infogenmed
IST-2001-39013

Coordinator:	Universidade de Aveiro – IEETA
Partners:	UPM, ISC, LinkU, Genomica
Web site:	http://www.infogenmed.net

TECHNICAL REPORT

PROSPECTIVE STUDY ON GRID TECHNOLOGY

IEETA

Abstract: This report presents the results of the prospective study on GRID technology and its applications undertook by the Consortium. The document provides a review of concepts, relevant GRID research and exploitation initiatives, main technologies and tools, and tries to put in perspective the application of GRIDs in the life science domain.

Version:	1.1
Reporting period:	n.a.
Contributing partners:	IEETA
Due on:	n.a.
Preparation date:	2003-10-09

Information Society

Project funded by the European Community under the “Information Society Technology” Programme (1998-2002) – Key Action 1: Systems and Services for the Citizen

Quality Assurance

Document title	Prospective study on GRID technology
Version	1.1
Filename	INFOGENMED_GridReport_v1.1_forDissemination.doc
Lead partner	IEETA
Reviewers	IEETA / PM
Circulation	INFOGENMED Consortium
Copyright notice	Copyright © INFOGENMED
Comments	

Document history

Version	Date	Main changes
1.0	2003-09-05	First version.
1.1	2003-10-08	General revision for public dissemination.

Coordinator

Universidade de Aveiro

IEETA - Instituto de Engenharia Electrónica e Telemática de Aveiro

Address: Campus Universitário de Santiago
3820-193 Aveiro — PORTUGAL

Telephone: + 351 234 370 500

Fax: + 351 234 370 545

More at: www.infogenmed.net

Acronyms

CA	Certificate Authority
DUROC	Dynamically-Updated Request Online Coallocator
GASS	Global Access you the Secondary Storage
GDMP	Grid Data Mirroring Package
GIIS	Grid Index Information Service
GRAM	Globus Resource Allocation Manager
GRIS	Grid Resource Information Service
GSI	Grid Security Infrastructure
GSS-API	Generic Security Service-API
GTK	Globus Toolkit
HPC	High Performance Computing.
IEFT	Internet Engineering Task Force
LDAP	Lightweight Directory Access Protocol
LHC	Large Hadron Colider
MDS	Metacomputing Directory Service
RSL	Resource Specification Language
UNICORE	Uniform Interface to Computing Resources
VO	Virtual Organizations

Table of contents

1 Executive summary

The GRID technology still is in its infancy, but the promised capabilities are highly encouraging. It's still not clear which tools and architectures will prevail, and developers are lacking established programming practices. Given the exciting opportunities, in one hand, and the novelty of the technology, in the other, the INFOGENMED Consortium undertook a prospective study on the GRID technology, herein reported.

The GRID is a solution for high-end computation, large data and collaborative projects, such as environmental research (*e.g.*: Earth System Grid, EcoGrid), physics research (*e.g.*: GriPhyN, CERN OpenLab), astrophysical research (*e.g.*: SpaceGrid, NASA Information Power Grid), life sciences research (*e.g.*: HealthGrid initiative, BIRN), gaming (*e.g.*: Butterfly Grid) and many others.

The deployment of GRID enabled settings follows a layered architecture, which defines mechanisms to allow resources and users to negotiate, manage and interoperate. Efforts are being developed to standardise a reference architecture to facilitate GRID expansibility, interoperability, portability and code sharing: the Open GRID Standard Architecture (OGSA), which mainly represents an evolution towards a GRID system architecture based on Web services concepts and technologies.

Making applications for the GRID depends fundamentally on the supporting middleware. The Globus Toolkit, for example, provides SDKs and APIs to program the GRID, in which a program can be built that runs a job in remote machines, copying the input to and downloading the output from the remote machine(s) according to a "sandbox" paradigm. IBM, for example, has developed the GRID Application framework for JAVA (GAF4J), which is a lightweight framework that encapsulates the GRID middleware details, providing a more familiar programming model that lines up smoothly with common Java abstractions.

With the increase of data transfer rates and the ability for resources clustering, biomedical applications that are not possible today due to technology constraints, may become reality in the future. Representation and simulation of life dynamics (*e.g.*: cell functioning, 3D visualization of organs, rich video based diagnosis, etc.) are expected to become an emerging area, where the demanding computational and data requirements will be addresses by GRID technology.

One topic to be addressed in future biomedical applications is the ability to link heterogeneous sources of knowledge, as addressed by the INFOGENMED project. A GRID infrastructure could be applied in the deployment such environments to ensure the distributed operations (data transfer, remote method invocations, etc.). The system would them be able to delegate issues like authentication, resources location, secure data transfer, and more, on the GRID layer. The GRID alone, however, is not able to provide the semantic reasoning support to enable the integration of heterogeneous, independent database.

2 Introduction

2.1 Purpose

This report presents the results of the prospective study that took place in IEETA on GRID technology and its applications, under the INFOGENMED project. The document provides an initial revision of concepts, relevant GRID research and exploitation initiatives, main technologies and tools, and tries to put in perspective the application of GRIDs in the life science domain.

The work herein reported has not bee specifically foreseen in the INFOGENMED work plan. However, it was found important to assess the opportunities and anticipate the strategic impact concerning the use of GRIDS is biology and medicine, in the scope of WP 5. Although this document was not included in the deliverables list, it provides technical information relevant to INFOGENMED goals and is therefore being included in the work reported to the European Commission.

Links to the INFOGENMED project, pointing out ways on how the project can benefit from the GRID technology, were stripped out from the original report, since they are specific to the project and of lower importance to the research community.

2.2 Prospective experiments under INFOGENMED

GRID technology still is in its infancy, but the promised features highly encouraging. In fact, given the ubiquitous use of the Internet, tools enabling for decentralized, yet highly controlled aggregation of resources for collaboration, are the most valuable for scientific research and knowledge sharing. The road ahead is misty, however. It's still not clear which tools and architectures will prevail, and established programming practices for the GRID aren't available. Given the exciting opportunities, in one hand, and the novelty of the technology, in the other, the Consortium decided to perform a prospective study on the GRID technology.

The endeavours undertaken in this first year of the project regarding the study and assessment of GRID technology, include:

- Release of a scholarship with the specific work programme of studding the GRID technology, its applications, middlewares and set up of a small testbed at IEETA. The present document is mainly reporting this research work.

- Hands-on tutorial on the European DataGrid middleware, given by CERN. This course consisted on a two full day course, with lectures and practical exercises. The layout of the tutorial is available from the European DataGrid web site [17]

2.3 Essential GRID glossary

Grid computing

A type of distributed computing in which a wide-ranging network connects multiple computers whose resources can then be shared by all end users [1]

Virtual Organization (VO)

An organization (typical set up for a limited time frame, for addressing a specific problem or theme) that defines rules that guide membership and use of individuals, resources, and institutions within a community production GRID [7]

GRID middleware

The services needed to assist a set of applications in a distributed environment, composed by the collection of architecture, API and SDK.[2]

Service is a network-enabled entity that provides a specific capability, for example the ability to move files, or to create processes; [7]

Grid Portal

A specialized portal, providing an entry point to the GRID to access applications, services, information and data available within a GRID; [7]

3 Background on GRID technology

"When the network is as fast as the computer's internal links, the machine disintegrates across the net into a set of special purpose appliances".

(Gerge Gilder)

The word GRID was chosen because of the analogy with the electrical grid. Like the electrical grid, computational grids provide an immersive access to computational resources and services, on a "plug-in" paradigm.

GRID infrastructures allow collaborative and integrated use of high-end computers, networks, data, and scientific instruments in multiple organizations. They enable easy sharing of computational power and data repositories, thus effectively fostering the collaboration within a (virtual) organization.

The GRID applications typically involve the use of large amounts of data and computing resources, including the sharing of geographically distributed resources, belonging to different organizations. This scenario is poorly handled by current Internet and web infrastructures.

One of the big problems behind the construction of GRID architectures is the coordination of resource sharing and solving the problem of dynamic multi-institutional virtual organizations. The sharing consists in direct access to computers, software, data and other resources. It has to be sophisticatedly controlled enabling that what is shared can be defined by the resource providers and consumers. [2]

3.1 The goals of the GRID

Modern science is based on computation, data analysis and collaboration. Even with the increase of computational power, data storage and communication speed increasing exponentially, scientists keep demanding for more.

A personal computer is today as fast as a supercomputer in the early 90's. But while in those days simple molecular structures where computed, today scientist calculate structures of complex assemblies of macromolecules and screen thousands of drug candidates.

The demand for storage is also raising the limits. A personal computer is assembled with thousands of Giga Bytes of storage, as much as an entire supercomputer centre in the 90's. In few years CERN predicts to produce several Petabytes (10^6GB) of data per year with its Large Hadron Collider (LHC).

The growth of communication speed between computers has been extraordinary. Nowadays some wide area networks (WAN) operate at 155 Mbps, compared with the 56Kbps that connected supercomputers in 1985, we have an increase of 10^4, but if we compare it with the 40 Gbps connection speed in the TeraGrid project, the difference increases to 10^6 in less than 20 years.

Moore's Law tells us that the number of transistors doubles every 18 months, the storage capacity doubles every 12 months and the connection speed doubles every 9 months. As such, the separation of components becomes inevitable. [5]

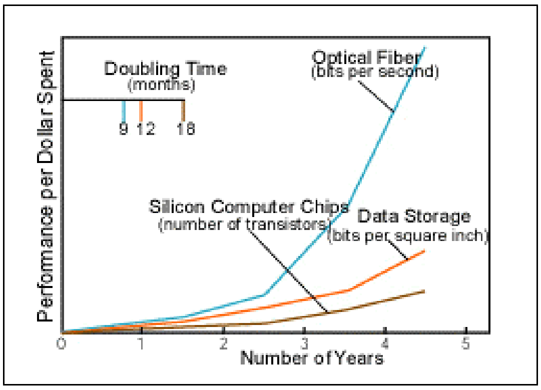

Figure 1: Key performance components and their evolution along the time.

It is expected that the GRID will bring a system that coordinates resources not subject to centralized control, integrating and coordinating resources available from different control domains. To this end, the GRID should use standard, open, general purpose protocols and interfaces, addressing fundamental issues as authentication, authorization, resource discovery and resource access. Finally, one expects the GRID to deliver nontrivial quality of service, allowing its constituent resources to be used in a coordinated fashion to deliver various qualities of services such as response time, security and others. [4]

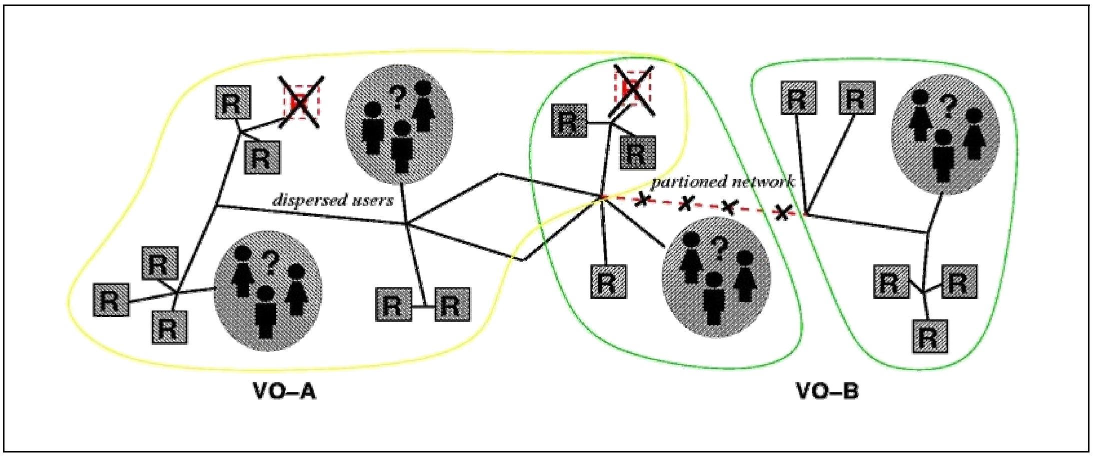

Figure 2 - A graphical representation of how a GRID operates. Two virtual organizations are depicted.

