HPC Portal: a Repository of Portal Resources

Xiao Dong Wang and Rob Allan

STFC, Daresbury Laboratory, e-Science Centre, Warrington WA4 4AD, UK
x.d.wang@dl.ac.uk, r.j.allan@dl.ac.uk

Abstract

As computational science is developing, multiple Grid resources are being used widely by scientists and engineers in their work. A Grid portal is an environment and a Web-based application which should be familiar to anyone who has used a Web browser. The growth of Grid applications brings a challenge of how to develop a successful Grid portal. HPC Portal is a generic Grid portal development project to provide not only a customized portal framework, but also a range of Grid portlets and tools. End users can adopt HPC Portal directly as their production portal or do further customization and also enrich the existing portal functions by deploying portlets and tools additional to those provided by HPC Portal.

1. Introduction

As computational science is developing, multiple Grid resources are being used widely by scientists and engineers in their work [1, 2, 3]. A Grid portal is an environment and a Web-based application which should be familiar to anyone who has used a Web browser. So the Grid portal acts as a "science gateway" between users and a range of such distributed Grid resources [21, 35]. In this way it is possible to hide the complexity of accessing Grid resources from end users, enabling them to concentrate on the enhanced science which they can achieve. The portal provides personalization, Single Sign-On, resource aggregation and customization features in addition to the usual powerful Grid functionalities, such as job management and data management. Current portal research is now looking to provide even more powerful functionality using workflow together with service registries and to adopt Web 2.0 technologies [31, 32, 34].

Based on the services provided from distributed Grid resources, portals can be constructed for e-Science, e-Business or e-Learning, Grids and Knowledge management, etc. [1]. As a subject-specific example, an e-Social Science portal might focus on providing data and information for social science researchers or students. In industry, most companies focus initially on building internal corporate (enterprise) portals that give employees and business users a personalized view of information on the corporate intranet. With the growth of using e-Business, such portals are developed to enable companies extending portal access to external trading partners, suppliers and clients which help to improve business relationships and communications. E-Learning portals provide an integrated interface to learning and training resources which include Web-based training, virtual classrooms and digital collaboration. The e-Learning portals are used widely in higher education and the students and staff are end-users with different roles.

The growth and complexity of Grid applications brings a challenge of how to develop a successful Grid portal. There is growing interest from commercial software vendors as well as open-source software developers. On the market, some commercial portals support Grid resource management and application, for example WebSphere Portal [4], BEA WebLogic Portal [5], Sun One Portal [6], MOAB Access Portal [8], etc. Among the open-source portal frameworks, GridSphere [7] is popular to support a range of pre-built Grid portlets which can access distributed remote resources via the Globus Toolkit middleware. uPortal [11] is the most widely deployed open-source framework as an institutional portal and adopted by many universities for both e-Learning and on-line administration purposes. These portals are provided as portal frameworks. The portal developers used to make some extra effort, for example portal customization, when a portal framework is selected as development platform.

Two key points to the success of portal development are the guarantee to users that their information will remain secure and the developed Grid tools can be reusable. For system administrators it is equally important to know that the portal tools are well tested and will not cause problems in submitting jobs to or managing data on their Grid resources. HPC

Portal is a generic Grid portal development project which tries to meet the above points. This project provides not only a customized portal framework, but also a range of standard tools. End users can adopt HPC Portal directly as their production portal or do further customization based on HPC Portal. HPC Portal uses the latest platform-independent Java and Web Services technologies, such as the JSR-168 portlet and WSRP standards [24, 25]. As a repository of portlets, a subset of HPC Portal can be "cloned" to support other projects and the generic tools customized or extended for specific science applications.

This paper first points out the motivation of HPC Portal, then describes its GUI customization. We next discuss the security customization which includes end-users' authentication mechanisms and Grid resource authorization based on role management. As a portlet repository, the range of portlets and tools currently developed with HPC Portal is introduced and the portlets package is released for the users to adopt. Future work and conclusions are outlined at the end.

2. Motivation of HPC Portal

2.1 Motivation of HPC Portal

With the emergence of more and more projects wishing to adopt Grid portals for their applications, there is a high requirement to reduce development burden from portal developers and avoid duplication of effort. The motivation of HPC Portal is to provide a customized portal framework with a suite of pre-built and tested portlet tools. This portal framework should support a range of authentication mechanisms and implement Single Sign-On and Grid resources authorization management. The set of tools in HPC Portal are provided as JSR 168 standard compliant portlets which are platform independent and can be made "plug-and-play". This set of portlets can be released freely as open source. Developers of a new portal for an e-Science project may then get the portlets package and deploy them in their portal framework of choice and can then continue to enrich their portal's functions.

