



DELIVERABLE D4.2
TEST AND VALIDATION TESTBED
ARCHITECTURE

WP4

Document Filename:	CG4.4-D4.2-v1.0-LIP011-ValidationOfTestbedArchitecture
Work package:	WP4
Partner(s):	LIP
Lead Partner:	CSIC
Config ID:	CG4.4-D4.2-v1.0-LIP011-ValidationOfTestbedArchitecture
Document classification:	PUBLIC

Abstract: This documents provides an overview of the technologies that can be used in CrossGrid and describes the initial architecture for the test and validation testbed.



Delivery Slip

	Name	Partner	Date	Signature
From				
Verified by				
Approved by				

Document Log

Version	Date	Summary of changes	Author
1-0-DRAFT-A	17/4/2002	Draft version	Jorge Gomes
1-0-DRAFT-B	21/7/2002	Draft version	Jorge Gomes, Mario David
1-0-DRAFT-C	04/9/2002	Draft version	Jorge Gomes, Mario David

CONTENTS

1. INTRODUCTION.....	4
1.1. DEFINITIONS ACRONYMS AND ABBREVIATIONS.....	4
1.2. REFERENCES.....	6
2. STATE OF THE ART.....	8
2.1. INFORMATION SYSTEM.....	8
2.1.1. MDS.....	8
2.1.2. LDAP NAMING AND STRUCTURE.....	10
2.1.3. MDS AND FTREE.....	12
2.1.4. MDS AND THE INFORMATION TREE IN EDG 1.2.0.....	13
2.1.5. MDS AND R-GMA.....	13
2.2. THE WORKLOAD MANAGEMENT SYSTEM.....	14
2.3. COMPUTING ELEMENTS, GATEKEEPERS AND WORKER NODES.....	18
2.4. GDMP, REPLICATOR AND THE REPLICATOR CATALOGUE.....	21
2.4.1. REPLICATOR CATALOGUE.....	21
2.4.2. GDMP.....	21
2.4.3. EDG REPLICATOR.....	22
2.5. STORAGE ELEMENT.....	23
2.6. INSTALLATION SERVER.....	24
2.6.1. LCFG.....	24
2.7. VIRTUAL ORGANIZATIONS.....	26
2.8. GSI AND PROXY CREDENTIALS.....	28
2.9. MONITORING.....	29
2.9.1. NETWORK MONITORING.....	30
2.9.2. TESTBED MONITORING.....	31
2.9.3. APPLICATION MONITORING.....	31
3. THE CROSSGRID TEST AND VALIDATION TESTBED.....	32
3.1. TESTBED COORDINATION AND SCHEDULING OF TEST ACTIVITIES.....	32
3.2. CURRENT TESTBED STATUS.....	33
3.2.1. CROSSGRID ACTIVITIES.....	33
3.2.2. ACTIVITIES WITH DATAGRID.....	36
3.2.3. MAIN SITE CONFIGURATION.....	37
3.3. INFORMATION SYSTEM.....	40
3.3.1. INFORMATION TREE TOPOLOGY.....	40
3.3.2. INTEGRATION WITH OTHER MDS TREES.....	41
3.4. THE WORKLOAD MANAGEMENT SYSTEM.....	42
3.5. COMPUTING ELEMENT.....	44
3.6. REPLICATOR CATALOGUE AND REPLICATOR SOFTWARE.....	46
3.7. STORAGE ELEMENT.....	47
3.8. INSTALLATION SERVER.....	50
3.9. CERTIFICATES, VIRTUAL ORGANIZATIONS AND THE PROXY SERVER.....	51
3.10. MONITORING.....	52
3.10.1. APPLICATION MONITORING.....	53
3.10.2. TESTBED MONITORING.....	54
3.10.3. NETWORK MONITORING.....	55
3.11. NETWORK INFRASTRUCTURE.....	56
3.12. NETWORK SECURITY ISSUES.....	59
3.13. TESTBED CONFIGURATION.....	61
4. FINAL REMARKS.....	63

1. INTRODUCTION

The reliability of the CrossGrid production testbed will depend much on the reliability of the underlying middleware. CrossGrid software distributions will be based on the Globus Grid toolkit, on DataGrid middleware and on CrossGrid middleware written to enable parallel and interactive applications as well as user-friendly access to applications through portals. The complexity of the middleware makes it prone to development and configuration errors hence a comprehensive test phase will be required before the production testbed deployment.

The middleware testing activities must be performed using a separated testbed infrastructure called the “**Test and Validation Testbed**”. This is required in order to not disturb the production and development testbeds where new applications and middleware are being developed. Also the volatile nature of the test activities where the middleware, configurations and even system software must change frequently is not compatible with a production or even development infrastructure.

This document discusses the architecture of the “Test and Validation Testbed” starting with the state of the art in terms of Grid middleware covering both Globus and DataGrid, from here possible configurations are discussed. Since Grid middleware is being developed in a fast rhythm it’s impossible to establish a static architecture. The architecture of the “Test and Validation Testbed” will depend mainly on the requirements of the middleware being tested. However CrossGrid aims to be compatible with Globus and DataGrid. Starting with these goals and using the CrossGrid software architecture as input, possible testbed configurations can be foreseen.

1.1. DEFINITIONS ACRONYMS AND ABBREVIATIONS

Acronyms and Abbreviations

ACL	Access Control List
AFS	Andrew File System
API	Application programming interface
ATM	Asynchronous Transfer Mode
CA	Certification Authority
CASTOR	CERN Advanced Storage Manager
CE	Computing Element
CES	Component Expert Subsystem
CN	Common Name
CRL	Certificate Revocation List
CrossGrid	The EU CrossGrid Project IST-2001-32243
DataGrid	The EU DataGrid Project IST-2000-25182
DBMS	Database Management System
DHCP	Dynamic Host Configuration Protocol

EDG	European DataGrid
FTP	File Transfer Protocol
GGF	Global Grid Forum
GMA	Grid Monitoring Architecture
HTTP	HyperText Transport Protocol
HSM	Hierarchical Storage Management
JDL	Job Description Language
JSS	Job Submission Service
LB	Logging and Bookkeeping
LCAS	Local Centre Authorization Service
LCFG	Local ConFiGuration system
LDAP	Lightweight Directory Access Protocol
LFN	Logical File Name
MAC	Media Access Control
MyProxy	An Online Credential Repository for the Grid
MDS	Monitoring and Discovering Service (used to be called Metacomputing Directory Service)
NFS	Network File System
NTP	Network Time Protocol
OU	Organizational Unit
PXE	Pre boot eXecution Environment
PKI	Public Key Infrastructure
PFN	Physical File Name
QoS	Quality of Service
GDMP	Grid Data Mirroring Package
GID	Unix Group ID
GIIS	Grid Information Index Service
GRAM	Grid Resource Allocation Manager
GRIS	Grid Resource Information Service
GSI	Grid Security Infrastructure
RA	Registration Authority
RC	Replica Catalogue
RM	Replica Manager
RB	Resource Broker
RDBMS	Relational Database Management System
RDN	Relative Distinguish Name
RFIO	Remote File I/O
R-GMA	Relational Grid Monitoring Architecture

RSL	Resource Specification Language
SE	Storage Element
UI	User Interface
UID	Unix User ID
VO	Virtual Organization
VOMS	Virtual Organization Membership Service
XML	Extensible Markup Language
WMS	Workload Management System
WN	Worker Node
WP	Work Package

1.2. REFERENCES

Software Requirements Specification for MPI Code Debugging and Verification; CG-2.2-DOC-0003-1-0-FINAL-C

General Requirements and Detailed Planning for Programming Environment; CG-2-D2.1-0005-SRS-1.3

Software Requirements for Grid Bench; CG-2.3-DOC-UCY004-1-0-B

Software Requirements Specification for Grid-Enabled Performance Measurement and Performance Prediction; CG-2.4-DOC-0001-1-0-PUBLIC-B

Portals and Roaming Access; CG-3.1-SRS-0017

Access to Remote Resources State of the Art; CG-3.1-SRS-0021-2-1-StateOfTheArt

Grid Resource Management; CG-3.2-SRS-0010

Grid Monitoring Software Requirements Specification; CG-3.3-SRS-0012

Optimization of Data Access; CG-3.4-SRS-0012-1-2

Optimization of Data Access: state of the art; CG-3.4-STA-0010-1-0

Detailed Planning for Testbed Setup; CG-4-D4.1-0001-PLAN-1.1

Testbed Sites and Resources Description; CG-4-D4.1-002-SITES-1.1

Middleware Test Procedure; CG-4-D4.1-004-TEST-1.1

Evaluation of Testbed Operation; DataGrid-06-D6.4-0109-1-11

Data Access and Mass Storage Systems; DataGrid-02-D2.1-0105-1_0

EDG-Replica-Manager-1.0; DataGrid-02-edg-replica-manager-1.0

Data Management Architecture Report Design Requirements and Evaluation Criteria; DataGrid-02-D2.2-0103-1_2

WP4 Fabric Management Architectural Design and Evaluation Criteria; DataGrid-04-D4.2-0119-2_1

LCFG The Next Generation, P. Anderson; A. Scobie, Div. of Informatics Univ. of Endinburgh

Middleware Test Procedure, CrossGrid; CG-4.4-TEMP-0001-1-0-DRAFT-C

Definition of Architecture, Technical Plan and Evaluation Criteria for Scheduling, Resource Management, Security and Job Description; DataGrid-01-D1.2-0112-0-3

An Online Credential Repository for the Grid; J. Novoty, S. Tuecke, Von Welsh
VO Server Information; J. A. Templeton, D. Groep; NIKHEF 23-Oct-2001

Testbed Software Integration Process; DataGrid-6-D6.1-0101-3-3

Grid Network Monitoring; DataGrid-07-D7.2-0110-8-1

Network Services: requirements deployment and use in testbeds; DataGrid-07-D7-3-0113-1-5

WP5 Mass Storage Management Architecture Design; DataGrid-05-D5.2-0141-3-4

Information and Monitoring Architecture Report; DataGrid-03-D3.2-33453-4-0

Information and Monitoring Current Technology; DataGrid-03-D3.1-0102-2-0

Definition of Architecture, Technical Plan and Evaluation Criteria for Scheduling, Resource Management, Security and Job Description; DataGrid-01-D1.2-0112-0-3

European DataGrid Installation Guide; DataGrid-06-TED-0105-1-25

Information and Monitoring Services Architecture Design Requirements and Evaluation Criteria; DataGrid-03-NOT

WP4 Architectural Design and Evaluation Criteria; DataGrid-04-D4.2- 0119-2_1

2. STATE OF THE ART

The middleware for the initial CrossGrid testbed prototype in month six will be based on the Globus and DataGrid distributions. This will ensure compatibility with other sites running Globus and EDG middleware thus extending the geographic coverage of the Grid in Europe and at the same time providing a basis for the development and test of CrossGrid middleware and applications.

The first prototype will be extremely important to gain experience with the deployment and maintenance of the Grid technologies over which future releases of the CrossGrid testbed will be built. At the same time these technologies will be tested and evaluated contributing to improvements of the middleware quality and providing input for the definition of future CrossGrid testbed architectures.

In this context understanding the existing technologies over which the CrossGrid architecture will be based is essential. These technologies are described in the following section.

2.1. INFORMATION SYSTEM

Information about existing Grid resources and their characteristics is essential to make the best possible job scheduling decisions. Information about available resources is made available to the whole Grid through a distributed information system.

2.1.1. MDS

The Globus toolkit uses the MDS information directory system to publish static and dynamic information about existing resources. The current implementation of MDS is based on the LDAP protocol a standard that defines a way for clients to access information objects in directory servers.

Since MDS is based on LDAP it inherits all the advantages and weaknesses of the LDAP standard and corresponding software implementations. The MDS directory service is therefore a database optimised for read and search operations that provide the means to organize and manage information hierarchically and also to publish and retrieve the information by name.

In MDS, nodes that need to publish information about them selves must run a local GRIS service. The GRIS service is basically a set of scripts and a LDAP server. The scripts gather the information and publish it into the LDAP server. Each GRIS registers itself to an upper LDAP server called GIIS (Grid Information Index Server) thus creating a hierarchy of LDAP servers. The GIIS can then be queried to for information contained in the GRIS nodes below.

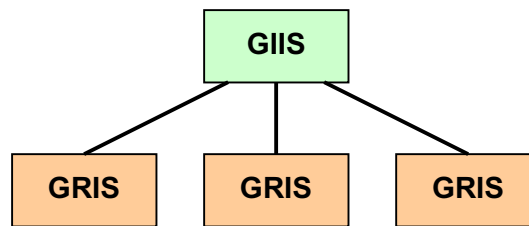


Figure 1 – A simple MDS tree

Using this approach a tree structure can be built where the tree leafs produce information and the upper nodes provide entry points to access that information in a structured way. GIIS servers have also the ability to cache information based on the TTL (Time to Live) fields associated to each piece of information. Caching ensures the tree scalability and a reasonable response time.

GRIS servers are usually run in Gatekeeper nodes and publish information about computing resources served by the Gatekeeper, worker nodes are not required to have a GRIS. The storage elements also require a GRIS server to publish information about supported protocols, the closest CE, storage size and the virtual organizations supported with the corresponding directories.

GIIS servers are responsible for the aggregation of several sources of information. Usually one or more GIIS servers are deployed per institution depending on the number of information sources and internal administrative organization. For organizational scalability purposes is also usual to aggregate several organizations under a single country or project GIIS server as shown in the next figure.

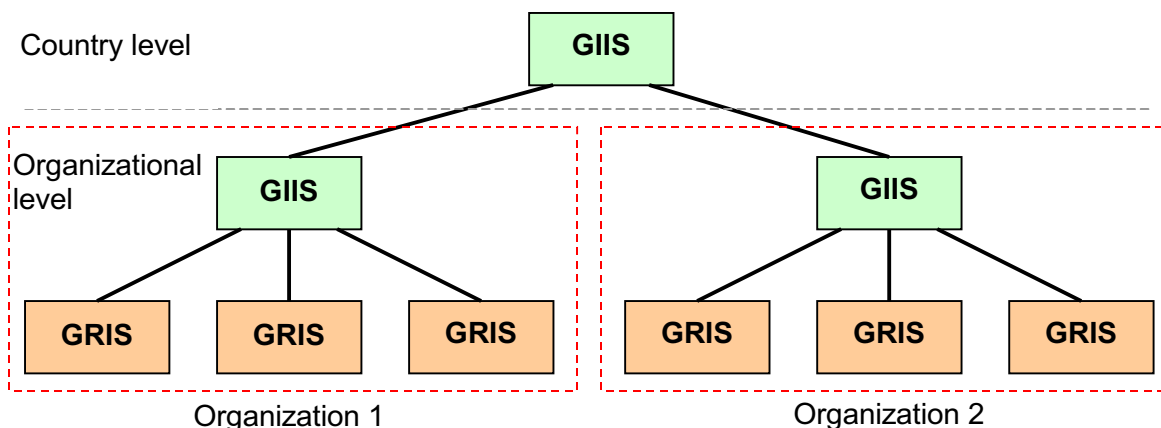


Figure 2 – Country MDS tree

A more complex example of an information tree can be found in the next diagram.

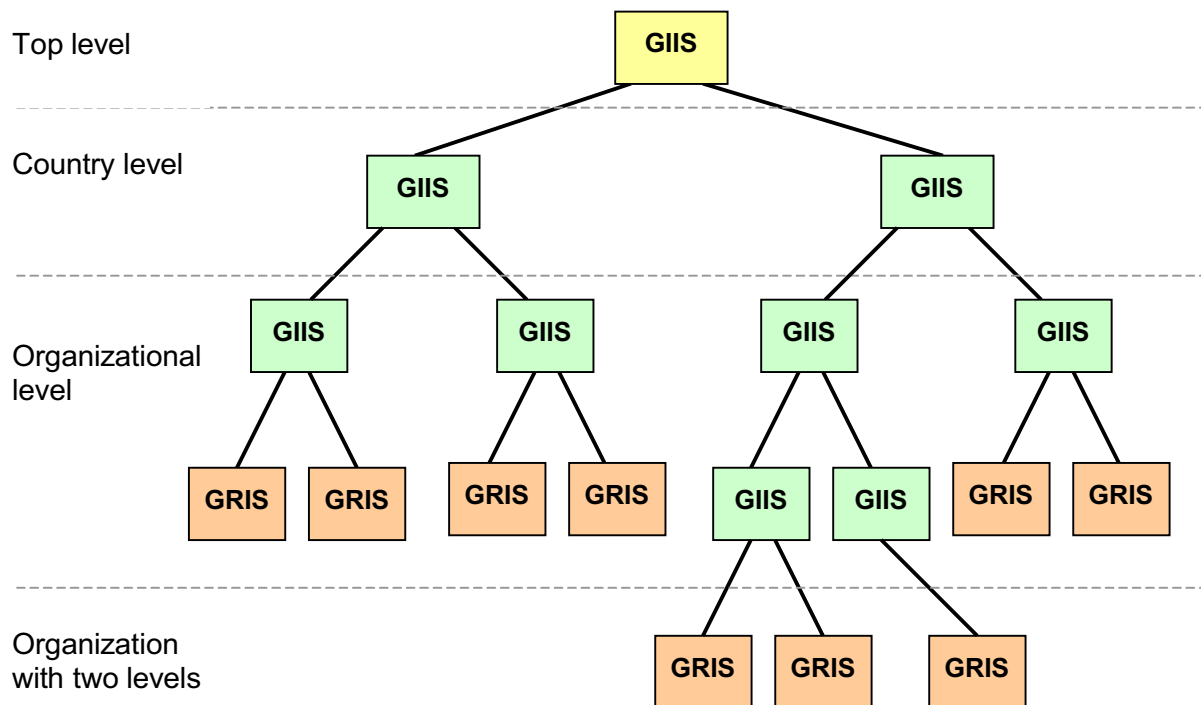


Figure 3 – A complete MDS tree

In the example above the top of the information tree is visible and four layers of GIIS servers are present. Each GIIS only knows about the GIIS server immediately above it to which registration requests must be sent to build the tree.

2.1.2. LDAP NAMING AND STRUCTURE

MDS is developed and maintained by the Globus project and is based on the OpenLDAP software an open implementation of the LDAP protocol.

Entries in a LDAP directory are identified by a name. LDAP uses the X.500 naming convention where each object name is separated from its naming attribute by an equal sign; this combination is called RDN (Relative Distinguish Name).

Understanding the X.500 naming scheme is important not only to understand the MDS but also for understanding other grid services that are based on LDAP, or rely on the X.500 naming scheme such as the VO LDAP servers, CA/RA LDAP servers, RC servers and the X.509 certificate names.

