Developing Application-Specific Portals
With GridSphere / GridPortlets
The GridSphere Project

- We all know a little about GridSphere, but we must stress that the primary goal of the GridSphere Project has always been to develop a Grid portal framework.
- We know that GridSphere is a JSR-168 compliant portlet container that offers a set of core portlets that provide the base functionality we think is required for all Web portals.
- GridSphere also provides a framework for developing and packaging portlets, which includes some base portlet classes, JSP tags and a security model for assigning access rights to portlets.
- So where is the “Grid” in GridSphere?
GridSphere’s Grid Portlets

- The GridSphere portlet container is designed to be web application independent. Indeed, one of the key advantages of the Portlet API is the reuse of web applications.
- Thus, the GridSphere portlet container does not contain any support for using Grid technologies.
- Instead, GridSphere’s Grid related functionality is contained in a web application we call Grid Portlets.
- Grid Portlets, together with the GridSphere portlet container, offers a generic Grid portal and can be used to develop application-specific Grid portal applications.
How To Obtain Grid Portlets

• Grid Portlets can be obtained from the GridSphere Website or from CVS. Create a directory called “projects” (if it doesn’t exist) wherever you have GridSphere installed, change to that directory, then place Grid Portlets there.
• Here is how to obtain Grid Portlets from CVS...

> cd gridsphere
> mkdir projects
> cd projects
> cvs -d :pserver:anonymous@cvs gridsphere.org:/home/repository login
> cvs -d :pserver:anonymous@cvs gridsphere.org:/home/repository co gridportlets
> cvs update -dP
Required Software

- On the portal host you will need:
  - GridSphere 2.0.4 or higher.
- On the Grid, at minimum you will need:
  - MyProxy (for credential retrieval)
  - Remember, Grid portals are no good without a Grid!
Installing Grid Portlets

- Then follow steps outlined in QUICKSTART.txt to get started right away, or run “ant docs” to generate the Grid Portlet Administrator Guide (and other guides that come with Grid Portlets). After you’ve followed those directions, you’re ready to install Grid Portlets.

```
> ant install
> $CATALINA_HOME/bin/startup.sh
```
The Grid Portlets Group

- After installing Grid Portlets and (re)starting your web application server (Tomcat), make sure to add the Grid Portlets group to your user profile.
Make it a default group

- We recommend setting the Grid Portlets group to be a “default group” so all users will see our grid portlets by default. See the “Administration/Groups” tab.
Our Grid Portlets

- Resource Registry Portlet (Grid/Registry Tab)
- Resource Browser Portlet (Grid/Resources Tab)
- Credential Retrieval Portlet (Grid/Credentials Tab)
- File Browser Portlet (Grid/Files Tab)
- Job Submission Portlet (Grid/Jobs Tab)
- ...and more coming...
• Administrators can use the Resource Registry Portlet to specify what resources to make available to portal users.
Resource Browser Portlet

- Users can then see the resources that have been made available with the Resource Browser Portlet.
- The Resource Browser Portlet provides a simple collection of user interfaces for viewing resources.
- As we’ll see later, the Resource Browser Portlet also supports the ability to display custom user interfaces for viewing resources.
Viewing a resource

- Grid Portlets provides the ability to display custom HTML and images in its genericre.
Credential Retrieval Portlet

Users can retrieve credentials from a MyProxy credential repository and enable their credentials for “single sign-on” to computing resources at login time.
File Browser Portlet

- Users can browse files on remote computing resources in a manner similar to how they might browse files on their desktop. We have made it relatively simple to create new directories, transfer and delete files all with simple HTML interfaces.
Support for logical files

• Support for logical files in one browser interface!
So, instead of “registering” a physical file, they
would “upload” or “import” a file.
Job Submission Portlet

- Users define jobs using the job submission portlet. The Job Submission Portlet presents a generic interface for specifying a job.
- As we’ll see later, Grid Portlets supports the ability to add custom user interfaces for specifying and monitoring jobs.
Selecting a job resource

