White Rose Grid Portals: Practice and Experience

M Thompson (mart@comp.leeds.ac.uk), J G Schmidt (j.g.schmidt@leeds.ac.uk), P M Dew (dew@comp.leeds.ac.uk)
White Rose Grid at the University of Leeds, Leeds, LS2 9JT

Abstract

This paper reports our experience gained with the development of portals for the White Rose Grid user community over the last three years. During this period we have studied and applied a variety of technologies in an effort to improve access to the White Rose Grid resources from the user's desktop. Such technologies include the Sun Grid Engine Portal, Grid Portal Development Kit software, Struts web application framework and recently the GridSphere portlet container. In particular, we highlight the pros and cons of using these different approaches for portal developments in our environment.

1. Introduction

Despite the considerable efforts recently contributed by the e-Science community towards the development of Grid middleware it is apparent that the Grid uptake among a wider community of traditional application scientists is still slower than anticipated [1]. Grid access through the Globus Toolkit involves a relatively steep learning curve which may be one of the main obstacles to widespread acceptance by users outside e-Science projects. For many users, it would be far more convenient to access Grid resources through a graphical user interface from their desktop, particularly for those application scientists who routinely execute production codes, e.g. parametric simulations. One approach to satisfying this requirement is to construct Grid-enabled portals which can be accessed via the ubiquitous web browser. The experiences reported here add to the many concurrent efforts [3, 4, 6, 11] to develop Grid portals, which the White Rose Grid (WRG) actively pursues [2, 5].

2. Aims and Objectives

The key features of a Grid portal required by our users are as follows:

- an easy-to-use single point gateway to information and computational assets
- to be accessible from anywhere e.g. desktop, laptop or PDA
- to offer secure access with digital certificates
- to provide a comprehensive and dynamic view of all available resources
- to offer the functionality of the creation of job specifications and their submission, transfer of data and files, and graphical presentation of results for analysis

Moreover we aim to:

- engage application scientists in portal developments to best meet their requirements
- minimise the development and long-term maintenance workload on the limited number of web application staff resources within the White Rose Grid
- make use of existing open source software

3. WRG software environment

One of the essential requirements for the portal was that it fully integrates with the current WRG software stack, which is composed largely of open source components. The two main building blocks deployed across the four WRG nodes are: Sun Grid Engine (SGE) Enterprise Edition and the Globus Toolkit (GT) v 2.4.3. The SGE has been selected for the WRG because as a powerful batch processor it also offers resource management control which is essential for maintaining shares within the WRG community.

4. WRG Portal Evolution

4.1 Sun Grid Engine Portal (GEP)

The Sun Technical Compute Portal, renamed to the Grid Engine Portal (GEP), was part of our first phase for portal implementation on the White Rose Grid (WRG). The GEP [11] is a Java-based web application built on top of the Sun ONE Portal Server, and it offers simple role-based access control for both portal administrators and users. Administrators can easily upload new applications, specify which users have access to these applications, and create web forms for capturing application input data. Users can upload data and select applications for submission. The portal launches jobs using automatically created SGE

submission scripts and monitors job status. It is even possible to launch interactive X-windows applications through the portal. However, the GEP integrates directly with SGE; consequently it is suitable for campus Grid installations, but at present it does not support inter-enterprise Grids that span multiple institutions.

4.2 Grid Portal Development Toolkit (GPDK)

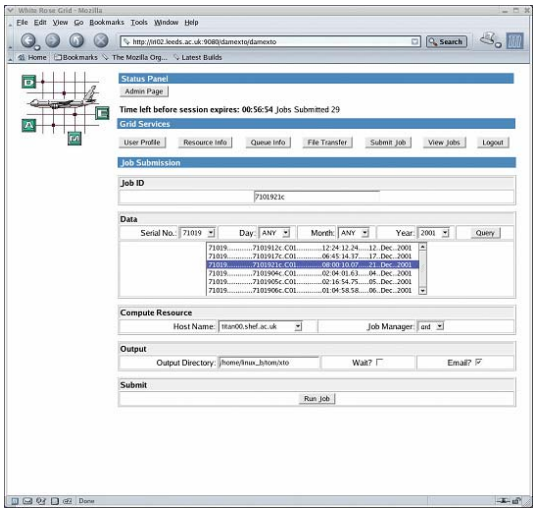

Figure 1: WRG portal based on GPDK; a job submission interface

For further investigations into Grid portals we chose the Grid Portal Development Kit (GPDK), which was produced as part of the USA Department of Energy (DOE) Science Grid project. This toolkit used the Java CoG Kit [7] to integrate with Globus (GT) and more importantly it offered inter-enterprise Grid capabilities that were missing in the GEP. Some re-engineering of GPDK was required to integrate it with GT2, which is deployed on the WRG, as the original GPDK was designed for use with Globus 1.1.4. It provided a very capable generic portal which proved invaluable in terms of demonstrating the potential of a Grid portal technology to prospective users.

