site. The server stores data published from the local clients (*primary producers*) and may also collect data from other sites and republish it (*secondary producers*). See Sect. 5.2.1 of the gLite Users Guide for further details.

2.4 Core Grid-Wide and VO-Wide Services

The core Grid and VO services are responsible for the overall management of the user's access to the Grid.

2.4.1 Information Systems and Information Providers

The provision of consistent state information from all of the resources on the Grid is paramount. Currently there are two commonly used information systems used with the EGEE[2] – the Berkeley Database Information Index (BDII) and the Relational Grid Monitoring Architecture (R-GMA) system.

[2] gridICE is also commonly used. It allows queries of historical site status data.

GIIS – BDII

The EGEE BDII ([6], Sect. 3.3.5) implements a Grid Information Index Service (GIIS) via a hierarchical system with three levels of *information providers*. As illustrated in Fig. 2, information is "pulled" from the lower to higher levels.

1. Resource-BDII
2. Site-BDII
3. Top-level BDII

At the lowest level is the *resource-BDII*. This service runs on all machines providing one or more grid services (such as a GateKeeper or MyProxy [37]). The resource-BDII will periodically run a set of scripts to determine the current status of that service, and this information is processed and may have extra infomation relating to supported VOs appended. The next level is the *site-BDII*. This periodically aggregates and caches the data provided by the resource-BDIIs. Similar to the resource-BDII, extra data relating to supported VOs and about the site itself will be appended. At the highest level is the *top-level BDII*. The top-level BDII queries the set of site-BDIIs (as determined from a pre-generated list). As the information is stored in an LDAP [38] database, the status of the Grid or services on the Grid can be queried and filtered using standard *ldap* commands. Further details about querying the BDII may be found in Sect. 5.1 of the gLite Users Guide [6].

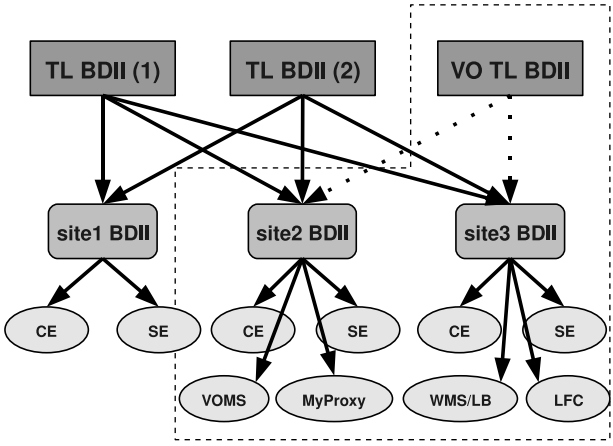

Fig. 2 The hierarchy of informaion

R-GMA

R-GMA [39] is an implementation of the Grid Monitoring Architecture (GMA). It implements a distributed "virtual relational database" which can be manipulated using SQL-like statements.

At an application developer or user level R-GMA is very interesting. As we have seen, the BDII information system deals exclusively with hierarchical site status information. The schema (allowed valid data) is well defined and is "read-only". A user has no ability to modify this data, but may hierarchically query it and filter it. By contrast, R-GMA is relational and with R-GMA the grid user has the ability to create new schemas and tables, insert data with a limited lifetime, relationally query data, subscribe to data publishers (application code that inserts data), etc. R-GMA applications can be built using a number of different languages including C, C++, Python and Java™. A command-line utility can also be used to interact with shell scripts, etc. The R-GMA system is quite extensive and is beyond the scope of this chapter. The reader is refered to Sect. 5.2 of the gLite Users Guide for more details.

2.4.2 Authentication and Authorisation

While authentication (*AuthN*) refers to the process of unrepudiately verifying the identity of a subject (person/host/service), authorisation (*AuthZ*) is the explicit act of denial or approval of access to a service once the subject has correctly authenticated his/her identity. AuthN is supported by a range of facilities as explained in Sect. 1.4. There are two important core AuthN services: CRL services and MyProxy services. AuthZ is supported in EGEE via the VOMS service.

CRL Service
Each Certificate Authority (CA) must publish its certificate revocation list (CRL) via a basic web service so that all dependent services (almost everything) can determine what certificates are no longer valid.

Virtual Organisation Management Service (VOMS)
VOMS [40] is a crucial service in facilitating the organisation of the members of a VO by defining the members' roles and privileges. It is maintained by agreed VO managers. Resource providers have no direct role in the management of a VOMS server.

When a user "logs on" to the Grid, they do so by selecting their VO and optionally one or more of their roles. During the logon, the VOMS server will be contacted and this will yield a short-term VOMS proxy certificate that includes attributes saying what the user's roles and privileges are (an Attribute Certificate (AC)). This dialogue between the user's logon client and the VOMS server is secured via PKI mechanisms.

MyProxy
The MyProxy service is a long-term credential service that a user can employ to

- interact with the Grid from different locations without duplicating their certificate;
- delegate to a portal the power to act on their behalf;
- run jobs that might last longer than the lifetime of a short-lived VOMS proxy certificate.

2.4.3 Workload Management System (WMS)

The WMS [41] is at the heart of managing the user's jobs on the Grid. As such it is a rather complex service which accepts an authorised user's job and assigns it to an appropriate site's CE. It employs a technique known as *match-making* to make the decision to send a job to a chosen CE based on the job's description (expressed using the *Job Description Language*) and other top-level Grid status information. It also manages the staging of small input/output/stderr files – large data should be handled using the data management tools (see Sects. 2.3.2 and 4).

When a job is submitted through the WMS, the state changes are logged in the *Logging and Bookkeeping* (LB) [42] service. This allows the job to be traced fully from the time of submission until after the job's results have been retrieved. The WMS is also capable of implementing some job resilience and can re-submit after some detected failures.

2.4.4 Data Management

There are currently three high-level data management services for the Grid: the LCG File Catalog (*LFC*), the File Transfer Service (*FTS*) and the *AMGA* metadata service. We shall cover the topic of data management in more detail in Sect. 4.