Moreover, the Job Submission Portlet naturally supports wizards. The generic job user interface that come with Grid Portlets is implemented as wizard.
Viewing job history & output

Job Submission Portlet

<table>
<thead>
<tr>
<th>Job Id</th>
<th>Description</th>
<th>Job Type</th>
<th>Resource</th>
<th>Status</th>
<th>Date Submitted</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://peyote.aei.mpg.de:23683/27046/1103033836/">https://peyote.aei.mpg.de:23683/27046/1103033836/</a></td>
<td>Cactus Simulation Generic Application</td>
<td>peyote.aei.mpg.de</td>
<td>Job failed with error code 5; the executable does not exist</td>
<td>Dec 14, 2004 3:17:14 PM</td>
<td></td>
</tr>
</tbody>
</table>

Job Submission Portlet

Generic Application

Job Id: https://peyote.aei.mpg.de:23683/29923/1103034259/
Job Description: Cactus Simulation
Job Status: Job is active with message success

Job Resource: peyote.aei.mpg.de
Job Scheduler: Cactus Scheduler
Job Queue: GridQueue
Date Submitted: Tuesday, December 14, 2004 3:24:19 PM CET
Last Changed: Tuesday, December 14, 2004 3:24:20 PM CET
Date Ended: Tuesday, December 14, 2004 3:24:20 PM CET

Job Spec

cactus version: 4.0.14
compile date: Feb 10 2004 (11:26:52)
run date: Dec 14 2004 (13:23:20)
run host: peyote
executable: /home/brad/kex/.
parameter file: ./KexHistoryRun10.par

Copyright The Authors
All Rights Reserved.
Configuring Grid Portlets

- Configuration amounts to specifying what resources Grid Portlets makes available to users.
- The first thing you should do is specify what MyProxy resource your portal will use for retrieving / storing Grid credentials (next slides).
- If you are there are information resources on your Grid (MDS2, MDS4, etc), then you can specify where those resources are located to setup resource discovery.
  - Otherwise, you should make sure to specify manually the hardware and service resources that make up your Grid...
How To Specify Resources

- There are 2 ways to specify resources in the Grid Portlets web application.
  - Online with the Resource Registry Portlet, shown earlier, see the Grid Portlets Administrator Guide or the Resource Browser Portlet help mode for more information.
  - Offline with ./WEB-INF/Resources.xml. After editing this file, you must then (re)deploy the Grid Portlets web application. More information about how to use this file is in the Grid Portlets Administrator Guide, an example will follow...
Resources.xml

- ./WEB-INF/Resources.xml can be used to describe an initial set of resources to make available. If there are information resources provided (see next slides), then other resources can be discovered.
- Resource “mappings” (the names and attributes they take) are described in ./WEB-INF/mappings/resource-mappings.xml
- More information can be found in the Grid Portlets Administrator Guide.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<grid-resources>
  <!-- DESCRIBE RESOURCES HERE -->
  <!--!-- DESCRIPT RESOURCES HERE -->
</grid-resources>
</xml>
```
MyProxy Resource

- Specify the MyProxy resource your Grid portal will make available to users. Our credential retrieval service works with just one (active) MyProxy resource because we take the view that users should not be able to use just any MyProxy they want... We take security very seriously!

```xml
<hardware-resource label="GridLab MyProxy"
    description="Hosts The GridLab MyProxy"
    hostname="myproxy.gridlab.org">
  <myproxy-resource label="MyProxy"
    description="Online Credential Repository"
    portalCertFile="/etc/grid-security/hostcert.pem"
    portalKeyFile="/etc/grid-security/hostkey.pem"/>
</hardware-resource>
```
Information Resources

- Here is an example of specifying GRIS resources to gather information about two computing clusters in the GridLab Virtual Organization.

