European DataGrid Project: Experiences of Deploying a Large Scale Testbed for E-science Applications

Fabrizio Gagliardi[1], Bob Jones[1], Mario Reale[2], and Stephen Burke[3]

On behalf of the EU DataGrid Project

[1] CERN, European Particle Physics Laboratory,
CH-1211 Geneve 23, Switzerland
{Fabrizio.Gagliardi, Bob.Jones}@cern.ch
http://www.cern.ch

[2] INFN CNAF, Viale Berti-Pichat 6/2,
I-40127 Bologna, Italy
mario.reale@cnaf.infn.it

[3] Rutherford Appleton Laboratory,
Chilton, Didcot, Oxon, UK
s.burke@rl.ac.uk

Abstract. The objective of the European DataGrid (EDG) project is to assist the next generation of scientific exploration, which requires intensive computation and analysis of shared large-scale datasets, from hundreds of terabytes to petabytes, across widely distributed scientific communities. We see these requirements emerging in many scientific disciplines, including physics, biology, and earth sciences. Such sharing is made complicated by the distributed nature of the resources to be used, the distributed nature of the research communities, the size of the datasets and the limited network bandwidth available. To address these problems we are building on emerging computational Grid technologies to establish a research network that is developing the technology components essential for the implementation of a world-wide data and computational Grid on a scale not previously attempted. An essential part of this project is the phased development and deployment of a large-scale Grid testbed.

The primary goals of the first phase of the EDG testbed were: 1) to demonstrate that the EDG software components could be integrated into a production-quality computational Grid; 2) to allow the middleware developers to evaluate the design and performance of their software; 3) to expose the technology to end-users to give them hands-on experience; and 4) to facilitate interaction and feedback between end-users and developers. This first testbed deployment was achieved towards the end of 2001 and assessed during the successful European Union review of the project on March 1, 2002. In this article we give an overview of the current status and plans of the EDG project and describe the distributed testbed.

M.C. Calzarossa and S. Tucci (Eds.): Performance 2002, LNCS 2459, pp. 480–499, 2002.

1 Introduction

Advances in distributed computing, high quality networks and powerful and cost-effective commodity-based computing have given rise to the Grid computing paradigm [6]. In the academic world, a major driver for Grid development is collaborative science mediated by the use of computing technology, often referred to as e-science. While scientists of many disciplines have been using computing technology for decades (almost pre-dating computing science itself), e-Science projects present fresh challenges for a number of reasons, such as the difficulty of co-ordinating the use of widely distributed resources owned and controlled by many organisations. The Grid introduces the concept of the Virtual Organisation (VO) as a group of both users and computing resources from a number of real organisations which is brought together to work on a particular project.

The EU DataGrid (EDG) is a project funded by the European Union with €9.8 M through the Framework V IST R&D programme (see www.eu-datagrid.org). There are 21 partner organisations from 15 EU countries, with a total participation of over 200 people, for a period of three years starting in January 2001. The objectives of the project are to support advanced scientific research within a Grid environment, offering capabilities for intensive computation and analysis of shared large-scale datasets, from hundreds of terabytes to petabytes, across widely distributed scientific communities. Such requirements are emerging in many scientific disciplines, including particle physics, biology, and earth sciences.

The EDG project has now reached its mid-point, since the project started on January 1st 2001 and the foreseen end of the project is on December 31st 2003. At this stage, very encouraging results have already been achieved in terms of the major goals of the project, which are the demonstration of the practical use of computational and data Grids for wide and extended use by the high energy physics, bio-informatics and earth observation communities.

A production quality testbed has been set up and implemented at a number of EDG sites, while a separate development testbed addresses the need for rapid testing and prototyping of the EDG middleware. The EDG production testbed consists currently of ten sites, spread around Europe: at CERN (Geneva), INFN-CNAF (Bologna), CC-IN2P3 (Lyon), NIKHEF (Amsterdam), INFN-TO (Torino), INFN-CT (Catania), INFN-PD (Padova), ESA-ESRIN (Frascati), Imperial College (London), and RAL (Oxfordshire). The EDG development testbed currently consists of four sites : CERN, INFN-CNAF, NIKHEF, and RAL. The reference site for the EDG collaboration is at CERN, where, before any official version of the EDG middleware is released, the initial testing of the software is performed and the main functionalities are proven, before distribution to the other development testbed sites.

The EDG collaboration is currently providing free, open source software based on the Linux Red Hat 6.2 platform. A range of standard machine profiles is supported (Computing Element, Storage Element, User Interface, Resource Broker, Worker Node, Network Monitoring Node). The testbed provides a set of common shared

services available to all certified users with valid X.509 PKI certificates issued by a Certificate Authority trusted by EDG. A set of tools is provided to handle the automatic update of the grid-map files on all hosts belonging to the testbed sites which allow users to be authorised to use the resources. A number of VOs have been defined for the various research groups involved in the project. Each VO has an authorisation server (using LDAP technology) to define its members, and a Replica Catalogue to store the location of its files.

In this article, besides this short introduction about the history of the project and its current status, we give an overview of the technology developed by the project so far. This is a concrete illustration of the level of maturity reached by Grid technologies to address the task of high throughput computing for distributed Virtual Organisations. The work of the project is divided into functional areas: workload management, data management, grid monitoring and information systems, fabric management, mass data storage, testbed operation, and network monitoring. In addition there are user communities drawn from high energy physics, earth observation and biomedical applications. In November-December 2001 the first testbed was set up at CERN, merging and collecting the development work performed by the various middleware developers, and the first release of the EDG software was deployed and successfully validated. The project has been congratulated "for exceeding expectations" by the European Union reviewers on March 1st, 2002, during the first official EU review .

2 The European Data Grid Middleware Architecture

The EDG architecture is based on the Grid architecture proposed by Ian Foster and Carl Kesselman [6], with a reduced number of implemented services.

Sixteen services have been implemented by the middleware developers, based on original coding for some services and on the usage of the Globus 2 toolkit (see www.globus.org) for basic Grid infrastructure services: authentication (GSI), secure file transfer (GridFTP), information systems (MDS), job submission (GRAM) and the Globus Replica Catalogue. In addition the job submission system uses software from the Condor-G project [8]. The middleware also relies on general open source software such as OpenLDAP.