LFC
The LFC [43] lies at the heart of data management and is a critical service for efficient management of files across the grid infrastructure. This service helps the user locate files and their replicas on the Grid. The LFC allows a user to associate an *Alias* (LFN), a user-friendly name, to a file that has been copied onto the Grid using the replica management tools. It will also keep track of all replicas of that file. The LFC implements a hierarchical namespace and allows the user to systematically manage collections of aliases in that namespace. The LFC will publish information into the BDII and this is integrated with the WMS match-making system. This allows jobs to match against sites which store the required files. The use of replicas is a latency-hiding stratagem tantamount to whole-file cacheing.

FTS
The FTS [44] is a new service allowing the user to schedule asynchronous and reliable file replication from one SE to another. It operates similar to a batch system in that the user can submit a "Job" to the FTS. The "Job" has a "state" that can be queried by the user. As such, it is simply an aid to placement of replicas.

AMGA
AMGA [45] is a grid-enabled metadata service. The metadata catalog stores data describing the contents of files registered (using the LFC) on the Grid. The metadata service then allows grid users to search for files that fit a given description. AMGA is an optional service, but is highly recommended where large volumes of data are processed. AMGA is an advanced topic and is beyond the scope of this chapter. The reader is refered to the AMGA users manual [46].

2.5 Administrative Services

2.5.1 GGUS Helpdesk

The *Global Grid Users Support* (GGUS) [47] portal provides a central helpdesk for sites, users and VOs. All problems submitted via GGUS are filtered and distributed to the relevant *Support Units*, i.e. helpdesks or parties. Additionally, GGUS maintains a list of these support units and a list of links to a Frequently Asked Questions (FAQs). Grid operations tickets relating to problematic sites and services are co-ordinated via GGUS. Each EGEE federation provides a team of people to handle operational issues. These teams work on a weekly rotation basis.

2.5.2 Grid Operations Centre DB

The Grid Operations Centre Database (GOCDB) [48] is a service that maintains a list of all sites, site managers, contact details and site status (e.g. the site has not yet been certified to run grid jobs, or the site has been certified and is now a production site). The site manager uses the GOCDB to schedule maintenance periods and to announce these periods to the wider community. The GOCDB may also be queried to list all sites that are currently in production.

2.5.3 Service Availability Monitoring

The EGEE infrastructure is periodically tested via the Service Availability Monitoring system (SAMS) [49]. The test-suite checks if all production services are working correctly. The results of these tests are maintained and visible via a web-interface. In addition, VOs can integrate their own test-suite into SAMS. This allows the VO to test all production sites which support that VO and to use the results of these tests to assemble a list of correctly functional and non-functional production sites. The results can be used to raise tickets with the failing site via GGUS or can be used by VOs who maintain their own top-level BDII to improve the quality of service by guaranteeing that jobs will only go to non-failing sites.

2.5.4 Core Infrastructure Centre

The Core Infrastructure Centre (CIC) [50] provides a portal which provides EGEE management and operational tools; it acts as an entry point for EGEE operational needs to manage available information about VOs and to monitor and ensure grid-wide operations of resources and services.

2.5.5 Grid Training

Grid training and realistic grid training test beds are fundamental to encouraging new users and communities onto the Grid. The EGEE-II NA3 activity coordinates

training activities and hosts a repository of training material. EGEE partners are encouraged to run regular training events and to localise the material [51] as needed. In addition, NA3 maintains a list of up-and-coming training events which is available on the EGEE NA3 web site [52]. The *Gilda* [30] training environment gives the curious scientist the oppurtunity to use a dedicated set of training resources via a web portal. Grid-Ireland offers an adaptive e-Learning infrastructure which monitors a user's progress through a set of available courses [53] that can be integrated into discipline-specific syllabii, such as those for pan-institutional Graduate School courses.

3 Computing on the EGEE Grid

In Sect. 1, we argued that one of Grid's many advantages is that it enables researchers to work in distributed collaborations and that it facilitates the secured sharing of resources and results. We now describe how one may perform computational work on the Grid.

The authors do not argue that Grid should be seen as a replacement for high-performance computing (HPC) systems, since Grids and HPC are complementary to one another and both enhance the range and quality of scientific output. HPC systems may be attached to the Grid, as is the case with DEISA or in the case of some EGEE sites where speciality hardware, such as the Myrinet node interconnects, are installed. The vast majority of resources attached to the EGEE Grid are, however, loosely coupled commodity resources without any specialised hardware. In our view, Grid is best suited to high-throughput computing jobs where any of the following conditions are satisfied:

- The jobs have a known medium-term lifetime; however, support for long-running jobs is available via the MyProxy service.
- The jobs are relatively independent, for example Monte Carlo type simulations or Parameter Sweeps.

The Grid can also handle large volume batches of single jobs, complex workflows and MPI jobs (MPI support has been a particular outcome and success of the Int.EU.Grid project). However, cross-site MPI is not widely supported despite the technical success of Int.EU.Grid in providing for this paradigm.

3.1 Running Jobs on the Grid

At the time of writing there is a migration from the existing *gLite 3.0* middleware originally developed by the EU Datagrid (EDG) project and the LCG projects, and further developed by EGEE, towards a new *gLite 3.1* middleware based on web services. This is still taking place across the EGEE infrastructure. The migration involves many changes to how workload management is handled. Consequently it

introduces a set of new workload management commands. The examples presented here shall only use gLite 3.0 commands. These are listed in Table 1.

Handling jobs on the Grid is one of its most fundamental operations. In this section we will cover, with examples, jobs that assume the user is a member of a fictitious VO, *myvo*, that illustrate the basics of

- Dealing with grid credentials
- Logging on/off the Grid using VOMS
- The Job Description Language (JDL)
- Matching a job against available resources
- Submitting a job
- Saving the job's global ID
- Querying the job's status
- Cancelling a grid job
- Retrieving the job's output.

