Grant agreement no.211714

neuGrid

A GRID-BASED e-INFRASTRUCTURE FOR DATA ARCHIVING/ COMMUNICATION AND COMPUTATIONALLY INTENSIVE APPLICATIONS IN THE MEDICAL SCIENCES

Combination of Collaborative Project and Coordination and Support Action

Objective INFRA-2007-1.2.2 - Deployment of e-Infrastructures for scientific communities

Deliverable reference number and title: **D6.1 Distributed Medical Services Provision (Portal Service)**

Due date of deliverable: **Month 12**

Actual submission date: **31st January 2009**

Start date of project: **February 1st 2008** Duration: **36 months**

Organisation name of lead contractor for this deliverable: **University of the West of England, Bristol UK**

Revision: Version **1**

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Contents

Intended Recipients

The WP6 workpackage entitled **"Distributed Medical Services Provision"** aims to design a group of *generic* services that can be used in a number of related medical applications. These will then be implemented in order to fulfil the neuGrid specific project requirements. The services will be built according to the design philosophy presented in the WP6 deliverable. This will help to enhance and promote their re-usability in other related applications.

This deliverable document presents a design philosophy that the generic services will follow, maps user requirements against suitable services and briefly presents a list of the services. An initial implementation of the services and their detailed API descriptions will be delivered in the year 2 deliverable.

The WP leaders, technical users and neuGrid developers are the intended recipients of this document. To a lesser extent, since indirectly concerned (through the natural abstraction of Workflow/ Pipeline authoring environments such as the ones proposed in WP6), neuro-scientists and prospective users (e.g. Pharmaceutical companies) as well as internal and external reviewers of the project activities, are anticipated as potential readers of this document.

10. The Portal Service

10.1 Introduction

The Portal Service will be the single point of entry for users to access the neuGrid services. It will hide the complexity of the underlying low-level neuGrid architecture from the users and will enable them to focus on using the services' functionality. It will allow the community users to easily authenticate, access the services, browse the data, launch analysis and visualize their results. The Portal Service is designed to provide access to other services in the neuGrid infrastructure using simple interfaces. It will integrate all of them into one single User Interface, hiding all the architecture internals and calling individual Web Services according to the needs of the users and the application workflow.

The Portal Service requirements (which have been collected in work package 9) state the following important functionality to be addressed in the portal service:

- It should have an easy to use interface
- It should have a flexible interface that could be easily customized and reused
- It should be the single point of contact to access the underlying services that may be further composed with the different low level services

The Portal Service will help to:

- Authenticate users
- Access services
- Browse data: access and query data and intermediate/final results
- Launch analyses
- Visualize the produced results
- Track and manage workflows
- Submit Jobs and carry out related steering and monitoring
- Manage users/groups/rights/security through respective services and well-defined ACLs
- Manage services (registration and removal)
- Allow access to both Grid and non-Grid services alike

10.2 Architectural Models and Choices

There are three possible architectures for the Portal Service:

- a JSR 168 compliant portal/portlet architecture [69]
- a Java rich application (e.g. JNLP)
- a Web SSO [70] for existing applications plus an AJAX [71] based portal for new services

JSR 168, the Java Specification Requests 168, is created to enable interoperability between Portlets and Portals. A JSR 168 compliant portal is a site that manages and aggregates portlets

(pluggable user interface software components). Each portlet will produce fragments of markup code that will be incorporated into one single page. A portal is responsible for handling the inter-portlet communication and other things like the session management. A rich Java application such as JNLP is a desktop application, that could be linked to other applications and/or call some remote functions. JNLP (Java Network Launch Protocol) is an XML-based protocol that can be used to deploy Java and JavaFX applications on the Internet.