The X.500 naming attributes and corresponding abbreviations are the following:

Naming Attribute	Abbreviation
Country	C
Locality	L
Organization	O
Organizational Unit	OU
Common Name	CN
Domain component	DC

The following examples show six independent RDNs using several abbreviations:

c=pt	cn=Jorge	o=lip	o=csic	c=pl	ou=Engineering
------	----------	-------	--------	------	----------------

Each directory entry in a LDAP tree has a unique name called DN (Distinguish Name). A DN is formed by joining all the corresponding RDNs separated by commas starting at the top of the tree to the object location.

The following example shows on the left a LDAP directory tree where each circle represents a directory entry. On the right the DNs corresponding to the leaf entries are shown.

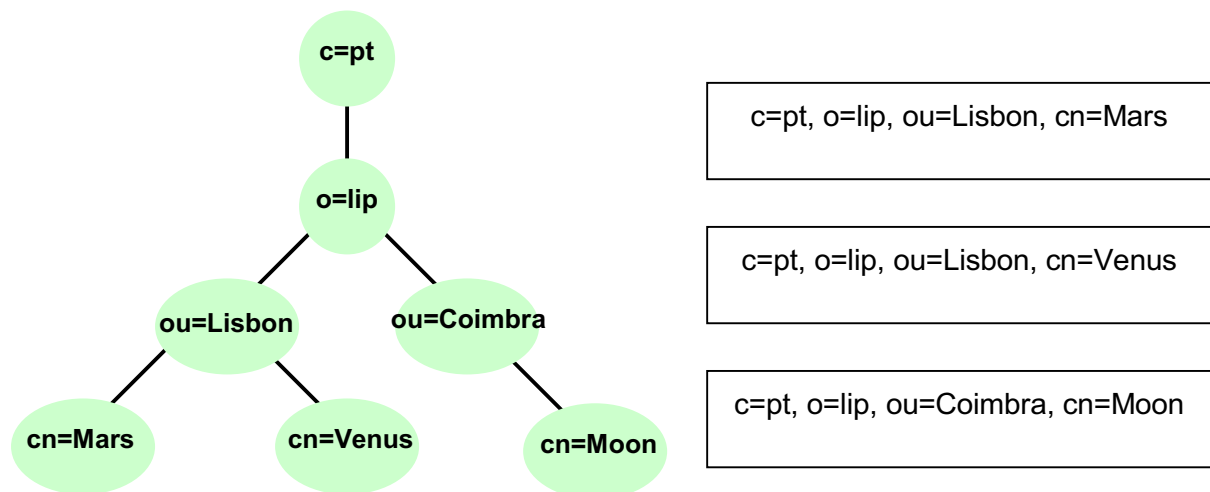


Figure 4 – LDAP tree example

Due to the LDAP distributed nature the whole information tree can reside in just one server or branches of the tree can be delegated to several servers. Most of the time the directory tree reflects the geographic or organizational structure of the entities maintaining it. Another approach is to build the tree to match the Internet DNS tree structure; this arrangement permits finding the location of LDAP services by using the DNS.

Distributing the tree across several servers according with the organizational structure contributes to maintenance of the directory scalability since each organization or organizational unit will be responsible for maintaining its own directory server. Simultaneously, robustness is increased since a failure in one directory server will not affect the other servers.

The following examples show two possible configurations for the same directory tree. In the left example the whole tree is kept in just one server. In the right example the tree is distributed across three servers, one server is responsible for the top of the tree matching the organization head quarters and two other servers are responsible for keeping information related with the branch centres.

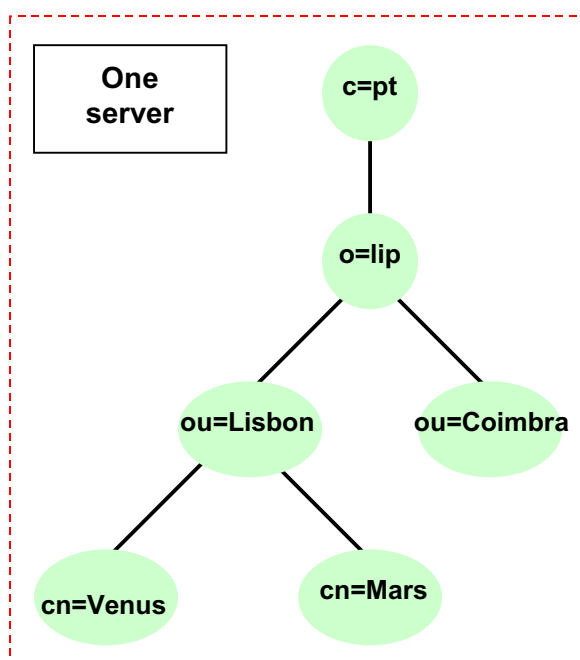


Figure 5 – LDAP tree in one server

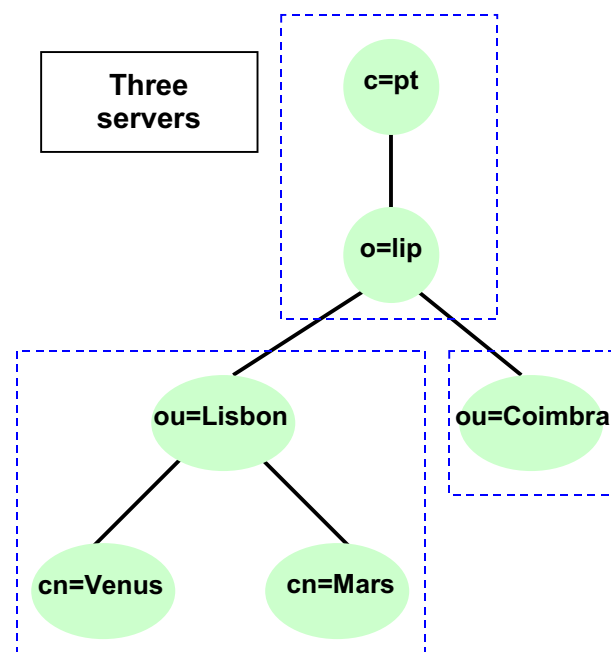


Figure 6 – Distributed LDAP tree

The MDS approach is to use an LDAP server inside each leaf node containing information about the node and its services, this is called the GRIS. Several other LDAP servers containing references to the other LDAP servers in the layer below are used to glue the whole information tree, these are called GIIS servers. GIIS servers are usually deployed matching the organizational structure with one GIIS per site, one GIIS server above the sites to integrate them into national organizations and countries and one GIIS at the top to integrate the country GIIS servers.

2.1.3. MDS AND FTREE

FTREE was developed by DataGrid to resolve some of the performance issues of the Globus MDS. FTREE is capable of caching information in memory and push information from the

bottom of the tree to the upper GIIS servers and thus answer queries faster. FTREE is based on a modified version of OpenLDAP and was designed to be compatible with the MDS API thus making the replacement of the information directory system completely transparent. The FTREE modifications will likely be integrated in a future release of OpenLDAP, at that point MDS and FTREE will probably be merged.

Currently the EDG middleware contains both the Globus MDS and FTREE. However FTREE has not been extensively tested and the performance benefits have not yet been verified in practice. The current DataGrid approach is to install the two information systems however only MDS is currently used.

2.1.4. MDS AND THE INFORMATION TREE IN EDG 1.2

The recommend information system for the current EDG 1.2 release is still the Globus MDS. Unfortunately there is a problem in the propagation of information in the MDS tree that prevents the wide deployment of trees with several layers. In EDG 1.2 there are only two layers in the MDS tree composed by an information top index installed in the RB system in the first layer and a second layer with the MDS servers running in gatekeepers. It is expected that this situation will change in a near future enabling the deployment of full MDS trees.

2.1.5. MDS AND R-GMA

R-GMA is an information management service for distributed systems that was initially developed to support the information management requirements of the DataGrid application monitoring middleware. R-GMA is based on the Grid Monitoring Architecture (GMA) specified by the Global Grid Forum (GGF).

The GMA architecture is composed of three components, they are: consumers, producers and the registry directory service. Producers register themselves into the registry, the registry can then be queried by the consumers to locate the producers. Once located by the consumers the producers can be queried directly to obtain the relevant data.

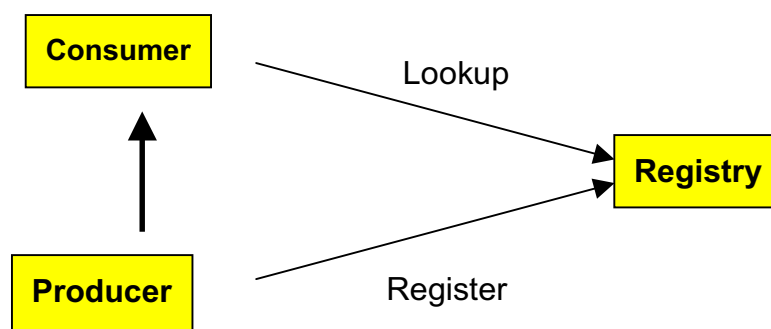


Figure 7 – R-GMA architecture

R-GMA exposes a relational model with SQL support; therefore SQL statements can be used through the API to perform queries, and manipulate the data organized in the form of

database tables. The R-GMA architecture and SQL capabilities make it a flexible and powerful tool to manipulate static or dynamic information.

DataGrid is considering the replacement of the whole Globus MDS information system by R-GMA. This is still an open issue that may have a deep impact in the current EDG middleware, and as consequence in CrossGrid components that make use of Globus and DataGrid services. Possibly EDG release 1.3 will support both Globus MDS 2.2 and R-GMA alongside. A complete phase out and replacement of MDS by R-GMA may happen with EDG release 2.0. Besides the immediate technical impact this decision will have a strategic impact on the middleware architecture since it represents the abandon of one of the most important components of the Globus toolkit. Therefore future interoperability problems with other Globus toolkit components may arise in the future due to differences between both information systems implementations.

DataGrid is developing an MDS API wrapper for R-GMA with the objective of allowing the transparent replacement of the Globus MDS implementation by R-GMA without the need to change the existing applications and middleware currently relying on the Globus MDS. However this can be a complex task since hiding the specificities of R-GMA will also result in losing much of its advantages such as much of the SQL capabilities.

For the testbed architecture this represents among other changes the removal of the GIIS tree (used for resource discovery), and its replacement by the R-GMA registry directory service (used for service discovery).

2.2. THE WORKLOAD MANAGEMENT SYSTEM

The workload management system (WMS) is the component that manages the Grid resources making sure that jobs are executed in the best possible way taking advantage of the available resources. The WMS consists of the following components:

- Resource Broker (RB) is the component responsible for matching job submission requests with the available resources. The RB receives job submission requests in the form of job descriptions written in job description language (JDL), which is based on the "Class Advertisements" developed by the Condor project. They contain information about the user application and required resources, its up to the RB to find a Computing Element (CE) somewhere in the Grid that is both available and satisfies the job requirements.
- Job Submission Service (JSS) is the component that performs the actual job submission to the remote CE found by the RB. The JSS uses the Globus GRAM service to submit the jobs. JSS is actually a thin wrapper around CondorG.
- User Interface (UI) is the component that sits between the end user and the RB. Users submit jobs to the Grid by writing a job description in JDL that is submitted to the RB through the UI. The UI also allows the user to:

-
- Obtain logging and bookkeeping information about the submitted jobs;
 - Transfer the job input and output files;
 - Cancel a submitted job;
 - Find a list of resources suitable to run a specific job.
- Logging and Bookkeeping (LB) is the component that keeps information about the job scheduling operations. The resource broker, the user interface and the gatekeepers interact with the LB to store and retrieve this information. The bookkeeping information consists of short-term data about current jobs while the logging information is long term data about the jobs and the whole workload management system.

The WMS components RB, JSS and LB servers are usually run in the same physical computer. The UI can be run in a user workstation or in a dedicated system. If the testbed contains several large VOs then a WMS per VO might be advisable. However there is no support for state synchronization between RBs, this can cause resource management problems when several RBs are deployed hence this option should be carefully considered.

In order to find a CE capable to satisfy a job request, the RB uses the MDS directory tree to search for available computing resources with the required characteristics. The RB is configured to perform searches over the MDS directory tree starting from a root GUIS.

The Resource broker can also interact with the Replica Catalogue in order to find the location of the data required by job requests.

The user written JDL file may specify a ranking expression to sort the matching CE's according to the user preferences. Ranking expressions are built using rank attributes; these are variables stored in the CE GRIS that change frequently such as the number of CPU's available or the average performance.

The WMS also allows the transfer of input files from the UI during the job submission process. The job input files to be transferred must be specified in the job input sandbox. The input sandbox is a list of files to be transferred with the job, and is usually used to transfer the programs to be executed in the job context. It's also possible to transfer back to the UI any output files produced by the job. In this case an output sandbox containing the list of file names to be transferred must be specified in the JDL file.

The following diagrams show examples of job submissions with and without data requirements. Understanding both types of job submissions is extremely important to test and validate the middleware components involved in the Job submission process. Simple tests will not include the replica catalogue while more complex tests will include the replica catalogue.

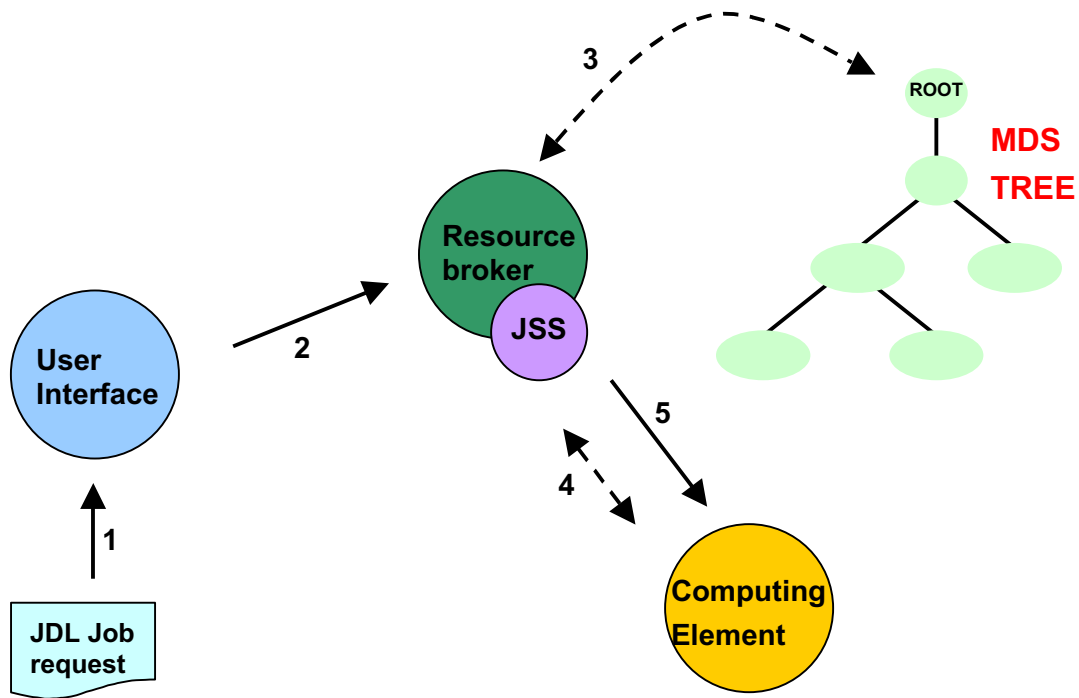


Figure 8 – Grid job submission

The previous diagram shows in a simplified way a Grid job submission without external data requirements (RC intervention) the flow is as follows:

1. A job submission request written in JDL is submitted to the RB through the user interface.
2. The user interface sends the JDL job description together with the input files specified in the input sandbox to the resource broker after successful GSI user authentication.
3. The resource broker searches the MDS directory tree to build a list of Computing Elements (CE) that:
 - a. match the job requirements;
 - b. allow the user to submit jobs.
4. If the JDL description specifies a ranking expression then the resource broker contacts directly the GRIS at each matching CE to obtain information about the rank variables needed to compute the rank expression. The CE with the highest rank expression value will be selected to run the job. If several CE's share the same highest rank value then the first listed CE will be selected. In the future top equally ranked CE's should be randomly selected.
5. Once a match is found the job is submitted to the remote computing element by the JSS using the globus job submission service (GRAM). This is done after translation of

the JDL file to Globus RSL, this step is required since gatekeepers don't understand JDL.

However most jobs require large input data files previously stored and possibly replicated in several Storage Elements. In this case the Resource Broker must contact the Replica Catalogue in order to find the Storage Elements containing the replicas of the required input files. The following diagram illustrates a job submission with data requirements where the Replica Catalogue is queried before searching the MDS tree. The resource broker does not start replication copies hence the files must already exist in one or several Storage Elements and they must have been registered in the Replica Catalogue.

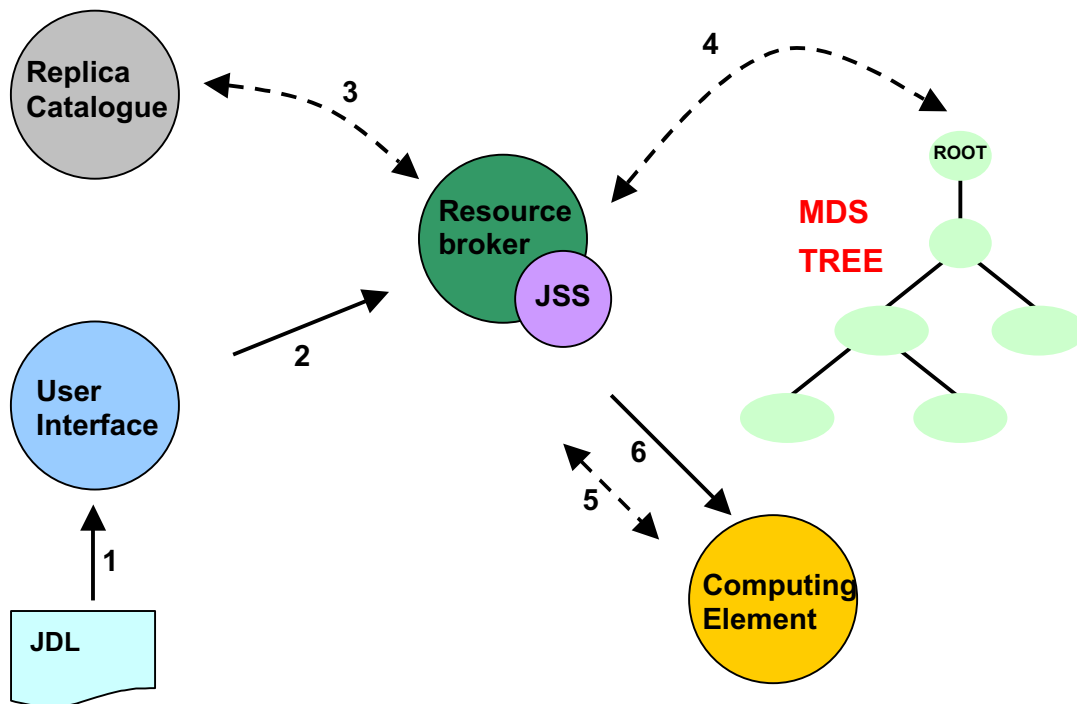


Figure 9 – Grid job submission with data requirements

The job submission is as follows:

1. A job submission request written in JDL is sent to the RB through the user interface.
2. The user interface sends the job request together with the input files specified in the input sandbox to the resource broker after user authentication.
3. The Resource Broker interacts with the Replica Catalogue to translate each input Logical File Name (LFN) into a list of Storage Elements and names of Physical File replicas corresponding to each LFN.
4. The Resource Broker interacts with the MDS:
 - a. The resource broker searches the MDS directory tree to build a list of Computing Elements (CE) where:
 - i. The user certificate is allowed to submit jobs.