According to Bertsis [33] the GRID can:

- Exploiting the underutilized resources available in the organizations, involving idle resources to optimize the load balancing.
- Parallelize computing, by providing an extensive infrastructure to run very demanding computations. The underlying algorithms will take full advantage of the GRID if they are decomposable in units that can be executed concurrently.
- Enable and simplify the collaboration within dynamic Virtual Organizations. The GRID provides a secure, controlled environment to share resources, according to a

given policy. These resources may include data sets, computational power, software and specific equipments (*e.g.*: costly laboratory instrumentation).

- Balance resource usage among multiple networked sites, graciously handling peaks, in a seamless way to the user.
- Increase the system reliability due to the dispersion of resources, which can tolerate isolated failures by design. In presence of a power loss, for example, resource allocation can be dynamically routed to the available resources.
- Enable the monitoring and management of distributed resource usage, as well to apply quality of service policies. This will empower the collaborations within virtual organizations and support the commercial exploitation of GRID services.

3.2 Envisaged key applications

The GRID is a solution for high-end computation, large data and collaborative projects, such as environmental research (Earth System Grid, EcoGrid), physics research (GriPhyN, CERN OpenLab), astrophysical research (SpaceGrid, Nasa Information Power Grid), life science research (HealthGrid project, BIRN) and many others.

The next enumeration provides a far from extensive list of key applications where GRID technology is being used with success.

- **Smart instruments**. Advanced scientific instruments, such as electron microscopes, particle accelerators, and wind tunnels, coupled with remote supercomputers, users, and databases, to enable interactive rather than batch use, online comparisons with previous runs, and collaborative data analysis.
- **Teraflop desktops**. Chemical modeling, symbolic algebra, and other packages that transfer computationally intensive operations to more capable remote resources.
- **Collaborative engineering** (also know as teleimmersion). High-bandwidth access to shared virtual spaces that support interactive manipulation of shared datasets and steering of sophisticated simulations, for collaborative design of complex systems.
- **Distributed supercomputing**. Ultra-large virtual supercomputers constructed to solve problems too large to fit on any single computer.
- **Parameter studies**. Rapid, large-scale parametric studies, in which a single program is run many times in order to explore a multidimensional parameter space. [20]
- **On-line gaming**. Playing collaborative games on-line is an application area that raises demanding computational requirements. Delays in system response affect user experience, graphics rendering require high computations, and total transparency regarding the players' sites is required [32].

3.3 Architectural view

The deployment of GRID systems is framed by a layered architecture, which defines mechanisms to allow resources and users to negotiate, manage and interoperate.

Protocol stack

The GRID architecture is organized in several layers (Figure 3). The components in each layer share common characteristics, but can build on capabilities and behaviours provided by any lower layer.

These protocols are a proposal for a standardization of the GRID architecture, which will be used by the middlewares. [2] [7] [20]

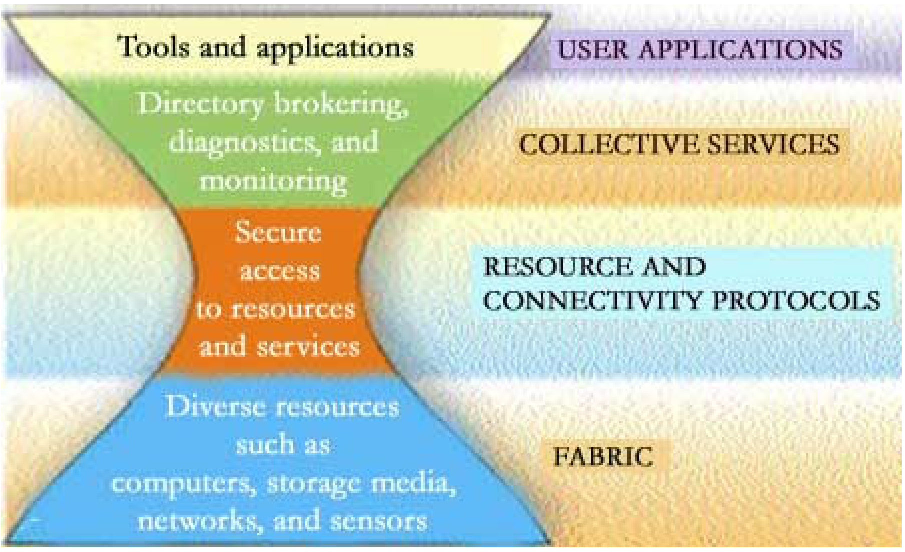
Tools and applications
Directory brokering, diagnostics, and monitoring
Secure access to resources and services
Diverse resources such as computers, storage media, networks, and sensors
USER APPLICATIONS
COLLECTIVE SERVICES
RESOURCE AND CONNECTIVITY PROTOCOLS
FABRIC

Figure 3: Grid Architecture.

We shall now briefly characterize each layer.

Layer 1: Fabric

The Fabric layer provides the resources to which shared access is mediated by Grid protocols, for example: the computational resources, storage systems, catalogs, network resources and sensors.

Layer 2: Connectivity

Defines the necessary communication and authentication protocols for transactions in the GRID. The communication protocols allow data exchange between the different resources in the Fabric layer.

The communication requirements include transport, routing and name attribution. These protocols are developed from existing Internet protocols such as TCP/IP, UDP and DNS.

The security of the connectivity layer is a very complex subject, this obligates the use of solutions based on existing standards whenever possible. As in the communication issue the security standards are built over the Internet protocols.

GRID security solutions have to provide a flexible support for communication protection, i.e. control over the degree of protection, independent data unit protection for unreliable protocols, and support for reliable transport protocols other than TCP.

Layer 3: Resource

The Resource layer was built over the communication and authentication protocols from the Connectivity layer, for defining protocols (APIs and SDKs) for the negotiation, initiation, monitoring, control, accounting, and payment of sharing operations on individual resources. The implementation of these protocols calls Fabric layer functions for accessing and controlling local resources. Resource layer protocols are concerned only with individual resources, ignoring issues of global state and atomic actions between distributed groups. The Collective layer uses these aspects.

Layer 4: Collective

On the opposite from the Resource layer the Collective layer contains protocols that aren't associated with any particular resource, but are global in nature and capture interactions across collections of resources.

Layer 5: Application

This layer comprises the users' applications, which will operate in a GRID environment.

The applications are built in terms of and calling services defined in any other layer. Each layer has well defined protocols that provide access to some useful service: resource management, data access, resource discovery, and so forth.

Open GRID Standard Architecture

Efforts are being developed to standardise a reference architecture to facilitate GRID expansibility, interoperability, portability and code sharing. This is the Open GRID Standard Architecture (OGSA), which mainly represents an evolution towards a GRID system architecture based on Web services concepts and technologies.

The OGSA itself is described by the Global Grid Foundation as follows:

"Successful realization of the Open Grid Services Architecture (OGSA) vision of a broadly applicable and adopted framework for distributed system integration requires the early definition of a core set of interfaces, behaviours, resource models, bindings, and so forth: what we call the OGSA Platform. The OGSA working group within the Global Grid Forum has been formed to define this OGSA Platform by (a) specifying, in broad but somewhat detailed terms, the scope of important services required to support both e-science and e-business applications, (b) identifying a core set of such services that are viewed as essential for many Grid systems and applications, and (c) specifying at a high-level the functionalities required for these core services and the interrelationships among those core services."

3.4 Management view

A distributed system as complex and wide as the GRID can be implies issues that go beyond those from common workstations. These issues include (among others):

- **system security**, to maintain total integrity;
- **data and information management**, to guarantee that relevant user and system data or experiments are available to users and programs on the GRID;
- **resource and execution management**, to manage resource allocation and guarantee that the tasks are executes in a synchronous way;
- **software management** that deals with software packages; and
- **hardware management** to guarantee the integrity of the physical part of the GRID.

The successful deployment of these services must be ensured by an extensive set of management features.

Security management

Security aspects on the GRID include several services:

- Authentication: verifies the identity of an individual in the GRID. The GRID also requires authentication of resources and services.
- Authorization: verifies the actions that an individual can do after he successfully authenticates to the GRID. Are the policies established to determine *who* can do *what* and *when* and using *which* resources.
- Encryption: mechanism to protect the message confidentiality between two points.
- Non-repudiation: provides the data integrity, for that it verifies if the data has reached its destiny without any changes, accidental or malicious.
- Single sign-on: mechanism that supports authentication in several GRID resources in behalf of the user.
- Delegation: process of an GRID entity act in behalf of another entity.
- Community Authorization: defines security policies for group users.
- Secure Execution: when a community becomes so big, that is necessary to have a service that can run a non authorized program in an authorized environment [7]

Data and Information management

In a GRID, the information about the users and the system is critical. The user's information helps to establish collaborative sessions and the information about the system helps the users selecting the appropriated resources and applications. This information is vital for the maintenance, configuration and use of dynamically and heterogenic GRID infrastructures. [7]

Applications management

Each program executed in the GRID is data dependent, and the data requirements for the applications that run in the GRID are enormous. Because of the space limitation, data can be saved in dispersed storage systems or in a central one. If the computing has to be done in several machines, all of part of the data has to be replicated to the machines. It has to be secure ways to transfer files. Filters can be used to reduce the amount of data to be transferred. [7]

Resources management

The computing in the resources is controlled by execution services. The simplest execution service is part of the operating system and authorizes the execution of jobs and tasks in a simple resource.

To allow the use of multiple instances of some resources (as clusters, farms or supercomputers) a co-allocation mechanism is needed. That mechanism will identify the set of resources based on the information service of the GRID and verify the availability of the selected resources, reserve resources, and finally execute the users' task in the resources, avoiding deadlocks and provide a workflow management. [7]

Software

The main concern is to guarantee the interoperability between the different software versions and libraries in installed software services. [7]

Hardware

The hardware management is made by the resource providers. The information about downtimes and maintenance has to be reported to the information service to simplify resource finding by the user. [7]

4 Enabling technologies

The deployment of GRID systems requires the availability of supporting infrastructures and services, as revised in this chapter.

4.1 Networking infrastructure

For the deployment of a GRID infrastructure, there are physical and application requirements. The physical requirements refer to all the structure comprising the computational resources, databases and storage systems; the interconnection between the geographically (or not) distributed resources usually has to have high throughput to keep up with the high end computing scenarios, requiring fast transfer of raw data or control information.

Besides the bandwidth, no special requirement is imposed by GRID systems on the network. However, this requirement alone is challenging and is being tackled by GRID related initiatives (e.g.: European project GEANT[1]).

At the software layer, the infrastructure includes a middleware that supports certificate handling, secure transfer of data, resources management and provides a programming model, among other features.

4.2 Programming the GRID

4.2.1 Programming models

Making applications for the GRID depends fundamentally on the middleware. The Globus Toolkit, for example, provides SDKs and APIs to program the GRID, where a program can be built that runs a job in remote machines copying the input files to the remote machine(s) and downloading the output files from the remote machine after the job ends. The data (individual files) is transferred between computing resources using the concept of a “sandbox”, *i.e.*, a protected, limited environment where applications are allowed to "play" without risking damage to the rest of the system.

