Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática
2009

Micael Cardoso Gonçalves Pedrosa

Um portal para rede de neuroimagem portuguesa

A web portal for Portuguese Brain Imaging Network

Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática
2009

Micael Cardoso Gonçalves Pedrosa

Um portal para rede de neuroimagem portuguesa

A web portal for Portuguese Brain Imaging Network

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática (M.I.E.C.T.), realizada sob a orientação científica do Professor Doutor José Maria Amaral Fernandes, Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

Apoio financeiro da FCT e do FSE no âmbito do III Quadro Comunitário de Apoio.

FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E ENSINO SUPERIOR

UNIÃO EUROPEIA
Fundo Social Europeu

Dedico este trabalho à minha família.

o júri / the jury

presidente / president

Joaquim Arnaldo Martins
Professor Catedrático da Universidade de Aveiro

vogais / examiners committee

António Miguel Pontes Pimenta Monteiro
Professor Auxiliar da Faculdade de Engenharia da Universidade do Porto

José Maria Amaral Fernandes
Professor Auxiliar da Universidade de Aveiro

agradecimentos

Professor Doutor José Maria Fernandes, Professor Doutor João Paulo Cunha, Mestre Ilídio Oliveira, Eng. Luís Alves.
O presente trabalho foi efectuado no âmbito projecto "*BING -Brain Imaging Network Grid*" financiado pela FCT e pelo FSE no âmbito do III Quadro Comunitário de Apoio GRID/GRI/81833/2006.

palavras-chave

Portal de Imagiologia Cerebral, Gestão de Recursos, Repositórios Científicos Distribuídos

resumo

A Imagiologia Cerebral (IC) está na fronteira entre a neurologia, engenharia e física. écnicas de imagens médicas multimodais, tais como a Ressonância Magnética (MRI e fMRI) e Espectroscopia (MRS), Tomografia Computadorizada por Emissão de Fotões/Positrões (SPECT/PET), entre outros, são emergentes ferramentas de pesquisa médica que pode fornecer informações valiosas para o diagnóstico de doenças do cérebro. Eletroencefalograma de alta resolução (HR-EEG), técnicas para sincronizar e fundir seus resultados de análise e várias técnicas de imagem são também parte de IC.

Em Portugal, dado o facto que a maioria das áreas relacionadas com IC (por exemplo, medicina, engenharia ou física) são assuntos de investigação em muitos grupos de P&D, um consórcio de universidades de Aveiro, Coimbra, Minho e Porto criou a Rede Nacional de Imagiologia Funcional Cerebral (RNIFC). A RNIFC é uma associação sem fins lucrativos que foi formalizada e assinada em fevereiro de 2009.

Actualmente, com o suporte de sistemas digitais para armazenar imagens médicas, é possível partilhar dados entre essas instituições para melhorar o diagnóstico, e permitir investigações entre a comunidade médica de diferentes instituições.

O principal objectivo desta dissertação é descrever a implementação dos serviços de sistemas de informação essenciais para a Brain Imaging Network (BIN) que suportam actualmente o RNIFC acessível através do Portal BIN, o principal ponto de entrada para a BING. O Portal BIN permite aos pesquisadores na comunidade BING espalhadas pelo país e no estrangeiro, quer para solicitar o acesso a instrumentos científicos ou para recuperar os seus casos e executar as suas análises.

keywords Brain Imaging Portal, Resource Management, Grid-Enabled Scientific Repositories,

abstract

Brain Imaging is in the frontier between neurology, engineering and physics. Multimodal medical imaging techniques, such as Magnetic Resonance Imaging (MRI and fMRI) and Spectroscopy (MRS), Single Photon/Positron Emitting Tomography (SPECT/PET) among others, are emergent medical research tools that can provide valuable information for diagnosis of brain diseases. High-resolution electroencephalogram (HR-EEG), techniques for synchronizing and fuse its analysis results and several imaging techniques are also part of BI.

In Portugal, given fact that most of the BI related areas (e.g. medical, engineering or physics) are subjects of research in many R&D groups, a consortium of the universities of Aveiro, Coimbra, Minho and Porto created the National Functional Brain Imaging Network (RNIFC). The RNIFC is a non-profitable association that was formalized and signed in February 2009.

Currently, with the support of digital systems to store medical images, it is possible to share data among these institutions to improve diagnosis, and allow investigations by the medical community among different institutions.

The main objective of this thesis is to describe the implementation of the essential Brain Imaging Network (BIN) information systems services that currently support the RNIFC accessible through the BIN Portal, the main entry point for the BING. BIN Portal enables researchers in the BING community scattered along the country and abroad either to apply for access to the scientific instruments or to retrieve their cases and run their analysis.

Index

Table of Figures

List of Acronyms

AD	Active Directory
AG	An Access Grid
AJAX	Asynchronous Javascript And XML
API	Application Programming Interface
BI	Brain Imaging
BOLD	Blood Oxygen Level Dependent
CRUD	Create Read Update Delete
CSS	Cascading Style Sheets
DOM	Document Object Model
DTI	Diffusion Tensor Imaging
DTO	Data Transfer Objects
EEG	Electroencephalogram
EGEE	Enabling Grids for E-sciencE
fMRI	Functional Magnetic Resonance Imaging
GUI	Graphical User Interface
HSM	Hierarchical Storage Management
HTML	HyperText Markup Language
HR-EEG	High-Resolution Electroencephalogram
IoC	Inversion of Control
JDBC	Java Database Connectivity
JDO	Java Data Objects
JPA	Java Persistence API
MVC	Model View Controler
MRI	Magnetic Resonance Imaging
MRS	Magnetic Resonance Spectroscopy
OOP	Object Oriented Programming
PoC	Proof-of-Concept
POJO	Plain Old Java Objects
PSE	Problem Solving Environments
R&D	Research and Development
RAID-DP	RAID Double Parity
SAN	Storage Area Network
SARS	Severe Acute Respiratory Syndrome
SMTP	Simple Mail Transfer Protocol
SOA	Service Oriented Architecture
SRB	Storage Resource Broker
SPECT/PET	Single Photon/Positron Emitting Tomography

UC	Use Case
UI	User Interface
URI	Uniform Resource Identifier
VFS	Virtual File System
VRS	Virtual Resource System
XML	eXtensible Markup Language

Glossary

Condor: Project develops 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 queuing mechanism, scheduling policy, priority scheme, resource monitoring and resource management.

EcoGRID: Project cooperation supporting the functioning of the growing Dutch Flora and Fauna Database.

EGEE Project: Aims the creation of a seamless Grid infrastructure for e-Science. (see http://www.eu-egee.org/). The EGEE project is composed by three phases (EGEE I (2004-2006), EGEE II (2006-2008) and EGEE III (2008-2010)), being currently at the third and last phase of the project.

e-Health: Is the term given to modern information and communication technologies that support healthcare services and practice.

ESA-Flysafe: Project of the European Space Agency. Aims at demonstrating the added value of integrating different systems across national borders, to improve military flight safety and by reducing the risk of bird-aircraft collisions.

e-Science: The term used to describe computationally intensive science that is carried out in highly distributed network environment.

GeMS: Software repository for exporting and share data sets, responsible for processing generated unstructured data from a main repository to distributed read only servers.

Gfarm: Network shared file system, which can be an alternative solution of NFS, meeting a demand for a larger, reliable, and faster file system.

Globus Toolkit: Middleware that supplies the essential grid services layer that includes authentication, encryption, resource management, and resource reporting

MOGAS: System that records the resources used by each job in the Grid, identify users, organizations and virtual organizations.

Netapp: Is a proprietary computer storage and data management company providing filers, a type of disk storage device which owns and controls a filesystem, and presents files and directories to hosts over the network.

SCMSWeb: Monitoring tool that provide resource and job monitoring, accounting, reports and analysis for Grid environments.

1. Introduction

1.1. Brain Imaging in Portugal

Brain Imaging is in the frontier between neurology, engineering and physics. Multimodal medical imaging techniques, such as Magnetic Resonance Imaging (MRI and fMRI) and Spectroscopy (MRS), Single Photon/Positron Emitting Tomography (SPECT/PET) among others, are emergent medical research tools that can provide valuable information for diagnosis of brain diseases [1, 2]. High-resolution electroencephalogram (HR-EEG) and techniques for synchronizing and fuse its analysis results and several imaging techniques are also part of BI.

Development of new and better processes for healthcare community is one of the biggest concerns in this clinical related field. We have witnessed a great evolution in areas of medical science namely in diagnosis processes and in particular in imaging techniques that can provide valuable information for medical researchers and in some cases becoming mandatory in clinical diagnose [3, 4]. This has also been translated in a great demand for solutions that make efficient use of the technological resources to support and make available medical imaging in medical environments. One example is the size and number of medical images, increasing the need to store and share these images for diagnosis and research purposes.

With the support of digital systems to store medical images, it is possible to share these images among several institutions to improve diagnosis, and allow investigations by the medical community among different institutions.

1.2. The Portuguese Brain Imaging Network

In Portugal, given fact that most of the BI related areas (e.g. medical, engineering or physics) are subjects of research in many R&D groups within the Portuguese scientific system, originated a “bottom-up” process, organized initially around a group of young researchers in BI working in Portugal and abroad, to propose the creation of a BI centre and network to join all community efforts.

After a call for proposals process held by the Ministry of Science, a consortium of the universities of Aveiro, Coimbra, Minho and Porto was chosen and awarded a 4.3 million € grant to implement the National Functional Brain Imaging Network (RNICF). The RNICF is a non-profitable association that was formalized and signed in February 2009. This association is now in charge of obtaining the remaining funds and coordinates the project implementation. This network is the "embryo" of a collaborative cyber-infrastructure composed by a data provider (BI centre), located at the University of Coimbra, two integrated data processing and storage provider nodes located at the Universities of Aveiro and Porto, and a basic and clinical neuroscience data access client node at the University of Minho (Figure 1).

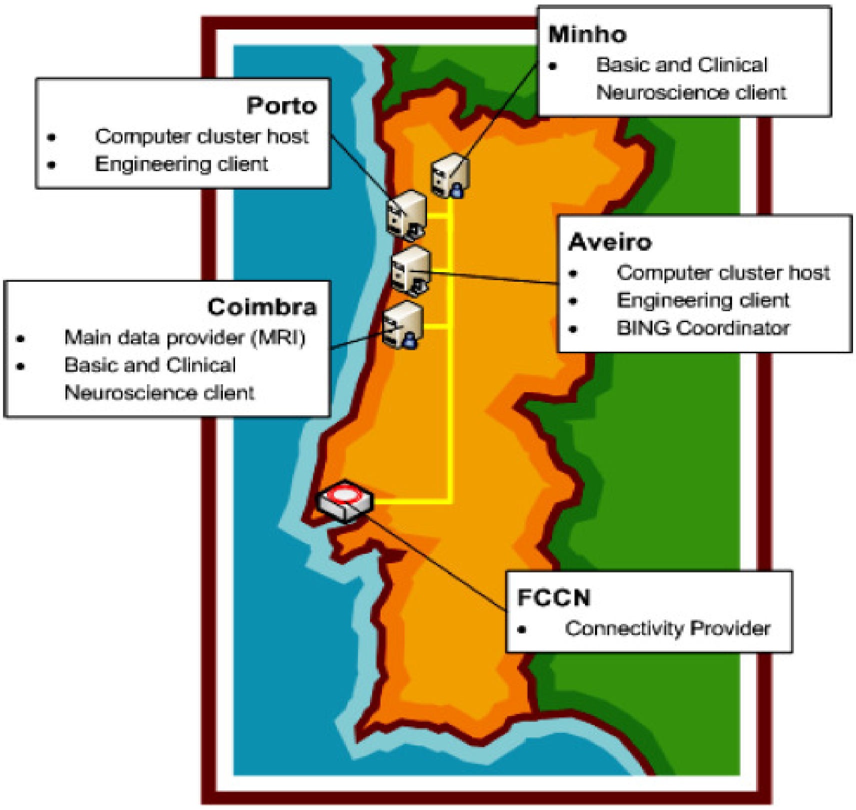

Figure 1 - Rede Nacional de Imagiologia Funcional Cerebral (RNIFC)

1.3. Objectives

The main objective of this thesis is to implement the essential Brain Imaging Network (BIN) IT services that will support the RNIFC and will be accessible through a portal, so that researchers in the BING community scattered along the country and abroad can either to apply for access to the scientific instruments, as to retrieve their cases and run their analysis. The BIN Portal as the main entry point for the BING.

The BIN Portal should provide

- Dedicated front-ends to several research BIN profiles from informatics engineers, digital signal processing researchers to clinical experts.

- Back Office, provide an information system allowing managing the BING community and production resources, including user details, instruments booking, applications for machine time and exporting work lists to equipments.
- Front Office, enabling access to acquisition processes, complex analysis on-demand for datasets. Users, using their familiar resources, will be able to harness from computing centres and Grid services integrated in the network.

1.4. Thesis Contributions

This thesis main result is the BIN Portal already deployed since January 5, 2009 and being used on a daily basis (i.e. basic time-slot registration) having already a growing amount of data registered: more than 200 exams, 10 projects and more than 200 subjects. This work also originated two publication: one the HealthInfo conference as main author [5] and more recently in the MICCAI-Grid workshop [6] .

1.5. Dissertation structure

This dissertation describes the genesis of the BIN portal - where the main contribution of this thesis lies - and the main options taken in the currently working portal. This dissertation is divided into the following chapters:

- **Chapter 2 - e-Science and networks,** presents the main projects that inspired BIN in both the high level concepts and on deployment and computational infrastructure. Like with BIN, several of these projects emerged with the increase relevance of e-Science concept in life sciences.

- **Chapter 3 - BING: a proposal for a portal and services,** is focused on BIN requirement capture, main usage scenarios and actor classification. An analysis of the domain, architecture design for BIN Portal and services. Is also included together an earlier User Interface prototype for the portal.

- **Chapter 4 - Implementation of the BIN Portal,** presents the BING structure based on 3-tier architecture and details like how to access client API and explanation of technological options used in the implementation.

- **Chapter 5 – BIN Portal Deployment,** describes the deployment of the BIN Portal including the actual hosts, equipments and service packages involved.

- **Chapter 6 - Conclusions,** sums up of the most important aspects of the development of this work, discusses the results obtained and presents some of the lessons learned as well as possible future work within this area of research.

2. e-Science and networks

The emergence of e-science networks enabled the inter-disciplinary research in biomedical areas such as medical informatics, bioinformatics or system biology, creating new opportunities for research. Of course different sciences, although sharing some common needs, naturally follow paths that lead to some differentiation. Good examples are particle physics that generates enormous amounts of data which must be quickly stored, but not necessarily instantly processed; on the other hand, data in bioinformatics is not large by comparison but requires intensive processing. One of these ramifications originated the area of e-Health [7]. e-Health is looking for an extension of the concept to consider how to provide large scale services to the user, dealing with specific problems arising in the processing of biomedical data. In e-Health, there is need for both storing and processing medical images, genetic information and other patient data. Also a large amount of computing power is needed for keeping statistics of patient records, for knowledge extraction using data mining, and for the simulation of organisms and diseases using complex biomedical models. For that reason Grid technology [8] appears often associated with e-Health. Grid has a concept has undoubtedly much to offer in dedicated networks of distributed resources (data and processing) as is attested by many projects that already present a (near) worldwide scale [9]. It is in this context that Portuguese Brain Imaging Network appears and from the beginning sees Grid as a useful resource to be integrated within a research brain imaging network rationale [10].

In this chapter we present some e-Science network projects that inspired BING and discuss their influence in some core options taken in the initial stages of BING.

2.1. SARSGrid (Severe Acute Respiratory Syndrome Grid)

The Severe Acute Respiratory Syndrome (SARS) Grid project [11] developed by Taiwan’s National Center for High-Performance Computing (NCHC) [12] - An Access Grid (AG) - based disease management and collaborative platform that allowed for SARS patient's medical data to be dynamically shared and discussed between hospitals and doctors using AG's video teleconferencing capabilities. Now that the SARS epidemic has ended, the primary function of the SARS Grid project is that of a web-based informatics tool to increase pubic awareness of SARS and other epidemic diseases. Additionally, the SARSGrid project can be viewed and further studied as an outstanding model of epidemic disease prevention and containment.

SARSGrid [13] had a significant role in helping doctors managing the in-hospital and in-home care of the quarantined SARS patients through medical data exchange and the monitoring of the patient's symptoms. SARSGrid is supported by the Pacific Rim Applications and Grid Middleware Assembly (PRAGMA) [14] infrastructure. The PRAGMA test bed is an international collaboration between different countries of America and Asia.

PRAGMA is supported by software middleware like:

- Gfarm [15] file system is a network shared file system, which is used as an alternative solution of NFS, meeting a demand for a larger, reliable, and faster file system.
- SCMSWeb monitoring tool [16] that provide resource and job monitoring, accounting, reports and analysis for Grid environments.
- MOGAS [17] records the resources used by each job, identify users, organizations and virtual organizations.

