Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática
2008

Rodolfo João Oliveira de Andrade

Infra-estrutura Grid para a Análise Multi-voxel de RMf

Multi-voxel fMRI Analysis Using an High Throughput Grid Framework

Universidade de Aveiro Departamento de Electrónica, Telecomunicações e
2008 Informática

**Rodolfo João
Oliveira de Andrade**

Infra-estrutura Grid para a Análise Multi-voxel de RMf

Multi-voxel fMRI Analysis Using an High Throughput Grid Framework

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Biomédica - Ramo de Instrumentação, Sinal e Imagem, realizada sob a orientação científica do Professor Doutor João Paulo Trigueiros da Silva Cunha, Professor Associado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro e do Professor Doutor Augusto Marques Ferreira Silva, Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

À Cláudia,
aos meus pais, à minha irmã
e à minha avó

o júri

presidente	Prof. Doutor João Paulo Trigueiros da Silva Cunha professor associado da Universidade de Aveiro
	Prof. Doutor Francisco António Cardoso Vaz professor catedrático aposentado da Universidade de Aveiro
	Prof. Doutor António Miguel Pontes Pimenta Monteiro professor associado da Faculdade de Engenharia da Universidade do Porto
	Prof. Doutor Augusto Marques Ferreira da Silva professor auxiliar da Universidade de Aveiro

agradecimentos

À Universidade de Aveiro e ao IEETA pelo apoio prestado durante a realização desta dissertação.

Ao Doutor João Paulo Cunha e ao Doutor Augusto Silva pelo apoio, orientação e constante motivação.

Ao Mestre Ilídio Oliveira e ao Doutor José Maria Fernandes pela paciência e disponibilidade demonstradas especialmente nas alturas mais críticas.

À comunidade do EGEE, pelo excelente trabalho que têm vindo a fazer, principalmente na ajuda a novos utilizadores. Dentro desta comunidade gostaria de fazer um agradecimento especial à Federação do Sudoeste do EGEE e à equipa do LIP pelo incansável esforço que tem feito na divulgação e apoio a novos sites, sem o qual esta dissertação não teria sido possível.

A todos os meus colegas pelo constante apoio e encorajamento.

palavras-chave

Computação distribuída, RMf, processamento de imagem, Grid.

resumo

O presente trabalho apresenta uma nova abordagem à análise de imagens de RMf do cérebro, especificamente a utilização de medidas associativas na análise de séries temporais de RMf. Este tipo específico de análise, computacionalmente intensivo, requer recursos que normalmente não se encontram disponíveis em ambientes clínicos.

Redes Grid é um novo paradigma de computação distribuída de elevada performance que pode ser utilizado para potenciar a utilização deste tipo de análise, disponibilizando a capacidade de computação necessária. Implementou-se um framework que permite a utilização de uma infra-estrutura Grid para correr este tipo de análise de forma transparente, viabilizando a sua utilização em ambientes clínicos, onde o tempo é um factor crítico.

keywords Distributed computing, fMRI, image processing, Grid.

abstract

This work, introduces a new approach to fMRI brain image analysis, namely *multi-voxel fMRI association analysis.* The problem associated with this type of approach is that requires a large computing capacity that is not normally available at clinical sites.

To enable this specific type of analysis we are required to use High Performance Computing paradigms. In this context we analysed the use of Grid computing and implemented a framework that allows running the *multi-voxel fMRI association analysis* using a grid infrastructure resources. The use of this framework makes this type of analysis usable in clinical environments where time constraints can have a vital importance.

Contents

List of Figures

List of Tables

List of Acronyms

AC	Attribute Certificate
ACL	Access Control List
AJAX	Asynchronous Javascript And XML
AMGA	ARDA Metadata Catalogue
APEL	Accounting Processor for Event Logs
APGridPMA	Asia Pacific Grid Policy Management Authority
API	Application Programming Interface
BDII	Berkeley Database Information Index
BIRN	Biomedical Informatics Research Network
BIRN-CC	BIRN Coordinating Center
BOLD	Blood Oxygen Level Dependent
C-OMEGA	Open Middleware Enabling Grid Applications
CA	Certification authority
caBIG	cancer Biomedical Informatics Grid
CASTOR	CERN Advanced STORage manager
CE	Computing Element
CERN	European Organization for Nuclear Research

CGSP	ChinaGrid Supporting Platform
CLI	Command Line Interface
CT	Computed Tomography
CVA	Canonical Variates Analysis
DAG	Directed Acyclic Graph
DAGMan	DAG Manager
DGAS	Distributed Grid Accounting System
DICOM	Digital Imaging and Communications in Medicine
DILIGENT	A Digital Library Infrastructure on Grid Enables Technology
DPM	Disk Pool Manager
DRS	Data Replication Service
EC	European Commission
EDG	European Data Grid Project
EDT	DataTAG–Data TransAtlantic Grid Project
EEG	Electroencephalogram
EELA	E-infrastructure shared between Europe and Latin America
EGA	Enterprise Grid Alliance
EGEE	Enabling Grids for E-sciencE
EHR	Electronic Healthcare Records
ELSE	Ethical, legal and socio-economic
EU	European Union
EuGridPMA	European Grid Policy Management Authority
fMRI	functional MRI

GAT	Grid Application Toolkit
GFAL	Grid File Access Library
GG	Grid Gate
GGF	Global Grid Forum
GIIS	Grid Index Information Server
GLM	General Linear Model
GLUE	Grid Laboratory Uniform Environment
GMA	Grid Monitoring Architecture
GRAM	Grid Resource Allocation Manager
GRIDCC	Grid Enabled Remote Instrumentation with Distributed Control and Computation
GRIS	Grid Resource Information Server
GSIDCAP	GSI dCache Access Protocol
GT	Globus Toolkit
GUID	Grid Unique Identifier
GWT	Google Web Toolkit
HEP	High Energy Physics
HPC	High Performance Computing
HRF	Hemodynamic Response Function
HTML	HyperText Markup Language
ICT	Information and Communication Technology
IGTF	International Trust Grid Federation
INFN-GRID	Italy's National Institute for Nuclear Physics
IS	Information System

iVDGL	International Virtual Data Grid Laboratory
JDL	Job Description Language
JP	Job Provenance
JSDL	Job Submission Description Language
JSDL-WG	JSDL Working Group
LB	Logging and Bookkeeping
LCG	LHC Computing Grid
LDAP	Lightweight Directory Access Protocol
LFC	LCG File Catalogue
LFN	Logical File Name
LHC	Large Hadron Collider
LRMS	Local Resource Managing System
MDS	Monitoring and Discovery System
MEDICUS	Medical Imaging and Computing for Unified Information Sharing
MR	Magnetic Resonance
MRI	Magnetic Resonance Imaging
NIRS	Near Infrared Spectroscopy
NMI	National Science Foundation's Middleware Initiative
NS	Network Server
OASIS	Organization for the Advancement of Structured Information Standards
OGF	Open Grid Forum
OGSA	Open Grid Services Architecture
OGSA-DAI	OGSA–Data Access and Integration

OGSI	Open Grid Service interface
OMII-CHINA	Open Middleware Infrastructure Institute for China
OMII-EUROPE	Open Middleware Infrastructure Institute for Europe
OMII-UK	Open Middleware Infrastructure Institute for United Kingdom
OPENLDAP	Open source implementation of LDAP
PERMIS	PrivilEge and Role Management Infrastructure Standards validation
PET	Positron Emission Tomography
PFN	Physical File Name
PLS	Partial Least Square
PM	Package Manager
PPS	Pre-Production Service
R-GMA	Relational GMA
RA	Registration Authority
RB	Resource Broker
RFIO	Remote File Input/Output protocol
RFT	Reliable File Transfer
RLS	Replica Location Service
ROI	Region Of Interest
SARS	Severe Acute Respiratory Syndrome
SARSGRID	SARS Grid
SE	Storage Element
SHARE	Supporting and structuring Healthgrid Activities and Research in Europe
SOA	Service Oriented Architecture

SPM	Statistical Parametric Mapping
SRB	Storage Resource Broker
SRM	Storage Resource Manager
SURL	Storage URL
TAGPMA	The Americas Grid Policy Management Authority
TGCP	TeraGrid Copy
TURL	Transport URL
UI	User Interface
UNICORE	UNiform Interface to COmputing REsources
URI	Uniform Resource Identifier
UUID	Universally Unique Identifier
VDT	Virtual Data Toolkit
VO	Virtual Organization
VOMS	Virtual Organization Management System
WISDOM	Wide In Silico Docking On Malaria
WLCG	Worldwide LHC Computing Grid Project
WMS	Workload Manager Service
WN	Work Node
WS	Web Service
WS-GRAM	Web Service–Grid Resource Allocation and Management
WSRF	WS-Resource Framework
XML	eXtensible Markup Language

Chapter 1

Introduction

1.1 Motivation

Healthcare is currently one of people's biggest concerns. Great efforts are being made to increase the quality of the Healthcare services. This effort is reflected in the increase of life expectancy. But with this increase, brain diseases like Alzheimer and Parkinson, normally associated with older people, are increasingly becoming a critical factor in people's quality of life. This has led to an increase in the search for new methods of diagnosis and cure for brain diseases.

New methods of diagnosis are being developed and other being improved, while other are becoming more widely available. In particular, medical imaging is becoming a essential method of diagnosis and mandatory for all healthcare institutions. With the increase of medical images, medical imaging systems that allow their management and storage are becoming vital for the correct operation of Healthcare institutions.

From the moment these images are available in a digital format, it becomes possible to share them among institutions and specialists allowing new opportunities for collaboration, diagnosis and medical studies. However we are still far from such a reality. The reasons are very diverse and can be controversial. Most of them are related with privacy and security concerns around patients data. Besides the technological problems, there are also political issues that should be given equal or more attention.

Nonetheless it is not only the access to data that is important but also the access to computing resources. New types of analysis require more powerful computing solutions than the ones that are currently available.

A new paradigm for HPC (High Performance Computing) is emerging as the future for collaboration networks – *Grids*. They are becoming the solution to problems that were impossible to solve with old paradigms but more importantly, solving problems that influence directly people lives. Furthermore they are creating new opportunities for researchers of several scientific fields, ranging from HEP (High Energy Physics) to biomedical.

The concept behind *Grids* is that several institutions can create pools of resources shared among them. A user only sees a single pool of resources despite the fact that they can be geographically distributed among several institutions. These resources can vary from computational power, storage or bandwidth to scientific instruments.

The use of *Grids* in Healthcare can provide a infrastructure for the creation of collaborative networks among institutions, allow sharing data and provide the computational resources that are needed for image analysis. Moreover they can serve as the base for the development of more complex systems.

A area in Healthcare where *Grids* can have a potentially important role is in medical imaging. The quantity of images produced in healthcare is increasing very fast, creating problems related to storage and processing. At the same time the digitization of these images provides new opportunities for knowledge sharing and collaboration.

In the specific field of brain studies, fMRI (functional MRI) is becoming one the most important instruments to study brain diseases with functional impairment and in trying to understand its functions and how they relate among each other. Some of the methods used for analysis are highly computational intensive, making them useless from a practical point of view in environments where the time is a critical factor. One of these approaches to fMRI analysis is the multi-voxel fMRI association analysis. This method, *multi-voxel association analysis*, that is being introduced in this work, has some key aspects that make him both appealing and a novelty. Its big disadvantage is that it is computationally intensive.

Epileptic patients, for example, are admitted for decision of surgery within a three to five days period. In these cases fMRI exams can be performed in the first or second day. *Multi voxel fMRI analysis* could be of help for the clinical decision however, the results must be made available in less than 24 hours. This time limit restriction is unimaginable using only local resources. The use of *Grids* can make this type of analysis usable within clinical environments.

1.2 Objectives

The aim of this work is to create the conditions necessary to test and implement new approaches to fMRI analysis. This implies testing and integrating new paradigms for HPC, implementing fMRI analysis tools and understanding the environment in which these tools would be more valuable.

Due to its distributed nature *Grids* require normally relative large and complex setups that can be very hard to gather and manage. Our objective is to collaborate with the European project EGEE (Enabling Grids for E-sciencE), so that instead of building our own *Grid* we can become part of a larger *Grid* infrastructure. This still implies creating a local *Grid* node.

Because the access to these types of infrastructure is normally not very user friendly, we have to develop a framework that allows users to run their analysis without having to worry about the management of the *Grid* infrastructure resources. Ideally a user should not need to know that he is using a *Grid* infrastructure.

The objective of this framework is to allow a user to run intensive computing fMRI analysis. For this purpose, at least one fMRI analysis with such requirements should be implemented and tested using the proposed framework. In this particular case we will implement the multi-voxel fMRI association analysis using a linear and a non-linear association measure.

In the end we should have a framework that allow us to run computing intensive fMRI analysis using a *Grid* infrastructure and also be able to evaluate if this technology is mature enough or fulfils the necessary requirements associated with this type of analysis.

Another objective is to try to understand if this technology can be used not only for providing computing resources but also data management with special relevance to Healthcare.

1.3 Structure

This dissertation is divided into the following chapters, excluding this one:

- **Chapter 2 - Background Concepts and State of the Art**, makes an introduction to fMRI imaging and *Grid* technology. In the fMRI section a brief introduction about the technology is given and the methods of analysis used. In the *Grid* section, introduction to the concept behind this technology and its community. The technology behind *Grids*, namely the services that support these infrastructures is also discussed.

Finally, we discuss the relation between *Grids* and *Healthcare*, with special relevance to the problems that have already been identified and some of the solutions proposed.

- **Chapter 3 - Multi-voxel fMRI Association Analysis**, presents a new approach to fMRI analysis – *Multi-voxel fMRI association analysis*, indicating its main advantages and disadvantages. We also describe the aspects associated to its implementation, specially the association measures used. In the end we discuss one of the major problem related with this type of analysis, namely the computational resources needed.
- **Chapter 4 - A Grid framework for fMRI research**, describes the solution concept to the problem of fMRI association analysis. It presents the proposed system architecture and its implementation. Then it concludes with the presentation of the results obtained.
- **Chapter 5 - Discussion and Conclusion**, tries to sum up the most important aspects during the development of this work, discuss the results obtained and present some of the lessons learned.

1.4 Main Contributions

We implemented the multi-voxel fMRI association analysis using two association measures, namely linear correlation (r^2) and V. Using this tool we tested several fMRI datasets and compare the performance of each association measure in a single workstation and using the EGEE PPS (Pre-Production Service) testbed. To enable the use of the *Grid* infrastructure we developed a framework that target users can access through a web interface and run multi-voxel fMRI association analysis.

The use of *Grids* enables the use of multi-voxel fMRI analysis in clinical environments reducing greatly its execution type.

To test and develop the framework we had to setup a EGEE PPS node. We were the first Portuguese institution to collaborate with the EGEE Project, apart from LIP that is a contracting partner of this project.

Chapter 2

Background Concepts and State of the Art

2.1 fMRI background

2.1.1 Medical imaging and healthcare

Medical imaging has become a essential and central part in healthcare [1, 2, 3]. Nowadays is used in the diagnoses process mainly through the use of medical imaging techniques like ultrasound, radiography, CT (Computed Tomography), MRI (Magnetic Resonance Imaging) and nuclear medicine images (see figure 2.1) [4]. In addition to diagnosis it is also used in the guidance of procedures or in the assessment and planning to evaluate, for example, the progression of a disease in response to a specific treatment. The use of medical imaging in the cases described above depends a lot on the skills of the healthcare professional where, once again, medical imaging plays a vital part in medical education and training. In several areas of research, medical imaging play a key part. In epilepsy, for example, it can be necessary the removal of specific areas of the brain and so it is extremely important for a surgeon to be able to identify critical areas of the brain (e.g. associated with movement, language). Functional mapping is a technique that allows the establishment of relations between specific images and particular functions of the Human body [5].

The different imaging modalities are used in different types of analysis. For example some are more suited for soft tissues and others offer a better trade off between spatial and time resolution. In some cases data from different types of imaging are being combined to allow

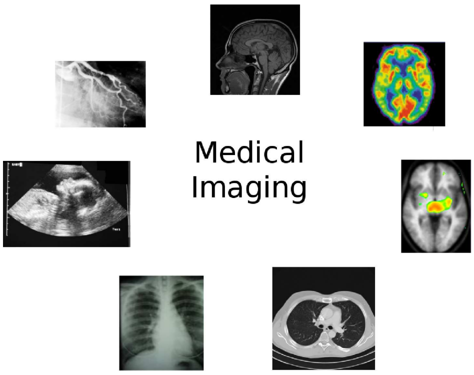


Figure 2.1: Images produces by different medical imaging modalities - Starting clockwise first we have a MRI, followed by a PET scan, a multimodal image (MRI + PET), a CT, a X-Ray, a ultrasound and finally a angiogram

a better interpretation of the data. This approach called multimodal imaging is used for example with EEG (Electroencephalogram) and MRI. In this case the EEG provides a very good temporal resolution while the MRI provides the high level spacial resolution. This is specially important in brain studies.

2.1.2 fMRI and Human brain study

From the several existing imaging techniques fMRI is a relatively new MRI based technique that allows the monitoring of brain activation patterns from magnetic variation induced by changes in regional cerebral blood flow.

Several imaging techniques were developed, such as the PET and the NIRS (Near Infrared Spectroscopy), that used brain hemodynamics to infer brain activity before the fMRI was developed in the mid 1990s. The fMRI brought a few advantages like good spatial resolution, reasonable temporal resolution and the fact of not being a invasive technique nor using radiation thus increasing patients safety [6].

The idea that hemodynamics, changes in blood flow and blood oxygenation, are closely related with to neural activity is more than a hundred years old, as shown by experiments in the 19th century by the Italian scientist Angelo Mosso, Charles S. Roy and Charles S. Sherrington [7, 6]. However, only in 1948, Seymour Kety and Carl Schmidt, confirmed the relation between brain activity and changes on blood flow [8, 9].

Cerebral neural activity produces a hemodynamic response characterized by a increase in blood flow, with a delay of about two seconds, richer in oxyhemoglobin to compensate the increase in oxygen consumption. This change in oxyhemoglobin is called the BOLD (Blood Oxygen Level Dependent) effect [10]. The MR (Magnetic Resonance) signal is able to capture these small variations due to the differences present in the magnetic properties of the different forms of haemoglobin – oxyhemoglobin is diamagnetic but deoxyhemoglobin is not.

The key principle behind the development of fMRI was the use of the BOLD contrast magnetic resonance imaging to map brain function. When BOLD activations and deactivation's are time related with specific events, they can be correlated to the events metabolic response in the brain [11]. This is a valuable tool for studies ranging from brain diseases (e.g. Alzheimer [12], Parkinson [13]) to normal brain function in aging [14].

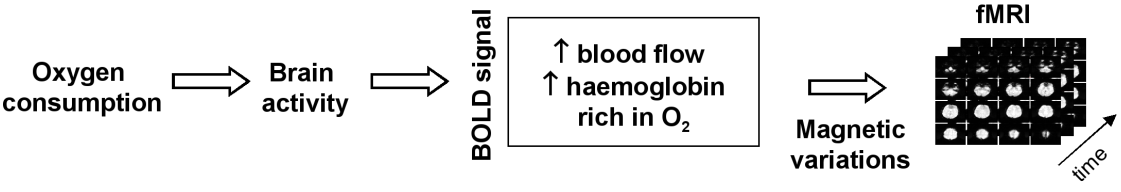


Figure 2.2: Principles behind fMRI imaging

fMRI produces series of 3D images or volumes, where each volume is associated to a specific time instant (figure 2.2).Thus it is possible to submit a patient, during an exam, to a series of events and see the variation of the BOLD signal.

Each fMRI generated is formed by a matrix that represents the signal intensity in a specific location. Because each image contains also information that is dependent of the thickness of the slice, we say the image is composed by a matrix of volume elements called voxels.

