Top Ten Questions To Design A Successful Grid Portal

Xiao Dong Wang, Xiaobo Yang, Rob Allan
CCLRC e-Science Centre, Daresbury Laboratory
Warrington, WA4 4AD, UK
{x.d.wang, x.yang, r.j.allan}@dl.ac.uk

Abstract

With the enhancement of Grid applications, Grid portals provide a useful interface for end-users to access distributed resources and are used widely in Knowledge Grids, e-Science, e-Learning and e-Business. The success of a Grid portal relies on many factors. This paper focuses primarily on the design aspects of a successful Grid portal implementation. Ten questions on designing Grid portals are discussed in no particular order. The questions detail with the key areas of Grid portal architecture, Grid portal contents, portal security, reusability, software design, management and actual implementation..

1. Introduction

As computational science and knowledge mining are developing, multiple distributed resources are being used widely by scientists and engineers in their work. A portal is a Web-based application that acts as a gateway between users and a range of such Web services. It provides personalization, single sign-on, aggregation and customization features in addition to other Grid functionality [2][3]. Based on the services provided from distributed resources and end users, portals can be constructed for e-Science, e-Business or e-Learning, Grids and Knowledge management, etc. As a subject-specific example, an e-Social Science portal might focus on providing data and information which social science need as the main end-users. 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 improve business relationships and communication [4]. E-Learning portals provide an integrated interface to learning and training resources which include Web-based training, virtual classrooms and digital collaboration [5]. The e-Learning portals are used widely in higher education and the students and staff are end-users with different roles. It is not clear how to define a Grid portal currently, but basically when a portal is designed for e-Science the goal is to access distributed resources and information which are delivered for specific research purposes - we can call this a Grid portal. Knowledge Grid portals, as extensions of Grid portals, are designed on top of the semantic Grid mechanisms and enable particularly on knowledge retrieval, filtering, mining and question-answering techniques [1][12].

The growth 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 (http://www-128.ibm.com/developerworks/websphere/zones/portal), BEA WebLogic Portal (http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/weblogic/portal), Sun One Portal (http://www.sun.com/software/products/portal_srvr/index.xml), etc. Among the open-source portals, GridSphere (http://www.gridsphere.org) is popular and supports a range of Grid Portlets which can access distributed remote resources via Globus Toolkit middleware. Activities are organized regularly to provide training and to exchange experiences and best practice in design for portal developers, for instance a series of workshop on Grid Computing Portals, and workshop of portal and portlets (http://www.nesc.ac.uk/action/esc/contribution.cfm?Title=261, http://www.nesc.ac.uk/esi/events/686).

This paper will discuss the top ten questions of developing successful Grid portal in no particular order. The questions are hoped to give a guide to the software developers who are keen to design Grid and

Knowledge Grid portals. The questions detail the key areas of Grid portal architecture, Grid portal contents, portal security, reusability, software design, management and actual implementation.

2. What is the aim of a Grid portal?

Many organizations and Grid communities want to develop Web-based portals which usually focus on sharing information and centralizing their resources. But before developing a Grid portal, they should answer three questions:

2.1 Why do end-users need Grid portals?

Basically a Grid portal acts as an information aggregator which can access different information resources [7][8]. It is a Web-based application allowing users to access a range of different high-level services using a browser interface. Traditionally users can access distributed system resources by logging in the remote servers individually and processing jobs. This method needs users to install an access mechanism on their local machine, for example, SSH Security Shell can be used to implement the login function (http://www.ssh.com), also PuTTY is another good selection (http://www.chiark.greenend.org.uk/~sgtatham/putty). Using SSH login and submitting job remotely is suitable for running a job on single machine, and probably still the most suitable for complex jobs running on very large supercomputers. But if the users' jobs need to be repeated on several distributed servers, such as in a task-farming application, and also if the jobs cause huge data or information transfer among machines, the SSH mechanism is no longer adequate. A new unique interface should be provided to users, preferably for them to access and manage all of their jobs without any complex software installation. In this case, a Grid portal is a good choice.