2.2. caBIG (cancer Biomedical Informatics Grid)

The cancer Biomedical Informatics Grid (caBIG) [18] initiative, overseen by the National Cancer Institute Center for Bioinformatics (NCICB) [19] in USA, was conceived to address the needs of all constituencies in the cancer community-researchers, clinicians, patients to share data and knowledge, simplify collaboration, speed research to get diagnostics and therapeutics from bench to bedside faster and more cost effectively, and ultimately realize the potential of personalized medicine. Although initially focused on cancer research and care, caBIG technology is extending the initial goals to other therapeutic areas:

- Connect scientists and practitioners through a shareable and interoperable infrastructure
- Develop standard rules and a common language to more easily share information
- Address a critical problem of data explosion that requires new approaches for collection, management and analysis.
- Build or adapt tools for collecting, analyzing, integrating, and disseminating information associated with cancer research and care.

2.2.1. Infrastructure

Under caBIG, the main engine is the caGrid [20]. CaGrid provides a service-oriented infrastructure that is driven primarily by scientific use cases from the cancer research community. caGrid provides the core infrastructure necessary to compose the Grid of caBIG, the basis for connectivity between all of the cancer community institutions, allowing research groups to access the collections of cancer research data while supporting their individual investigations. caGrid manages and securely shares information and analytic resources using locally managed access control policies and by using strongly typed data objects in XML format. The caGrid core infrastructure services (Metadata Services, Security Services and Business Activity Services) are maintained by the NCICB.

Currently caBIG is mainly focused on data sharing namely raw data generated by sequencers and labs are stored in GeMS repository [21]. GeMS have many common services available like: file storage services, authentication, authorization, messaging, plugin manager and workflows. But the most important service is exposed by the GeMS exporter architecture presented on Figure 2, built on data sharing supported by the generic publication control system (export server). This mechanism is responsible for processing generated unstructured data from main repository to distributed read only servers, providing formally structured data for data mining and specific domain applications.

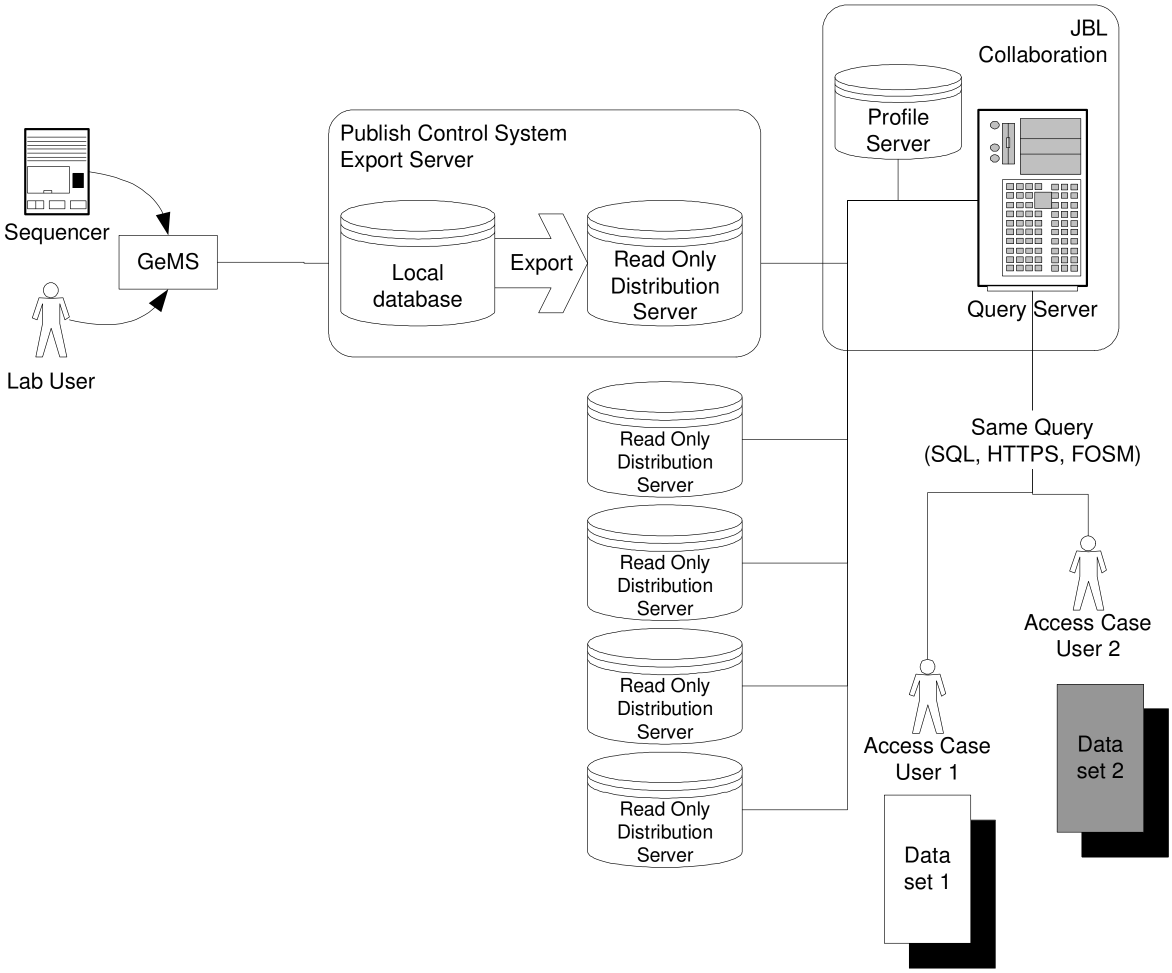

Figure 2 – caBIG GeMS exporter architecture [21]

The exporter obeys to a schema, providing mapping from generic raw data to structured data. This structured data can then in phase two, be used in domain specific applications. One of these specific application examples are Early Detection Research Network (EDRN) [21] that gathers information from a range of exported databases (Pittsburgh, Colorado, New York, etc) and generates a pool of domain specific services.

2.3. VL-e (Virtual Laboratory for e-Science)

The aim of the Virtual Laboratory for e-Science [22] project is to bridge the gap between the technology push of the high performance networking, the Grid and the application pull of a broad range of scientific experimental applications. It will provide generic functionalities that support a wide class of specific e-Science application environments and set up an experimental infrastructure for the evaluation of ideas.

VL-e infrastructure is based on the Proof-of-Concept Environment (PoC) [23], which is the shared environment for e-Science of the Virtual Laboratory for e-Science. In the PoC, the different tools and services used by and provided by the project are available and bound together in a service oriented approach. The VL-e project has four program lines, most of them containing more than one subprogram. These program lines structure the research and facilitate the dissemination of results:

- P1 e-Science in applications
- P2 Generic Virtual Laboratory methodology
- P3 Large-scale distributed systems
- P4 Scaling up to & validating in “real-life applications”

The PoC Central Facilities consist of a wide variety of resources and access points, including large compute clusters, and storage space in near-line tape storage, all accessible via grid and data management tools. Also contained in the PoC are various other application services like databases and user interface systems. The grid resources for VL-e are provided jointly by Nikhef (National Institute for Nuclear Physics and High Energy Physics in Amsterdam) [24] and Sara (SARA Computing and Networking Services) [25], who participate in a larger framework for grid computing worldwide.

2.3.1. Architecture

VL-e is essentially oriented to architecture evolutions, beginning in the PoC layer and evolving to a production stage. In Figure 3 supporting the large distributed infrastructure P3 we have the Grid middleware that will provide the Generic Virtual Laboratory P2 for the more oriented domain applications P1. The work on Grid middleware (bottom-up) and on top of it, (top-down) can be separated by different development teams. The VL-e has a concept of developing in parallel with P2 and P3 layers for rapid prototyping and proof of concept applications P4. This gives a sandbox for middleware and domain developers to search in the space of architecture solutions. These solutions when stabilized could be integrated in production layers.

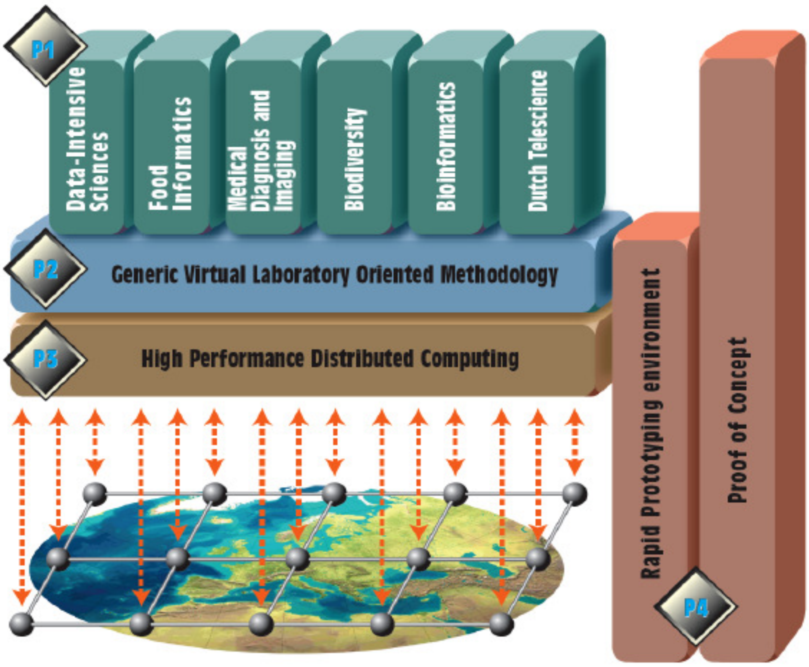

Figure 3 - VL-e global architecture (adapted from [26])

The P2 layer approach is to integrate observation and contextual data sources in a VL-e science Problem Solving Environment (PSE) [27], making these environments accessible to applications, for example through web services. They are successfully using this approach in several real life applications like EcoGRID [26] and ESA-Flysafe [26].

2.3.2. VBrowser

Among the several VL-e subprojects the VBrowser project [28, 29] is worth mentioning as it has application in the neuroimaging field and aims at providing a seamless access to data and computing within a distributed research community. By implementing a framework that abstracts several different resources under the Virtual File System (VFS) and Virtual Resource System (VRS) concepts (Figure 4), The VBrowser has a modular structure and provide a plug-in model to support resource viewers or dedicated applications.

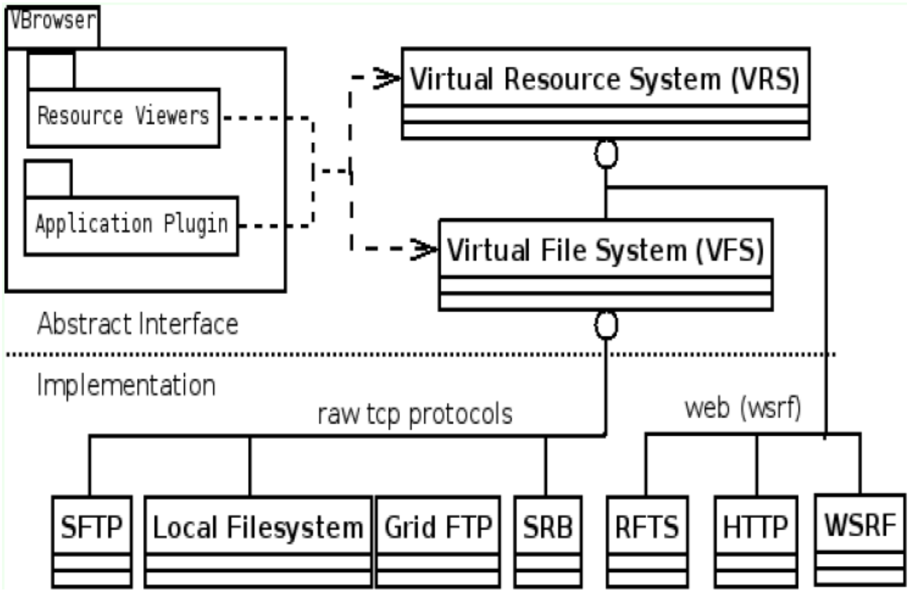

Figure 4 - VBrowser architecture [30]

2.4. Biomedical Informatics Research Network (BIRN)

The Biomedical Informatics Research Network (BIRN) [31] was launched in 2001 with the goal of fostering large-scale collaborations in biomedical science by utilizing emerging cyber infrastructure. An essential feature of the project is the collaboration of computer scientists and biomedical researchers from different research disciplines. Their goal is to design and implement a distributed architecture of shared resources usable by all biomedical researchers in order to advance the diagnosis and treatment of disease.

BIRN has four test beds, infrastructure and collaborative tools are suitable to support a wide array of clinical or scientific research questions:

- BIRN Coordinating Center (BCC) maintains and develops the hardware infrastructure for the BIRN Test Beds.
- Morphometry Test Bed focuses on understanding the issues involved in performing multi-site structural and DTI imaging studies in human populations, the recommendation of protocols that can be used across platforms and the interoperability of analysis tools and database software used in these studies.
- Function BIRN Test Bed focuses on methods for prospective, multi-site clinical functional imaging studies in human populations.
- Mouse BIRN Test Bed focuses on multi-site, multi-technique, and multi-scale integration of imaging and gene expression data in mouse models of neurodegenerative disease.

BIRN enhances the scientific discoveries of biomedical scientists and clinical researchers across several disciplines.

- Host a collaborative environment rich with tools that permit uniform access to hundreds of researchers, enabling cooperation on multi-institutional investigations.

- Synchronizes developments in wide area networking, multiple data sources, and distributed computing.
- Designs, tests, and release new integrative software tools that enable researchers to pose questions and share knowledge across multiple animal models (mouse, human, and non-human primate).

2.4.1. Infrastructure

The BIRN network is a geographically distributed virtual community of shared resources that currently consists of four test beds, supported by the BIRN Coordinating Center (BIRN-CC) [32]. Test bed science drives the continual development of BIRN infrastructure and resources while BIRN's current research projects are targeting questions in neuroscience.

The BIRN-CC implements and supports the information technology infrastructure necessary to achieve distributed collaborations and data sharing among the test bed participants. The BIRN Coordinating Center (BIRN-CC) is designed around a large-scale grid model through a grid middleware that enables the integration of geographically disparate resources into an application-specific virtual resource. The low-level grid software layer for BIRN is built upon a collection of community-accepted software systems and services distributed as part of the NSF Middleware Initiative (NMI) [33]. The Globus Toolkit [34] supplies the basic grid services layer namely authentication, encryption, resource management, and resource reporting. Globus also supports security and job submission components that allow test bed researchers to launch long running jobs on the BIRN grid. A GSI-enabled Condor [35] is also utilized with the vast resources of the BIRN Grid available through a simplified single interface. Grid middleware also provides the infrastructure to interconnect independent data collections, a fundamental requirement of scientific collaboration.

2.4.2. Architecture

Even though BIRN is based in distributed resources, users must have a main entry point. BIRN is also "built" around a BIRN Portal [36] that provides a comprehensive collaborative environment for biomedical scientists to form new projects with robust data management functionality. Project-specific forums, blogs, and various lists foster improved communication among project members. The collaborative features support access control by assigning user or administrator roles to members. In addition, projects, blogs, and forums can be marked public or private, depending on the requirements of the research groups. The Portal architecture is composed of many modular "layers", allowing for extension of each layer without disrupting the entire system.

The BIRN Portal uses the Storage Resource Broker (SRB) [37] to ensure secure access to the BIRN test bed. Through SRB, BIRN supports users to download and upload data to the SRB as well to launch tools that can be used in conjunction with selected data, such as ImageJ [38], 3D Slicer [39]. Users can apply permissions to data hosted on the SRB along with its associated metadata.

The data grid provides a way to access data sets and resources based on their attributes and/or logical names rather than their names or physical locations and allows file security to be managed on a network shared resource.

Portal centralizes all administrative details of each layer into a single unifying username and passphrase.

2.5. BIN and the BING (Brain Image Network Grid)

Since Portuguese Brain Imaging Network (BIN) is the focus of this work, we should present it and describe the current infrastructure.

As referred previously, the BIN is supported in a consortium of the universities of Aveiro, Coimbra, Minho and Porto that recently created the National Functional Brain Imaging Network (RNICF). This network is the “embryo” of a collaborative cyber-infrastructure composed by a data provider (BI centre), located at the University of Coimbra, two integrated data processing and storage provider nodes located at the Universities of Aveiro and Porto, and a basic and clinical neuroscience data access client node at the University of Minho.

As higher level objective BIN aims at setting up scientific instruments, a distributed layer and community best practices to enhance brain research. In this context, researchers in the BIN community, scattered along the country and abroad, will rely on the BIN Portal as the main entry point, either to apply for access to the scientific instruments, as to retrieve their cases and run their analysis. The BIN Portal has a twofold purpose:

- To provide an information system allowing managing the BIN community and production resources, including user details, instruments booking, applications for machine time and exporting work lists to equipments.
- To enable a friendly interface to all the layer features, including accessing user datasets and run complex analysis on-demand. Distributed users, using their familiar resources, will be able to harness from computing centers and Grid services integrated in the network (the grid is the reason of the “G” in the BIN Portal denomination).

2.5.1. BIN and BING Infrastructure

The BI centre construction in Coimbra was concluded in early 2009 and a 3T Siemens Trio MRI machine is already in trial operation on site. The IT infrastructure to support distributed collaborations between different neuroimaging research member groups is in place as described in a previous papers [10] addressing the national wide deployment of the BI network at a organizational and physical level.

The infrastructure that we call “Brain Imaging Network Grid (BING)” is offering the first services to the Portuguese BI scientific community taking the BI R&D towards an innovative Brain Imaging Grid Virtual Community.