- ii. The output Storage Element specified in the job description is close to the CE.
 - b. The CE's are then classified based on the number of required input files present in Storage Elements close to them that use the file transfer protocol specified in the job description. This is accomplished by finding the SE's close to each CE in the list and matching them with the SE list built on step 3.
 - c. Once the CE's classification is done the resource broker starts searching for the classified CE's that meet the user specified job requirements.
5. If the JDL description specifies a ranking expression then the resource broker contacts directly the GRIS at each matching CE to obtain information about the rank variables needed to compute the rank expression. The CE with the highest rank expression value will be selected for running the job. If several CE's share the same highest rank value then the first listed CE will be selected.
6. Once a match is found the job is submitted to the remote computing element by the JSS using the GRAM service. Again this is done after translation of the JDL file into Globus RSL.

It's important to mention that at this stage the JDL file has been translated from the WMS JDL to Globus RSL by the JSS, this is required since the actual Grid job submission from the WMS is performed using the Globus GRAM service that only understands RSL.

2.3. COMPUTING ELEMENTS, GATEKEEPERS AND WORKER NODES

The basic Grid computing infrastructure called CE (Computing Element) contains the following components:

- Gatekeeper: is the system that provides the gateway through which the WMS submits jobs to local farm nodes. The Gatekeeper sits between a local computing farm and the WMS, jobs are submitted from the WMS to computing farms using the Globus GRAM service. The GRAM service allows jobs to be submitted to local job-managers running in the CE's that in turn will send them to local batch scheduling systems thus hiding the complexity and heterogeneous nature of each computing facility. The gatekeeper also has a GRIS information system that publishes information about allowed users and local resources to the whole Grid. This information is used by the WMS to make Grid scheduling decisions.
- Working Node (WN): is a local farm computing node where the job is actually executed. Jobs arrive to the WN through the local batch scheduling system. Many worker nodes can exist behind a single Gatekeeper.
- A batch scheduling system. Processing load is distributed across WN's by a batch scheduling software such as PBS, LSF or Condor. Jobs reach the WN through the Gatekeeper that acts as a local batch submission interface.

The next diagram shows a job being submitted to a Grid computing farm through a gatekeeper and a PBS batch scheduler. The example starts with the RSL job specification therefore the RSL job request could have been submitted either by the WMS JSS component or even through a user specified globus-job-submit command.

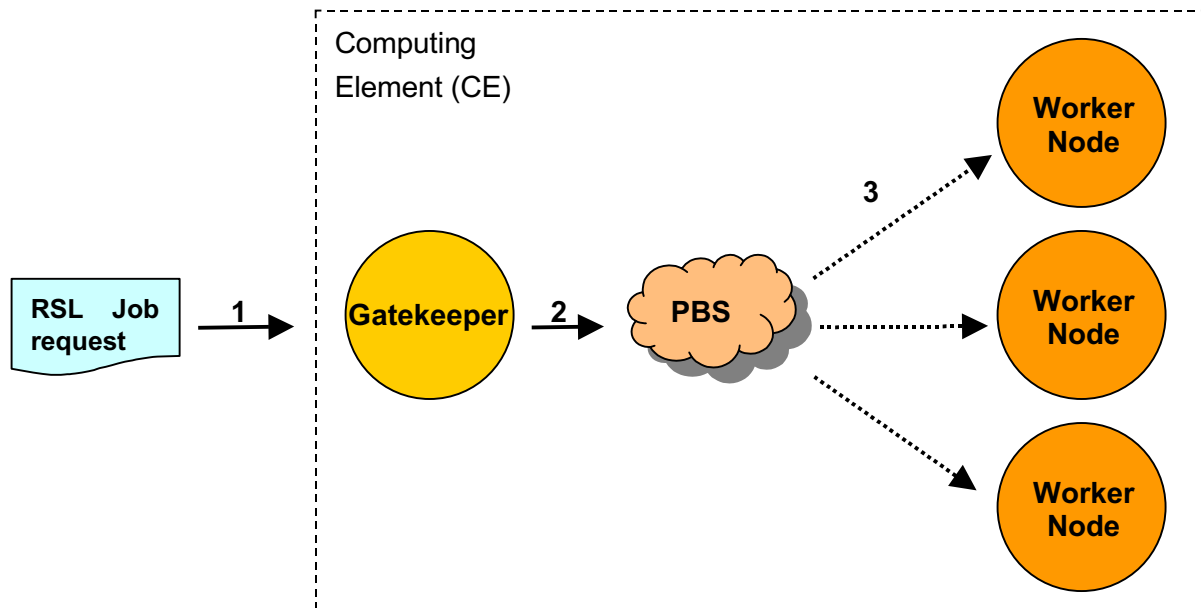


Figure 10 – Local job submission

The job submission to a computing farm is as follows:

1. The gatekeeper receives a job submission request from the WMS.
The job is authenticated by the gatekeeper and the user certificate distinguish name is mapped to a local UNIX username.
2. The gatekeeper submits the job to the local batch system scheduler (in this diagram PBS is used).
3. The PBS batch scheduler starts the job execution in a free farm WN.

The mapping between user certificates and UNIX usernames is accomplished through an authorisation file (gridmapfile) that contains all the user certificate distinguish names authorised to access the CE. Each distinguish name in the file has a corresponding UNIX username or pooled account provided by the local system administrator.

Pooled accounts are an automatic way of mapping a user certificate distinguish name to one local UNIX username of a pool (set) of usernames that is made available for this purpose by the local system administrator. With pooled accounts the first time the user tries to access the CE his distinguish name is automatically mapped into one free account of the pool, this mapping is permanently registered in a directory (called gridmapdir). Therefore there is no need to manually map a certificate distinguish name to a specific username, making the process of adding new authorised users easier.

By looking at the job submission model in the above diagram it might seem that the WN does not require any Grid middleware, however this is not true because:

- The WN is responsible for transferring the input files (input sandbox) from the resource broker and transferring the output files (output sandbox) back to the

resource broker. A job wrapper script generated by the JSS accomplishes these file transfers using globus-url-copy.

- The WN might need to interact with the RC and SE's to retrieve or save other required data files and/or publish them.

For these actions to succeed the WNs need at least outbound network connectivity. However this restriction will likely be removed in the EDG release 2 allowing worker nodes to reside in an entirely private network. However other issues may affect the connectivity of the worker nodes, such as the communication requirements of parallel applications running across CEs.

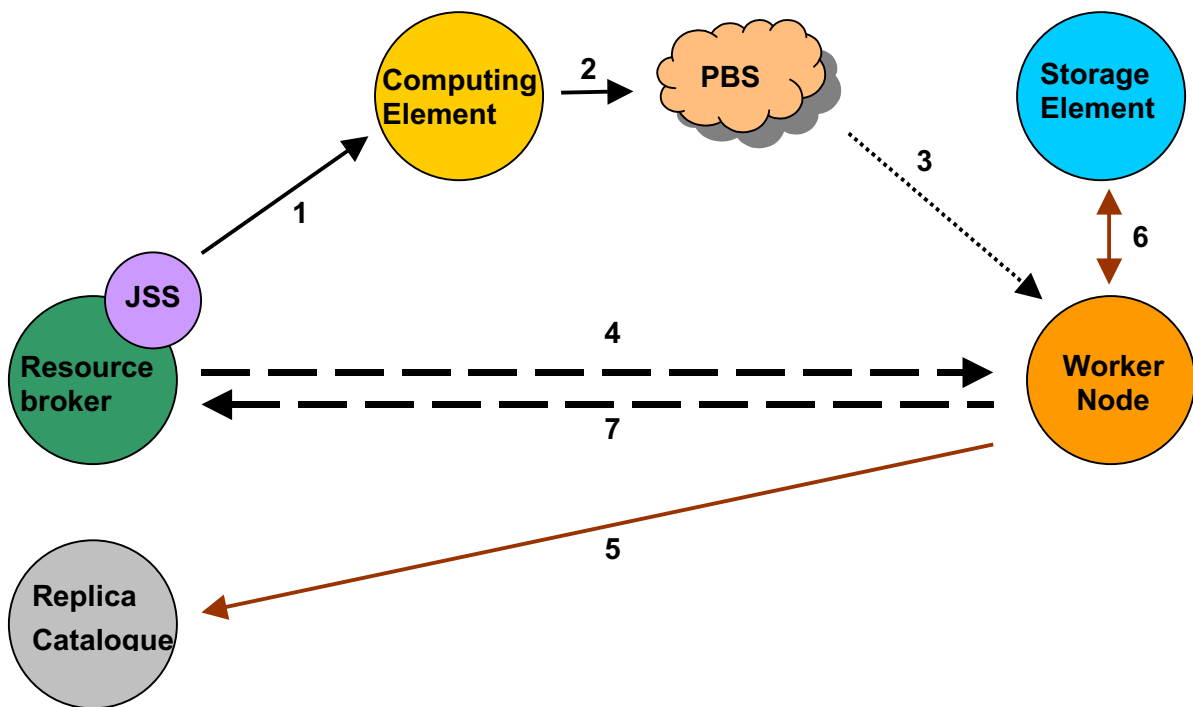


Figure 11 – job submission and sandboxes transfer

The above diagram shows a job submission and the input and output sandbox transfer. The sequence of actions is as follows:

1. The job is submitted to the gatekeeper by the JSS.
2. The gatekeeper receives the job and after authentication sends it to the PBS batch scheduler.
3. The batch scheduler starts the job in a free WN.
4. The Job wrapper script uses globus-url-copy (gsiftp) to transfer the input sandbox files from the RB machine to the WN.
5. The WN may contact the RC to:
 - a. Register/Unregister files in the RC database.
 - b. Obtain the location of physical replica files.
6. The WN may contact a SE to store or retrieve files. This can also be done in the context of the Replica Manager software in order to create or delete a file replica.

7. The Job wrapper script uses globus-url-copy (gsiftp) to transfer the output sandbox files from the WN to the RB machine.

2.4. GDMP, REPLICATOR AND THE REPLICATOR CATALOGUE

There are currently three components involved in the file replication process. The Replicator Catalogue is a database containing information about the location of file replicas. GDMP and the Replicator are two different packages to achieve the same goal - replicate files.

2.4.1. REPLICATOR CATALOGUE

The Replicator Catalogue (RC) is the fundamental building block of the replication service. The RC keeps track of the multiple physical file copies (replicas) in Storage Elements, mapping them into a single logical name. The RC is basically a LDAP server containing the description of the logical files. For each logical file the following information is stored:

- Logical File Name (LFN).
- Physical File Name (PFN), many PFNs may exist for a single LFN.
- File metadata such as size, timestamp, ACLs, master flag and file type.

A logical file name is used to identify a set of identical physical files in different storage locations (SEs). LFNs and PFNs must be globally unique hence a physical file cannot have more than one logical name.

Each time a file is copied to a SE or replicated between SEs it should be registered in the RC so that later it can be easily located for further usage by the same or other users. For this purpose both gdmp and the replicator interact with the RC. GDMP and the replicator are two systems to replicate and synchronize file replicas they are deployed in storage elements. However the replicator has broader capabilities.

Although a distributed RC could be deployed, currently a central RC per Virtual Organization is recommended. However the RC structure will likely evolve into a hierarchical distributed architecture with possibly:

- A top RC per VO containing the mapping between LFN's and site RC's.
- A site RC containing the mapping between LFN's and local SE's.
- A SE RC containing the local mapping between the LFN and the PFN.

2.4.2. GDMP

GDMP (Grid Data Mirroring Package) is a data replication and mirroring software. GDMP is a client/server file replication software system initially designed to replicate Objectivity database files between storage locations, however the newer versions can work with arbitrary file types.

The replication process is based on the producer/consumer model where each data production site publishes a list of the newly created files to the consumer sites making them

available for transfer. To ease the synchronization process, GDMP replicated files are read-only.

GDMP works through a user command and does not provide a programming API. The command interface provides the following capabilities:

- Subscribe to a remote site to get information when new files are created and made available.
- Publish new files making them available for transfer by remote sites.
- Transfer files from remote locations to the local site.
- Obtain a remote file catalogue for recovery.

The GDMP architecture is explained in the next diagram where the four main components of GDMP are shown.

- The Request Manager generates requests on the client side and interprets them on the server side.
- The Replica Catalogue is wrapper over the Globus Replica Catalogue, and is used to make available the information about existing file replicas in the Grid.
- The Data Mover transfers files in a secure, fault tolerant and efficient way using GridFTP.
- The Storage Manager interacts with external mass storage systems (MSS) triggering file-staging operations between the MSS and disk space when needed.

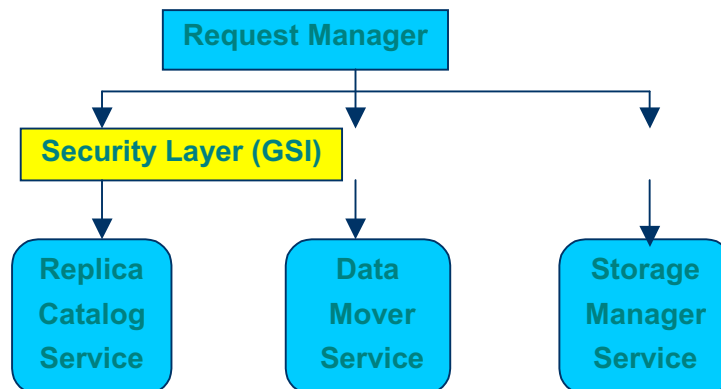


Figure 12 – GDMP architecture

2.4.3. EDG REPLICATOR MANAGER

The EDG Replicator Manager (RM) is a file replication software similar in function to GDMP that can also be used in a CE or UI to transfer files to or from SEs. The RM will include in the future a replica optimiser component. The RM can be used to control the creation, moving and deletion of file replicas and also for the update of the Globus Replica Catalogue.

The EDG Replicator Manager is based on the Globus Replicator Manager with several additions to fit into the DataGrid environment.

The EDG Replica Manager provides user commands and a programming API for:

- Register/unregister a file in the replica catalogue.
- Copy a file without registering it.
- Copy and register a file.
- Replicate a file between storage elements.
- Delete a file.

The Replica Manager interacts with the Replica Catalogue as in the following diagram where two RMs register and unregister files into the RC.

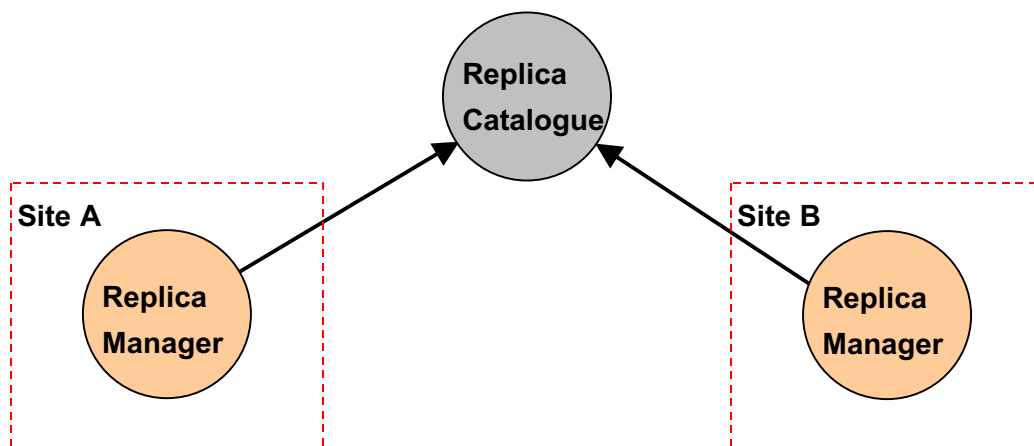


Figure 13 – Replica catalogue and replica manager interaction

Operations over the Replica Catalogue are always started from the Replica Manager side.

2.5. STORAGE ELEMENT

A Grid Storage Element is the generic name for any storage resource that includes a Grid interface ranging from large Hierarchical Storage Management Systems such as HPSS or CASTOR to disk pools. The SE main goal is to provide storage that can be accessed by Grid applications.

Authentication through GSI is required for performing SE operations such as data access, file transfer, file replication, file removal, file registration and file unregistration. Therefore all users wishing to use a certain SE must have their distinguish name present in the local gridmapfile.

Each SE has replication software that allows the creation and management of file replicas. This is done through the GDMP or the Replica Manager components. The replica management software interacts with the Replica Catalogue in order to keep the replication catalogue updated.

Data access from applications running in worker nodes can be done through:

- Spitfire an EDG Grid middleware for relational database access. Contains both a client component and a Java server component that interfaces with a RDBMS. Spitfire provides a uniform way to access many RDBMS systems through standard protocols and well-published interfaces.
- GridFTP the Globus File Transfer Protocol implementation supporting GSI authentication, partial file transfers, negotiation of TCP buffer/window sizes, file transfer monitoring and parallel data streams for faster file transfer.
- RFIO the CERN input output software package that allows remote high performance file access.

Complete file transfer between Storage Elements or between Worker Nodes and Storage Elements can be accomplished through the usage of GridFTP and the tools described in the previous section.

2.6. INSTALLATION SERVER

In a Grid environment the effort to configure manually hundreds of systems per site is not acceptable and clearly there is a need for a global automated installation and configuration system capable of keeping all sites updated and synchronized, running the same middleware properly configured, thus avoiding unpredictable failures caused by incorrect system configurations. Assuring that all systems run the correct software is essential to make the testbed stable for the intended target applications.