A Web Single Sign On (SSO) system is a mechanism that allows a user to share its authentication between different web applications. One account is used to access all the web applications and one single login is required in order to access all the websites. AJAX (Asynchronous JavaScript and XML) is a set of technologies allowing users to create rich Internet applications with significant interactivity. It allows users to send requests to the server and helps in updating different parts of the page independently. A wide variety of server side technologies such as PHP, JSP, ASP etc could be used to implement a portal using AJAX techniques. One main advantage of a Web Portal based solution is that it does not require the client to install a specific application, a web browser is sufficient. A desktop application requires the client to deploy software on a deskptop, but it allows things that are hardly possible when using a Web Portal.

10.3 Description and Justification of the proposed architecture

Since the neuGrid architecture is being proposed and will be implemented incrementally in the following years, the use of an existing grid-oriented portal could add its own complexity and constraints. There are number of interfaces that can enable services access through a portal. The following section presents a short status of existing interfaces:

- LORIS has an existing infrastructure, with a PHP-based front-end. It also looks like that it will not be easy to switch LORIS to a Java Portlet infrastructure.

- Security management: multiple interfaces are already available: 1) Web 2) API AJAX.

- Kepler and LONI have been short listed as the contenders for the workflow authoring environments in the pipeline service.

- Collaboration and Statistical tools: HealthGrid has already developed a similar application to allow users to search and download traces in the Grid Observatory project [72]. We intend to use these developments due to their flexibility and ease of use.

There are some components already available in different portal implementations for some of the functionality required in the neuGrid project and it seems obvious that an effort to reuse to the maximum existing GUIs will allow neuGrid partners to save time and effort. It will help us to capitalize on the effort that has already been invested in the development of these components. As all these components are quite heterogeneous, and as the portal requires a common authentication and security scheme, the principle of using a Web SSO (Single Sign On) facility is quite natural. Another necessary feature is trying to establish the most possible coherent and user-friendly 'look and feel' between the different neuGrid applications. This can be achieved by harmonizing both the navigation facilities and the overall colour scheme. Finally, as some of the neuGrid features cannot be exposed through a web interface, they should be made available through a custom portal (see figure 59). This portal will be using AJAX to aggregate the different functionalities offered by the neuGrid Web Services.

10.4 Components, API and Interfaces

The existing web interfaces can be made reusable through a portal architecture. The Web Services that are not already accessible via a Web Interface will be exposed through an AJAX-based portal. All the web interfaces, including the AJAX portal will have their users and logins unified using the SSO facility.

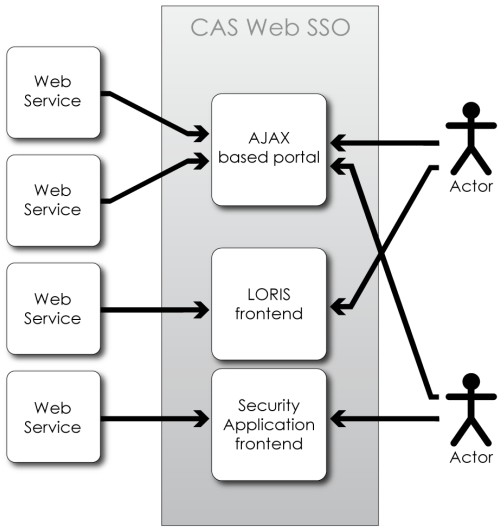

Figure 59: The Portal service architecture

10.4.1 SSO facility

One of the most mature and widely used SSO technologies is the Central Authentication Server formerly known as CAS [73]. CAS is an authentication system originally created by Yale University to provide a trusted way for an application to authenticate a user. Some efforts have been made into the evaluation of CAS and its integration into portals for providing the necessary security for neuGrid. CAS also offers a framework allowing to set up an identity service, centralizing all the information of the user accounts and allowing every site member of the SSO to request them for their local use, so the user's profiles can now be coherent and synchronized between all applications (see figure 60).

