Flexible Grid Portlets to Access Multi Globus Toolkits

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

Abstract

Grid systems at different levels have already been constructed around the world to provide users an easy way of accessing/integrating distributed computing, data and other resources. Portal layer is on the top of typical three layers in Grid system and provides presentation interface to the end-users. Portlets as executing components in portal layer usually access a specific Grid middleware rather than multi Grid middlewares. A new Grid system is implemented in this paper. It follows typical three layer structure of Grid system. By using UDDI registry, Grid Portlets in the portal layer can select services interfaces and find out suitable Grid middleware. Abstraction and Business Logics which are Java CoG-based in service layer will map to suitable Globus Toolkit middlewares. The end-users can use the Grid Portlets to access GT2, GT3 or GT4 middleware.

1. Introduction

Grid is now generic in research communities while the industry is gradually adopting and deploying. Grid systems at different levels have already been constructed around the world to provide users an easy way of accessing/integrating distributed computing, data and other resources. Currently typical Grid systems include three layers: portal layer, service layer and resource layer[1]. Portal layer which mainly consists of portal framework provides presentation function to the end-users. Service layer implements the calling from portal layer and accesses the physical remote resources. Usually business logic is included in the service layer to invoke different physical resources. Resource layer covers all the remote physical resources which usually including databases, visualization resources, storage resources, data collections and HPC resources.

Portals acting as information aggregators can access different information resources[2][3]. It is a Web-based application allowing users to access range of different high-level services using a browser interface. A portal framework can provide presentation abilities for Grid services in additional to personalization and content management[4][5]. A so-called 2nd generation portal normally consists of different portlets to process consumer requests to these services and generate dynamic content from the response[6]. Portlets usually act as components managed by portlet container to access a set of information resources. Portlet standard provides the possibility of resource reusability[7][8]. For example, JSR 168 portlet can be plugged and played in a portal framework which supports JSR168 standard without source code modification. The supports of JSR 168 to window status and page mode provide the personalization to the end-users. The usage of Java Server Page (JSP) successfully divides the control and presentation function, and hide the complex resource calling and control logic in the back. It is possible to represent friendly User Interface (UI) easily. In Grid portal system, portlets as executing components are linked to the remote resources eventually, the portal performance is highly relayed by the implementation of Grid portlets[16].

Typical remote resources are deployed by Globus Toolkit[17]. Globus Toolkit is an open source software toolkit used for building Grid systems and applications[14]. The first version of Globus Toolkit is GT1.0 which released in 1998. GT1.0 is written in C and implements several Grid resource management and monitoring functions. For example, the Grid Resource Allocation and Management (GRAM) provided by GT1.0 can submit job to remote machines, and also can monitor the submitted job status, the Monitoring and Discovery System (MDS) provides a mechanism to monitor the remote resource status, the search result can be feedback as reference for end-users to submit the job to suitable remote machines. Although GT1.0 is available earlier, the first productive Globus Toolkit software (GT2.0) is released till 2001. GT2.0 is still written in C. It inherits the features of GT1.0, also adds file process function, for instance, the Grid File Transfer Protocol (GridFTP) can transfer file among local site, remote and third party sites. As GT2.0 improves reliability, scalability and security, it is used widely in UK e-Science area, for instance the National Grid Service[18], e-HTPX[19], Integrative Biology[20], etc. Followed with the emerging of the Open Grid Service Infrastructure (OGSI), GT3.0 which is the implementation of OGSI specification is issued in January 2003. GT3.0 is written in Java and C. GT3.0 implements the service status stateful. When clients access GT3.0 server, the server will create an instance first and return the instance handle back to clients. By connecting with the instance handle, the server can provide the status of the job submitted by the users. But as Web Service is still popular and get widely support from industry, a new specification of Web Service Resource Framework (WSRF) is finally proposed. WSRF includes a set of Web Service specifications. These specifications are WS-ResourceLifetime, WS-ResourceProperties, WS-RenewableReferences, WS-ServiceGroup, WS-BaseFaults and WS-Notification which are W3C standards[7][8]. Based on the specification of WSRF, GT4.0 is implemented and released in April 2005.