Another way to execute useful work in the GRID is by using the command line shell in addition to a script file (RSL), where the Globus commands are responsible for parsing the RSL file and copying the necessary files to and from the remote machine(s); the UNICORE middleware provides a graphical interface for programming a basic GRID program.

[1] http://www.geant.net/geant/index.html

IBM has developed the GRID Application framework for JAVA (GAF4J[2]), which is a lightweight framework that abstracts GRID semantics from the application logic and provides a simpler programming model that lines up smoothly with common Java programming models.

GAF4J abstracts the details of interfacing with a GRID middleware (which is assumed to be the Globus Toolkit) for transferring files to remote nodes, starting remote jobs, and monitoring their execution status. GAF4J provides a simpler programming model that enables the development of maintainable Java applications uncluttered with calls to GRID services.

A detailed discussion of GRID programming models is available in [34].

Appendix A provides examples on how to program Globus from the command line.

4.2.2 Middleware overview

The core of the GRID is the middleware, the services layer required to enable a distributed execution environment for applications. The middleware is structured according to an architecture, and provides API and SDK that allow the creation of a distributed system. It's at a lower level than the end-user application set, but at a higher level than the underlying network transport methods. A variety of middleware packages are available, some freely available (e.g.: Globus Toolkit, UNICORE, Legion) and other proprietary (e.g.: Sun GRID Engine, GRIDSystems).

We shall now study the Globus middleware in detail, briefly reviewing other relevant products.

4.3 Globus middleware

The Globus Project supplies software tools that facilitate the construction of computational grids and applications based on GRID. The set of these tools is called Globus Toolkit (GTK). The discussion in this document is based on Globus 2.4.

A more in depth revision of Globus, as well as tutorials, are available from the IBM Redbooks [1] [35] [36].

4.3.1 Globus overview

Globus Toolkit can be seen as three pillars, where each one represents a primary component from the Globus Toolkit and all three of them are supported by the security foundation.

[2] http://www.alphaworks.ibm.com/tech/GAF4J

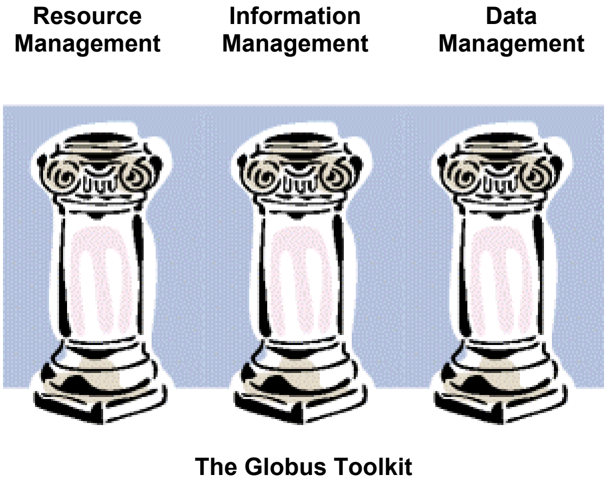

Figure 4 - Globus Toolkit representation

The protocol that manages the resources (resource Management) is implemented by a module called GRAM. The information services protocol (Information Management) is implemented by MDS and GridFTP is responsible by the data transfer protocol (Data Management). All of the use the security protocol GSI in the Connection layer.

GSI

Globus Toolkit uses GSI (Grid Security Infrastructure) to enable authentication and secure communications over the network. GSI provides several useful services to the GRID, including mutual authentication and single sign-on.

The motivation behind GSI lays in the need for secure communications between GRID elements; avoid a centralized security system; the inclusion of single sign-on for the GRID users, including credential delegation to computations that demand multiple resources.

GSI is based in the public key encryption, X.509 certificates and communications based in the SSL protocol. Some extensions to these standards where added to allow single sign-on and delegation.

GRAM

In Globus the architecture that manages the resources is a layer system where in a higher level are the global resources managing services and in the lower level are the local resources allocation services.

In Figure 5 a representation of the functioning of the several components in the resource management architecture is shown. There we can see that the architecture is divided in three main components, namely: language RSL (Resource Specification Language), the interface for the resource management tools and a co-allocator.

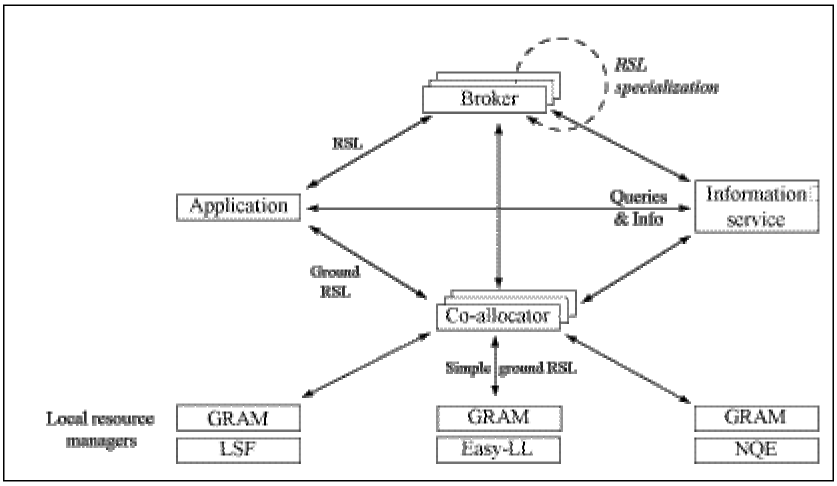

Figure 5: Resource Management Architecture

RSL: provides a method to exchange information concerning the requirements between all the components in the resource management architecture.

GRAM (Globus Resource Allocation Manager): supplies a standard interface to all the resource management tools that a site can have.

DUROC: it supplies the co-allocation services, i.e. coordinates the requests and can generate multiples GRAM.

Currently the Globus toolkit doesn't support a Ressource Broker.

MDS

MDS (Metacomputing Directory Service) provides the necessary tools to build an infrastructure of information based on LDAP for the GRID.

The MSD uses the LDAP protocol as a uniform mean to have access inside the information system to a variety of system components. Optionally a set of uniform names can be created for the resources in a system that can involve some organizations.

GRIS (Grid Resource Information Service) supplies a uniform mean to catalogue the resources in a GRID having as base its current configuration, capacity and state. Such resources must contain among others: computational nodes, data storage systems, instrumentation systems, network connections and databases.

GIIS (Grid Index Information Service) supplies the means to organize the amount of GRIS services and to make available a coherent image of the system that can be explored by GRID applications. The GIIS supply a mechanism to identify the most "interesting" resources where the degree and type of "interest" is specified by the application.

For example: a GIIS can list all the computational resources available inside a set of laboratories, or all the distributed data storage systems that a given organization has. GIIS is able to retrieve information about all GRID resources (computation, data, net, instrumentation) belonging to an investigation consortium, providing a coherent image of the GRID consortium.

GridFTP

The GridFTP is a secure, reliable and high performance data transfer protocol, it was optimized for broad band WANs. The GridFTP protocol is based on ftp and has characteristics such as: use of the GSI for data and control channel security; multiple data channels for parallel transference; partial transference of files; third-party transferences (server-to-server); authenticated data channels; reuse of the data channels and command pipelining.[20]

4.3.2 Components of Globus Toolkit

4.3.2.1 GSI

The security mechanism used by the Globus Toolkit is the GSI (Grid Security Interface) that allows the use of certifications and of some files to provide services of authentication and authorization. The Toolkit implements GSI protocol and APIs to satisfy the security needs of the GRID.

GSI provides the generic security services for the applications that will be executed in the GRID, it also provides the programs that will facilitate multiple logins in remote sites, even if those have different security measures.

The secure communications and the authentication are guaranteed by the GSI, among them the mutual authentication and single sign-on. The single sign-on authentication service allows that an user authenticates himself only one time and has access to the multiple resources of the GRID. The local access control and the remote user mapping to local users also are of the competence of the GSI.

Based in the public key encryption, X.509 certificates and in the SSL/TSL (Secure Socket Layer/Transport Layer Security) communication protocol, being added extensions to allow single sign-on and delegation. GSS-API (Generic Security Service-API) is added to the GSI implementation which is a standard API for security systems, promoted for the IEFT (Internet Engineering Task Force).

GSS-API is independent from any specific security implementation. It is the IETF standard to add authentication, delegation, integrity and message confidentiality to applications, thus guaranteeing safe communications between two parties, in a reliable channel. Separating security from communication, security is easily added to the existing communication code, placing transformation filters in each communication link terminal. All the components of the Globus Toolkit use the GSS-API.

The central point of GSI authentication is the certification, that serves as a identifier to all the elements of the GRID (users and services), contending vital information to identify and to legalize the user or service.

A GSI certificate is identified by the following identification parts: subject name, that identifies the person or object represented by the certificate; public key belonging to the subject; the identity of the Certificate Authority (CA) that has signed the certificate, to prove that the public key and the identification belongs to the subject; the digital signature of the said CA. GSI certificates are coded in the format X.509, a certificate standard created by the IETF.

If two parts have certifications and trust the CAs that trust both certificates, then they can prove each other that they are who they say they are. This process is called mutual authentication. The GSI uses SSL for this protocol. To proceed with the mutual authentication in a secure way, the GSI uses the public key encryption, i.e. sends a message encrypted with the private key and together it sends the public key, if the addressee is able to decrypt the message, this implies that who sent the message is who he says to be, now he encrypts the message again, this time with his private key and together sends it back with his public key, the shipper decrypts the message, compares with the message that he sent, if the messages are equal the addressee is who he says to be, thus finishing the process of mutual authentication.

Certificates are needed to guarantee the access to the Globus Toolkit by the user, i.e. user certificate. To this a certificate for each service that the user wants to execute is added. If he wants to run a local gatekeeper, it also needs to have a host certificate. To run the MDS it needs to have a LDAP service certificate.

GSI supplies the capacity of delegation and single sign-on, which reduces the number of times that the user will have to introduce password. For this a proxy is created, that is defined as a temporary link between a key (key) and the identity of the user.

Delegation allows the remote creation of a user proxy, which results in new private key and a X.509 certificate, signed by the original key. This allows the remote process to authenticate itself in behalf of the user, and to prevent sending of passwords or private keys through the net, thus the remote process represents the user.

When proxies are used, the mutual authentication is slightly different. The remote entity not only receives the certificate from the proxy (signed for the user), but also the user certificate. During the mutual authentication, the users' public key (gotten by the certificate) is used to validate the signature in the proxy certificate. The CAs Public key is then used to validate the signature in the user certificate. Thus a reliable chain is established that goes from the CA until the proxy, passing by the user.

The creation of a proxy allows the user to connect itself to the GRID, to have authorized access and to execute Globus Toolkit programs. The command to create a proxy is grid-proxy-init. To finish the proxy before it dies, the grid-proxy-destroy command is used. [20]

4.3.2.2 GRAM

GRAM is the module that supplies the functionality of remote execution and the the execution state management. When a job is submitted by a client the request is sent to the remote host where it is treated by the gatekeeper (local daemon in the remote host). Then gatekeeper creates a jobmanager to initiate and monitor the job, communicating the state changes to the customer, in the local machine. When the job finishes, the jobmanager sends the information concerning the state to the client and ends.

GRAM is responsible for parsing and processing of the RSL (Resource Specification Language) specifications, that are the profile of the requested job. The request specifies the resource selection, job process creation and the job control. That is obtained denying the request or creating one or more processes (jobs) to satisfy the request.

GRAM is also responsible for allowing the remote monitoring and management of jobs that already had been created.

The RSL is a structured language in which resource requirements and parameters can be defined by the user (client).