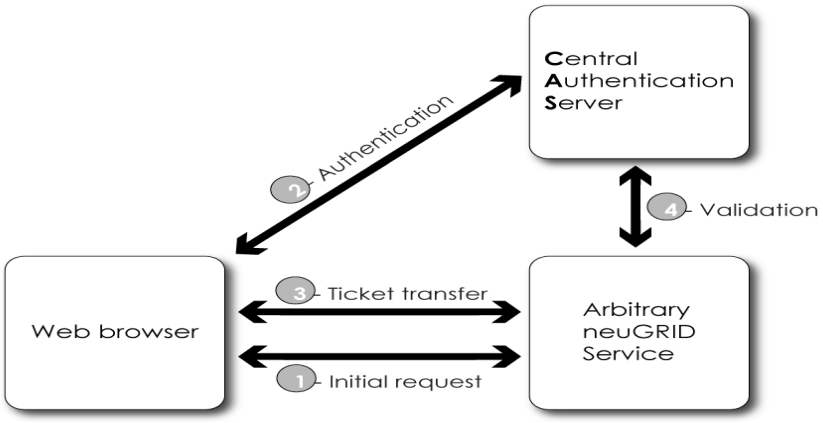

Figure 60: CAS workflow

An alternative to the CAS Single Sign On service could be The Open Web SSO project (OpenSSO) [74]. The Open Web SSO project provides core identity services to simplify the implementation of transparent single sign-on (SSO) as a security component in a network infrastructure. OpenSSO (see figure 61) provides the foundation for integrating diverse web applications that might typically operate against a disparate set of identity repositories and are hosted on a variety of platforms such as web and application servers. This project is based on the code base of Java System Access Manager, an identity infrastructure application offered by Sun.

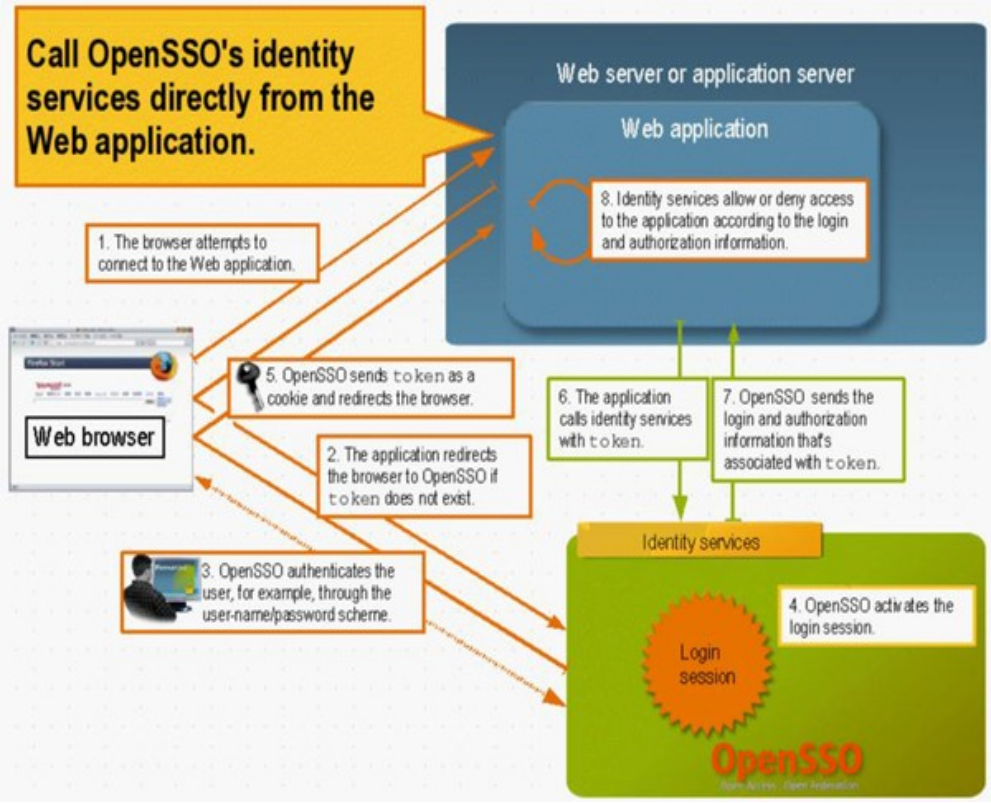

Figure 61: OpenSSO architecture (Integrating Applications with OpenSSO) [75]

Another alternative to provide Single Sign On functionality could be Shibboleth [76]. The Shibboleth System (see figure 62) is a standards based, open source software package for web based single sign-on across or within organizational boundaries. It allows sites to make informed authorization decisions for providing access to protected on-line resources in a privacy-preserving manner.