The middleware development is divided into six functional areas: workload management, data management, Grid Monitoring and Information Systems, fabric management, mass data storage, and network monitoring. A sketch of the essential EDG architecture is shown in Figure 1 [1], where the relationship between the Operating System, Globus tools, the EDG middleware and the applications is shown. The EDG architecture is therefore a multi-layered architecture. At the lowest level is the operating system. Globus provides the basic services for secure and authenticated use of both operating system and network connections to safely transfer files and data and allow interoperation of distributed services. The EDG middleware uses the Globus services, and interfaces to the highest layer, the user applications running on the Grid.

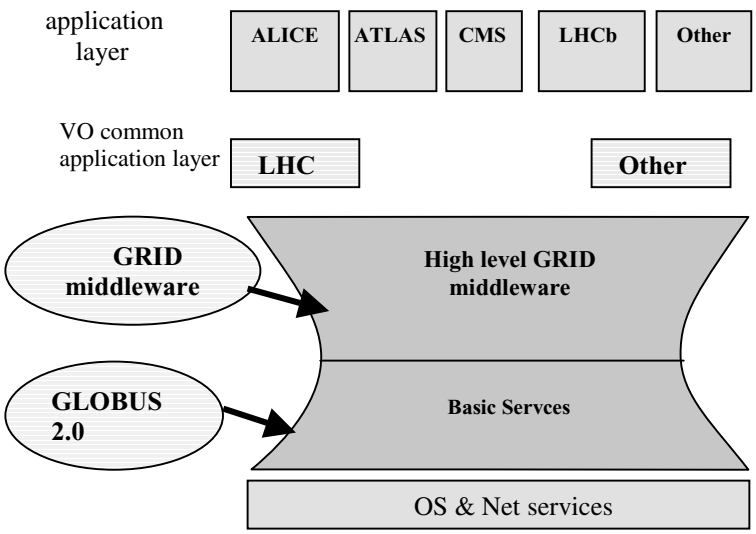

Fig. 1. The schematic layered EDG architecture: the Globus hourglass

The multi-layered EDG Grid architecture is shown in Figure 2 and Figure 3, which show the different layers from bottom to top, namely: the Fabric layer, the underlying Grid Services, the Collective Services, the Grid Application layer and, at the top, a local application accessing a remote client machine. Figure 3 groups together and identifies the main EDG services. At the top of the whole system, the local application and the local database represent the end user machine, which executes an application requesting Grid services, either submitting a Grid job or requesting a file through the interfaces to the list of files stored on the Grid and published in a Replica Catalogue.

2.1 Workload Management System (WMS)

The goal of the Workload Management System is to implement an architecture for distributed scheduling and resource management in a Grid environment. It provides to the Grid users a set of tools to submit their jobs, have them executed on the distributed Computing Elements (a Grid resource mapped to an underlying batch system), get information about their status, retrieve their output, and allow them to access Grid resources in an optimal way (optimizing CPU usage, reducing file transfer time and cost, and balancing access to resources between users). It deals with the Job Manager of the Grid Application layer and the Grid Scheduler in the Collective Services layer. A functional view of the whole WMS system is represented in figure 4.

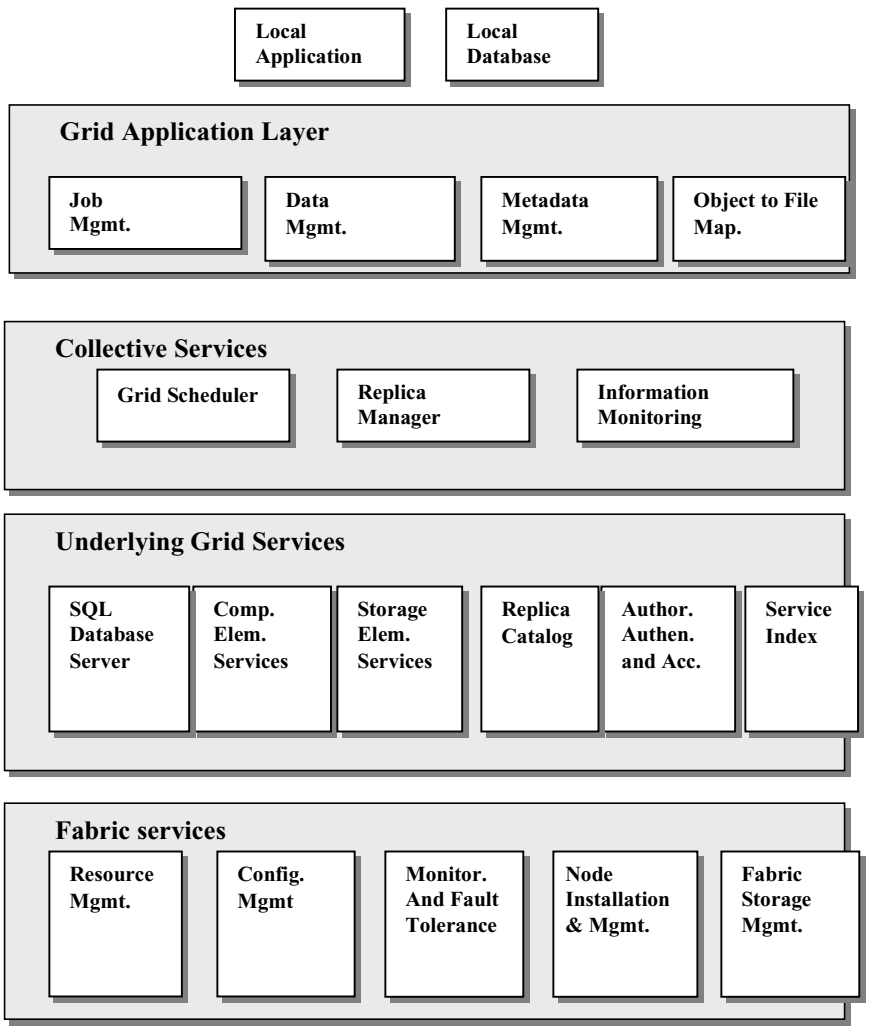

Fig. 2. The detailed multi layered EDG GRID architecture

The WMS is currently composed of the following parts:

– **User Interface** (UI): The access point for the Grid user. A job is defined using the JDL language (see below), which specifies the input data files, the code to execute, the required software environment, and lists of input and output files to be transferred with the job. The user can also control the way in which the broker chooses the best-matching resource. The job is submitted to the Resource Broker using a command line interface or a programmatic API; there are also several groups developing graphical interfaces.