A private Gbps V-LAN connects, through a dark fibre backbone operated by FCCN, the 4 nodes of the network where fibre routers manage data traffic. At the Aveiro node we have the main data storage equipment - a NetApp FAS3140 - with an overall of 72 TB storage space. A RAID-DP data scheme is implemented, offering a 5.5 TB fibre-channel storage and 47 TB of storage in SATA technology. In the same node, an HP 16 blade cage is installed holding 64 CPU cores on 8 blade computers at the moment. With the project evolution, we plan to upgrade this equipment to hold up to 128 cores, as the demand for computer power increases. To enable access to the internet with the maximum security, a “firewall” will be operated at the Aveiro node.

At the Coimbra node, an HP 8 blade cage is installed offering more computer power that is used for operations (internal information systems, database management systems, PACS support, etc.) and a 7 Tb SATA storage device - a NetApp 2020HA working as a "buffer" for the data produced by the local "monster" MRI machine, as we nickname the Siemens Trio system. A pre-existing cluster - IBM eSERVER Blade Center JS21 - operated by the University of Porto will also be connected to this infrastructure.

For data backup, protection and disaster recovery, we are implementing a multi-strategy security scheme with three different components:

- For the mission-critical data, a 5 Tb snap-mirror scheme between our NetApp system [40] and a twin system located at the University of Aveiro main datacenter will be used. In the case of data loss, this data can be recovered in a few minutes.
- The most recent BI data will be available from two data sources, the Coimbra "buffer" and the main storage system in Aveiro using Hierarchical Storage Management (HSM) scheme.
- For the long-term data, we will use a tape library, mounted in the Aveiro node to backup the main storage system.

Apart from the private BING network, processing and storage capabilities are also available from the Enabling Grids for E-sciencE (EGEE) Project [41] that we intend to use as a backup and reinforcement system, and that IEETA/UA has been exploring since 2005 under the EGEE Biomed VO.

2.5.2. Architecture

During the early stage of this dissertation work and of the BIN, some reflection was made in relation to BIN Portal and an underlying BING project infrastructure, based in what we have seen namely different architectures on the presented e-Science projects, some oriented to service share architecture like BIRN [31], data share provided by GeMS on caBIG [21] architecture and the test bed and prototyping approach by VL-e [22]. There's not a big difference between presentation layers and logic layers compared to BING (if we disregard implementation). The main concern is in data layer, presenting the Grid infrastructure that must provide a solution for supporting both the distributed nature of BING and its scalability. BING is concerned in scaling the data layer not only because of predicted huge data requisites, but also for handling information like subject information, acquisition information and data categorization. Although not the ideal solution, the initial BING design centralizes the information control and indexing in a catalogue system. The catalogue is a good solution in relation to grid as all stored data (e.g. brain image and signals) is and can be referenced in the catalogue at all time.

BIN Portal initial architecture (as depicted in Figure 5) is highly abstract reflecting the initial assessment made at the time. BING has a 3-tier architecture.

- The main decision on the presentation layer is the separation from the application layer in a client-server configuration. The network is explicit marked as the connection between these two layers. BING API will expose the BING workflows, rules and services to communicate with the data and middleware services layer.
- Data tier will provide storage for brain images and signals, and catalog services for metadata information. Metadata is treated here as small information parts

connected to images and signals in a central repository for future data mining maintain structure information, is data over data. BING repositories could be configured in a distributed way, in sighting here a possible solution to scale huge data sets.

- A processing unit based on cluster computing or possibly Grid environment is also covered. The processing power required for signal, image processing and data format conversions demands for this unit.

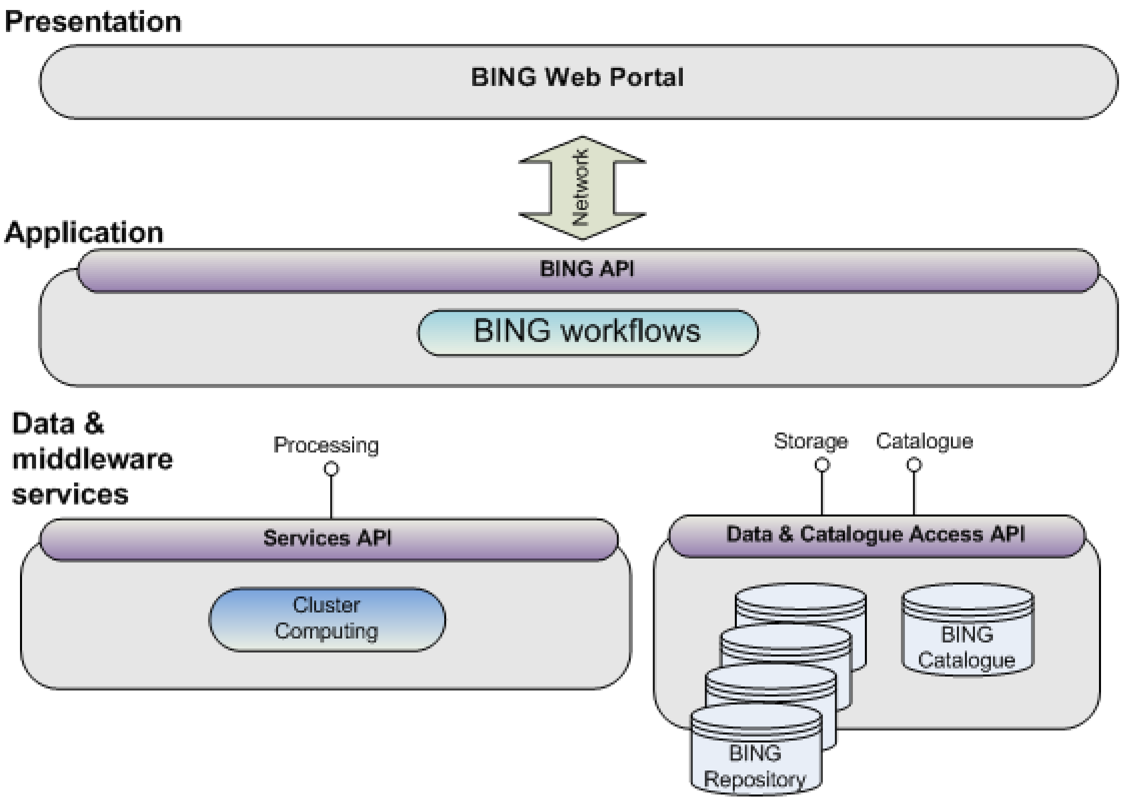

Figure 5 – First glance at BING architecture.

We will present and explain the architecture that we used to support more specifically in chapter 3.5.

2.5.3. Conclusions and decisions

Some parts of the BING project share similar concepts and functionalities that are found in other e-health science networks BIRN, caBIG and VL-e:

- Shape a community of researchers in a same place as in SARSGrid effort on joining people to attack a common problem.
- The idea of entry point (portal) for the community.
- Main objective of all projects is to share information, not regarding on how this is specifically done.
- The idea of Virtual File System to manage heterogeneous storage resources as in VBrowser of VL-e project.
- Community tools integrated into one virtual environment.

- The idea of providing mapping from generic raw data (the storage container) to structured data (shared application specific data) used in caBIG.
- Concern of expressing specific domain models for human interpretation and data processing, similar to the mouse BIRN Test Bed.

Nevertheless BING project has some specificity. As a portal, BING Portal brain imaging oriented and has is highly biased to resource allocation and acquisition process. In BING there is also a clear separation between acquisition repositories (specific to some data formats outputted from the acquisition equipments and raw data storage) and final repository (stored in a multi-modality data format for direct usage).

3. BING: a proposal for a portal and services

This chapter will present the BING project focused on requirement capturing, main usage scenarios and actor classification, analysis of the domain, identifying the main domain concepts and architecture design for BIN Portal and services.

3.1. Requirements capturing

To present the BING scenarios and requirements we addressed BING having in mind two different perspectives:

1. Data acquisition and portal specific utilities, centred on BIN Portal and services for management of security and access, equipment requests allocation and acquisition, request status notifications, storage of DICOM images, search and view.
2. Data mining, data processing and image analysis centred on BING has a provider of workspace storage for users, data transformations, workflows of job processing and personal workstations.

The focus of this thesis is the first perspective: support the data acquisition processes and users to access those support facilities through the BIN Portal.

The BIN Portal scenarios are highly related to who uses the system. The simplest scenario is the one where an anonymous user accesses the portal for public information, user request registration and login form. After being sign there are two main types of actors: the users of the portal (i.e. signed researchers) and the ones that ensure that back office operations of the Portal run smoothly (i.e. technician and administrator). We will address both in more detail.

The Researcher

Researchers (use cases on Figure 6) are responsible for project submissions (project can be a research project financed by an organization or personal project), time-slot request submission for time allocation on acquisition equipments like MRI scan. Researchers also can consult and change there information, consult projects and time-slot requests status. After acquisition data is available for search in the BING content database and accessible to a DICOM viewer.

Data sets that are in the main storage of BING can be retrieved for a workspace folder only accessible for one researcher (the owner).

Through a remote workstation the researcher should be able to tools for image analysis like FSL [42] and SPM [43] and rely on bigger computational resources like cluster environment or Grid environment to perform more intensive processing jobs ensuring faster processing results for data examination.

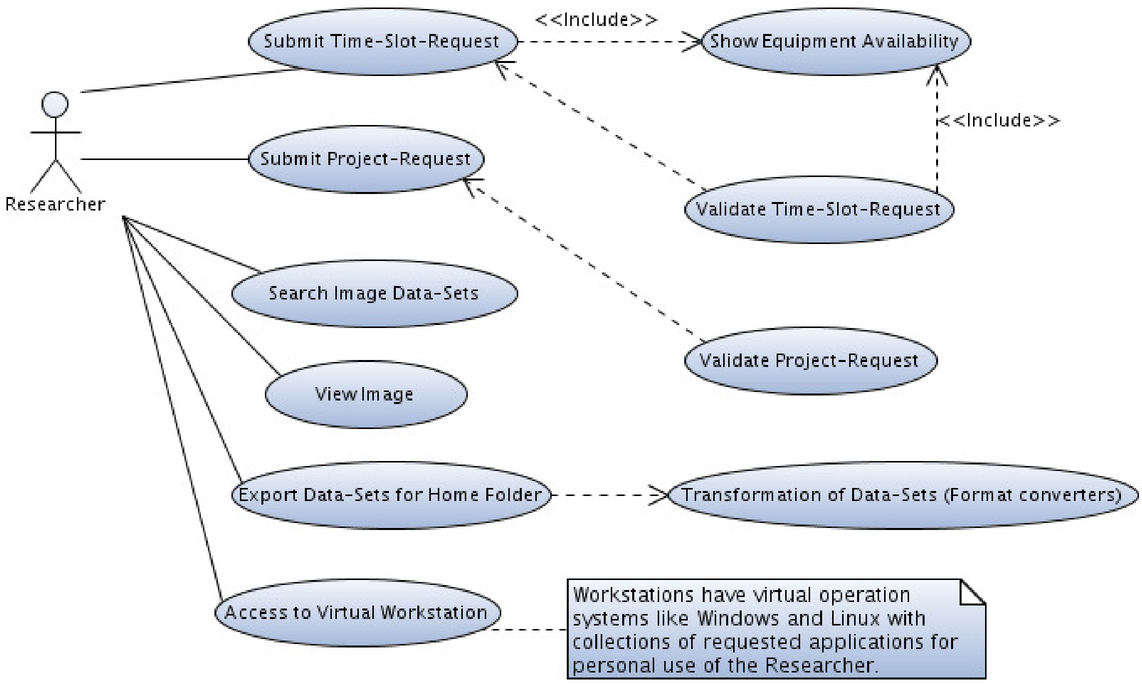

Figure 6 - Researcher use cases scenarios

Among these several options offered to the researcher the following are of certain relevance as they support legal procedures essential to access to the BING resources:

- Submit a project
- Request time slot

To submit a project (Figure 7) the Researcher starts by submitting the essential data to the project request (Project-Request). Note that this may include activities like listing modalities, exporting data options, submitting various phases of data (main, accounting, resource) that are not explicitly shown here. After introducing and sending the information, the process must pass to the Administrator. The Administrator lists all pending Project-Request and approves or rejects the request. If approved the request is originates a project within BING. All this process can be tracked through mails that are sent by the system to inform about the status of the project.

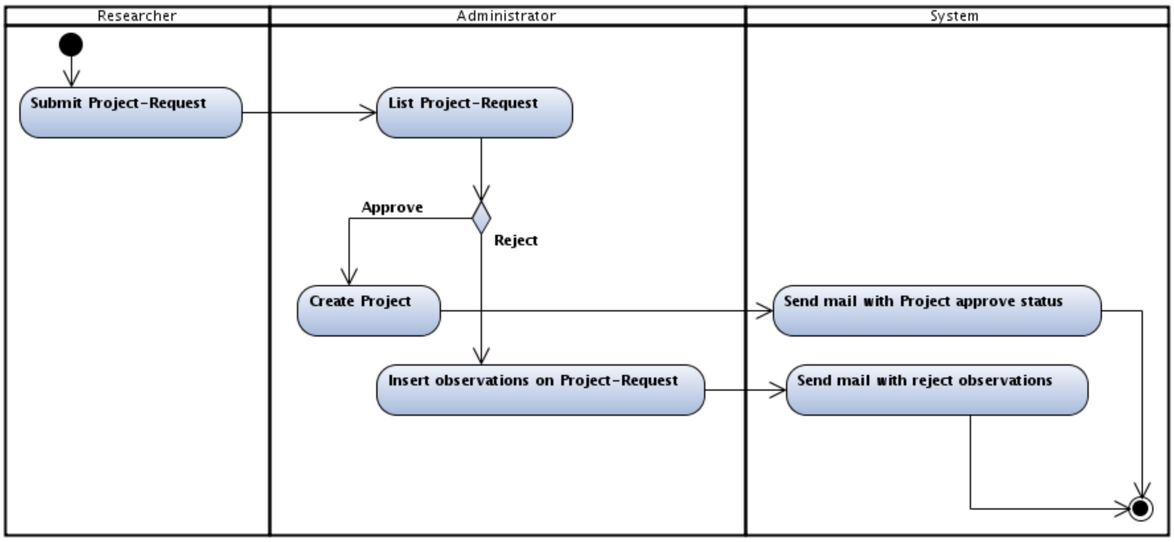

Figure 7 - Project-Request process flow

To request a time slot (Time Slot-Request) (Figure 8) we must reserve a time slot for the data acquisition equipment. First Researcher requests a time slot by selecting the equipment and date for the acquisition. In the process he can check on the availability of the equipment ("View availability" activity). If a given time slot for the equipment is not available another request must be selected.

Technician is responsible for controlling the status of the request by aborting, scheduling and finishing the Time Slot-Request in the main decision flow. If the Time Slot-Request is accepted it will generate a session with a modality specific work list ("Generate Worklist (Session)") that will support the actual data acquisition and processing in the equipment .Once again, mails are sent by the system to inform about the status of the request.

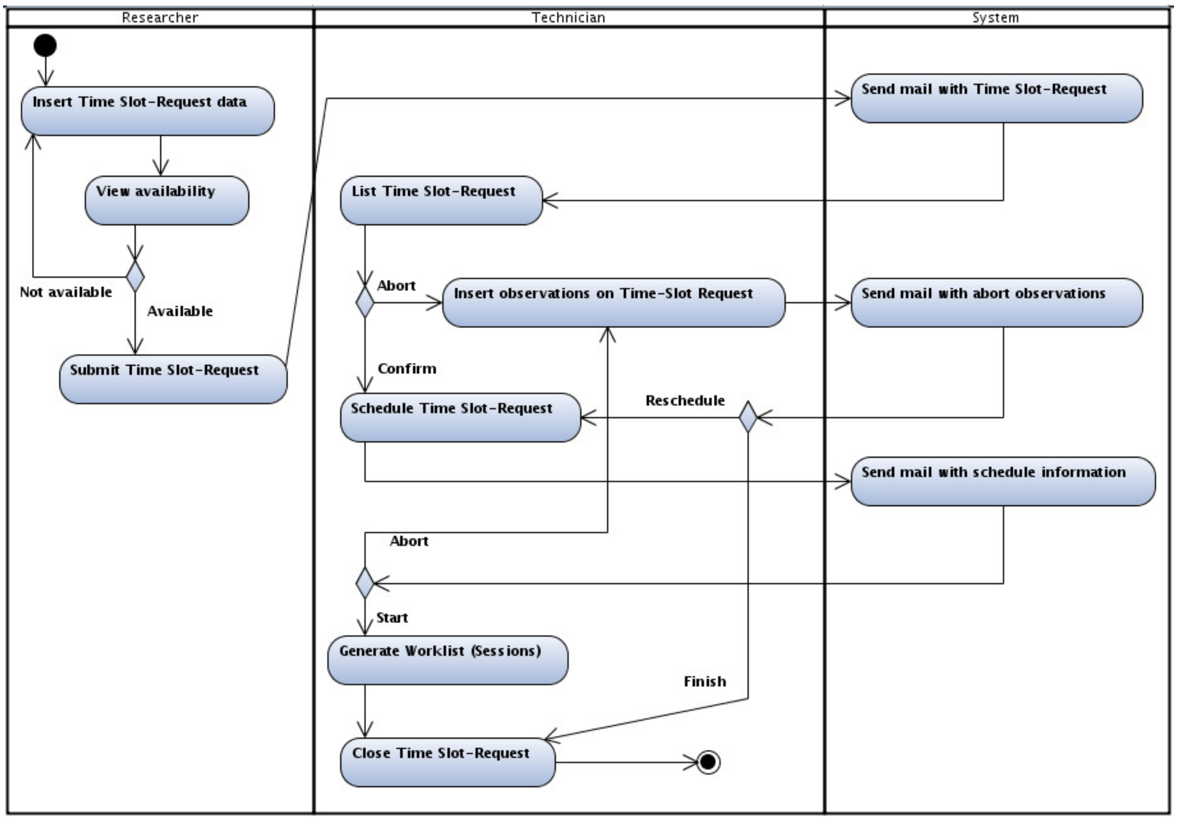

Figure 8 - Time Slot-Request process flow

The Technician

The Technician (use cases on Figure 9) has most of the time allocated to equipment and their main responsibilities are: processing project requests (accept, refuse), processing time-slot requests (accept or refuse for equipment allocation), processing acquisitions based on time-slot request for work-list generation. Technicians will provide in with there actions the images data sets for researchers.