To execute a job remotely, is necessary that in the remote machine a GRAM (server) gatekeeper listening in a pre-configured port; the application has to be compiled in the remote machine, or to have been constructed in a way to prevent incompatibilities. The execution starts when an application (GRAM) runs in the local machine, sending the request to the remote computer. The executable, *stdin* and *stdout*, such as the name and the port of the remote computer have to be specified as part of the request. The request is treated by the gatekeeper, which creates a job manager for the new job. The job manager deals with the execution of job and with the communications with the user.

Figure 6 presents the diagram of GRAM architecture.

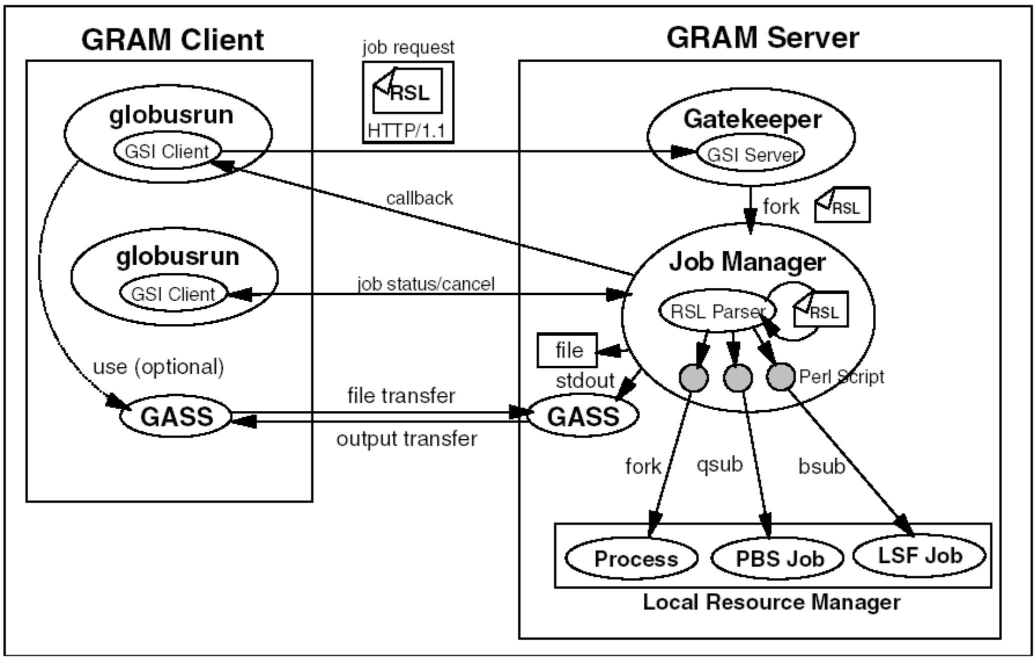

Figure 6: GRAM Architecture

Conceptually the GRAM contains the following elements:

- The command globusrun
- RSL Language
- gatekeeper Daemon

- Job manager
- The process to be executed
- GASS (Global Access you the Secondary Storage)
- DUROC (Dynamicaly-Updated Request Online Coallocator)

The `globusrun` command

The `globusrun` command, submits and generates remote jobs and is used in almost all the GRAM client tools (example: `globus-job-run`, `globus-job-submit`). This command supplies the following functions:

- Request to submit a job to remote machines. The job submission uses security functions, such as the GSS-API, to make the mutual authentication between the clients and servers and also to verify the right to submit a job.
- Transference of executable and the resultant output files from the submission. The `globusrun` command receives the standard result output from the job requests from the remote machines. It uses the GASS to supply safe transferences of files between the machines on the GRID.

RSL- Resource Specification Language

RSL is the language used by the clients to submit a job. All the job request submissions are described in RSL, that includes the executable and the conditions where it will be executed. For example, the necessary memory to execute a job in the remote machine can be specified.

The gatekeeper

The gatekeeper daemon constructs safe communication between the client and the server. The Deamon is similar to the inetd deamon in terms of functionality. However it supplies safe communication. It communicates with the GRAM client (globusrun) and authenticates the right to submit jobs. After the authentication, the gatekeeper creates a job manager delegating to it authority to communicate with the client.

Job manager

The Job manager is created by the gatekeeper and is a part of the job request process. It supplies an interface that controls the allocation of each local resource manager, such as the job scheduler, as the PBS, LSF or LoadLeveler.

The job manager functions are:

- To make parse of the RSL
- Analyses the RSL script
- Allocate the job request to the local resource manager

The local resource manager is normally a job scheduler such as the PBS, the LSF or the LoadLever. Normally the resource manager is written in Pearl.

- Send callbacks to the client, if necessary
- Receives the state requests and cancellations from the client.
- Sends the results (output) to the client using the GASS

GASS Global Access to Secondary Storage

GASS supplies transfer mechanisms, which are used by the GRAM to transfer the output files from the server to the client. Some API use the GSI protocol to supply safe transferences. This mechanism is used by the globusrun command, gatekeeper and job manager.

DUROC – Dynamically-Updated Request Online Coallocator

When using the DUROC the user is capable to submit different jobs to different job managers in different machines or to different job managers in the same machine

Script RSL that contains the DUROC is analyzed by the GRAM client and placed in different job managers.

Figure 7 shows a schematic functioning of DUROC.

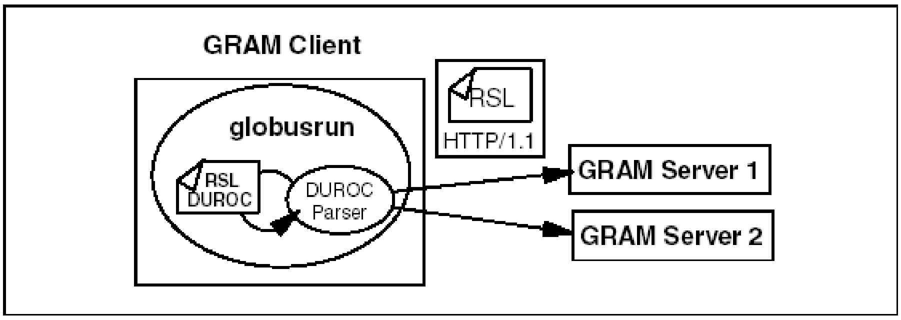

Figure 7 – DUROC model.

GRAM supports the following schedule model. A user (client) or a Resource Broker submits a job request, that is registered initially as pending.

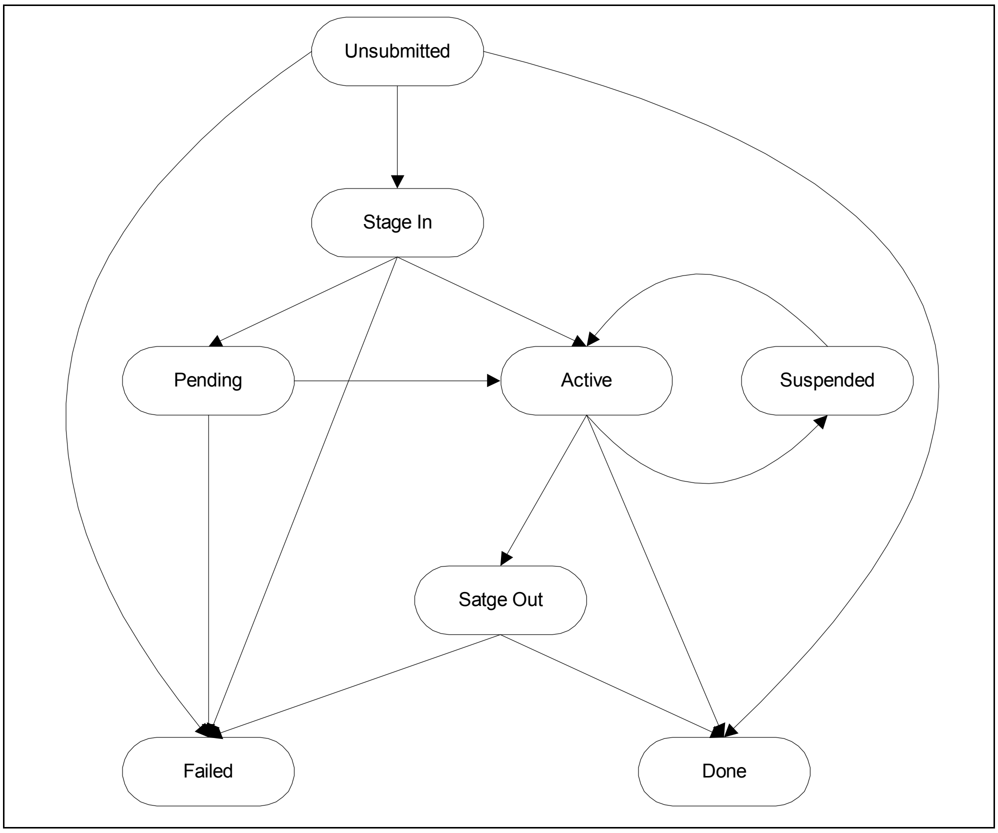

Figure 8: GRAM schedule Model.

Next the description of each one of the possible states of the job is presented [1] [20].

Unsubmitted	The job was not submitted to the scheduler. The callback indicating this job state is never sent; but it is included for the case when the job manager is stopped and restarted before the job was submitted.
Stage In	jobmanager is preparing itself to run the job, i.e. is to copy executable (staging), the input or the data files to the place where it will be executed. Jobs that do not need staging will not enter in this state.
Pending	The job was submitted to the scheduler, but the job resources weren't yet allocated
Active	All the necessary resources are available and the application is being executed.
Suspended	The job was stopped temporarily by the scheduler. Only some schedulers have the capacity to make that a job enters in the suspended state.
Stage Out	The job manager erases the executable and any files copied in the stage in state. Such as in the stage in, jobs that do not need staging will not enter in this state.
Done	The job was finished successfully
Failed	The job finished without success, as resulted of an error, from the user or system cancellation

4.3.2.3 MDS

The MDS (Monitoring and Discovery Service) supplies access to information about resources, being they either static or dynamic. Using a extensive framework to manage the static and dynamic information concerning the state of a computational GRID and all its components, such as networks, computational nodes, storage systems and instruments.

The MDS is build to answer the questions concerning the information system such as:

- Which resources are available?
- What is the state of the computational GRID?
- How can we optimize an application based on the system configuration?

MDS uses the LDAP (Lightweight Directory Access Protocol) as a uniform interface uniform with such information. A great variety of software (commercial servers or open source) can be used to implement the basic functionalities of the MDS: generating, publishing, saving, looking, questioning and showing the data.

Queries are made to the MDS to discover the properties of the machines, computers and networks that the user wants to use, i.e. how many processors are available at the moment; which is the bandwidth; storage capacity; Using a LDAP server, the MDS supplies information to the middleware with a common interface to give a unified image of the different resources.

Information is a critical resource in computational GRIDS. MDS helps creating virtual organizations, where groups of people are involved in collaborative activities and share resources among them. The virtual organization collects and organizes the information of these resources in a uniform way. An infrastructure that supplies an information system that joins the different virtual organizations is needed, so that the applications will be capable of if configuring and adapting to the dynamic conditions of the GRID.

MDS supplies the necessary tools to construct an information infrastructure fort the GRID, based on LDAP. MDS uses the LDAP protocol as a uniform mean to get the information about a wide variety of system components, and optionally construct a uniform namespace system for the resources throughout a system that can involve several organizations. MDS defines an approach to present the data to the users, through LDAP protocols, particular schemas and servers such as the GRIS and the GIIS, which are part of Globus.