Table 1 Current job management commands for EGEE

LCG-2/gLite 3.0	gLite 3.1	Action
edg-job-list-match	glite-wms-job-list-match	Find resources which match job's requirements
edg-job-submit	glite-wms-job-submit	Submit a job to the WMS
edg-job-status	glite-wms-job-status	Get the job status from LB server
edg-job-get-output	glite-wms-job-output	Retrieve the output sandbox of a successful job
edg-job-get-logging-info	glite-wms-job-logging-info	Query logged job information from LB
edg-job-cancel	glite-wms-job-cancel	Cancel a submitted job
	glite-wms-job-delegate-proxy	Delegate a proxy to WMProxy

3.1.1 Handling Grid Credentials

In Sect. 1.4, we discussed the role of Certificate Authorities in establishing the trust relationship between the user and the Grid itself. Once the grid certificate has been approved by the CA, the user must

- Import the .p12 file from CA web site into a web browser
- Save the certificate locally in .p12 format from web browser
- Join a virtual organisation using the VO's secure membership entry page
- Obtain a login account on a UI or portal
- Login to the UI and create a .globus directory on the UI
- Copy the .p12 file to their .globus directory on the UI (using secure copy)
- Unpack the .p12 file to X.509 PEM format
- Set secure permissions on the new files
- chmod 400 userkey.pem to restrict private key read access to only the user
- chmod 644 usercert.pem to allow others to read the public key, but restrict write access to only the user

It is worth stressing that certificates must be kept safe and secure at all times. It is also worth repeating that there are combined efforts within the network and grid communities to greatly reduce the above complexity using federated identity.

3.1.2 Logging Onto/Off the Grid

Single-signon refers to logging onto the Grid, not logging onto the UI. Assuming that the user has already been given an account on a UI and is already logged into it in the usual manner (e.g. via ssh login) and that the user's credentials have been put in place in the user's *.globus* directory, then to signon to the Grid they need to enter:

```
$ voms-proxy-init  --voms myvo
Enter GRID pass phrase:
Your identity: /C=IE/O=Grid-Ireland/OU=tcd.ie/L=RA-TCD/CN=Jo Doe
Creating temporary proxy ..............................Done
Contacting  cagraidsvr10.cs.tcd.ie:15013
[/C=IE/O=Grid-Ireland/OU=tcd.ie/L=RA-TCD/CN=host/cagraidsvr10.
cs.tcd.ie] "myvo"
Done
Creating proxy ................................ Done
Your proxy is valid until Tue Jan  8 03:36:29 2008
```

This action creates a *VOMS proxy certificate* – a temporary certificate that is used for grid authentication. The proxy is valid for 12 hours by default. The *voms-proxy-init* command offers a number of extra command-line options which can

- Change the lifetime of the generated proxy (set by the VO managers),
- Use a different location for the certificate rather than the default,
- Assign specific role attributes.

The user may also employ the MyProxy service as a means of obtaining a valid VOMS proxy certificate.

3.1.3 Checking Proxy Certificate Status

As the proxy certificate is copied with the submitted job eventually to a Worker Node, the user must ensure that the proxy does not expire before the job is completed. The proxy certificate status can easily be checked prior to job submission.

```
$ voms-proxy-info
```

This command displays default information about the proxy certificate on standard output. In addition, if the certificate is valid the command will return code 0 to the shell and non-zero otherwise. This allows the command to be used for scripting

more complex workflows. Other options to display more specific information about the proxy are to

- check if the proxy exists and is valid (-exists),
- check the remaining proxy lifetime in seconds (-timeleft),
- print all known information about the proxy (-all),
- print the Distinguished Name (DN) of the proxy (-subject).

```
$ voms-proxy-info  --all
subject   : /C=IE/O=Grid-Ireland/OU=tcd.ie/L=RA-TCD/CN=Jo Doe/CN=
proxy
issuer    : /C=IE/O=Grid-Ireland/OU=tcd.ie/L=RA-TCD/CN=Jo Doe
identity  : /C=IE/O=Grid-Ireland/OU=tcd.ie/L=RA-TCD/CN=Jo Doe
type      : proxy
strength  : 512 bits
path      : /tmp/x509up_u59999
timeleft  : 11:44:31
=== VO myvo extension information ===
VO        : myvo
subject   : /C=IE/O=Grid-Ireland/OU=tcd.ie/L=RA-TCD/CN=Jo Doe
issuer    : /C=IE/O=Grid-Ireland/OU=tcd.ie/L=RA-TCD/CN=host/cagra
idsvr10.cs.tcd.ie
attribute : /myvo/Role=NULL/Capability=NULL
timeleft  : 11:44:32
```

3.1.4 The Job Description Language (JDL)

The JDL [54, 55] has a simple syntax for describing the properties of the job that the user would like to run on the Grid. The syntax is in the form of key-value pairs, i.e. “Parameter = Value”. Pairs can be used to define:

- The Executable,
- Stdin/stdout/stderr Files,
- Input Files,
- Output Files,
- Optional Catalogued Files,
- Optional requirements from known batch systems states,
- Optional memory, number of CPUs, required etc.,
- Optional storage requirements,
- Optional *Rank* requirements expressions.

Rank requirement expressions provide a way to prioritise the selection of site resources. These expressions are highly coupled with information published by the site’s BDIIs, so a typical rank requirement expression might be, for example, to match against those sites that use a certain batch system (e.g. PBS) or to match against those sites where the expected waiting time before the job will run is low.

Rank requirements are beyond the scope of this chapter and are covered in the gLite Users Guide Sect. 6 in greater detail.