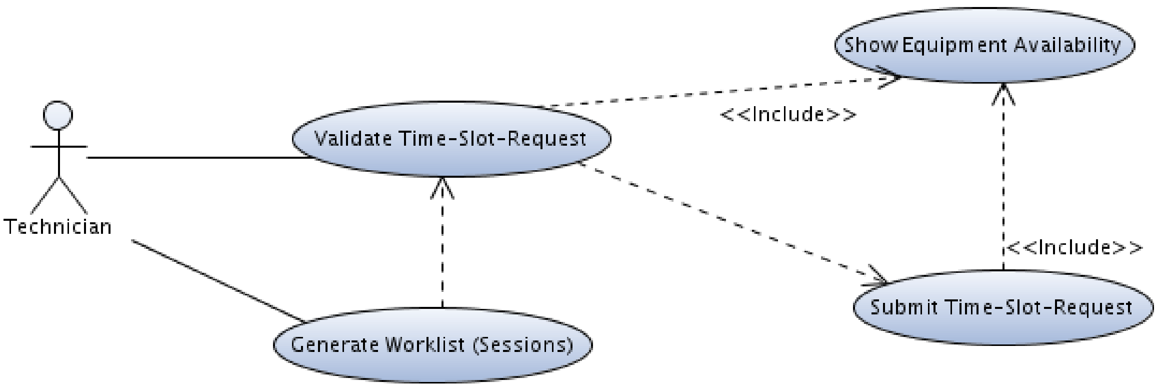

Figure 9 - Technician use cases scenarios

The Administrator

The Administrator use case scenarios are directly related to its responsibilities, namely approving or rejecting user requests and project requests. He is also responsible for data managements of equipments, modalities, organizations and subject information.

3.2. Brain imaging and the DICOM requirement

The BIN's main node has a state of the art 3T MRI scanner with respective picture archiving and communication systems (PACS) that is responsible for storing and providing the acquisitions of the scanner. For that reason, from the beginning, BIN has to manage imaging data in DICOM format [44] as DICOM is the standard de facto among the imaging acquisition equipments and related PACS system. For that reason, to accommodate data access scenarios (see previous section) implied considering basic DICOM handling functionalities to the BING.

3.2.1. The DICOM format

DICOM is a standard format from National Electrical Manufacturers Association (NEMA) [44] format. As DICOM is not this dissertation main focus we will provide only a short introduction on it.

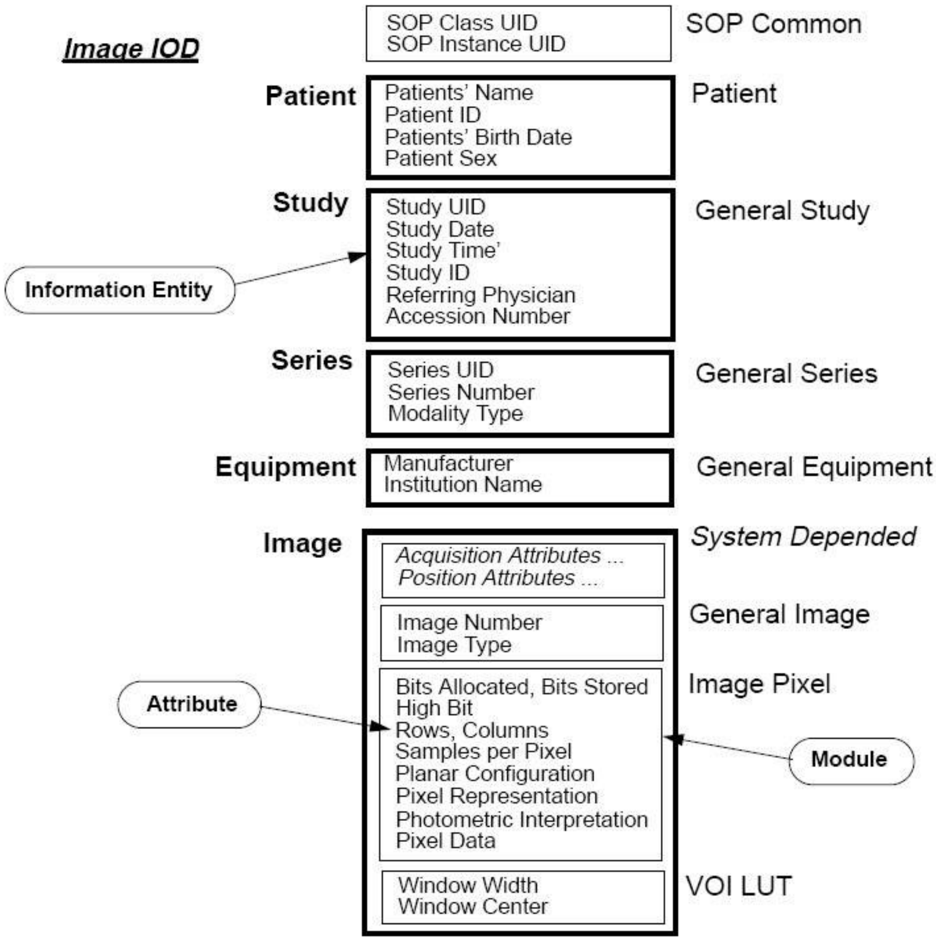

Figure 10 - DICOM file structure [45]

In DICOM model, a patient can have 1..n studies (sometimes referred to as exams or procedures). Each study consists of 1..n series. A series generally equates to a specific type (modality) of data, or the position of a patient on the acquisition device. Each series contains 1..n DICOM object instances (most commonly images, but also reports, waveform objects, etc.). All of this information is contained in each DICOM object of a study. Therefore, if a study is performed on a patient, containing 2 series, each with 10 instances, all of the instances will contain the patient and study information in it's header. The instances will also contain information regarding the series it is in, as well as its instance information. The structure of the file is a flat relational model.

Several single identification numbers - Unique Identifiers (UID) - are automatically generated for DICOM modalities and are mandatory in each DICOM file and no UID is shared by different sets of information. For example the UID is coupled to the acquisition machine, its localization, to a specific date, patient, study, a hospital and if some UID is shared by two DICOM files that means that are related (e.g. same session) or even the same dataset (e.g. a specific raw data file). This identification is necessary not only for medical and medico-legal reasons, but also to allow the devices for the formation and the management of hospital or imaging databases.

A number of online references exist on DICOM but we recommend the Chris Rorden's site as starting point (http://www.sph.sc.edu/comd/rorden/dicom.html).

3.2.2. The DICOM basic services

The essential DICOM BING services are related to import and export of metadata from and to DICOM servers and console clients. The main necessity for data importation comes from already available data in other systems that will be imported to BING and export functionality will help integration with acquisition equipments.

These two basic services must provide:

- Import of metadata from DICOM files to BING catalogue. Minimum required data is a unique UID reference for sessions and exams identification.
- Export of work list (to do list of sessions and exams) to a DICOM client or acquisition equipment console.

3.3. Domain model

From the analysis of BING we identified the main domain concepts related to (Figure 11):

- Actors: User
- Organization: Equipment, Project, Organization, Subject
- Data acquisition and resource allocation: TimeSlotRequest, Session, Exam, Sequence, Modality, Status

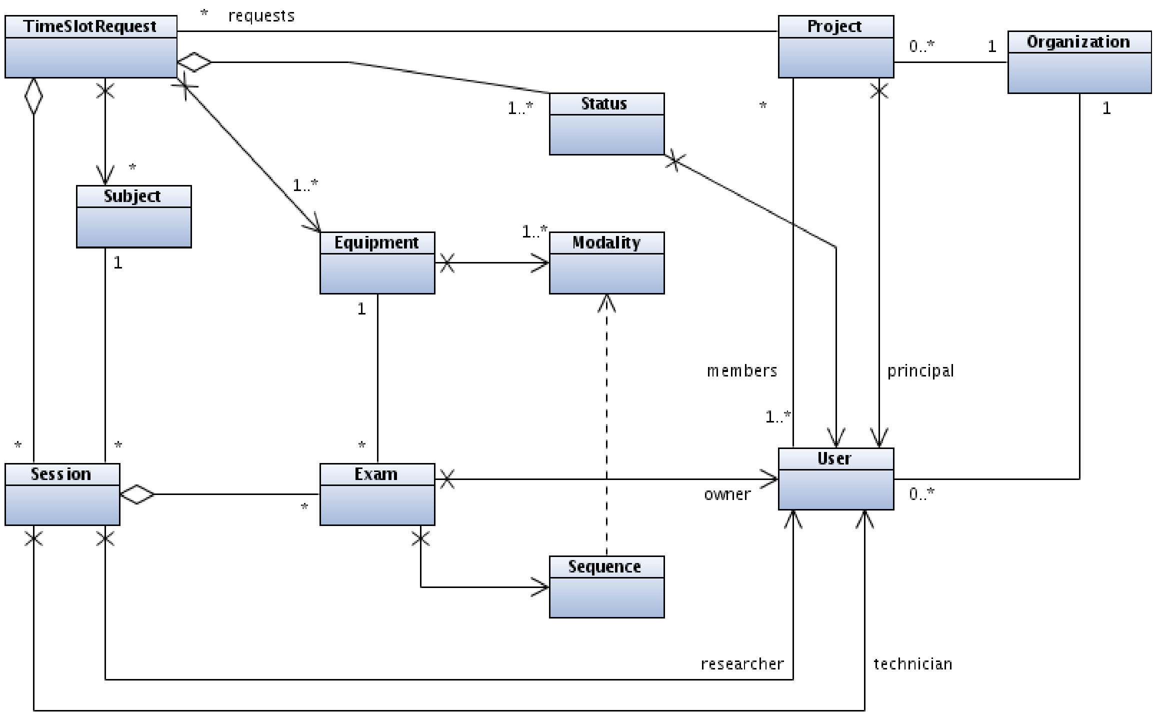

Figure 11- BING overall concept model

User

The User represents every client of the BING system. User can be associated to an Organization. The Group concept is also present to allow grouping Users in "ad-hoc" categories. The User is associated to security policies and depends, on the registration phase, on the approval of a UserRequest to access BIN Portal (Figure 12). Users are managed by Administrators.

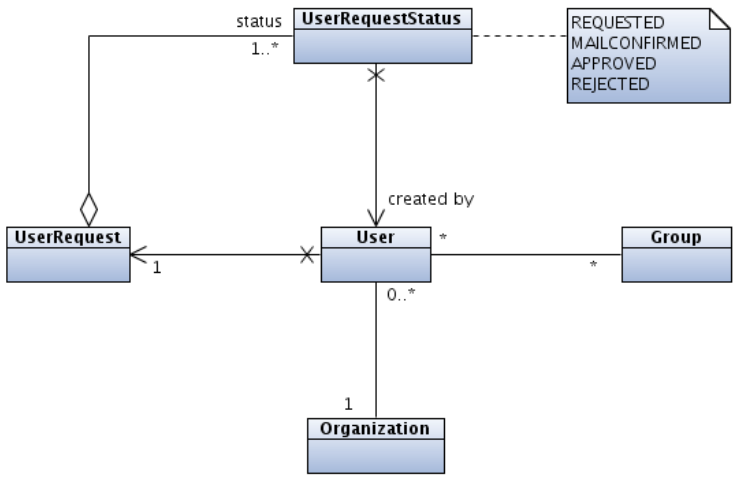

Figure 12 - User Class Diagram

Project

A Project is valid only after a ProjectRequest is approved. The Project concept glues together the information on the project's financer, ethical commission, all related future acquisition datasets and finally the member's information namely the principal investigator (Figure 13). It is managed by researchers and Administrators.

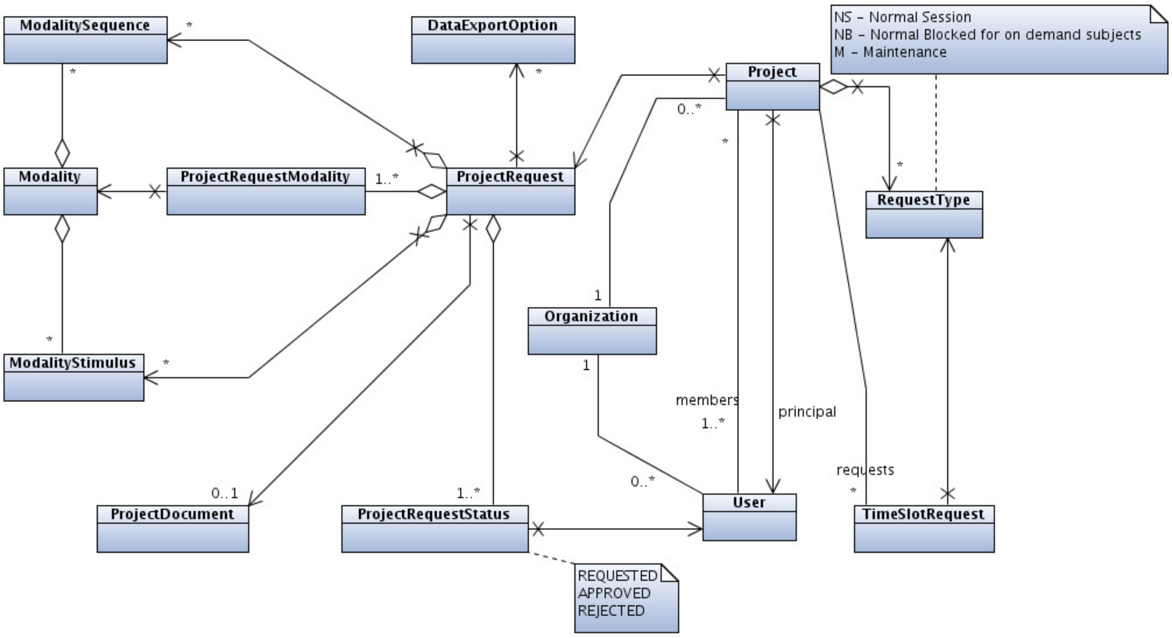

Figure 13 - Project Class Diagram

TimeSlotRequest

One of the most relevant functions of the portal is the resource allocation. TimeSlotRequest defines time slots on acquisition equipments requested by BING users (Figure 14). Requests are attached to one Project for future accounting. Subjects can also be assigned to a request, but this is not mandatory as it can be done in later stages during acquisition. Time slot requests are submitted by Researcher and managed by Technicians.

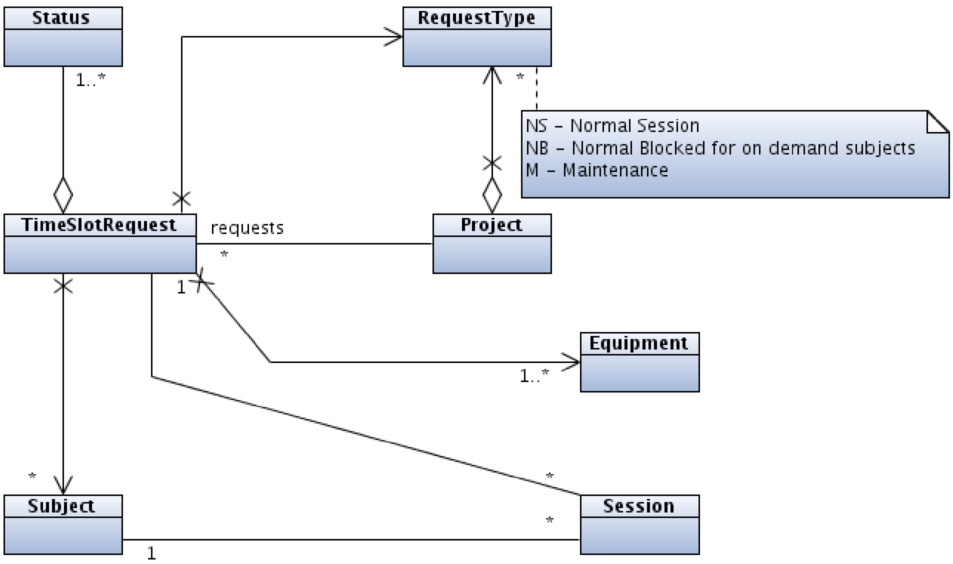

Figure 14 - TimeSlot-Request Class Diagram

Session

Sessions are part of the acquisition process related data (Figure 15). Sessions are associated to TimeSlotRequest and are restricted to that slot. The Session defines the subject associated for the acquisition and the Exams taken in the selected equipment. Exams have a sequence, in a given Modality. Sessions are directly managed by Technicians only.