If we consider a voxel at a specific position in all volumes, we will have a time-serie that shows the evolution of the intensity of the voxel during that time [15]. In figure 2.3 the green voxel represent the same voxel from different volumes that will form the time-series. Usually, fMRI analysis is based on the study and analysis of these time series.

fMRI generates a large quantity of images. For example, during a five minute exam if a volume is acquired every three seconds it will generate a set of a hundred volumes. A volume is normally composed of 16 slices, with 64×64 pixels as shown in figure 2.3. This, fMRI trial of 5 minutes, has typically a size of approximately 13MB.

One major problem of this technique is that BOLD changes are small, in the order of 3-5% from the background MRI imaging signal [16]. A methodological problem with fMRI analysis

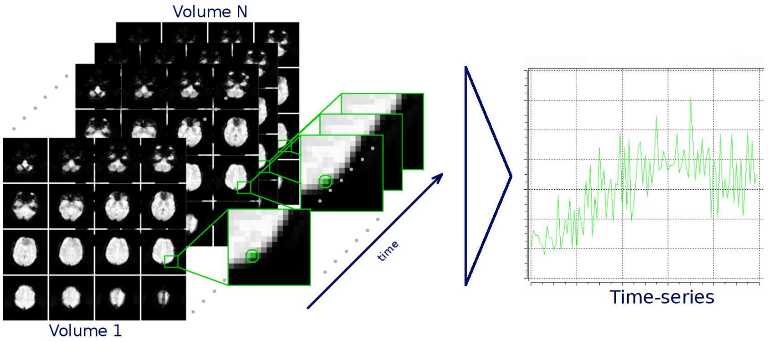


Figure 2.3: fMRI time series

is that there is no unique model for the HRF (Hemodynamic Response Function) [17]. This may compromise the fMRI interpretation since it depends on the HRF activation model used.

2.1.3 fMRI analysis

fMRI analysis involves three main steps. First, after the images are acquired by the MR, they have to be transformed into images that we can understand. The purpose of the pre-processing is to prepare the data so that it can be statistically analysed. This processing includes techniques that make slice-timing corrections because slices are acquired in different instants of times and the voxels that are present in a volume should appear as being acquired in the same instant. Another typical process is the motion correction that allow the alignment of brain images along the time also known by image registration. This is done by co-registering the volumes through transformation techniques, like rotation and translation.. After the images are aligned, other techniques can be applied such as image blurring, intensity normalization, reduction of low or high frequency, among others [18, 19].

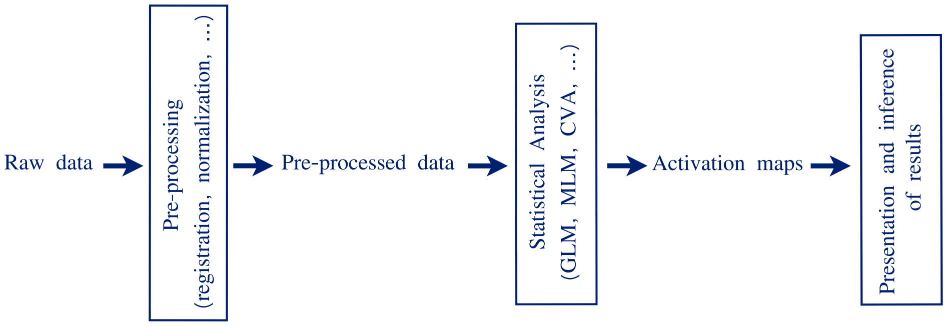


Figure 2.4: Typical fMRI analysis

After the images are pre-processed the activation maps are obtained through statistical analysis. This can be done using different approaches. One strategy is to analyse each voxel's time series individually – "univariate analysis". Another is to analyse all data together or study the relation between voxel's time series in a specific region – "multivariate analysis". The CVA (Canonical Variates Analysis) and PLS (Partial Least Square) are examples of "multivariate analysis" [19, 18]. It is also possible to characterize the type of analysis as "model-based" or "model-free". The GLM (General Linear Model) analysis, a example of "univariate" and "model-based analysis", compares statistically a model, that represents the expected response, with the data gathered [20]. Model-free methods, on the other hand, are not based in models but in specific statistical criteria (for example, statistically independent temporal or spatial components), to find components or effects of interest in the data [19]. A methodological problem with fMRI analysis is that there is no unique model for the HRF [17]. This may compromise fMRI analysis interpretation that are based in the HRF. Model-free methods are a quite interesting approach to this problem.

Depending on the type of the analysis, the needed computational resources may vary. The analysis and storage of the fMRI images is usually done locally making them only available at the site where they were acquired.

2.2 eScience and Grid Computing

2.2.1 The Grid Concept

The expression "The *Grid*" comes from a analogy with the electric grid. This concept was created in the mid-1990 to denote a proposed "distributed computational infrastructure for advanced science and engineering". The goals of this infrastructure were:

- Computationally demanding data analysis;
- Federation of distributed data sets;
- Coupling of scientific instruments with remote computers.

The problem that *Grid* technologies are trying to solve is the *coordinated resource sharing and problem solving in a dynamic, multi-institutional virtual organization.* The current distributed technology present us with some possible solutions but only when working in a single organization. *Grid* technology does not compete with current distributed technology, it complements the existing solutions [21].

Since their appearance in the mid 1990, *Grid* technologies have been evolving. In 1997 the GT (Globus Toolkit), an open source project was considered as the standard for *Grid* computing. With the start of a increased interest in *Grids*, new projects appeared and the Grid community started to search for standards to guarantee that the interoperability of the existing and future projects. In 2000 several existing Grid Forums merged to create the GGF (Global Grid Forum) that became a standards body. Finally in 2002 the OGSA (Open Grid Services Architecture) appeared as the true community standard for Grid infrastructures [22, 23].

To a user that wants to run a application, the *Grid* provides a abstraction layer to all the resources available. The user only sees a single pool of resources. The group of services that create this abstraction layer is called the *Grid middleware*. From a user perspective the middleware provides a set of API (Application Programming Interface) and CLI (Command Line Interface) that allow the access to several services.

The services that compose the Grid middleware have very specific functions and interact among each other to provide the capabilities of the infrastructure to the user.

Some services allow resource discovery and provide management mechanisms that are specific to the resource type. For example if it is a computational resource it should provide the mechanisms to start, monitor and retrieve results of a program and could also optionally provide advance reservation capabilities. In the case of a storage resource they need to provide, for example, mechanisms for data movements [23].

The users have also access to coallocation, scheduling and brokering services that allow the request and allocation of one or more resources. Other services like monitoring and diagnostic help prevent problems related with security, failure and other problems that might affect the infrastructure resources.

The resources that can vary from simple data to software or scientific instruments and others can be owned by different organizations that must be able to restraint the access to these resources to authorized users only. The access to these resources has to be necessarily highly controlled. To enable this, the concept of a Grid VO (Virtual Organization) was created in order to identify a particular but dynamic, group of users that have access to a specific set of resources [24].

The controlled environment described above imply that each user must be authenticated. It is important that users authenticate only once when they are accessing multiple remote resources. For this to be possible methods must exist that allow the users authentication to be delegated between services or resources [23].

Another important aspect of a distributed shared environment is the accounting. Besides giving access to their resources, organizations must be able to measure the amount of resources that were used by, for example, a specific VO. This is quite difficult because of the heterogeneity of the resources present in a Grid infrastructure.

The applications that run in these type of infrastructures have special requirements. We can divide them into five major classes: distributed supercomputing, high throughput, on demand, data intensive and collaborative [25].

2.2.2 Grid Community and Standards

In 1998 the United States, European and Asia-Pacific Grid Forums were established by the Grid community. They eventually merged in 2000 to create the GGF that became a standards body. The main goal of this organization was to create new standards to guarantee the interoperability of the existing and future Grid projects [26]. A few years later, in 2002, the OGSA finally appeared as the true community standard for Grid infrastructures [22, 23]. In 2004 several industry leaders aware of the growing potential and importance of Grid technology formed a EGA (Enterprise Grid Alliance), a consortium created to accelerate the development of enterprise Grid solutions and deployment of Grid computing [27]. In 2006 the OGF (Open Grid Forum) was formed from the merger between the GGF and the EGA in an attempt to focus the development of standards [28].

One of greatest accomplishments of this community was the OGSA, currently at version 1.5. This service-oriented architecture is built in top of WS (Web Service) standards addresses some aspects that are relevant to Grid services such as service creation, life-time and several others. This was done in a first stage with the creation of a core set of interfaces called OGSI (Open Grid Service interface), that was eventually abandoned and replaced the WSRF (WS-Resource Framework). The WSRF is a set of WS specifications being developed by the OASIS (Organization for the Advancement of Structured Information Standards) that describes how to implement OGSA capabilities using WS [29, 30, 22, 31]. The OGSA is still evolving through the work done by the Grid community at the OGF and others. The GT 4 is a example of a middleware that is providing some OGSA capabilities based on WSRF.

2.2.3 Virtual Organizations

Grid can provide us with an incredible amount of resources. The access to these resources by users has to be made in a highly controlled environment. Lets imagine a single Grid were we have users that are running experiments related with high energy physics, bioinformatics or earth sciences, among others. Each group of users will have different needs relating software, some might need to run data intensive processes during a small period of time and others might even need only access to large distributed sets of data. The organizations that participate in the infrastructure described above can decide to whom they will give access to their resources. They can share their resources with only a group a users, like the bioinformatics, with several group of users or even decide that each group of users will have access to a different type or amount of resources. To manage this, each group of users that have common interest or objective are grouped into a VO. In the above case we would have the bioinformatics VO, the high energy physics VO and earth science VO for example.

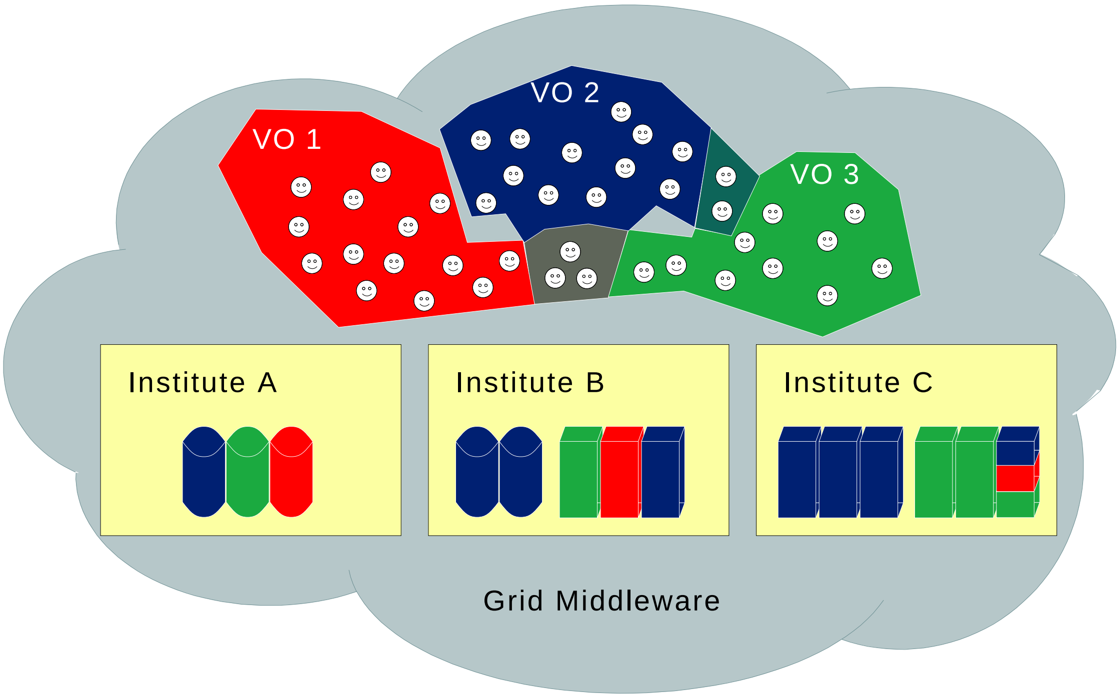


Figure 2.5: Virtual Organization

A likely VO scheme to appear in a Grid infrastructure is the one presented in figure 2.5. In this example we have three VO, identified by the colors red, green and blue, and three organisations or institutes identified by the letters A, B and C respectably. Each institute is sharing resources of different types. The color associated with each resource indicates that it is being shared with the corresponding VO color. These resources can be shared with one or with more VOs. There are also users that belong solely to one of the three VOs and users who belong to more than one. As shown in this example the organisations can decide the amount of resources they will share with each VO and share the same resources with several VOs. From the user point of view, he sees the resource in single pool and not distributed by several organisations.

2.2.4 Grid Architecture

OGSA was an important achievement in the history of Grid computing by defining a set of standards for Grid infrastructures. The existing infrastructures have different architectures that are implementing or evolving forward OGSA although most implement SOA (Service Oriented Architecture) [30]. Despite this variety of architectures they all implement services that provide similar functions. Based on these we can define a general architecture divided in three main layers.

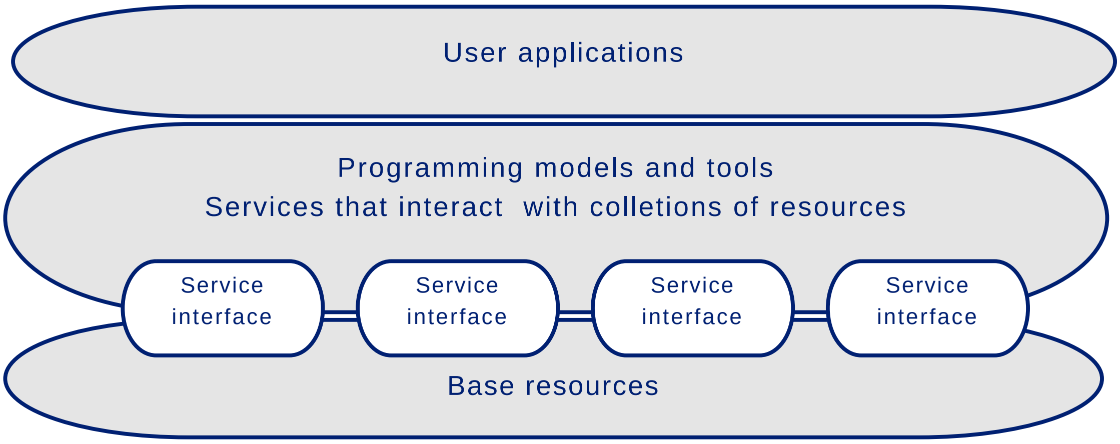


Figure 2.6: Grid Generic Architecture

In the bottom layer we have the base resources that are usually managed locally but shared remotely. These can be data, computational or network resources, specific programs or even scientific instrumentation, among others. In the middle layer we can group all the services that interact with the collections of resources in the lower layer, programming models and tools, accounting and authorization services. These services that define the capabilities that the infrastructure provides are usually called the Grid middleware stack (Table 2.1).

Services that interact with collections resources	
Services	Description
Directory services	VO specific services that allow users to view the resources available and their attributes.
Coallocation, scheduling and brokering services	Enable the allocation and scheduling of a group of resources for a specific task.
Monitoring and Diagnostic services	Monitor resources–protection against failure or intrusion
Data replication services	Management of VO storage to maximize data access performance.
Community authorization server	Manage VO policies to control the access to resources.
Accounting and payment services	These services allow the accounting and payment due to the use of the shared resources.
Programming models and tools	
Services	Description
Grid-enable programming systems	Allow the use of familiar programming models in Grid environments making use the available Grid services.
Workflow systems	Enable the use and management of group of tasks that can have dependencies between each other.
Collaboration services	Enable the collaboration and exchange of information between large communities.

Table 2.1: Service likely to appear in a Grid middleware [23]

Since most architectures are starting to implement SOA, the resources that are shared must also be accessible through services. To allow this a group of services is used to make a interface between the Grid middleware and the resource shared. These services allow the management and monitoring of the specific resource and offer a similar interface since the type of the resources shared can be very high. Obviously depending on the type of resource shared we will have resource specific services available.

The upper layer is the user applications that work in a VO environment. These applications interact with the Grid infrastructure by interacting with the services available in each area. This interaction is made through well-defined protocols and APIs. These applications can vary from simple to very complex frameworks [31].

2.2.5 Middleware

The Grid middleware is a group of services that enables the access of users to the resources shared in a transparent way. Besides the access to resources it can also provide monitoring, coallocation, scheduling and security services among others (Table 2.1). The most popular middleware stacks used are the GT, gLite and the UNICORE (UNiform Interface to COmputing REsources). Another interesting aspect in all of the existing Grid projects is the variety of solutions adopted for the middleware. Some use a specific middleware like GT and other use a mix of existing Grid Services or develop their own middleware.

As we can see in table 2.2 that presents the middleware stacks used by some of the largest Grid projects, the diversity is very high.

Projects	Middleware Stacks
ChinaGrid	CGSP (ChinaGrid Supporting Platform) is based on GT
D-Grid	GT, UNICORE, gLite, dCache, SRB, OGSA-DAI, GridSphere, GAT, VOMS and Shibboleth
NAREGI	NAREGI middleware, GT 4.0.1, GSI and WS-GRAM
Open Science Grid	VDT is based in GT, Condor
UK e-Science	2001-2003 – GT, Condor, SRB 2003-2006 – GT, OGSA-DAI, Web Services
EGEE	gLite middleware uses components from several Grid projects namely EGEE, EDG, EDT, INFN-GRID, GT and Condor
TeraGrid	GT: GRAM, MDS, GridFTP, TGCP, RLS and MyProxy

Table 2.2: Middleware Stacks [32, 33, 34]

Despite the diversity we have in the type of middleware used, two of the most popular middleware stack available are the GT and gLite. Next, we describe in more detail these two middlewares with special attention to the gLite the middleware chosen for this work.

2.2.6 Globus Toolkit

GT, currently in version 4.0, is a open source middleware that has been evolving since the mid 1990s. It is presently being developed by a large community of organizations and individual users named *Globus Alliance*. The last release already provides a set of OGSA capabilities based on WSRF. It is very important to understand that GT services can be used to solve simple distributed problems but they are generally used in conjunction with other services for more complex situations [35].

Nowadays, GT services are being used in several grid project like EGEE, NAREGI or TeraGrid (see table 2.2). These services address grid specific concerns like execution management, data access and movement, replica management, monitoring and discovery, credential and instrument management. Most of them are implemented using Java WS. Beside these service there exist containers to host user-developed services written in Python, C or Java, providing them with a common interface and mechanisms that allow, for example, management and discovery functions. Client programs (written in Python, C or Java) can access the GT4 and user-developed services through a set of client libraries [36].

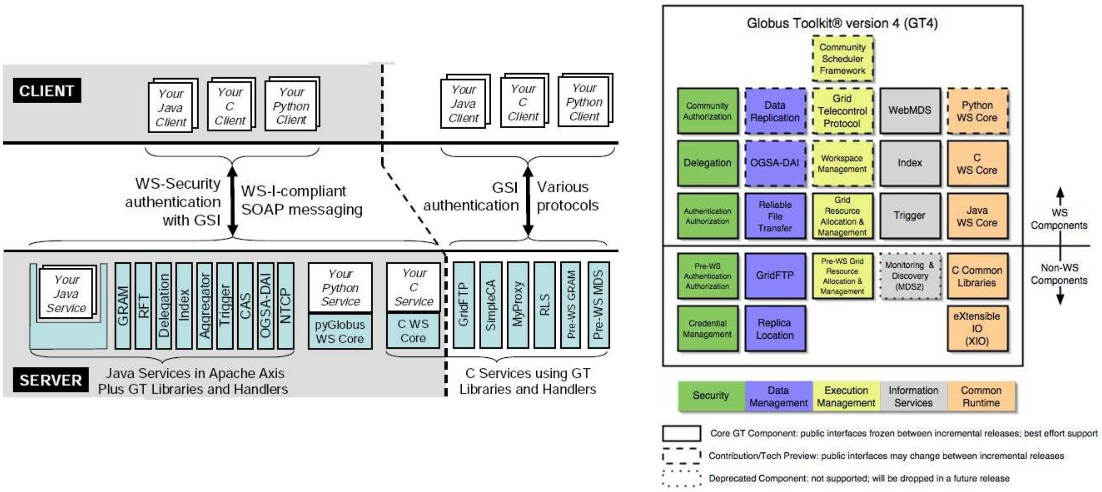


Figure 2.7: Globus components and interactions [36, 35]

To run a specific task we use the GRAM (Grid Resource Allocation Manager) service. This service offers a WS interface that permits initiating, managing and monitoring the execution of tasks. It also allows the user to select some parameters related with execution such as the number and types of resources used and the data that is needed for the execution or that is going to be retrieved after the execution ends. It is also used in other scenarios like service deployment and management where it controls the execution of services and resources consumption [36]..