The most common method has been to clone the installation across systems. Small differences in the configuration can be implemented by running configuration scripts. However this method has problems when frequent updates are required, when the hardware is heterogeneous or when in spite of the small changes needed, the number of systems to install is too high. In these cases a network automated installation and maintenance system is a better approach.

DataGrid has study two interim solutions (for testbed 1) one based on the LCFG installation management tool and the second based on an image installation system. Currently only the LCFG tool is being used due to the lack of man power to support the image system and also because tests have shown that there is no major performance difference between both installation methods.

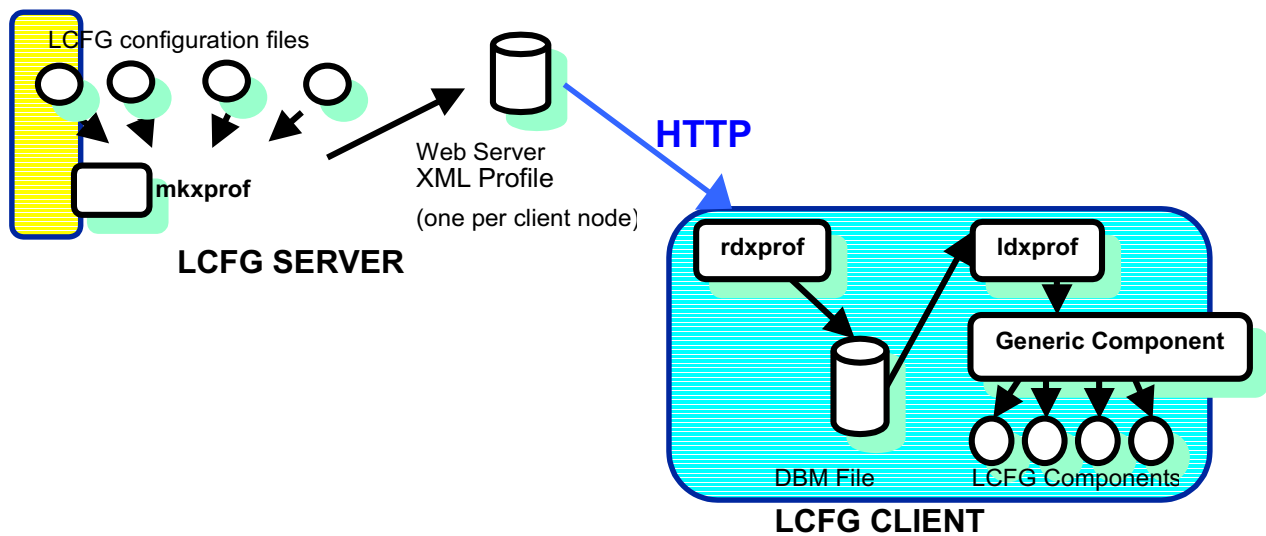
2.6.1. LCFG

LCFG is a system to automatically install and manage the configuration of large numbers of Unix systems. It is particularly suitable for environments with very diverse and rapidly changing configurations. LCFG is developed by the division of informatics of the University of Edinburgh. DataGrid has provided significant input to the LCFG team.

LCFG provides a configuration language and a central repository of configuration specifications from which actual systems can be automatically installed and configured. Changes to the configuration specifications trigger the update of the corresponding systems. The system has the capacity to manage around 1000 nodes from a central installation point.

The source files containing the configuration information are compiled into profiles, each one corresponding to one system. Profiles are made available to the systems through a web server. When information in a profile is modified a notification message is sent to the corresponding system that in response retrieves the new profile from the web server. Each system also polls periodically the web server to make sure it hasn't missed any profile update notification.

The source files for the profiles are written by using key value pairs containing a resource variable and its corresponding value. Resources are written in a similar way to the X resources, they contain the component name, the item in the component to be affected and optionally the system name. The source files also make use of the C pre-processor to implement simple inheritance. Inheritance is implemented by including pre-defined source files making the creation of new profiles easier. Source files are then converted into XML profiles and published in a web server. One XML profile is created for each client node to be installed.



The diagram above describes the LCFG architecture. On the top left the LCFG server contains the configuration files that are used as input to the mkxprof utility that generates profiles in XML and populates the web server. On the client side rdxprof is used to transfer the client profile to a local DBM file. Component scripts use ldxprof to retrieve information from the DBM file and update the LCFG components accordingly.

Each subsystem on a host has a controlling component script that performs the actual system update by taking the required actions to implement each new profile. There is a special component script that is responsible for maintaining the software packages updated in each system according with the central specifications. The package management profits

from the LCFG inheritance mechanism thus allowing the use of a pre-defined package list and specify replacements to it depending on each system requirements.

LCFG also performs the initial system installation. This is accomplished by booting a minimal system from floppy or from the network. Once the minimal system has booted the local disk is partitioned and the RPMS are installed in the same way that the packages update is performed. Once the packages are installed the system reboots and the LCFG components perform the remaining configuration.

Network boot is essential for unattended installation. LCFG uses DHCP and PXE for network boot, this implies the deployment of a server providing these services and the presence of PXE network boot capability in the systems to be installed.

2.7. VIRTUAL ORGANIZATIONS

Virtual organizations are logical views of the Grid communities organized by area of activity. DataGrid has built ten VO's one for each research area they are given below:

WP6	For DataGrid WP6 workpackage members.
Iteam	For DataGrid Integration Team members.
Atlas	For Atlas CERN/LHC High Energy physics experiment members.
Alice	For Alice CERN/LHC High Energy physics experiment members.
CMS	For CMS CERN/LHC High Energy physics experiment members.
LHCb	For LHCb CERN/LHC High Energy physics experiment members.
BaBar	For BaBar High Energy physics experiment members.
Earth Obs	For DataGrid WP9 workpackage members.
Genomics	For DataGrid WP10 workpackage members.
Medical Imaging	For DataGrid medical imaging users.

In Grid environments the classic authentication method based on local user accounts will not scale. Thousands of users will use thousands of resources across the world, it will be impossible to create personal accounts for all users in all these systems not only because of the administrative work required to make all the account databases synchronized but also because each site has its own characteristics and uses its own authentication methods making user databases incompatible across sites. The Globus Grid middleware relies on X.509 certificates for user and system authentication as well as secure communication. Each user needs a personal certificate signed by a national certification authority recognized by the whole community with which resource sharing is desired.

The authenticity of a certificate can be verified by checking the certificate signature against the certification authority public certificate, therefore only the certification authorities certificates must be installed in all Grid nodes making the authentication process scalable.

However a problem still remains, UNIX systems use usernames and numeric Ids internally hence a mapping between each certificate and a UNIX account is needed. This is accomplished by using a local file that maps each acceptable user certificate to a local UNIX account. To automate the creation of the local Grid map file the concept of a Virtual Organization (VO) has been introduced. A Virtual Organization is basically a database of users belonging to a community that wishes to share its resources between their members. The database contains the distinguish name of each user certificate and can be used by client software to automatically generate the Grid map file for local authorized communities.

LDAP has been chosen as the software to enable access to the VO database. The VO database has a database manager responsible for its maintenance. The database manager can create new VO's in the database and appoint a manager for each new VO that will be responsible for maintaining the VO members list.

The VO authorization is very likely to change in EDG Testbed 2 with the introduction of a VO Membership Service (VOMS) that will replace the current VO LDAP servers. At this point the VO membership information will be embedded into the user's proxies in the form of an optional attribute.

The VOMS servers receive requests from clients regarding user VO membership; once a request is validated an answer is sent back to the client with authorization information to be included in a proxy certificate. This information contains a list of roles, groups and capabilities for the user. The proxy certificate containing this information can then be generated and used to access the VO resources.

In order to support the VOMS some existing tools such as grid-proxy-init and mkgridmap are being modified. The command grid-proxy-init will receive a new option that will allow the user to select which VO he wants to use. This allows a user owning a single certificate to be member of more than one VO. With VOMS users can also belong to groups and have roles. Several administration enhancements have also been introduced such as: support for multiple administrators, an administration GUI, replicas and traceability. The VOMS authorization data is stored in a relational database. Migration tools to convert the LDAP VOs to VOMS have been developed.

To use the authorization information inserted into the proxy additional support is required on all Grid systems that perform GSI authentication. The additional support is required to interpret the authorization information kept in a non-critical extension inside of the certificate. If the support is not present the authorization information will pass unnoticed and the proxy certificate will be evaluated as if the authorization information was not there.

The support to interpret the authorization information on the resources side will be provided by LCAS (Local Centre Authorization Service). Currently LCAS is a library developed by DataGrid to be used by a modified gatekeeper. LCAS validates the authorisation information by presenting it to a succession of plug-in authorisation modules that will decide whether the user is allowed to proceed. A plug-in to validate the authorization information contained in the proxies has been developed.

2.8. GSI AND PROXY CREDENTIALS

The Grid security infrastructure (GSI) is based on a public key infrastructure (PKI) where users are identified by their distinguish name (DN). In order to prove the user authenticity a certificate containing the DN is signed by a trusted party called a certification authority (CA). The CA signature proves that the object or person that requested the certificate identified by the DN is the correct person or object and that the certificate can be accepted as a proof of identity. Each certificate has a related cryptographic key called the private key. Anything encrypted with the certificate public key can only be decrypted with the private key and vice versa. While the certificate content must be made public the private key content must be kept confidential otherwise a party gaining access to the private key may impersonate the certificate owner. Therefore the private key is encrypted using a password only known by the owner, and is stored in a secure place (user private file, smart card etc.).

Frequently a user must authenticate several times to several resources requiring the user to type the private key password frequently. In the Grid environment proxy credentials are used to solve this problem. A proxy credential is a certificate containing also a private key that is signed by the user certificate and that can be used to perform authentication on the users's behalf. The proxy credential has a short time validity ranging from hours to a few days, and the key is kept unencrypted but protected using regular file system protection. Since the proxy private key is kept unencrypted it can be used for authentication without the need of entering a password.

However sometimes it is important for an application to act on a user's behalf, that's the case with submitted Jobs that need to authenticate themselves to access resources located in other sites. Another similar scenario is when a user wants to access the Grid from a browser installed at a location where his private key is unavailable.

The GSI answer to this problem is the proxy credential delegation. Delegation allows an application to delegate a certificate to other application over a secure authenticated connection. Unfortunately not all applications support the GSI delegation mechanism.

To cover these issues, a credential repository system called MyProxy has been developed aiming to:

- Allow users to access their credentials even from systems that do not support GSI.
- Allow delegating credentials from an application that does not support GSI.
- Remove certificates from applications except when they are needed.
- Scalability. One proxy server should be able to support multiple applications and users.
- Give to the user control over the delegation process.

MyProxy contains a credential repository server and a set of tools to delegate and retrieve credentials from the repository server.

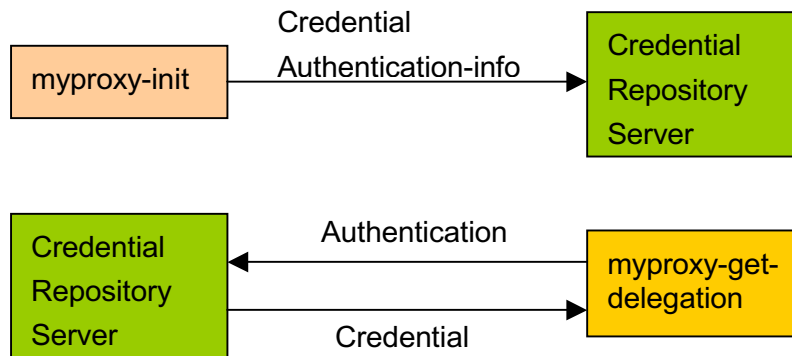


Figure 15 – MyProxy

The diagram above shows how MyProxy works:

- First the user delegates a credential to the repository using myproxy-init. The credential has a lifetime specified by the user and is protected and identified by a user Id and a password.
- Applications and the user can get a credential from the repository using myproxy-get-delegation after proper authentication (using the password for the MyProxy server account).

Portal applications with web interfaces can ask to the user his user Id and password and then use the myproxy-get-delegation tool to obtain a proxy credential. A similar mechanism is used to allow long-lived jobs to obtain and renew their credentials.

2.9. MONITORING

Grid monitoring can be divided in three main areas: network monitoring, testbed monitoring and application monitoring. Network monitoring verifies the bandwidth, delays, jitter and packet loss between sites, testbed monitoring verifies the availability and correct behaviour of required grid services running at each site, application monitoring records the application behaviour and it can be used for code optimisation.

In each of these areas several tools are being developed or improved by ongoing projects. More recently GGF has proposed GMA a Grid monitoring architecture based on a consumer producer architecture where producers register themselves in an information directory and consumers can find producers by querying the directory. This is an attempt to build a generic approach to the Grid monitoring problem where many different types of data probes could be plugged in.

2.9.1. NETWORK MONITORING

Network monitoring is assuming special relevance not only to ensure the quality of the network links, track down possible bottlenecks and establish a global view of sites and services available but also to provide information that can be used for bandwidth prediction and establish the “costs” associated with each data transfer request.

Grid technologies are based on IP networks, hence existing network monitoring tools that collect information such as round trip times, packet loss, jitter and bandwidth can be used to monitor the paths between Grid sites. The difficulties are the scalability and coordination of the network monitoring activities. In a real world with thousands of sites it's not possible to monitor all the paths between all the sites. Hence monitoring activities must be planned carefully in order to not disturb the stability of the network infrastructure itself.

Current tools are:

Traceroute	These are very well-known network management tools that are used to establish instant basic metrics such as reachability, round trip times and packet loss.
Pathchar	
Netperf	
Ping	
Pinger	Measures the round trip time, packet loss and response time variation using ICMP echo (ping) packets. The Pinger project has a monitoring infrastructure focused primarily on the HEP community.
RTPL	Measures round-trip and throughput between locations using ping and netperf. It is intended to make measures on the user perspective between systems where the users have an account. It is not intended to measure the path behaviour but the end-to-end behaviour.
Iperf	Measures the maximum TCP bandwidth, delay, jitter and datagram loss, it is based on Pinger but uses TCP instead of ICMP.
UDPmon	Measures bandwidth, packet loss, jitter and time variation between packets using UDP datagrams,
Surveyor	Measures one way delay and packet losses to establish the performance of Internet paths. It is a measurement infrastructure based on IETF monitoring standards.
NIMI	Is a modular measurement infrastructure based on external measurement tools.
MRTG	Measures traffic load on network interfaces through SNMP and provides historical graphs of the network interface usage. It can be configured to monitor any SNMP variable or to obtain monitoring values from external scripts.
Netflow cflow	, Are used to extract and analyse information collected from network devices.
NWS	Is a monitoring and forecasting distributed system tool. It collects

	information about the state of resources such as network and CPU and tries to generate forecasts for a given time interval.
--	---

2.9.2. TESTBED MONITORING

DataGrid WP7 has chosen to develop a tool called MapCenter that is used to monitor system reachability and the availability of services such as Globus GRAM, GSIFTP, MDS, RB and others running in the same systems. MapCenter provides a global view of the grid sites and historical data about the status of each system. However the MapCenter approach is still very network oriented in the sense that verifies the presence of remote services in predefined TCP ports but it does not verify whether those services are behaving as expected.

Some of the network monitoring tools can also be used for testbed monitoring namely Ping can be very useful to detect systems reachability.

Current tools are:

MapCenter	Provides a global view of grid sites using: Ping to detect connectivity. TCP to detect the presence of processes listening on specific ports. Grid information systems to check for the presence of services.
Nagios	Is a modular distributed monitoring system based on plug-ins that collect information and provide it to the servers. Servers can be organized hierarchically. Many plug-ins already exist.

2.9.3. APPLICATION MONITORING

Application monitoring is mostly valuable for developers helping them to detect application bottlenecks and optimise the code. For the test and validation testbed the main concerns are the network and testbed monitoring, hence application monitoring will not be considered at this stage.

The metrics provided by application monitoring may constitute in the future a useful source of information to detect performance problems and verify improvements in the middleware. This can be achieved by comparing reference application performance results with actual measurements. Some of these issues will be briefly covered ahead.

3. THE CROSSGRID TEST AND VALIDATION TESTBED

In CrossGrid there will be separated testbeds for development, validation and production. However initially a single testbed prototype will be built. This will allow efforts to be concentrated in a single infrastructure reducing the deployment time. As the initial testbed becomes more stable it will become the production testbed and a testbed dedicated for test and validation of new middleware will be deployed separately.

The main goal of the test and validation testbed is to reproduce a real production testbed as close as possible creating an isolated environment where new middleware releases and components can be tested without disturbing other project activities. To achieve this goal the testbed must be auto sufficient. Unfortunately this will have a resource duplication impact on the sites involved, production systems and test systems will need to be deployed and maintained separately, this is felt as a major requirement for accurate middleware testing.

The architecture of the test and validation testbed will be basically dictated by the middleware being tested. The CrossGrid goal is to develop new applications, services and tools maintaining compatibility with existing middleware such as Globus and EDG therefore enhancing the capabilities of these distributions and contributing to extend the Grid to new sites. For this reason and because CrossGrid middleware is still being developed the first CrossGrid production testbed will be completely based on Globus and EDG middleware. Hence the architecture of the test and validation testbed here described is strongly based on the EDG middleware.

Because of the complexity involved in the configuration and test of the middleware components the validation testbed should be kept reasonably small. The main site will be deployed at LIP in Lisbon and will be followed in a second stage by the addition of sites in Greece, Poland and Spain. A second LIP site hosted at the LIP facilities in Coimbra might be added in 2003. The deployment of a second LIP site will allow testing configurations that include more than one site per organization. As the complexity and number of components to be tested grows it might be required to add more resources to the testbed. The addition of more sites will be carefully evaluated and should only be done when enough experience has been gained in the management and coordination of the test activities and the testbed itself.

3.1. TESTBED COORDINATION AND SCHEDULING OF TEST ACTIVITIES

The configuration of the validation testbed will be highly dynamic changing according with the characteristics of the middleware being tested. In certain cases testing different components in parallel may even require incompatible configurations. This will require careful test scheduling or even a "split" of the validation testbed where some computing resources will be running different middleware versions. The allocation of computing resources will be also dynamic depending on the tests. Still a basic configuration per site is foreseen with a set of fully dedicated systems to which more systems could be added on a need basis. Strong coordination will be required to avoid problems and false test results caused by incompatible configurations.