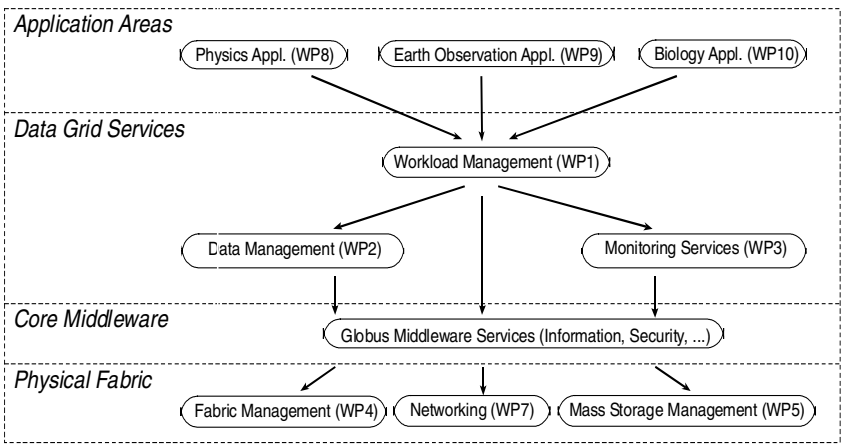

Fig. 3. The EDG service architecture

- **Resource Broker** (RB): This performs match-making between the requirements of a job and the available resources, and attempts to schedule the jobs in an optimal way, taking into account the data location and the requirements specified by the user. The information about available resources is read dynamically from the Information and Monitoring System. The scheduling and match-making algorithms used by the RB are the key to making efficient use of Grid resources. In performing the match-making the RB queries the Replica Catalogue, which is a service used to resolve logical file names (LFN, the generic name of a file) into physical file names (PFN, which gives the physical location and name of a particular file replica). The job can then be sent to the site which minimises the cost of network bandwidth to access the files.

- **Job Submission System** (JSS): This is a wrapper for Condor-G [8], interfacing the Grid to a Local Resource Management System (LRMS), usually a batch system like PBS, LSF or BQS. Condor-G is a Condor-Globus joint project, which combines the inter-domain resource management protocols of the Globus Toolkit with the intra-domain resource and job management methods of Condor to allow high throughput computing in multi-domain environments.

- **Information Index** (II): This is a Globus MDS index which collects information from the Globus GRIS information servers running on the various Grid resources, published using LDAP, and read by the RB to perform the match-making. Information items are both static (installed software, number of available CPUs etc) and dynamic (total number of running jobs, current available disk space etc). The information is cached for a short period to improve performance.

- **Logging and Bookkeeping** (LB): The Logging and Bookkeeping service stores a variety of information about the status and history of submitted jobs using a MySQL database.

Job Description Language (JDL)
The JDL allows the various components of the Grid Scheduler to communicate requirements concerning the job execution. Examples of such requirements are:

- Specification of the executable program or script to be run and arguments to be passed to it, and files to be used for the standard input, output and error streams.
- Specification of files that should be shipped with the job via Input and Output Sandboxes.
- A list of input files and the access protocols the job is prepared to use to read them.
- Specification of the Replica Catalogue to be searched for physical instances of the requested input files.
- Requirements on the computing environment (OS, memory, free disk space, software environment etc) in which the job will run.
- Expected resource consumption (CPU time, output file sizes etc).
- A ranking expression used to decide between resources which match the other requirements.

The *classified advertisements* (*ClassAds*) language defined by the Condor project has been adopted for the Job Description Language because it has all the required properties.

In order for a user to have a job correctly executed on a worker node of an available Computing Element, the user's credentials have to be transmitted by the creation of a proxy certificate. A user issues a grid-proxy-init command on a user interface machine to create an X.509 PKI proxy certificate using their locally stored private key. An authentication request containing the proxy public and private keys and the user's public key is sent to a server; the server gets the request and creates a coded message by means of the user's public key, sending it back to the user process on the User Interface machine. This message is decoded by means of the user's private key and sent back again to the server (in this case normally the Resource Broker). When the server gets the correctly decoded message it can be sure about the user's identity, so that an authenticated channel can be established and the user credentials can be delegated to the broker.

Users use Condor ClassAds-like statements inside a JDL (Job Description Language) file to describe the job they want to be executed by the Grid. This includes a list of input data residing on Storage Elements (Grid-enabled disk or tape storage), and places requirements on the features of the compute nodes on which the job will execute. These can be chosen from the set of information defined by the schema used by the information system, and includes such things as operating system version, CPU speed, available memory etc.

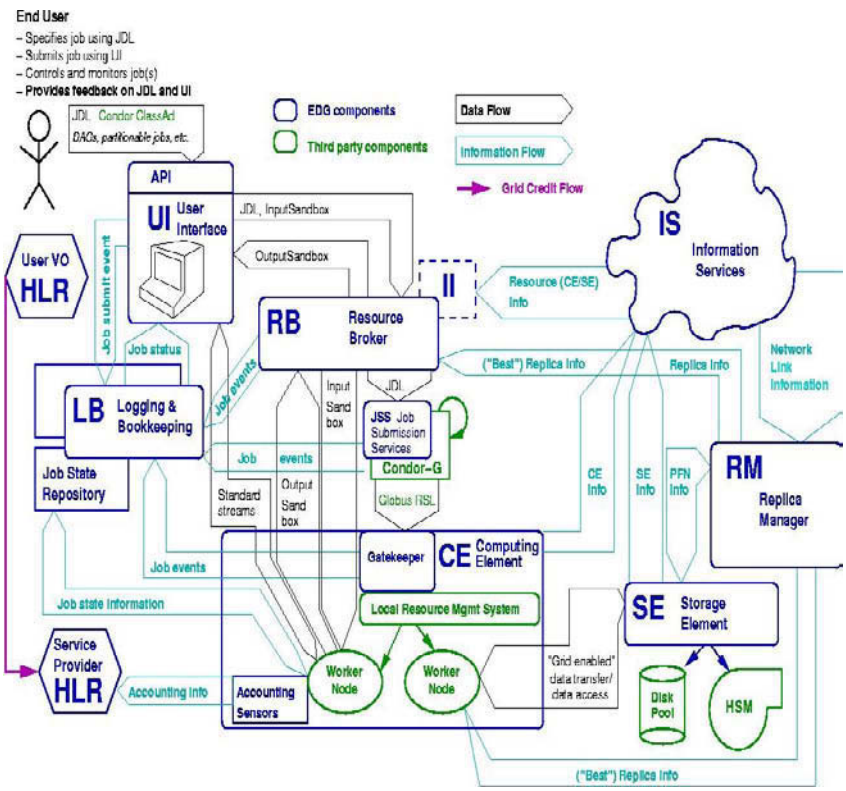

Fig. 4. The Workload Management System and its components: interaction with other EDG elements. The future component HLR (Home Location Register) is also shown.