GT has some services that enable data management. Once again these services can be used individually or in conjunction depending of the requirements. The GT data management services are:

- GT GridFTP implementation – Optimized for reliable and secure, data transfer on high bandwidth networks;
- RFT (Reliable File Transfer) service – Manages multiple GridFTP transfers;
- RLS (Replica Location Service) service – Decentralised service that manages the information and location of replicated files;
- DRS (Data Replication Service) service – Uses GridFTP and RLS to manage data replication.
- OGSA-DAI (OGSA–Data Access and Integration) tools – Provide data access and integration using relational and XML (eXtensible Markup Language) data.

The monitor and discovery in GT4 can be done through standardized mechanisms, based in the WSRF and WS-Notification implementations, built in every GT4 service. These provide the access to resources properties, based in XML. The information is collected and published by the aggregator service Index. Another service, named Trigger, collects information in a event-based. All of this information can be view through the WebMDS service [35, 36].

GT4 security components are also highly based on standards. Despite the fact that it supports several security protocols[1], by default it is used the Transport-Level based security with X.509 public key credentials because it is faster. Besides the GT4 components described above, other services are used in conjunction to support credential management (e.g. MyProxy, PERMIS (PrivilEge and Role Management Infrastructure Standards validation) and VOMS (Virtual Organization Management System)) [36].

2.2.7 gLite

The EGEE project is a 4 year European project funded by EC (European Commission), that aims the creation of a grid infrastructure for e-Science. Originally target to scientific

---

[1] The protocol supported are: Message level with X.509 credentials (WS-Security-compliant implementation) or username/password (WS-I Base Security Profile compliant) and Transport-level security with X.509 credentials

fields of HEP and Life Sciences, it has now expanded to fields that range from geology to computational chemistry. This project is supported by more than 240 institutions from 45 countries world-wide. To enable such a infrastructure, one of the main objectives of this project is the development of a grid middleware that supports the requirements of such a large infrastructure with such a large number of scientific fields, each with its own particular necessities [37, 38].

gLite current version (3.0) is based in a wide number of grid projects like DataGrid, DataTag, Globus Toolkit, GriPhyN, iVDGL (International Virtual Data Grid Laboratory), EGEE and LCG (LHC Computing Grid). This middleware provides high level services that enable the scheduling and analysis of computational jobs, information gathering about the infrastructure, data access and transfer. All these service share a common security framework.

Usually these services are grouped as security, information and monitoring, data, job management or access services (see figure 2.8).

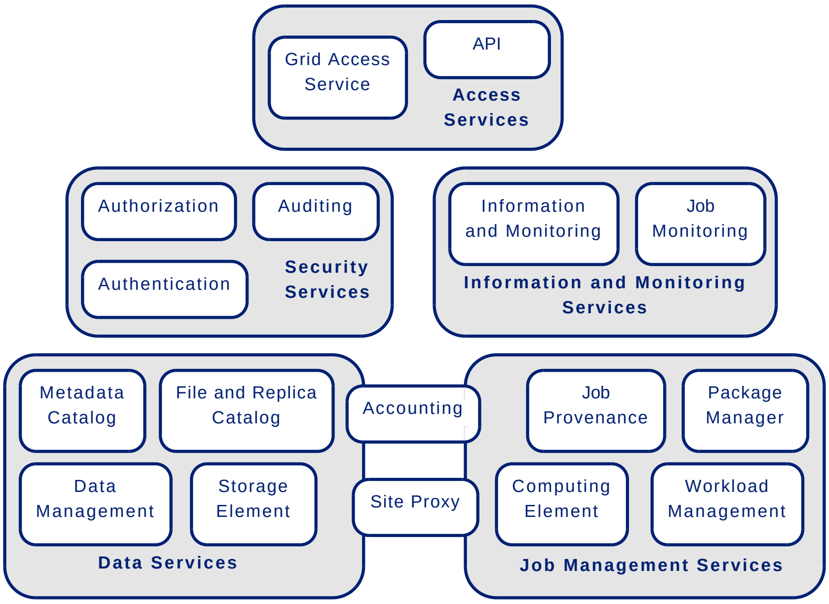


Figure 2.8: gLite Services [38]

The access to a grid infrastructure using gLite is usually done through the UI (User Interface). The UI is a group of CLIs and APIs that provide access to services available in the grid infrastructure. It also allows the authentication of users.

The security services provide the tools for authorization, authentication and auditing. Basically control the access to grid resources and provide information for analysis in case of security relevant events.

Each user, grid resource or service is identified by a digital X.509 certificate. These certificates are composed by a public and private key. The idea is that when a user needs to access a specific resource, he generates and signs a temporary certificate (called a temporary proxy) and uses it to authenticate himself to grid services or resources. Because there are several services that a user request might "encounter" methods have been implemented that allow the proxy to be delegated between services. After the user submits its request he cannot cancel the proxy, so usually user proxies have a short life time, normally 12 hours. This authentication framework allows a user to sign on only once, independently of the services accessed. To manage VOs (see section 2.2.3) gLite uses the VOMS service. This service allows a more controlled access to resources. When a user creates a proxy, the VOMS service is contacted and returns a signed AC (Attribute Certificate) that contains the information relative to that user. Then an extension is added to the proxy generated where the user permissions inside each VO are described. The authentication in the resource can be done through two mechanisms. The first compares the *subject name* contained in the proxy with a local grid-mapfile that maps users to local accounts. The second method relies on the VOMS and the LCAS/LCMAPS mechanism and allows a more detailed control of the user "rights".

Due to the distributed nature of grid infrastructures, the issuing of certificates besides being a highly secure and controlled process, must be widely available to users world wide. IGTF (International Trust Grid Federation) is the international organization, composed by the EuGridPMA (European Grid Policy Management Authority), APGridPMA (Asia Pacific Grid Policy Management Authority) and TAGPMA (The Americas Grid Policy Management Authority) that aims the establishment and maintenance of a global trust relationship between its members. Inside each region we have several CA (Certification authority), one for each country, that are responsible for providing the X.509 certificates in the corresponding country. But before this can be done a procedure must be follow to guarantee, for example, the identity of a user. This process is usually done in a RA (Registration Authority) that are normally distributed throughout each country.

For a user to be eligible to access a grid infrastructure he must follow a simple procedure:

- Request a personal certificate in a local RA;
- Provide proof about its identity in a local RA;
- After having its personal certificate, join the target VO with a specific role;

- Create a proxy using its personal certificate;
- Access grid services using the generated proxy.

In gLite we have two elements that provide uniform access to computer and data resources respectively. The CE (Computing Element) enables a abstraction of the computing resources available in a site. This enable the use of different computing resources, from batch queues of clusters to simple workstations, but providing a common interface for job management and information gathering. It includes the GG (Grid Gate), LRMS (Local Resource Managing System) and a collection of WN (Work Node) 2.9. The GG acts as a interface between the CE and the rest of grid services, is responsible for accepting and dispatching jobs to the WN through the LRMS. Currently gLite has two GG, the LCG CE and the gLiteCE. The LRMS[2] is responsible for sending the jobs to available WNs, basically controlling the resources inside a CE. The jobs are executed in the WNs that provide most of the CLI and API also available in the UI.

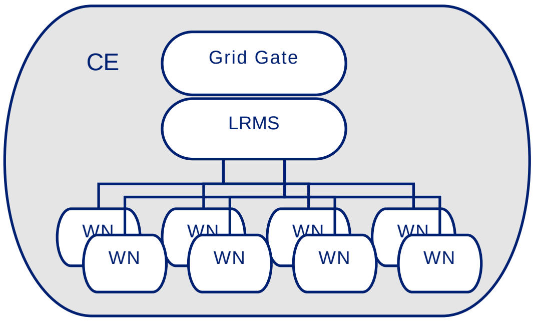


Figure 2.9: gLite computing element components

The SE (Storage Element) allows the access to diverse types of data storage's, ranging from simple disk servers to tape-based storage. There are currently three types of storage elements: DPM (Disk Pool Manager), dCache and CASTOR (CERN Advanced STORage manager). Every one of these SEs can be accessed through a common interface – SRM (Storage Resource Manager).

The protocol used for file transferring is the *gsiftp*. For file I/O operations the protocols used are *GSIDCAP (GSI dCache Access Protocol)* and RFIO (Remote File Input/Output protocol) depending of the type of SE used. RFIO hyas a insecure and a secure version (*gsirfio*).

---

[2] The currently supported LRMS are the OpenPBS/PBSPro, LSF, Maui/Torque, BQS and Condor

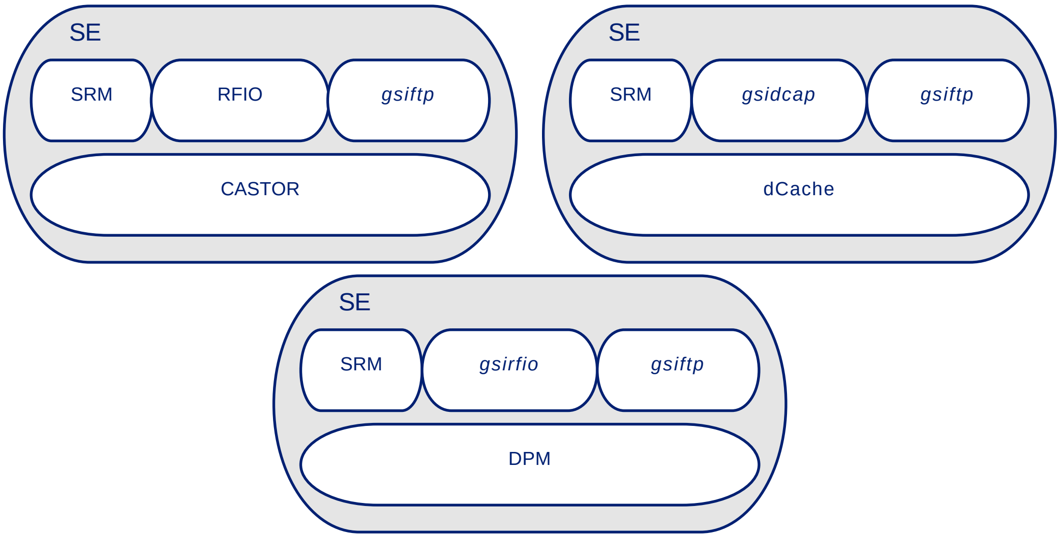


Figure 2.10: Some of gLite storage element types and supported protocols

Job management is essential in a computational grid. It allows job monitoring, accounting, scheduling and execution according to the availability of resources. gLite has several services that are responsible for job management like the already discussed CE, the WMS and accounting services.

To control and monitor the job submission to the CE, the gLite as a meta-scheduler named WMS (Workload Manager Service). Beside managing jobs execution, it also monitors job state through the LB (Logging and Bookkeeping) service. The WMS is composed by several components that are responsible for matchmaking resources and jobs, submit, cancel, monitor and keep a record of the state of each job (LB). The access to the WMS can be made through a network daemon, NS (Network Server), enabling the user to run job execution related commands (e.g. job submission, cancellation or monitoring). Currently the NS is being deprecated and replaced by the WMProxy service that in conjugation with the LB provide a WS based interface with similar functionalities. Another important component of the WMS is the DAGMan (DAG Manager), a meta-scheduler responsible for managing groups of jobs with special dependencies among each other. The WMS is also responsible for the proxy renewal using the MyProxy service. The node where the WMS runs is called the RB (Resource Broker).

There exist two other services, that are still being tested, the PM (Package Manager) and the JP (Job Provenance). The first enables the dynamic deployment of application software and the other provides persistent information about job execution on the grid.

A user before submitting a job has to create a file that describes among other things the type of job, rank, file to execute, input data, output data and requirements. These parameters are described using the JDL (Job Description Language) [39]. One of the parameters that can be defined is the *Input* and *Output Sandbox* that correspond to the location of the files that will be transferred to the WN where the job will be executed and the files that will be retrieve from the WN. The WMS is responsible for the transfer of these files that can be local or remote. Other parameters are related with the target resource like the *Requirements* and *Rank*. The first allows a user to define the requirements a resource must fulfill like the LRMS type or software installed. *Rank* allows the user to define rules so that the WMS is able to decide among the resources that satisfy the *Requirements*. An example could simply be the number of WN available [40, 41].

A new approach to the job description is the JSDL (Job Submission Description Language) that uses a XML-based language to describe jobs. JSDL was proposed by the OGF were it is being developed by the JSDL-WG (JSDL Working Group). Although, JSDL is not currently being supported by the gLite WMS, there are being made efforts to develop a JSDL to JDL converter.

It is also possible to define special types of jobs with specific attributes [39]. The types of jobs available are:

- Job – Simple job that can be one of the following subtypes:
  - Normal – Simple batch job;
  - Interactive – Simple job with its standard streams forwarded to submitting client;
  - MPICH – Parallel application that uses MPICH-P4 implementation;
  - Partionable – Job that can be divided into smaller independent jobs ;
  - Checkpointable – This type of job allows the execution of the program to be paused by defining pause flags in its code;
  - Parametric – Job that has parametric attributes that can be defined with several values. A job instance is created for each value of each attribute.
- DAG – Job with dependencies among each other, described by a DAG (Directed Acyclic Graph);
- Collection - Group of independent jobs.

A user can get information about the job state through the LB service (see table 2.3). The states a job can go through during a execution are shown in figure 2.11.

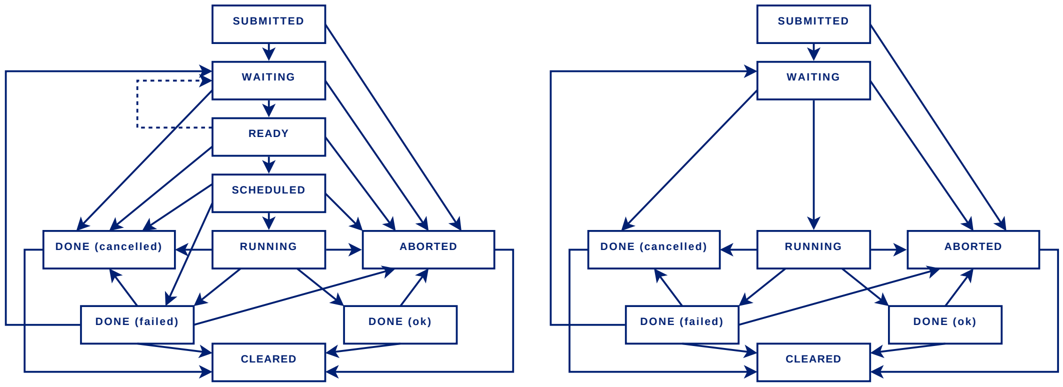


Figure 2.11: States of a normal (on the left) and DAG (on the right) job during its execution

State	Description
Submitted	User submits a job in the UI
Waiting	Job is accessed by the WMS and is waiting for resource allocation
Ready	A resource as been allocate for this job
Schedule	LRMS has accepted the job and is in queue
Running	Job is sent to WN and is being executed
Done	The job has finished its execution and its output is available
Cleared	The output was transferred to the user and the job has been freed
Aborted	Job was aborted by the system
Cancelled	User cancelled the job
Unknown	Status can not be determined
Purge	Job has been deleted from the LB server

Table 2.3: Job States [41]

All the information regarding grid resources and their status is managed by the IS (Information System). gLite has two different IS where much of the published data conforms with the *GLUE (Grid Laboratory Uniform Environment) schema*, a common information model for resource discovery and monitoring [42]. The two information model being used are R-GMA (Relational GMA) and MDS (Monitoring and Discovery System). The first is used to publish accounting, monitoring and user related information while the MDS is being used for resource discovery and status.

The MDS uses the OpenLDAP (Open source implementation of LDAP) information model to implement the GLUE schema. It does not allow secure access.

MDS architecture is based on *Information Providers* installed in each site that gather static and dynamic information relative to that site that is them published by the GRIS (Grid Resource Information Server). The GRIS is a a LDAP (Lightweight Directory Access Protocol) server that is usually installed locally in the resource. Then in each site a GIIS (Grid Index Information Server) collects the information published by the existing GRIS and republishes this information to a higher level GIIS. The information gathered by the GIIS is stored in a BDII (Berkeley Database Information Index).

The R-GMA is based on the GMA (Grid Monitoring Architecture) initially proposed by the GGF. R-GMA architecture is based in three main component:

- *Producers* are responsible for gathering information and informing the *Registry* about the information they are publishing and how it is accessed;

- *Consumers* contact the *Registry* to discover what *Producers* publish the target data and how it can be retrieved. Them they contact directly the *Producers* for the target data;

- *Registry* contains the information about the data and structure that each *Producer* has.

The information presented by the R-GMA is in the form of virtual database of virtual tables which structure is defined by the *Schema*. The R-GMA system is defined by the *Registry* and *Schema*.

Each site as a *MON box* that contains the *Producers* and *Consumers* services. Presently R-GMA is used for accounting, system and user monitoring.

In Grids data is managed, like in computers, with files. Data grids, as it was discussed in section 2.2.1, enable the user to look at a distributed data as it was a single pool of files. These files can be geographically distributed and also replicated throughout several sites to provide a more efficient access. Because of this, files can only be read or deleted to guarantee their consistence. gLite provides several services for data management like the SE, the file and replica catalog, a metadata catalog and data management services. The access to data is controlled by ACL (Access Control List) that attached to files. File metadata is supposed to be saved in application specific metadata catalogs.

Grid files can be identified by a GUID (Grid Unique Identifier), LFN (Logical File Name), SURL (Storage URL) and TURL (Transport URL).

The GUID is unique identifier, based on the UUID (Universally Unique Identifier) standard, that identifies every single file (e.g. *guid:81adf875-ccd7-44fa-89b5-0fef8ae9cd34*).

LFN provides files with a human readable name similar to the ones found in computers file systems (e.g. *lfn:///grid/dteam/fmriData/exemplo.nii.gz*).

SURL or PFN (Physical File Name) is used to identify files or replicas that are in a specific SE. The name has a prefix (*srm or sfn*) that identify SE with or without a SRM interface.

In the case of SE that do not have a SRM-based SE the SURL has a specific structure that identifies the host and the physical location of the file (e.g. *sfn://bone.ieeta.pt/data/dteam /grid/dteam/fmriData/exemplo.nii.gz*).

On the other side, in SRM-based SE's the SURL can or not include files physical location. Normally SRM-based SE use virtual file systems to map files (e.g. *srm://bone.ieeta.pt/dpm /ieeta.pt/home/dteam/grid/dteam/fmriData/exemplo.nii.gz*).

TURLs provide the necessary information for accessing files in SE like hostname, path, protocol and port. Due to the fact that TURLs are obtained dynamically, users should be aware that they may change over time. In a SE there can be more than a TURL for each file, because the SE can have multiple access protocols and the SE might have several copies of each file for load-balancing.

The data management can be made through the use of CLIs or APIs available in the WN and UI.

LFC (LCG File Catalogue) is the file catalogue adopted by gLite. This service provides several file related functions like mapping between the GUID, SURL and LFN (see figure 2.12), system (e.g. file size and checksum) and user metadata and replicas information. It also allows a user to attach to a file a ACL.