In the example below, we show how easy it is to specify a basic grid job. The grid job will run the executable script “test.sh”. This script will be passed to two arguments. Any errors on stdout while running will be saved to the file “std.err” and stdout will be saved to “std.out”. The job will also return an output file called “ResultsOverview.txt”. In addition, the job will run on any CE where there are at least 10 free CPUs available and where the host name does not match “cern.ch”. The script “test.sh” may be any bash script that one would expect to run on a Worker Node.

```
Executable = "test.sh";
InputSandbox =  {"/home/jodoe/test.sh"};
Arguments = "3.141592653589 data-set-01";
StdOutput = "std.out";
StdError = "std.err";
OutputSandbox = {"ResultsOverview.txt"};
RetryCount = 3;
Rank = {other.GlueCEStateFreeCPUs >= 10};
Requirements  = (!RegExp("cern.ch",other.GlueCEUniqueID));
```

3.1.5 Matching Jobs Against Resources

Before submitting a job to the Grid, the user may wish to see what resources are available. The example below shows, for brevity, a subset of the list of Compute Elements where the grid job can run.

```
$ edg-job-list-match example.jdl

Selected Virtual Organisation name (from proxy certificate
extension): myvo
Connecting to host rb.example.com, port 7772

**********************************************************************
                         COMPUTING ELEMENT IDs LIST
The following CEs matching your job requirements have been found:

                    *CEId*
 ce1.example.com:2119/jobmanager-lcgpbs-long
 ce2.example.com:2119/jobmanager-pbs-myvo
 ce3.example.com:2119/jobmanager-pbs-short
**********************************************************************
```

3.1.6 Submitting a Job

The example below shows how a job may be submitted through a LCG Workload Management System. The output returns a globally unique *Job Identifier* in the form of a URL. This Job ID is then used by subsequent job management commands.

```
$ edg-job-submit example.jdl

Selected Virtual Organisation name (from proxy certificate extens
ion): myvo
Connecting to host rb.example.com, port 7772
Logging to host rb.example.com, port 9002

**********************************************************************
                              JOB SUBMIT OUTCOME
 The job has been successfully submitted to the Network Server.
 Use edg-job-status command to check job current status. Your job
 identifier (edg_jobId) is:

 - https://rb.example.com:9000/qEml-pDbzP2NaPs8u3QTPg
**********************************************************************
```

3.1.7 Managing Groups of Jobs

When submitting and managing multiple jobs, job submission command-line options can be used to force the Job ID to be appended to a chosen file. Corresponding job management commands have corresponding command-line options to read in and process Job IDs from a file. Further details can be found in the gLite Users Guide.

3.1.8 Job States

When a job is submitted to the Grid, its progress is tracked via a series of state changes. Under normal execution the job states change as per the progression listed in Table 2.

Table 2 Job status states

	States	Meaning
1.	Submitted	The job was submitted by the user, but not yet processed by the WMS
2.	Waiting	The job has been accepted by the WMS, but has not yet been assigned to a CE
3.	Ready	The job has been assigned to a CE, but the job data has not yet been transferred
4.	Scheduled	The job is queued on the CE's batch system
5a.	Running	The job is running on the CE
5b.	Aborted	The job could not be run
6.	Done	The job has completed on the CE and WN, and an output sandbox has been transfered to the WMS
7.	Cleared	The job's output sandbox has been transferred to the UI

3.1.9 Querying the Jobs Status

The user may also query the default WMS to print the status of all of their jobs. The example below shows the user querying an LCG WMS; however, the gLite 3.1 equivalent has many more command-line options for parsing/selecting a restricted number of jobs matching particular requirements.

```
$ edg-job-status GlobalJob.id
```

```
$ edg-job-status --all
```

The example output below shows the status of a completed job. Note that the "Current Status" indicates that the output of the job has not yet been retrieved.

```
*************************************************************
BOOKKEEPING INFORMATION:

Status info for the Job : https://rb.example.com:9000/3B2
CA1WB5AEgSrQXhgxMOg
Current Status:     Done (Success)
Exit code:          0
Status Reason:      Job terminated successfully
Destination:        ce3.example.com:2119/jobmanager-pbs-short
reached on:         Thu Oct 18 21:26:41 2007
*************************************************************
```

3.1.10 Job Cancellation

It may become necessary to cancel a grid job, for example the job may have become stuck due to a Worker Node at a site failing, or the job details may have been incorrect (e.g. the parameters may have been in error) at submission.

```
$ edg-job-cancel GlobalJob.id
```

3.1.11 Retrieving Job Outputs

To download the job's output sandbox files (as specified in the JDL) to the UI, the following command should suffice:

```
$ edg-job-get-output GlobalJob.id
```

4 Data Management

Data management support offered by EGEE/gLite is one of its greatest strengths. As we have seen, the WMS helps orchestrate the management of jobs in the Grid by copying a small amount of job data from the UI to the CE. Typically this is in the order of 20 MB maximum. This not a large amount of data by today's standards, but does represent a "copy-by-value" chain rather than a more efficient "copy-by-reference" chain. Hence, this system is not intended to copy large volumes of job-related data around the Grid. For better efficiency, specific data management tools and services, independent of the WMS, are needed to enable

- data procurement,
- data distribution/replication,
- data lifetime management.

These tools must be fast and scalable. We examine firstly the problems in dealing with data in a grid environment, and then show how the EGEE/gLite tools help in solving this problem.

4.1 The Storage Problem

In implementing a distributed storage system, a number of issues and challenges arise:

- How do we support copy-by-reference?
- How do we guarantee uniqueness of the names that refer to the data?
- Although the data location may be unique, tracking this is difficult in a changing environment.
- The data often needs to be moved – e.g. it may need to be close to computation facilities.

4.2 The Storage Solution

To solve these problems EGEE/gLite provides namespace services and tools which can "catalog" files on the Grid. Files copied onto the Grid this way are "registered" into a global logical namespace, i.e. when a file is registered in a catalog, it is assigned a unique global identifier. A registered file can be assigned user-friendly "aliases" and these aliases can be used for most data-related operations. The "LFC" catalog service implements a hierarchically structured namespace, allowing the user to store and access data in a location-independent manner, and just refer to it by name during job submission (copy-by-reference). The introduction of *Storage Resource Manager* (SRM) [56] services at many of the

sites has enabled sites and VOs to support access controls and security on this registered data.

Note that it is possible to work on the Grid without catalogs by using some of the more basic underlying protocols such as gsiftp [57]. However, in doing so, the user discards the many advantages that the file catalogs bring.