The HPC Portal is composed of four layers: interface layer; secure layer; tool layer and resource layer. The architecture is shown in Figure 1. Interface layer sends/gets information in HTML protocol to end users who usually can access via a Web browser. This layer is implemented and supported by the portal framework. Secure layer was developed for HPC portal. It provides three authentication methods: login locally; login via MyProxy and login via Shibboleth. UDDI is adopted to publish MyProxy server handle. Section 5 will introduce the secure layer in detail. Tool layer consists a range of portlets which we have developed. These portlets are allocated in categories as authentication, Grid computing, Grid resource information, data/file management, entertainment, date and event. Database and portlet session are adopted to store credential and shared data among portlets. Resource layer includes distributed Web resources that are called by portlets.

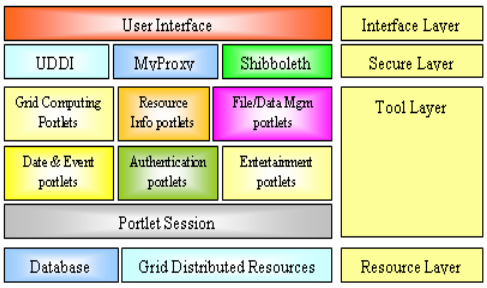

Figure 1: HPC Portal Architecture

2.2 HPC Portal Platform

For our development work on HPC Portal we have adopted uPortal 2.5.3, which is the latest stable version as a portal development framework. uPortal is popular for institutional administration services and for e-Learning. It is developed by the Java Architectures Special Interest Group (JA-SIG) as an open source portal framework [9] and currently has some 800 installations worldwide (approximately half supported by SunGard SCT for their Lumenis e-Learning product [10]). uPortal is stable and supports the JSR 168 portlet standard via the integration of Pluto [11]. Currently, uPortal is considered to be the open source portal framework which supports the maximum number of portals ranging from Java portals to HTML portals, text portals to XML portals [12]. Many of these were developed in the institutions which have adopted uPortal as channel-based tools which have now been converted to portlets. Our experience [12, 23] shows that uPortal has excellent JSR 168 compliance. The deployment and usage are simple and straightforward; the portal administration is easy and powerful.

3. GUI Customization of HPC Portal

uPortal adopts a standards-based approach to design its "look and feel". It processes the visualization of content in multi-stages. These stages include layout, structure transform and theme transform. The final output stage is rendering by the end user's browser. All these stages are configured via XML documents. XSL transformations [33] are used at each stage to parse the documents and implement the configuration. It is therefore possible to customize at each stage in a very flexible way. For example, the file of org.jasig.portal.layout.AL_TabColumn.integratedModes.integratedModes.xsl defines the GUI display and GUI component arrangement. The component and portlet color can be called from the two files called immII.css and immII_portlet.css. Figure 2 shows the customized main page of HPC Portal. It is clearly possible for any project using HPC Portal to do their own customization.

Figure 2: Main page of HPC Portal

4. Security Customization of HPC Portal

Security is an import issue in the portal system. It is necessary to ensure not only no leaking of information about registered users and running jobs and data, but also to protect the portal resources from compromises. Grid portal security should include two parts: authentication and authorization.

4.1 HPC Portal Authentication

HPC Portal verifies users' identities in a three stage mechanism: login locally; login via MyProxy; and login via Shibboleth.

4.1.1 Login Locally

e-Science Grid resources in the UK are deployed for use by the UK national academic users and partners. All users in the UK with certificates issued by the UK e-Science Certification Authority [13] can therefore be treated as valid users for the e-Science portals. These users should be able to login to the HPC Portal automatically. The HPC Portal login module has therefore been customized so that those users can login and process jobs in their personal pages. This is based on the fact that a user's certificate can be treated as a proxy for the user and the identity retrieved from the user's proxy credential is unique. Our scenario is that HPC Portal will use the user's identity to map to the portal's user account. Thus only the user with the identity typed in and matched to a password can be authenticated to access portal resources. This method can assure that users are able to login to the portal and do their jobs even if for some reason the MyProxy server can not be accessed provided the proxy lifetime of a credential stored in the portal local database is still valid since last time the user logged.