Users can define an Input Sandbox, which is a set of files transferred to a Worker Node by means of GridFTP by the Resource Broker, so that any file required for the job to be executed (including the executable itself if necessary) can be sent to the local disk of the machine where the job will run. Similarly, the user can specify an Output Sandbox, which is a set of files to be retrieved from the Worker Node after the job finishes (other files are deleted). The files in the Output Sandbox are stored on the RB node until the user requests them to be transferred back to a UI machine.

The JDL can also specify a particular required software environment using a set of user-defined strings to identify particular features of the run-time environment (for example, locally installed application software).

A special file, called the BrokerInfo file, is created by the Resource Broker to enable a running job to be aware of the choices made in the matchmaking, in particular about the Storage Element(s) local to the chosen Computing Element, and the way to access the requested input files. The BrokerInfo file is transferred to the Worker Node along with the Input Sandbox, and can be read directly or with an API or command-line tools.

Users have at their disposal a set of commands to handle jobs by means of a command line interface installed on a User Interface machine, on which they have a normal login account and have installed their X509 certificate. They can submit a job, query its status, get logging information about the job history, cancel a job, be notified via email of the job's execution, and retrieve the job output. When a job is submitted to the system the user gets back a Grid-wide unique handle by means of which the job can be identified in other commands.

2.2 Data Management System (DMS)

The goal of the Data Management System is to specify, develop, integrate and test tools and middleware to coherently manage and share petabyte-scale information volumes in high-throughput production-quality grid environments. The emphasis is on automation, ease of use, scalability, uniformity, transparency and heterogeneity. The DMS will make it possible to securely access massive amounts of data in a universal global name space, to move and replicate data at high speed from one geographical site to another, and to manage synchronisation of distributed replicas of files or databases. Generic interfaces to heterogeneous mass storage management systems will enable seamless and efficient integration of distributed resources. The main components of the EDG Data Management System, currently provided or in development, are as follows:

- **Replica Manager**: This is still under development, but it will manage the creation of file replicas by copying from one Storage Element to another, optimising the use of network bandwidth. It will interface with the Replica Catalogue service to allow Grid users to keep track of the locations of their files.

- **Replica Catalogue**: This is a Grid service used to resolve Logical File Names into a set of corresponding Physical File Names which locate each replica of a file. This provides a Grid-wide file catalogue for the members of a given Virtual Organisation.

- **GDMP**: The GRID Data Mirroring Package is used to automatically mirror file replicas from one Storage Element to a set of other subscribed sites. It is also currently used as a prototype of the general Replica Manager service.

- **Spitfire**: This provides a Grid-enabled interface for access to relational databases. This will be used within the data managementmiddleware to implement the Replica Catalogue, but is also available for general use.

The Replica Manager
The EDG Replica Manager will allow users and running jobs to make copies of files between different Storage Elements, simultaneously updating the Replica Catalogue, and to optimise the creation of file replicas by using network performance information and cost functions, according to the file location and size. It will be a distributed system, i.e. different instances of the Replica Manager will be running on different sites, and will be synchronised to local Replica Catalogues, which will be

interconnected by the Replica Location Index. The Replica Manager functionality will be available both with APIs available to running applications and by a command line interface available to users. The Replica Manager is responsible for computing the cost estimates for replica creation. Information for cost estimates, such as network bandwidth, staging times and Storage Element load indicators, will be gathered from the Grid Information and Monitoring System.

The Replica Catalogue

The Replica Catalogue has as a primary goal the resolution of Logical File Names into Physical File Names, to allow the location of the physical file(s) which can be accessed most efficiently by a job. It is currently implemented using Globus software by means of a single LDAP server running on a dedicated machine. In future it will be implemented by a distributed system with a local catalogue on each Storage Element and a system of Replica Location Indices to aggregate the information from many sites. In order to achieve maximum flexibility the transport protocol, query mechanism, and database backend technology will be decoupled, allowing the implementation of a Replica Catalogue server using multiple database technologies (such as RDBMSs, LDAP-based databases, or flat files). APIs and protocols between client and server are required, and will be provided in future releases of the EDG middleware. The use of mechanisms specific to a particular database is excluded. Also the query technology will not be tied to a particular protocol, such as SQL or LDAP. The use of GSI-enabled HTTPS for transport and XML for input/output data representation is foreseen. Both HTTPS and XML are the most widely used industry standards for this type of system.

The Replica Manager, Grid users and Grid services like the scheduler (WMS) can access the Replica Catalogue information via APIs. The WMS makes a query to the RC in the first part of the matchmaking process, in which a target computing element for the execution of a job is chosen according to the accessibility of a Storage Element containing the required input files. To do so, the WMS has to convert logical file names into physical file names. Both logical and physical files can carry additional metadata in the form of "attributes". Logical file attributes may include items such as file size, CRC check sum, file type and file creation timestamps.

A centralised Replica Catalogue was chosen for initial deployment, this being the simplest implementation. The Globus Replica Catalogue, based on LDAP directories, has been used in the testbed so far. One dedicated LDAP server is assigned to each Virtual Organisation; four of these reside on a server machine at NIKHEF, two at CNAF, and one at CERN. Users interact with the Replica Catalogue mainly via the previously discussed Replica Catalogue and BrokerInfo APIs.