Within the DAME e-Science project the GPDK was used to construct a Grid portal (Figure 1) for specific application which performed a DSP analysis of vibration data collected from sensors on aircraft engines. This portal offered an appropriate interface for this particular application, and the burden of specifying the physical location of executables and input data was removed; with just a few mouse clicks it was possible to select an engine dataset from a database and launch an analysis of that data on a particular Grid resource (Figure 2). Our users welcomed this improvement, but it was clear that the development effort required to create and maintain portals that suit each user group and their applications would be prohibitively large.

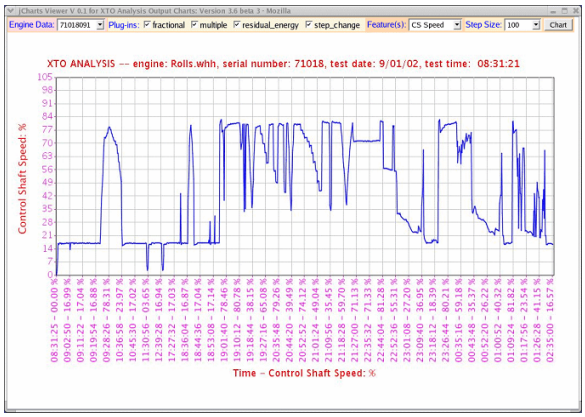

Figure 2: Visualisation of results within the portal based on GPDK

4.3 Portal employing web services and Struts technology

Further developments within the DAME project led to an architecture in which the core functionality was provided by an array of Grid services, orchestrated by a workflow manager. This meant that the portal (Figure 3) itself need only interact with the workflow manager; it no longer required the Globus functionality within the GPDK, and it could become a more lightweight web application, purely concerned with aspects of the presentation layer (Figure 4).

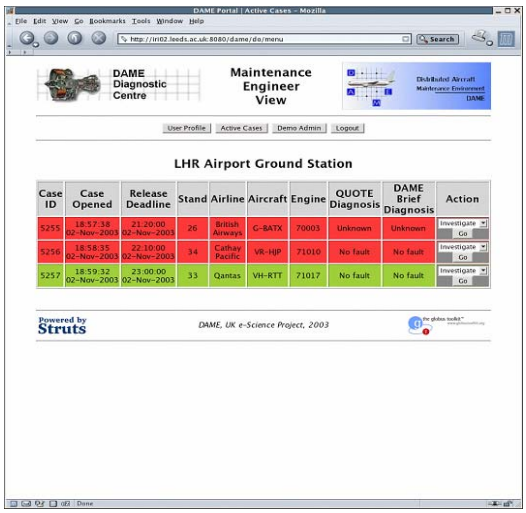

Figure 3: Struts based portal

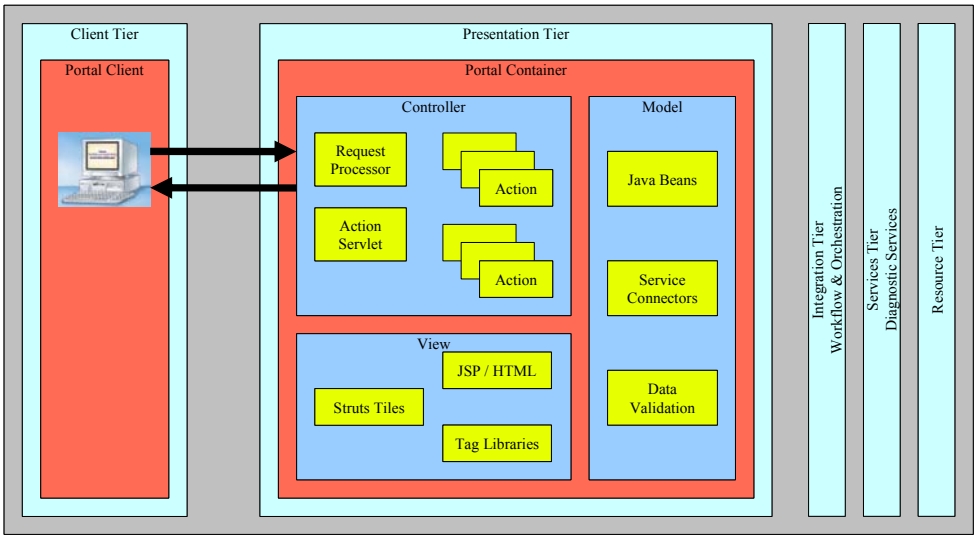

Figure 4: DAME Struts Model-View-Controller Portal

The portal adopts the Model-View-Controller (MVC) design pattern. It is composed of three core components: incoming Requests and Actions are dealt with by the Controller through a Java servlet, that forwards them to the Model which performs various portal operations, whereas the View component is concerned with a display. Subsequently, the Struts framework [8] was employed to provide the new portal with a JSP Model 2 Architecture [9]. The advantage of this approach is that it centralises request handling and decision points (e.g. security), avoids "cut & paste" code distribution, and reduces the maintenance overhead.