Basically MDS contains the following components:

- GRIS - Grid Resource Information Service
- GIIS - Grid Index Information Service
- Information Provider
- MDS client.

Figure 9 represents the conceptual vision of the connections between MDS components. The information about the resource is gotten by the information provider and passes it to the GRIS. The GRIS registers the local information in the GIIS, that register it in another GIIS and thus successively. Client MDS can get information directly from the GRIS (local resources) or from the GIIS (for resources distributed in the GRID).

The MDS uses the LDAP, which supplies a decentralized maintenance of the resource information.

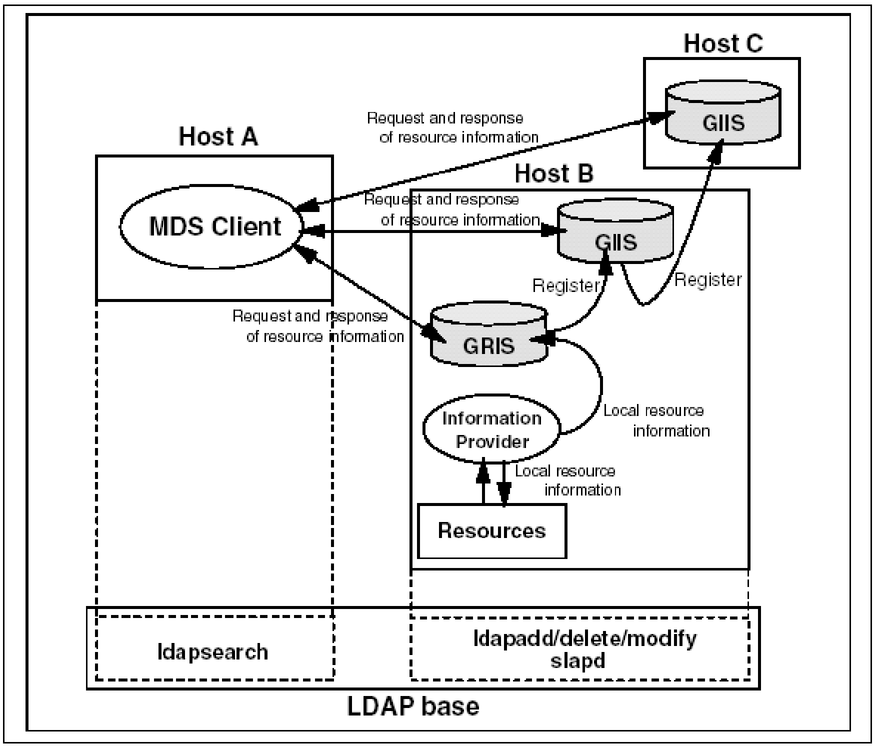
Host C
GIIS
Host A
Request and response of resource information
Host B
MDS Client
Request and response of resource information
GIIS
Register
Register
Request and response of resource information
GRIS
Information Provider
Local resource information
Local resource information
Resources
ldapsearch
ldapadd/delete/modify slapd
LDAP base

Figure 9: Conceptual view of MDS.

Now the MDS architecture is going to be described:

- The resources execute an information service (GRIS) that "speaks" LDAP and supplies information concerning the resources (no query is executed).
- The GIIS supplies a caching service very similar to the one of a web search engine. The resources are registered in the GIIS, the GIIS will always ask the resources information every time it is asked for by a client or when the cache expires.
- The GIIS supplies the Indexation/Search functions of the collective layer.

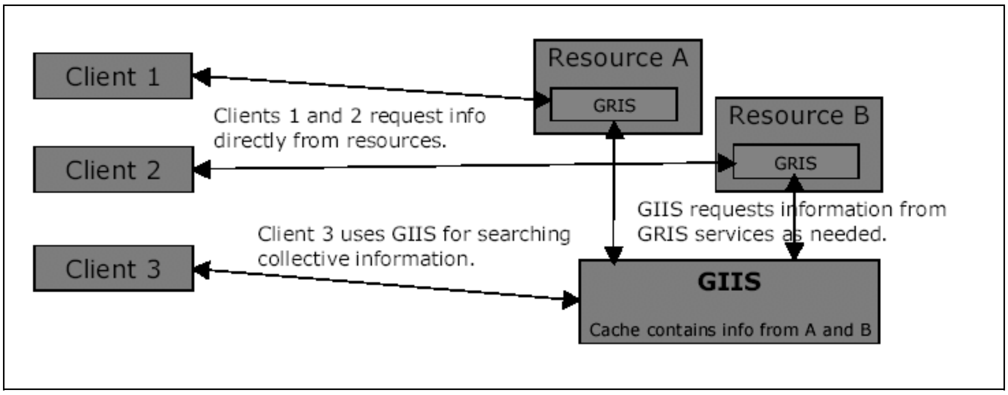
Client 1
Client 2
Client 3
Resource A
GRIS
Resource B
GRIS
Clients 1 and 2 request info directly from resources.
GIIS requests information from GRIS services as needed.
Client 3 uses GIIS for searching collective information.
GIIS
Cache contains info from A and B

Figure 10 – MDS interactions.

GRIS- Grid Resource Information Service

GRIS is the local resources information repository from the information provider. The GRIS can register its information with the GIIS, but the GRIS does not receive register queries. The local information kept by the GRIS is refreshed each time that it is acceded, and is kept during a period of time known as time-to-live (TTL). If there's no information request, the information is deleted after the TTL. Later if there's an information request, the GRIS will contact the relevant information provider to get the most recent information.

GIIS- Grid Index Information Service

Has the resource index information, which was registered by the GRIS and other GIIS. It can be seen as a resource information server for the entire GRID. GIIS has a hierarchic mechanism, such as the DNS and each GIIS has its own name. This means that a client user can specify the name of a GIIS node to search for information.

Information Provider

Takes the local resources properties and state and transforms them into the format defined by the schema and configuration files. In order to add a resource so that it is used by the MDS, it has of be created a specific information provider to translate the properties and states into GRIS.

MDS client

Based in the LDAP ldapsearch command. The search of existing resource information on the GRID is initially made in the MDS client.

Hierarchical MDS

The MDS hierarchic mechanism is similar to the one used in the DNS. The GRIS and the GIIS in the lower layer of the hierarchy, are registered in the GIIS of the higher layers. [20]

4.3.2.4 GridFTP

The GridFTP supplies a safe and reliable transference between the nodes of the GRID. The term GridFTP refers to the protocol itself, the server or to the toolset.

GridFTP protocol

The GridFTP protocol was planed to be used in all data transfers in the GRID. Based in the ftp, but with extensions to the standard protocol, that allow to multistreamed transferences, auto-tuning and security based in the Globus Toolkit.

GridFTP server and client

Globus Toolkit provides the GridFTP server and client, who are implemented by daemon in.ftpd and globus-url-copy command respectively. Supporting the majority of the GridFTP functionalities.

The server and the client GridFTP support two types of file transference: standard and third party. The standard file transference happens when a client sends a local file for the remote machine that runs the GridFTP server (Figure 11).

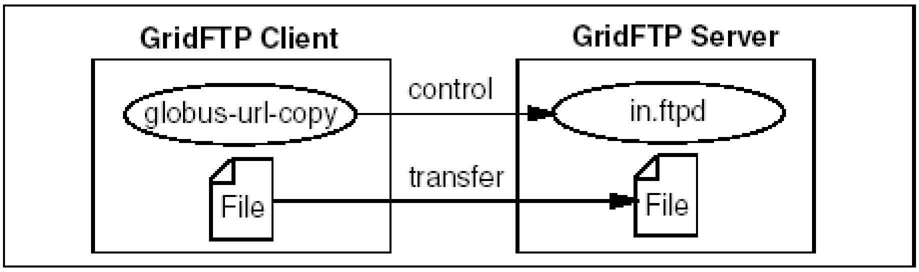

Figure 11: Standard file transference.

The third-party file transference happens when there’s a file in a remote storage system and the client wants to transfer it to another remote server (Figure 12).

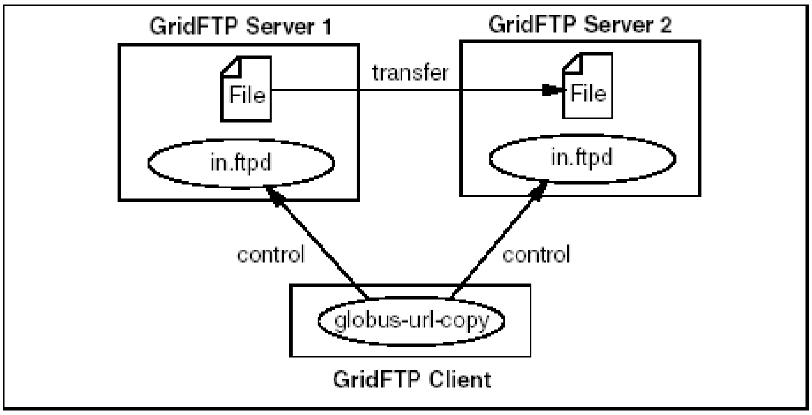

Figure 12 - Third-party transfer.

GridFTP tools

The Globus toolkit supplies a set of tools that support the GridFTP file transference. The gsi-ncftp package is one of the tools that is used to communicate with the GridFTP server. The GASS API is part of the GridFTP tools, it is used by the GRAM to transfer output files from the server to the client. [20]

4.4 Other middlewares

4.4.1 UNICORE

UNICORE is a German project to unite supercomputer centres [30]. It provides a vertical integration environment for GRIDS, including the access to resources through a portal. In the client side it provides a graphical interface that enables to assist in the workflow management of tasks to be scheduled on resources part of supercomputer centres.

UNICORE addresses security on all levels: user authentication is performed using X.509 certificates. A public-key infrastructure has been established for the German HPC centres that enforces rigorous control of certificates. User authorization is handle by the participating sites using their proven mechanisms.

The UNICORE client enables the user to create, submit and control jobs from any workstation or PC on the Internet. The client connects to a UNICORE gateway that authenticates both client and user, before contacting the UNICORE servers, which in turn manage the submitted UNICORE jobs. They incarnate abstract tasks destined for local hosts into batch jobs and run them on the native batch subsystem. The protocol between the components is defined in terms of Java objects.

4.4.2 DataGrid

DataGrid is a European Union funded project, with aiming to build a data-intensive resource GRID, to analyse the data originated from scientific exploration [14].

DataGrid consists of physical resources (computers, disks and networks) and "middleware" software that ensures the access and the co-ordinated use of such resources. The middleware is based on the Globus Toolkit 2.2, so it uses the Globus Data Grid framework: GridFTP protocol and replica management.

The project implements a new layer that is between the applications and the Globus Toolkit.

The Grid Data Mirroring Package (GDMP) is a file replication tool that replicates files from one site to another, managing replica catalogue entries for file replicas.

4.4.3 Gridsystems

GridSystems is a Spanish company that develops a commercial GRID application (InnerGRID) for using the GRID technology in medium and large enterprises.

InnerGrid is a set of several software applications that collaborate together. At its core, one or more servers, integrated with the company’s research or production applications, split data and processes into small units, which are dynamically fed to intelligent agents residing on all the machines that constitute the GRID.

InnerGrid desktop portal allows the users to see the GRID as a single entity, making it possible to manage and use. InnerGrid allows users to “gridify” their applications with a tool called GridStudio, which generates graphical modules that can be deployed instantly in the GRID, without any change in the binary applications themselves. [22]

4.4.4 IBM Grid Toolkit

It’s a package available from IBM that includes the Globus Toolkit [26]. The package also includes additional documentation and installation scripts for the IBM eServer hardware running LINUX or AIX.