GDMP

The GDMP client-server software system is a generic file replication tool that replicates files securely and efficiently from one site to another in a Data Grid environment using several Globus Grid tools. In addition, it manages replica catalogue entries for file replicas, and thus maintains a consistent view of names and locations of replicated files. Any file format can be supported for file transfer using

plugins for pre- and post-processing, and for Objectivity database files a plugin is supplied.

GDMP allows mirroring of uncatalogued user data between Storage Elements. Registration of user data into the Replica Catalogue is also possible via the Replica Catalogue API. The basic concept is that client SEs subscribe to a source SE in which they have interest. The clients will then be notified of new files entered in the catalogue of the subscribed server, and can then make copies of required files, automatically updating the Replica Catalogue if necessary.

Spitfire
Spitfire is a secure, Grid-enabled interface to a relational database. Spitfire provides secure query access to remote databases through the Grid using Globus GSI authentication.

2.3 Grid Monitoring and Information Systems

The EDG Information Systems middleware implements a complete infrastructure to enable end-user and administrator access to status and error information in the Grid environment, and provides an environment in which application monitoring can be carried out. This permits job performance optimisation as well as allowing for problem tracing, and is crucial to facilitating high performance Grid computing. The goal is to provide easy access to current and archived information about the Grid itself (information about resources - Computing Elements, Storage Elements and the Network), for which the Globus MDS is a common solution, about job status (e.g. as implemented by the WMS Logging and Bookkeeping service) and about user applications running on the Grid, e.g. for performance monitoring. The main components are as follows:

- **MDS:** MDS is the Globus Monitoring and Discovery Service, based on soft-state registration protocols and LDAP aggregate directory services. Each resource runs a GRIS (Grid Resource Information Server) publishing local information as an LDAP directory. These servers are in turn registered to a hierarchy of GIISs (Grid Information Index Servers), which aggregate the information and again publish it as an LDAP directory.

- **Ftree**: Ftree is an EDG-developed alternative to the Globus LDAP backend with improved caching over the code in the Globus 1 toolkit.

- **R-GMA:** R-GMA is a relational GMA (Grid Monitoring Architecture) implementation which makes information from producers available to consumers as relations (tables). It also uses relations to handle the registration of producers. R-GMA is consistent with GMA principles.

- **GRM/PROVE:** GRM/Prove is an application monitoring and visualisation tool of the P-GRADE graphical parallel programming environment, modified for application monitoring in the DataGrid environment. The instrumentation library

of GRM is generalised for a flexible trace event specification. The components of GRM will be connected to R-GMA using its Producer and Consumer APIs.

A number of alternatives, MDS, Ftree and R-GMA, are being considered as the basis of the final EDG information service. These implementations are being evaluated and compared using a set of performance, scalability and reliability criteria to determine which is the most suitable for deployment.

In the current testbed all relevant Grid elements run a GRIS, which carries the information for that element to an MDS GIIS where the information is collected, to be queried by the Resource Broker and other Grid servers.

2.4 EDG Fabric Installation and Job Management Tools

The EDG collaboration has developed a complete set of tools for the management of PC farms (fabrics), in order to make the installation and configuration of the various nodes automatic and easy for the site managers managing a testbed site, and for the control of jobs on the Worker Nodes in the fabric. The main tasks are:

User Job Control and Management (Grid and local jobs) on fabric batch and/or interactive CPU services. There are two branches:

- The **Gridification** subsystem provides the interface from the Grid to the resources available inside a fabric for batch and interactive CPU services. It provides the interface for job submission/control and information publication to the Grid services. It also provides functionality for local authentication and policy-based authorisation, and mapping of Grid credentials to local credentials.

- The **Resource Management** subsystem is a layer on top of the batch and interactive services (LRMS). While the Grid Resource Broker manages workload distribution between fabrics, the Resource Management subsystem manages the workload distribution and resource sharing of all batch and interactive services inside a fabric, according to defined policies and user quota allocations.

Automated System Administration for the automatic installation and configuration of computing nodes. These three subsystems are designed for the use of system administrators and operators to perform system installation, configuration and maintenance:

- **Configuration Management** provides the components to manage and store centrally all fabric configuration information. This includes the configuration of all EDG subsystems as well as information about the fabric hardware, systems and services.

- **Installation Management** handles the initial installation of computing fabric nodes. It also handles software distribution, configuration and maintenance according to information stored in the Configuration Management subsystem.

– **Fabric Monitoring and Fault Tolerance** provides the necessary components for gathering, storing and retrieving performance, functional, setup and environmental data for all fabric elements. It also provides the means to correlate that data and execute corrective actions if problems are identified.

The fabric installation and configuration management tools are based on a remote install and configuration tool called LCFG (Local Configurator), which, by means of a server, installs and configures remote clients, starting from scratch, using a network connection to download the required RPM files for the installation, after using a disk to load a boot kernel on the client machines.

The basic architectural structure and function of LCFG are represented in Figure 5 and are as follows: abstract configuration parameters are stored in a central repository located in the LCFG server. Scripts on the host machine (LCFG client) read these configuration parameters and either generate traditional configuration files, or directly manipulate various services. A daemon in the LCFG server (mkxprof) polls for changes in the source files and converts them into XML profiles, one profile per client node. The XML profiles are then published on a web server. LCFG clients can be configured to poll at regular intervals, or to receive automatic change notifications, or they can fetch new profiles in response to an explicit command. A daemon in each LCFG client (rdxprof) then reads its associated XML profile from the web server and caches it locally (DBM file). LCFG scripts access the local cache to extract the configuration values and execute changes accordingly.