With the expense of Grid system, more Grid softwares are installed in the remote sites. The Grid software may be installed as different Globus Toolkit versions. We can provide different portlets to access these resources which installed as different Grid software versions on the distributed sites separately. But can we provide one unique portlet interface to access all of the gird resources with different versions?

In this paper, we will first introduce several popular Grid systems which are using Globus Toolkit and then analyze the need of new portlet interface to access Globus Toolkit of different versions. A new Grid system is presented. The Grid Portlets in the system are designed to access different resources versions in unique interface. Finally future works and conclusions will be drawn.

2. Related Works

Currently some projects have developed Grid Portlets to implement the Grid applications based on Globus Toolkit:

GridPort: The latest version of GridPort is Java based and JSR 168 compliant. Similar with OGCE 2, a set of Grid-related portlets such as GridFTP (for file transferring) and GRAM (for job submission) are developed to help users set up Grid portal. GridPort 4 has a clear three-tier architecture which includes portal layer, service layer and resource layer. The portal layer is responsible for interacting with end-users and the service layer, which then handles interaction with the resource layer[21]. Currently the Grid Portlets of GridPort 4 only support GT2 to implement grid application. GridPort has been used in some other projects, for example, the Open Grid Computing Environment (OGCE) Release 2 is built on the GridPort 4 source code[22].

GridSphere: It is funded by the EU and hosted in GridLab[23]. It is open source portal framework and supports JSR 168. Since June 2005, it releases a set of Grid Portlets. The portlets can manage jobs and data on the grid. The Grid Portlets support Grid applications: GRAM, gridFTP, etc… SRB portlet will release soon. The Grid Portlets only access GT3 in the latest Grid Portlets 1.3 and by using of MyProxy server to hold end-users credential. Then the Grid Portlets will use the credential to implement grid jobs.

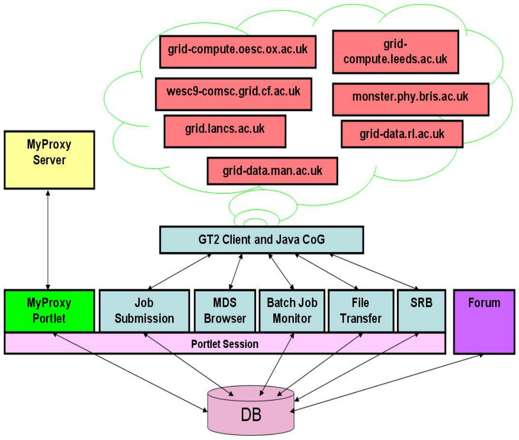


Fig. 1. NGS Portal Architecture

NGS Portal: The UK NGS (National Grid Service) Portal is developed by the Grid Technology Group at Daresbury Laboratory[15]. It follows similar procedure as OGCE. The first release of the NGS Portal is CHEF based while release 2 is JSR 168 compliant. StringBeans, an open-source portal framework[24] has been customized to support MyProxy logon to the portal. Portlet session function is adopted for the inter-portlets communication. NGS Portal provides a set of Grid Portlets to access Grid applications[25]. The Grid Portlets only access GT2 services currently. The NGS Portal architecture is shown in Fig. 1.

3. Several Comments to Current Grid Portlets

Portals acting as information aggregators can access different information resources. Portlets usually act as executing components to process and manage the information resources.

From the projects introduction above, we know that the Grid Portlets can only access a typical set of grid sources, for example GT2, or GT3 only. It means that when the Grid Portlets are invoked to access information resources, the middleware in resource layer has to be same version. Obviously this limits the usability of Grid Portlets. The end-users have to know the remote resources, for example which version Globus Toolkit installed, when deploying the Grid Portlets in portal framework. Furthermore the end-users can not set other services interface which different with the original support to work by editing the Grid portlets, for example, currently NGS Grid portlets can not access the job submission by just adding ManagedJobFactoryService interface of GT3 or GT4 in edit page. Usually end-users just care of their job implementation. They have not much more ideas what kinds of services the Grid systems can provide. Also it is difficult for them to understand what resource middlewares which installed in the remote resource layer can be used as not all end-users are expected to be grid experts.