The main site will provide the central services required for the testbed operation. Since the remaining sites will depend on these services special care and efforts should be taken to ensure a fast test and deployment of these central services for each new middleware release.

The test scheduling and distribution will take into account the dependencies of the components to be tested. For instance components that don't require any of the central grid services may be tested in parallel with the first tests at the main site. In other cases, duplicate some test activities might be useful. This is especially true for critical components that need to be thoroughly tested. In this case some tests may be carried out in parallel in two or more sites including the main site (LIP).

The CrossGrid middleware will make use of middleware developed by Globus and DataGrid. Collaboration in activities aiming to test middleware shared in common with other projects is expected. Collaboration with the DataGrid test team has been started recently with the evaluation of the GDMF data replication package performed in cooperation with the DataGrid test team, and involving several DataGrid sites and also including LIP. Therefore coordination will be required not only between CrossGrid partners involved in Task 4.4 but also with other test groups working in external projects. A web site for the coordination of CrossGrid test and validation activities is being created.

3.2. CURRENT TESTBED STATUS

As planned the main test and validation site at LIP is operational. This site will provide the main Grid services required for the first crossgrid testbed release, including a RB/LB server, VO server, MyProxy server and RC server.

The main site is actively involved in:

- Test of the services required for the crossgrid testbed in cooperation with other crossgrid sites already running.
- Test activities in the context of the CrossGrid / DataGrid collaboration on middleware testing.
- Integration tests between CrossGrid and DataGrid.

3.2.1. CROSSGRID ACTIVITIES

LIP is providing several central Grid services to be used by the CrossGrid project. This is a temporary measure while the testbed architecture is being defined, the first sites deployed, and the basic middleware tested. At the same time experience must be obtained to provide the same services in the context of the Task 4.4 where LIP will hosts these services for the **test and validation testbed**.

In this context a VO server hosting crossgrid VO's has been established. This is an LDAP server containing the distinguish names of CrossGrid user certificates. The objective is to

provide a **crossgrid** VO making possible to build the authentication databases for the CrossGrid systems independently from the DataGrid VO servers.

The system hosting the VO server is called **grid-vo.lip.pt** and the port number for the VO server is 9990. Users are being added to the VO with the help of EDG modified VO management tools. Two VO's have been created, the "crossgrid" VO and the "gdmpservers" VO. The "crossgrid" VO contains the distinguish names of the CrossGrid users and is required for user authentication while the "gdmpservers" VO contains the distinguish names of CrossGrid GDMP servers and is used for authentication of replica operations between storage elements. The following table lists the existing VO's and corresponding groups.

VO Name	VO Group	Description
crossgrid	testbed1	CrossGrid users.
gdmpservers	apptb	Production GDMP servers.
gdmpservers	devtb	Development GDMP servers, currently not used.

In order for CrossGrid sites to recognize and accept the crossgrid VO's some changes to the standard LCFG profiles for all CE's, SE's and WN's are required. To help the implementation of these changes LCFG configuration files have been created. However some manual changes are still required namely to the configuration file that contains the list of the VO's recognized by each local system. This is required since this step may depend on each site policies to accept new users; also the order in which the VO's are specified may change the rights of each user. For instance one user registered in two VO's will be mapped to a local group that depends on what VO appears first in the list of authorized VO's. This is most important for sites that accept both CrossGrid and DataGrid VO's.

A Resource Broker has also been established at LIP, and has been configured to accept both CrossGrid and DataGrid VO's. This service enables CrossGrid users to perform job submissions to four CrossGrid sites in three different countries. These sites have successfully configured their gatekeepers with EDG 1.2. The next table shows the gatekeepers currently registered into the RB. More gatekeepers are currently under test and will be added to the RB once declared stable.

Gatekeeper	Site	Country	Job manager
------------	------	---------	-------------

Ingrid02.lip.pt	LIP Lisbon	Portugal	PBS
ce001.crossgrid.fzk.de	FZK Karlsruhe	Germany	PBS
bee001.ific.uv.es	IFIC Valencia	Spain	PBS
cgnode00.di.uoa..gr	Demokritos Athens	Greece	PBS

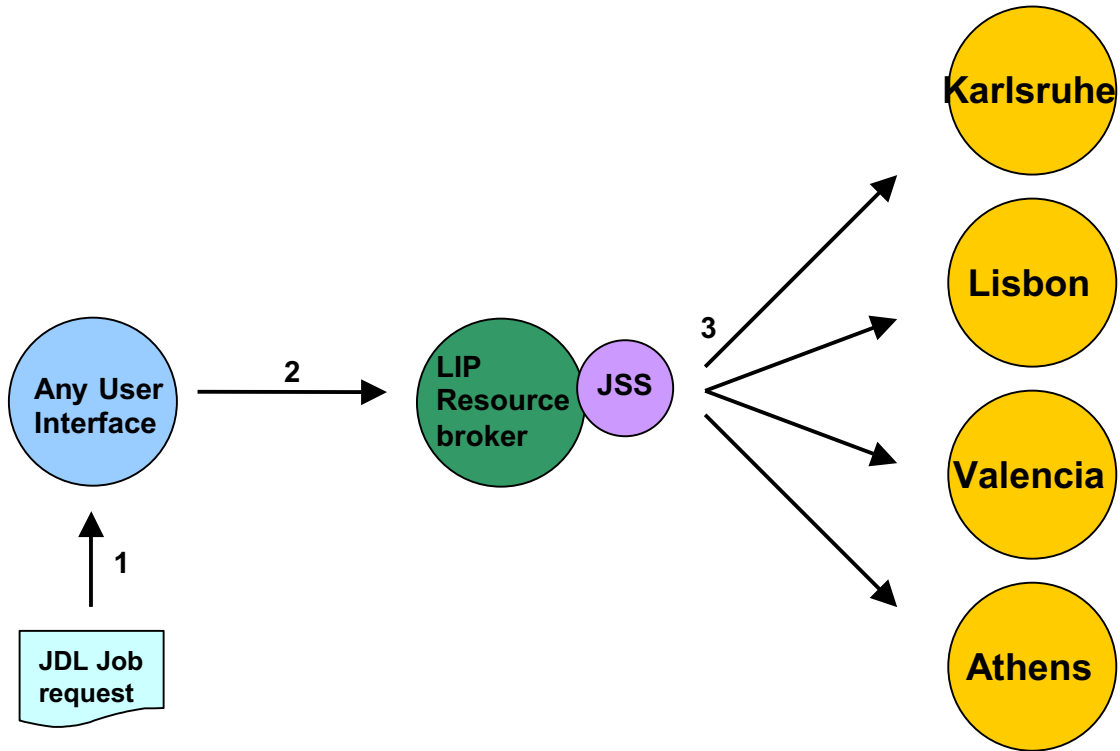


Figure 16 – The CrossGrid RB and the Gatekeepers

In order to submit jobs through the RB minor configuration updates to the each UI are required. A CrossGrid specific LCFG configuration file has been produced to install or update automatically any UI to use the CrossGrid RB.

Several successful test jobs have already been submitted through the RB. CrossGrid users in Germany, Greece, Portugal and Spain have performed these job submission requests using properly configured UI's.

An authentication proxy server is also available for long lived jobs at **Ingrid07.lip.pt**. This service is required for long duration jobs running in the Grid allowing them to automatically renew the GSI proxy when the proxy under whom the job was submitted has a lifetime smaller than the expected job duration.

Work is currently under way to test a recently deployed RC for CrossGrid. The RC is expected to become available to CrossGrid users soon, and it will allow the location of

replica files in CrossGrid SE's. This is an essential step for the use of the RB scheduling capabilities based on the JDL specified data files.

A web site providing information on how to configure grid systems to use these services has been created at LIP (<http://www.lip.pt/computing/projects/crossgrid/crossgrid-services>). This is the main point of reference for the status of these services. The site contains examples of the modifications necessary to use the services, usage examples and some guidelines to help on tracing possible problems.

3.2.2. ACTIVITIES WITH DATAGRID

LIP is representing CrossGrid as a member of the DataGrid test team. The LIP site has also been following the evolution of the EDG middleware since the beginning of the project. The first integration tests of the LIP site with EDG version 1.2-Beta have been accomplished in June 2002 in the context of the EDG development testbed.

Currently the site is integrated into the EDG production testbed. LIP is currently actively testing the EDG version 1.2 features including:

- LCFG installation software.
- EDG WMS job submission system.
- File transfer with GridFTP.
- The replica management packages GDMP and RM.
- The replica catalogue.
- The resource broker.

These tests in coordination with DataGrid are being performed locally and across sites namely between LIP in Portugal, CERN in Switzerland, UAB in Spain and others. The replica catalogue being used in these tests is the Atlas RC located at CNAF in Italy.

Other CrossGrid sites (FZK and IFIC) have also joined the DataGrid testbed enabling cross tests between both testbeds.

3.2.3. MAIN SITE CONFIGURATION

The configuration of the test sites is expected to change often based on the evolution of the middleware. Therefore the configuration of the main test site (or any other test site) will never be complete.

So far the emphasis has been put into testing and gaining experience with the common EDG and Globus services required in all sites. However most of the central Grid services required for a testbed have also been deployed, and made available to all CrossGrid sites.

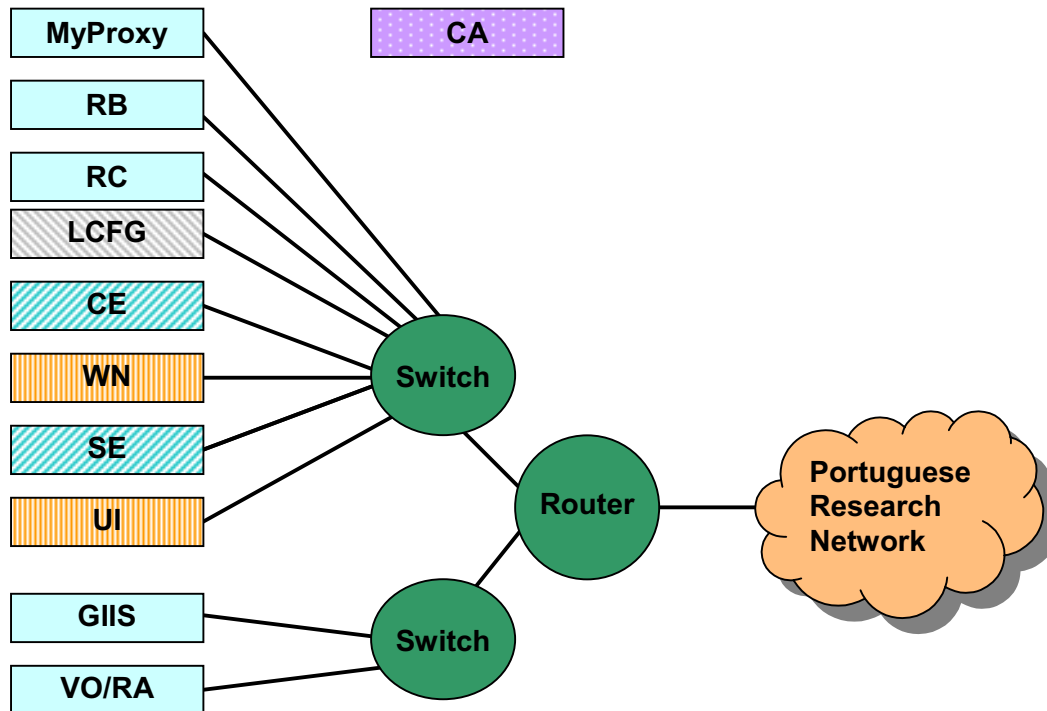
The current configuration of the main test site at LIP is as follows:

Node name	Description
Ingrid01.lip.pt	LCFG installation server.
Ingrid02.lip.pt	Gatekeeper.
Ingrid03.lip.pt	Storage Element.
Ingrid04.lip.pt	Worker Node.
Ingrid05.lip.pt	User Interface.
Ingrid06.lip.pt	Resource Broker.
Ingrid07.lip.pt	Authentication Proxy.
Ingrid08.lip.pt	Replica Catalogue.
grid-vo.lip.pt	Virtual Organization server.
Innet07.lip.pt	GIIS server (not used due to the MDS tree problems).
ca.lip.pt	CA/RA server.
CA SIGNER	CA signing machine (is kept offline).






All systems are physically installed at the LIP Computer Centre in a computer room with proper cooling and power protection through an UPS. Most systems are Pentium 4 machines at 1.7 and 1.8 GHz with 512MB of main memory with the exception of the national GIIS that is running on a Pentium III 800MHz with 256MB of memory, the user interface that is running on a Pentium II 600MHz with 256MB of memory and the CA signing machine that is running on a PentiumPRO 200MHz with 64MB of memory. The CA signing machine is kept offline without any network connectivity.

Two network switches are used, one for external services such as the Portuguese national GIIS server, the RA services and the VO server and a second switch (Cisco Catalyst 4000) for the remaining services that are more internal and require higher performance. Both switches are connected to the LIP core router (Cisco 7204 VXR) that provides the external connectivity to the Portuguese National Research Network through an ATM link over optical fibre with a capacity of up to 34Mbits/s.

With the exclusion of the systems holding central services the current LIP site configuration that can be seen in the next diagram can be used as reference for other sites in the deployment phase. A simpler topology excluding the central services is shown further ahead.



Meaning

-  Systems with inbound and outbound connectivity.
-  Systems with outbound connectivity.
-  Systems without inbound or outbound connectivity.
-  Systems hosting central services with inbound and outbound connectivity.
-  Systems without network connectivity.

Inbound connectivity means that connections from the Internet to local system are allowed.
 Outbound connectivity means that connections the local systems to the Internet are allowed.

Figure 17 – LIP site configuration

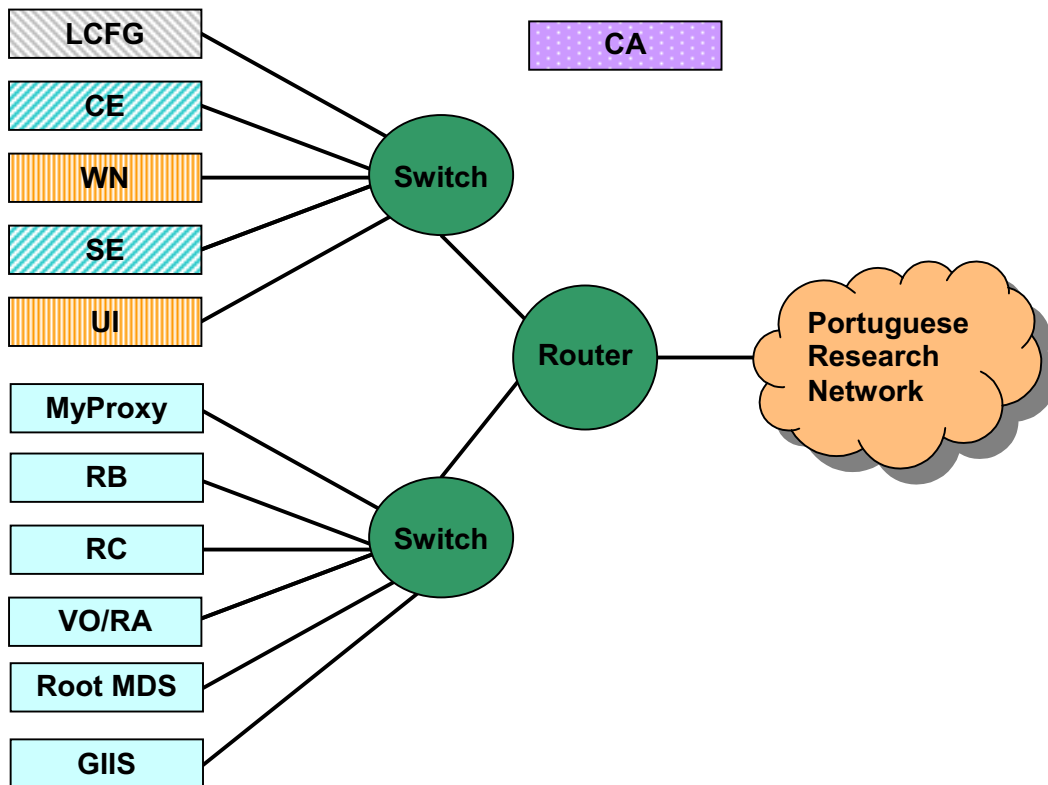
Although the current configuration shows the RB, RC and MyProxy servers in the same switch as the local computing services, this configuration will likely be modified and these servers will be moved near to the GIIS and VO servers. The main reasons for the current configuration are:

- Easier system installation since the servers are in the same network as the LCFG installation server.
- The ability to test the servers installation without the interference of the router ACL's.

The main benefits for changing the current configuration and move the servers to the second switch are:

- Improve the systems security by putting services with high exposure in a separated network.
- Improve performance by applying different policies on each router interface and keeping external traffic separated from internal traffic.

The diagram below shows how this modified configuration will look.



Meaning






-  Systems with inbound and outbound connectivity.
-  Systems with outbound connectivity.
-  Systems without inbound or outbound connectivity.
-  Systems hosting central services with inbound and outbound connectivity.
-  Systems without network connectivity.

Figure 18 – Future LIP site configuration

As explained before the first CrossGrid testbed is a unified infrastructure that will be split into three testbeds for production, development and validation. This means that the current LIP site infrastructure will be in the future partially duplicated, with systems dedicated to test and

validation activities and systems dedicated to the production testbed. Support for a development testbed is not foreseen at the LIP site.

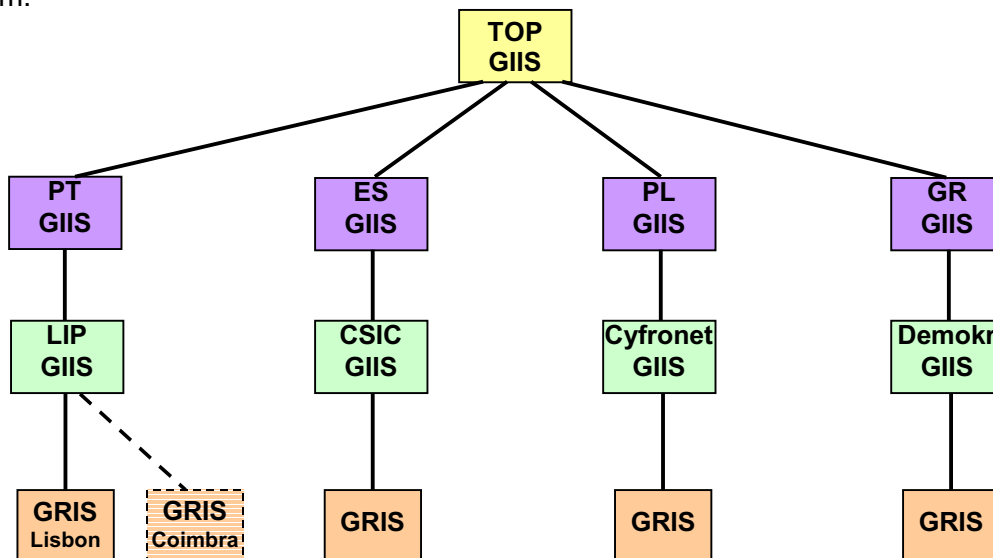
Possibly the central services provided today by LIP to the whole CrossGrid project such as the RB/LB, VO server, RC, MyProxy server and GIIS (not used currently) will be moved to another production site in a future testbed release. The systems at LIP holding these services will then be reconfigured to provide the same services but only for the test and validation testbed.