2.2 What do end-users want a portal to do?

Grid portals are used by end-users. It is important to understand their requirements in using the portal. Firstly end-users want to understand how the use the portal functionality. They need to decide if this Grid portal provides sufficient functionality for their applications. Also they want to feel comfortable when acing the portal, and to be use and operate it simply. End-users also need to be assured their jobs can be run in a reliable and secure way on the Grid.

Their main concern is with running their jobs rather than Grid portal itself, so the portal should provide easy and simple operation, even for complex and applications. As the functionality of Grid portals grow and more information is supported, ease of use becomes more important. Not all users need to access all of the information or functionality. End-users from different working area focus on different information sources or resources. So a well designed Grid portal should support personalization. Each user can design their personal space and put useful information in their pages. They should be able to edit and update this personnel space in future.

2.3 Buy a commercial portal, use open source portal or D.I.Y.?

Typically the development of a Grid portal is based on a portal framework. A portal framework provides the possibility for portal developers to implement Grid applications and customize users' personnel pages. Some commercial portal vendors provide high performance frameworks. Some organizations release open-source portal frameworks for downloading free. Also portal developers can do 'D.I.Y.' to develop their own Grid portal matching their particular requirements. Portal developers should however understand the differences between the options before deciding their development scenario.

- Commercial portals are stable, powerful and provide good after-sales service, but the license fees probably are expensive. Commercial portals can provide enterprise functions, but if the development budget is limited, Grid portal developers may consider to use open source portal framework.
- Open source portal frameworks are free to download and deploy. New versions are released frequently with new functionality and bug fixes. This means that the portal developers will follow these releases to benefit the from new features and support. It happens more frequently when the new portal framework is being actively developed. Portal developers will often find bugs and should report these to the organization developing the framework. This requires extra effort, so there is a trade-off between the cost of a commercial solution and the time required to support an open-source solution. Indeed it is often found that there is no long-term technical support once development work on an open-source framework is finished, usually

because they have no ongoing funding stream. Typical open-source portal frameworks are uPortal (http://www.uportal.org), GridSphere (http://www.gridsphere.org), JetSpeed (http://portals.apache.org/jetspeed-2), eXo (http://www.exoplatform.com), LifeRay (http://www.liferay.com), StringBeans (http://www.nabh.com/projects/sbportal), etc….

- D.I.Y., developing your own special Grid portal, is another solution. There are many portals built around content management systems (e.g. Zope, PHP-Nuke, ColdFusion, Flash, Mambo…) or using basic servlet technology. But you should consider that the Grid portal development will take longer this way and will not be standards compliant, so cannot benefit from existing tools such as open-source portlets. This will be hard to evolve and maintain and may eventually be more expensive than buying a commercial portal framework.

3. How to treat the end-users?

The effectiveness of a Grid portal depends on how many people are using it. We need to recognize that users can add value to the portal.

- Anonymous or guest users visiting a portal site for the first time are likely to be interested in what the Grid portal can offer but haven't yet decided whether it is worth using or joining in the community using the portal. The portal should provide a mechanism of showing the functions and features to attract people to use it. That mechanism should not require registration before visiting Grid portal and accessing Grid application.
- The portal should provide registration mechanism for the users who decide to use it. Grid portals can provide registered users with premium contents, the ability to collaborate on line, interaction Grid resources and portal personalization.
- Furthermore Grid portals should make registered users feel part of a virtual community and encourage them to extend the community. The registration form should be simple, but should allow users to add extra information. After submission, a confirmation will be sent and information can be updated in future.

4. What services does the portal provide?

Grid portals are only as good as the services they provide. So even a well-designed portal will fail if the services aren't valuable or interesting. A well designed Grid portal should possibly have its contents in two categories: core services and additional services.

Core services may consist of execution management, data/ file management and information services. Execution management services are concerned with the initiation, monitoring, management, scheduling, and/ or coordination of remote Grid computations. The users can invoke their jobs to run on distributed resources. Globus Toolkit supports the Grid Resource Allocation and Management (GRAM) as a basic mechanism for this purpose. Data/ file management services are concerned with the location, transfer, and management of distributed data and files. Users can manage results of computations put them in local or remote storage. The typical data/ file management services are GridFTP, RFT, RLS, and OGSA-DAI which are integrated in Globus Toolkit, and SRB which is supported by San Diego Supercomputer Center (http://www.sdsc.edu/srb/). The information services can monitor, discover and test distributed resources and services of Grid systems. Users can easily find out system resources and suitable machines to submit jobs via information services. The typical information services are MDS supported by Globus Toolkit, Ganglia (http://ganglia.sourceforge.net) and Inca provided by San Diego Supercomputer Center (http://inca.sdsc.edu).