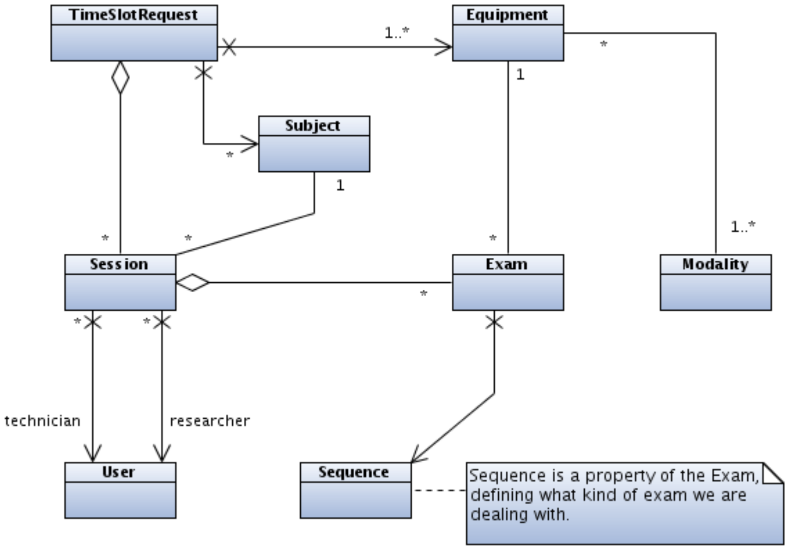

Figure 15 - Session Class Diagram

Modality

Modality entity and there associated classes shown in Figure 16 form a dynamic structure with various levels. For example MRI modality structure has levels: Region, Exam, Program and Sequence. Other modalities like EEG could have different hierarchy levels. This aggregate is prepared to hold different depths on level modality definitions. ModalityLevelDesc class define this levels and ModalityLevel hold instance values like IBILI for Region, Cranio for Exam, BOLD for Program, MPRANGE for Sequence.

Sequence is also the last level on every Modality and has the special class Sequence that is referenced by every Exam to define what kind of exam we are dealing. The Sequence is used instead of ModalityLevel because it has more information relative to the last level like description and procedures, and could use the Tag system to extended more properties (Tag System is explained below).

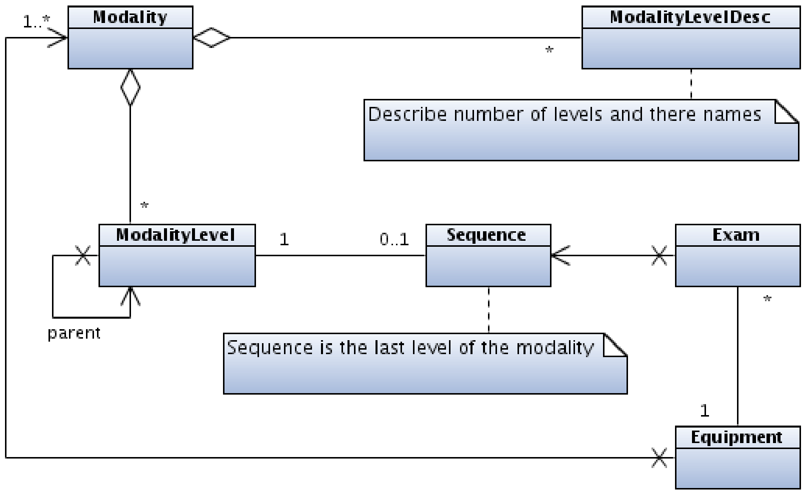

Figure 16 - Modality Class Diagram

Tag

Tags are a way of extend the Domain dynamically without changes on the scheme. The users are able to insert new fields on the entity via user interface without resourcing to development teams.

In Figure 17 we perceive that Tag has a classtype attribute that defines a relaxed association with existent realizations of TObject, like Exam and Subject. This association must be relaxed because neither the realization of TObject or Tag can depend on each other. This type of relaxed association provides an extension point for new TObject realizations without modifying the Tag class. For example, eye color is associated to the Subject class and excludes relations to other classes like Exam.

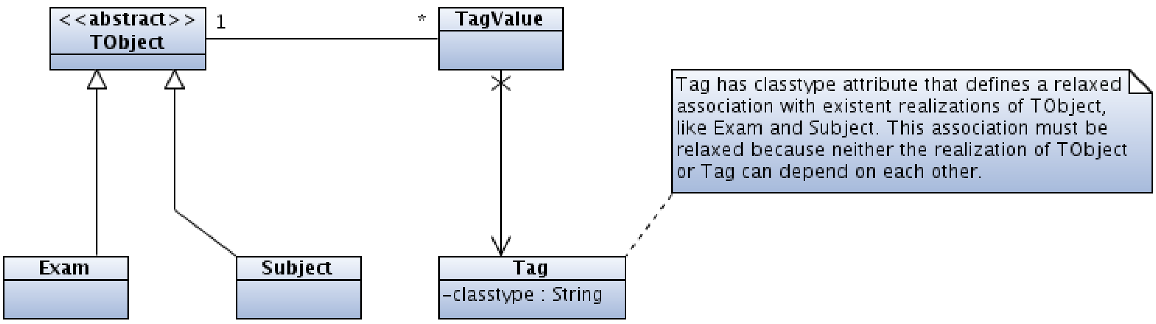

Figure 17 - Tag Class Diagram

3.4. Early BING User Interface Prototype

The first BING user interface prototype was focused on the User-, Project- and Time Slot-Request processes. The focus of the prototype was on the identification of data inputs needed and not only the presentation style. In the process of prototyping, the validation had a special relevance, namely through the identification of the data integrity rules. The prototype was initially based on a static HTML pages later extended to a fully functional version using a simulated services layer. For instance, in Figure 18, the form for adding a subject in BING is presented. Important fields were identified for the subject, (i.e., name, gender, birthdate) there data types, units and validation processes.

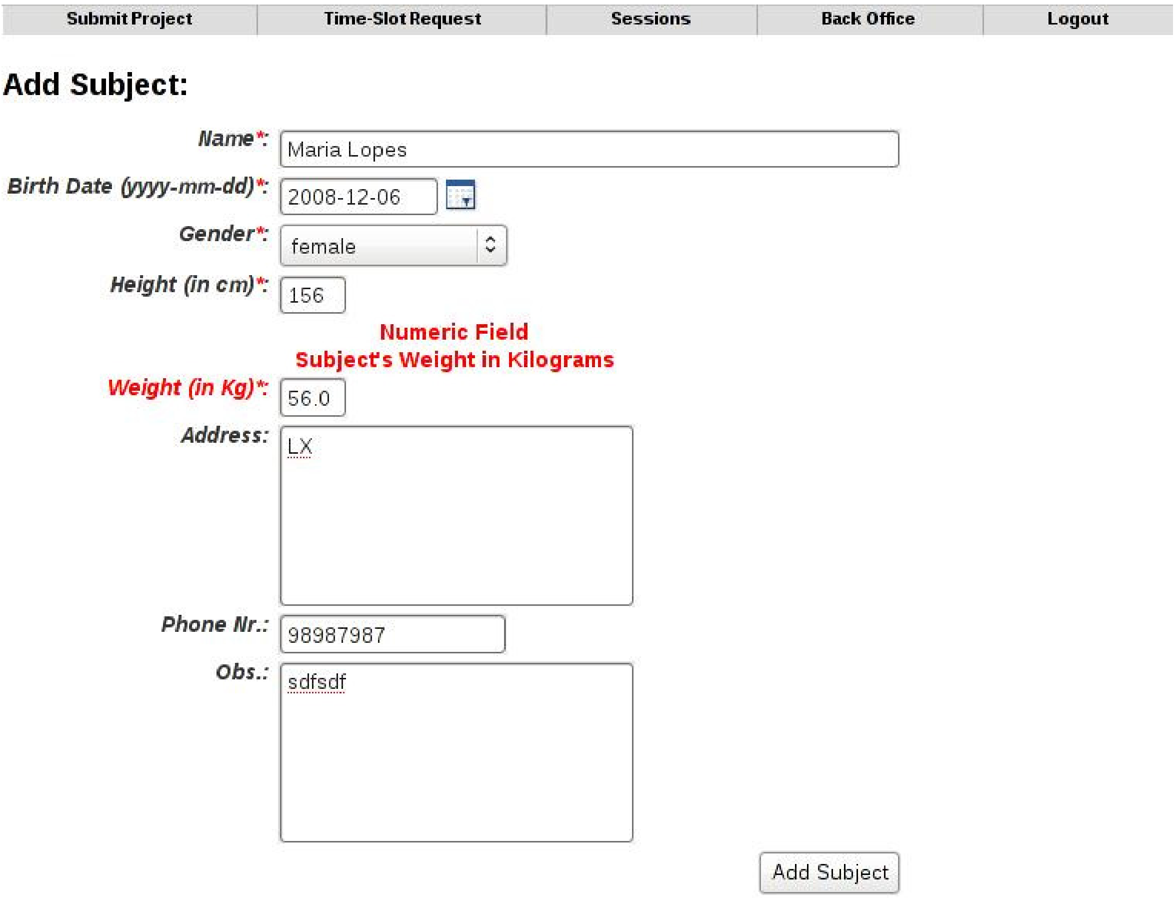

Figure 18 - Add Subject BING form

The coverage of relevant workflows was also ensured. For example the Project request submission workflow step presented on Figure 7 was prototype as a sequence of four phases (Figure 19): phase (a) related to project main information (name, summary, objectives, principal investigator, etc), phase (b) related to cost center and ethic commission approval, phase (c) to predict on future required resources (i.e. like acquisition equipments) and last phase (d) with protocol and modality related information needed for the project's future acquisitions.

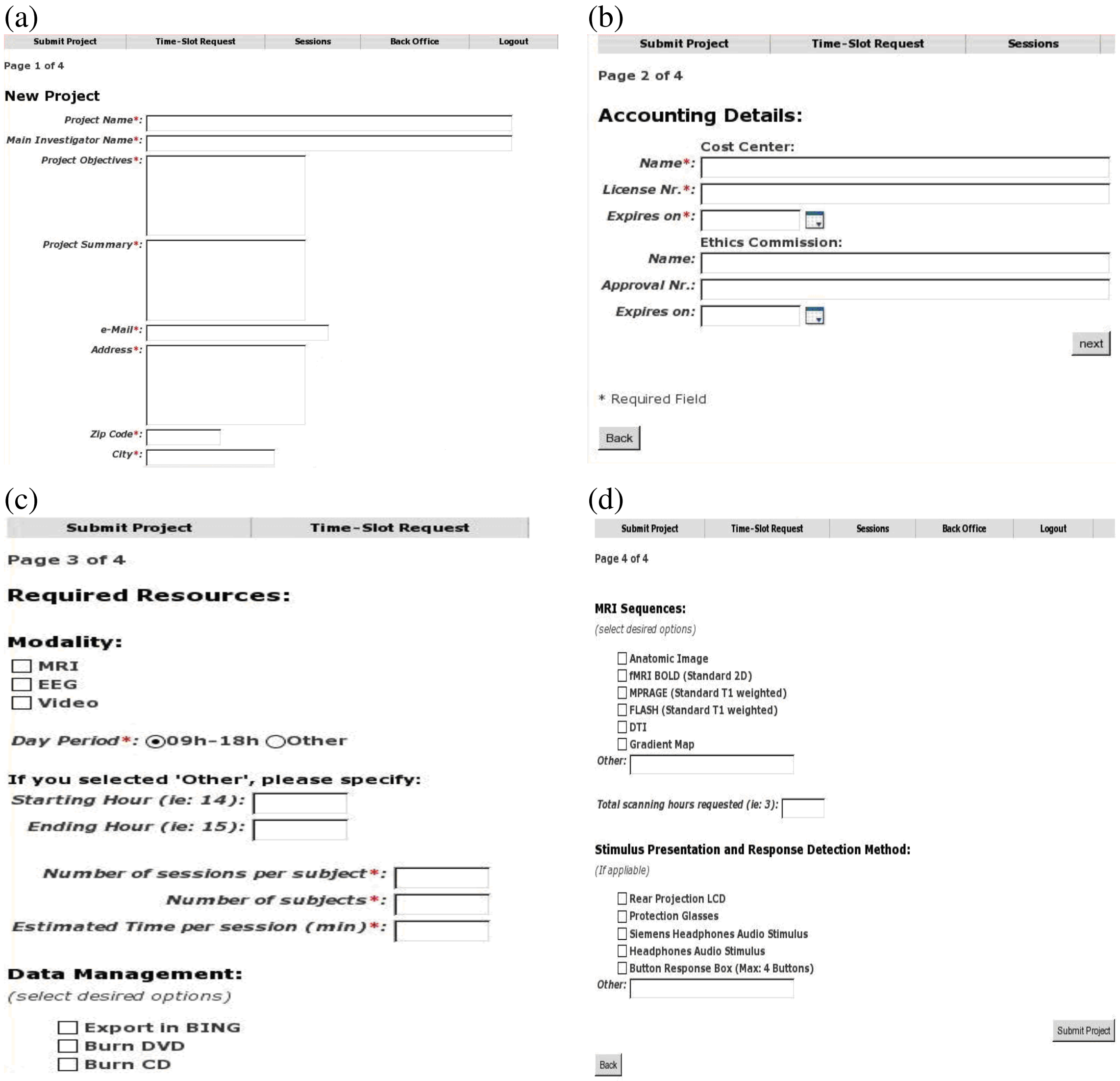

Figure 19 - Project-Request screen phases: From project (a) to protocol (d)

3.5. Architecture

A major challenge in BING is providing a long-term, high-quality data repository to serve current demanding data types, but also anticipating future value of subject cases. For that reason, although outside the scope of the BIN Portal, we addressed also the BING overall architecture in order to fit all the BING requirements and allow a natural fit for our BIN Portal solution.

From the beginning we tried to maximize the use of application programming interfaces (API) between the BING components to enhance the resilience of the solution along time. For example, data storage services abstract the actual storage technology, allowing the integration of local and Grid storage as a part of the BING data storage solution. For that reason the BING architecture (Figure 20) comprises several layers, from presentation services, based on AJAX asynchronous web technologies originally described by Garret [46], to middleware services, in change of data storage and computing.

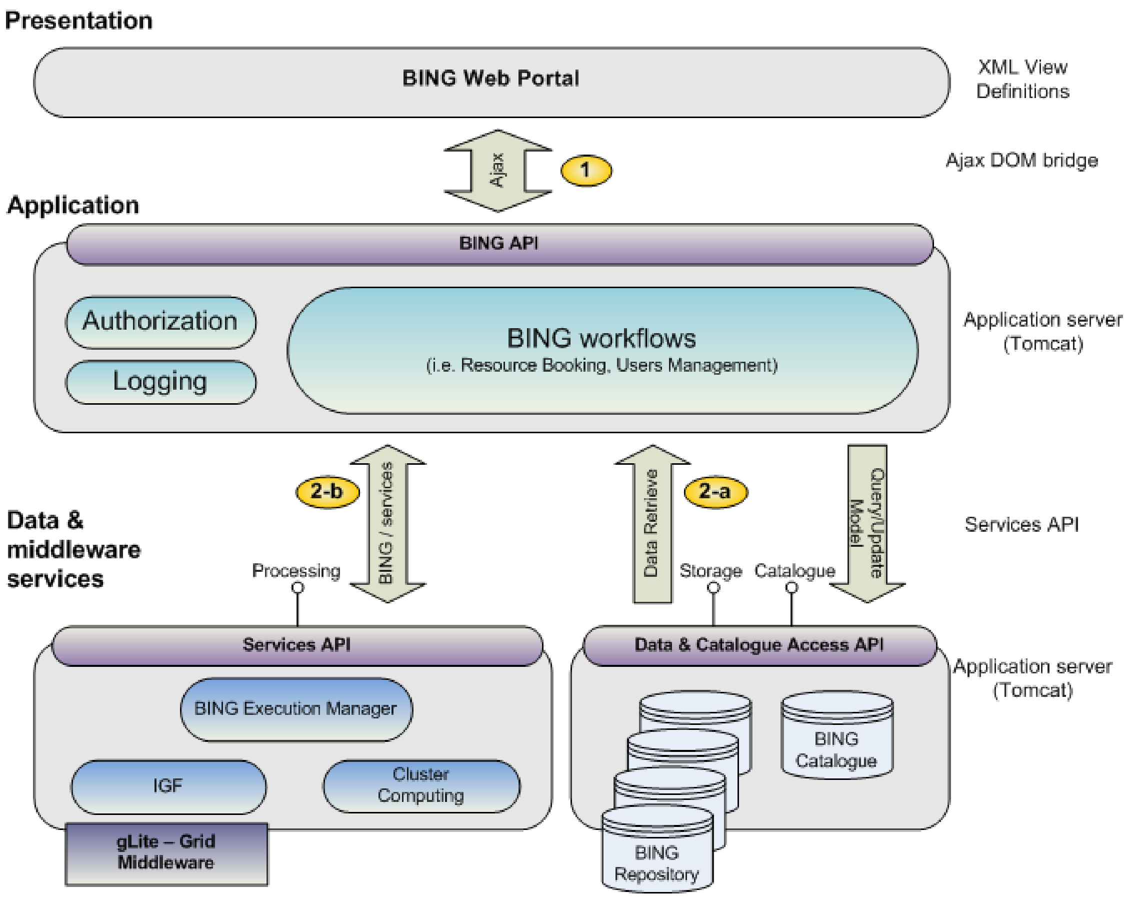

Figure 20 - BING software multi-layer services architecture.