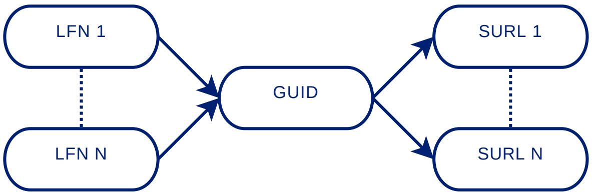


Figure 2.12: Relation between the LFN, SURL and GUID

Although LFC provides some basic metadata capabilities, in some applications,like biomedical, there is a need for more complex metadata. To overcome this problem gLite chose as its official metadata catalogue, the AMGA (ARDA Metadata Catalogue) [43].

gLite as two services, APEL (Accounting Processor for Event Logs) and DGAS (Distributed Grid Accounting System), that are responsible for accounting.

2.2.8 Grid Projects

Nowadays we have a large diversity of Grid projects all around the world. Some of these projects are national initiatives, other include several countries from all the world. These projects can also be associated with a specific scientific area like for example health care or astrophysics or include a wider range of scientific communities like the EGEE and EELA (E-infrastructure shared between Europe and Latin America) [44, 39]. Most of these project start from the need of a specific scientific community that can be distributed worldwide.

Some communities have very specific requirements, an example is the GRIDCC (Grid Enabled Remote Instrumentation with Distributed Control and Computation) and the Int.eu.Grid that work in interactive Grids [45, 46].

The increase in the number of countries that are promoting National Grid Initiatives whose main objective is the creation of a national research infrastructure show a growing awareness of the importance that Grid is gaining by part of governments. International Grid initiatives that are normally associated with a specific community like the HEP are also encouraging the collaboration and exchange of knowledge. A simple example is the AccessGrid that allows the support of group-to-group integration through Grid or the creation of large distributed digital libraries such as the DILIGENT (A Digital Library Infrastructure on Grid Enables Technology) project [47, 48].

There also initiatives whose aim is joining infrastructures from different continents like the EELA project that is working with Latin America and European infrastructures .

Since the number of Grids is increasing the question of interoperability is gaining importance. In response to this problem national projects are appearing with the aim of source key software components that can interoperate across several Grid middlewares. Several national initiatives exist like OMII-UK (Open Middleware Infrastructure Institute for United Kingdom), NMI (National Science Foundation's Middleware Initiative), C-OMEGA (Open Middleware Enabling Grid Applications) and OMII-China (Open Middleware Infrastructure

Institute for China). To complement these efforts, the OMII-Europe (Open Middleware Infrastructure Institute for Europe) project was created having 16 partners (8 European, 4 USA, 4 Chinese). Its goal is to re-engineer, based in open standards, and not develop new software components to guarantee the interoperability between existing Grid infrastructures [49].

2.3 The HealthGrid

Nowadays it is common to talk about e-Health as the application of ICT (Information and Communication Technology) to the services related with the health sector. This concept as allowed the increase of productivity in healthcare by providing tools, such as health information services, that assist prevention, treatment, health monitoring and lifestyle management [50].

The emergence of grids enabled the interdiscipline research in biomedical areas such as medical informatics, bioinformatics or system biology, creating new opportunities for research. A infrastructure that allows sharing heterogeneous and disperse medical relevant data, that can be used for processing and can be accessed by actors of healthcare in a secure manner, according to their authorization, is called a healthgrid [51, 52, 53]. To support the development of healthgrids the HealthGrid initiative was created.

The vision shared by the community behind the HealthGrid initiative, is the *creation of an environment where information at the 5 levels (molecule, cell, tissue, individual, population) can be associated to provide individualized healthcare* [54, 55].

When we talk about healthgrids, we are talking about a diverse group of communities that vary from life sciences to drug discovery and medical research, but have a common goal that is the improvement of life quality.

In life sciences the grid technology is specially important in addressing biological data complexity and in allowing the interoperability between the large number of databases that provide specific representations of biological data like Embrace [56]. The last is a key aspect in the development of models of living organisms. In molecular biology, grid also play a important part in comparative data analysis, mandatory in most of molecular biology data analysis workflows. Also important, is the constant need for molecular biologists to access databases to retrieve information related with their research. Because of this it is important to distribute databases, make them accessible to biologists and provide the resources to analyse them.

Drug discovery is also a area of research that is extremely important to improve life quality and fight epidemics. In silico drug discovery is one of the most promising strategies to decrease the time of drug development process. The first step in silico drug discovery, consists on the computation of the docking probabilities for millions of ligands, can be easily run in a grid infrastructure. This has already been done in the WISDOM (Wide In Silico Docking On Malaria) initiative where in 6 weeks were calculated 46 million ligands corresponding to a total of 80 CPU years. This initiative used the EGEE infrastructure in which 1000 computers were simultaneously used in 15 countries around the world [57].

Another area of research that is closely related with healthcare is research in medical imaging. Since medical imaging plays a key role in diagnosis, therapy planning and treatment follow-ups, the amount of data produced in hospitals is increasing every year. Moreover in some countries this data must be accessible to patients and so hospital are obligated to keep archives for 7 to 20 years. In Europe it is estimated that the volume of data produced in hospitals is comparable with the one expected for CERN (European Organization for Nuclear Research) LHC (Large Hadron Collider) which is in the order of Peta Bytes per year [58, 59, 52].The amount of medical data available, if accessible, could boost epidemiology and pathology studies. Grid technology appears as the ideal candidate to create medical federated databases and provide the resources necessary to analyse this data [60].

Despite of the promising results in projects from other areas that are using grid technologies to share data, medical related data have special requirements. The major issues that need to be addressed [51], before the HealthGrid vision can be fulfilled, are:

- **Grid technology**: Current middleware solutions do not offer the security, stability and scalability necessary to run healthcare applications. There are solutions, like gLite and UNICORE, that have demonstrated their scalability but have limitations in data management. Other, like the SRB (Storage Resource Broker) have powerful data management capabilities but do not provide job management. Furthermore, most of these middlewares do not offer standard interfaces;

- **Deployment**: There exist several limitations to deploy grid nodes in healthcare facilities like as hospitals. These limitation are the security requirements, the lack of friendly interfaces, the interoperability between Grid services and existing data management solutions already adopted and the difficulty in installing the grid nodes;

- **Standards**: Before the data can be shared international standards and mechanisms that allow for example the anonymization and pseudonymization of data must be defined;

- **Management**: The concept of VO is not flexible enough in managing healthgrids. It must be possible, for example, to define VO of VOs.

Although these restrictions pose several problems in the implementation of healthgrids, specially the ones that deal with medical data, there exist several projects that are developing solutions to some of these problems. Beside the ones already referred, there are other healthgrid projects like the SARSGrid (SARS Grid), caBIG (cancer Biomedical Informatics Grid) and BIRN (Biomedical Informatics Research Network).

The SARSGrid is a AccessGrid based collaborative platform that was used, in 2003, to share and discuss SARS (Severe Acute Respiratory Syndrome) patient's medical data between healthcare professionals to diagnose, treat and monitor home and hospital quarantined SARS patients in Taiwan [61].

The caBIG is a initiative that wants to link researchers, physicians and patients in the cancer community to foster the research in this area. This way they are able to share research results, information and foster the collaboration between researchers [62].

The BIRN is a infrastructure that supports neuroimaging research. Its is divided in three testbeds that research specific areas in neuroimaging:

- Function BIRN: Develop tools and methods that solve problems associated with multisite functional MRI;
- Morphometry BIRN: Analyze data from a large group of subjects, suffering from memory dysfunction or depression, that came form different neuroimaging sites and study structural differences;
- Mouse BIRN: Use mice to study neurodegeneratives diseases. Analyze multi-scale structural, functional, genomic and gene expression data acquired from mice's brain.

The information technology infrastructure that enable the research in BIRN is managed and supported by the BIRN-CC (BIRN Coordinating Center).

2.3.1 Supporting Healthgrids

The HealthGrid initiative, a international open initiative born in Europe in 2003, is attempting to raise awareness about the grid technologies and their importance. The goal of this initiative is to create a fully operational European/International healthgrid described in the previous section. The technical problems, already discussed, raise more complex issues that are related with ELSE (Ethical, legal and socio-economic) aspects of such infrastructure.

To overcome these problems the Share (Supporting and structuring Healthgrid Activities and Research in Europe) project was created, funded by the EU (European Union), with the objective to establish a 10 year strategy that would enable the successful deployment of healthgrids. This strategy includes the definition of a road map to identify and undertake important issues that range from technical to ELSE issues. From the technical point of view, as it was described before, the main problems are related with the management of personal data, that obey country specific regulations, technical solutions, standardization and communication issues. To solve this it was proposed a roadmap [63, 51] with seven milestones divided into three categories:

- **Deployment Milestones**
  - MD1 “Computing Grid” – Deployment of computing grid nodes inside European research centres;
  - MD2 “Data Grid” – Deployment of a data grid nodes inside European research centres;
  - MD3 “Research K-Grid” – Deployment of a knowledge grid nodes inside Europeans research centres;
- **Standards Milestones**
  - MS1 “Grid DICOM” – Creation of a standard that enables exchange of medical images on the grid;
  - MS2 “Grid EHR” – Creation of a standard to allow the exchange of EHR (Electronic Healthcare Records) on the grid.
- **Technical Milestones**
  - MT1 “Grid middleware testing” – Test grid middleware with medical applications for scalability and robustness;
  - MT2 “Production of a reference distribution” – Development of a reference distribution simple and fast to install but should accompany the evolution of the standards for Web Services and grid security services.

From the ELSE point of view the main problems have a similar origin to the technical problems. The main problem continues to be personal data processing and transfer but also liability issues since the products and services offered by a healthgrid can, for example, influence in a decisive manner a diagnostic of a patient [63, 64]. To solve the ELSE problems it was also defined a roadmap with seven milestones divided into three categories:

- **Ethico-Legal Milestones**
  - MEL1 – Determine the responsibilities of each of the healthgrid actors;
  - MEL2 – Study how the required patient consent it should be obtained and recorded for data processing and transfer.
- Data Protection Milestones
  - MDP1 – Identify patient identification issues, analyse and evaluate the state of art of medical image de-identification tools available;
  - MDP2 – Develop, deploy and test identity protection techniques, such as pseudo-anonymisation, in the grid infrastructure.
- **Policy Milestones**
  - MP1, MP2, MP3 – These milestones are related with the deployment of computing, data Grid and knowledge Grid. There objective is to identify the privacy, confidentiality, ethical and quality policies of each grid respectably.

Figure 2.13 presents a complete integrated view of the described road map and shows how they are intimately connected. This interconnection of technical and ELSE issues allow us to see the dimension and complexity of such a infrastructure but at the same time its great potential.

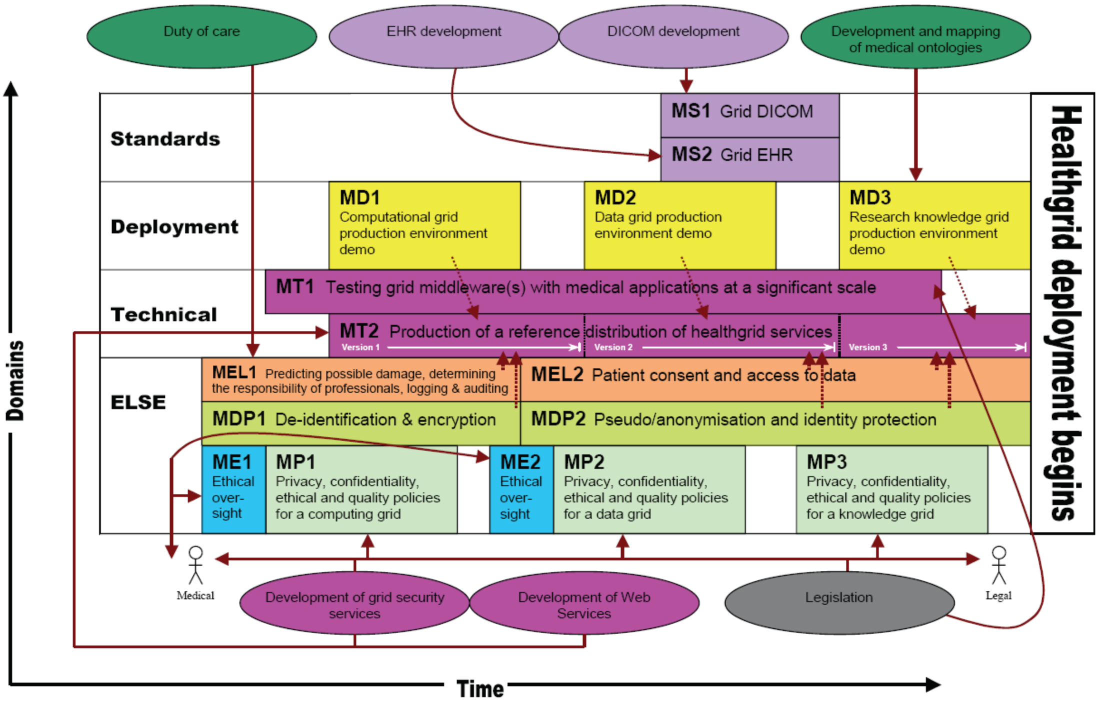


Figure 2.13: Share Integrated Road Map 1 [63].

Chapter 3

Multi-voxel fMRI Association Analysis

3.1 A new approach to fMRI analysis

We can define an analysis as being parametric or non parametric. The difference between these two techniques is that the former assumes a model with parameters, while the latter does not make any assumption in the model used.

Traditionally, fMRI (functional MRI) analysis has been supported by the SPM (Statistical Parametric Mapping) [65] package. By assuming a linear model that relates the brain function, specific events sequences with the BOLD (Blood Oxygen Level Dependent) response and the brain spatial volume, SPM is able to provide estimates of brain areas more or less correlated with the study target activity, while providing statistical significance levels. The problem with SPM is that no model clearly relates studied tasks and the BOLD response in the brain. This goes against the response models and linearity assumptions made. This can also question the use of general linear models (GLM) with SPM. Another problem is that by assuming specific delays between the activation model and the target event, we are ignoring information.

Non-parametric approaches on the other hand are not based on accepted assumptions but rely on a hypothesis that depends on relating spatial areas with relevant occurring events [66]. Using non parametric association measures (linear or non linear) in this scenario avoids having to assume considerations on the model of the BOLD response and fMRI protocol.

Our approach is to start using non parametric association measures on fMRI analysis, namely *multi-voxel fMRI association analysis*. We used the r^2 (linear correlation) and V (non linear) association measures. The advantage of using this type of analysis is the fact of not making any assumptions about the models used, like in SPM, that might condition the analysis or the results obtained. This approach has already been successfully used in the study of EEG (Electroencephalogram) signals [67, 68, 69].

3.2 Method

The method consists in calculating for each pair of voxels an association measure. This is done by first defining a target voxel or a ROI_1 (Region Of Interest) that we would like to study, select a ROI_2 and then calculate the association measure between the two target regions (see figure 3.1). This analysis will result in a number of association measures equal to the number of voxels present in the $\mathrm{ROI}_1 \times \mathrm{ROI}_2$.

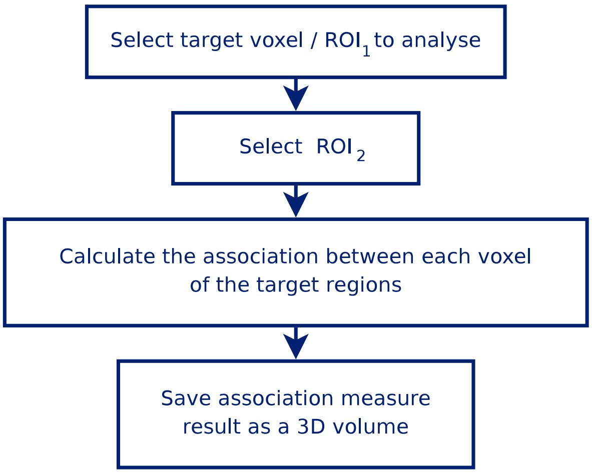


Figure 3.1: Running a Multi-voxel fMRI association analysis

The association measures allow us to obtain the level of the relation between two series. This type of measures has already been used to study and analyse specific biological signals. A successful example was the use of h^2 and V association measures that were used to study EEG signals [67, 68, 69].

When we use association measures we are comparing two series and basically doing pair-wise analysis. In our particular case we decided to test and implement the linear correlation and V that are a linear and a non linear association measure.

3.2.1 r^2

Linear correlation is a simple association measure that allows the comparison between two series of number, assuming that they have a linear relation. Its is calculated through the following formula:

$$\mathrm{corr}(X,Y) = \frac{1}{1-n}\sum_{i=1}^{n}\frac{(x_i-\overline{x})(y_i-\overline{y})}{S_xS_y}, \quad {S_x}^2 = \frac{1}{1-n}\sum_{i=1}^{n}(x_i-\overline{x})^2, \quad {S_y}^2 = \frac{1}{1-n}\sum_{i=1}^{n}(y_i-\overline{y})^2$$

The linear correlation coefficient ($corr(X,Y)$ or r) varies between -1 and 1. If r is negative this means that the series have a negative linear relation or that when one grows the other tends to decrease. Negative correlation is higher when it is closer to -1. Positive correlation means that the series have a similar behaviour. The correlation is higher when it is closer to 1. If r is equal to 1 then the series are exactly the same [70].

3.2.2 V

The V measure is a nonlinear association measure, that is faster to compute and yet more robust to nonlinearities than the classical cross-correlation ratio (r^2). This measure belongs to a class of statistical association measures named "general association measures". V is a measure, proposed by Cramér, derived from Pearson'coefficient of mean square contingency (ϕ^2) [67].

The statistical association measures are usually applied to two dimensional contingency tables. These tables show the frequency of two or more variables f_{ij}, the sum of each column f_{i+} and row f_{+j} and N that corresponds to the sum of all frequencies as follows:

$$\begin{array}{cccc|c}
f_{11} & f_{12} & \cdots & f_{1i} & f_{1+} \\
f_{21} & f_{22} & \cdots & f_{2i} & f_{2+} \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
f_{41} & f_{42} & \cdots & f_{4i} & f_{4+} \\
\hline
f_{+1} & f_{+2} & \cdots & f_{+i} & N
\end{array}$$

To calculate the V association measure we must first build a contingency table, apply the ϕ^2 to the table and then retrieve Cramér V (see figure 3.2). In our method, the classes chosen to build the table were the signal polarity and slope. Using these classes we obtain the following table

	$+\nearrow$	$+\searrow$	$-\nearrow$	$-\searrow$	
$+\nearrow$	f_{11}	f_{12}	f_{13}	f_{14}	f_{1+}
$+\searrow$	f_{21}	f_{22}	f_{23}	f_{24}	f_{2+}
$-\nearrow$	f_{31}	f_{32}	f_{33}	f_{34}	f_{3+}
$-\searrow$	f_{41}	f_{42}	f_{43}	f_{44}	f_{4+}
	f_{+1}	f_{+2}	f_{+3}	f_{+4}	N

where $\nearrow$ means a positive slope, $\searrow$ is a negative slope, $+$ is a positive signal and $-$ a negative signal.

After generating this table for the two target time series we are able to calculate the V value using the Pearson's coefficient of mean square contingency (ϕ^2) given by

$$\phi^2 = \sum_{i=1}^{n} \left(\frac{f_{ij}^2}{f_{i+} f_{+j}} \right) - 1$$

From ϕ^2 we can extract Cramér's V using the following relation

$$V = \sqrt{\frac{\phi^2}{N.min[(I-1),(J-1)]}}$$

where I and J are the size of a contingency table (in our case is 2×2) [67].