Additional services are not associated with Grid computation normally, but they can add value to a Grid portal. For instance, calendar services allow users to arrange their job schedules properly; e-mail services may notify users of changes in job status; discussion forum provides a platform to share technical knowledge, collect feedback by portal developers, and announce software updates.

5. What kind of portal architecture?

A typical first-generation Grid portal architecture consists with three tiers [6]. The first tier is the Web browser, Web services and proxy credential services are in the middle tier, and the third tier includes backend services and resources, such as databases, high performance computers, storage and specialized devices. The first generation Grid portal has some limitations, mainly lack of customization and restricted Grid services. Access for end-users is static and it is almost impossible to dynamically customize a portal to

meet their special needs. As early Grid portals are coupled with specific Grid middleware technologies, it is hard to integrate in other services later.

To overcome the above limitations, the second-generation portal architecture is designed as a Service-Oriented Architecture (SOA) and consists of three layers which are portal layer, service layer and resource layer [11]. The portal layer mainly includes a portal framework which provides presentation functions to the end-users. The service layer implements services called from the portal layer and accesses the physical remote resources. Usually most of the business logic is included in the service layer. The resource layer covers all the distributed resources of the Grid system. It generally includes database, HPC resources, visualization resources, etc. and may include instruments such as telescopes on line.

Portlets act as components managed by a portlet container to access a set of services. The Java portlet standard provides the possibility of component reusability [9]. For example, a JSR 168 portlet can be plugged and played in any Grid portal which supports this standard without source code modification (http://www.jcp.org/aboutJava/communityprocess/final/jsr168). JSR 168 supports a number of window status and page modes providing personalization for end-users. The use of portlets is highly recommended in designing Grid portals, as for instance is done in GridSphere.

6. Is the portal secure to use?

Security is an import issue on the Grid. It is necessary to ensure not only no leaking of information about registered users and running jobs and data, but also to protect the Grid system from compromises. With regard to the Grid portal architecture, the portal and service layers are usually hosted on a server, the resource layer consists of distributed resources which map to remote machines. The communication between service layer and resource layer could be standard Web service protocol (SOAP, WSDL) or other protocol such as used by GRAM. End-users visit the portal via Web browser, so the security should focus on the communication between end-users and portal layer, and between service layer and resource layer. Grid portal security should include two parts: data transfer and authentication. Most data transfer happens between end-users and portal layer and between service layer and resource layer. Authentication is typically by an X.609 certificate validation which includes checking user's identity and resources usage permission, the latter being a simple form of authorization control.

Traditional transfer protocol is HTTP which transfers data in plain text type without any security. HTTPS adopts the Secure Sockets Layer (SSL, now known as TLS or Transport Level Security). SSL is a protocol originally developed by Netscape for transmitting private documents via the internet. Most browsers support SSL and many Web sites use the protocol to secure confidential user information.

Certificates are digital documents attesting to the binding of a public key to an individual or other entity. They allow verification of the claim that a given public key does in fact belong to a given individual. Certificates help prevent someone from using a phony key to impersonate someone else. Usually a proxy of user's digital certificate is used to validate their identity. But can you create a proxy certificate easily in any machine which uses a Web browser to access Grid portal? The answer is no. NCSA provide MyProxy as a good solution (http://grid.ncsa.uiuc.edu/myproxy/). MyProxy combines an online credential repository with an online certificate authority to allow users to securely obtain proxy credentials when and where needed. It is used widely in Grid community particularly for portals.

7. Can portal tools be reused?

With the growth of powerful and complex Grid portal tools, more and more portal developers want to collaborate and share their work. Usually one Grid portal tool requires several developers to write code, test and deploy it. So we need a special design model to ensure that developers can work in parallel and efficiently. Portal tools should also be written in a way which is reusable for future development.