```xml
<hardware-resource label="Peyote"
  description="AEI Peyote Computing Cluster"
  hostname="peyote.aei.mpg.de">
  <gris-resource basedn="Mds-Vo-name=gridlab,o=grid"/>
</hardware-resource>

<hardware-resource label="Helix"
  description="LSU Helix Computing Cluster"
  hostname="helix.bcvc.lsu.edu">
  <gris-resource basedn="Mds-Vo-name=gridlab,o=grid"/>
</hardware-resource>
```
Developing With Grid Portlets

• The Grid Portlets was designed for reuse.
• Grid Portlets provides a high-level model of the Grid that:
  • Abstracts developers from particular Grid technologies and infrastructure (a high-level API).
  • Is extensible, can build upon its concepts to provide more complex services, resources and tasks.
  • Supports multiple implementations of its API.
  • Is configurable at runtime (more work needed on portlet side, but API supports this already)
• This design is achieved both in our API, implemented as portlet services, and in our portlets, which contain reusable action components.
Grid Portlets Architecture

- Grid Portlets has a layered architecture. Its portlets are composed of reusable “Action Components” that make use of a collection of reusable “Portlet Services”. This collection of portlet services presents an abstract API to which the relevant Grid resources are adapted. Grid Portlets supports multiple Grid architectures / platforms in this manner.
Resource! The Core Concept

- The core concept in Grid Portlets is “resource”.
- A resource is defined simply anything that can be utilized. A resource has a distinguished name that uniquely identifies it, has various attributes and can contain other child resources.
- This simple definition goes a long way...
Basic Idea...

Resource

- Hardware Resource
- Service Resource
- Resource Account
  - Web Service Resource
  - Hardware Account
Hardware Resource Model

- Hardware resources are utilized for performing computations, storing files, etc.
- Hardware resources can contain:
  - Service resources (such as a Grid FTP resource)
  - Software resources (such as Globus client tools)
  - Hardware accounts (allowing people to login and perform operations in a shell).
Job Resource Model

- Job resources (such as Globus Gatekeepers) are service resources that enable remote clients to submit jobs to computing resources.
- Job resources provide access to job schedulers (such as PBS).
- Job schedulers contain queues with different resource capabilities and usage policies.
Some Job Resources

- In the GridLab Project, we modeled two types of service resources as job resources, both of which have different capabilities for supporting job submission.
- GTX Gatekeepers provide the ability to submit jobs directly to job managers running on hardware resources using the GRAM protocol.
- GridLab Resource Manager Services (GRMS)s are web services that provide resource brokering capabilities and support job submission.
Tasks modeled as resources

- Resources can represent anything we want.
- For example, we define a task as a resource for performing a particular operation. Tasks have a well-defined lifecycle, with specific task statuses representing that lifecycle.
Simple Task Diagram

- Resource
  - Task
    - Job
    - File Task
      - File Transfer
      - File Deletion
Key Grid Portlet Services

• Grid Portlets delivers its API via portlet services, the following of which were used to build our existing Grid portlets:

  ● Credential Manager Service
  ● Credential Retrieval Service
  ● File Browser Service
  ● Logical File Browser Service
  ● Job Submission Service
  ● Resource Registry Service
Credential Manager Service

• The CredentialManagerService provides methods for listing, adding, editing and deleting “credential contexts”, represented by the CredentialContext interface, which describes a GSSCredential with a unique DN as belonging to one user only.

• The CredentialManagerService provides methods for keeping track of the GSS credentials that are currently active for a given user. All portlets and portlet services that require active credentials for performing operations for a given portal user on the Grid should use this service for those credentials.
Credential Retrieval Service

- The **CredentialRetrievalService** is an interface for retrieving credentials. It supports GSS credential retrieval from “credential repository resources” that implement the **CredentialRepository** interface.

- The **CredentialRetrievalService** provides methods for listing, adding, editing and deleting “credential retrieval contexts”, represented by the **CredentialRetrievalContext** interface.

- There is currently one implementation of the **CredentialRepository** interface, the **MyProxyResource**.
File Browser Service

- The **FileBrowserService** provides methods for creating “file browsers”, represented by the **FileBrowser** interface, to remote “file resources”, represented by the **FileResource** interface, and provides methods for submitting “file browser tasks”, represented by the **FileBrowserTask** interface.