Presentation Layer

Presentation layer is served by a web Portal, including technologies that support AJAX push and W3C Document Object Model (DOM) [47] incremental updates in the browser [48]. The DOM state is replicated in browser and server-side, when application state changes manifesting themselves in the server-side DOM propagate to browser DOM changes. Incremental DOM updates are transmitted to the browser via AJAX engine where the changes are reconstructed and the browser DOM is modified appropriately. This mechanism results in superior presentation updates because only minimal incremental changes are required, not full page refreshes as we have come to expect from web applications. Component layouts and data bindings are defined in XML files and application design is defined with Cascading Style Sheets (CSS) defined by W3C [49], these are the only development steps used in the presentation tier. The user events are processes by the AJAX bridge and forwarded to server side processing (step 1).

Application Layer

The application layer has two roles: 1) provide an abstract layer of services with BING semantics and 2) coordinate the data/service tier resources in order to implement such high level services (e.g. register a new subject implies preserving a session with the presentation tier while executing several BING catalogue operations). During these operations a security module enforces access policies both at user and user group level. The module uses a simple Uniform Resource Identifier (URI) based strategy together with application server authentication policies.

The coordination of BING workflows is based on controllers (available as public services) that implement the BING logic associated with each available service. Among the basic services are the data access and catalogue that rely on data/services tier accessed through a BING specified API.

The Model-View-Controller (MVC) pattern [50] between the presentation and application tier is achieved using the BING high level API, that abstracts both direct access to the BING catalogue (step 2-a) or, in some cases when using external services, through specific BING workflows (steps 2-b) (e.g. format and type conversions).

Data and middleware Services Layer

Besides the basic information system, data stored in BING is mainly related to brain studies. Theses studies are characterized with metadata stored in the Catalogue. The Catalogue allows to discover which information exists and to establish structural relationships between data fragments (e.g. the results of several modalities for set of subjects in a study). The Data layer provides the services to query and feed the catalogue (e.g. register a new case).

Different types of repositories can be maintained for different acquisition systems always with referential connection to the catalogue. At the moment there is a predicted repository for DICOM data acquired from an MRI equipment.

Along with storage, there are basic services to run computation on data. These are included in BING middleware services API and allow submitting tasks to the BING Execution Manager. This component coordinates input data retrieval (from BING storage) and job submission to two the cluster centres in the network.

Integrating Grid middleware

The BING software services were designed to work both with and without a supporting Grid infrastructure. The later is currently implemented and resorts to the BING software modules to abstract storage (keep track of physical copies, allow descriptive metadata, move data sets, etc.) and computing elements (meta-schedule to distribute computing tasks between the two data centers and collect results). While the BING software modules and the hardware architecture in place support user needs, the natural evolution of the system is to connect to existing Grid infrastructures and harness from available resources, as to share brain data available in the BING.

These gateway services to the Grid are being developed in the scope of another project named IEETA Grid Framework middleware (IGF) [6, 51, 52] to be later integrated in the BING system, under the control of the BING Execution Manager. Users are expected to use the BIN Portal with the same level of transparency; Grid operations will be handled by the application logic, ensuring the authentication (through proxy certificates), data movement, and job descriptions automatic generation.

As explained previously this document scope is on BIN Portal and the next chapters will only provide information on this.

4. Implementation of the BIN Portal

This chapter describes the BIN Portal implementation, and the deployed application software according to basic requirement identified previously. The BING web portal was designed to be dynamic and user friendly similar to a desktop application. We also tried to maintain a coherent look between pages, so the human interfacing interaction could be more intuitive.

BING web portal is divided in two main projects called BingModel and BingPortal. In the MVC concept [50], BingModel maps to the Model and BingPortal to Views and Controllers. Next chapters will describe these two sub projects of the BIN Portal.

The web portal was developed using open technologies: ZK [53] providing rich user interfaces based on AJAX [46] technology. ZK also follows the MVC [50] pattern and a server-centric approach that most of AJAX frameworks don't use. As a server-centric solution, the BIN Portal application is hosted on the application server and all processing is done on the server. Communication is based on events and the client browser is only used for data presentation, in contrast with applications that do all of their processing on the client side, and communicate by a service layer. Since the component states are maintained on the server side, retrieving data or processing business logic requests require no extra work.

4.1. BingModel project

In Figure 21 is presented a package diagram of BingModel. The BingModel is in the "pt.ieeta.bin.bis" package and is provided in different sub-packages. The "api" is where the BING API interfaces are located; the "dal" for Data Access Layer is where the "api" implementations are located; "dicom" for some DICOM file converters and readers; "mail" for mail queue implementation, and model for BING domain classes referred in domain chapter. The most important and focused here is the "api" package, obviously because is the main bridge to the other related project BingPortal.

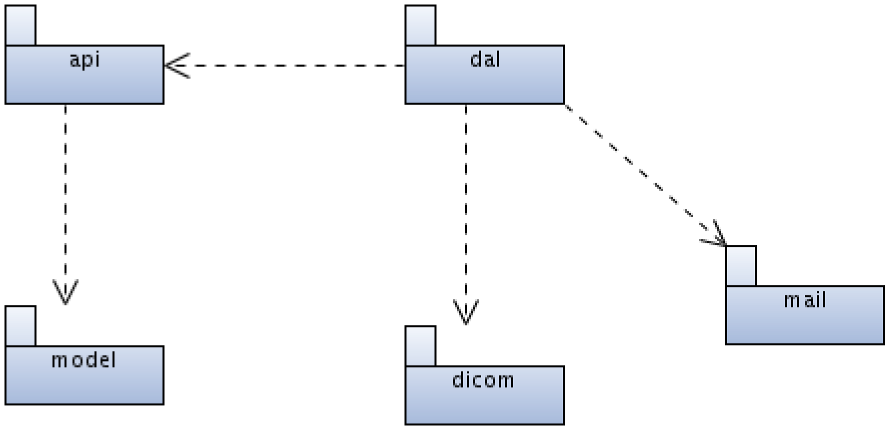

Figure 21 - BING package diagram

4.1.1. BING API

The BING API provides the basic services (i.e. Create, read, update and delete BING information also known as CRUD) to access to the BING model. API services are accessed through a proxy class named "P" that is responsible for returning the implementation of a service interface specified in the request. Through the P class we keep a record of the client access to the services using a logging interceptor. The main classes of services in the BIN API are (Figure 22):

- IUserManager used to query, manage security and access control configurations of BIN Portal users (e.g, users, groups).
- IOrganizationManager used to manage and query for organization information.
- ISubjectManager used to manage and query subject information.
- IEquipmentManager used for manage and query equipment information.
- ISessionManager used to create acquisition sessions with exams work list for future DICOM export.
- IModalityManager used to manage and query modality information.
- IProjectManager used to manage, query and submit projects and project requests.
- IRequestManager used to submit and query time slot requests.
- IDashBoardManager used to manage and query the message board.
- IMailManager used to query mails in mail queue.
- IOtherResourceManager for other related resources that don't fit in any other service interface but are necessary to expose (e.g, time slot request types, export option types, sequence types).

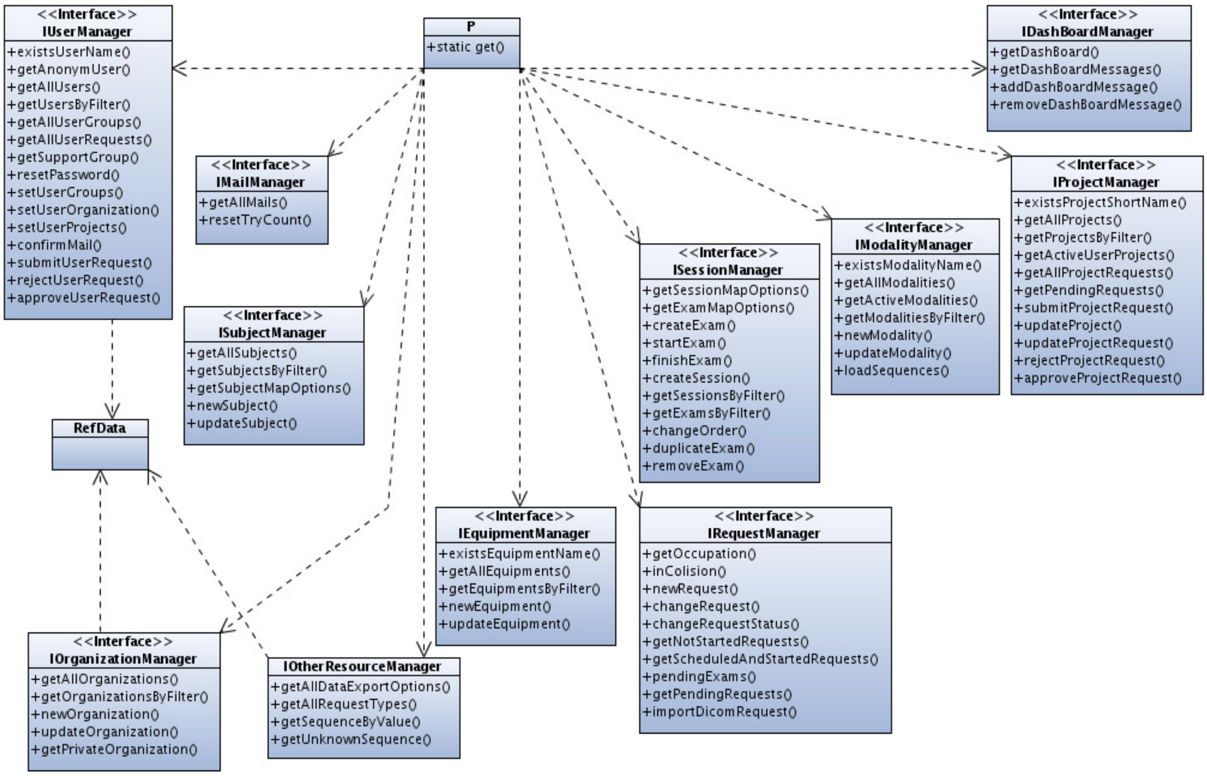

Figure 22 - BING API overall diagram

Some services depend on a RefData class that has references to essential data that must be present in the database tables for the application to work. These references are basically defaults for example the Private Organization (the default choice for Organization), Anonym User used in public BIN Portal scenarios, Unknown Sequence when the exam sequence is not defined or group names (researcher, support, etc).

4.1.2. Using the BING API

In order to provide a simple entry point for the BING API, a main P (Proxy) class was created based on the Factory pattern [54]. To instantiate a service class, the user just has to access the proxy and select the desired service interface. A reference to the new chosen interface is returned, and the user can start using the desired functions. We will present a few examples on how to use the API.

```
IUserManager u_mng = P.get(IUserManager.class);
List<Group> groups = u_mng.getAllUserGroups();

List<User> users = u_mng.getUsersByFilter("micael");
Organization p_org = P.get(IOrganizationManager.class).getPrivateOrganization();

if(!users.isEmpty()){
    u_mng.setUserOrganization(users.get(0), p_org);
}
```

Figure 23 – Example of using the entry point from BING API

In Figure 23 the IUserManager interface is returned by the P (Proxy class), used to list all groups and to get a list of users whose login name begins with “micael”. The private Organization entity is returned by the IOrganizationManager. The two returned entities are therefore used in the “setUserOrganization()” method associating the Organization to the User.

The approach on how to use the BING API is the same for other interfaces, only class methods are different. The other way to get or change data from the domain is to access directly to the entity objects demonstrated in Figure 24. A list of active Modalities are returned by API and then Modality description and a set of ModalityLevelDesc are returned from the first Modality and printed to the standard output.

```
List<Modality> a_modalities = P.get(IModalityManager.class).getActiveModalities();

if(!a_modalities.isEmpty()){
    Modality first_m = a_modalities.get(0);
    System.out.println("Modality: "+first_m.getName());

    Set<ModalityLevelDesc> m_l_desc = first_m.getLevelDesc();

    for(ModalityLevelDesc ml : m_l_desc){
        String description = ml.getDescription();
        int level = ml.getLevel();

        System.out.println("\tLevel: "+level);
        System.out.println("\tDesc: "+description);
    }
}
```

Figure 24 – Example for getting the first modality and description

The real object returned by the P class is an instance of the “java.lang.reflect.Proxy” class that is a cast to the selected interface by method “P.get()”. The actual implementation of the interface is selected from the “dal” package. This selection is hard coded configuration on the P by a “java.util.Map” class, more exact “Map<Class<?>,Object>” that maps interfaces with implementation. This is a fine place to use some kind of Inversion of Control (IoC) as originally proposed by Johnson and Foote [55] [56] that would separate the configuration from the code. In this case a hard code version was preferred, because it’s a simple case of a generic IoC system. The implementation class when submitted to changes must be recompiled, but, since these configurations won’t be a target for run time alterations a hard code version is fine contrasting to the mail configurations that can be changed if the SMTP server is moved to a different place.

4.1.3. BING Object-Relational Mapping

The resource to relational databases for storing objects in Object oriented languages needs the definition of an Object-Relational Mapping (ORM). A good introduction on ORM is given by Ambler in [57] - parts are available at http://www.agiledata.org/essays/mappingObjects.html. In short, an ORM establishes a relation between object concepts, attributes and relations to tables, fields and table relations respectively. Although several standard options exist namely the Java Persistence API

(JPA) or the Java Data Objects (JDO) in BING we use the JDO-ORM system. More information on JPA and JDO can be seen in the official sites http://java.sun.com/javaee/overview/faq/persistence.jsp and http://java.sun.com/jdo/ respectively.

For the back-end supporting database, the technologies chosen were PostgreSQL [58] as database management system, JDO standard and JPOX/DataNucleus that provides a persistence solution for java from DataNucleus project [59] implementation for Object-Relation Mapping between the domain model classes and the relational database tables. At the date the choice of JDO over the recent JPA standard was essentially stability, JPA also allows you to develop Plain Old Java Objects (POJOs) and persist them using a standardized API but it's specification is not as mature or as feature rich as the JDO API, nor does it provide the flexibility of using any type of datastore [60].

The OR-Mapping on JDO is located at "package.jdo" file by defining the relations from the domain classes to the database tables. The mappings are all in XML, separated from the domain class code. This decrease dependencies, and is a way to change mappings without code change. A second approach with java annotations could be used, is slightly better for the compilation processes since the compiler can verify some annotation aspects against the associated code. In the project we used the XML definition version.

The ORM database connections and other configuration flags are in "persistence.xml" following a standard description used by both JDO and JPA. It defines what persistence "spec" is in use, what implementation (in case of BING is JPOX implementation), database dialect, JDBC drivers (Postgresql) and JDBC string configurations.

4.1.4. Low level Mail queue services

BING relies on messages to communicate between the system and the users. To ensure that, regardless of the status of the network, BING messages are not lost, we added a failsafe. The Mail queue (MailQueue class) and mail services (MailService class) ensure that mails are delivery even when the SMTP server is not responding (momentarily). This is done by keeping a record of the mail queues in the database, and by ensuring the unsuccessful messages sends are repeated when SMTP server is available again. All this process of sending and storing the message is supported by the transaction system of the database.

The Mail queue is provided by the Mail entity and associated classes. The low level API is provided by the classes MailService, the bootstrap of the service and entry point to put mails in queue, and MailQueue that inherits from a Java Thread that, when started, control the in and out mail messages, the persistence state and SMTP communications.

The mail queue configurations are placed in a java property file "mail.properties", this file contain SMTP host name, user, password, protocol, port, etc. Mail configurations are separated from the code, so that it can be changed when application is running (need MailService restart; this operation can only be executed by the support team).

4.2. DICOM services

As stated previously (chapter 3.2) BIN Portal must provide two basic DICOM services (import and export). The import services are essentially used for metadata importation of DICOM files to the BING catalog, the minimum important information to retain is a unique UID (already explained in chapter 3.2.1) that will refer the images on the acquisition repository. The images itself are imported to the acquisition repository and depends on the type of repository used.

This service is backed up by helper classes located at “dicom” package. They are interface IDicomImage and implementation DicomImage (maybe the names are not so correct since DICOM is not just for images). Essentially this classes map the DICOM domain and data from DICOM files to the BING domain. The existent methods in IDicomImage like “getSubjectName”, “getSessionDescription”, “getExamOID”, etc, are BING domain concepts that return values from the DICOM groups and elements inside the file header. A simple library was used for DICOM reader from UCLA Department of Radiology [61].

Also the export flow of information to DICOM clients is included in the application. After the technician creation of sessions and exams it is created a work list. This work list is properly formatted to inform DICOM clients on the work to do. Available at the moment is only the work list for MRI exams. They are exported by a database view and client connection using ODBC driver [62].