The Model-View-Controller (MVC) pattern was originally proposed by Xerox in 1979 (http://heim.ifi.uio.no/~trygver/ themes/mvc/mvc-index.html). The key point of MVC is to separate a tool application's data model, user interface, and control logic into three distinct parts, so that modifications to one part can be made with minimal impact to the others. JSP is widely used to create View components. JavaServer Faces (JSF) is a new technology to build user interfaces (UI) (http://java.sun.com/javaee/javaserverfaces/). It is a standardized specification for building UIs for portal server-side applications. Grid portal developers of various skill levels can quickly build Grid applications by: assembling reusable UI components in a page; connecting these components to a Grid application data source; and wiring client-generated events to server-side event handlers.

As described above, JSR 168 portlets provide the possibility of reusability. Web Services for Remote

Portlets (WSRP) (http://www.oasis-open.org/committees/download.php/ 3343/oasis-200304-wsrp-specification-1.0.pdf), is another promising specification that is attempting to standardize presentation-based and interactive portlets. WSRP defines a communication protocol between a Consumer, which is typically a Grid portal, and a Producer, which is a remote portlet container. WSRP allows functionality that was previously difficult to achieve, such as deploying portlets once, but calling them from anywhere, bringing together third-party portlets, and enhancing interoperability between portals from different vendors. WSRP also gives Grid portal developers a feasible way to build distributed Grid portals consisting of a network of interoperating Grid portals, whereby services hosted in one Grid portal can be made available to many. This innovation gives WSRP the potential to enhance portal deployment flexibility with access to remote resources.

8. How to share and debug source codes?

With the Grid applications growing in complexity, many developers will be involve intesting, debugging and implementation. So we need a method to simplify the complex tasks of software development, transfer structure thoughts, clarify communication, and find right abstraction. UML, the Unified Modeling Language is useful for these purposes (http://www.omg.org/technology/documents/formal/uml.htm). UML provides a notation for describing software systems in an object-oriented way, for instance using a mixture of text and graphics. It is also general enough to describe non-software systems, for example, workflow or other business logic processes. Some UML tools can support UML features and generate software directly. There are many tools available, again some being commercial and highly functional, some open source and less so.

Successful projects are often the result of successful organization and communication. During software development and even when software is running in the portal, there may still be exceptions and errors discovered. A mechanism needs to be provided for the users to report any exceptions during use of the Grid portal and for software developers to solve any problem. Bug tracking sustems such as Bugzilla and Elips are the useful. Bugzilla for instance is server software designed to help developers manage software upgrades (http://www.bugzilla.org). It allows individuals or groups to keep track of outstanding bugs in their product effectively. Such tools can be directly built into the portal.

Another solution to communicate and announce information is to create e-mail group mailing list. Groups can be either public for reporting any problems and asking for help or they can be private, for instance for researchers working on a specific project. The mailing list design should allow people to subscribe and unsubscribe by themselves and it will have a portal interface available alongside the other tools.

9. How to manage software versions?

Software development needs version control. The history of source files and documents need to be recorded and retrieved with the source. Developers may work together on the same source code. Tools such as CVS, Concurrent Versions System, (http://www.nongnu.org/cvs/) can allow software developers to save and retrieve different development versions of source code and add comments to it. It is a system for managing simultaneous development of files, and is in common use in large programming projects [10]. CVS is designed for developers, either individually or in teams. For individuals, CVS provides a repository from which you can work from home, the office, or the client site without having to haul disks around. For teams, it also keeps a record of who changed which lines of a file and prevents direct overwriting of each other's work. SubVersion is another successful open source version control software (http://subversion.tigris.org) now becoming widely used.

10. How to design code guidelines?

With the enhancement of Grid applications using Java, more and more software is written as classes and called by other classes. If the software development involves many developers, it may happen that the same class name is used for entirely different functions. This can at best lead to confusion, and will cause serious problems if calling the wrong class.