4.1.2 Login via MyProxy

A Grid proxy credential needs to be delivered to portal's Grid resources as the user authentication before accessing any portlet. HPC Portal customizes the uPortal login module by redirecting to a MyProxy server and retrieves the user's proxy credential from that server. With a hierarchical CA now in place, we have to consider that additional MyProxy servers may be deployed by some institutions and users should be provided options to upload/ retrieve credentials to/ from those local MyProxy servers. This may for instance be done for Grid training purposes. A central registry needs to be used to publish the existence of these MyProxy servers.

The Universal Description, Discovery and Integration (UDDI) specification provides a mechanism for publishing web services and for the users to query and find the available services (http://www.uddi.org). A UDDI registry is adopted in HPC Portal and a data model is designed to publish MyProxy server information. HPC Portal customizes the login module to link to the UDDI registry and allows users to select a MyProxy server from the

registry to retrieve their proxy credential. The data model structure is showed as Figure 3. This model includes three types of data: *businessEntity*, *businessService* and *bindingTemplate*. *businessEntity* publishes a business with the name of "MyProxy Server List". All Myproxy servers may be stored in *businessServices* which belong to the *businessEntity*. The MyProxy server handle is presented as the value of *<accessPoint>* element which is the sub-element of *bindingTemplate.*

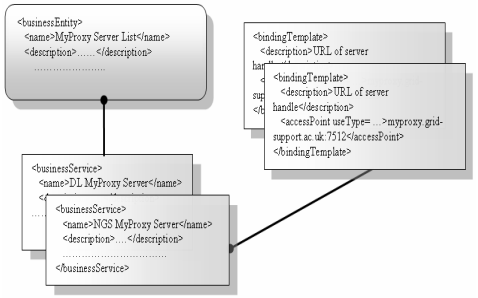

Figure 3: UDDI data model

4.1.3 Login via Shibboleth

The use of MyProxy as a credential repository has made the authentication procedure easier, but users still have to upload their proxy credentials securely into the appropriate MyProxy server before they can access Grid resources. This is a sort of Central Authentication System (CAS). Actually when users log into their machines by typing in a username and password, their identity has been authenticated by their local institution in some way. If this identification can be used as portal authentication, then it will save users from the burden of dealing with X.509 certificates. It also can be thought of as a Distributed Authentication System (DAS). The JISC-funded ShibGrid project, which is Shibboleth based, can implement the above processes [14, 15]. ShibGrid provides a Shibboleth login component as part of a Java Authentication and Authorization Service (JAAS) module which can be plugged into HPC Portal without changes to application level code. The general process is that users will get identification from their local institution when they try to log in to HPC Portal via Shibboleth. The identification attributes are delivered to a ShibGrid adapted MyProxy Server and a proxy certificate will be generated (using the hierarchical CA mechanism) and delegated to HPC Portal.