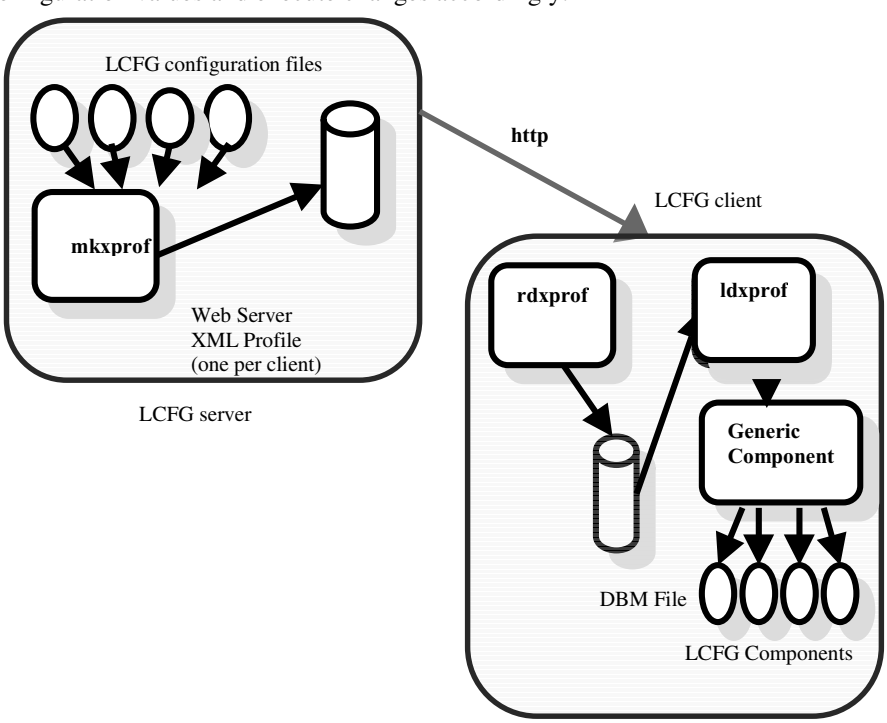

Fig. 5. LCFG internal operation

2.5 The Storage Element

The Storage Element has an important role in the storage of data and the management of files in the Grid domain, and EDG is working on its definition, design, software development, setup and testing.

A Storage Element is a complete Grid-enabled interface to a Mass Storage Management System, tape or disk based, so that mass storage of files can be almost completely transparent to Grid users. A user should not need to know anything about the particular storage system available locally to a given Grid resource, and should only be required to request that files should be read or written using a common interface. All existing mass storage systems used at testbed sites will be interfaced to the Grid, so that their use will be completely transparent and the authorisation of users to use the system will be in terms of general quantities like space used or storage duration.

The procedures for accessing files are still in the development phase. The main achievements to date have been the definition of the architecture and design for the Storage Element, collaboration with Globus on GridFTP/RFIO access, collaboration with PPDG on a control API, staging from and to the CASTOR tape system at CERN, and an interface to GDMP. Initially the supported storage interfaces will be UNIX disk systems, HPSS (High Performance Storage System), CASTOR (through RFIO), and remote access via the Globus GridFTP protocol. Local file access within a site will also be available using Unix file access, e.g. with NFS or AFS. EDG are also developing a grid-aware Unix filing system with ownership and access control based on Grid certificates rather than local Unix accounts.

3 The EDG Testbed

EDG has deployed the middleware on a distributed testbed, which also provides some shared services. A central software repository provides defined bundles of RPMs according to machine type, together with LCFG scripts to install and configure the software.

There are also automatic tools for the creation and update of grid-map files (used to map Grid certificates to local Unix accounts), needed by all testbed sites to authorise users to access the testbed resources. A new user subscribes to the EDG Acceptable Usage Policy by using their certificate, loaded into a web browser, to digitally sign their agreement. Their certificate Subject Name is then added to an LDAP server maintained for each Virtual Organisation by a VO administrator. Each site can use this information, together with local policy on which VOs are supported, to generate the local map file which authorises the user at that site. This mechanism is sketched in 6.

The testbed sites each implement a User Interface machine, a Gatekeeper and a set of Worker Nodes (i.e. a Grid Computing Element), managed by means of a Local

Resource Management System, and a Storage Element (disk only at most sites, but with tape storage at CERN, Lyon and RAL). Some sites have also set up a local Resource Broker. As a reference, Fig. 7 shows a typical site setup in terms of machine composition, for both development and production testbeds, namely the current CERN testbed, with production, development and service machines (network time server, NFS server, LCFG server, monitoring servers).

New sites are welcome to join the testbed, with a well-defined set of rules and procedures. In addition the EDG middleware is freely available, documented, and downloadable from a central repository (the exact license conditions are still under review, but will allow free use). All required RPMs are available for download to an LCFG server, which is generally the first element to be set up, by means of which all other components can be easily installed and configured following the EDG documentation. Alternatively it is possible to install and configure the software by hand. Only Red Hat Linux version 6.2 is currently supported, but more platforms will be supported in due course.