- A **FileBrowser** can maintain a “stateful” connection to a file resource and provides methods for listing files, changing directories, creating directories, and other basic file operations.
Logical File Browser Service

- The *LogicalFileBrowserService* extends *FileBrowserService* for creating “logical file browsers”, represented by the *LogicalFileBrowser* interface, to logical file resources, represented by the *LogicalFileResource* interface.
Job Submission Service

- The JobSubmissionService provides methods for creating job specifications, represented by the JobSpec interface, and submitting job specifications to generate jobs, represented by the Job interface.

- Additionally, JobSubmissionService includes methods for listing the job resources, represented by the JobResource interface, to which it supports job submission.
Resource Registry Service

• The **ResourceRegistryService** provides methods for listing, adding, editing and deleting resources, represented by the **Resource** interface. All portlets and portlet services that require handles to resources should use this service.

• The **ResourceRegistryService** can also use “resource provider services” (**ResourceProviderService**) to “discover” or “provide” resources automatically.

• For example, the **Mds2ResourceProviderService** polls available GRIS resources for information about hardware resources, job queues, jobs, etc.
Building a custom portal...

- Override the GridSphere / GridPortlets default layouts, themes (and localization settings).
- Use our Grid “portlet services” (next slides) to authenticate, monitor and perform tasks on remote computing resources.
- Develop custom “resources” (next slides) for gathering information, accessing files, submitting jobs, etc.
Example: Cactus Code Portlets

- Cactus Code Portlets is a portlet application that builds upon our Grid portlet services to make it easy to deploy and execute Cactus applications on the Grid.
- Our approach to Grid portal development is to offer users a layered architecture. There should be no reason why you have to implement job submission and file management functionality.
- GridSphere + Grid Portlets provides the base functionality with which you can develop interfaces tailored to particular application needs.
Cactus Installation Manager Portlet

- For example, the Cactus Installation Manager Portlet enables users to deploy Cactus software onto remote resources.
- It uses the `ResourceRegistryService` to display a list of resources, the `FileBrowserService` to get access to directories on those resources and the `JobSubmissionService` to submit jobs to checkout Cactus code from CVS repositories onto those resources.
Customizing Grid Portlets

- Grid Portlets can be extended to utilize specific Grid services and technologies. This can be achieved by developing custom “resources” (next slides) for gathering information, accessing files, submitting jobs, etc.

- Grid Portlets also supports customization at the portlet level. You can write custom interfaces for viewing resources (“Resource Profiles”) and specifying jobs (“Job Profiles”) that plug into the Resource Browser Portlet and Job Submission Portlet respectively.
Create a grid portlet project

Inside the gridportlets project, create a new grid portlet project.

> cd projects/gridportlets
> ant new-project
  [echo] Creating a New Grid Portlet Project
  [input] Please enter a Project Name this will be used for your portlet web application and should be lowercase e.g. mygridportlets
  mygridportlets
  [input] Please enter a Project Title e.g. My Grid Portlets
  My Grid Portlets
  [echo] Creation of new portlet project My Grid Portlets in projects/mygridportlets is complete.
  [echo] Please edit src/ webapps/ and webapps/WEB-INF/{web.xml, layout.xml, PortletServices.xml} appropriately
  [echo] Please place portlets in src/**/portlets/ directory and service in src/**/services directory for proper compilation!
> cd ../mygridportlets
> ls
build.properties config src
build.xml lib webapp
Grid Portlet Projects

- Grid Examples - Being upgraded now
- GT3 Portlets - No longer supported
- GT4 Portlets - Being upgraded now
- GLite Portlets - Under development now!
- HPC Europa Portlets - Under development now!
- GridScreen Portlets - For gathering and displaying information from GridScreen
- Mercury Portlets - For monitoring resources with the Mercury monitoring service
- CactusCode Portlets - For monitoring Cactus simulations
Custom Information Resources

• Grid Portlets does not say much about how information is provided to the portal.

• However, the general approach we adopt is to:
  • Use information resources to discover and/or populate the GridPortlets Resource Registry (backed by a SQL database) with basic information about computing resources, job queues, user accounts, etc.