4.2 HPC Portal Authorization

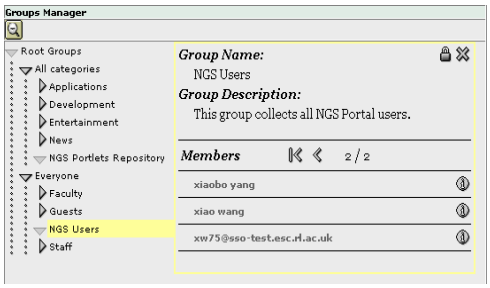

Figure 4: NGS user group

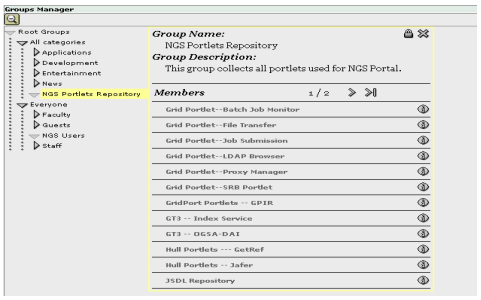

Figure 5: NGS portlets repository group

Portlet Grid resources need to authorize the caller's credential when the resources are invoked. The users who play different roles may get different privileges to access Grid resources. Currently all Grid resources supported by e-Science portals are in principle open to all the UK nationwide e-Science users. As more and more new Grid partners bring resources, license or other issues may arise which restrict access to some particular users. For future extensions, HPC Portal manages the users and Grid resources in two set of groups. That has been done by customizing HPC portal administration mechanism and managed by the portal administrator only. All users are allocated in "User Groups", and Grid resources are implemented in "Resource Groups". Based on the Grid resource (or data or application) license agreement, users with the same license are allocated in one user group, and the licensed Grid resources are deployed in resource group with permission for the user group to access only. For example, two groups have been deployed as "NGS Users" and "NGS Portlets Repository" in NGS Portal. All NGS portal users having the same role are allocated to the "NGS Users" group. NGS Grid resources which are implemented as portlets are put in the "NGS Portlets Repository" group. Figures 4 and 5 show the GUI of NGS Users group and NGS Portlets Repository group separately. As a collection of Grid resources, "NGS Portlets

Repository" group only allows access by users from the "NGS Users" group. Later, more user groups will be launched with different collections of privileges to access resource groups.

4.3 Portlets Communication

When a user accesses a Grid resource via a portlet the logic checks the credential to verify the user's identity and grant authorization if appropriate. Portlets need to be capable of retrieving and sharing the user's proxy credential for the implementation of Single Sign-On. All portlets are deployed in portlet applications separately. The user's credential is put in the portlet session and database. The first time a portlet is called, the proxy credential is retrieved from the database if the credential cannot be found from the portlet session. Thus all portlets have proxy credential communication implemented via database and portlet session. As portlets are designed and packaged separately, the portlet deployment is very straightforward, and it even can be done by copying a portlet war file to the Web application directory in the servlet container, for example, Tomcat.

5. Portlets Package

HPC Portal provides a suite of Grid tools and Grid portlets which are JSR 168 Java standard compliant as shown in Figure 6. The portlets and tools can be delivered as a package named myNGS. The contents of MyNGS are allocated in six categories as authentication, Grid computing, Grid resource information, data/file management, entertainment, date and event.

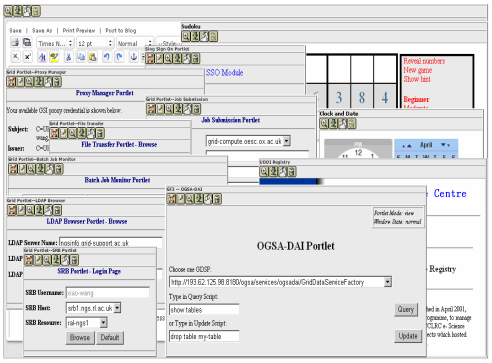

Figure 6: HPC portal portlets

5.1 Authentication

Two portlets are developed in this category. These portlets are Proxy Manager portlet and SSO portlet.

Proxy Manager portlet is designed to retrieve user's proxy credential from MyProxy server. After user types in username, password and lifetime that makes the proxy credential valid for the user's operation in the HPC portal. Proxy Manager portlet will redirect to MyProxy server and retrieve back the proxy credential which was uploaded by the user. The successful credential will be put in local database for use by other portlets.

SSO portlet is developed to implement auto login to another portal. The principle is that when user selects another portal as login destination and runs SSO portlet, SSO portlet then redirects to the selected portal URL with the credential attached. The selected portal will retrieve the user account information from the credential and login if the account is valid. To use SSO portlet, the portal login mechanism should be customized to support login via credential. Currently, SSO portlet works successfully between HPC portal and a portal cloned for testing.

5.2 Grid Computing

Currently this category includes one portlet, Job Submission portlet.

Job Submission portlet can run jobs in distributed machines. Globus Toolkit v2.4 is used as middleware and Java CoG4 is adopted to access the middleware as it has simple usability and powerful functionality [26]. On the Job Submission portlet page, the remote machine can be selected from a pull-down menu and the job running attributes can be input by typing in the text field or loaded from the local machine. When "submit" button is clicked on the portlet, the user's credential is retrieved from database or portlet session and sent to the selected remote machine. After passing the identify verification, the job will run in the remote machine.

5.3 Grid Resource Information

There are three portlets and Web application in this category: Ldap Browser portlet, Batch Job Monitor portlet and UDDI registry client Web application.

Ldap Browser portlet can retrieve distributed system resource information that is pre-stored in an LDAP server, for example, CPU number, whole memory size, available memory, etc. it is very useful for the end-user to find out a suitable machine before the job is submitted.