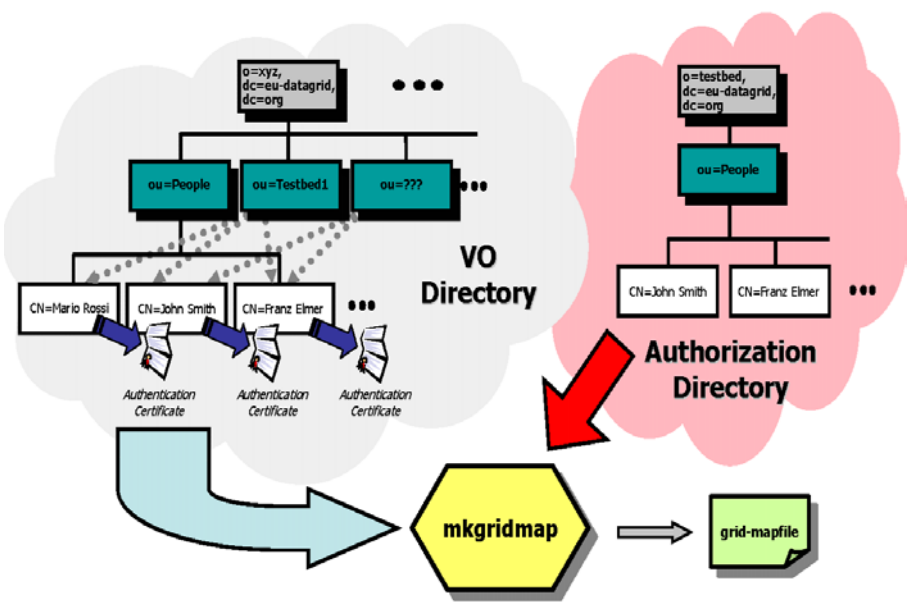

Fig. 6. The operation of the makegridmap daemon

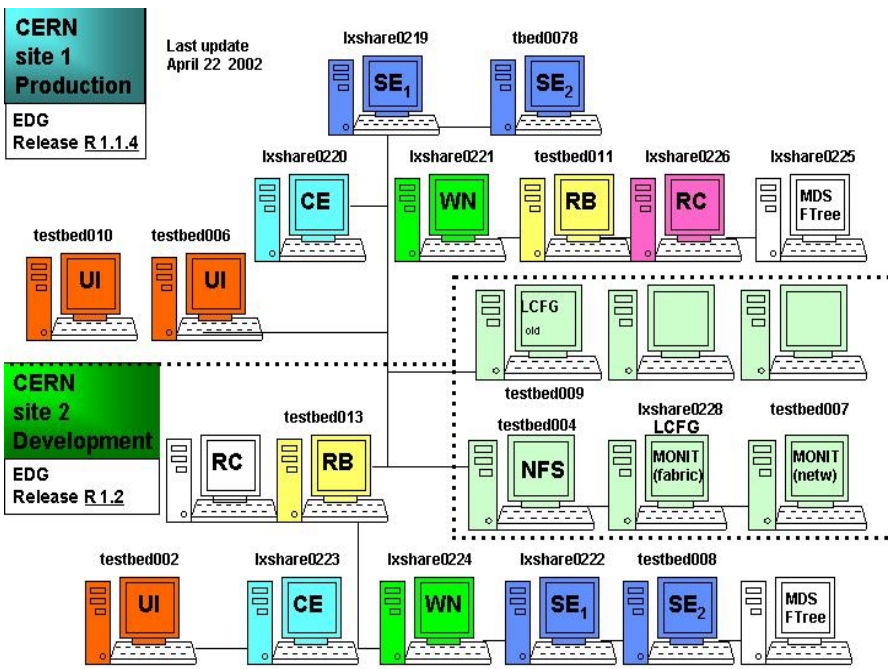

Fig. 7. The CERN testbed cluster composition

4 Future Developments

There are a number of important services still in development. The WMS will introduce an implementation for billing and accounting, advance reservation of resources, job partitioning and checkpointing. In the data management area there will be APIs for Replica Selection through a Replica Optimiser. The first full implementation of R-GMA will be used and compared with the existing information systems (MDS and Ftree). For fabric management there will be the LCAS (Local Credentials and Authorization Server) and further tools to replace the use of Grid map files to authorise users and introduce effective differentiated authorisation of users according to the Virtual Organisation they belong to (VOMS, Virtual Organisation Membership Server). Also a new high level configuration description language will be provided. Interfaces to Storage Elements will continue to be developed. Network monitoring information will be used to influence the decisions of the Resource Broker.

5 Conclusions

The European Data Grid project has already achieved many of its goals, stated at the time of the project conception two years ago. A production quality distributed computing environment has been demonstrated by the EDG testbed. It will now be enriched in functionality, further improved in reliability and extended both geographically and in terms of aggregate CPU power and storage capacity. The community of users has already successfully validated the use of a large set of applications, ranging from High Energy Physics to Bio-Informatics and Earth Observation [2, 3, 4]. At the same time development is currently ongoing to extend the range of functionality covered by the EDG middleware.

All work packages have defined an intense schedule of new research and development, which will be supported by the progressive introduction of high-speed scientific networks such as those deployed by RN GEANT. This will increase the range of possibilities available to the EDG developers. As an example, EDG has proposed the introduction of a Network Optimiser server, to establish in which cases it is preferable to access a file from a remote location or to trigger local copying, according to network conditions in the end-to-end link between the relevant sites. The development of Differentiated Services and Packet Forwarding policies is strongly encouraged, in order to make Grid applications cope better with the dynamic network performance and create different classes of services to be provided to different classes of applications, according to their requirements in terms of bandwidth, throughput, delay, jitter etc.

The impact of the new Globus features foreseen by the introduction of the OGSA paradigm suggested by the US Globus developers, where the main accent is on a Web Services oriented architecture, is being evaluated by EDG, and an evolution of the current architecture in that direction could be envisaged. This is proposed for future releases of the EDG middleware and it may be continued with initiatives in the new EU FP6 framework. An important collaboration has already been established via the GRIDSTART initiative (www.gridstart.org) with the other ten existing EU funded Grid projects. In particular, the EU CrossGrid project (www.crossgrid.org) which will exploit DataGrid technologies to support a variety of applications, all demanding guaranteed quality of service (i.e. real time environment simulation, video streaming and other applications requiring high network bandwith).

Collaboration with similar Grid projects in the US, especially PPDG (www.ppdg.net), GriPhyN (www.griphyn.org) and iVDGL (www.ivdgl.org) is being pursued in collaboration with the sister project EU DataTAG (www.datatag.org). The main goal of DataTAG is the establishment of a transatlantic testbed to deploy and test the software of the EDG project. Interest in the EDG project and its production-oriented approach has already reached beyond the borders of the European Union: after Russia and Romania, which have already installed the EDG software, some Asian sites (in Taiwan and South Korea) have applied to become members of the distributed EDG testbed, in order to participate in High Energy Physics data challenges for data production and simulation.

Reference Documents

Note: all official EDG documents are available on the web at the URL:
http://eu-datagrid.web.cern.ch/eu-datagrid/Deliverables/default.htm

[1] DataGrid D12.4: "DataGrid Architecture"
[2] DataGrid D8.1a: "DataGrid User Requirements and Specifications for the DataGrid Project"
[3] DataGrid D9.1: "Requirements Specification: EO Application Requirements for Grid"
[4] DataGrid D10.1: WP10 Requirements Document
[5] DataGrid D8.2: "Testbed1 Assessment by HEP Applications"
[6] "The Anatomy of the Grid", I. Foster, C. Kesselman, et al. Technical Report, Global Grid Forum, 2001, http://www.globus.org/research/papers/anatomy.pdf
[7] DataGrid D6.1: "Testbed Software Integration Process"
[8] Condor Project (http://www.cs.wisc.edu/condor/). Jim Basney and Miron Livny, "Deploying a High Throughput. Computing Cluster", High Performance Cluster computing,Rajkumar Buyya, Editor, Vol. 1, Chapter 5, Prentice Hall PTR,May 1999.
Nicholas Coleman, "An Implementation of Matchmaking Analysis in Condor", Masters' Project report, University of Wisconsin, Madison, May 2001.
[10] DataGrid Architecture Version 2, G. Cancio, S. Fisher, T. Folkes, F. Giacomini, W. Hoschek, D. Kelsey, B. Tierney, http://grid-atf.web.cern.ch/grid-atf/documents.html
[11] EDG Usage Guidelines (http://marianne.in2p3.fr/datagrid/documentation/EDG-Usage-Guidelines.html)
[12] Software Release Plan DataGrid-12-PLN-333297; http://edms.cern.ch/document/333297
[13] Project technical annex.
[14] DataGrid D12.3: "Software Release Policy"