4.3 Grid Files

By "file on the Grid" it is generally accepted that a "file" is

- a data file stored physically on an SE and
- registered in a catalog.

There are several ways for the user to get data into grid storage. The data can be copied onto the Grid using standard data transfer commands, e.g. the data is first copied from the user's desktop to the UI using *scp*, then the user invokes the data management commands provided by the "lcg_utils" suite ([6], Sect. 7.5) – see Sect. 4.6 for further details.

4.4 Write Once/Read Many

Grid facilitates the concept of an optimum location of data that will be processed. This often involves replication of data from the orginal location to those locations that provide the fastest access, best bandwidth, most available space, etc. The "lcg_utils" are designed to deal with this. However, it is important to realise that the original and any replicas must remain consistent. A design decision that helps enforce this consistency was to mandate that grid files are "write once, read many". The implication is that files can be created, read and deleted, but not modified. "Modifications" can be achieved by copying to a new file without registering it and then modifying it. The resulting file can then be copied back to an SE and registered. Naturally, this introduces some logistical overhead as new replicas may need to be copied to other SEs.

4.5 File Namespaces

In reality EGEE supports a number of different file namespaces, so a file can be referred to by a number of names. Each namespace has its own advantages and disadvantages. Table 3 summarises the EGEE namespaces. The "lcg_utils" provides a number of commands to convert from one namespace to another. Examples are shown in Sects. 4.7.8 and 4.7.9.

Table 3 EGEE file namespaces

Namespace	Description
GUID	A globally unique idenitifier Assigned when the file is registered in the LFC Format is consistent, but non-memorable or user-friendly e.g. $guid : 38ed3f60 - c402 - 11d7 - a6b0 - f53ee5a37e1d$
LFN	Logical file name Inituitive user-assigned user-friendly name Name is an *Alias* to a grid file and its replicas Alias is set after the file has been registered Stored in a tree-like structure e.g. */grid/myvo/user4/mylfn_alias*
SURL/PFN	Storage uniform resource location or physical file name Identifies the location of a file as seen by the SE's Storage Manager e.g. *srm://se1.example.com/dpm/example.com/home/myvo/generated/2008-04-11/file901b5fe6-a16b-4d73-beda-1f80e5f46b93*
TURL	Transport URL Identifies the physical location on the SE's disk Identifies the protocol supported by the SE e.g. *gsiftp://se1.example.com:/storage/myvo/2008-04-11/file2d2521b5-4d41-444d-9660-4e7a2f3f1b87.106656.0*

4.6 Data Primitives

The "lcg_utils" data management tools offer an extensive number of commands to manipulate files on the Grid, allowing the user to add, remove and replicate files. Files can be copied between the UI, CE, SE and WN (Table 4).

Table 4 Replica management

Command	Action
lcg-cr	Copies a file to an SE, registers the file in the catalogue (upload)
lcg-cp	Copies a catalogued file to a local destination (e.g. UI or Worker Node) (download)
lcg-del	Deletes one catalogued file (either one replica or all replicas)
lcg-rep	Copies a catalogued file between named SEs, then registers the copy
lcg-gt	Gets the TURL for a given SURL and transfer protocol

In addition, these utilities provide a number of commands to manipulate the file catalog. Some of the most common are tabulated in Table 5.

Table 5 Catalog management

Command	Action
lcg-aa	Adds an alias in the catalogue for a given GUID
lcg-ra	Removes an alias in the catalogue for a given GUID
lcg-la	Lists the aliases for a given LFN, GUID or SURL
lcg-rf	Registers in the catalogue a file residing on an SE
lcg-uf	Unregisters in the catalogue a file residing on an SE
lcg-lr	Lists the replicas for a given LFN, GUID or SURL
lcg-lg	Gets the GUID for a given LFN or SURL

Tracking files on the Grid by reference to an absolute location and path is a complex task. The LFC simplifies this by allowing the user to create easy-to-use and user-friendly *aliases* that can be used by the user to refer to catalogued files. The LFC includes a set of commands for managing these aliases within its hierarchical namespace (Table 6).

Table 6 Namespace management

Command	Action
lfc-ls	Lists files in the LFC namespace
lfc-mkdir	Creates a directory in the LFC namspace
lfc-rename	Renames a file or directory
lfc-rm	Removes a file or directory in the namespace
lfc-chmod	Changes the access permissions on an LFC registered file or directory
lfc-chown	Changes ownership of an LFC registered file or directory
lfc-ln	Creates a symbolic link to an LFC registered file or directory
lfc-getacl	Gets the current access controls on a file or directory
lfc-setacl	Gets the current access controls on a file or directory

4.7 Data Management Examples

We present a simple set of examples of how to copy files onto the Grid, add an alias, replicate the file to another site using that alias, etc. In these examples, we assume that the user (user4) is logged onto the Grid and is a member of the *myvo* virtual organisation. In addition, the examples assume that there is a pair of SEs on the Grid: se1.example.com and se2.example.com.

4.7.1 Listing Files in the LFC Hierarchy

The normal convention when using the LFC is that users will store their registered LFNs under the namespace path of */grid/VONAME/USERDIR/*. One may list files and directories in the LFC namespace as follows:

```
$ lfc-ls /grid/myvo/
SAM
desktop
generated
myvouser1
myvouser2
myvouser3
```

4.7.2 Creating a Subdirectory in the LFC Namespace

The user can create a directory where they can store and manage their aliases (their LFNs).

```
$ lfc-mkdir /grid/myvo/myvouser4
$ lfc-ls /grid/myvo/
SAM
desktop
generated
myvouser1
myvouser2
myvouser3
myvouser4
```

4.7.3 Copying a File from the UI to an SE

The user can copy and register a file on the Grid as follows. The unique global ID of the file on the Grid is returned as output.

```
$ lcg-cr --vo myvo -d se1.example.com \
            file:/home/user4/myfile.txt
guid:2e9f6653-f804-4d66-be76-50d1a0d1defb
```

4.7.4 Adding an Alias

To add a user-defined alias:

```
$ lcg-aa --vo myvo \
           guid:2e9f6653-f804-4d66-be76-50d1a0d1defb \
           lfn:/grid/myvo/user4/myfile_alias.txt

$ lfc-ls /grid/myvo/user4/
myfile_alias.txt
```

Now the LFN `lfn:/grid/myvo/user4/myfile_alias.txt` can be used to refer to the file. A listing of the directory shows that the alias is added.