4.4 Grid portlets

The Struts framework may have proved useful in easing the maintenance of an established portal, but it does not necessarily help reduce the cost of creating new portals for several user groups, with a varied set of requirements in terms of functionality and user-interface.

Ideally, each user group, or the research community to which they belong, would produce their own portal components that suit their needs. Until recently, this approach was impossible because a lack of standards meant that there was little chance of re-using or sharing portal components.

However, in October 2003 work began on the final version of the Portlet Specification (JSR 168). This is a Java Community Process effort to produce a standard for portal components, otherwise known as portlets. An early implementation of this standard is the open source GridSphere portlet container created by the GridLab project. This includes: a JSR 168 compliant portlet container; a portal incorporating user, group and portlet management facilities; and a set of Grid-enabled portlets. The latest WRG portal [see Figure 5] is based on the GridSphere software, and currently we are helping to produce portlets for various user groups around the WRG.

5. Conclusions

We believe that portals are important for the extensive take-up of Grid computing and for reaching potential Grid users outside of current e-Science projects. In particular for application scientists they offer an easier way for running routine production jobs over the Grid without the need for deep understanding of the Globus Toolkit. Each of the portal technologies we have investigated has provided technical inspiration, but most importantly, due to the portlet standardisation efforts the prospect of re-using portal components is becoming an attractive reality.

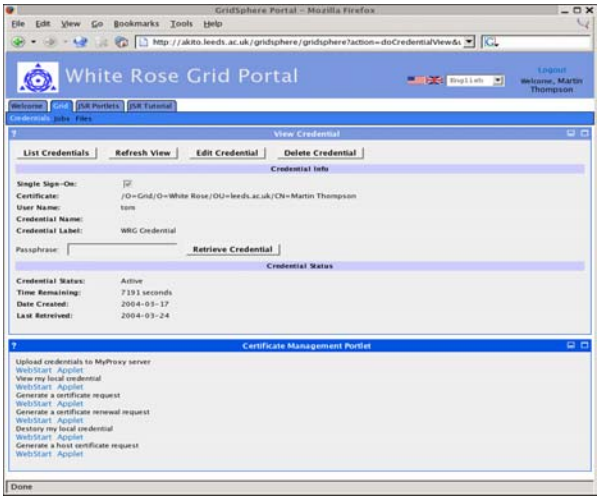

Figure 5: Portal based on GridSphere

Building on our current experience it is envisaged that further work will focus on developments of a set of re-usable portlets, each performing a specific role. These portlets will enable our WRG users to build portals offering the required functionality, and later maintain them within their research groups with minimal support efforts from developers.

Acknowledgements

The authors are grateful for the funding received from EPSRC and the UK e-Science Core Programme, Yorkshire Forward, Sun Microsystems and Esteem Systems.

References

[1] J Chin, P Convey, "Towards tractable toolkits for the Grid: a plea for lightweight, usable middleware", http://www.chem.ucl.ac.uk/ccs

[2] P M Dew, J G Schmidt, M Thompson, and P Morris, "The White Rose Grid: Practice and Experience" in *Proc. UK e-Science All Hands Meeting*, Simon J. Cox Eds, Nottingham Conference Center, U.K., Sept. 2nd-4th, 2003, ISBN 1-904425-11-9

[3] HPCGrid InfoPortal: http://tyne.dl.ac.uk/InfoPortal/

[4] HPC Portal: http://wk-pc1.dl.ac.uk/HPCPortal

[5] WRG DAME portal: http://akito.leeds.ac.uk:8080/dame/

[6] GPDK: http://doesciencegrid.org/projects/GPDK/

[7] G von Laszewski, I Foster, J Gawor, and P Lane, "A Java Commodity Grid Kit", *Concurrency and Computation: Practice and Experience*, vol. 13, no. 8-9, pp. 643-662, 2001, http:/www.cogkits.org

[8] Struts: http://jakarta.apache.org/struts

[9] MVC: http://www.javaworld.com/javaworld/jw-12-1999/jw-12-ssj-jspmvc.html

[10]GridSpehere: http://www.gridsphere.org

[11] Sun GEP: http://www.sun.com/solutions/hpc/tech_computing_portal.html