Batch Job Monitor portlet restores and displays status of all jobs submitted by the client user. The job parameters, which include

username, host name, job manager, executable, arguments, job status and job submitted time, can be displayed in all or part.

UDDI registry client tool is designed as Web application. It supports all the UDDI v2.0 functions. *Business* and *Service* can be searched out by keywords. After subscribing, user can publish his own *Business* and *Service* in the UDDI server. UDDI registry client tool is developed based on UDDI4J client API [27] and jUDDI [28] is deployed as our UDDI server.

5.4 Data/ file Management

We provide four portlets and Web application tool to implement data and file management separately.

OGSA-DAI portlet can access distributed databases managed by OGSA-DAI server. The aim of the OGSA-DAI project is to develop middleware to assist with access and integration of data from separate sources via grid [29]. It provides a unique Web service interface to access distributed database. OGSA-DAI portlet supports a GUI presentation to integrate and simplify OGSA-DAI client operation, for example, after selecting a Grid Data Service Factory (GDSF) that maps to database, end-user can manage the database via querying, updating or deleting data in the database. Currently OGSA-DAI portlet supports OGSA-DAI v3.2.

Storage Resource Broker (SRB) supports shared collections that can be distributed across multiple organizations and heterogeneous storage systems [30]. It manages distributed data as file and end-user doesn't need to care of the file management, for example, where is the file, how to allocate. SRB portlet implements SRB client functions, such as creating directory, uploading/ downloading files and removing files.

File Transfer portlet can manage files hosted in distributed machines. It supports a variety of functions, for example, download file to local machine, upload file to remote machine, transfer file between two remote machines. When end-user finishes the job by calling Job Submission portlet, the job file can be downloaded to the local machine or transfer to the third machine by accessing File Transfer portlet. Our next plan is to integrate SRB portlet and File Transfer portlet.

Online Word Editor, as Web application tool, can edit a word file online by calling Web API powered by Zoho [16]. Currently this tool allows the user to makes use of the feature rich editor of Zoho Services to edit and update content into local servers by calling the Zoho remote API. Later the Zoho storage API will be integrated to implement collaborative online editing by multiple users.

5.5 Date and Event

Clock and Date portlet will display time and date. The code development is based on the Dojo API [18].

A Web application tool of calendar with event map is developed. This tool can publish events by calling the Google calendar API [20]. With the support of Google map, this tool can show the location of the event. This tool can be deployed as portal calendar to publish some common events.

5.6 Entertainment

A sudoku game originally from DHTMLGoodies [17] has been implemented as a test case.

6. Acknowledgements

Funding for HPC Portal development was provided via the CCLRC e-Science Centre (now STFC) and we thank JISC for funding the ShibGrid project.

7. Future Work and Conclusions

Currently HPC Portal tools are cloned to NGS Portal RC1a [19] which also uses the uPortal framework. In future work a HPC Portal instance will be deployed for the North West Grid.

HPC Portal has login mechanisms via MyProxy and via Shibboleth and can retrieve different proxy credential. Currently HPC Portal treats these credentials in different ways and records all working history separately. Our next work will combine these two proxy credentials together. End users can then be mapped to the same account no matter which login mechanism is used.

With the emergence of Web 2.0, portlets with collaborative functions will be required more and more as they will be in widespread usage as remotely hosted tools. A few were illustrated above. HPC Portal will continue to develop these kinds of portlets using innovative technologies.

The HPC Portal project provides a customized portal framework with multi authentication and authorization mechanisms, and the myNGS package with a range of tested portlets and tools. The successful production application of NGS Portal verifies that HPC Portal is suitable as a template of portal

development and myNGS package is capable to provide a rich and powerful suite of portlets and tools for other Grid application projects.

Further work is required to re-deploy myNGS in other portal frameworks. Several discussions have been held with the GridSphere developers and testing of interoperability has been carried out looking at GridSphere, eXo Portal, LifeRay, uPortal and StringBeans. Sakai v2.4 is also currently under investigation as a JSR 168 compliant hosting framework.

8. References

[1] Wang, X.D., Yang, X., Allan, R.: *Top Ten Questions to Design a Successful Grid Portal.* The 2nd International Conference on Semantics, Knowledge and Grid. Nov. 2006, Guilin, China.