4.7.5 Listing Replicas Using the Alias

The user can list a replica of a given LFN as follows. The output shows the SURL format of the file's location.

```
$ lcg-lr lfn:/grid/myvo/user4/myfile_alias.txt
srm://se1.example.com/dpm/example.com/home/myvo/generated
    /2008-04-11/file901b5fe6-a16b-4d73-beda-1f80e5f46b93
```

4.7.6 Replicating the File

The user can now copy the file to another location on the Grid and then list the replicas. Usually a suitable location for such files can be determined by using the client-side information tools. The "lcg-rep" command produces no output unless an error were to occur.

```
$ lcg-rep -d se2.example.com --vo myvo \
         lfn:/grid/myvo/user4/myfile_alias.txt

$ lcg-lr lfn:/grid/myvo/user4/myfile_alias.txt
srm://se1.example.com/dpm/example.com/home/myvo/generated
     /2008-04-11/file901b5fe6-a16b-4d73-beda-1f80e5f46b93
srm://se2.example.com/dpm/example.com/home/myvo/generated
    /2008-04-11/file5def7494-5fa9-41d1-97cb-a152744f48b2
```

4.7.7 Renaming an Alias

The user may wish to rename an alias, for example, to indicate that the data has been processed. In this simple example, the user renames the alias and then lists the replicas. As can be seen, the replica SURLs have not changed.

```
$ lfc-rename /grid/myvo/user4/myfile_alias.txt \
            /grid/myvo/user4/myfile_new_alias.txt
srm://se1.example.com/dpm/example.com/home/myvo/generated
    /2008-04-11/file901b5fe6-a16b-4d73-beda-1f80e5f46b93
srm://se2.example.com/dpm/example.com/home/myvo/generated
    /2008-04-11/file5def7494-5fa9-41d1-97cb-a152744f48b2
```

4.7.8 Retrieving the GUID

In this example, the user knows the SURL, but would like to determine its GUID. The command also works with LFNs. Users may want to do this, for example, if they decide to remove an alias in the catalog but want to keep track of the file for further processing later.

```
$ lcg-lg --vo myvo srm://se1.example.com/dpm/example.com/home
    /myvo/generated/2008-04-11/file901b5fe6-a16b-4d73-beda-1
    f80e5f46b93
guid:2e9f6653-f804-4d66-be76-50d1a0d1defb
```

4.7.9 Retrieving the TURL

One may wish to determine the TURL for a grid file so that it may be manipulated with lower level data management commands such as *globus-url-copy*. Note that the transport protocol needs to be specified in this command.

```
$ lcg-gt srm://se1.example.com/dpm/example.com/home/myvo/
      generated/2008-04-11/file2d2521b5-4d41-444d-9660-4
      e7a2f3f1b87 gsiftp
gsiftp://gridstore.cs.tcd.ie/gridstore.cs.tcd.ie:/storage/
      cosmo/2008-04-11/file2d2521b5-4d41-444d-9660-4e7a2f3f1b87
      .106656.0
106660
0
```

4.8 Deleting a File

In this example, the user removes a file from the Grid and removes it from the LFC. The example also shows that trying to subsequently manipulate the file generates errors:

```
$ lcg-del --vo myvo \
         -a guid:2e9f6653-f804-4d66-be76-50d1a0d1defb

$ lcg-lr lfn:/grid/myvo/user4/myfile_new_alias.txt
lfc-host.example.com: /grid/myvo/user4/myfile_new_alias.txt:
    No such file or directory
lcg_lr: No such file or directory
```

5 Putting it All Together

Managing multiple grid jobs is often viewed as a difficult process. A single job may be one of many tasks from a much larger and complex workflow. Breaking down this work into individual jobs, gathering input, submitting jobs and gathering the output in the correct order requires time, effort and planning.

Portals, such as P-Grade (which has a workflow editor), can help manage this and the introduction and support of workflows based around Directed Acyclic Graphs [55] in gLite and its JDL can certainly ease this problem.

However, as an aid to understanding the handling of a relatively minor sets of tasks, we encourage the reader to explore the following example that encompasses a complete workflow – breaking a task into smaller tasks, generating a set of JDL

scripts, submitting a set of jobs, waiting for all jobs to finish and then gathering and processing the collection of job outputs. The example was developed by Kathryn Cassidy for a Grid Tutorial Workshop held in Trinity College Dublin in March 2006, and is freely and publicly available.[3] Although the example was developed for the LCG-2 infrastructure, it can be easily modified to use the gLite 3.1 WMS. The example is comprised of two scripts:

1. submit-dictionary.sh – the workflow script executed by the user
2. echoword.sh – the job that is executed on the Worker Node

The pseudo-code algorithm for the workflow is presented as follows:

```
Check that there is a single command-line argument

if number of command-line arguments is not 1
then
 Print Usage Text
 exit with error
else
 Print number of jobs to submit
endif

set values for location of dictonary loccation,
    JDL job file location and the result file.

for index = 1 to value of command-line argument
  Let RAND be random value between 1 and 45427
  Select the word from line number RAND from dictionary
  Create a JDL file with the index and the word set as
      arguments
  Submit the JDL to WMS and save the Job ID to a file
end for

while jobs are still running
 Check all the job status status changes to one of Done,
     Cancelled, Aborted, Cleared
 if all jobs in one of these status then
  All Jobs are finished
  Exit loop
 end if
 Sleep for a polling interval of 30 seconds
end while

Retrieve the output files for all jobs
Concatenate all outputs into a single Results file.
```