Furthermore with more and more grid applications, a Grid system will cover much more resources. These resources may be running in different Grid middleware versions with the Grid system expanding and resources upgrading, for instance NGS will provides GT2 and GT4 middlewares soon. So it brings the needs that one Grid Portlet can access a set of Grid resources of different versions, for instance a Grid Portlet may access GT2, GT3 and GT4 as well.

In the next part, we will introduce a new Grid portal system. It follows the three layer architecture: portal layer, service layer and resource layer. Moreover, the services information which published in UDDI registry can be retrieved to Grid Portlets by end-users. By involving complex business logic and Java CoG 4, the Grid Portlets can access Grid resources of different versions and record the resource interface by using *service provider* item.

4. Grid Portlets to Access a Set of Grid Middleware

Based on the above analyze, we design a new Grid portal system. The system architecture is shown as Figure 2. The whole system consists in three layers structure: portal layer, service layer and resource layer. As we focus on portlets which can access multi Globus Toolkits, so security issue will not be discussed in this paper.

4.1. Grid System Architecture

Portal layer includes portal framework and UDDI Registry. Universal Description Discovery Interface (UDDI)[26] as W3C standard provides a mechanism for Web services owners to publish their own Web services and for the end-users to query the Web services provided. It defines standard protocol of services owners and services themselves. The Grid services in the remote resource layer are published in UDDI. End-users of Grid portal system can get the standard description and interface of the resource services by searching UDDI registry. The Grid Portlets are hosted in portal framework and managed by portlet container. When end-users plug the Grid Portlets in the user page, the portlets will visit UDDI registry, retrieve resource services interface and display back on the portlets. So the end-users can get the resource service information and decide to put their jobs to the service.

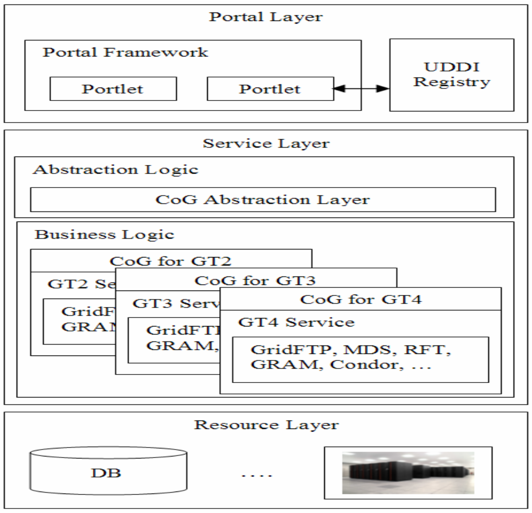


Fig. 2. Grid System Architecture

Service layer consists of Abstraction logic and Business logic. Abstraction logic provides a generic mechanism to load java library of different grid middleware by using java thread loader. This logic is Java CoG 4 based and is identified by the item of *service provider*. Following with the instruction from Abstraction logic, Business logic maps to the suitable Globus Toolkit services by loading Java CoG API[27].

Resource layer covers all the remote resource in the Grid system. It usually includes database, HPC resources, visualization resources etc .