[2] Yang, X., Wang, X.D., Allan, R.: *Development of standard-based grid portals, Part1: Review of Grid portals*. http://www-128.ibm.com/developerworks/grid/library/gr-stdsportal1/

[3] Baker, M., Ong, H., Allan, R., Wang, X.D.: *Virtual Research in the UK: Advanced Portal Services*. UK e-Science AHM, Nottingham, 2004, UK(2004).

[4] Websphere portal. http://www-128.ibm.com/developerworks/websphere/zones/portal

[5] BEA WebLogic portal. http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/weblogic/portal

[6] Sun One portal. http://www.sun.com/software/products/portal_srvr/index.xml

[7] MOAB Access Portal http://www.clusterresources.com/pages/products/moab-cluster-suite/access-portal.php

[8] GridSphere portal. http://www.gridsphere.org

[9] uPortal and JA-SIG http://www.ja-sig.org/

[10] SunGard SCT Luminis III platform http://www.sungardhe.com/default.aspx?id=115

[11] uPortal project. http://www.uportal.org

[12] Akram, A., Chohan, D., Wang, X.D., Yang, X., Allan, R.: *A Service Oriented Architecture for Portals using Portlets*. UK e-Science AHM 2005, Nottingham, UK(2005).

[13] NGS Project. http://www.grid-support.ac.uk

[14] ShibGrid Project. http://www.oerc.ox.ac.uk/activities/projects/index.xml.ID=ShibGrid

[15] Shibboleth Project, Internet2. http://shibboleth.internet2.edu/

[16] Zoho project. http://www.zoho.com

[17] DHTMLGoodies project. http://www.dhtmlgoodies.com

[18] Dojo project. http://www.dojotoolkit.org

[19] NGS Portal. https://portal.ngs.ac.uk/NGSPortal

[20] Google calendar. http://www.google.com/googlecalendar/overview.html

[21] Natrajan, A., Nguyen-Tuong, A., Humphrey, M.A., Herrick, M., Clarke, B.P., Grimshaw, A.S.: *The Legion Grid*. Portal. Concurrency Comput. Pract. Exp. 14, No.13-15, 1365-1394 (2002).

[22] Severance, C.: *Sakai and E-Science- Looking Forward*. http://www-personal.umich.edu/~csev/papers/2005/Sakai_E_Science_final.doc

[23] Yang, X., Wang, X. D. Allan, R.: *JSR 168 and WSRP – How Mature are Portal Standards?* WEBIST-2006, 11-13 April 2006 in Setúbal/ Portugal

[24] JSR-168 Portlet Specification, http://www.jcp.org/aboutJava/communityprocess/final/jsr168

[25] Web services for remote portlets specification v1.0, http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf

[26] The Globus Alliance. http://www.globus.org

[27] UDDI4J project. http://uddi4j.sourceforge.net

[28] jUDDI project. http://ws.apache.org/juddi

[29] OGSA-DAI project. http://www.ogsadai.org.uk

[30] SRB project. http://www.sdsc.edu/srb/index.php/Main_Page

[31] Muller, T.: *Creating Web 2.0-Enabled Communities With Sun Java System Portal Server*. Sun Developer Network, 18 April 2007. http://developers.sun.com/portalserver/reference/techart/psandweb2.html

[32] Franklin, T.: *Briefing paper on Web 2.0 technologies for content sharing: Institutional good practice*. http://franklin-consulting.co.uk/LinkedDocuments/Content%20sharing%20Briefing%20paper%20on%20Web%202.pdf

[33] XSL Transformations (XSLT) v1.0 Specification. http://www.w3.org/TR/xslt

[34] Thomas, M.P., Burruss, J., Cinquini, L., Fox, G., Gannon, D., Gilbert, L., von Laszewski, G., Jackson, K., Middleton, D., Moore, R., Pierce, M., Plale, B., Rajasekar, A., Regno, R., Roberts, E., Schissel, D., Seth, A., and Schrosder, W.: *Grid portal architectures for scientific applications*. Journal of Physics: Conference Series 16 (2005) 596-600.

[35] Gannon, D., Alameda, A., Chipara, O., Christie, M., Dukle, V., Fang, L., Farrellee, M., Kandaswamy, G., Kodeboyina, D., Krishnan, S., Moad, C., Pierce, M., Place, B., Rossi, A., Simmhan, Y., Sarangi, A., Slominski, A., Shirasuna, S., and Thomas, T.: *Building Grid Portal Applications From a Web Service Component Architecture*. Proceedings of the IEEE, Vol. 93, No. 3, March 2005, 551-563.