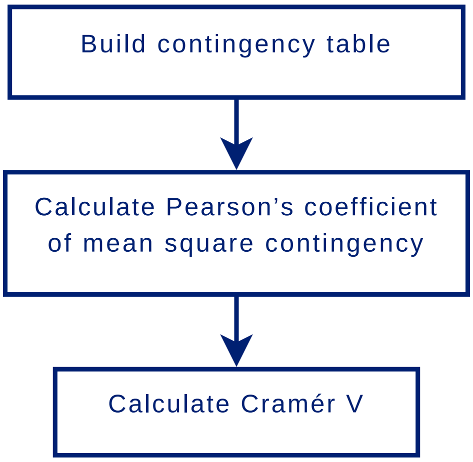


Figure 3.2: Calculate V association measure

3.3 Some Considerations

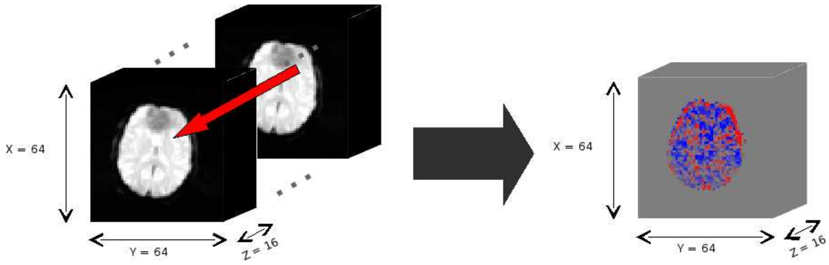
The *Multi voxel fMRI analysis* consists on calculating the association measure between two fMRI series. This is done for all the voxels in a specific region. If we consider a fMRI sequence taken from bottom to top of a head, each 3 sec, during 5 minutes we will get 100 3D volumes, each with $16 \times 64 \times 64$ voxels. The association of one voxel with all the others will generate $16 \times 64 \times 64$ values. To ease the visualisation of this data we save it in a $3D$ volume where each value corresponds to the association measure, V, of the target voxel with the voxel in the corresponding position.

In this case we do a pair-wise analysis where we will have to calculate $16 \times 64 \times 64 \times 16 \times 64 \times 64$ association measures, approximately 4.3 billion combinations, considering a non-symmetric association analysis (e.g. the h^2 association measure). If we consider delay of N we will have a increase by a factor of N (e.g. 3 sec delay will correspond to approx. 13 billion combinations).

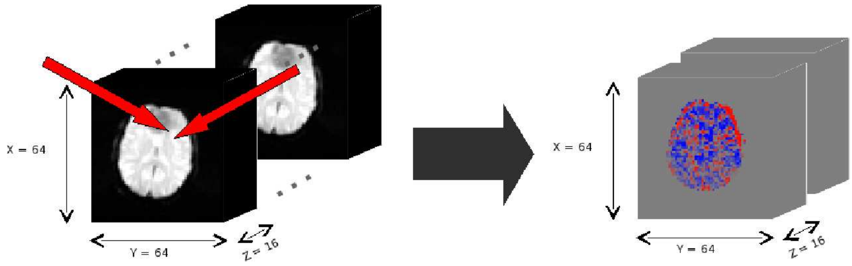
This number can be reduced if we consider a smaller ROI (Region Of Interest). Nevertheless this type of analysis requires a lot of computational power, making it unpractical if used in environments where time is a critical factor.

For this work we tested two association measures: linear correlation and V.

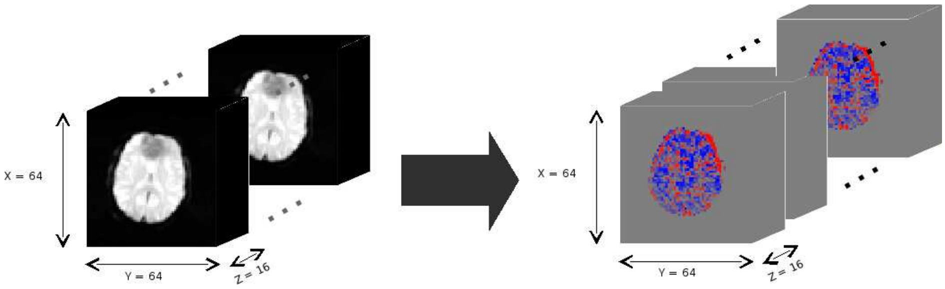
This type of analysis is very computationally intensive not because of the complexity of the association measures but because of the large number that have to be calculated. HPC (High Performance Computing) solutions like *Grid* are necessary to support and promote the use of these type of fMRI analysis.



(a) Association analysis using one voxel



(b) Association analysis using two voxels



(c) Association analysis using N voxels

Figure 3.3: Multi-voxel fMRI association analysis (on the left) and corresponding results (on the right)

Chapter 4

A Grid framework for fMRI research[1]

4.1 Solution Concept

Grid infrastructures are being successfully used, to handle the demanding requirements that are associated with the storage, communication, process and enable complex analysis workflows of medical imaging data [60, 59, 73]. Being a new paradigm in distributed computing it provides us the opportunity to work with state of the art technology and embrace old problems with a new approach. One of the great advantages of this technology is the capacity to manage resources between different institutions and the fact that it provides a great scalability.

In medical imaging they can provide the seamless aggregation of distributed resources, computational and storage, high-bandwidth networking but also the necessary security mechanisms to manage personal identity and the access to data through the use of personal certificates and VO (see section 2.2.1).

In the particular case of the *Multi voxel fMRI analysis* that is very computationally intensive, we need a platform that can provide us the computational power needed (see section 3) [71, 72]. A epileptic patient that is admitted for decision of surgery within a three to five days period, can have his fMRI exams performed in the first or second day. For the *Multi voxel*

---

[1] The contents of this chapter were partly published in the Proceedings of Healthgrid 2007 and in Ibergrid 2007 [71, 72]

fMRI analysis to be of help for the clinical decision, the results must be made available in less than 24 hours. Grid appears as a natural solution to this problem since it already covers some of the needs associated with the secure access and processing of distributed medical databases and can provide the resources needed to run this analysis.

We propose to develop a Grid enabled framework that allow us to run and monitor the evolution of *multi-voxel fMRI association analysis* using a intuitive interface. The target users are healthcare professional, neuroscientists and biomedical engineers. Both have different needs and necessarily different uses for this framework. The healthcare professionals and neuroscientists want to submit a analysis and monitor its evolution. In this case we assume the fMRI data is transferred to the system automatically via a process similar to the one used by the gLite Medical Data Manager [58]. In the case of a biomedical engineer, its main objective would be to develop new methods of analysis using existing reference datasets or special cases. So they must be able to register new data and new analysis methods to the infrastructure. The needs of the neuroscientist user are the same as the biomedical researcher. There must also exist a system administrator who is able to add new analysis and data to the system making them available for every user (see figure 4.1).

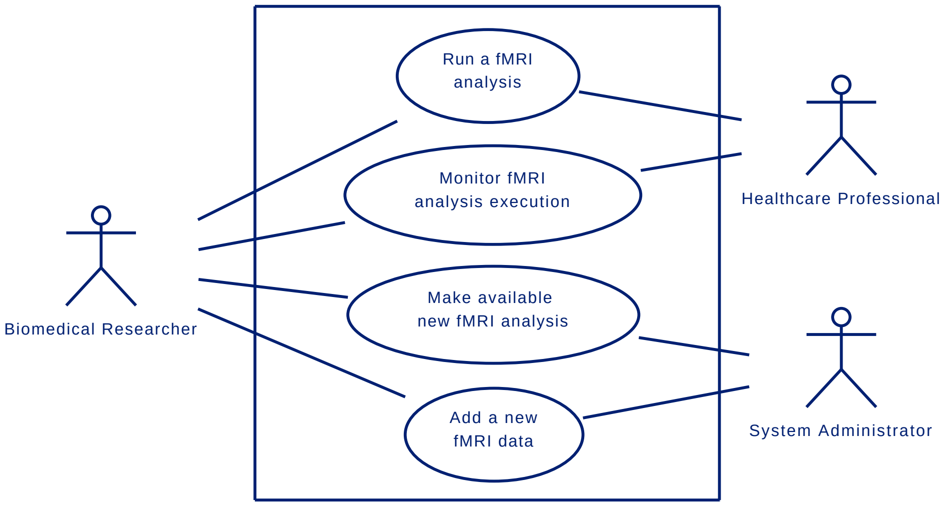


Figure 4.1: Use case diagram

Ideally a user, to run some type of analysis would first need to be authenticated then he would select the analysis and submit it through a web portal. The request to run the application would then be delegated to a application layer where specialized services would retrieve all the information needed to generate the Grid jobs and posteriorly submit and monitor them. When the analysis was completed the results would be generated and then the user would be able to retrieve them. These steps can be mapped to the proposed infrastructure (see figure 4.2).

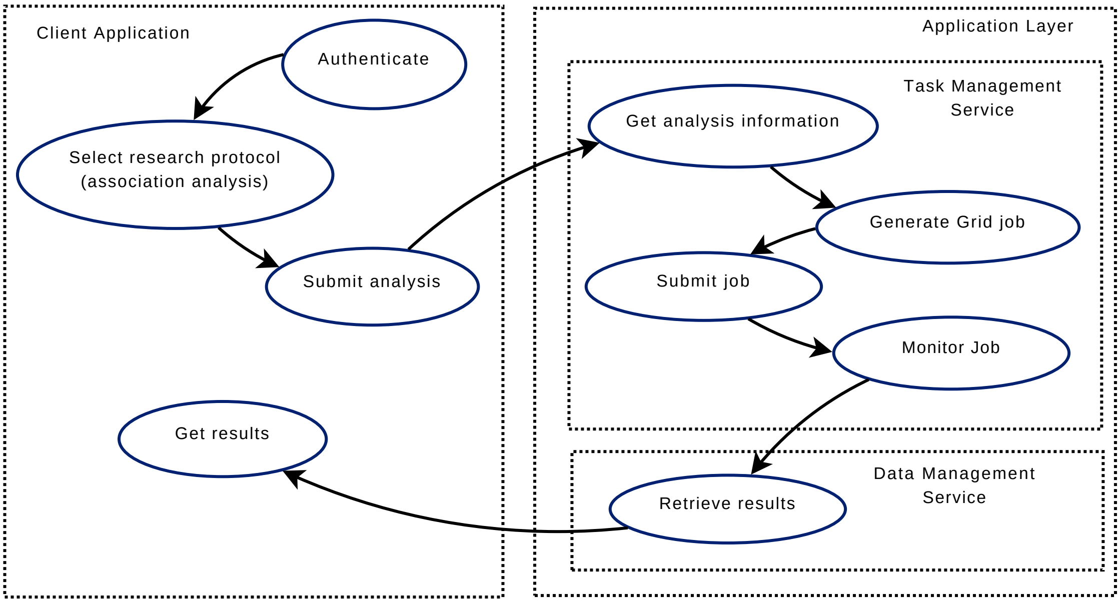


Figure 4.2: Ideal workflow when running an fMRI association analysis on the Grid

The association analysis that we want to run can be applied to different regions of the fMRI volume independently. We can for example divide a volume into smaller volumes, process each one independently and then use the different outputs to reconstruct the final result (see figure 4.3). This enables a natural parallelization of the association analysis. This is ideal in a Grid environment because the analysis of each volume partition can be mapped to the creation of one job.

In reality we do not divide the volume into smaller parts but apply the analysis to smaller parts of the region that is going to be analysed. Another possible solution would to previously divide the volume into smaller parts and then run the analysis in each one. In a first approach this would look like a better solution because we would have to transfer smaller volumes over the Grid infrastructure. However, we assume that the data is already stored in the Grid infrastructure and so, to divide the volume we would have to first do some pre-processing (divide the data) and then run the analysis in each of the generated volumes. The problem is that the Grid middleware introduces a large overhead and the cost of having to pre-process the data is larger than transferring the entire volume. This is true because the data that we are currently analysing is quite small (approximately 13MB). However we know that in the future this might not be true, specially when working with large volumes of data.

Nevertheless, the solution we choose to implement can be easily adapted to support the other discussed above.

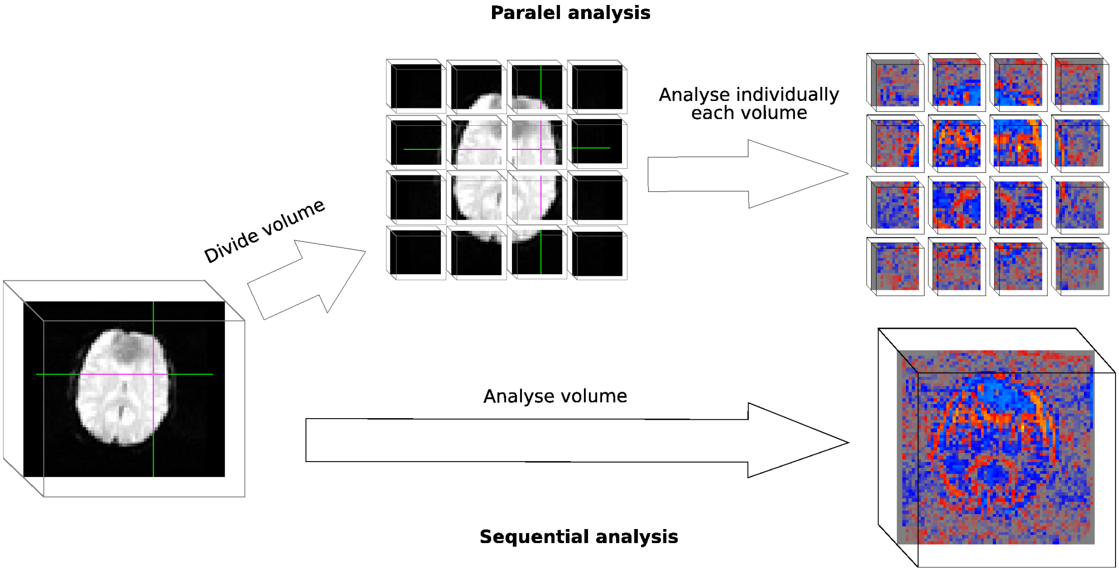


Figure 4.3: Running the association analysis using a Grid infrastructure vs a sequential implementation

To enable the use of the proposed framework by its target users, it is vital to provide a intuitive and simple work interface. To ease the access and avoid installation issues, this interface was developed as a web portal from which users harness from the computational resources available. This portal provide users the ability to select the type of analysis, the target data and manage the execution of the analysis.

The portal delegates the user requests to another service that works as an interface to the Grid environment, completely isolating users from the complexity associated with the Grid resource management.

The medical data we are processing is stored in specific repositories, normally in platforms that support the DICOM (Digital Imaging and Communications in Medicine) standard [74]. Because the access to data is done through the Grid middleware, it is vital that it provides services that allow the access to these specific data repositories. Several services are being developed for this purpose, like the Medical Data Manager that provides anonymization, security and integration with the DICOM servers [58] and the MEDICUS (Medical Imaging and Computing for Unified Information Sharing) project [75]. The first is being developed and tested with the gLite middleware.

The objective of this work is to make a proof of concept infrastructure to show the feasibility of this type of solution. Despite all the considerations made, namely in authentication, data security and management issues, only the essential services necessary to run the proposed fMRI specific analysis were implemented. This is the reason because the proposed architecture differs a little from the real implementation. These services are described in more detail in following section.

4.2 System Architecture

To ensure that the users would be able to access the resources available provided by the Grid infrastructure we devised a three layer architecture. On the top of the layer we have a web portal that works as a interface between the user and the rest of the infrastructure. This portal provides the functionalities necessary for the user to access and use the Grid resources. The user requests are then delegated to the application layer, *BImageG*, that stands for brain imaging Grid services. This layer services will then interact with the Grid infrastructure, more precisely with the Grid middleware service. So in the base layer we have the Grid infrastructure that is composed by the Grid middleware and the Grid sites that provide the Grid resources (figure 4.4). The interaction between the Grid resources and the BImageG services is done through the middleware services.

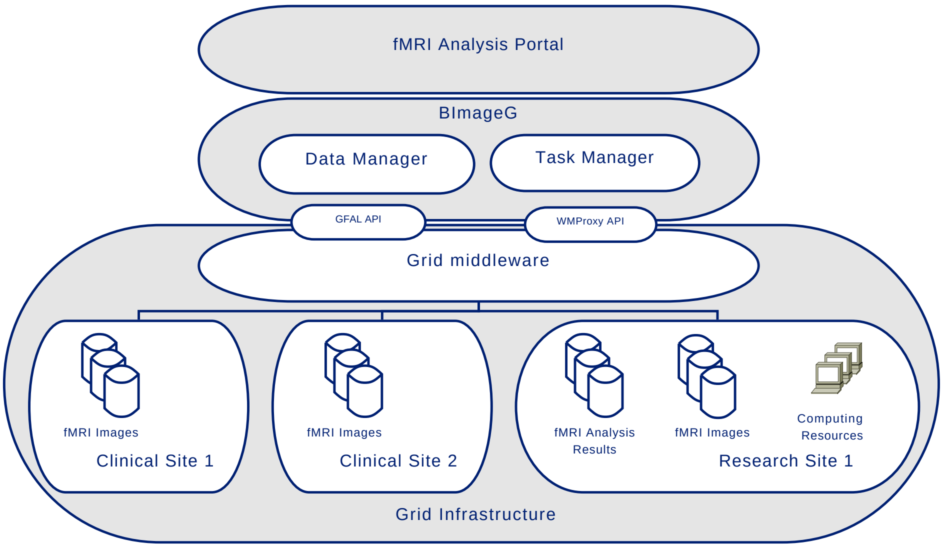


Figure 4.4: fMRI analysis framework, building on top of Grid services

With this architecture we can create a research framework that can harness the Grid resources in a simple and transparent manner and maximizing the use of the services provided by the Grid infrastructure.

4.2.1 Grid infrastructure

The basis of a Grid infrastructure is the middleware that provide us with the tools to manage and access all the resources that make part of such infrastructure (see section 4.1 and 2.2.5). The resources provided by the Grid infrastructure are scattered through out several sites that can be locate on healthcare institutions or in research institutes. These can provide several types of resources like data storage or computing resources. The data in our particular case is fMRI image sets and corresponding analysis results.

The middleware selected was the gLite that is currently being developed and used by the European project EGEE. Besides being a state of the art middleware it provided us with the opportunity to collaborate with the EGEE project that has some characteristics that were extremely important for the development and validation of this work, namely:

- Largest Grid infrastructure in the world – *WLCG/EGEE*[2];
- The development and maintenance of the Grid infrastructure is supported by a highly specialized community;
- Provides medical data management specific Grid services;
- Large biomedical community – *biomed* VO.

gLite provides APIs, normally in C++, Java or WS and CLIs to access middleware core services. The integration of the Grid services with this framework services is described in more detail in its implementation.

We used the EGEE PPS testbed to simulate our Grid infrastructure. The PPS testbed is a Grid infrastructure, composed by about 30 sites, that is used to evaluate new services, middleware updates and test new applications before deploying them into the production testbed. In our particular case we used our site as a repository for the fMRI data and corresponding analysis results. The computational resources used where the ones available in the PPS testbed.

---

[2] The WLCG (Worldwide LHC Computing Grid Project) was created to prepare the computing infrastructure necessary for the LHC experiment. Because the EGEE and the WLCG share a large part of their infrastructure, its normally referred to as the WLCG/EGEE Grid infrastructure.

4.2.2 BImageG

The interface with the Grid infrastructure is done through the *BImageG* layer. This layer has two main services: the Task Manager and Data Manager. These provide the functionalities needed to run and monitor the selected task in the Grid infrastructure. They interact directly with the Grid middleware services through its APIs and CLIs.

4.2.2.1 Task Manager

The main functions of the Task Manager service are to parse specific tasks to JDL files, implementing specific job workflows, so that they can be submitted to the Grid infrastructure. It is also responsible for the continuous monitoring of each job state and the correct management of the data generated during the job workflow. The jobs generated are also divided into smaller jobs so that they run in parallel, allowing a more efficient use of resources.

4.2.2.2 Data Manager

The Data Manager service main responsibilities are to keep track of the files stored in the Grid and provide simple data operations like data retrieval and storage. It can also provide more complex operations, namely fMRI specific data operations, such as normalization, fMRI 3D volumes data extraction, data conversion and retrieval.

4.2.3 fMRI Analysis Portal

The interface with the user is a web portal. This way it is possible to guarantee a flexible, familiar and easy to use interface. It should provide the user, from a analysis management perspective, the following functionalities:

- Define task parameters;
- Monitor tasks;
- Task specific operations (start, cancel, restart);
- Provide a dynamic interface that can be described in a XML file.

This interface should be flexible enough so that it can be easily adapted for new types of analysis and also be capable of supporting user authentication and data management.