DataGrid Publications

Gagliardi, F., Baxevanidis, K., Foster, I., and Davies, H. Grids and Research Networks as Drivers and Enablers of Future Internet Architectures. *The New Internet Architecture* (to be published)

Buyya, R. Stockinger, H. Economic Models for resource management and scheduling in Grid computing. *The Journal of Concurrency and Computation: Pratice and Experience (CCPE) Special issue on Grid computing environments*. 2002

Stockinger, H. Database Replication in World-Wide Distributed Data Grids. PhD thesis, 2002.

Primet, P. High Performance Grid Networking in the DataGrid Project. Terena 2002.

Stockinger, H., Samar, A., Allcock, B., Foster, I., Holtman, K.,and Tierney, B. File and Object Replication in Data Grids. *10th IEEE Symposium on High Performance Distributed Computing (HPDC 2001).* San Francisco, California, August 7-9, 2001.

Hoschek, W., Jaen-Martinez, J., Samar, A., Stockinger, H. and Stockinger, K. Data Management in an International Data Grid Project. *IEEE/ACM International Workshop on Grid Computing Grid'2000* – 17-20 December 2000 Bangalore, India. "Distinguished Paper" Award.

Balaton, Z., Kaczuk, P. and Podhorski, N. From Cluster Monitoring to Grid Monitoring Based on GRM and PROVE. *Report of the Laboratory of Parallel and Distributed Systems*, LPDS – 1/2000

Dullmann, D., Hoschek, W., Jean-Martinez, J., Samar, A., Stockinger, H.and Stockinger, K. Models for Replica Synchronisation and Consistency in a Data Grid. *10th IEEE Symposium on High Performance Distributed Computing (HPDC 2001).* San Francisco, California, August 7-9, 2001.

Stockinger, H. Distributed Database Management Systems and the Data Grid. *18th IEEE Symposium on Mass Storage Systems and 9th NASA Goddard Conference on Mass Storage Systems and Technologies*, San Diego, April 17-20, 2001.

Serafini, L., Stockinger H., Stockinger, K. and Zini, F. Agent-Based Query Optimisation in a Grid Environment. *IASTED International Conference on Applied Informatics (AI2001)* , Innsbruck, Austria, February 2001.

Stockinger, H., Stockinger, K., Schikuta and Willers, I. Towards a Cost Model for Distributed and Replicated Data Stores. *9th Euromicro Workshop on Parallel and Distributed Processing PDP 2001*, Mantova, Italy, February 7-9, 2001. IEEE Computer Society Press

Hafeez, M., Samar, A. and Stockinger, H. A Data Grid Prototype for distributed Data Production in CMS. *VII International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT 2000)*, October 2000.

Samar, A. and Stockinger, H. Grid Data Management Pilot (GDMP): a Tool for Wilde Area Replication. . *IASTED International Conference on Applied Informatics (AI2001)* , Innsbruck, Austria, February 2001.

Ruda, M. Integrating Grid Tools to build a computing resource broker: activities of DataGrid WP1. *Conference in Computing in High Energy Physics (CHEP01),* Beijing, September 3-7, 2001

Cerello, P. Grid Activities in ALICE. *Proceedings of the Conference in Computing in High Energy Physics (CHEP01),* Beijing, September 3-7, 2001.

Harris, F. and Van Herwijnen, E. Moving the LHCb Monte Carlo Production system to the Grid. *Proceedings of the Conference in Computing in High Energy Physics (CHEP01),* Beijing, September 3-7, 2001.

Fisk, I. CMS Grid Activities in the United States. *Proceedings of the Conference in Computing in High Energy Physics (CHEP01),* Beijing, September 3-7, 2001.

Grandi, C. CMS Grid Activities in Europe. *Proceedings of the Conference in Computing in High Energy Physics (CHEP01),* Beijing, September 3-7, 2001.

Holtman, K. CMS requirements for the Grid. *Proceedings of the Conference in Computing in High Energy Physics (CHEP01),* Beijing, September 3-7, 2001.

Malon, D. et al, Grid-enabled Data Access in the ATLAS Athena Framework. *Proceedings of the Conference in Computing in High Energy Physics (CHEP01),* Beijing, September 3-7, 2001.

Others Grid Publications

Foster, I., Kesselman, C., M.Nick, J. And Tuecke, S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration.

Foster, I. The Grid: A new infrastructure for 21st Century Science. *Physics Today*, 54 (2). 2002

Foster, I. And Kesselman, C. Globus: A Toolkit-Based Grid Architecture. In Foster, I. and Kesselman, C. eds. *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 1999, 259-278.

Foster, I. and Kesselman, C. (eds.). *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 1999.

Glossary

AFS	Andrew File System
BQS	Batch Queue Service
CE	Computing Element
CVS	Concurrent Versioning System
EDG	European DataGrid
EIP	Experiment Independent Person
Ftree	LDAP-based dynamic directory service
GDMP	Grid Data Mirroring Package
II	Information Index
ITeam	Integration Team
JDL	Job Description Language
JSS	Job Submission Service
LB	Logging and Bookkeeping
LCFG	Automated software installation system
LDAP	Lightweight Directory Access Protocol
LFN	Logical File Name
LSF	Load Sharing Facility
MDS	Globus Metacomputing Directory Service
MS	Mass Storage
NFS	Network File System
PBS	Portable Batch System
RB	Resource Broker
RC	Replica Catalogue
RFIO	Remote File I/O software package
RPM	Red Hat Package Manager
SE	Storage Element
TB1	Testbed1 (project month 9 release of DataGrid)
UI	User Interface
VO	Virtual Organisation
WN	Worker Node
WP	Workpackage

Acknowledgments. The authors would like to thank the entire EU DataGrid project for contributing most of the material for this article.