The IBM Grid Toolbox is an integrated set of tools and software that facilitate the creation of grids and applications that can exploit the advanced capabilities of the grid using a combination of this toolbox and other technologies.

4.4.5 Sun ONE GRID Engine

It’s the GRID solution from SUN for building Cluster Grids, providing transparent resource access, distributed resource management, high utilization and increased computing throughput [28].

Sun also provides the Sun ONE GRID Engine, Enterprise Edition that features the same functionalities as the Sun ONE GRID Engine, but with advanced features and is the foundation for adopting enterprise grid computing. It consolidates all cluster grids into one enterprise campus grid, allows multiple projects to share resources, and enables to continuously align compute resources with the business needs of your organization.

4.5 Selected projects and initiatives

In the following, we highlight some representative projects. A comprehensive list of GRID related projects is available elsewhere[3].

4.5.1 DATAGRID

DATAGRID is enabling next generation scientific exploration that requires a large amount of computing and shared large-scaled database analysis. The project develops software and testbeds to handle the distributed data, which can have the size of hundreds of TeraBytes, and have thousands of computing resources and a large amount of users around the world in collaborating research institutes. [14]

4.5.2 CROSSGRID

CROSSGRID intends to extend the GRID environment to 11 new European countries, providing answers to the interactive compute- and data-intensive needs of a wide user’s

[3] http://www.gridstart.org/links.shtml

community, promoting the use of the Grid as an instrument for European competitiveness in research, industry and business.

Addressing issues as the source data distribution, simulation and visualisation, virtual time management, interactive simulation and virtual rollback and platform independent virtual reality, CROSSGRID will develop techniques for large scale GRID enabled simulations and visualisations, which require responses in real time, such as: GRID based support for flood defence and prevention; Analysis of simulations in physics; Pre-treatment planning in vascular intervention and surgery; and weather forecasting and air pollution modelling. [13]

4.5.3 EUROGRID

Develops core GRID software components for fast file transferring, resource brokering, interfaces for third-party applications and interactive access. This environment will be tested using applications in biochemistry, weather forecast, computer aided engineering, structural analysis and real time data processing. [18]

4.5.4 GRIDLAB

GRIDLAB's software enables simulation and visualization codes to adapt to changing GRID environments and to exploit the dynamic resources. This will allow the construction of innovative applications that possesses the ability to migrate from site to site during their execution, both in whole or in part, to spawn relative tasks and to acquire resources depending on the availability of GRID resources, or the application needs. [21]

4.5.5 DATATAG

DATATAG is connecting the European and American individual GRID domains. The project is implementing advanced network services for guaranteed data delivery, transport protocol optimisation, QoS and middleware interoperability across different domains.[15]

4.5.6 GRIP

The GRIP is connecting Globus and UNICOR, which are to leading software packages central to the operation of the GRID. The interoperability will be demonstrating by launching jobs via UNICORE and running on resources controlled by the Globus Toolkit. Demonstrations will be made using biomolecular applications and a relocatable weather forecasting model which can adapt dynamically both to locally and remotely available resources. [23]

4.5.7 TERAGRID

TERAGRID intents to build the world's largest, fastest, distributed infrastructure for open scientific research. When completed, the TeraGrid will include 20 teraflops of computing power distributed at five sites, facilities capable of managing and storing nearly 1 petabyte of data, high-resolution visualization environments, and toolkits for grid computing. These components will be tightly integrated and connected through a network that will operate at 40 gigabits per second.

The main objective of TERAGRID is to provide an unprecedented increase in the computational capabilities available to the open research community, both in terms of capacity and functionality; deploy a distributed "system" using Grid technologies rather than a "distributed computer" with centralized control, allowing the user community to map applications across the computational, storage, visualization, and other resources as an integrated environment; and To create an "enabling cyberinfrastructure" for scientific research in such a way that additional resources (at additional sites) can be readily integrated as well as providing a model that can be reused to create additional Grid systems that may or may not interoperate with TeraGrid (but are technically interoperable nonetheless).[29]

5 Life sciences meet the GRID

5.1 Life sciences requirements

Healthcare is moving from diagnosis and treatment into early stage of prevention, taking better account of the characteristics of the individual in healthcare.

Medical information for diagnosis, follow up and pathologies treatment, go from microscopic biomolecular data to macroscopic 3D images and environmental influences. Population-level data suitable for epidemiological studies is increasingly needed both for preventive studies and for an improved understanding of pathologies.

Biomedical data regarding the healthcare of a single individual is fragmented over different sites, and tools providing a homogeneous access to disparate resources are valuable to help health professional to manage the most up-to-date medical information on patients and medical knowledge.

The anonymous medical data can offer new research opportunities and facilitate large scale population studies. This means a growth of medical data dispersed and with no correlation between them. The GRID technology may be a good solution for the need to storage and transfers the large amounts of data, but also for accessing it, updating it, and intelligently combining heterogeneous data to create new knowledge.

Initial projects are starting to apply GRID technologies covering fields such as medical imaging, sequence analysis and biological databases, and database knowledge extraction in order to provide guidelines to health professionals.[9]

5.2 The HealthGrid

The HealthGrid is an association of interested parties aiming to deploy a European stable and unified structure that provides a rapid and intelligent data access to large heterogeneous biomedical databases and applications, which can process and extract knowledge from the data [24].

Based on the grid technologies, the vision is to create an environment where information at the five levels (molecule, cell, tissue, individual, population) can be associated to provide individualized healthcare.

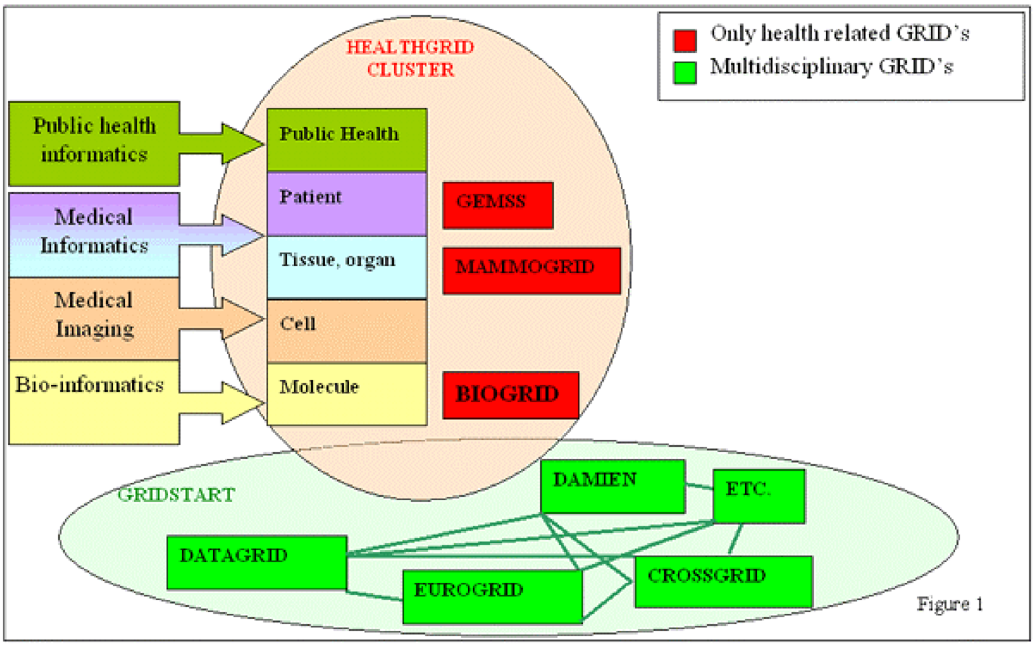

Figure 13 - Overview of the 5 level information system. Some projects are shown at different conceptual positioning.

The HealthGrid community are the participants in the realisation of an environment where people are connected through common technologies, where new partners can join or leave without needing major changes in the system where data can be shared inside virtual communities with access rules and where different fields can work together by making resources accessible to others.

HealthGrid end-users are the healthcare professionals, healthcare providers, researchers and bioinformatics and medical-informatics industries.[9] [24]

5.3 Selected biomedical projects and initiatives

5.3.1 BioGRID (Europe)

Founded by City University, Universities of Cyprus, Paris and Groningen, MRC and ZooRobotics, this project aims to develop interfaces to enable chemists and biologists to be able to submit work to HPC facilities.

It focuses the development of an information and knowledge GRID, allowing knowledge discovery and access to multiple types of unstructured data, effectively visualised and accessible in a structured data model.

The main task of Bio-GRID is to integrate selected applications with UNICORE and provide tools for non-experts in high performance computing. The toolsets and user interfaces for both simulations and visualization of biomolecules are being developed.

The end result of the project will consist of a working prototype of a next generation classification research infrastructure for biotech knowledge interoperability. This consists of (i) automatic classification in a hierarchy metadata repository (ii) intelligent agent sourcing system (iii) accessible visualization user interfaces. [10]

5.3.2 DEVASPIM

This project is developing a telematic service accessible via Internet for surgeons and spine implants manufacturers, which will be a help for lumbar spine implant design.

The developed application will transform the input parameters into a macro understandable by the FEM program, and convert the output parameters supplied by the FEM into relevant and understandable information for the user.

Some of the input parameters may be: patient data (weight, height, sex and age), type of instrumentation, instrumented levels, type of operation, fracture description, etc.

Some of the possible output parameters may be: spine implant stiffness, maximum stress in rods and screws, movements of functional units adjacent to implant, pull-out forces in the screws, etc.[16]

5.3.3 GEMSS

GEMSS demonstrates that Grid can be used to provide medical practitioners and researchers with access to advanced simulation and image processing services for improved pre-operative planning and near real-time surgical support for specific time-critical applications.

This project will be build over existing GRID technologies ensuring future extensibility and interoperability, to provide support for sophisticated authorisation, workflow, security, error detection and recovery.[19]

5.3.4 HKIS

New techniques using mixed data from the genome, transcriptome and proteome can be used for cancer detection and evaluation of therapy resulting in better diagnosis and treatment efficiency. To become widespread these techniques require integration of fast and intelligent processing, and easy control and access to analysis data and results.

The HKIS project proposes the realization of a platform integrating the most recent advances in data processing that will be validated by 3 European oncology hospitals owning very rich but under-exploited tumors and clinical databases.[25]

5.3.5 MAMMOGRID

The objective of the MAMMOGRID project is to develop a European database of mammograms that will be used to investigate a set of important healthcare applications as well as the potential of this GRID to support effective co-working between healthcare professionals throughout the EU.

The project aims to concentrate on the application of Grid technology rather than merely focusing on its further development.[27]

5.3.6 BIRN

The BIRN objective is to create a testbed to address biomedical researchers' need to access and analyze data at a variety of levels of aggregation located at diverse sites throughout the US. The BIRN testbed will bring together hardware and develop software necessary for a scalable network of databases and computational resources. Issues of user authentication, data integrity, security, and data ownership will also be addressed.[12]

5.3.7 BioGRID (Japan)

Biogrid project has been initiated as Construction of a Supercomputer Network, which is a part of IT-program of Ministry of Education, Culture, Sports, Science and Technology. Under the project, Osaka university and other relevant institutions are in the process of developing a Computer Grid Technology to meet IT needs specialized in biology and medical science.[11]

5.4 GRID features and opportunities for biomedical data integration

The availability of GRID infrastructures across several countries, either general purpose or domain oriented, can provide a valuable infrastructure to build information retrieval and integration services. As an example, one can imagine a pan-European GRID for life sciences as a natural environment for distributed information integration tools.