4.3 Running a fMRI Analysis

From the a fMRI analysis creation to its execution, there are several steps that must accomplished by the different services present in the proposed architecture.

A user to run a analysis must first create a task where he describes the analysis type and all its parameters. To do this he accesses the fMRI analysis Portal, fills the information relative to the task namely, task id, analysis type and respective parameters. Depending on the selected analysis the *fMRI analysis portal* will show different parameters that the user must configure.

When the task is created, the *BImageG* services will generate the Grid jobs and register them on to the *Grid Infrastructure*. This might involve transferring files necessary for the jobs execution. When the jobs are registered, the user will be able to see the task created and it will be allowed to start or cancel it.

All the tasks that were created are accessible through the *fMRI Analysis Portal*. This allows the user to see task related information, namely its status.

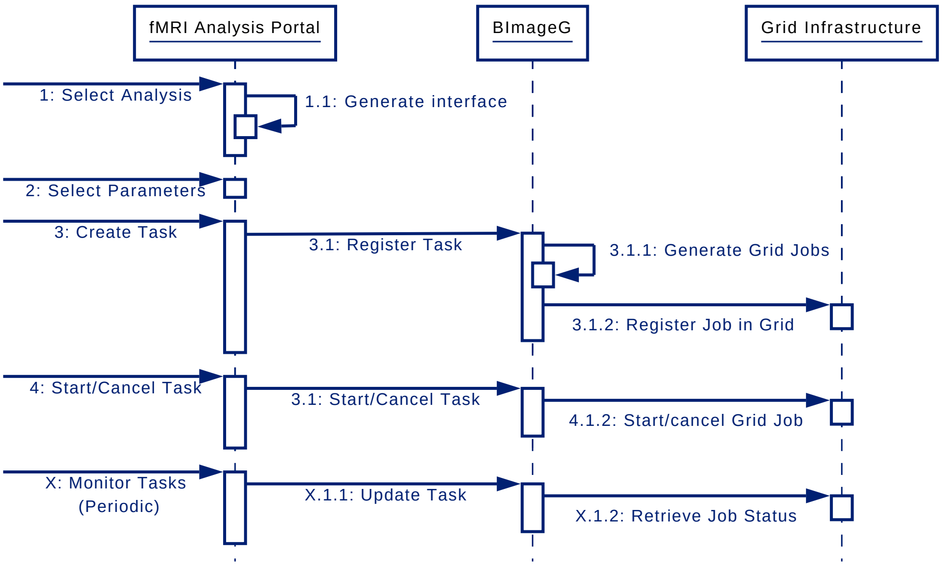


Figure 4.5: Integration between the different architecture layers while running a fMRI analysis

The *fMRI Analysis Portal* periodically enquires the *BImageG* services and updates all the information about the tasks that have been created.

4.4 Prototype Implementation

The development and implementation of this system was done in four main steps:

- Implementation of the site IEETA-PPS so that we would be able to integrate in the EGEE PPS testbed;
- fMRI analysis implementation using Grid resources;
- Development of the service responsible for the management and generation of the tasks – *BImageG (Task Manager)*;
- Development of a web interface – *fMRI Analysis Portal*.

Each of these steps is described in more detail in the following sections.

4.4.1 IEETA-PPS

Before we were able to use the EGEE PPS testbed services we had to create a local site named IEETA-PPS that had to provided at least the CE, SE and MON services.

The machines that compose the site IEETA-PPS are virtual machines. This has several advantages like better use of the resources available and the capacity to save and move the virtual machines to other computers. The platform that supported the virtualization was Xen$^{\mathrm{TM}}$ [76]. Our site is composed by five virtual machines. The main services, the CE and SE, require more resources and so were installed in dedicated virtual machines. The other services like the WNs, UI and MON require less resources shared virtual machines.

IEETA-PPS was installed in two main server, that we will call *svr1* and *srv2*. Each one of these servers had Linux (CentOS and Ubuntu) and Xen installed. On top of these two servers we had four virtual machines (*VMx*) running the services describe above (see figure 4.6).

Besides the machines described we had a fifth one that was also a virtual machine, but had only installed a UI and was used primarily for development.

4.4.2 fMRI Analysis Components

The fMRI analysis was developed in two steps. First we developed a version that uses the resources available locally. The main objective was to have a program that would allow us to

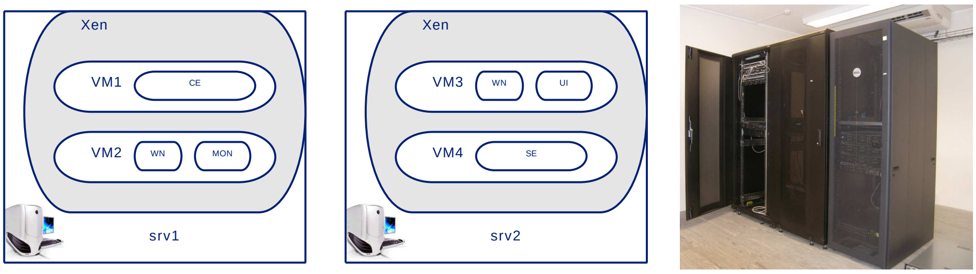


Figure 4.6: IEETA-PPS site (right) and machine organisation (left)

test it without having to worry about Grid specific issues and so ease the development. Then we developed a new version that permits the use of files stored in the Grid infrastructure, and so that it is able to run in the Grid infrastructure. Next we describe in more detail the modules implemented.

4.4.2.1 fMRI Analysis Module

The association analysis was implemented using two different association values that were calculated between two time-series. The first to be implemented was the r^2 (linear correlation) and the second was the V (see section 3.2).

We used a C library named *fsliolib*[3], that is available to the public [78], to run simple I/O operations in the fMRI data sets. This library allows us to read/write files in the NIfTI-1 and ANALYZE$^{\text{TM}}$7.5 format and run simple operations like extract and save time series and volumes. It had to be altered so that we would be able to, for example, catch error messages. Another problem was that it used environment variables for some configurations like file output format. To solve this problem some functions were added whose purpose was to set these values.

The different analysis and some auxiliary functions, were them implemented as methods of class template named *CImageOps*, allowing this way the analysis of images with different data types as described in figure 4.7.

The main class *CImage* uses *CImageOps* to analyse the images according to their type. This class also wrapped some of *fsliolib* functions in their methods to group specific tasks to simplify its use and ease future developments.

Beside these class, two more classes were created, one to treat errors, *CError*, and other

---

[3] Input and output routines for images in the FSL package [77]

that enables the creation of a log, *CLog*, with some useful information, such as the time of execution and the command run.

A program, based in the previous classes, was created for each type of analysis.

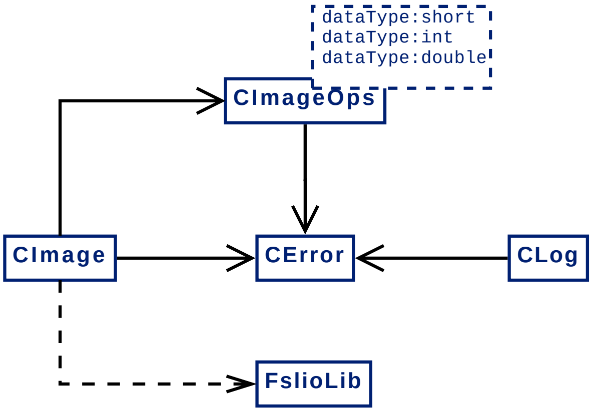


Figure 4.7: Relation between classes that compose the fMRI analysis module

The user has only to choose the target data that was going to be analysed, one voxel (figure 4.8) and a region defined by a cube with which it is calculated the target association analysis. To chose the region that is going to be analysed the user simply has to chose a coordinate, that corresponds to the corner of the cube that limits the region (in red on figure 4.8), and the cubes side. This method is standard for both analysis methods. This analysis returns a volume where each voxel corresponds to the association measure between the voxel in that same position and the initially chosen voxel as shown in figure 4.8. The resulting volume will have the same dimension as the initially selected cube.

4.4.2.2 fMRI Analysis Grid Module

The jobs that are submitted to the Grid infrastructure run as batch jobs. Since they are in a Grid environment they need to access Grid resources (data to be analysed) that are in remote repositories. To access the data we use a library, named *GFAL (Grid File Access Library)* that provides us the access to files in a Grid infrastructure using several protocols like *dcap* or *rfio*.

The main difference of running the analysis locally or using a Grid infrastructure is the location of the data. The strategy used was to make a local copy of the remote files in the machine where the analysis is being run, process them and then copy the results back to a Grid storage element (see figure 4.9).

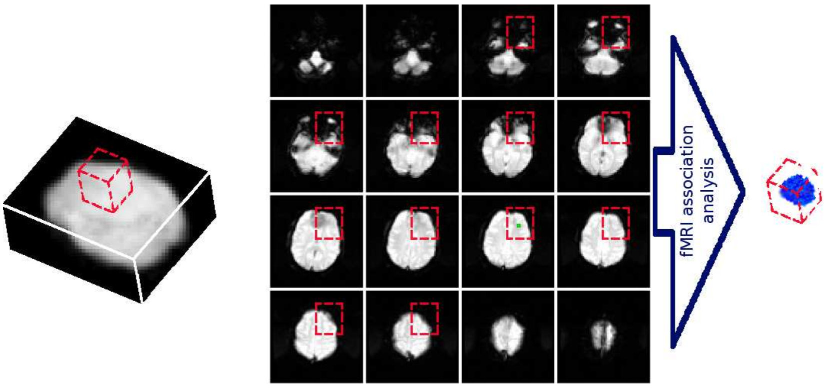


Figure 4.8: Selection of the region of interest (red), voxel to be analysed (green) and the resulting volume (on the right)

This was achieved by developing a new class named *CImageG* that in turn uses the *CImage* class (see section 4.4.2.1) providing similar methods. The only difference is that it uses files in gLite SE, instead of local files.

One program for each type of analysis, based in the *CImageG* class.

4.4.3 Task Management Service

This *BImageG* service is responsible for managing the execution of tasks in the Grid infrastructure. For this it has to provide the following capabilities:

- Generation of Grid jobs associated with each task;
- Run or cancel a specific task;
- Continuously monitor task progression.

It is important to be aware that before being able to manage the execution and monitoring of a task, the following conditions must be fulfilled[4]:

- Have a valid proxy file (generated from the user certificate);
- The data that is going to be analysed must already be available in the Grid infrastructure.

---

[4] We assume these conditions have been meet.

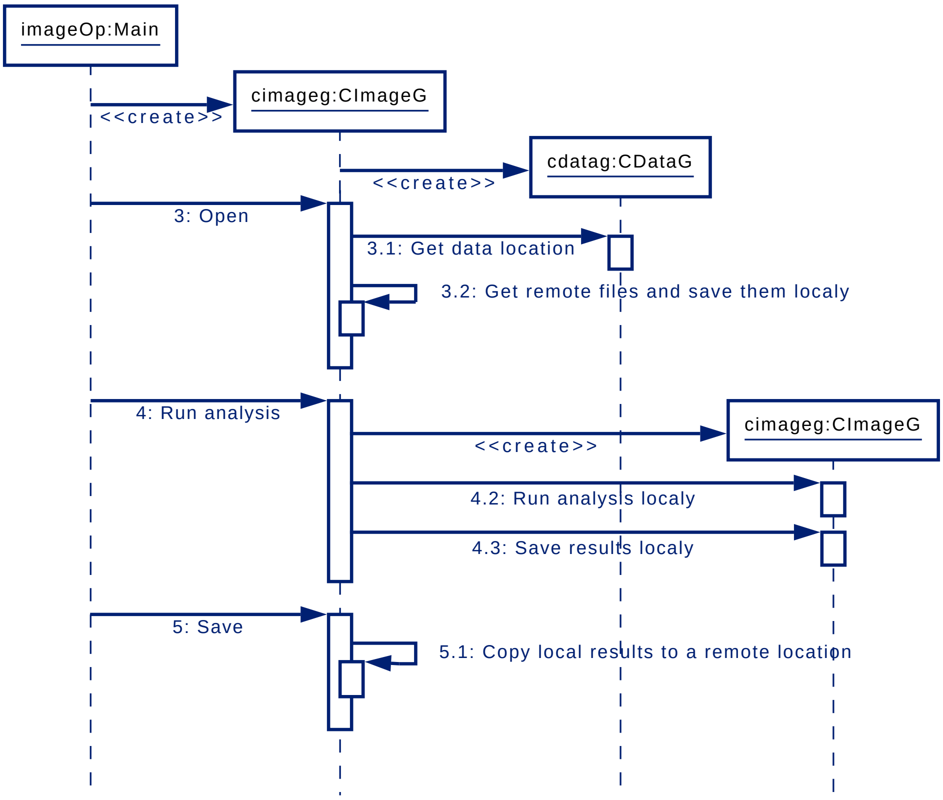
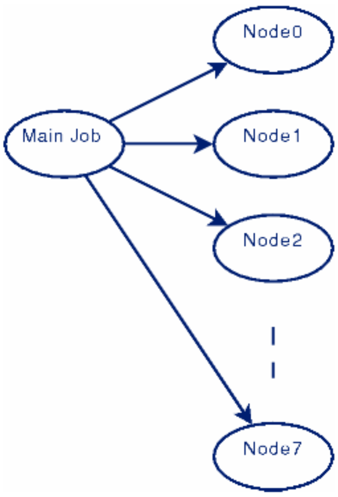


Figure 4.9: Image processing in the Grid using *CImageG*

When a user submits a specific task, like the association analysis of a specific ROI of a fMRI volume, the *Task Manager* will interpret the arguments and then will divide the ROI into smaller volumes. For each one of these, a Grid job is generated (see figure 4.10). The division of the tasks is done based in a pre-defined parameter, namely the maximum size of the volume that is going to be analysed.

From the several types of jobs available in the JDL definition, we choose the DAG job type. The DAG is characterized by allowing a user to create a single job composed by sub jobs, known as nodes, that might have or not dependencies among each other. It is also possible to define an output of a node as the input of another [39]. In our particular case, the DAG nodes generated do not have any dependencies among each other (see figure 4.10). However the use of this type of job will allow us in the future to implement more complex analysis frameworks. Another advantage of using this type of job is that the management of the nodes execution is the Grid middleware responsibility.



```
[
 Type="dag";
 Nodes=[
  Node0=[
   Description=[
    Type="job";
    InputSandbox="voxcorrg";
    Executable="voxcorrg";
    OutputSandbox={"std.err","std.out",".BrokerInfo","last.log"};
    Arguments="  /grid/dteam/randrade/HS_seq_mcfnii.gz 8 13 10  8  13 14
11 /grid/dteam/randrade/Node0Task 1190037185983nii.gz"
   ];
  ];
  Node1=[
   Description=[
    Type="job";
    InputSandbox="voxcorrg";
    Executable="voxcorrg";
    OutputSandbox={"std.err","std.out",".BrokerInfo","last.log"};
    Arguments="  /grid/dteam/randrade/HS_seq_mcfnii.gz 8 13 18  8  13 14
11 /grid/dteam/randrade/Node1Task 1190037185983nii.gz"
   ];
  ];
  (...)
  Node7=[
   Description=[
    Type="job";
    InputSandbox="voxcorrg";
    Executable="voxcorrg";
    OutputSandbox={"std.err","std.out",".BrokerInfo","last.log"};
    Arguments="  /grid/dteam/randrade/HS_seq_mcfnii.gz 16 21 18  8  13 14
11 /grid/dteam/randrade/Node7Task 1190037185983nii.gz"
   ];
  ];
 ];
 Dependencies={};
 Requirements=other.GlueCEInfoTotalCPUs > 2;
 Rank=other.GlueCEInfoTotalCPUs;
]
```

Figure 4.10: Example of a Grid DAG job graph and the corresponding JDL

The *Task Manager* is organised by its two main functions: task description and management. Figure 4.11 shows the relation between the *Task Manager* main classes and their functions.

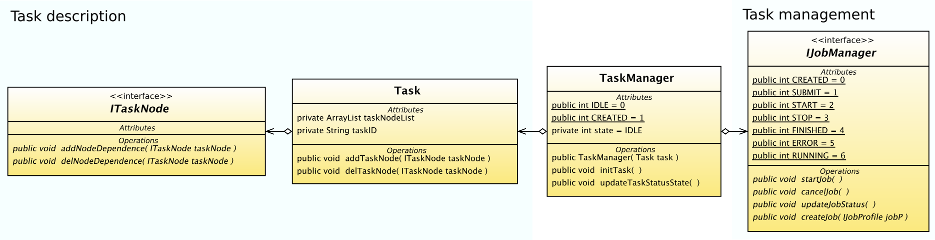


Figure 4.11: Organisation and relation between the main classes of the *TaskManager* and its function

The *task description* allows the definition of relevant attributes for each task. This description is independent from the middleware layer. The main attributes of a task are the analysis dependent parameters, files needed and the files that have to be retrieved. The files referred do not include the data files because these are assumed to be already stored in the Grid infrastructure but the analysis program and log files. The description of a task is done using two classes: *Task* and *TaskNode*. The base structure of a task is a main task that is composed by a list of task nodes where it is possible to define dependencies among each other, data (input/output), the analysis program and arguments. This structure is mapped directly into the classes *Task* and *TaskNode* (see figure4.11). The analysis dependent parameters, such as the ROI to be analysed are passed as arguments to the analysis program (see section 4.4.2). These parameters are be defined in the attribute *arguments* of each task node. This means that the only difference between the DAG nodes description is the *arguments* attribute, more precisely the region of the image that is analysed.

The *task management* provides the methods necessary to generate the JDL, submit and monitor the task and corresponding Grid jobs evolution. This is done through the use of the *IJobManager* and the *IJobProfile*, to which each created task is associated. These provide interfaces to other classes that are dependent of the middleware used. In our case these are the *gliteJobProfile* and the *gliteJobManager* classes. The last class uses the gLite APIs for submitting and monitoring the generated jobs. The decision to separate middleware specific classes provided us with a framework that can be more easily integrated with other middleware stacks besides gLite.

The methods available in (*IJobManager*) provide the following capabilities relative to rid job management:

- Generate JDL;
- Submit and start job execution;
- Cancel job execution;

The generation of the JDL is done by a class named *gliteJobProfile.* Although there are gLite APIs that provide some support [5] to JDL files support, they do not allow the correct generation of DAG type jobs. To overcome this problem *gliteJobProfile* generates its own DAG type JDLs, using only the gLite APIs for the description DAG nodes (See example in figure 4.10.

gliteJobProfile constructor accepts as a argument *ITaskNode* or *Task.* A user to generate a JDL, only needs to create a *gliteJobProfile* object with the target *Task* object as a argument and then retrieve the JDL using the correspondent *gliteJobProfile* method. This class stores the information of the target task within a structure similar to *Task* and the *TaskNode*s. Besides keeping the information relative to each job and respective nodes it also saves their

---

[5] The class used to generate the JDL file used with APIs to access the deprecated service *NetworkServer* that is being replaced by the *WMProxy.*

corresponding state and status. The difference among these, is that the state is related to the execution of the task inside the *TaskManager* while the status is relative to its execution in the Grid infrastructure (see table 4.1). This way only the status is dependent of the middleware being used.

We are using the new WMProxy service to access the gLite WMS to submit jobs. The access to this service is done through several gLite APIs namely *org.gLite.wms.wmproxy.WMProxyAPI*.

A simple job submission using the WMProxy implies several additional steps apart the ones already discussed:

- Create a client (*WMProxyAPI object*) that will provide us with the methods to access *WMProxy* service;
- Transfer the client proxy credentials to the WMS service. Each delegated credential is identified by a "delegation identifier" (e.g. myname). This delegation process allows the user to invoke services in the WMS using his credentials;
- Transfer the files needed to run the application (described in the *InputSandbox* JDLs attribute) to the *WMProxy* machine[6] and trigger the job submission.