3.3. INFORMATION SYSTEM

The root of the MDS information system tree for the test and validation testbed will be hosted at the main test site (LIP). The Resource Broker is the major user of the MDS information therefore for performance and reliability reasons the root MDS should be kept near the RB. This setup will contribute to reduce the occurrence of possible network related problems that might interfere with a correct evaluation of the MDS and RB behaviours. The testbed root GIIS has not been deployed yet due to problems in the MDS software that affect the propagation of information in the tree.

3.3.1. INFORMATION TREE TOPOLOGY

The first implementation of the full MDS tree will be based on a single unified tree. The top GIIS will be hosted at the main site (LIP). Country GIIS servers will register to the top GIIS and below them the organizational GIIS servers will register to each corresponding country GIIS. Each organization involved in the tests will provide one or more Computing Elements running a GRIS service that will register to the organizational GIIS as in the following diagram.



This way the validation testbed will mimic a true MDS tree including country GIIS servers. However each country will need to provide a country GIIS dedicated to the validation testbed hosted at a test site.

A second approach could be the deployment in parallel of one top GIIS per virtual organization, this way finding resources for specific applications could be much faster. The penalty will be a much higher complexity since organizational GIIS servers will need to register themselves both to the top GIIS and to each supported VO GIIS. Such an arrangement can be seen in the next diagram.

Basically the MDS tree topology will have to follow the expected production testbed MDS tree topology in order to test and validate it before actual deployment. However it is also expected that the test and validation testbed will be used to test other possible configurations. In this sense different tree topologies that might provide performance or reliability advantages will also be tested.

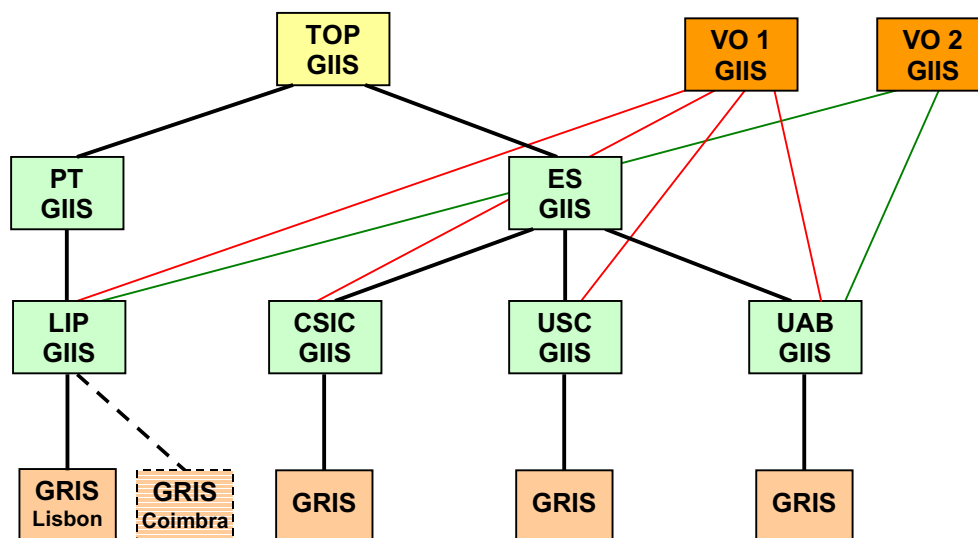


Figure 20 – MDS TREE WITH GIIS SERVERS PER VO

3.3.2. INTEGRATION WITH OTHER MDS TREES

One of the short-term objectives of CrossGrid is the integration with the DataGrid testbed. In order to achieve this goal both MDS trees must be somehow merged. This is a technical and an organizational issue since some CrossGrid sites are also participating in DataGrid and therefore are already inside the DataGrid testbed infrastructure. However the majority isn't. The addition of these new sites to DataGrid may take a long time and some sites might not even be interested in joining. This is especially true for sites not involved in High Energy Physics applications. Issues such as the user certification and cross access to resources will make a complete merge of both testbeds a challenge.

The strategy described in the next paragraphs has the objective of allowing sites interested in DataGrid to join it in parallel with their participation in CrossGrid. Three steps are required:

1. Sites involved in DataGrid will join the DataGrid MDS tree.
2. A CrossGrid MDS tree must be built for the project internal use.
3. Sites already in the DataGrid MDS tree will join also the CrossGrid MDS tree staying in both trees in parallel.

This implies that countries participating in both DataGrid and CrossGrid might need two country GIS, one for each testbed. This is especially true when several institutions in the same country have different interests and some want to join DataGrid and others CrossGrid, but it can also be true due to technical differences between the GIS implementations. For instance when one of the projects is upgrading the testbeds to a newer release of the middleware and the other is still not ready to make the same move due to other middleware dependencies.

3.4. THE WORKLOAD MANAGEMENT SYSTEM

A dedicated WMS for the test and validation testbed is required at the main site (LIP) providing job submission services for the testbed. The system must contain a Resource Broker and a Logging and Bookkeeping service running in a single server. As said previously a RB has already been installed at the main site.

In the future, access to the RB will be controlled by only authorizing job submissions coming from users registered in the test and validation VOs. The current RB configuration is open to all CrossGrid users. In order to perform authentication and authorization the RB requires an updated gridmapfile, the recognized CA certificates and updated CRL's. Hence the RB must also run the CRL update daemon.

The CRL update daemon is a process that must be running in all CE and SE nodes that is responsible for retrieving the latest Certificate Revocation Lists from the web servers of all the recognized CAs. The CRLs contain the indexes of the revoked certificates, they are the only mechanism for a CA to cancel a compromised certificate. Therefore the CRLs should be downloaded at least once a day and its restrictions should be applied in the authentication process.

The server hosting the RB and the LB must have the following network access enabled:

- Inbound TCP from the User Interface to allow reception of job management requests, and to transfer the input and output sandboxes to the UI. This means that inbound access to the RB service must be open at least to all testbed sites.
- Inbound TCP access from the Worker Nodes to transfer the input and output sandboxes. This means that inbound access to the gsiftp daemon located at the RB server must be open to all testbed sites.
- Outbound TCP to the Replica Catalogue to obtain the locations of required files.

- Outbound TCP to the GIIS and GRIS servers to obtain information about computing resources, authorized users and SE's near to the CE.
- Outbound TCP to the GRAM services running in the Gatekeepers.

The RB will need to use a MDS tree and a RC therefore these two services must also be deployed to ensure the RB independence from external services.

CrossGrid will also develop a grid resource management system based on scheduling agents that make decisions about where, when and how to run jobs on the grid. Scheduling Agents will be responsible for optimizing scheduling and node allocation decisions. Their basic role is to tell the grid resource manager what to do, when to run jobs and where. The Scheduling Agents (SA's) are being developed to support parallel applications and portals.

A Scheduling Agent will not be a central service like the RB, instead each user will have its own agent that will interact with both the RB and LB. Therefore the SA approach will be compatible with the DataGrid WMS and will improve its job scheduling. This is specially important since the current EDG design of the RB does not cover communication and state synchronization across RBs, this makes real global optimization of the resource usage impossible.

The CrossGrid resource management configuration is still being defined by WP3 (CrossGrid), namely it is not yet clear how many RB's will be deployed. One possibility is to deploy one or more RB's per VO as the following diagram from WP3 shows.

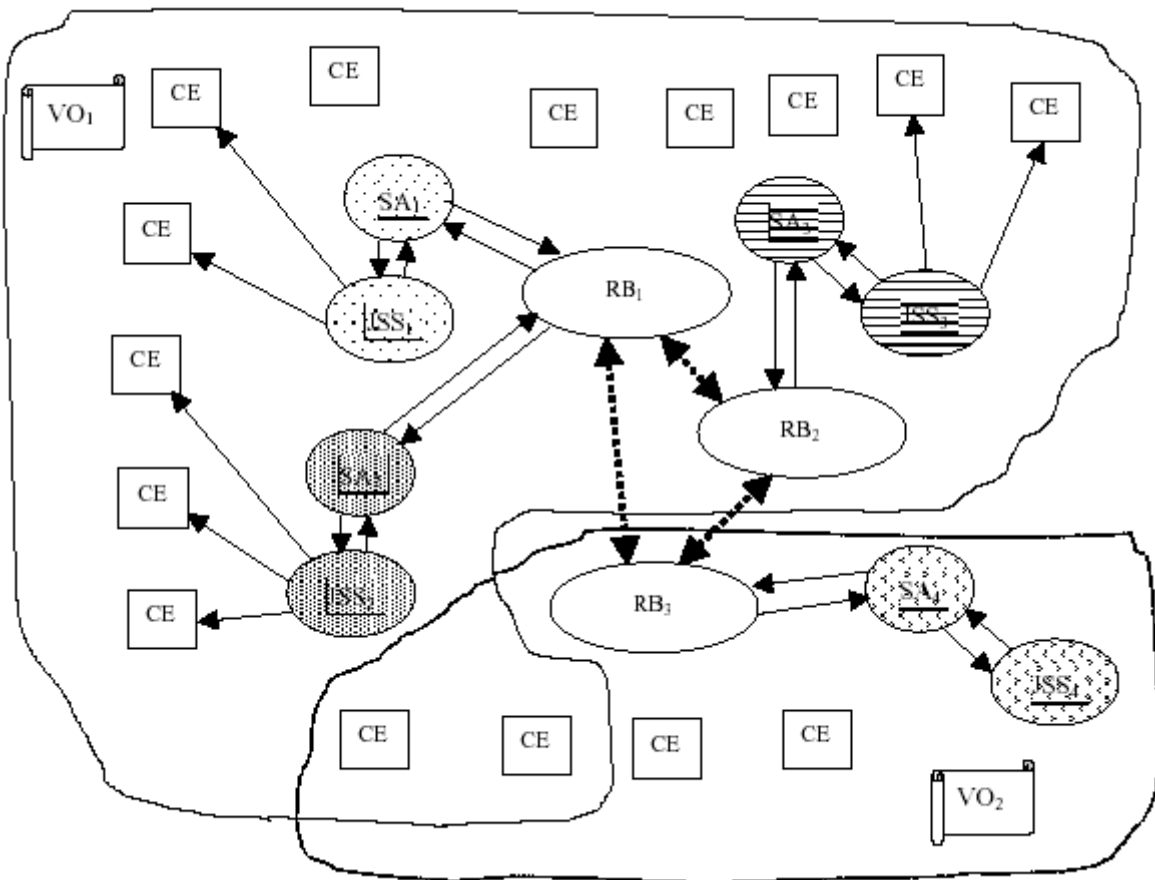


Figure 21 – SA's and RB's with multiple VO's

The diagram shows the interaction of each SA with one RB to obtain the list of resources and the SA interaction with its own JSS to submit the job to the selected CE. For each VO one or more RB's are used to improve performance and robustness. The RB's must communicate between each other to share information regarding the availability of the computing resources.

The resource management system is the most important component in the grid job submission. The architectural choices of WP3 will have a deep impact on the CrossGrid testbeds architecture.

3.5. COMPUTING ELEMENT

The existence of a Computing Element composed by one gatekeeper and at least one worker node is the most basic requirement for a testbed site. Being a basic requirement this should be the first component to be installed and tested at each site after the LCFG server is configured.

The CrossGrid validation testbed Gatekeeper will have basically a Globus GRAM service to allow the submission of jobs both to the local system through a FORK job-manager and to a set of Worker Nodes through a batch job-manager. The batch system to be initially used will be PBS since it is reliable, well supported under Globus and already used in many sites. Other possible configurations will be also evaluated including the usage of the Condor batch scheduler.

The presence of a FORK job-manager is required for testing the GRAM service. This allows job submission problems to be detected and identified more easily without the interference of the batch scheduling system, allowing even to test some of the gatekeeper GRAM functionality without the presence of working nodes. Also the tests performed so far have shown that the FORK job-manager is a very good tool to obtain information about the configuration of remote systems especially when interactive login is not allowed. This can be used to diagnose remote problems and compare site configurations among testbed participants.

The Gatekeeper must also have a GRIS server to make local information available to the Grid. This information contains the CE characteristics, status, (including ranking variables) and allowed users. Only users from the test and validation VOs should be allowed to submit jobs to test and validate the gatekeepers.

If local storage is provided in the CE that is not shared with the SE through a network file system then a gsiftp daemon should also be installed in the gatekeeper to allow remote access to the local user files.

For proper integration of the gatekeeper with the WNs the following restrictions must be fulfilled in each site:

- The password file must be equal across all SE, WN and Gatekeeper nodes in a site. Basically this means that the UNIX UIDs and GIDs must match the same usernames in the gatekeeper, WN and SE. This is required for proper NFS operation.
- The user home directories must be shared through NFS.
- The file containing the mapping between certificates distinguish names and unix user accounts must be equal across all nodes (preferably shared through NFS).
- If pooled accounts are used then the gridmapdir must also be shared across all nodes. When pooled accounts are used a dynamic mapping between distinguish names and previously created user accounts is performed at login time. In order to preserve this mapping across all nodes the gridmapdir directory where these mappings are kept must be shared through NFS across all nodes.
- The directory containing the CRLs must be shared through NFS between the gatekeeper and the worker nodes. This is required because WNs need the CRLs to authenticate the data transfers and only the gatekeeper runs the CRL update daemon.

These same restrictions apply to the SE when the SE directories are exported through NFS to the CEs and WNs. In addition all site nodes using grid services (CE, WN or SE) need to install the root certificates for all recognized certification authorities and update daily the corresponding CRLs.

TCP inbound access is required for the GRAM, GRIS and gsiftp services. When only job submissions through the RB are desired the access to the GRAM and GRIS services can be limited to inbound access from the RB. However the gsiftp service when installed should be available to all sites for direct access.

3.6. REPLICA CATALOGUE AND REPLICA SOFTWARE

CrossGrid will use the Globus/EDG replica catalogue hence this central service will need to be provided at the main site. To reproduce a real production testbed a RC will be provided for each VO. Currently a RC is already under test. The RC services will be used by the replica middleware described below and by applications to:

- Create new logical file entries.
- Update the logical file entries with new physical file replicas.
- Remove logical file entries.
- Update the logical file entries with the removal of physical file replicas.
- Search for physical file locations.

The RC server requires inbound TCP access.

CrossGrid needs to deploy initially both GDMP and the EDG Replica Manager middleware. They are required since the EDG RM is still being developed and the recently released version is not yet completely functional. Furthermore the new RM will require careful testing. Meanwhile GDMP will be used as the replica middleware.

CrossGrid WP3 is also developing a Replica Manager middleware that will complement the functionalities provided by Globus/DataGrid RM. The CrossGrid RM is an advisor for selection of migration or replication policy. Its main goal will be to suggest whether and when a chosen data file should be replicated into the local environment in order to shorten the waiting time for data availability. A sophisticated method will be used to take the right decision. It will be based on the Globus/EDG Replica Manager.

The CrossGrid RM will interact with the Globus MDS, RC, and with the CrossGrid Data Access estimator (to be installed in each SE, see next chapter) to obtain information about data access cost. Portals and applications will use the replica manager. The general diagram of the components involved is as follows:

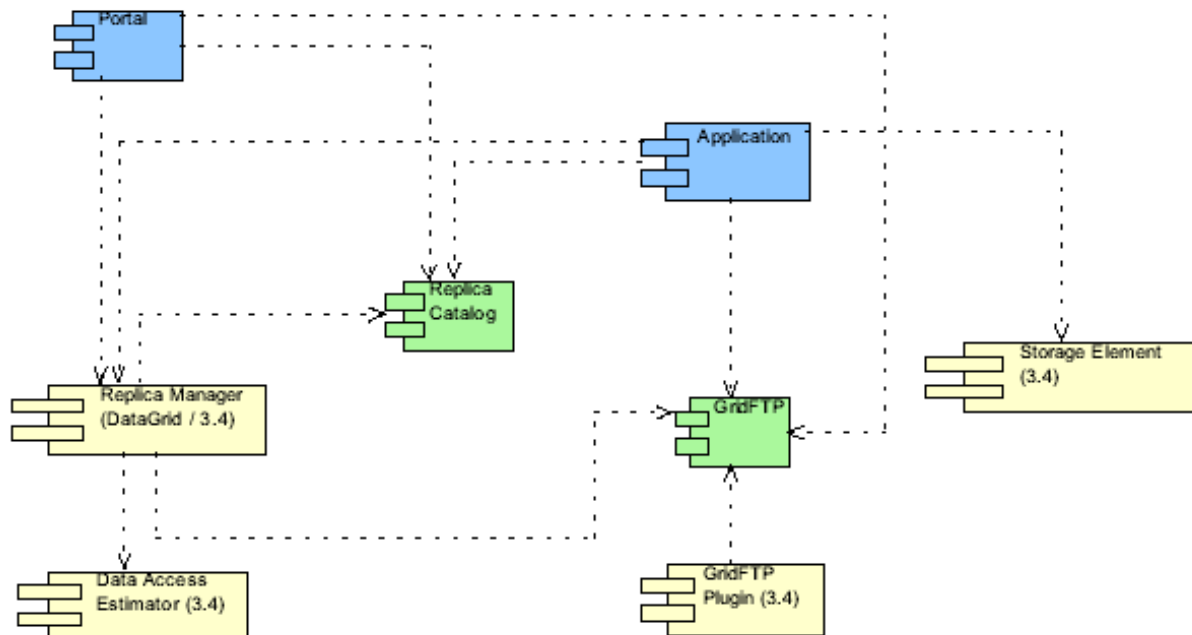


Figure 22 – CrossGrid RM component dependencies

3.7. STORAGE ELEMENT

As explained in the state of the art section the existence of a storage element in each site is essential since large data files should be kept in SEs near to the CE. This is also essential to use the data requirements scheduling feature of the RB where the data must first be available in one or more SEs before the job is submitted to the GRIS through the resource broker. Each test site must mimic a production site configuration and deploy at least one SE. The presence of a SE near each Computing Element is also one extremely important assumption that is made by the Resource Broker scheduling software.