  • Then query the ResourceRegistryService, rather than information resources directly, in the portlets.

• This is partly historical (MDS used to be quite slow for example).

• And practical...
  • Both in terms of performance
  • And in terms of reuse of information among portlets and “portlet services”.
Information Resources (part 1)

- Implement the `InformationResource` interface. Your implementation should include the necessary code to connect to the information resource it represents and to query that resource for information.
- For example, the `Mds2Resource` and its subclasses contain code for connecting to MDS 2 services.
- Add a *hibernate mapping* for the `InfoResource` class to your project’s `webpp/WEB-INF/gridportlets/persistence/Resource.hbm.xml`.
- Add a *castor mapping* for the `InfoResource` class to your project’s `webapp/WEB-INF/gridportlets/mappings/ResourceMappings.xml`. 
Information Resources (part 2)

- Implement the `ResourceProviderService` interface. Your implementation will be responsible for setting up communication with any information resources you wish to utilize in your portal.
- For example, the `Mds2ResourceProviderService` starts a thread that polls the GRIS resources it finds in the resource registry every x minutes for information about computing hosts, job queues, etc.
- On the other hand, if your information resource supports a call back mechanism or notification, then you should take advantage of this feature.
- Add a portlet service entry to your project’s `webapp/WEB-INF/PortletServices.xml`. 
Custom Job Resources

• The first problem we worked on in our parent project was job submission, yet it remains the most difficult.
• There are a variety of services and approaches to job submission, and we want to support as many as possible.
• Yet, most are built upon similar concepts and therefore we would like to use the same user interfaces wherever possible.
Job Resources (part 1)

- Extend from the `BaseJob` and `BaseJobSpec` classes. Your `JobSpec` must implement `marshalToString()` to produce the appropriate job specification string your job resource requires.
- For example, `GramJobSpec` marshals to a string containing RSL, which is then submitted to the appropriate `GramJobResource` (based on the hostname attribute of the job specification).
- Add a `hibernate mappings` for the Job and JobSpec classes to your project’s `webpp/WEB-INF/gridportlets/persistence/Task.hbm.xml` and `TaskSpec.hbm.xml` files respectively.
Job Resources (part 2)

- Implement the JobResource interface. Your implementation should include the necessary code to connect to the job resource and submit job specifications to it.
- Add a hibernate mapping for the JobResource class to your project’s webpp/WEB-INF/gridportlets/persistence/Resource.hbm.xml.
- Add a castor mapping for the JobResource class to your project’s webapp/WEB-INF/gridportlets/mappings/ResourceMappings.xml.
Job Resources (part 3)

- Extend `AbstractJobSubmissionService`. Your implementation should include the code for finding the appropriate `JobResource` based on the `JobSpec` provided to it.
- For example, `GramJobSubmissionService` finds the `JobResource` that has the same host as the provided in the `JobSpec`.
- Add a portlet service entry to your project’s `webapp/WEB-INF/PortletServices.xml`. 
## Job Submission Portlet

### Job Setup

<table>
<thead>
<tr>
<th>Type of job to submit</th>
<th>Job submission service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic user interface for submitting jobs</td>
<td>Globus Resource Allocation Manager Service</td>
</tr>
<tr>
<td>Cactus Toolkit user interface for submitting jobs</td>
<td>GridLab Resource Management System</td>
</tr>
<tr>
<td>Shell Script Application</td>
<td>MMJFS Resource Management System</td>
</tr>
</tbody>
</table>

Select the type of job you would like to submit and the service you would like to use to submit the job.

**The Job Submission Portlet provides:**

- The ability to plug in custom-defined user interfaces
- The ability to plug in different job submission services.
And more...

- In Grid Portlets, we have our own UI development model (though we may migrate to JSF at some point in the future).
- We have been using this model to develop our plugin design for supporting custom interfaces in the Resource Browser Portlet and the Job Submission Portlet.
- Our plugin model could (should) be much simpler, and for this reason we have not really advertised this aspect of Grid Portlets.
- However, we are more than happy to help set developers up with template projects for adding custom UI to Grid Portlets.
- This will be a major feature in future releases!
• One can define a custom resource profile for displaying custom information about resources of a particular type (Hardware Resource or Job Queue).

• All it is really is a pointer to a particular UI component class.... described in gridportlets/WEB-INF/ResourceProfiles.xml.