4.2. UDDI Data Structure

The remote resources are published in a UDDI registry. The publishing data structure is shown as Figure 3. In the UDDI registry, the businessEntity structure is presented as top level to restore all information about a remote resource. The <name> and <description> elements are set to provide the name of remote site and describe which Globus Toolkit can be supported. Any query of the remote site can find out the suitable contacted person in the <contact> element. Also further information can be accessed by linking website provided in the <discoveryURL> element. One remote site can provide more services. Some services are GT2 supported, others may be GT3 or GT4 based. All the services can be hosted in businessService structures. One businessService structure maps one resource service. The <name> element in the businessService structure provides the name of the service. The description of the service can be retrieved from the <description> element of the businessService structure. The service interface is published in the <accessPoint> element of the bindingTemplate structure.

```
<businessEntity    businessKey=”E39A39B0-5C2D-11DA-BB66-DF6B8F5B1501”
operator=”CCLRC e-Science Centre”>
  <name>a new site to NGS from CCLRC e-Science</name>
  <description>This site provides GT2 and GT4 services.</description>
  <discoveryURLs><discoveryURL
userType=”businessEntity”>http://grid12.esc.rl.ac.uk</discoveryURL></discovery
URLs>
  <contact>
    <personNane>Dr. Xiao Dong Wang</personName>
    <description>If you have any problem, please contact me.</description>
    <phone>01925 603821</phone>
    <email>x.d.wang@dl.ac.uk</email>
  </contact>
…………………………………………………………………
  <businessService serviceKey=”E3A0F070-5C2D-11DA-BB66-BEE6D7B39B41”
businessKey=” E39A39B0-5C2D-11DA-BB66-DF6B8F5B1501”>
    <name>ManagedJobFactoryService</name>
    <description>This service can let you submit job on it and it is GT4 supported.
</description>
    <bindingTemplate>
      <description>web service interface to submit job.</description>
<accessPoint>http://grid12.esc.rl.ac.uk:7000/wsrf/services/ManagedJobFactoryServ
ice</accessPoint>
    </bindingTemplate>
  </businessService>
  <businessService  serviceKey=”E3A0F070-5C2D-11DA-F390-D4E314741DB6”
businessKey=” E39A39B0-5C2D-11DA-BB66-DF6B8F5B1501”>
    <name>ManagedJobFactoryService</name>
    <description>This service can let you submit job on it and it is GT2 supported.
</description>
    <bindingTemplate>
      <description>a interface to submit job.</description>
      <accessPoint>grid12.esc.rl.ac.uk</accessPoint>
    </bindingTemplate>
  </businessService>
</businessEntity>
```

Fig. 3. Remote Resources Data Structure in UDDI

4.3. Grid Portlets Implementation

Our proposed Grid Portlets support several grid application: to submit job to remote machines; to transfer files remotely; to operate file system remotely; to submit jobs parallel; to support condor etc.

Basically the implementation of the Grid Portlets can be described as below:

```
get user credential from MyProxy server ;
visit UDDI to select services;
if( provider.isKnown ) {
   getProvider( service );
   access business logic based on the provider;
   call remote resource;
}
else {
   do {
      select a provider from providers list;
      access business logic based on the provider;
      call remote resource;
   }while( job.isSuccessful )
   restore provider and service interface in pair;
}
```

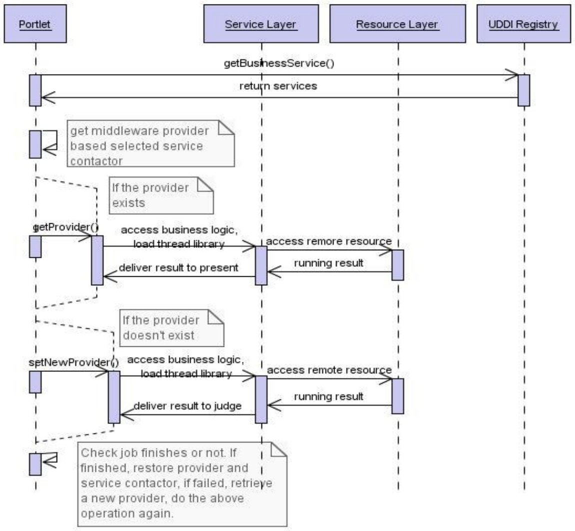

Portlet
Service Layer
Resource Layer
UDDI Registry
getBusinessService()
return services
get middleware provider based selected service contactor
If the provider exists
getProvider()
access business logic, load thread library
access remore resource
running result
deliver result to present
If the provider doesn't exist
setNewProvider()
access business logic, load thread library
access remote resource
running result
deliver result to judge
Check job finishes or not. If finished, restore provider and service contactor, if failed, retrieve a new provider, do the above operation again.