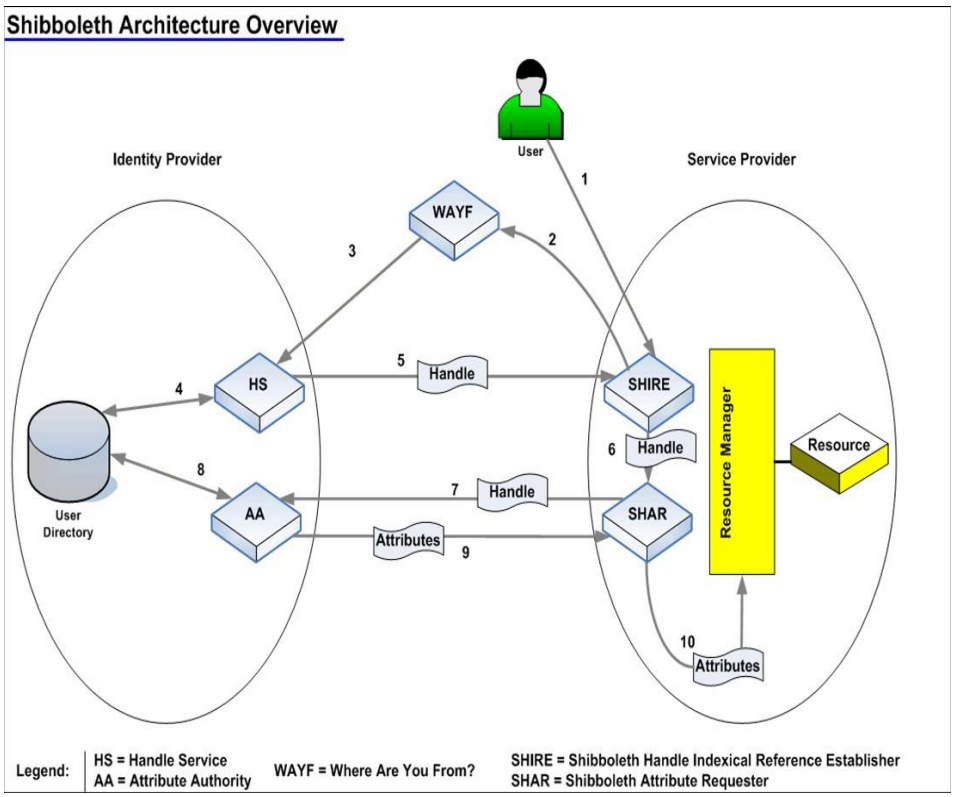

Figure 62: Shibboleth architecture [77]

10.5 Portal technologies

The following are widely available portal technologies. Most of these technologies have JSR 168 compliant implementations.

- Apache portlet [78] and Apache JetSpeed [79]
- P-Grade Portal [80]
- GridSphere portal framework [81]
- UC Grid Portal [82]
- GridPort [83]
- Health-e-Child gateway and it's AJAX API [84]

In the following sections, an analysis of the available portal technologies has been presented.