```xml
<resource-profile
    key="GeneralHardwareResource"
    description="General">
    <resource-type>org.gridlab.gridsphere.services.resource.HardwareResource</resource-type>
    <resource-list-page>org.gridlab.gridsphere.services.ui.resource.browser.profiles.HardwareResourceListViewPage</resource-list-page>
</resource-profile>
```
Example: GridScreen Profile

- Here we see the GridScreen user interface for viewing a resource. This user interface displays information gathered from an information service developed at the Albert Einstein Institute.
**Custom Job “Profiles”**

- Similarly, one can define a custom job profile for displaying custom interfaces for submitting and monitoring jobs.
- All it is really is a pointer to a particular UI component classs.... described in gridportlets/WEB-INF/JobProfiles.xml.

```xml
<job-profile
    key="org.gridlab.gridsphere.services.ui.job.generic"
    description="Generic Application">
    <job-spec-view-page>org.gridlab.gridsphere.services.ui.job.generic.GenericJobSpecViewPage</job-spec-view-page>
    <job-spec-edit-page>org.gridlab.gridsphere.services.ui.job.generic.GenericApplicationSpecEditPage</job-spec-edit-page>
    <job-spec-edit-page>org.gridlab.gridsphere.services.ui.job.generic.GenericJobSpecConfirmPage</job-spec-edit-page>
</job-profile>
```
Summary Tips

- Override the GridSphere / GridPortlets default layouts, themes (and localization settings).
- Use our Grid “portlet services” (next slides) to authenticate, monitor and perform tasks on remote computing resources.
- Develop custom “resources” (next slides) for gathering information, accessing files, submitting jobs, etc.
- Develop portlets tailored to the particular needs of your target applications / communities using our portlet development tools, JSF, Spring MVC, or any model of your choice.
- The important thing is to adopt an MVC model.
- If you do choose to adopt the Action Component Model (provided in Grid Portlets), you can then reutilize the various user interface components from Grid Portlets is composed.
- Moreover, you can develop custom user interfaces for the Resource Browser and Job Submission portlets.
More Information

- More information can be found in the Grid Portlets guides that get generated with the “ant docs” command. There are 3 guides available:
  - **Grid Portlets Administrator Guide**
    - Describes how to install and configure Grid Portlets.
  - **Grid Portlets User Guide**
    - Describes how to use Grid Portlets.
  - **Grid Portlets Developer Guide**
    - Describes how to develop custom Grid portal applications.
Conclusion

- Grid Portlets contains the Grid functionality in the GridSphere Project. Together with the GridSphere portlet container it provides a framework for developing Grid portals.
- The GT2 implementations are in working order and we are continually adding enhancements.
- We experienced a few hiccups this summer, but we will be able to provide clear examples of how to extend Grid Portlets in the very near future.
- We will make announcements on our website at www.gridsphere.org as the status of Grid Portlets changes.
- Feel free to try it out and send us comments.