The transfer of the files needed for the job submission to the *WMProxy*, imply also several steps:

- Register the job using a *String* describing the JDL and the "delegation identifier";
- Discover the transfer protocols supported by the *WMProxy* (e.g. *https* or *gsiftp*) and get the corresponding URI (Uniform Resource Identifier) to where the files will be transferred;
- Transfer the files using, for example, the *globus-url-copy* command (for each DAG node);
- Trigger the job submission.

The *gliteJobManager* provides also methods for monitoring the task execution. This is done through the use of system calls to gLites CLIs, namely the command "glite-wms-job-status". The status information is extracted through the use of regular expressions.

---

[6] In reality, this is only needed if the files needed are stored locally in the machine. If they are already in the Grid infrastructure they will be automatically transferred to the WN, so the user is able to submit the job directly to the WMS

During its creation and execution, each task can go through a pre-defined group of states. These are obviously related with the status of the Grid job and in this case with gLite. Table 4.1 shows the relation between the states of a task (see figure 2.11) and the status of the corresponding Grid job in gLite. The mapping between the state and status is not direct, although they are closely related.

Task states	Description	gLite job (DAG) states
IDLE	Task initial state	–
CREATED	Generation of Grid jobs (JDL files)	–
SUBMIT	Prepare the execution of the task remotely (copy files referred in *InputSB*)	SUBMITTED
START	Start the Grid jobs execution	WAITING
RUNNING	Grid jobs are being executed	RUNNING
FINISHED	Grid jobs ended with success	DONE(ok), CLEARED
STOP	Grid jobs are cancelled because of the user intervention	DONE(cancelled)
ERROR	Grid jobs were not able to end correctly	DONE(failed), ABORTED

Table 4.1: Relation between the states of a task and the corresponding gLite job

Both status and state of each task or respective nodes are available through *gliteJobManager* and *gliteJobProfile* methods.

4.4.4 User Portal

The *fMRI Analysis Portal* was implemented using the GWT (Google Web Toolkit) development framework. This framework allows a development of AJAX (Asynchronous Javascript And XML) based web applications similar to java applications. First the user develops its application just like a java application using the GWT libraries and then the GWT compiler translates the java code to JavaScript and HTML (HyperText Markup Language). Allowing a easier and faster development of a web application.

This portal is composed by a client application that runs in a web browser and a server application that uses the services provided by the *BImageG* namely the *TaskManager*.

The browser application is composed by a panel where we are able to choose our target analysis that is dependent of the analysis chosen and another where it is possible to monitor

Elements	Description
bingProcess	Analysis description
InputArgData	Input data description
InputArgPoint	3D or 4D point. Contains three or four Point elements
InputArgInt	Integer value
InputArgString	String value
CubeVerticeArgPoint	3D or 4D point that define the vertice of the region (Cube) to be analysed. Contains three or four Point elements
CubeEdgeArgInt	Value of the side of the region (Cube) to be analysed
Point(X/Y/Z/V)	Value of a coordinate in a a 3D or 4D point

Table 4.2: XML Schema elements of the analysis description files

and manage the tasks created.

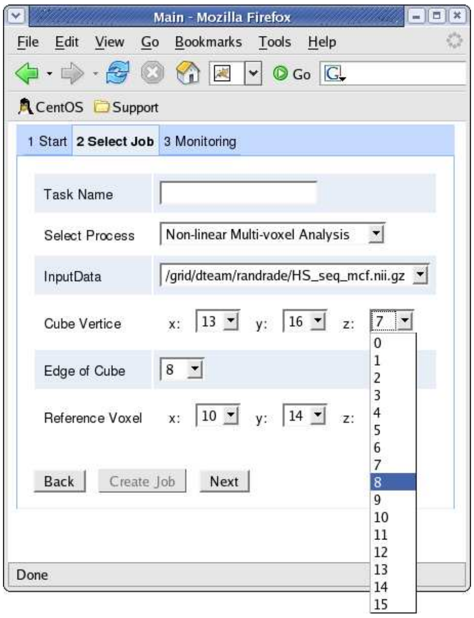
One of the functions of the client application is to monitor the tasks. This is done by periodically asking the server application for the current status of all the tasks. Two classes, the *TaskDesc* and the *TaskNodeDesc* are used to save the information of each task. It was not possible to use the *Task* and *TaskNode* classes described in the *Task Manager* despite their similarity, because the GWT has some restrictions to the type of objects used in each class.

Another functionality of the client application is a dynamic interface that is built using the information present in XML file. In this file, we describe each type of analysis, namely the parameters needed and their type. To implement this, we had to first define what types of objects are allowed (see table 4.2) and then created a XML Schema file to which all XML analysis description files comply.

We have defined seven elements (see table 4.2) that have several attributes namely the description, type and some element specific attributes. Using this file it is also possible to describe the values allowed for each element 4.12.

4.5 Running a fMRI analysis in EGEE PPS testbed

The *Grid Infrastructure* allows a user to access its resources without having to know what machines are currently being used, where the data is stored or where the analysis is being run.