The deployment of science GRIDs not only sets up an infrastructure into which advanced services can be plugged in, as it promotes a collaboration culture and foster the exchange of knowledge. This would favour resources sharing and the availability of information sources that could be plugged into an integration system.

The GRID paradigm offers several features that can be exploited by the distributed information integration environment in future developments of the system (Table 1).

GRID features **Requirements for distributed information integration**	**Controlled access to distributed computing power. Support for high performance computing.**	**Controlled access to information resources and storage space. Support for high performance storage.**	**Decentralized collaboration model (dynamic resources registration and discovery).**	**Scalable and extensible deployment.**
Access to remote databases, over the Internet.		Can be used as the infrastructure to securely transfer remote data. Target sites would need to adhere to the GRID.	GRID dynamic resource allocation and discovery compatible with discontinuous availability of Internet resources. Emerging, redefined or withdrawn data sources can be decentralized managed.	Appropriate to cope with wide area data sources and clients location. Easier to include "heavy" data types without compromising the overall system.
On demand query planning, access to data and unification.	Can distributed the processing at the mediators (query planning and data unification)	Dynamic catalogues of available data resources.	"Just-added" information resources can be readily included in query planning.	
Access to rich data types (e.g.: anatomic imaging, microscopic imaging, etc.)	Visualization may required intensive computing power (e.g.: rendering, multimodal data fusion, etc.)	Special data types may impose high storage requirements. They may be linked to system and retrieved just when needed, from the "nearest" location.		Providers of "costly" data types may allocate resources to enable better GRID performance.
Semantic integration of information from multiple domains.			Catalogues may help on the location of data, but are insufficient to the semantic management of knowledge.	

Table 1: Relationships between GRID features and the requirements of distributed information integration system.

An important benefit from the use of the GRID would arise from the controlled, secure environment provided by the middleware. The system would them be able to delegate issues like authentication, resources location, secure data transfer, and more.

Table 1 also reveals that the GRID alone is not able to provide the semantic reasoning support to enable the integration of heterogeneous, independent databases. In fact, a core problem addressed in INFOGENMED is the representation of domain conceptualizations to enable the correlation of concepts from multiple sources and even domains (biology, genomics, health, etc.).

5.5 Future GRIDs for life sciences

The currently situation is one of expectation, while the results of the first proofs of concept that are up and running come up. As soon as the first applications succeed in being used in the real world work situation, more will follow and will create new opportunities in which GRID technologies can benefit science and industry.

With the increase of data transfer, applications that are not possible today due to technology constraints may become reality in the future; as *in Silico* models of diseased cells, or organs, using GRID power and networks to mimic disease situations.

Simulation and representation of life dynamics (*e.g.*: cell functioning, 3D visualization of organs, rich video based diagnosis, etc.) are expected to be an emerging area, where the demanding computational and data requirements will be addresses by GRID technology.

The creation of virtual organizations in the health domain, where research groups and hospitals collaborate with each other will be facilitated; for example, a network of physicians taking advantage of the GRID technology to feed a database that provides information on a web portal for physicians. Another example may be a GRID between European hospitals and third world countries' institutes allowing physicians in third world countries to fill in patient records that European physicians can use for remote diagnosis.[9]

6 Conclusions

6.1 Key benefits

GRID key benefits include the possibility to establish virtual organizations in which Researchers can collaborate among them sharing their (digital) resources and applications, under a high performance orientation. In a broad sense, current GRIDs are about sharing (research) laboratories!

Creating more flexible, resilient operational infrastructures, avoids the common pitfalls of over-provisioning and incurring in excessive costs. It allows the distribution of computer power, addressing concerns like security for data transfer, integrated approaches to the coordinated use of resources at multiple sites, accelerating the time to results, which allows for the provisioning of extra time and resources to solve problems that were previously unsolvable. In addition to balance the computational load, GRID makes very flexible the addition and removal of resources, being naturally scalable. The GRID aims at exploiting synergies that result from cooperation, the ability to share and aggregate distributed computational capabilities and deliver them as a service.

The key distinction between clusters and GRIDs mainly lies in the way resources are managed. In case of clusters, the resource allocation is performed by a centralised resource manager and all nodes cooperatively work together as a single unified resource. In case of GRIDs, each node has its own resource manager and doesn't aim for providing a single system view. In fact, the decentralization of resources provides the bases for a fault tolerant GRID, coping with the dynamic availability on wide scale systems.

6.2 Current limitations

The GRID is at its beginning and many challenging problems remain to be overcome before having a fully functional GRID, like the creation of high-performance communication methods and protocols.

GRID based biomedical applications will deal with the data internationally distributed. This data isn’t always publicly available, and legal aspects should be taken into account, constituting a problem that can delay the dissemination of biomedical GRID projects.

The capacity of communications and the transfer of large amounts of data has still to be improved to enable computation for massive data analysis.

The generalisation of common reference architecture and the availability of stable middlewares, along with well defined programming models, are also required to foster the development of GRID applications.

6.3 Future developments

Today the GRID is still at the early stage of providing a reliable, well performing, and automatically recoverable virtual data sharing and storage. In the future there will be products that take on this task in a GRID setting, federating data of all kinds, and achieving better performance, integration with scheduling, reliability and capacity.

The Global Grid Foundation[4] is promoting OGSA that is an open standard at the base of the future GRID enhancements. OGSA will standardize the GRID interfaces that will be used by the new schedulers, autonomic computing agents, and any number of others services yet to developed for the GRID. It will make it easy to assemble the best products from various vendors, increasing the value of the GRID computing.

Very promising signals that the GRID technology will make its way to the mainstream are being released by the industry, with large companies announcing their commitment to the GRID [37]. SUN's offers a GRID-based clustering solution[5]; Oracle is adding grid computing capabilities to a new version of its application server software[6]; HP promises to grid-enable all systems[7]; IBM is being very active developing pioneering GRID tools[8].

Communication capacity and transfer rates are increasing very fast, as well as the computational power and the storage systems; this will allow building enhanced Biomedical GRIDs and the development of new applications, currently just imaginable.

[4] http://www.ggf.org

[5] http://wwws.sun.com/software/gridware/sge.html

[6] http://www.infoworld.com/article/03/08/20/HNorappgrid_1.html

[7] http://www.infoworld.com/article/03/09/04/HNhpgrid_1.html

[8] http://www-1.ibm.com/grid/

7 References

[1] L. Ferreira (2003) *Introduction to Grid Computing with Globus* [online], Available from: http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sg246895.html?Open [Accessed on 2003-10-09].

[2] Foster, I., Kesselman, C. and Tuecke, S. (2001), *The Anatomy of the Grid: Enabling Scalable Virtual Organizations*, International Journal of High Performance Computing Applications, 15, 200-222, 2001.

[3] I. Foster, C. Kesselman, J. Nick e S. Tuecke (2002) *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration* [online], Global Grid Forum, Available from: http://www.globus.org/research/papers/ogsa.pdf [Accessed on 2003-10-09].

[4] I. Foster (2002) "What is the Grid? A three point checklist", *Grid Today*, 1(6).

[5] I. Foster (2002) "The Grid: A New Infrastructure for 21st Century Science", *Physics Today*, 55(1).

[6] GridFTP Update, http://www.globus.org/datagrid/deliverables/GridFTP-Overview-200201.pdf, *accessed on* 2009-10-09.

[7] G. v. Laszewski, G. Pieper e P. Wagstrom (2002) *Gestalt of the Grid* [online], Available from: http://www.mcs.anl.gov/~laszewsk/papers/vonLaszewski--gestalt.pdf [Accessed on 2003-10-09].

[8] MDS 2.2 user guide

[9] Proceedings of the first European HealthGRID Conference, January 2003

[10] BioGRID biogrid.icm.edu.pl/, *accessed on* 2003-09-07

[11] BioGRID www.biogrid.jp/, *accessed on* 2003-09-07

[12] BIRN www.nbirn.net/, *accessed on* 2003-09-07

[13] CROSSGRID www.eu-crossgrid.org, *accessed on* 2003-09-07

[14] DataGRID Project www.eu-datagrid.org, *accessed on* 2003-09-07

[15] DataTag datatag.web.cern.ch/datatag/, *accessed on* 2003-09-07

[16] DEVASPIM www.devaspim.com/en/home.htm, *accessed on* 2003-09-07

[17] EDG Tutorial eu-datagrid.web.cern.ch/eu-datagrid/Tutorial/, *accessed on* 2003-09-07

[18] EUROGRID www.eurogrid.org, *accessed on* 2003-09-07

[19] GEMSS www.ccrl-nece.de/gemss/, *accessed on* 2003-09-07

[20] Globus Project www.globus.org, *accessed on* 2003-09-07

[21] GRIDLab www.gridlab.org, *accessed on* 2003-09-07

[22] GridSystems www.gridsystems.com *accessed on* 2003-09-07

[23] GRIP www.grid-interoperability.org/, *accessed on* 2003-09-07

[24] HealthGRID www.healthgrid.org, *accessed on* 2003-09-07

[25] HKIS dbs.cordis.lu/, *accessed on* 2003-09-07

[26] IBM GRID Computing www.ibm.com/grid/ *accessed on 2003-09-07*

[27] MAMMOGRID mammogrid.vitamib.com/, *accessed on* 2003-09-07

[28] Sun ONE GRID Engine www.sun.com/gridware/ *accessed on* 2003-09-07

[29] TeraGRID www.teragrid.org, *accessed on* 2003-09-07

[30] UNICORE www.unicore.org, *accessed on* 2003-09-07

[31] OGSA, http://www.globus.org/ogsa/, *accessed on* 2003-09-07.

[32] The Butterfly Grid, http://www.butterfly.net/, *accessed on* 2003-10-07

[33] V. Berstis (2003) *Fundamentals of Grid Computing* [online], Available from: http://publib-b.boulder.ibm.com/Redbooks.nsf/RedpaperAbstracts/redp3613.html?Open [Accessed on 2003-10-09].

[34] C. Lee, S., Matsuoka, D. Talia, A. Sussman, M. Mueller, G. Allen e J. Saltz (2001) *A Grid Programming Primer* [online], Advanced Programming Models Research Group, Available from: http://www.eece.unm.edu/~apm/docs/APM_Primer_0801.pdf [Accessed on 2003-10-09].

[35] B. Jacobson, L. Ferreira, N. Bieberstein, C. Gilzen, J.-Y. Girard, R. Strachowski e S. Yu (2003) *Enabling Applications for Grid Computing with Globus* [online], Available from: http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sg246936.html?Open [Accessed on 2003-10-09].

[36] L. Ferreira, B. Jacob, S. Slevin, M. Brown, S. Sundararajan, J. Lepesant e J. Bank (2003) *Globus Toolkit 3.0 Quick Start* [online], Available from: http://publib-b.boulder.ibm.com/Redbooks.nsf/RedpaperAbstracts/redp3697.html?Open [Accessed on 2003-10-09].

[37] F. Douglis e I. Foster (2003) "The Grid Grows Up", *Internet Computing*, 7(4).

8 Appendix 1: GTK cookbook

The current *de facto* standard for GRID middleware is the Globus Tool Kit (GTK). In the following, a practical tutorial on the installation and use of GTK is provided. See also [1] and [36] for additional information.