[3] http://www.grid.ie/grid-event-2006/march-course/code/dictionary.tgz

6 MPI Support in EGEE

Initially, the support for MPI on the EGEE Grid had been rather weak, whereas MPI was always well supported on the Int.EU.Grid infrastructure. An *EGEE technical working group* discovered [58] that support in the WMS was not flexibile enough, that support for configuring sites for MPI was lacking, that relatively few sites supported it by default, that MPI service discovery was not sufficient due to limitations in the Glue Schema and that many of the sites in EGEE were and are physics-orientated and in some cases did not have any experience in supporting MPI. These issues were not major technical problems, but rather small issues with a big impact on a large MPI user base and their overall experience of the Grid. Indeed, many of the non-HEP communities such as Earth Sciences, Nuclear Fusion, Astrophysics and Computation Chemistry have a major dependence on MPI. These problems have been solved and the solution is gradually being deployed at the EGEE sites (at the time of writing).

6.1 Running MPI on the Grid

In Sect. 3.1.4, we saw a simple example of a grid job expressed using the JDL. The sample code below shows how to grid-enable MPI code using a very similar framework. Further information, including the required wrapper scripts, can be found on the EGEE MPI web page.[4]

```
1 JobType = "MPICH";
2 NodeNumber = 8;
3 Executable = "mpi-start-wrapper.sh";
4 Arguments = "mpi-test OPENMPI";
5 InputSandbox = {"mpi-start-wrapper.sh","mpi-hooks.sh",
6                 "mpi-test.c"};
7 Requirements =
8   Member("MPI-START", other.
        GlueHostApplicationSoftwareRunTimeEnvironment)
9   && Member("OPENMPI", other.
        GlueHostApplicationSoftwareRunTimeEnvironment)
```

Line 1 of the JDL shows that MPI jobs must be submitted with a JobType of *MPICH*. Line 2 sets the number of CPUs or cores required for the job. Most importantly, in line 3, we see that the executable is set to `mpi-start-wrapper.sh`. This script handles the setting up of the MPI environment and the execution of the real MPI executable as determined by the arguments set in Line 4. The wrapper script, the MPI code and a special set of shell functions (job hooks) are passed in with the input files (lines 5 and 6). The role of these hooks are explained below.

[4] http://egee-docs.web.cern.ch/egee-docs/uig/development/uc-mpi-jobs_2.html

Finally, a requirement should be specified (lines 7–9) that selects only sites that have installed the required MPI support scripts and that support the desired version of MPI (OPENMPI in this case). Additional JDL requirements, such as output files, can be added as wished.

6.1.1 mpi-start-wrapper.sh

`mpi-start-wrapper.sh` is the JDL executable and the user's script that aids the execution of the user's MPI code. It is supplemented by an additional call to a site-installed script defined by the variable **$I2G_MPI_START** which sets up the correct MPI environment and performs the actual execution of the MPI code. It is the site manager's responsibility to install this latter script and to define the environment variable properly.

```
# Setup for mpi-start.
export I2G_MPI_APP=$MY_EXECUTABLE
export I2G_MPI_TYPE=$MPI_FLAVOUR
export I2G_MPI_PRE_RUN_HOOK=mpi-hooks.sh
export I2G_MPI_POST_RUN_HOOK=mpi-hooks.sh
# Invoke mpi-start.
$I2G_MPI_START
```

6.1.2 mpi-hooks.sh

The role of the `mpi-hooks.sh` script is to provide a user-controlled way of executing pre-processing tasks before the actual MPI program runs and post-processing tasks after the MPI code has executed. In the example below, the `pre_run_hook` compiles C code using `mpicc`. Indeed, it is recommended that the user applies the `pre_run_hook` so that the MPI code is tailored exactly to the site's runtime environment. Similarly, the `post_run_hook` can be defined to allow the user to process the completed MPI job's output data or to copy and register large files to the Grid, etc.

```
pre_run_hook () {
mpicc -o ${I2G_MPI_APP} ${I2G_MPI_APP}.c
}
```

7 Grid-Secured Applications

The establishment of globally recognised credential services has enabled the scientific community to develop large distributed collaborations. These may take advantage of the underlying PKI infrastructure to grid-secure many applications requiring

authentication and authorisation. A very small sample of these and how they are used is described below:

GridSite: "Grid Security for the Web" [59] is an Apache Web server module that allows a grid user's X.509 credential to be used to interact with a web service. A typical use is to allow an authorised user to add or edit web pages on a remote GridSite-enabled web server, upload files to the web server or permit/deny authorised users access to web pages or services. This is a very widely used enhancement to web sites for grid collaboration.

MySQL: Authentication and authorisation to databases can now be established using X.509 credentials [60].

GSI-OpenSSH: A modified openssh can use a grid user's X.509 certificate or proxy certificate to enable secure shell access to remote computers [61].

8 Conclusion

Grids have become more pervasive as a technological means for solving medium to large-scale scientific problems and enabling distributed collaborations to form, operate and coordinate themselves in an efficient manner. The establishment of globally recognised identity credentials and the move to federated identity management will ease access to grid infrastructures and e-infrastructures.

However, despite the rather noble goals set by many of the grid infrastructures, there remain weak points that still need to be addressed. It is as yet unclear whether Grids meet their collaborational and computational aspirations.