Task Name

Select Process: Non-linear Multi-voxel Analysis

InputData: /grid/dteam/randrade/HS_seq_mcf.nii.gz

Cube Vertice x: 13 y: 16 z: 7

Edge of Cube: 8

Reference Voxel x: 10 y: 14 z:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bingProcess xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="/opt/analysis/bingProcess.xsd"
 processFile="/opt/analysis/voxcorrg" outputDataNumber="1" inputDataNumber="1"
 processDescription="Non-linear Multi-voxel Analysis">
        <InputArgData processInp="" desc="InputData"/>
        <CubeVerticeArgPoint processInp="" desc="Cube Vertice">
                <PointX  offSet="8" start="0" end="63" threshold="1"></PointX>
                <PointY  offSet="8" start="0" end="63" threshold="1"></PointY>
                <PointZ  offSet="8" start="0" end="15" threshold="1"></PointZ>
        </CubeVerticeArgPoint>
        <CubeEdgeArgInt processInp="" desc="Edge of Cube" start="1" end="16"
         threshold="-1"></CubeEdgeArgInt>
        <InputArgPoint processInp="" desc="Reference Voxel">
                <PointX start="0" end="63" threshold="1"></PointX>
                <PointY start="0" end="63" threshold="1"></PointY>
                <PointZ start="0" end="15" threshold="1"></PointZ>
        </InputArgPoint>
</bingProcess>
```

Figure 4.12: Example of a fMRI analysis interface generated and corresponding XML file

This is a big advantage from the user point of view, however it is important for a developer to be aware of the services and machines that are being used.

For this work we used the EGEE PPS testbed during its development and testing that although not being a dedicated infrastructure it allowed us to have a real distributed environment and to better understand the problems that are associated with Grid applications development.

When running a fMRI analysis we generate a Grid job that consists of several sub jobs that will run in the WNs that make part of the several CEs available. The VO used to run this analysis was the swetest VO that is available to the South West Federation Infrastructure of the EGEE project to test the provided functionality and its availability. Before being able to use this VO the user had to first register using the VOMS interface.

Each VO can have preferred services, namely the LFC, WMS and BDII service. In the case of the swetest VO we used the ones presented in figure 4.13[7].

The fMRI Analysis portal server was installed in the IEETA-PPS UI. The fMRI data that was going to be analysed and corresponding analysis results were also stored in the *PPS-LIP*

---

[7] This map was retrieve from the `http://egee-pre-production-service.web.cern.ch` in 06 of October of 2007. The services described in this map correspond only to some of the services that can be used for the *swetest* VO

Figure 4.13: Map of Grid services used when running the fMRI analysis with the *swetest* VO

SE. When a task is created, the generated Grid jobs are registered in the WMS present at the *PPS-IFIC* site. This registration includes the transfer of the files described in the *InputSB* between the local UI and the WMS.

When the user starts the job execution, the WMS service checks for available resources and submits the DAG job to a specific queue in a available CE , that in this case was at *PPP-IFIC*. When WNs became available, the CE starts sending the DAG sub jobs and starts their execution. In the WN each job will then retrieve the data from the SE at *PPS-LIP* and store it in the WN. In the end each sub job sends its results back to the SE and finishes its execution. However before being able to transfer the data they need to contact the LFC server and retrieve the location of the data, because it is identified by their LFN.

4.6 Results

We have implemented a framework that allows a user to select a type of analysis and then run it using the resources available in the Grid infrastructure.

The *Multi voxel fMRI analysis* was implemented using the r^2 (linear correlation) and the V association measures and was applied to a fMRI sequence of an epileptic patient.

The patient was selected from a population of non-lesional epilepsy patients that are possible candidates to epilepsy surgery. This population represents a difficult target because it has no visible clue on the epileptogenic area. However in a context of a epilepsy surgery, the patients have multimodal characterization and therefore a good reference in inter-modality

Submitted ⇒ Running (WMS) (WN)	Running ⇒ Done (WN) (WN)	Done ⇒ Done (WN) (WMS)	Total
0h04m14s	0h14m30s	0h02m46s	0h21m30s

Table 4.3: Average times during the several steps of a job (DAG Node) execution.

comparisons. The patient is part of the Hospital Garcia da Orta joint collaboration with the ongoing Epilepsy surgery unit.

The patient was submitted to a multimodal protocol comprising:

- Video-EEG acquisition (64 channels acquisitions);
- T1 MRI 3D volumetric sequence for morphology modelling;
- EEG fMRI acquisition using an fMRI compatible EEG system with a cap of 21 electrodes from MAGLink system (Compumedics,El Paso ,USA).

The Video-EEG was used to access the typical epileptiform manifestations of the patient and used as reference in the EEG-fMRI analysis.

The MRI studies where acquired at at the MRI centre Ressonância magnética de Caselas (RMC) (`www.rm-sdi.pt`), equipped with a GE 1,5 Tesla, Signa Infinity MRI machine.

The EEG-fMRI protocol consisted of consecutive continuous EEG-fMRI sequences of 5 minutes which resulted in all-brain 100 volumes. Each all brain volume consisted 16 64×64 slices acquired from top to bottom of the patient's head every 3 sec.

These analysis were run in a workstation (Intel Xeon Dual Processor with 4 MB cache at 2 GHz) and in the Grid framework prototype we implemented.

Using the infrastructure we were able to obtain average values of each node execution time and the overhead introduced by the Grid middleware. We have also gathered values of the *Multi-voxel fMRI association analysis* execution time in the proposed framework, in a dedicated workstation and using the two association measures implemented (V and r^2).

Taking into account that we are using a shared testbed used mainly for testing, it is important to be aware that the values obtained are only indicative.

The execution times (see table 4.3) are extremely variable because in a Grid infrastructure and specially in the PPS testbed we have a wide range of machines serving as WN.

Submitted ⇒ Running (WMS) (WMS)	Running ⇒ Done (WMS) (WMS)	Total
0h01m06s	0h46m39s	0h47m45s

Table 4.4: Duration of a *Multi-voxel fMRI Association Analysis* relatively to one voxel (r^2) using the Grid infrastructure.

Multi-voxel fMRI Association Analysis	
V	r^2
3h23m18s	3h23m23s

Table 4.5: Duration of a *Multi-voxel fMRI Association Analysis* relatively to one voxel with all the other, using V and r^2. This was run in a workstation equipped with a Intel Xeon Dual Processor with 4 MB cache at 2 GHz.

The run in the workstation (sequential) took approximately 3 hours and 20 minutes (see table 4.5). Using the proposed framework we run the same analysis dividing the raw data into 16 different parts. It took approximately 47 minutes (see table 4.4 and figure 4.14). Ideally and considering the times measured (see table 4.3) a analysis should had taken approximately the same time as a node job execution (approx. 22 minutes). However this would only happen if all the jobs started to run at the same time. In our testbed, this was not possible due to the number of available WN. Nevertheless the times measure still imply a 5 fold decrease.

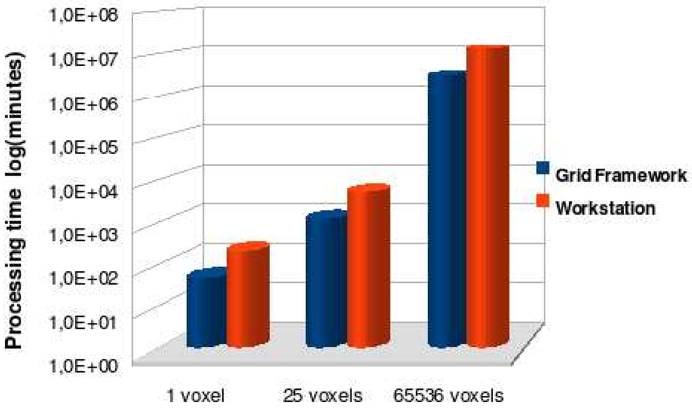


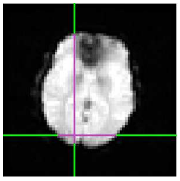
Figure 4.14: Comparison of the *Multi-voxel fMRI association analysis* execution times for 1, 25 and all ($64 \times 64 \times 16 = 65536$) voxels with all the others in a typical fMRI volume.

It is important to notice the overhead introduced by the middleware, that in our case was in average 7 minutes. This time corresponds to the scheduling process in the WMS and the time it takes for the WMS to acknowledge that the job has finished its execution in the WN.

Besides the times already described it is also important to be aware of the overhead associated with the job submission by the developed framework. This process implies the transference of files that in most cases can be disregarded, however when the number of job is very large (superior to 30) the submission time overhead can be of several minutes. In our specific case we used normally 16 jobs and had a overhead of about 1 minute.

Input

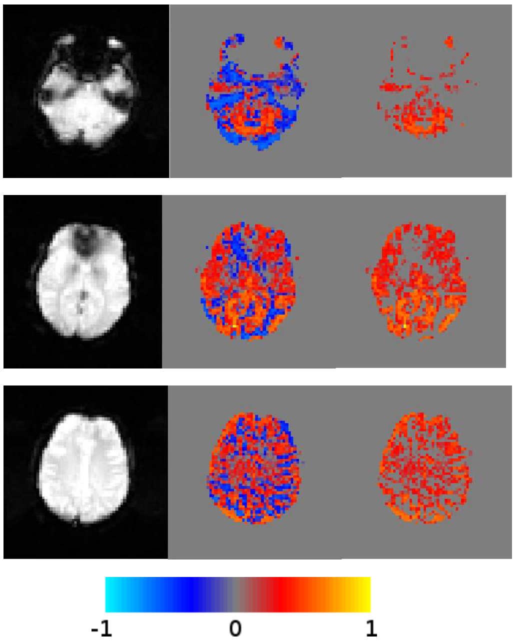
a)Brain area represented by a particular voxel.



Slice #7, voxel 26, 15.

Results

b)Association results generated by developed *Grid framework*.



Slice #4

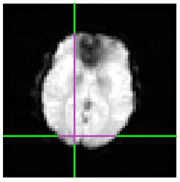
Slice #7

Slice #10

Figure 4.15: fMRI analysis results using r^2 (linear correlation) as a association measure: the correlation between brain areas, represented as association maps (b), and a specific brain areas represented in the example by a localized voxel in the target volume (a). In (b) warm colors represent positive linear correlation while blue tones negative

Input

a)Brain area represented by a particular voxel.



Slice #7, voxel 26, 15.

Results

b)Association results generated by developed *Grid framework*.

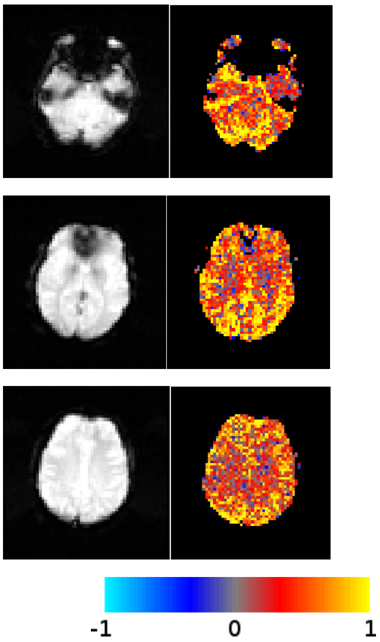


Figure 4.16: fMRI analysis results using V as a association measure: the correlation between brain areas, represented as association maps (b), and a specific brain areas represented in the example by a localized voxel in the target volume (a). In (b) warm colors represent higher levels of association

The values presented in table 4.4 and 4.5 correspond to a *Multi-voxel fMRI Association Analysis* relatively to only one voxel with all the other. To do a complete analysis we had to do the same analysis for all the voxels in a fMRI volume that would correspond to 65536 $(64 \times 64 \times 16)$. Doing such a analysis in a workstation would take approximately 25 years and 6 years while using our current Grid framework set we estimated it would take approximately 5 times less (see figure 4.14). Although we are able to reduce substantially the computational time, it is still too large to be useful in a clinical environment. Nevertheless, neuroscientists are interested in studying specific ROIs they know have clinical significance. Thus a typical analysis will mean selecting one or two dozens of voxels to analyze. In this case it is possible reduce the execution time to less than one day (see figure 4.14) making it acceptable for use in a clinical environment.

When we run the *Multi-voxel fMRI association analysis* using the two association measures, V and r^2, we would expect a better performance by V as shown in Cunha et al. [67], however the execution time was more or less the same (see table 4.5) [67]. The reason for this result can be associated with the efficiency of the implemented code, or even with the characteristics of the signal analysed. In this case we are analysing relatively small time series (100 samples) compared to the ones presented in Cunha et al. [67] (512 samples).

Regarding the fMRI image analysis results we were able to obtain promising results. We did several runs using the two implemented association measures (see figure 4.15 and 4.16). These were relative to a specific voxel (a). The runs resulted in two 3D volumes, one for each analysis type. We present three axial views (b) of the resulting volume where the warm colours correspond to higher correlation levels and the cooler to lower ones. Clinicians can obtain brain areas highly associated to the selected ROI by analysing these results, namely by observing the dispersion of the colours present in each image.

V^2 produces 3D association maps more closer to the selected ROI then r^2. Neuroanatomically, it is plausible that closer areas belonging to the same lobe are more tightly connected than far cortex areas. This may explain the difference between the two association measures maps, denoting less tolerance to noise of r^2.

Before running the analysis we applied a simple filter based in a threshold value to reduce the noise around the image.

The clinical significance of the results obtained is out of the scope of this work.

Chapter 5

Discussion and Conclusion

The developed Grid framework allowed us to work with state of the art computational paradigms, provided by *Grid* technology. However, despite of all its advantages, it also brought some disadvantages. In our particular case we used the gLite middleware, that is considered one of the most advanced Grid middleware available. The fact that *gLite* is still under development complicated the progress of our framework. The first problem was the installation of the IEETA-PPS that was performed in a period where the middleware was suffering a lot of changes, namely in the services it supported. The consequence was that the installation of the middleware took more than one month. However, this happened also because we were installing a EGEE PPS testbed node, that uses the latest services, and so, not necessarily the most stable ones. This also affected directly the development of the framework. In one occasion, we had to migrate a big part of the *Task Manager* code because a service that it was using was deprecated. The installation of current version of the gLite middleware is currently much faster and easier. On the other hand, using the gLite PPS testbed, allowed us to work with a *Grid* infrastructure and understand some problems that would be very difficult to detect if we were using a locally installed *Grid*. An example is running jobs remotely. In our case we had little guarantees about the type of libraries that would be available or even the operating system we would find in each WN. The JDL file allowed us to submit jobs only in WNs that fulfill the requirements that we defined, however we do not have the guarantee that this information will be available or if it is correct. Another great advantage was the EGEE community that helped us to quickly understand the "basics" of the gLite middleware allowing us to manage our site with some autonomy. They also provided precious help with development of our framework, specially in the interaction with the gLite services.

The developed framework allows a user to easily submit jobs and monitor them. This provides the user the necessary tools to run a fMRI analysis. The results obtained showed that *Grid* reduces several times the duration needed to run a fMRI analysis and that these could be reduced even further if we had CE with more WNs so that all the sub jobs could be run at the same time. Despite the large overhead introduced by the *Grid* middleware this can be easily overcome with the time gained by running the jobs in parallel. Another important point is that the times were obtained using the PPS testbed, that normally contains older machines and so it becomes very difficult to compare to the tests that in our case were done in a relatively recent workstation. Nevertheless, the results obtained were very good. With these results we proved it is possible to run computationally intensive fMRI analysis in medical data and retrieve the results in a reasonable period of time. In the case we presented, of a epiletic patient that is admitted for decision of surgery within a three to five days period, the fMRI analysis must be available in less than 24 hours. The results we obtained showed that it is possible to use non-linear association analysis, for a particular ROI, for fMRI sequences in this target scenario.

During this work we tried to identify aspects that would be important for a future development of this framework. In the implementation, the most important aspects were the user authentication and data management. The first is a extremely sensitive point, but *Grid* already provides some tools that can be easily used. When a user submits a job to the *Grid* infrastructure, for example, all the actions associated with the job submission are associated to the user through his proxy that is generated from its certificate. This authentication method can and should be used, for example, to authenticate the user before he can access the fMRI analysis Portal. The data management is also a very important aspect. The *Grid* provides the tools necessary for the data management among distributed repositories and it is starting to support specialized services for medical data management. The user should be able to manage these resources. To conclude it is important to provide the following capabilities:

- User authentication;
- Data storage, retrieval;
- Associate metadata;
- Data processing (normalization, volume extraction, ...).

The job management must also be improved. The generation of jobs according to the available resources can reduce an analysis execution time by reducing the overhead introduced by the *Grid* middleware.

The association analysis for fMRI sequences provided some interesting results. Due to the novelty of these techniques further studies, must be performed to understand what is the best manner to analyse them and present the corresponding results to the user. This may imply further processing of the resulting data and the implementation of more complex frameworks.

The *Grid* already provides very good tools for security, data and job management, so it is important to be aware of the capabilities provided by the *Grid* middleware, before starting the development of frameworks that uses *Grid* infrastructures.

One point that was not discussed in this work, which is vital in a clinical environment, is the response time. When using distributed environment we must guarantee that certain “requests” must be given higher priorities. In a distributed environments, this is not an easy task. There are several projects that are already searching for the solution to this problem, but it is still a work in progress.

Despite all the technological solutions available, the biggest problem associated these type of framework is their implementation in the real world. Problems associated with data privacy must be considered of a vital importance if these frameworks are to be implemented in the future.

Bibliography

[1] R. Acharya, R. Wasserman, J. Stevens, and C. Hinojosa. Biomedical imaging modalities: a tutorial. *Comput Med Imaging Graph*, 19(1):3–25, 1995.

[2] EC Ciccarelli, AJ Jacobs, P. Berman, JC Di John, EA Schwaber, MJ Schwaber, M. Friedman, GW Friedland, GJ Wise, FD Moore, et al. Looking Back on the Millennium in Medicine. *New England Journal of Medicine*, 342(1):42–49, January 2000.

[3] Carlos Costa, Augusto Silva, and José Oliveira. *Advanced Computational Intelligence Paradigms in Healthcare-2*, chapter Current Perspectives on PACS and a Cardiology Case Study, pages 79–108. Springer Berlin / Heidelberg, 2007.

[4] Joseph D. Bronzino, editor. *The Biomedical Engineering Handbook*. 2000.

[5] Robert A. Geenes and James F. Brinkley. *Medical Informatics: Computer Applications in Health Care and Biomedicine*, chapter Imaging Systems, pages 485–538. Health Informatics. Springer, second edition, 2001.

[6] Fmrib center, department of clinical neurology, university of oxford. Available at `http://www.fmrib.ox.ac.uk/`, July 2007.

[7] Seong-Gi Kim and Peter A. Bandettini. *Functional MRI, Basic Principles and Clinical Applications*, chapter Principles of Functional MRI, pages 3–23. Springer, 2006.

[8] S.S. Kety and C.F. Schmidt. The effects of altered arterial tensions of carbon dioxide and oxygen on cerebral blood flow and cerebral oxygen consumption of normal young men. *J Clin Invest*, 27(484-492):21–22, 1948.

[9] S.S. Kety and C.F. Schmidt. The nitrous oxide method for the quantitative determination of cerebral blood flow in man: theory, procedure and normal values. *J Clin Invest*, 27(4):476–483, 1948.

[10] S. Ogawa, T.M. Lee, A.R. Kay, and D.W. Tank. Brain magnetic resonance imaging with contrast dependent on blood oxygenation. volume 87, pages 9868–9872, 1990.

[11] K. Uludag, D. J. Dubowitz, E. J. Yoder, K. Restom, T. T. Liu, and R. B. Buxton. Coupling of cerebral blood flow and oxygen consumption during physiological activation and deactivation measured with fmri. *Neuroimage*, 23(1):148–55., 2004.

[12] S. A. Rombouts, R. Goekoop, C. J. Stam, F. Barkhof, and P. Scheltens. Delayed rather than decreased bold response as a marker for early alzheimer's disease. *Neuroimage*, 26(4):1078–85, 2005.

[13] A. O. Ceballos-Baumann. Functional imaging in parkinson's disease: activation studies with pet, fmri and spect. *J Neurol*, 250 Suppl 1:I15–23, 2003.

[14] M. D'Esposito, L. Y. Deouell, and A. Gazzaley. Alterations in the bold fmri signal with ageing and disease: a challenge for neuroimaging. *Nat. Rev. Neurosci.*, 4(11):863–72, 2003.

[15] Dominik Weishupt, Victor D.Köchli, and Borut Marincek. *How does MRI work?* Springer, 2003.

[16] R. Turner and T. Jones. Techniques for imaging neuroscience. *Br. Med. Bull.*, 65:3–20., 2003.

[17] D. A. Handwerker, J. M. Ollinger, and M. D'Esposito. Variation of bold hemodynamic responses across subjects and brain regions and their effects on statistical analyses. *Neuroimage*, 21(4):1639–51., 2004.

[18] Geoffrey K. Aguirre. *Functional MRI, Basic Principles and Clinical Applications*, chapter Experimental Design and Data Analysis for fMRI, pages 58–74. Springer, 2006.

[19] K. M. Petersson, T. E. Nichols, J. B. Poline, and A. P. Holmes. Statistical limitations in functional neuroimaging. *Philos Trans R Soc Lond B Biol Sci*, 354(1387):1239–81, 1999.

[20] S.M. Smith, M. Jenkinson, M.W. Woolrich, C.F. Beckmann, TE Behrens, H. Johansen-Berg, P.R. Bannister, M. De Luca, I. Drobnjak, D.E. Flitney, et al. Advances in functional and structural MR image analysis and implementation as FSL. *Neuroimage*, 23(Suppl 1):S208–S219, 2004.

[21] Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the Grid: Enabling scalable virtual organization. *International Journal Supercomputer Applications*, 15(3):200–222, Fall 2001.

[22] Ian Foster, Carl Kesselman, and Steven Tuecke. *The Grid: Blueprint for a New Computing Infrastructure*, chapter The Open Grid Services Architecture, pages 215–257. Morgan Kaufmann, 2 edition, 2005.

[23] Ian Foster and Cark Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*, chapter Concepts and Architecture, pages 37–63. Morgan Kaufmann, 2 edition, 2005.

[24] R. Alfieri, R. Cecchini, V. Ciaschini, L. dell'Agnello, Á. Frohner, A. Gianoli, K. Lörentey, and F. Spataro. *Grid Computing*, chapter VOMS, an Authorization System for Virtual Organizations, pages 33–40. Springer Berlin / Heidelberg, 2004.

[25] Ian Foster and Cark Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*, chapter Computational Grids, pages 15–51. Morgan Kaufmann, 1 edition, 1999.

[26] C. Catlett. Standards for grid computing: Global grid forum. *Journal of Grid Computing*, 1(1):3–7, 2003.

[27] Accelerating the adoption of grid solutions in the enterprise. Available at `http://www.ogf.org/`, December 2004.

[28] Thilo Kielmann. Ogf and standards. Available in `http://isoc.nl/activ/2006-OGF-MerzkyKielmann-OGF.pdf`, 2006.

[29] Organization for the advancement of structured information standards. Available at `http://www.oasis-open.org/`, July 2007.

[30] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An open grid services architecture for distributed systems integration. *Open Grid Service Infrastructure WG, Global Grid Forum*, 2002.

[31] Ian Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, and J. Von Reich. The open grid services architecture, version 1.5. Available in `http://www.ogf.org/`, July 2006.

[32] Wolfgang Gentzsch. The german d-grid initiative in international context. Communication in First Iberian Grid Infrastructure Conference, 2007.

[33] Ruth Pordes, Bill Kramer, Miron Livny, Paul Avery, Kent Blackburn, Torre Wenaus, Frank Würthwein, Rob Gardner, Alan Blatecky, and Rob Quick. The open science grid. Available in `www.iop.org`.

[34] Virtual data toolkit. Available at `http://vdt.cs.wisc.edu/`, July 2007.

[35] The globus alliance. Available on `http://www.globus.org/`, July 2007.

[36] Ian Foster. Globus toolkit version 4: Software for service-oriented systems. *Journal of Computer Science and Technology*, 21(4):513–520, 2006.

[37] E. Laure, SM Fisher, A. Frohner, C. Grandi, P. Kunszt, A. Krenek, O. Mulmo, F. Pacini, F. Prelz, J. White, et al. Programming the grid with glite. *Computational Methods in Science and Technology*, 12(1):33–45, 2006.

[38] Stephen Burke, Simone Campana, Antonio Delgado Peris, Flavia Donno, Patricia Méndez Lorenzo, Roberto Santinelli, and Andrea Sciabà. *gLite 3 User Guide*, January 2007. Available in `http://glite.web.cern.ch`.

[39] EGEE-Project. Job description language (jdl) attributes specification. Available at `https://edms.cern.ch/document/590869/1/`, 2006.

[40] Fabrizio Pacini. *EGEE User's Guide - WMS Service*, May 2006. Available in *https://edms.cern.ch/document/572489/*.

[41] A. křenek. *EGEE User's Guide - Service Logging and Bookkeeping (L&B)*, December 2005. Available in *https://edms.cern.ch/document/571273/*.

[42] S. Andreozzi, S. Burke, L. Field, S. Fisher, B. Konya, M. Mambelli, JM Schopf, M. Viljoen, and A. Wilson. GLUE Schema Specification-Version 1.2. Technical report, 2005.

[43] Arda metadata catalogue project. Available at `http://amga.web.cern.ch/amga/`, July 2007.

[44] E-infrastructure shared between europe and latin america. Available at `http://www.eu-eela.org/`, July 2007.

[45] Grid enabled remote instrumentation with distributed control and computation. Available at `http://www.gridcc.org/`, July 2007.

[46] Interactive european grid project. Available at `http://www.interactive-grid.eu/`, July 2007.

[47] Accessgrid. Available at `http://www.accessgrid.org/`, July 2007.

[48] A digital library infrastructure on grid enabled technology. Available at `http://www.diligentproject.org/`, July 2007.

[49] Project presentation for omii-europe. Available at `http://omii-europe.org/`, August 2006.

[50] e-health - making healthcare better for european citizens: An action plan for european e-health area. Available at `http://ec.europa.eu/information_society/`. COM(2004) 356, European Commission.

[51] V. Breton, I. Blanquer, V. Hernandez, Y. Legré, and T. Solomonides. Roadmap for a european healthgrids. In *Proceedings HealthGrid*. IOS Press, 2007.

[52] V. Breton, I. Blanquer, V. Hernandez, Y. Legré, and T. Solomonides. Proposing a roadmap for healthgrids. In *Proceedings HealthGrid*, pages 319–329. IOS Press, 2006.

[53] M. Olive, H. Rahmouni, T. Solomonides, V. Breton, Y. Legré, I. Blanquer, V. Hernandez, I. Andoulsi, J. Herveg, and P. Wilson. Share roadmap 1: Towards a debate. In *Proceedings of HealthGrid*, Geneva, Switzerland, April 2007.

[54] Martin Hoffman. From grid applications to integrated knowledge grids - current state of healthgrids and perspectives for the future. HealthGrid key note, Valencia, 2006.

[55] Healthgrid. Available at `http://www.healthgrid.org/`, July 2007.

[56] Embrace network of excellence. Available at `http://www.embracegrid.info/`, July 2007.

[57] Wisdom – initiative for grid-enabled drug discovery against neglected and emergent diseases. Available at `http://wisdom.eu-egee.fr/`, July 2007.

[58] J. Montagnat, D. Jouvenot, C. Pera, A. Frohner, P. Kunszt, B. Koblitz, N. Santos, and C. Loomis. Bridging clinical information systems and grid middleware: a medical data manager. *Studies in health technology and informatics*, 120:14–24, 2006.

[59] J. Montagnat, F. Bellet, H. Benoit-Cattin, V. Breton, L. Brunie, H. Duque, Y. Legré, I. E. Magnin, L. Maigne, S. Miguet, J. M. Pierson, L. Seitz, and T. Tweed. Medical images simulation, storage, and processing on the european datagrid testbed. *Journal of Grid Computing*, 2:387, 2004.

[60] V. Breton, K. Dean, T. Solomonides, I. Blanquer, V. Hernandez, E. Medico, N. Maglaveras, S. Benkner, G. Lonsdale, S. Lloyd, K. Hassan, R. McClatchey, S. Miguet, J. Montagnat, X. Pennec, W. De Neve, C. De Wagter, G. Heeren, L. Maigne, K. Nozaki, M. Taillet, H. Bilofsky, R. Ziegler, M. Hoffman, C. Jones, M. Cannataro, P. Veltri, G. Aloisio, S. Fiore, M. Mirto, I. Chouvarda, V. Koutkias, A. Malousi, V. Lopez, I. Oliveira, J. P. Sanchez, F. Martin-Sanchez, G. De Moor, B. Claerhout, and J. A. Herveg. The healthgrid white paper. *Studies in Health Technology and Informatics*, 112:249–321, 2005.

[61] Shu-Hui Hung, Tsung-Chieh Hung, and Jer-Nan Juang. Sars grid-an ag-based disease management and collaborative platform, 2006.

[62] cancer biomedical informatics grid. Available at `http://cabig.cancer.gov/`, July 2007.

[63] Share. D 6.1 integrated road map 1. Available in `http://www.eu-share.org/`, February 2007.

[64] Share. D 4.2 bottlenecks & challenges and rtd responses for legal, ethical, social and economic aspects of healthgrids - roadmap 1. Available in `http://www.eu-share.org/`, June 2007.

[65] K. J. Friston, P. Fletcher, O. Josephs, A. Holmes, M. D. Rugg, and R. Turner. Event-related fmri: characterizing differential responses. *Neuroimage*, 7(1):30–40., 1998.

[66] K. M. Petersson, T. E. Nichols, J. B. Poline, and A. P. Holmes. Statistical limitations in functional neuroimaging. ii. signal detection and statistical inference. *Philos Trans R Soc Lond B Biol Sci*, 354(1387):1261–81, 1999.

[67] J. P. Cunha and P. G. de Oliveira. A new and fast nonlinear method for association analysis of biosignals. *IEEE Trans. Biomed. Eng.*, 47(6):757–63, 2000.

[68] João Paulo Silva Cunha, P. Guedes de Oliveira, J. Ramalheira, J. Lopes, and A. Martins da Silva. Is background eeg analysis relevant on predicting epilepsy evolution? In *8th European Congress of Clinical Neurophysiology*, Munich, Germany, 1996.

[69] A. Martins da Silva and João Paulo Silva Cunha. Scalp eeg recordings: Inter ictal / ictal location and spreading of epileptiform events. *Acta Neurol Scand*, suppl 152:17–19, 1994.

[70] Charles M. Grinstead and J. Laurie Snell. Introduction to probability. Available in `http://www.dartmouth.edu/~chance/teaching_aids/books_articles/`.

[71] R. Andrade, I. Oliveira, J.M. Fernandes, and J.P.S. Cunha. A grid framework for non-linear brain fmri analysis. *Studies in Health Technology and Informatics*, 126, 2007.

[72] R. Andrade, I. Oliveira, J.M. Fernandes, and J.P.S. Cunha. Multi-voxel non-linear fmri analysis: A grid computing approach. *IberGRID*, pages 365–373, 2007.

[73] S. Roberto Amendolia, Jose Galvez, David Manset, Wassem Hassan, Florida Estrella, Chiara Del Frate, Mohammed Odeh, Tamas Hauer, and Richard McClatchey. Deployment of a grid-based medical imaging application. *Studies in Health Technology and Informatics*, 112(2005):59–69, 2005.

[74] Digital imaging and communications in medicine. Available at `http://medical.nema.org/`, 1996.

[75] S. G. Erberich, M. Bhandekar, A. Chervenak, M. D. Nelson, and C. Kesselman. Dicom grid interface service for clinical and research pacs: a globus toolkit web service for medical data grids. *Int. Journal of Computer Assisted Radiology and Surgery*, 1(87-105):p100–102, 2006.

[76] The xen$^{\mathrm{TM}}$virtual machine monitor. Available at `http://www.cl.cam.ac.uk/research/srg/netos/xen/`.

[77] Fmrib software library. Available at `http://omii-europe.org/`, August 2006.

[78] Fsliolib. Available at `http://nifti.nimh.nih.gov/`, July 2007.

Glossary

AccessGrid is an ensemble of resources including multimedia large-format displays, presentation and interactive environments, and interfaces to Grid middleware and to visualization environments. These resources are used to support group-to-group interactions across the Grid.

Condor project develops, imploys and evaluates policies and mechanisms to enable High Throughput Computing. The Condor is a specialized workload management system for computer-intensive jobs. It provides a job queueing mechanism, scheduling policy, priority sheme, resource monitoring and resource management.

D-Grid is the German National Grid Initiative (see `http://www.d-grid.de/`).

DataGrid is a project funded by European Union whose objective was to build the next generation computing infrastructure providing intensive computation and analysis of shared large-scale databases (see `http://eu-datagrid.web.cern.ch/`).

DataTag Project that aims the creation of a large-scale intercontinental grid testbed between the European Community and the USA, focusing mainly in the interoperability issues (see `http://datatag.web.cern.ch/datatag/`).

dCache is a system for storing and retrieving huge amounts of data, distributed among a large number of heterogenous server nodes, under a single virtual filesystem tree with a variety of standard access methods.

e-Health is the term given to modern information and communication technologies that support healthcare services and pratice.

EGEE Project aims the creation of a seamless Grid infrastructure for e-Science. (see `http://www.eu-egee.org/`).

Embrace is a project that aims to integrate the major databases and software tools in bioinformatics, using existing methods and emerging Grid service technologies (see `http://www.embracegrid.info/`).

gLite is the codename of the middleware that is currently being developed in the EGEE project (see `http://glite.web.cern.ch/glite/`).

GridFTP is a high-performance, secure, reliable data transfer protocol optimized for high-bandwidth networks.

GridSphere is a standards based portlet framework for building web portals. It also provides a set of portlet web applications that work seemlessly with the GridSphere framework to provide a complete Grid portal development solution.

GriPhyN (Grid Physics Networks) project develops Grid technologies for scientific projects that analyse or collect petabyte-scale datasets (see `http://www.griphyn.org/`).

GSI is a Grid Security Infrastructure based in the public key cryptography. It allows, among other things, the use of certificates to authenticate users and services (see `http://www.globus.org/security/`).

HealthGrid initiative is a community composed by specialists from diferent areas that aim to create a fully operational European/International HealthGrid.

IEETA-PPS Name of the EGEE PPS testbeb site at IEETA

Int.eu.Grid project aim is to deploy and operate a e-Infrastructure for demanding interactive applications.

LIP (Laboratrio de Instrumentao e Fsica Experimental de Partculas) is a scientific association that does research in the High Energy Physics field.

MyProxy is a credential management service. It manages X.509 Public Key Infrastructure service credentials and combines a online credential repository and a online certificate authority.

NAREGI is the Japonese National Research Grid Iniciative.

Shibboleth is standards-based, open source middleware software which provides Web Single SignOn (SSO) across or within organizational boundaries. It allows sites to make informed authorization decisions for individual access of protected online resources in a privacy-preserving manner (see `http://shibboleth.internet2.edu/`).

TeraGrid is a Grid infrastructure that connects the United States of America National Science Foundation Supercomputer Centers (see `http://www.teragrid.org/`).