Fig. 4. Grid Portlet Flow Diagram

The flow diagram is shown as Figure 4. When end-users plug in and try to invoke the Grid Portlets, the Grid Portlets will visit UDDI registry to get BusinessService object: getBusinessServices. Then a list business services published in UDDI registry will be retrieved back. The BusinessService object includes information of services and owner of services. We can call below methods to get furthermore:

String[ ] getBusinessServiceName( );

```
String[ ] getServiceHandles( );
String[ ] getDescriptions( );
String getContactPerson( );
String getContactPhone( );
String getContactEmail( );
```

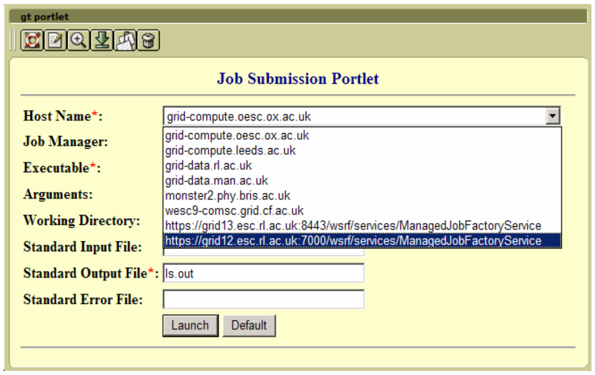
After one of the services interface handles is selected, the Grid Portlets will try to find out the *service provider* which matched with the selected service interface from database or portlet preferences. The setting of database or portlet preferences is deployed in configuration file. The configuration file is assigned as portlet.xml and is put in the directory of servler-container/webapps/grid-portlet/WEB-INF. The servlet-container is the home directory of a servlet container which can be used as Tomcat, JBoss, etc. grid-portlet is the name of our Grid portlets. A fragment of portlet.xml is showed as below:

```
<portlet-preferences>
  <preference>
    <name>jobsubmission_hostlist</name>
    <value>grid-data.rl.ac.uk,grid-compute.oesc.ox.ac.uk,http://grid12.esc.rl.ac.uk:7000/wsrf/services/ManagedJobFactoryService</value>
  </preference>
  <preference>
    <name>jobsubmission_providerlist</name>
  <value>Auto,gt2,gt3.0.2,gt3.2.0,gt3.2.1,gt4.0.0,gsiftp,gridftp,ftp,webdav,condor</value>
  </preference>
</portlet-preferences>
```

If the *service provider* exists, then processing Business logic with *service provider* directly. If the *service provider* is not found, then the Grid Portlet will retrieve a new *service provider* from service provider list, and process Business logic by using it. If the job is running successfully, the *service provider* and service interface will be restored in portlet preferences or database as a pair. So when end-users import a new service interface, the Grid Portlets will spend some time to find out suitable *service provider*. Once the *service provider* is found out and saved, the Grid Portlets will pick up the *service provider* directly and implement Grid job by accessing service layer when the Grid Portlets are called later by the same end-users.

4.4. Screen Shoot of Grid Portlets

We deploy and plug the Grid Portlets in several portal frameworks to test the usability and flexibility. The test result shows that the Grid Portlets can be plugged and play very well in some selected open source portal frameworks, for instance, StringBeans, uPortal[28]. Figure 5 shows a screen shoot of the Grid Portlets to submit job to GT2 and GT4 successful in uPortal.



Job Submission Portlet

Host Name*: grid-compute.oesc.ox.ac.uk
grid-compute.oesc.ox.ac.uk
grid-compute.leeds.ac.uk
grid-data.rl.ac.uk
grid-data.man.ac.uk
monster2.phy.bris.ac.uk
wesc9-comsc.grid.cf.ac.uk
https://grid13.esc.rl.ac.uk:8443/wsrf/services/ManagedJobFactoryService
https://grid12.esc.rl.ac.uk:7000/wsrf/services/ManagedJobFactoryService

Job Manager:

Executable*:

Arguments:

Working Directory:

Standard Input File:

Standard Output File*: ls.out

Standard Error File:

Fig. 5. Screen shoot of Grid Portlet

5. Future Works

The implementation of our Grid Portlets is based on Java CoG 4. As Java CoG 4 is still in developing, our Grid Portlets only implement Job Submission, File Transfer and File Operation which support by Java CoG 4 currently. Some extra functions, for example condor, will provide based on the support of Java CoG 4 lately. With the expanding of NGS and NW-Grid community, more and more sites will join in. Some new GT middlewares will be installed in the distributed sites. Our NGS Portal and NW-Grid Portal need to provide the support for the end-users to access these GT middlewares. Obviously our Grid system will be adopted and Grid Portlets will be imported in the latest NGS and NW-Grid Portals.

References

[1] Dahan, M., Boisseau, J., Roberts, E., Thomas, M., Urban, T., Walling, D.: Building grid portals with Grid Portal Toolkit 3. 19 October 2004. http://www-128.ibm.com/developerworks/grid/library/gr-gridport

[2] 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

[3] 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

[4] Sun Microsystems: Building a Global Compute Grid – Two Examples Using the Sun ONE Grid Engine and the Globus Toolkit. 22 Auguest 2003.

http://www.informit.com/articles/article.asp?p=100575&seqNum=7&rl=1

[5] Ichikawa, K., Date, S., Kaishima, T., Shimojo, S.: A Framework Supporting the Development of a Grid Portal for Analysis Based on ROI. 2005:44(2):265-9, Methods Inf Med.

[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] 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

[8] Wang, X. D., Yang, X., Allan, R.: Plug-and-Play Remore Portlet Publishing. In GCE05: Portals Workshop, Seattle, USA, submitted to Concurrency Computat.: Pract. Exper., 2006

[9] Argonne National Laboratory, Introduction to GT3. http://www.dma.unina.it/~murli/SummerSchool/presentations/Naples-02_GT_Evolution.ppt

[10] Laszewski, G., Amin, K.: Grid Middleware. http://www-unix.mcs.anl.gov/~laszewsk/papers/vonLaszewski--grid-middleware.pdf

[11] Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp 2-13, 2005

[12] Foster, I.: Service-Oriented Science. Science 6 May 2005: Vol. 308. no. 5723, pp. 814 – 817.

[13] Humphrey, M., Wasson, G., Jackson, K., Boverhof, J., Rodriguez, M., Bester, B., Gawor, J., Lang, S., Foster, I., Meder, S., Pickles, S., McKeown, M.: State and Events for Web Services: A Comparison of Five WS-Resource Framework and WS-Notification Implementations. 14th IEEE International Symposium on High Computing (HPDC-14), Research Triangle Park, NC, 24-27 July 2005

[14] Parashar, M., Lee, C. A.: Grid Computing: Introduction and Overview. Special Isssue on Grid-Computing. *Proceedings of the IEEE*, 93(3):479-484, March 2005

[15] Yang, X., Chohan, D., Wang, X. D., Allan, R.: A Web Portal for the National Grid Service. In UK e-Science AHM 2005, Nottingham, UK, available on CDROM, 2005

[16] JSR 168 Portlet specification. http://www.jcp.org/aboutJava/communityprocess/final/jsr168

[17] Globus Project. http://www.globus.org

[18] NGS Project. http://www.ngs.ac.uk

[19] e-HTPX Project. http://clyde.dl.ac.uk/e-htpx/index.html

[20] Integrative Biology Project. http://www.integrativebiology.ac.uk

[21] gridPort Project. http://www.gridport.net

[22] OGCE Project. http://www.ogce.org/index.php

[23] GridSphere Portal Framework. http://www.gridsphere.org

[24] StringBeans Portal Framework. http://www.nabh.com/projects/sbportal

[25] National Grid Service Portal. http://portal.ngs.ac.uk

[26] Universal Description Discovery Interface Specification. http://www.uddi.org

[27] Java CoG Project. http://wiki.cogkit.org

[28] uPortal Project. http://www.uportal.org