A solution is to wrap the source code in package with different paths. By this method, it is easy to distinguish the classes and organize the structure of source code. A package is suggested as "*organisation.country.institution.department.group.portalname.Grid-application-category.portlet-function-name(or utility-name)portlet-name.*". For example, in the NGS Grid portal design, a jobsubmission portlet class path is wrapped in a package as *org.uk.cclrc.esc.gtg.ngsportal.jobsubmission.jobsubmissionportlet.class*. It expresses that the jobsubmission portlet is developed by Grid Technology Group (gtg)

of the e-Science Centre (esc) in the Council for the Central Laboratory of the Research Councils (cclrc) in the UK and it is used for National Grid Service portal (ngsportal). The above description expresses the full information of this Grid application clearly.

11. How do we test Grid portals?

Test of the Grid portal are important for designing a successful, powerful and stable product. The tests includes software and operations. Software tests will focus on efficiency and functionality, and will mainly be done by portal developers. Some useful test tips could be:

- Set break points in conditional paths and test if it breaks when the condition happens;
- Run Grid portal in different Web browsers to check if GUI interfaces are compliant with these Web browsers;
- Run as many Web browsers as possible accessing the portal in parallel to test the scalability of the server;
- Stop a job and recover it to test fault tolerance;
- Run Grid jobs via internal and external firewalls.

Operational tests could be carried out by early-bird users. These users can be volunteers from colleagues of developers or from external organizations. The users are encouraged to test in several ways as below:

- Access the Grid portal concurrently on the same or different internal or external sites to test scalability;
- Access the Grid portal using IP addresses instead of domain names;
- Access the Grid portal with huge or complex jobs to test functionality and robustness.

12. Conclusions

A successful Grid portal depends on many factors. This paper has focussed primarily on the design aspects which might lead to a successful implementation. Ten questions in designing a Grid portal are discussed particularly covering key areas for consideration in a Grid portal architecture, its contents, portal security, reusability, software design, management and implementation testing. Hopefully it can provide helps and ideas for portal developers to design successful, powerful portals as scientific gateways to the Grid.

References

[1] Zhuge, H.: China's E-Science Knowledge Grid Environment, IEEE Intel. Systems, 19 (1) (2004) 13-17.

[2] Allan, R., Awre, C., Baker, M., Fish, A.: Portals and Portlets 2003. NESC Workshop, 14-17, July 2003. http://www.nesc.ac.uk/technical_papers/UKeS-2004-020.6.pdf.

[3] Wang, X. D., Allan, R.: Portlet, WSRP and Application. GridSphere and Portlets Workshop. Edinburgh, 3-4 March 2005.

[4] White, C.: The Evolution of the E-Business Portal. DM Review Magazine. August 2001 Issue. http://www.mdreview.com/article_sub.cfm?articleId=3822.

[5] Mulgan, F.: E-Learning = Environment Learning. Learning Circuits. http://www.learningcircuits.org/2006/April/mulgan.htm.

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

[7] Thomas, M. P., Burruss, J., Cinquini, L., Fox, G., Gannon, D., Gilbert, L., Laszewski, G. V., Jackson, K., Middleton, D., Moore, R., Pierce, M., Plale, B., Rajasekar, A., Regno, R., Roberts, E., Schissel, D., Seth, A., Schroeder, W.: Grid Portal Architechures for Scientific Applications. Journal of Physics: Conference Series 16 (2005) 596-600.

[8] Kim, M., Meliksetian, D. S., Malone, C.: A Mainframe Grid Computing Infrastructure. 25 May 2005. WebSphere Journal. http://websphere.sys-con.com/read/47894.htm.

[9] Lin, A. W.: Building a Unified Grid, Part 2: Portlet Interfaces in the Grid User Environment. 29 Nov. 2005. http://www-128.ibm.com/developerworks/grid/library/gr-unified2.

[10] Vesperman, J., Introduction to CVS. http://www.linuxdevcenter.com/pub/a/linux/2002/01/03/cvs_intro.html.

[11] Wang, X. D., Yang, X., Allan, R.: Flexible Grid Portlets to Access Multi Globus Toolkits. In GCC2006: CVRE06 Workshop. Changsha, China, 21-23 Oct. 2006.

[12] Zhuge, H.: Discovery of Knowledge Flow in Science. Communications of the ACM, 49(5) (2006) 101-107.