4.3. The BIN Portal

The BIN Portal (BingPortal project) is based on ZK framework. Through XML layout files (ZUL files), ZK generates dynamically both the presentation (the View) and backoffice handlers to be used by the controllers implemented, in the BING case, in Java. The ZK project reflects the organization of the BING main scenarios (i.e. “backoffice”, “publicv”, “researcher” and ”support”) or actors (i.e. “administrator/technician”, ”anonym”, ”researcher” and ”support” actors). This division will provide a better separation of UI screens based on security access. The current BIN Portal implementation is located in package “pt.ieeta.bing.portal” package.

For illustrating the current presentation of the ZK based portal we present some of the screens. In Figure 25, a screen to allow Creating, Reading, Update subjects to the BING repository (typical CRUD operations). In Figure 26 we can see the form to request for a user login (User Request).

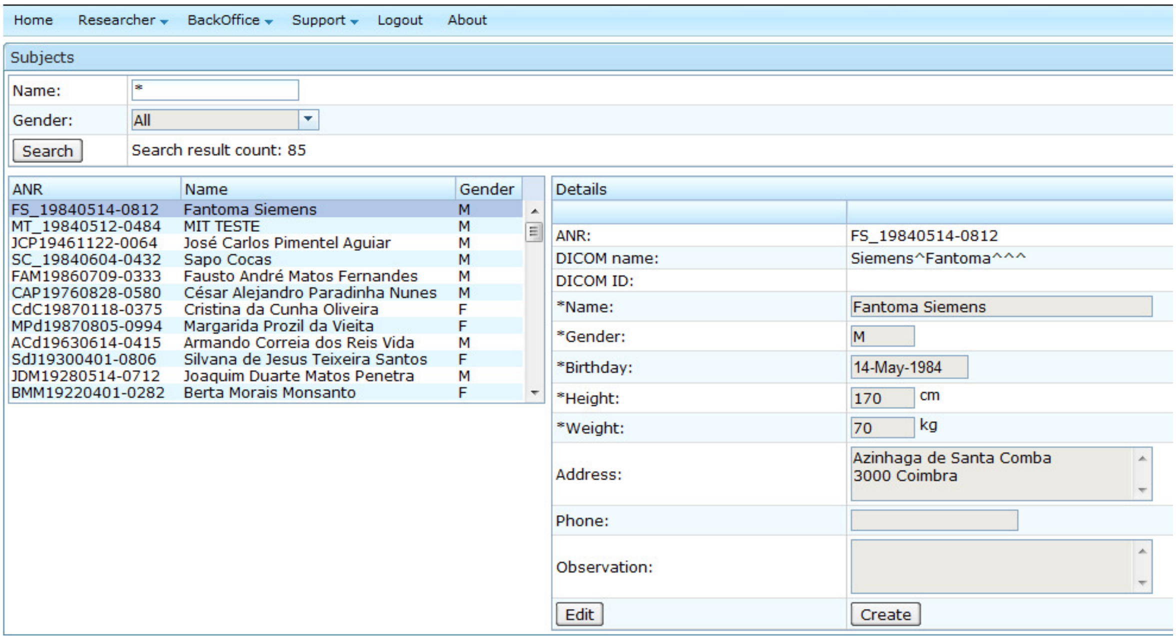

Figure 25 - Example of CRUD form (Sujects)

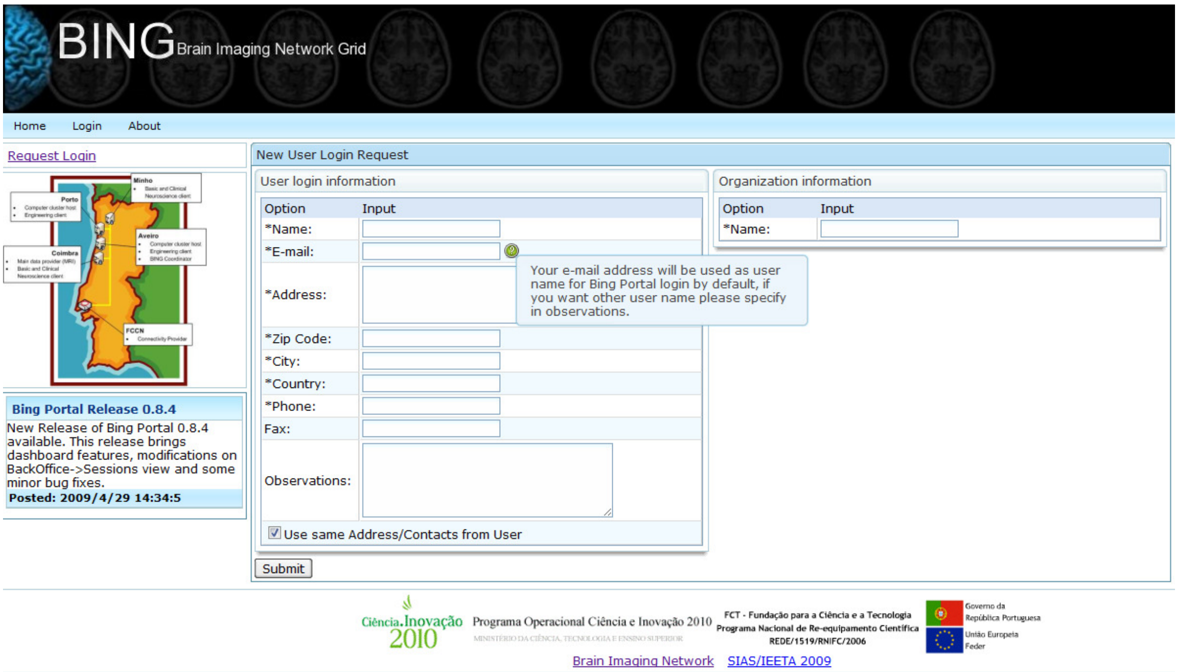

Figure 26 - Login request form

4.3.1. Use of the ZUL files

Although not essential in the context of this dissertation, we would like to illustrate the use of the ZK' ZUL files to define user interfaces. BING views are fundamentally ZK ZUL files (files with zul extension). These files in XML format respecting the ZUL schema provide a UI description with ZK specific components. This description with a controller binding, data binding and the ZK engine can output HTML and JavaScript that build interfaces like the one in Figure 27.

In comparison with the initial user interface prototype, the ZK enabled us to have a desktop like look and feel more in contrast to the initial web based concept. Through the ZK facilities we could use event based interface to respond to component changes without a page request, gathering information directly from the data services and distributing information to other components on an asynchronous publish/subscribe pattern [63].

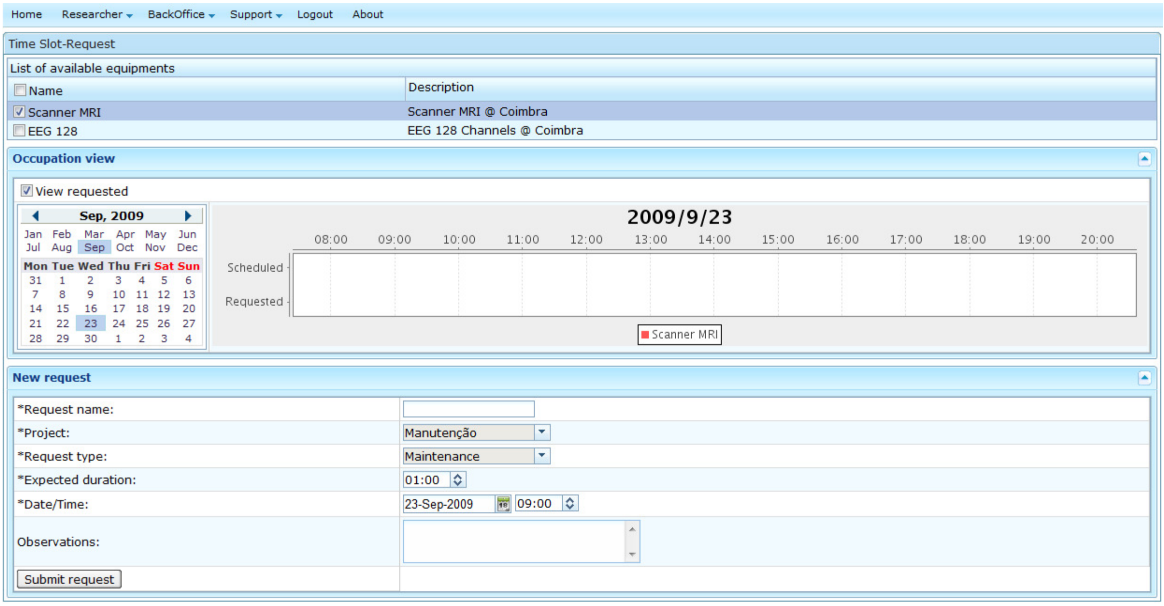

Figure 27 – Time-Slot Request form

From the ZUL definition of the Time-Slot request form (definition in Figure 28), we can perceive that no HTML or JavaScript is used in this view, even though this can be inserted in the view. The controller binding is provided on the window component by the attribute “apply”, in this case “ieeta.bing.portal.controller.researcher.TimeSlotRequest”. We have inside the window, a “listbox” component that lists all active BING equipments. Equipment information is provided by the data binding on “model” attribute that reference the “equipments” variable of type “java.util.List” provided by the controller. The “listitem” interact over the equipments list and for every one creates a data bind to equipment denomination and description. The ZK engine makes the rest to provide correct HTML. The rest of the definition uses the same principles with other ZK components [64].

```
<?page title="Time Slot-Request" contentType="text/html;charset=UTF-8"?>
<zk>
    <?init class="org.zkoss.zk.ui.util.Composition" arg0="/resources/template.zul"?>
    <window id="win" apply="ieeta.bing.portal.controller.researcher.TimeSlotRequest"
        self="@{define(content)}" title="Time Slot-Request" border="normal">
        <listbox id="equipmentList" model="@{win.equipments}" checkmark="true" multiple="true">
            <auxhead>
                <auxheader colspan="2" label="List of available equipments" />
            </auxhead>
            <listhead sizable="true">
                <listheader label="Name" width="100px" />
                <listheader label="Description" width="100px" />
            </listhead>
            <listitem self="@{each=equipment}" value="@{equipment}">
                <listcell label="@{equipment.denomination}" />
                <listcell label="@{equipment.description}" />
            </listitem>
        </listbox>
        <separator />
        <panel framable="true" title="Occupation view" border="normal"
            collapsible="true">
            <panelchildren>
                <grid>
                    <rows>
                        <row spans="2">
                            <checkbox id="viewRequestedCheckBox" label="View requested" checked="true"/>
                        </row>
                        <row>
                            <calendar id="calendar" />
                            <chart id="schedule" model="@{win.ganttModel}" title="@{win.chartTitle}"
                                width="1000" height="150" type="gantt" threeD="false"
                                fgAlpha="128"
                                dateFormat="yyyy/MM/dd-hh:mm" />
                        </row>
                    </rows>
                </grid>
            </panelchildren>
        </panel>
```

Figure 28 - TimeSlotRequest.zul file (part)

4.3.2. Session management

Session management encompasses the techniques employed by web applications to transparently maintain context data between HTTP requests. This problem is a particular concern in Web development because HTTP is a stateless protocol. Session management entails the application sending the client (in most cases, a web browser) a session token identifying a session variable in server.

ZK framework as most of UI web frameworks already has context variables available for server side code, session and application contexts. Session context is the most used because of the one-to-one client association that can be used for: security tokens, shopping cart list, transitional form data in a wizard, etc. Application context is not so used, most of the time because application configurations can be provided by static or singleton classes.

Although these two contexts, ZK is mentioned here because of other uncommon context used in web applications, the “desktop” context. This has one-to-one association to a client view instead of a client itself, and it is fine-grained relative to session context.

Desktop context are useful to maintain persistence sessions like JPA or JDO sessions. Persistence sessions are mostly used in two common ways; maintained in session context or not maintained (a new one by page request). The last one brings in a problem in persistence engines, when session is closed and data from an entity is still required but is not available, the lazy fetch problem [65]. So, the simple solution to solve the lazy fetch problem is to maintain a long persistence session in the client session variable. This also brings in a memory problem; referencing old data for a long time can "explode" the memory heap in Java. Desktop is ideal place to sustain the persistence session, because desktop is associated to a view, all required lazy fetched data is available for that view, but this data is not longer required if the view is closed, and persistence session is closed with the view.

The desktop management on BING is provided by a ZK listener located on "pt.ieeta.commons.listener.JDODesktopListener", class that essentially manages the JDO session. This listener maintains the desktop context variable in the correct "ThreadLocal" variable for every client HTTP request.

4.3.3. Input validations

Input validation is an important part of UI forms, maintaining data consistency and protecting against script injections. Nearly every active attack out there is the result of some kind of input from an attacker. Secure programming is about making sure that inputs from bad people do not do bad things.

Some input validations are located in the services (BING API) most of them related to data consistency, but, UI makes a part on this input validation, most of this UI validations are related to field validation, format and data types. Field validation on UI provides usability, for example by returning an error message on a change focus event instead of providing all input errors just on submission.

ZK have some input validation tags for the ZUL/XML files and components like on calendar the "no past" and "no future" validation, configured in the constraint attribute. However any class that extends the "org.zkoss.zul.Constraint" interface can be used by this attribute. BING includes some of these custom validations on "pt.ieeta.bing.portal.validator" package. These are essentially constraints for some related questions like "user name exists?", "equipment name exist?", "modality exists?", etc. These constraints connect directly to BING API services therefore consulting BING catalog and throwing a ZK WrongValueException with an error message if validation fails.

4.3.4. Security Filter

The portal security is based on Uniform Resource Identifiers (URI) defined Internet Engineering Task Force [66] . URI's are used in the verification for mapping from actors (groups) to folder names. Since security is a common part for a number of applications, the "SecurityFilter" class is located in "pt.ieeta.commons.security" package and not in BING specific packages. The "SecurityFilter" is an extension of "javax.servlet.Filter" and the main functionality is to compare the logged user group (the actor name) to the URI pretended access. If user is not logged in, it is redirected to the "login" page, or if there is a difference between the group and URI, is redirected to "access deny" page. The "login" and "access deny" URI's are configured in the "web.xml" file like any java servlet filter, these are "loginPage" and "accessDenyPage" servlet parameters. Filter mappings are also defined, the BING standard for these are "/backoffice/*", "/researcher/*" and "/support/*" for the respectively folders and actors defined previously.

The SecurityFilter must also contact a designated interface ISecurityAdapter that acts like a mediator between the Filter and the specific application, in this case the BIN Portal that has the SecurityAdapterBing class implementation, requesting and confirming user data from the BING database. The need for an interface is for different security implementations, BING uses a database but can be changed to AD [67] in the future. The servlet Filter parameter to configure this is “securityAdapter”.

4.4. Implementation related issues

The migration from the BIN Portal model to the BingPortal implementation implied solving some issues namely some identified through interaction with the final users.

We identified that in the transition that similar classes were used to store status with similar purposes (e.g. “Status”, “UserRequestStatus” and “ProjectRequestStatus”) that were merged into one. Some concepts were merged like “User” and “UserRequest” or “Project” and “ProjectRequest” pairs to simplify the implementation and enable a more “centralized” control of the related workflows.

The technologies initially used in the prototype were a mean to reach the final version of BIN Portal. These initial choices proved not to be the most suitable. For that reason it was necessary to develop manual code for DTO’s and repeat most of the validation and constraint codes in the client side (UI or presentation view).

Some problems reported by clients say that some UI’s are dense (dense as containing concentrated data in the same view) and need simplification. This could be true for some UI, but the main intention was to provide fewer steps to do the same flow. Achieving balance is some times hard work. More interaction with the client could improve this issue.

One of the major problems was the synchronization between the BIN Portal and the DICOM work list used by the equipment (e.g. the MRI scanner). The major issue was defining a common view model between the two parts. Although the integration was not as initially modeled, the synchronization is performed: the work list is exported to the MRI equipment console by a BING database view and ODBC [62] connection from the client. At the moment is possible to view the work list in the MRI console but some details (like sequences) are not available. This is most to a limitation on the client itself than a limitation from BING view.

4.5. Final considerations on the implementation

The decision to divide the implementation of the BIN Portal in two different projects guaranteed that, through the API, no hardwired dependencies existed between the model and the portal. Since the service API provides a secure way to access the data domain (integrity constraints and field validations), it is possible to access this layer (with just a few lines of code more) to provide web services for other future environments.

The usage of ORM solutions based on JDO standard and JPOX implementation (using a PostgreSQL back-end database), proved to be a good choice for a quick and effective development of the data access application layer. The JPOX and ORM/XML definitions permit an easy way to bind the domain model classes with the respective tables in the relational database.