10.5.1 JSR168 compliant portals

Pluto is the reference implementation of the Java Portlet Specification. The current version (2.0) of this specification is JSR-286. Pluto serves as a portlet container that implements the Portlet API and offers developers a working example platform from which they can test their portlets. Pluto's simple portal component is built only on the portlet container and the JSR 168's requirements. In contrast, the more sophisticated, Jetspeed project concentrates on the portal itself rather than the portlet container, and considers requirements from other groups. Jetspeed-2 is an enterprise portal made available by Apache. Jetspeed-2 offers several

architectural enhancements and improvements over Jetspeed 1.0. First, Jetspeed-2 is conformant to the Java Portlet Standard and provides a standard mechanism for the deployment of portlets. Second, Jetspeed-2 has matured to a more scalable architecture featuring multi-threaded functionality. Third, Jetspeed-2 is decoupled from several legacy open source projects. Fourth, Jetspeed-2 is based on a component architecture. Working with Apache Pluto and Jetspeed-2 will allow users to freely develop something that exactly fits their needs since they are just generic portlets containers. This will not restrict us to any specific/restricted grid-oriented portlets, which also means that we may have to develop functionality from scratch.

The GridSphere portal framework provides an open-source portlet based Web portal. GridSphere enables developers to quickly develop and package third-party portlet web applications that can be run and administered within the GridSphere portlet container. GridSphere integrates portlets for authentication and basic roles management. Using the Vine Toolkit [85], some portlets can be enabled to interact with many grid middleware such as gLite,Globus Toolkit, OGSA-DAI, UNICORE etc. From our experience GridSphere design customization is quite annoying and inefficient, and as we will develop the middleware abstraction layer in neuGrid project, the GridSphere specific features will be hard to customize and use for the project needs.

The GridPort Toolkit enables the rapid development of highly functional grid portals that simplify the use of underlying grid services for the end-user. It comprises a set of portlet interfaces and services in the portal layer that provides access to a wide range of backend grid and information services provided by lower-level grid technologies. Portlets expose the backend services via customizable web interfaces in order to enable personalization of grid portal user interfaces. Portal services support the portlets inside the portal layer by augmenting their capabilities in an extensible and reusable way while tying the portlets together in order to make them more cohesive. GridPort has recently become part of the Open Grid Computing Environments (OGCE) portal software project. As for GridSphere, the abstraction capabilities of GridPort will not be needed, and the use of this tool will force us to conform to the architecture used by GridPort.

10.5.2 Non JSR168 compliant portals

The P-GRADE Grid Portal is a workflow-oriented Grid portal that enables the creation, execution and monitoring of workflows in grid environments through high-level, graphical Web interfaces. Components of the workflows can be sequential and parallel MPI jobs. The P-GRADE Grid Portal hides the low-level details of Grid access mechanisms by providing a high-level Grid user interface that can be used in Globus, LCG-2 and gLite Grids. Workflows developed in the P-GRADE portal are portable between different Grids without any re-learning of the new Grid system or re-engineering of the application. P-GRADE portal can be configured to access several Grids and if a user has valid certificates for several Grids, then he/she can exploit all those Grids simultaneously during the execution of his/her workflow. Like the previous grid portals, as it is intended to directly interact with the grid, it seems to bypass the neuGrid SOA infrastructure but its Grid abstraction capabilities and design could be helpful.

The UC Grid Portal provides a single web interface to those computational clusters that have joined the Campus Grids at the various University of California campuses. Additionally, the UC Grid Portal can directly access some clusters outside of the UC Grid, including clusters on the TeraGrid. It is mainly designed to access a predefined set of American clusters. The UC Grid portal is itself based on GridSphere, it could add some useful functionality however another layer of complexity and rigidity is required to achieve this.

10.5.3 Justification of the technological choices

For the SSO implementation, CAS seems to be the technology that will best suit our needs. Writing authentication code to integrate applications into the SSO is quite simple and there is already in the consortium some knowledge on how to do it. CAS is a well featured and mature system which will simplify the authentication process for users. As an initial test, implementing an SSO facility should be the quickest and easiest choice, but in a second phase an identity service providing access to all the attributes of an account, (for example not only having a common authentication but also a common synchronized user account), could be implemented.