References

1. Carlisle Adams and Steve Lloyd: Understanding PKI: Concepts, Standards, and Deployment Considerations, Addison-Wesley Longman Publishing Co., Inc., 2002
2. Enabling Grids for e-Science, http://www.eu-egee.org
3. Interactive European Grid, http://www.interactive-grid.eu/
4. Grid-Ireland, http://www.grid.ie
5. gLite, http://www.glite.org/
6. Burke, S., et al., The gLite 3.1 User Guide, April 2008, https://edms.cern.ch/document/722398/
7. Foster, I., Kesselman, C., Tuecke, S., 2001, The Anatomy of the Grid - Enabling Scalable Virtual Organizations, Int. J. High Perform. Comput. Appl., 15, 3, Sage Publications, Inc., pp. 200–222
8. Foster, I., Kesselman, C., 2003, The Grid 2 - Blueprint for a New Computing Infrastructure, Morgan Kaufmann Publishers Inc.
9. e-Infrastructure Reflection Group white paper, http://www.e-irg.org/publ/2006-Austrian-eIRG-whitepaper.pdf
10. The Apache Software Foundation, http://www.apache.org/
11. Globus, http://www.globus.org/
12. The Open Science Grid, http://www.opensciencegrid.org/
13. TeraGrid, http://www.teragrid.org/
14. NAREGI, http://www.naregi.org/

15. National Grid Service, http://www.grid-support.ac.uk/
16. Distributed European Infrastructure for Super Computers, http://www.deisa.eu/
17. NorduGrid, http://www.nordugrid.org/
18. Open Grid Forum, http://www.ogf.org/
19. Grid Interoperability Now, https://forge.gridforum.org/sf/projects/gin
20. Overview of the Grid Security Infrastructure, http://www.globus.org/security/overview.html
21. Witzig. C. SLCS and Vash, NREN and Grids Workshop,http://www.terena.org/activities/nrens-n-grids/workshop-06/slides/witzig-switch-slcs-vash.pdf, Malaga, Nov 07
22. The DataGrid Project, http://eu-datagrid.web.cern.ch/
23. The Large Hadron Collider Computing Project, http://lcg.web.cern.ch/LCG/
24. The LHC Experiments, http://public.web.cern.ch/Public/en/LHC/LHC-en.html
25. Andronico, G., Barbera, R., Falzone, A., Lo Re, G., Pulvirenti, A., Rodolico, A., 2003, The GENIUS web portal: grid computing made easy, ITCC '03: Proceedings of the International Conference on Information Technology: Computers and Communications, IEEE Computer Society, pp. 425–431
26. EnginFrame Grid Portal, http://www.enginframe.com/
27. Migrating Desktop, http://desktop.psnc.pl/
28. Sipos, G. Kacsuk, P. 2006, Multi-Grid, Multi-User Workflows in the P-GRADE Portal. J. Grid Comput., 3, 3–4, Kluwer Academic Publishers, pp. 221–238
29. GridSphere Portal Framework, http://www.gridsphere.org/
30. The Gilda Testbed, https://gilda.ct.infn.it/
31. OpenPBS Batch System, http://www.openpbs.org
32. Torque and Maui Batch System, http://www.clusterresources.com/
33. LSF Batch System, http://www.platform.com/
34. The Condor Project, http://www.cs.wisc.edu/condor/
35. Sun Grid Engine, http://gridengine.sunsource.net/
36. OGF Glue Schema version 1.3, http://forge.gridforum.org/sf/go/doc14185
37. MyProxy Credential Management Service, http://grid.ncsa.uiuc.edu/myproxy/
38. Carter, G., LDAP System Administration, O'Reilly Media Inc., 2003
39. Cooke, A.W., et al. The Relational Grid Monitoring Architecture: Mediating Information about the Grid, J. Grid Comput., Vol 2 No. 4, Kluwer Academic Publishers, pp. 323–339
40. VOMS Core User and Reference Guide, https://edms.cern.ch/file/571991/1/voms-guide.pdf
41. Workload Management System User and Reference Guide, https://edms.cern.ch/file/572489/1/EGEE-JRA1-TEC-572489-WMS-guide-v0-2.pdf
42. Logging and Bookkeeping User and Reference Guide,https://edms.cern.ch/file/571273/1/LB-guide.pdf
43. LCG File Catalog, https://twiki.cern.ch/twiki/bin/view/LCG/LfcGeneralDescription
44. FTS Users Guide, https://edms.cern.ch/file/591792/1/EGEE-TECH-591792-Transfer-Java-v1.0.pdf
45. AMGA Homepage, http://amga.web.cern.ch/amga/
46. Koblitz, B., Santos. N., The AMGA User's and Administrators manual, http://project-arda-dev.web.cern.ch/project-arda-dev/metadata/downloads/amga-manual_1_2_3.pdf
47. Global Grid User Support, http://www.ggus.org/
48. Grid Operations Centre DataBase, https://goc.gridops.org/
49. Service Availability Monitoring, https://lcg-sam.cern.ch:8443/sam/sam.py
50. Core Infrastructure Service, http://cic.gridops.org
51. The EGEE Training Digital Library, http://egee.lib.ed.ac.uk:8080/
52. The EGEE NA3 Homepage, http://www.egee.nesc.ac.uk/
53. ELGrid – an adaptive e-Learning tool for Grid education, http://www.grid.ie/elgrid/
54. JDL Attribute Specification (submission via WMS Network Server), https://edms.cern.ch/file/555796/1/EGEE-JRA1-TEC-555796-JDL-Attributes-v0-8.pdf

55. JDL Attributes Specification (submission via WMS WMProxy), https://edms.cern.ch/file/571273/1/LB-guide.pdf
56. The Storage Resource Manager, http://sdm.lbl.gov/srm-wg/
57. GSIFTP Tools for the Data Grid , http://www.globus.org/toolkit/docs/2.4/datagrid/deliverables/gsiftp-tools.html
58. Childs, S. The MPI Working Group Report, http://egee-intranet.web.cern.ch/egee-intranet/NA1/TCG/wgs/EGEE-II-MPI-WG-TEC-2.pdf
59. GridSite, http://www.gridsite.org/
60. MySQL, Using Secure Connections, http://dev.mysql.com/doc/refman/6.0/en/secure-connections.html
61. GSI-Enabled OpenSSH, http://grid.ncsa.uiuc.edu/ssh/

http://www.springer.com/978-3-642-03369-8

Jets From Young Stars V
High Performance Computing and Applications
Gracia, J.; de Colle, F.; Downes, T. (Eds.)
2009, XI, 227 p. 88 illus., Hardcover
ISBN: 978-3-642-03369-8