8.1 Installation of GTK 2.4

- Create an account with the name globus – it will be with this account that the Globus Toolkit (GTK) will be installed.
- Create a directory where the GTK will be installed and another for the GPT.
- Define the environmental variable for the GTK and GPT respectively GLOBUS_LOCATION and GPT_LOCATION
- Download the GPT and the source bundles from www.globus.org
- Installing GPT 2.2.9
    - Extract the file `gpt-2.2.9.tar.gz`

    ```
    gzip -dc gpt-*.tar.gz | tar xf-
    ```

    - Execute the program `./build_gpt` to install GPT into the GPT_LOCATION directory
- Installing GTK
    - For compiling the source codes the following command is used

    ```
    $GPT_LOCATION/sbin/gpt-build <options> <bundle name> <flavor>
    ```

    - Repeat the last command for all the bundles that you wish to install.
    - After the bundle installation, run

    ```
    . $GLOBUS_LOCATION/etc/globus-user-env.sh
    $GPT_LOCATION/sbin/gpt-postinstall
    ```

    - Now for the GSI configuration execute as root

    ```
    $GLOBUS_LOCATION/setup/globus/setup-gsi
    ```

BUNDLE	**FLAVORS**
Data Management Client	gcc32dbg
Data Management SDK	gcc32dbg
Data Management Server	gcc32dbg
Information Services Client	gcc32dbgpthr
Information Services SDK	gcc32dbgpthr
Information Services Server	gcc32dbgpthr
Resource Management Client	gcc32dbg
Resource Management SDK	gcc32dbg
Resource Management Server	gcc32dbg
Replica	gcc32dbgpthr
GSI	gcc32dbg

8.1.1 Verifying the installation

- Before proceeding with the verification you have to obtain the certificates.
  - User Certificate

  The user certificate has to be requested as a regular user (the one who will use the GRID), for that the environment has to be loaded, for sh run

```
. $GLOBUS/LOCATION/etc/globus-user-env.sh
```

  next, the certificate files will be created with the following command `grid-cert-request`, a password will be requested to protect the private key.

  In the users home directory a directory called `.globus` will be created where the certificate files `usercert_request.pem`, `usercert.pem` and `userkey.pem` will be saved. The file `usercert_request.pem` is sent to ca@globus.org . the response from the CA will be saved in the file `usercert.pem`, which is empty.

  - Host Certificate (for the Gatekeeper)

  As root run the command

```
grid-cert-request -host <hostname> \

-key /etc/grid-security/hostkey.pem \

-cert /etc/grid-security/hostcert.pem \

-req /etc/grid-security/host.req
```

As with the user certificate, send the file `host.req` to ca@globus.org, the answer will be saved in the file `hostcert.pem` with the permissions 600.

- LDAP Certificate

As root execute the following command

```
grid-cert-request -cn "ldap/hostname" \
-cert $GLOBUS_LOCATION/etc/server.cert \
-key $GLOBUS_LOCATION/etc/server.key \
-req $GLOBUS_LOCATION/etc/server.request -nopw \
-dir $GLOBUS_LOCATION/etc
```

send the file `server.request` to ca@globus.org and save the answer in with the file `server.cert` permissions 600

- Testing GRAM
  - As a normal user and after loading the environment run

```
grid-proxy-init -debug -verify
globus-perdonal-gatekeeper -start
```

this command will return a contact string like

```
"hostname:1234:/O=Grid/O=Globus/CN=Name"
```

  - In the next command substitute <contact> by the contact string

```
globus-job-run <contact> /bin/date
```

This command will return the current date

  - Next terminate the proxy and the gatekeeper

```
globus-personal-gatekeeper -killall
grid-proxy-destroy
```

- Testing MDS
  - As a normal user run

```
$GLOBUS_LOCATION/sbin/globus-mds start
```

  - Test query

```
grid-info-search -anonymous -L
```

the information about the PC is returned.

- Testing GridFTP
  - Starting the GridFTP server

```
grid-proxy-init
```

starting the GridFTP server

```
$GLOBUS_LOCATION/sbin/in.ftpd -s -p 5678 &
```

- Move a test file

Create a file named /tmp/file1

Obtain the <subject> string returned by the command

```
grid-cert-info -subject
```

the command for transferring a file is

```
globus-url-copy "<subject>" gsiftp://hostname:5678/tmp/file1 \
file:///tmp/file2
```

now there will be a file called file2 in /tmp

8.1.2 Initializing the services

- Initializing GRAM
  - As root in `/etc/services` add `globus-gatekeeper` to port 2119, i.e. add the line

```
gsigatekeeper         2119        #globus gatekeeper
```

  - Modifications in xinetd

Add the file `globus-gatekeeper` to the directory `/etc/xinetd.d/` with the following content, where GLOBUS_LOCATION is substituted by the actual location of GTK

```
service gsigatekeeper
{
   socket_type = stream
   protocol    = tcp
   wait        = no
   user        = root
   env         = LD_LIBRARY_PATH = GLOBUS_LOCATION/lib
   server      = GLOBUS_LOCATION/etc/globus-gatekeeper
   server_args = -conf GLOBUS_LOCATION/etc/globus-gatekeeper.conf
   disable     = no
}
```

- Now is necessary to notify xinetd that the configuration has changed, for that run
- Add the certificate subject to the username putting this information in the file

```
/etc/grid-security/grid-mapfile
```

`grid-cert-info –subject` (returns the *subject*)

`whoami` (returns the *username*)

Add the line

```
"/O=Grid/O=Globus/CN=nome" username
```

- Initiate MDS
  - The daemon responsible by MDS is `GLOBUS_LOCATION/sbin/globus-mds`
  - To make the daemon iniciate every time you turn on your machine, the following steps have to be made:
    - Create a symbolic link in `/etc/rc.d/init.d` to `GLOBUS_LOCATION/sbin/globus-mds`

```
ln –s /GLOBUS_LOCATION/sbin/globus-mds globus-mds
```

    - Create a symbolic link in `rc0.d, rc1.d` e `rc6.d` to stop `globus-mds` in `init.d`

```
ln –s ../init.d/globus-mds K90globus-mds
```

    - Create a symbolic link in `rc2.d, rc3.d, rc4.d` e `rc5.d` to `globus-mds` in `init.d` to initiate the service.

```
ln –s ../init.d/ globus-mds S90globus-mds
```

- Initiate GridFTP
  - Add the following line to the file `/etc/services`

```
gsiftp 2811/tcp
```

  - In the directory `/etc/xinetd.d/` create a file named `grid-ftp` with the following content:

```
service gsiftp

{

   instance          = 1000

   socket_type       = stream

   wait              = no

   user              = root

   env               = LD_LIBRARY_PATH = GLOBUS_LOCATION/lib

   server            = GLOBUS_LOCATION/sbin/in.ftpd

   server_args       = -l -a -G GLOBUS_LOCATION
```

```
        log_on_success     += DURATION USERID
        log_on_failure     += USERID
        nice               = 10
        disable            = no
}
```

- Now to notify the xinetd run

```
/etc/rc.d/init.d/xinetd restart
```

8.2 Mini User Guide

8.2.1 Rapidly

Load the environment

```
. $GLOBUS_LOCATION/etc/globus-user-env.sh
```

Initiate the proxy

```
grid-proxy-init
```

Testing the authentication in a given machine

```
globusrun –a –r <hostname>
```

"Hello World"

```
globus-job-run <hostname> /bin/echo "Hello World"
```

8.2.2 Commands to run Jobs

8.2.2.1 Globus-job-run

`globus-job-run` is the basic command to run jobs in one or more remote resources. It translates the arguments in a RSL request using the `globusrun`. The program runs in foreground and prints the output in the users terminal.

globus-job-run: Hello World

The basic syntax of this command is the same as the rsh

Syntax:

```
 globus-job-run <hostname> </path/executable> <arguments>
```

Example:

```
globus-job-run greco.ieeta.pt /bin/echo "Hello World"
```

the path is the one from the home directory in the remote resource.

globus-job-run with files

the `globus-job-run` allows to run files that are found in a remote machine, the syntax is the same from the previous point.

Example:

In the remote machine exists a executable that prints in the screen the name of the machine where it is running, the executable's name is hn.

```
globus-job-run greco.ieeta.pt hn
```

in the monitor appears Greco.ieeta.pt

Staging

If the executable is located in the local machine, it is possible to run the file in a remote machine, staging the file, i.e. making a copy of the file in the remote machine and after the job executes the copy is deleted. For that the option –s (-staging) is added to the command `globus-job-run`.

Syntax:

```
globus-job-run <hostname> -s[tage] <executable> <arguments>
```

Example:

```
globus-job-run greco.ieeta.pt -s hw
```

The executable hw prints on the screen the message Hello from <hostname>, this file is located on the machine where the `globus-job-run` command is executed.

In the monitor appears:

```
Hello from

greco.ieeta.pt
```

Note: the –stage option only works with the `globus-job-run` command an not with `globusrun`.

Subjobs

It is possible to run from the same command line an executable in several different remote machines, controlling the number of processes from that executable in the different machines and also another executables.

Syntax:

```
globus-job-run -: <hostname1> -np <number of processes> <executble> <arguments> -: <hostname2> <executable2> <arguments>
```

Example:

```
globus-job-run -: greco.ieeta.pt -np 2 -s hw -: cortex.ieeta.pt /bin/ls /tmp
```

in the screen will appear:

```
Hello from

Hello from

greco.ieeta.pt

greco.ieeta.pt

<listing of /tmp in cortex.ieeta.pt>
```

Note:

-: represents the beginning of a new subjob.

The arguments that are putted after -args are maintained for all the command except for the ones in the <argument> field.

-np indicates the number of times the process is going to be running in that machine.

Multiple commands

Multiple commands can be executed in the same remote machine, for that the following syntax is used.

Syntax:

```
globus-job-run <hostname> /bin/sh –c "command1 ; command2"
```

Example:

```
globus-job-run greco.ieeta.pt /bin/sh –c "cd /tmp ; ls -l"
```

in the screen will appear the content of the /tmp directory

8.2.2.2 Globus-job-submit

Allows the user to run programs in batch mode in a remote machine, using a syntax like the one from the `globus-job-run`. After submitting the job, the communications between the client and the server are closed and a jobID is presented. The programs output is placed in a buffer in the remote machine, which is accessible any time with the `globus-job-get-output`. The buffer can be cleaned with the command `globus-job-clean`.

Syntax:

```
globus-job-submit <hostname> <executável> <argumentos>
```

Example:

```
globus-job-submit greco.ieeta.pt –s hn
```

the jobID returned was:

https://greco.ieeta.pt:33126/3139/1048261765/

with the command `globus-job-get-output` the output can be returned

```
globus-job-get-output https://greco.ieeta.pt:33126/3139/1048261765/
```

the output was:

```
Hello from

greco.ieeta.pt
```

8.2.2.3 globusrun

The `globusrun` command runs specifications written in the Globus RSL (Ressource Specification Language) language. The RSL script can be introduced directly in the command line or with a *.rsl file. The following script can be copied to the command line or to a file, this can give an idea about the structure of the language.

```
 & (count=1)

(executable=/bin/echo)

(arguments="Hello World")
```

Syntax:

When used with a RSL file:

```
globusrun -s -r <hostname> -f file.rsl
```

when used in the command line:

```
globusrun -s -r <hostname> 'RSL script'
```

Notes:

-s indicates that the output will be sent to the stdout

-r indicates that the resource name comes next

-f indicates that the filename comes next

globusrun doesn't use staging, but using the GASS server in globusrun, for that is added

```
(executable=$(GLOBUSRUN_GASS_URL) # "path/file") to the RSL script
```

8.2.2.4 Job monitoring

What is the state of the job? Is it still running? Which is the JobID?

Normally, if nothing returns to the UNIX console after the commands `globus-job-run` or `globusrun`, then it's because the job is still running.

The ps command can be used remotely, to return information about process that are running, without having to login.

```
globus-job-run <hostname> /bin/sh –c "ps –u <username>
```

Killing a job

Interrupting the commands `globus-job-run` or `globusrun` with the CTRL+C the job will automatically be killed.