The SE is basically a system that makes available its storage capacity to other systems using several protocols. In the future all these protocols will be grid enabled. This means that all the authentication and authorization will be based on grid certificates and not on UNIX usernames and UIDs. Today some of the protocols are already grid enabled, some are being gridified and others are being used with classic UNIX authorization. The SEs should be installed and configured in such a way that they cover as much as possible all of the protocols that can be used in combination.

Currently the following protocols are used and should be configured:

- GridFTP – A grid enabled File Transfer Protocol.
- RFIO – A library for remote file access being gridified.
- NFS – The standard network file system.

Other data access components are being developed and will be available in the future, namely:

- FALB is a library being developed by CrossGrid to make local and remote file access easier and to give a programmer access similar to UNIX standard I/O functions with additional parameters allowing controlling some special grid features.
- Slashgrid by DataGrid is a grid enabled file system that uses components of the Coda API to intercept I/O system calls and perform access control using certificate distinguish names. A local file system prototype based on an underlying ext2 filesystems been developed. SlashGrid will likely evolve into a remote file system.
- TRLFM (Tape Resident Large Files) is a CrossGrid middleware to be installed on top of existing HSM systems with the objective of improving the data access latency time by splitting large files into smaller pieces that can be staged faster.

Crossgrid is also developing the middleware described below that will be integrated into the SE's:

- A data access cost estimator component that will be able to return several measures of cost like latency, and bandwidth. This component can be used by the Replica Manager to make decisions on which source data file to use to make a replica.
- Support for local site data storage strategy optimisation based on Component Expert Subsystem (CES) architecture. CES will be responsible for choosing the best component (strategy) for data operation using knowledge rules stored in the expert system.

The next diagram shows the interaction between a local storage element, a worker node in the same site, a user interface at a remote site and a second storage element also at a remote site. GridFTP can be used in all the file transfer scenarios while RFIO and NFS are more oriented for communication inside a site LAN.

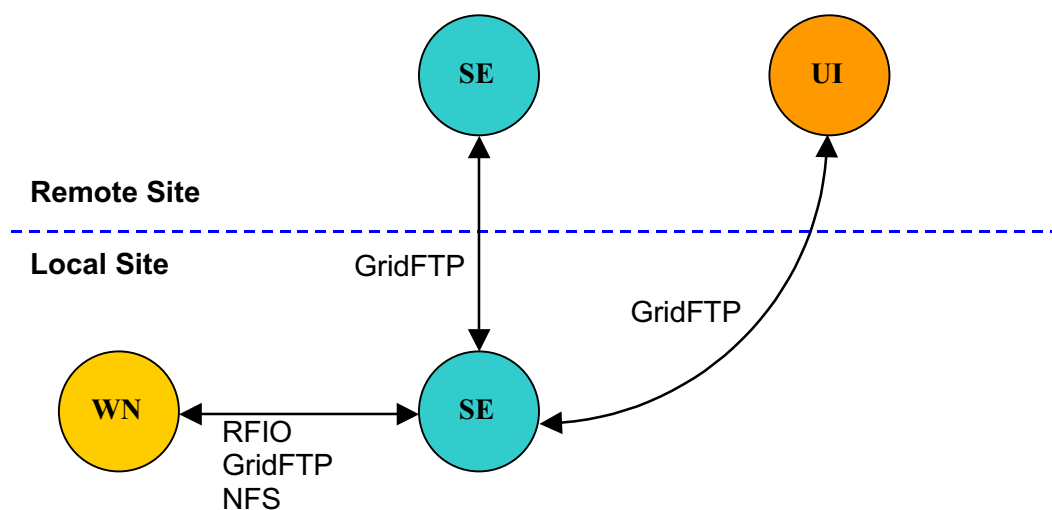


Figure 23 – SE interaction

Another important SE feature is the file replication and synchronization that should be accomplishing through GDMP and the EDG Replica Manager. They are the key components of the Grid storage element. Both GDMP and the RM make use of GridFTP to transfer files between storage elements and keep them in sync automatically.

The following requirements apply to a storage element:

GDMP

- TCP inbound access.
- A GRAM fork job-manager must be configured.
- The gridmapfile must contain mappings for all allowed distinguish names.
- Each specific VO authorization file must contain the allowed distinguish names.
- If pooled accounts are used the gridmapdir file must be shared between the all CE's, SE's and WN's.
- The distinguish name of remote SE's must be registered in the gridmapfile pointing to the local account gdmp.

RM

- The gridmapfile must contain mappings for all allowed distinguish names.
- If pooled accounts are used the gridmapdir file must be shared between the all CE's, SE's and WN's.

RFIO

- TCP inbound access.
- The gridmapfile must contain mappings for all allowed distinguish names.
- If pooled accounts are used the gridmapdir file must be shared between the all CE's, SE's and WN's.

GridFTP

- TCP inbound and outbound access.
- The gridmapfile must contain mappings for all allowed distinguish names.
- If pooled accounts are used the gridmapdir file must be shared between the all CE's, SE's and WN's.

NFS

- Should only be used inside the local area network.
- If pooled accounts are used the gridmapdir file must be shared between the all CE's, SE's and WN's.
- UNIX username to UID's and GID's mappings must be the same between all nodes sharing the same file system (SE, CE, WN).

As it can be seen a SE needs both inbound and outbound network access this means that a valid fixed IP address must be allocated to each SE. Routers and firewalls must be properly configured to allow inbound access to GDMP, GRAM and GridFTP from any IP address, and outbound access from the SE to any IP address.

A standardized Grid enabled access to databases for short lived, small amounts of data and metadata that needs to be accessible to many users and applications is clearly a necessity for many applications and middleware components. Hence middleware components such as spitfire from EDG will be required inside the CrossGrid SE's. This approach can also introduce performance benefits since usually a client only needs a reduced set of information in a database, therefore it is much more efficient to have a database server inside the SE performing the search operations and providing only the required sets of information than transferring the entire database to the WN.

3.8. INSTALLATION SERVER

The software installation server to be used in the first CrossGrid production testbed will be the LCFG-one package as distributed by EDG. This is important to maintain compatibility with the EDG middleware distributions and facilitate the integration process of the two production testbeds. As a consequence the test and validation testbed will test and use the LCFG installation software.

Nevertheless CrossGrid is also looking into other methods of installation and deployment. The installation of grid middleware through LCFG has several limitations, one of them is the necessity of a full reinstallation of the operating system for each client installation. This is felt as a major obstacle to the deployment of grid middleware in a wider basis since the system disks of the existing farm nodes would have to be completely erased and reinstalled from scratch. Grid middleware in order to be successful must be seen as just one more package that can be added to already running systems without major disturbance. CrossGrid is searching for solutions that could make this possible by installing the middleware in one system and exporting the installation to other existing systems in a secure way through AFS.

The first step for the deployment of each site will be the installation and configuration of a LCFG installation server in a dedicated system. LCFG also requires the installation of a web server to export client profiles, a DHCP server for network boot and allocation of IP addresses and a PXE server also for network boot. The DHCP and PXE server should be hosted in the system that also hosts the web server and the remaining LCFG components.

Some attention must be put on the network boot configuration since DHCP and PXE requests from local grid systems can be answered by other DHCP servers in the network, also other workstations in the network may issue DHCP requests that can interfere with the DHCP server used to boot the grid systems, this situation only happens when the grid systems network is shared with non-grid systems. This problem can be avoided by:

- Using a dedicated network connected to a router interface with the router interface configured not to forward broadcasts in any of the directions.
- Reconfiguration of all the DHCP servers. The LCFG DHCP server must only answer requests with MAC addresses belonging to grid systems. The remaining DHCP servers must not answer requests with MAC addresses belonging to grid systems.

This approach may not be possible in sites where the DHCP servers are not under control of the grid test team.

- Using boot floppies. This solution should only be used when no other possibility exists. This is a less flexible approach that requires manual intervention each time a full reinstallation must be performed. The large majority of the production sites will use network boot therefore to recreate a production environment, network boot should also be used at the test sites.

Another situation that requires attention is the need for reconfiguration of the DHCP server once the installation completes and before the reboot of the client being installed. If the DHCP server configuration is not changed, the system will boot again from the DHCP server and a reinstallation will be triggered.

One additional characteristic of LCFG that can also be seen as a limitation is that all management actions to be performed on client systems should not be done directly on the clients but instead through changes in the LCFG client profiles. If they are performed on the client directly they will be rewritten by LCFG. When modifications not supported by LCFG need to be made they can be achieved either through the copy object that allows to copy configuration files from the server or by writing a LCFG object in shell and adding it to the client profile. The new version of LCFG will support easier object scripts in Perl.

Although the configuration of LCFG and its required software can be tedious and difficult it has also advantages in the grid test environment where many system reconfigurations and reinstallations are performed frequently. Once LCFG is correctly configured adding more nodes to the local grid infrastructure or converting systems between functions (ex. from a WN to a SE) can be performed quickly, easily and without many configuration problems.

3.9. CERTIFICATES, VIRTUAL ORGANIZATIONS AND THE PROXY SERVER

The testbed will use the certificates issued by the recognized CrossGrid certification authorities. The test and validation testbed will not impose any new requirements to certification authorities and certificates being issued besides the requirements specified by the middleware being tested.

The certificates that will be used both in the development and in the production testbed will be valid within the test and validation testbed. However the deployment of a test CA not valid outside the scope of the test activities might be necessary in the future to satisfy extraordinary middleware requirements that for some reason can not be satisfied by the production CA's due to policy or software incompatibility. The deployment of a test CA should only be used as a last resort measure since the deployment and maintenance of a CA should be left to security experts and is not in the scope of the test activities.

A dedicated virtual organization must be deployed for the test and validation testbed. This is required to establish an independent testbed infrastructure that will not depend on services running on non-test sites. VO independence is important because new middleware releases may require services from the VO server that might not be compatible with the production VO

servers. Also a validation of the VO software itself will require a full VO reinstallation, reconfiguration and test that is not compatible in terms of service disturbance with a production service.

The VO server is one of the most important central services. The security authentication procedure depends much on the certificate distinguish names stored in the VO LDAP directory. In production environments the VO server should be polled frequently by Grid systems running authentication services to check for updates in the VO members list and reflect those changes in the local gridmapfile databases.

An EDG VO server has been installed at LIP and is being used to support CrossGrid activities. Up to know most of the problems found are related with the documentation quality and have been successfully overcome. The experience gained with these tests will be used to configure and maintain a VO server for the testbed.

One virtual organization named "cgTV" (crossgrid Test and Validation) will be used for the authentication of the testbed users. Two groups will be created inside the VO.

cgTValpha	For middleware testers.
cgTVbeta	For application users.

This is in accordance with the CrossGrid WP4 - *Middleware Test Procedure* document where the test process is described being divided in two phases Alpha and Beta. In the Alpha phase the middleware is tested by a group of teams composed of WP4 members while in Beta phase the testbed will be open to a small group of invited application development experts from WP1 that will exercise the deployed middleware with their applications and test programs. Hence the cgTValpha group will contain the WP4 middleware experts and cgTVbeta group will contain the WP1 applications experts.

Both groups will be hosted in LDAP servers running in the same physical system at LIP. The VO server requires inbound access to the two TCP ports where the two VO LDAP servers will be listening for requests. Routers and firewalls must be configured to allow inbound access to these ports from any IP address.

The MyProxy service has been incorporated in the Globus and EDG releases. It is required for proper RB operation and Grid applications will start to use it, also CrossGrid WP 3 will develop portals and roaming environments that will require MyProxy or a similar mechanism. Therefore MyProxy must be tested and a server must be deployed to test other components requiring it. A MyProxy credential repository server has been deployed at LIP and will become a global service for the validation testbed.

3.10. MONITORING

CrossGrid is developing their own Grid monitoring services and tools in the context of WP3 activities. The information provided by these subsystems can be used to establish the current and past state of the Grid. They are:

- OCM-G: to gather information from user applications and provide it to higher level software components, typically tools such as performance analysis ones.
- SANTA-G: specifically intended to introduce information captured by external monitoring instruments into the Grid information system.
- Jiro-monitoring: will monitor infrastructure components using Jiro technology. Jiro supports network management protocols such as SNMP, thus it seems to be an appropriate approach to the network-related issues within Grid monitoring.

These tools will be complemented and combined with already existing tools. The CrossGrid monitoring tools (WP3) will be used as soon as they become available, however their development and test will take time therefore an interim solution is required for the monitoring of the CrossGrid testbeds. Meanwhile existing monitoring tools will be used to verify and predict the behaviour of the CrossGrid testbeds.

3.10.1. APPLICATION MONITORING

CrossGrid WP2 is also developing tools and benchmarks that have potential interest as test and validation tools. These developments will be followed closely.

Task 2.4 is developing G-PM a software performance measurement tool that enables the user to optimise the performance of the application with support for:

- Measurements of various aspects of the program execution on the Grid (computation time, communication time, data volume, response time), enabling the detection of execution bottlenecks by interpreting the performance data; this part of G-PM tool is realized by Performance Measurement Component (PMC)
- Extracting high level performance properties (load imbalance, application specific metrics) of an application with an automatic tool; this is the task of High Level Analysis Component (HLAC)
- Prediction of how an application will behave under certain conditions and parameters with a tool based on analytical model; this work is performed by Performance Prediction Component (PPC)
- Visualization of performance measurements is through a Visualization Component (VC)

The G-PM has also potential as a middleware performance evaluation tool since an application or test program properly instrumented with PMC can be used to extract performance information regarding the execution of middleware intensive tasks.

Task 2.2 is establishing a set of performance metrics to describe the performance capacity of Grid configurations and applications and at the same time will develop a suite of Grid benchmarks that are typical of Grid workloads. These benchmarks will be used to estimate the values of performance metrics for different Grid configurations, to identify important factors that affect end-to-end application performance, and to provide application developers with the initial estimates of expected application performance.

Four classes of benchmarks will be developed: Generic kernel, application kernel, middleware and micro benchmarks covering all the layers of the Grid architecture. These benchmarks can be used to evaluate the performance of the whole Grid infrastructure deployed in the test and verification testbed allowing detection of performance bottlenecks prior to the deployment in the production testbed.

3.10.2. TESTBED MONITORING

Monitoring is not the mission of CrossGrid Task 4.4, in this sense Task 4.4 will be mostly a consumer of common monitoring services, tools are provided elsewhere. However the complexity of the testing activities and the volatile nature of the testbed configuration will require some specific monitoring.

The installation and configuration of Grid middleware is a time consuming and error prone task, since middleware test and validation will require frequent installations with different configurations, it is essential to achieve correct middleware and site configurations quickly. Test and validation activities will require additional functionalities that go beyond monitoring; they are required to automatically verify the correct configuration of each site and perform tests in Grid services at each site and across sites. Reducing the amount of time spent on achieving a correct site configuration will provide more time for the actual test of new components and features. In the future these automatic test programs can perform many of the required steps for the middleware validation.

For the CrossGrid test and validation testbed a monitoring system capable of verifying not only the availability of the Grid services but also their proper behaviour is being studied in the context of the WP4 network monitoring and test activities.

Tests and diagnostics will be developed or enhanced to verify both the correct behaviour of the middleware and the system configuration. These tests will be consolidated into a tool that will show for each site a complete report of all systems (CE, SE, WN, RB, RC, II, UI) and services including:

- Service reachability monitoring by establishing TCP connections to detect basic network or system configuration problems.
- Service behaviour monitoring by exercising each service with a set of tests covering all the service functionality.
- System configuration monitoring by verifying information retrieved from the service being tested or obtained by submitting probe jobs to the CEs and SEs.

These monitoring tools will be started and controlled from central monitoring servers, no special software should be required on the systems being monitored.

A tool to integrate these tests, control their scheduling and publish the results will be required. MapCenter from DataGrid might be upgraded to support these new features, but also other tools such as Nagios will be evaluated.

3.10.3. NETWORK MONITORING

The network layer protocol used by the middleware to be deployed is the IP protocol and in most cases the academic research networks will provide the network paths between testbed sites. This means that CrossGrid can use existing Internet network monitoring and debugging tools without many changes.

However CrossGrid applications have requirements on high bandwidth and low delay due to the large amounts of data that need to be transferred and to the interactive and distributed nature of the applications. This means that routing devices along the network paths need to be optimised to maximise data transfer throughput without sacrificing the interactive response of the applications, and at the same time carefully monitored to verify the correctness of the configurations. Also the behaviour of the network paths may have a large influence in tests performed across sites. For instance very small packet losses have extremely negative effects in TCP data transfers over high bandwidth long distance network paths. These applications may require traffic prioritisation through QoS mechanisms in which case new parameters such as the length of the router queues for each traffic class and packet drops might need to be monitored.

Many of the test activities requiring high bandwidth and low delay can be scheduled on sites with good network connectivity. Network monitoring information can be used to choose the sites and the correct moment to perform the tests, thus avoiding collisions with other activities consuming bandwidth over the same network paths. Also the objective of the testbed is to test and validate the middleware including any network monitoring tools that may be included in middleware releases.

Two strategies are foreseen:

- Integrate the testbed network monitoring into a larger CrossGrid network monitoring infrastructure capable of providing the metrics required by the test and validation activities.
- Deploy an independent test and validation network monitoring.

The first approach has the advantage of reducing the monitoring effort in the project and combines it into a single integrated network monitoring infrastructure. The second approach has the advantage of adding more flexibility namely to test and validate new or modified monitoring tools in accordance with the test and validation objectives. Therefore the second approach will be required while a CrossGrid monitoring infrastructure is not in place and also to test the tools initially and later when testing new releases. Having monitoring data both from the CrossGrid monitoring and from monitoring tools being tested will be also highly valuable to verify the behaviour of the tools being tested.

For the initial test purposes Pinger and Iperf will be deployed, they seem to have the required functionalities namely the measure round trip time, packet loss and response time variation by using ICMP packets and TCP bandwidth, delay, jitter and datagram loss. There is discussion on whether monitoring using ICMP packets is capable of capture accurately the behaviour of a network path, due to the ICMP rate limiting and prioritisation policies that are often implemented in routers. However comparisons between Pinger and other non-ICMP based monitoring tools have not yet shown any major result divergence therefore Pinger will be the first monitoring tool to be deployed.

These monitoring tools must be installed in a dedicated system with enough processing capacity at each testbed site, this is necessary to obtain correct measures since other processes running in the system can have a negative effect over the time measurements hence making the network monitoring data useless. In this case the measures would not reflect the real network path behaviour but the end-to-end behaviour between the monitoring source and target systems. Another related requirement is that the monitoring system should be near to the router connecting the site to the service provider.