The use of ZK allowed quick, intuitive and easy development of an interactive web based portal. In that sense, ZK doesn’t differ much from other similar frameworks like RichFaces [68] and ICEFaces [69] (both providing AJAX communication). The most relevant difference is that ZK follows the convention over configuration principle simplifying the initial setup and subsequent UI configurations. However the usage of this kind of web frameworks can also bring some disadvantages to the development process. If the developer wants to implement some functionality that requires a more specific approach that is not supported by the available framework controls, the development of new custom controls or the customization of the available ones can be a painful task

5. BIN Portal Deployment

The proposed web portal is available in the BING production network since January 5, 2009. It is installed in the RNIFC data centre in Aveiro (Figure 29 a) and is supporting the operations of 3 Tesla Siemens Trio MRI (Figure 29 b) machine located in Coimbra on a daily basis being regularly used by BING technicians. There are more than 20 users registered in the system scattered along various organizations (i.e, ANIFC, IBEB-FCUL, Univ. Santiago de Compostela, Dep. of Psychology UMinho) with more than 10 projects approved and more than 200 exams acquired.

(a) (b)

Figure 29 – The RNIFC resources: (a) the Data centre in IEETA at Aveiro and (b) the 3 Tesla Siemens Trio MRI – the main RNIFC equipment - at Coimbra

The deployment diagram in Figure 30 capture the distinct number of computers involved and shows the communication modes employed between components. The core components for BIN Portal are an Apache Tomcat Server and PostgreSQL installed in BING hosts named SIGMANAGER and MPI respectively. The SmartLink is a component from Siemens connected directly to a PostgreSQL view that import the work list from BIN

Portal to the MRI Console.

We are also using UA SMTP mail services for the BING messages described earlier.

BING network considered a Metropolitan Area Network (MAN) connecting Coimbra and Aveiro datacenters with a fiber channel link is the network infrastructure supporting all BING services.

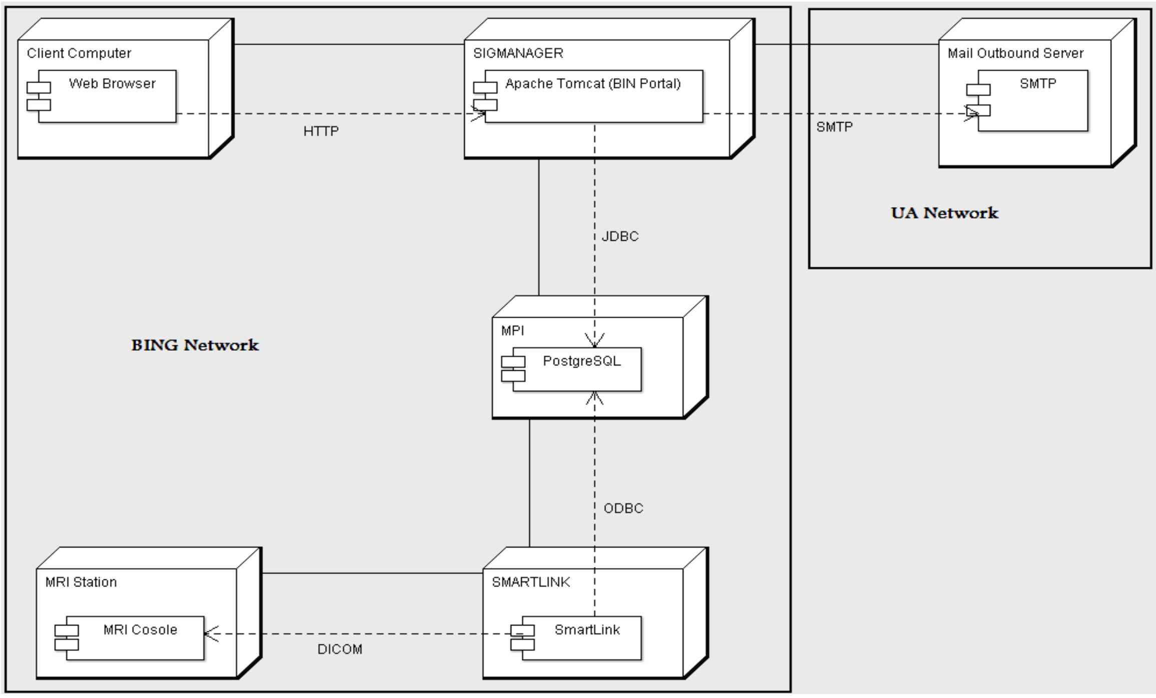

Figure 30 - Deploy diagram for BIN Portal

Additional attention is needed to the files “persistence.xml” (Figure 31 a) in BingModel and “mail.properties” (Figure 31 b) in BingPortal when deploying the BIN Portal to other environments. These files have configurations to database and mail server connections the only dependencies from BIN Portal showed in the deployment diagram.

(a)

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="1.0" xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001,
    <!-- Online Store -->
    <persistence-unit name="bis-unit">
        <properties>
            <!--DEVELOPMENT-->
            <property name="javax.jdo.PersistenceManagerFactoryClass" value="org.jpox.jdo.JDOPersistenceMan
            <property name="javax.jdo.option.ConnectionDriverName" value="org.postgresql.Driver"/>
            <property name="javax.jdo.option.ConnectionURL" value="jdbc:postgresql://localhost:5432/bing"/>
            <property name="javax.jdo.option.ConnectionUserName" value="[redacted]"/>
            <property name="javax.jdo.option.ConnectionPassword" value="[redacted]"/>

            <property name="javax.jdo.option.NontransactionalRead" value="true"/>
            <property name="org.jpox.identifier.tablePrefix" value="jdo_"/>
            <property name="org.jpox.identifier.case" value="LowerCase"/>
            <property name="javax.jdo.mapping.Schema" value="bing2"/>
        </properties>
    </persistence-unit>
</persistence>
```

(b)

```
#server numbers
num_servers = 1

#Server 1
mail1_protocol = smtps
mail1_auth = true
mail1_user = [redacted]
mail1_passwd = [redacted]
mail1_host = mail.ua.pt
mail1_port = 465
mail1_from_address = bin-operations@ieeta.pt
mail1_from_name = Brain Image Network (IT Operations Center)
mail1_debug = false
```

Figure 31- Configuration files: (a) persistence.xml file and (b) mail.properties file

6. Conclusions

As proposed initially, this dissertation presents BIN Portal that was designed, implemented and deployed during this thesis according to the specification and is currently accessible in http://bing.brainimaging.pt/Portal/. The most important results is that the current BIN Portal is begin used on a daily basis and is in function since January 5, 2009, has already registered more than 200 exams in 200 subjects, approved 10 projects and is used by 21 users. This thesis work also contributed to two publication: one the HealthInfo conference as main author [5] and more recently in the MICCAI-Grid workshop[6] .

As an evolving network it is natural that BIN requirements will change along time and some new requirements will be posed to BIN Portal. Some issues that could represent future lines of work were identified and may represent natural additions to the current work:

- Security access and Single Sign On are main central points to BING network. At the moment the access rules for BIN Portal is based on simple database information, but the plan is to override this to Active Directory (AD) integration. This was not initially planed because most of these network services where not available at the time of BIN Portal development. Future integration with AD is a must have feature.
- Integration with Grid and cluster environments are objectives for the BING although it was not the focus of this work. There are big problems to solve in Grid integrations and two master's degree works paying attention to this problem [51, 70] , one of them a running in parallel with this work. Future integrations are planed with IGF.
- Data mining support on acquired data. There is a decision to make, if use the heterogeneous storage systems already provided (at the moment a DICOM server) or build a new one with specific data formats.
- Advanced query facilities that allow non specialized querying on the multimodal data in the BING in a google-like fashion.

7. References

1. Turner, R. and T. Jones, *Techniques for imaging neuroscience.* Br Med Bull., 2003. **65**: p. 3-20.
2. Duncan, J., *Neuroimaging of epilepsy*. University of Munich Epilepsy Course. 2003, Munich, Germany: University of Munich.
3. Cascino, G.D., *Advances in neuroimaging: surgical localization.* Epilepsia., 2001. **42**(1): p. 3-12.
4. Koepp, M.J. and F.G. Woermann, *Imaging structure and function in refractory focal epilepsy.* Lancet Neurol, 2005. **4**(1): p. 42-53.
5. Pedrosa, M., et al., *A System Architecture for the Bing: Brain Image Network Grid*, in *HealthInf - Biostec 2009*. 2009: Porto.
6. Oliveira, I., et al. *The Portuguese BING Network: Towards a Brain Imaging Grid Virtual Community* in *MICCAI-Grid*. 2009. London, UK.
7. Eysenbach, G., *What is e-health?* Journal of Medical Internet Research, 2001.
8. GridTalk, *The Future of Healthcare: eHealth and Grid Computing*. GridBriefings, 2009.
9. SalesForce. *Welcome to AppExchange*. 2009 [cited; Available from: http://sites.force.com/appexchange/gettingStarted.
10. Cunha, J.P.S., et al. *BING: The Portuguese Brain Imaging Network GRID*. in *IberGRID* 2007. Santiago de Compostela.
11. SARS. *SARS Software Products*. 2008 [cited; Available from: http://www.sarsgrid.com/.
12. NCHP. *National Center for Hight-Performance Computing*. 2009

[cited; Available from: http://www.nchc.org.tw/en/.
13. newsVOAcom, *Doctors Hope Advances in SARS Research Will Help Future Epidemics*. 2005.
14. PRAGMA. *Pacific Rim Applications and Grid Middleware Assembly*. 2006 [cited; Available from: http://www.pragma-grid.net/.
15. Gfarm. *Gfarm file system*. 2007 [cited; Available from: http://datafarm.apgrid.org/.
16. SCMSWeb. *SCMSWeb PRAGMA monitoring site*. 2009 [cited; Available from: http://pragma-goc.rocksclusters.org/scmsweb/.
17. Lee Bu-Sung, F. *A Multi-Organizational Grid Accounting System*. 2009 [cited; Available from: http://www.ngp.org.sg/applications/Contents/IM07-MOGAS.pdf.
18. von Eschenbach A., B.K., *Cancer Informatics Vision: caBIG™*. Cancer Informatics, 2006.
19. NCICB. *National Cancer Institute Center for Bioinformatics* 2001 [cited; Available from: http://ncicb.nci.nih.gov/.
20. Saltz, J., et al., *caGrid: design and implementation of the core architecture of the cancer biomedical informatics grid*. Bioinformatics, 2006. **22**(15): p. 1910-1916.
21. Geraghty, D.E., et al., *caBIG CDE/V Kickoff Meeting Presentation Fred Hutchinson Cancer Research Center.* 2004.
22. VL-e. *Virtual Laboratory for e-Science site*. 2004 [cited; Available from: http://www.vl-e.nl/frame_home.htm.
23. VL-e. *Proof-of-Concept Environment (PoC)*. 2007 [cited; Available from: http://poc.vl-e.nl/.
24. Nikhef. *National Institute for Subatomic Physics site*. 2008 [cited; Available from: http://www.nikhef.nl/.
25. SARA. *Computing and Networking Services*. 2009 [cited; Available from: http://www.sara.nl/index_eng.html.
26. VL-e. *VL-e enables new approaches to traditional sciences*. 2007 [cited; Available from: http://www.vl-e.nl/outreach/eChallenges_2007/Flyers/flyer-VLe-SP1.4-complete.pdf.
27. Maheshwari, K.C., et al., *Problem Solving Environment for Medical Image Analysis*. 20th IEEE International Symposium on Computer-Based Medical Systems, 2007.
28. Olabarriaga, S.D., et al. *From 'low-hanging' to 'user-ready': initial steps into a healthgrid*. in *Healthgrid*. 2008. Chicago, USA.
29. VL-e. *VBrowser Overview*. [cited; Available from: http://staff.science.uva.nl/~ptdeboer/vlet/page_vbrowser.html.

30. VL-e. *VBrowser (VL-e Toolkit)*. 2007 [cited; Available from: http://staff.science.uva.nl/~silvia/vlemed/oldmeetings/2007-10-09-vbrowser.pdf.
31. Keator, D.B., et al., *A national human neuroimaging collaboratory enabled by the Biomedical Informatics Research Network (BIRN)*. 2008: p. 162-172.
32. BIRN-CC. *BIRN Coordinating Center site*. 2001 [cited; Available from: http://nbirn.net/research/bcc/index.shtm.
33. NSF. *National Science Foundation Middleware Initiative*. 2004 [cited; Available from: http://www.nsf.gov/pubs/2004/nsf04555/nsf04555.htm.
34. Globus. *Globus Alliance community* [cited; Available from: http://www.globus.org/.
35. Frey, J., et al., *Condor-G: A Computation Management Agent forMulti-Institutional Grids*. Cluster Computing, 2002.
36. Peltier, S. *The BIRN Portal: Leveraging Telescience and the NMI*. 2002 [cited; Available from: http://bcc-dev-web.nbirn.net/press/archive/ahm_2002/ppts/peltier_ahm2002.ppt.
37. Rajasekar, A., et al., *Storage Resource Broker – Managing Distributed Data in a Grid*. Computer Society of India Journal, 2003.
38. ImageJ. *Image Processing and Analysis in Java (software site)*. [cited; Available from: http://rsbweb.nih.gov/ij/.
39. Slider, D. *3D Slider (software site)*. [cited; Available from: http://www.slicer.org/.
40. NetApp, *Network Appliance NetApp FAS3040*. 2008.
41. EGEE. *Enabling Grids for E-sciencE site*. 2006 [cited; Available from: http://www.eu-egee.org/.
42. Smith, S.M., et al., *Advances in functional and structural MR image analysis and implementation as FSL*. Neuroimage, 2004. **23 Suppl 1**: p. S208-19.
43. Friston, K.J., *Statistical parametric mapping and other analysis of functional imaging data. In Brain Mapping: The Methods*. 1996, Academic Press. p. 363-385.
44. NEMA, N.E.M.A., *Digital Imaging and Communications in Medicine (DICOM):* . 1996, Rosslyn, Va: National Electrical Manufacturers Association.
45. Evans, D. *A Very Basic DICOM Introduction*. 2009 [cited; Available from: http://www.dcm4che.org/confluence/display/d2/A+Very+Basic+DICOM+Introduction.

46. Garrett, J.J. *Ajax: A New Approach to Web Applications*. 2005 [cited 2009 November]; February 18, 2005:[Available from: http://www.adaptivepath.com/ideas/essays/archives/000385.php.
47. W3C. *Document Object Model (DOM)*. 2005 [cited 2009 November]; Available from: http://www.w3.org/DOM/.
48. ICEFaces. *Incremental, In-place Page Updates*. 2009 [cited; Available from: http://www.icefaces.org/docs/v1_8_0/htmlguide/devguide/keyConcepts3.html.
49. W3C. *Cascading Style Sheets*. 2009 [cited 2009 November]; Available from: http://www.w3.org/Style/CSS/.
50. Grenyer, P., *Model View Controller.* Marauder Consulting Ltd, 2008.
51. Pacheco, D.J.G., *Science 2.0: Sharing Scientific Data on the Grid*, in *Departamento de Electrónica, Telecomunicações e Informática*. 2009, Universidade de Aveiro: Aveiro.
52. Pacheco, D., et al., *MAGI: A medical application Grid interfacing portal for eScience*, in *3rd Iberian Grid Infrastructure Conference (IberGrid'09)*, V.H. Garcia, et al., Editors. 2009, NetBiblo S.L.: La Coruña: Valencia, Spain. p. 31-42.
53. ZK Direct Ria. *ZK*. [cited 2009; Available from: http://www.zkoss.org/.
54. Gamma, E., et al., *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional Computing Series 1995: Addison-Wesley Professional.
55. Johnson, R.E. and B. Foote, *Designing Reusable Classes.* Journal of Object-Oriented Programming, 1988. **1**(2): p. 22-35.
56. Martin, R.C., *The Dependency Inversion Principle.* Object Mentor, 1996.
57. Ambler, S., *Agile Database Techniques: Effective Strategies for the Agile Software Developer*. 2003: Wiley.
58. PostgreSQL. *PostgreSQL Global Development Group site*. [cited; Available from: http://www.postgresql.org/.
59. DataNucleus. *JPOX heterogenous persistence solution for Java*. [cited; Available from: http://www.jpox.org/docs/index.html.
60. Apache. *Which Persistence Specification, JDO vs JPA*. [cited; Available from: http://db.apache.org/jdo/jdo_v_jpa.html.
61. UCLA. *Java 1.4+ DICOM ImageIO SPI*. 2004 [cited; Available from: http://www.rip.ucla.edu/~shah/dicom/.
62. Microsoft, *MSDN, Data Platform Developer Center, Microsoft Open*

Database Connectivity (ODBC). 2009.
63. Microsoft, *MSDN, Microsoft patterns & practices, Publish/Subscribe.* 2009.
64. ZK. *ZL live demo*. 2009 [cited; Available from: http://www.zkoss.org/zkdemo/userguide/#f1.
65. Javalobby. *Hibernate: Understanding Lazy Fetching*. 2005 [cited; Available from: http://www.javalobby.org/java/forums/t20533.html.
66. Berners-Lee, T., R. Fielding, and L. Masinter, *Uniform Resource Identifier (URI): Generic Syntax*. 2005, Network Working Group, Internet Engineering Task Force
67. Microsoft, *Windows Server 2003 Active Directory.* 2003.
68. JBoss. *RichFaces site*. 2009 [cited; Available from: http://www.jboss.org/richfaces.
69. ICEFaces. *ICEFaces site*. 2009 [cited; Available from: http://www.icefaces.org/main/home/.
70. Andrade, R., et al. *Multi-voxel Non-linear fMRI Analysis: A Grid Computing Approach*. in *IberGRID* 2007. Santiago de Compostela.