For the portal itself, as we will use our own middleware abstraction technologies, all the existing grid-aware portals are not useful and using such tools would only add more implementation constraints. Therefore the choice of a portal based on an AJAX API allowing for ease interoperation with Web Services seems to be the more flexible and lightweight solution. Flexibility will be key in addressing the requirements of users and will allow new services to be seamlessly integrated within the neuGrid web-based interface. Regarding the common look and feel, the navigation experience should be the same on every page of the portal, so a common navigation menu should be created. This could be achieved using some JavaScript code and eventually using a technique allowing to embed a site into one another, ideally the navigation menu should be hidden when not in use to maximize the space available for the application. The menu should give a quick access to the login box and to the different neuGrid applications. Some contextual information could also be displayed in a corner of the screen. Studies on such functions, which will inform our technology choices, will be conducted during the next phase of project development as we move from system design into implementation.

Then, for the harmonization of the designs, the existing applications seem to be rather too different to be integrated seamlessly. The full redesign of these applications could however lead to significant overheads; the harmonization of the colour schemes seems to be a good compromise and should be do-able quite easily since nowadays web sites are using CSS stylesheets to centralize the design. For this, a neuGrid-wide CSS sheet could be a good base, which all web applications could extend to suit their needs. It is clear that care needs to be taken during the design of all user interfaces and the portal in general so that users find them to be well integrated and intuitive to use. An evaluation of this will also take place in the next phase of the project.

10.6 Conclusions

Due to the very specialized and original architecture of the neuGrid system, the usage of a Grid-aware portal would probably only add constraints making it difficult to implement the Service Oriented Architecture that has been identified by the consortium as a means of constructing the infrastructure. Since Grid-aware portals are used to integrate all components into their own specific architecture, an implementation of this kind of framework in neuGrid would not be possible without sticking to designs which are not as flexible as the one required in neuGrid, particularly in terms of technical abstractions and decoupling. Therefore it seems reasonable to consider more general portal environments and there are a wide range of such technologies. Clearly the widely used AJAX platform has a range of features that makes it well suited to integration with web-services. This has led to its adoption as a candidate technology for providing the user facing aspects of neuGrid.

From the desktop application standpoint, the need to reuse what already exists (namely specialized and well designed Web interfaces) and the requirement to create an easily upgradable architecture are points, which certainly require further evaluation. So, from all these current evaluations and analysis it seems that the most flexible and simple approach and therefore the most reliable is the federation of all the web applications using a Single Sign On facility. This will provide the users with a comprehensive and harmonized interface. The creation of a dedicated portal aggregating service, without an overly restrictive web interface,

will enable users to add the missing building blocks to the portal. This will allow neuGrid to offer an harmonized and federated set of specialized and dedicated interfaces to the users. It is reasoned that this addresses the requirements of users and will deliver a satisfactory user experience, thereby encouraging the wider adoption of the neuGrid platform.

10.7 References

[69] JSR 168: Portlet Specification http://www.jcp.org/en/jsr/detail?id=168

[70] Single Sign On (SSO) http://en.wikipedia.org/wiki/Single_sign-on

[71] AJAX http://en.wikipedia.org/wiki/AJAX

[72] Grid Observatory http://query.grid-observatory.org

[73] Central Authentication Server (CAS) http://www.ja-sig.org/products/cas/

[74] OpenSSO https://opensso.dev.java.net/

[75] Integrating Applications With OpennSSO http://developers.sun.com/identity/reference/techart/app-integration.html

[76] Shibboleth http://shibboleth.internet2.edu/

[77] Single Sign On using Shibboleth http://ict-strategic-thinking.pbwiki.com/Single-Sign-On

[78] Apache Pluto http://portals.apache.org/pluto/

[79] Apache JetSpeed http://portals.apache.org/jetspeed-2/

[80] P-Grade Portal http://www.portal.p-grade.hu/

[81] GridSphere portal framework http://www.gridsphere.org
[82] UC Grid Portal https://portal.ucgrid.org:9443/gridsphere/gridsphere
[83] GridPort http://www.tacc.utexas.edu/projects/gridport.php
[84] Health-e-Child http://www.health-e-child.org/
[85] The Vine Toolkit http://www.gridsphere.org/gridsphere/gridsphere/guest/vine/r/