DataGrid has also adopt Pinger and Iperf as monitoring tools, for CrossGrid this choice has the advantage of both tools being available has RPMs in the EDG middleware releases. The EDG Pinger and Iperf have been modified to fetch configuration information from a central server and publish the monitoring results into the MDS information tree. A central configuration web server will be required to provide the necessary configuration information.

3.11. NETWORK INFRASTRUCTURE

The network infrastructure for the CrossGrid test and validation testbed will be based on the Internet with national network connectivity being provided by the national research networks (NRENs) and international connectivity being provided by the multi-gigabit pan-European network backbone Geant.

Contacts have already been established between the CrossGrid partners and the national research networks in the context of the WP 4 testbed setup activities. Close collaboration with the NREN's will be important for the provisioning of network bandwidth and services. In some cases collaboration in the evaluation of technologies of mutual interest has already been established.

The establishment of contacts with Geant will also be important for the provisioning of special network requirements. Four Geant points of presence will be used they are marked in green dashed circles in the next diagram.

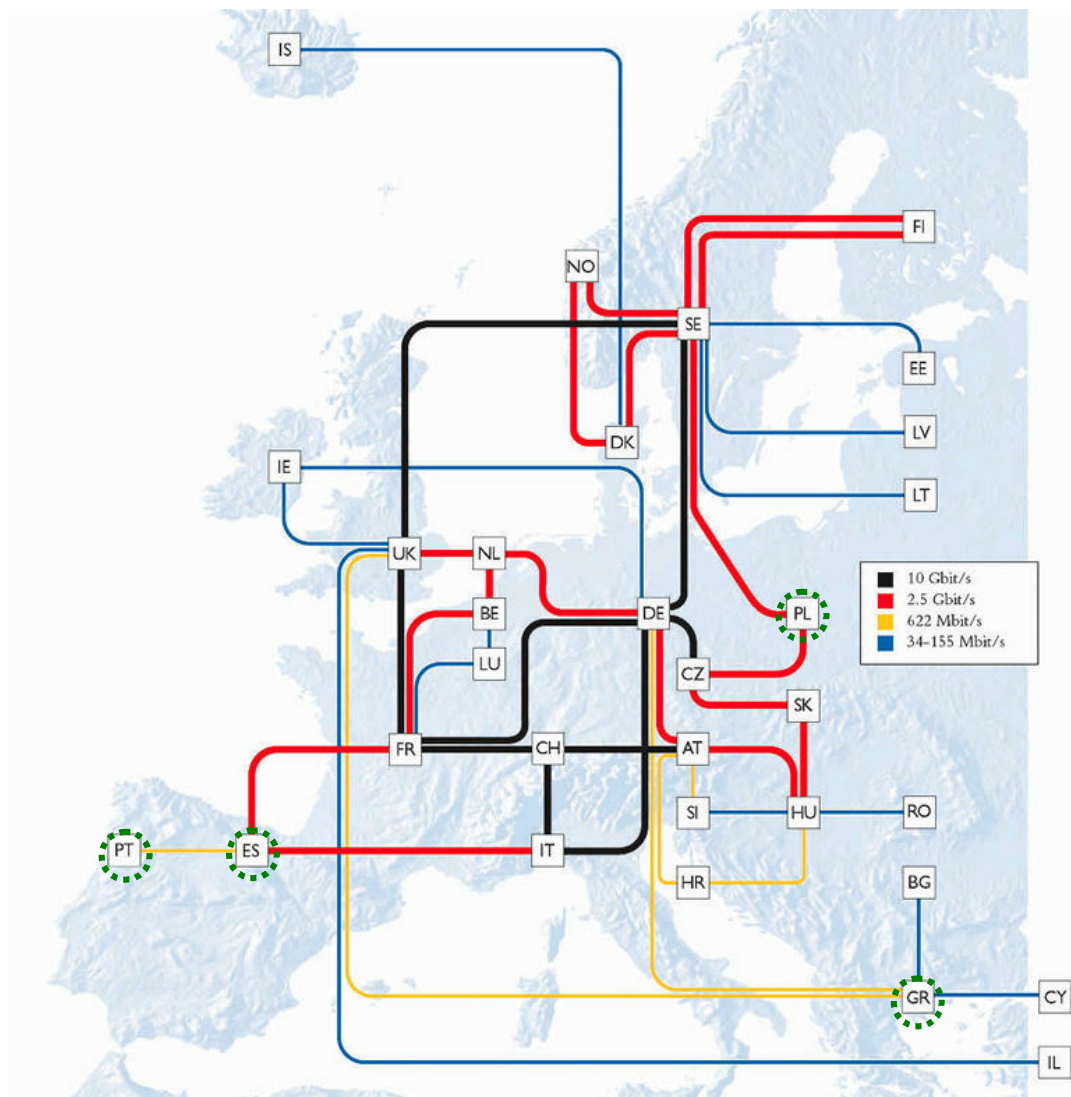


Figure 24 – Geant network map on February 2002

As the map shows the test sites are located in countries distant from central Europe. This has the disadvantage of not being able to take profit of the core high bandwidth links, however this is not seen as a major problem for testing activities since enough bandwidth is available in all involved countries and most of the middleware tests will not require high bandwidth. Most tests will be centred on verifying middleware functionality not performance therefore they will not have any special bandwidth requirements. Detailed middleware performance will be evaluated in the production testbed while using applications. Even so the validation testbed can provide valuable information about expected middleware performance allowing early detection of bottlenecks before deployment in the production testbed.

Most of the network testing and optimisation seems to be concentrated nowadays on the high bandwidth core of the Geant network. This is the case of DataGrid WP7 where the emphasis has been put on the high bandwidth paths between tier 1 High Energy Physics laboratories hosting large data stores. Nevertheless the lower bandwidth network paths are extremely important to ensure true high quality pan-European connectivity and allow the grid to take advantage of the computing resources distributed across all European countries. The

chosen topology has also the advantage of allowing the exploration of the effects of geographical distance over the middleware and applications.

The next table shows the foreseen bandwidth requirements for LHC computer centres involved in one experiment in 2007.

From	To	Bandwidth
Tier 0	Tier 1	1500 Mbps
Tier 1	Tier 2	622 Mbps
Tier 2	Tier 3	155 to 34 Mbps

In the real world only a few large research laboratories and companies will enjoy the benefits of grid computing over high bandwidth paths, most will use grid computing over low or medium bandwidth paths. Even in High Energy Physics most of the computing resources will be hosted at tier 2 and 3 centres. Hence testing the behaviour of the middleware in these conditions will be also important.

The bandwidth available at each test site is summarized in the table below.

Institute	Country	Geant bandwidth	Site bandwidth
Demokritos	Greece	622 Mbits/s	10 Mbits/s
LIP	Portugal	622 Mbit/s	Up to 34Mbit/s
Cyfronet	Poland	2.5Gbits/s	34 Mbit/s
CSIC	Spain	2.5Gbit/s	155 Mbit/s

The following table shows a good combination of countries with low and medium bandwidth covering well the several special network scenarios that some tests may require.

Test requirements	Countries
High bandwidth	Spain – Poland
Low delay	Portugal – Spain
High delay	Portugal – Poland, Portugal – Greece

High bandwidth is mostly important to test data access, file transfer and replication, while low and high delays will be important to test features related with interactive and parallel applications support under different network conditions.

Although CrossGrid is evaluating the necessity of network quality of service technologies (QoS) their deployment is not foreseen for the first releases of the testbed. QoS has been identified as one of the network topics of possible interest, regarding the deployment of

interactive applications and in particular parallel interactive jobs using MPI. QoS can reduce the switching delay and the packet loss by applying traffic classification and prioritisation. However wide deployment of QoS technologies is complex requiring support and configuration in all network devices along the paths that might not be possible for all sites and network providers. There is also the possibility that QoS improvements might also contribute somehow to improve the middleware behaviour and stability hence modifying the test results.

The deployment of QoS technologies in the test and validation testbed will be considered only if a clear well funded need for the wide deployment of QoS technologies in the production testbed appears, or if a specific request to evaluate the middleware behaviour with QoS is issued.

3.12. NETWORK SECURITY ISSUES

Most sites have local security policies and firewalls that impose restrictions on opening TCP ports to the exterior since this represents a high security risk. However for the grid to work some network connectivity is required, therefore several ports used by grid services must be open to the outside. The next table lists the services and ports used by Globus and EDG middleware. For CrossGrid specific middleware services the port numbers are not yet known. The table also includes some proposed port numbers for services required by the crossgrid testbed, most of them are already being used in the first crossgrid testbed release.

Service	Ports	Observations
RB port to listen for JSS	8881	Not required to be open.
RB port to listen for UI	7771	Must be open to all testbed sites.
JSS	9991	Not required to be open.
LB	7846 15830	Must be open to all testbed sites. Must be open to all testbed sites.
GRIS	2135	Should be open to all testbed sites.
GIIS	2170	Should be open to all testbed sites.
FTREE	2169	Should be open to all testbed sites.
FTREE INDEX	2171	Should be open to all testbed sites.
GRID FTP	2811	Must be open to all testbed sites.
RFIO	3147	Must be open to internal WN's.
GRAM	2119	Should be open to the world, at least to the RB.
GDMP	2000	The GDMP should be open to other remote GDMP servers. Since they will be hard to identify it should be open to all testbed sites.
RGMA	8080	Must be open to all testbed sites.
HTTP for net monitoring	80	Open to the central monitoring server.

NTP	123	UDP port open to the upstream NTP server.
RC		Must be open to all testbed sites.
WP6 (CERN)	9980	
Alice (NIKHEF)	10389	
Atlas (INFN)	9011	
CMS (INFN)	9011	
LHCB (NIKHEF)	10389	
Biomedical (NIKHEF)	10389	
EarthObs (NIKHEF)	10389	
ITEAM (CERN)	9011	
cgTV (LIP)	9981	
crossgrid (LIP)	9980	
VO		Must be open to all testbed sites.
All DataGrid (NIKHEF)	389	
cgTV (LIP)	9991	
crossgrid (LIP)	9990	
gdmppservers (LIP)	9990	
LCFG	732	Open to internal Grid nodes.
LCFG-ack	733	Open to internal Grid nodes.
MyProxy		Must be open to all testbed sites.
cgTV (LIP)	7512	
crossgrid (LIP)	7512	

For two-phase commit job submission ports above 1024 must be open. A two-phase commit job submission is a Globus GRAM feature in which a job submission request from a client is sent to the Gatekeeper but not immediately executed, waiting until the client sends a confirmation signal or a timeout occurs.

Another issue is the range of ports used by client applications when establishing connections to servers, for Globus middleware the range of ports can be restricted by specifying the following environment variables:

- GLOBUS_TCP_PORT_RANGE (example of possible value: "40000,42000")
- GLOBUS_UDP_PORT_RANGE (example of possible value: "40000,42000")

Network connectivity problems related with firewall configuration are likely to occur between testbed sites. They will be caused by human error and the introduction of new middleware services. New crossgrid specific functionalities will be added upon the release of the crossgrid middleware. They will use new hosts and port numbers that must be reflected in the firewall configurations. Direct control over the firewall will be highly important to test new components. Updated information on the required port numbers will be provided at the test and validation web site.

3.13. TESTBED CONFIGURATION

The following schema shows the minimum physical configuration foreseen for the test and validation testbed at each site.

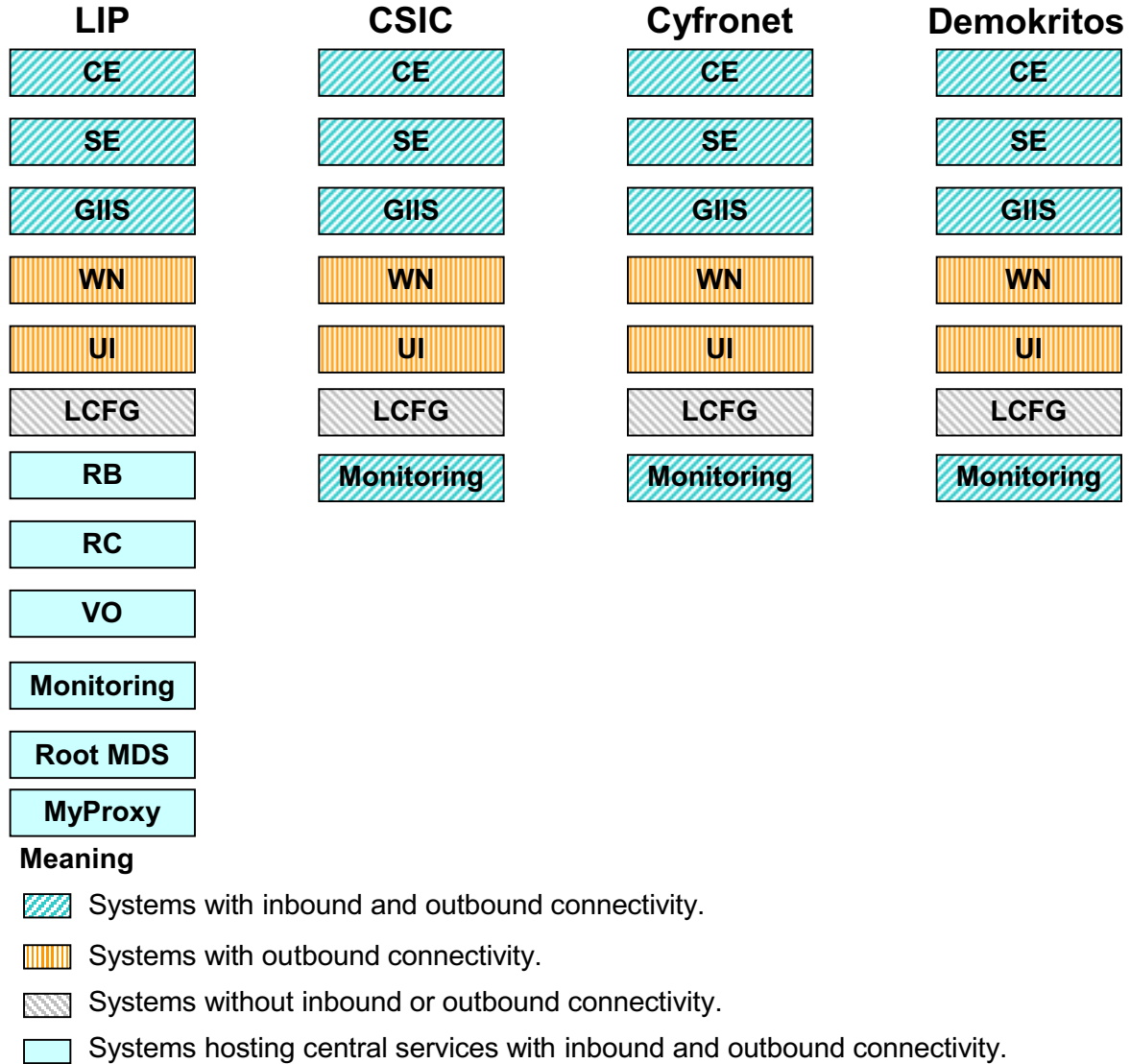
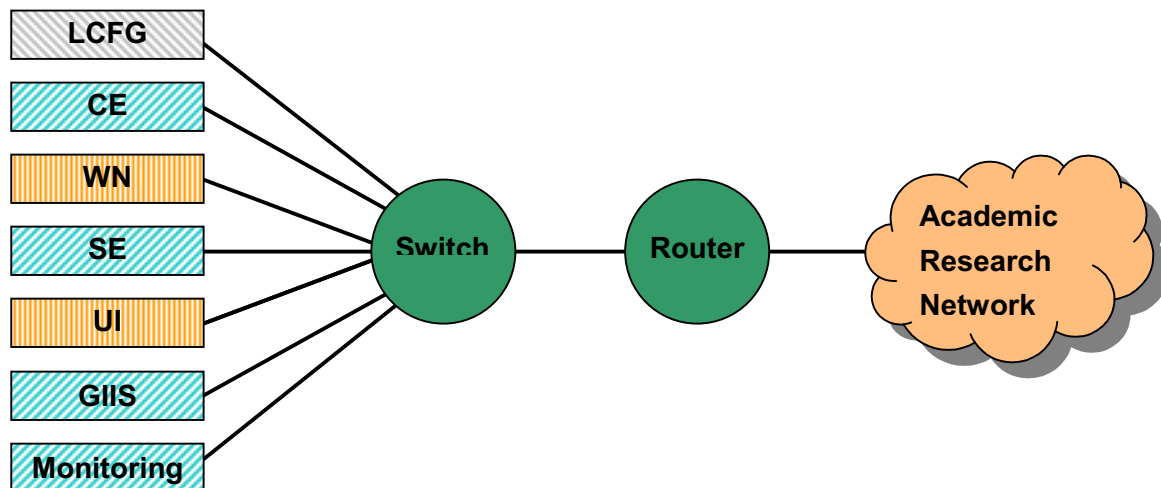


Figure 25. Physical site configuration

For certain tests different configurations may be required. For instance to test local interaction between storage elements it will be required that at least one of the sites will deploy two SEs, this can be accomplished by adding temporarily one more system to the site configuration.

The proposed site configuration follows the minimum hardware requirements guidelines established in the appendix A of the CrossGrid document D4.1 - *Detailed Testbed Planning* with the addition of the central testbed services that must be hosted at least in one site, initially LIP will be the only site supporting these services.

A reference for the minimum configuration to be deployed at each site can be found in the diagram below.



Meaning




-  Systems with inbound and outbound connectivity.
-  Systems with outbound connectivity.
-  Systems without outside connectivity.

Figure 26 – Reference site configuration

The router in the diagram is the router performing the network connection to the Internet/Geant through the academic research network and as such is shared by the whole site. The router interface connecting to the switch should be preferably dedicated for the Grid systems as well as the switch itself. Other solutions including VLANs will be deployed if they are more suited to each local site configuration.

The characteristics for each test system are the same mentioned in the CrossGrid minimum hardware requirements.

4. FINAL REMARKS

The CrossGrid testbeds will be based on a complex and rich set of components developed by the project itself and by other ongoing projects. However the first testbed release being deployed is completely based on the DataGrid testbed 1 middleware. CrossGrid components currently being developed will be tested and deployed when available.

The building blocks for the first CrossGrid testbed are now in place. The central services have been deployed and four Gatekeepers in four different countries have been added to the central Resource Broker. The infrastructure is being actively used to test the EDG middleware release 1.2.