GridSphere: A Portal Framework

Jason Novotny novotny@aei.mpg.de
Michael Russell russell@aei.mpg.de
Oliver Wehrens wehrens@aei.mpg.de

Albert Einstein Institute
Introduction

- Web based portals provide a customizable user environment.
- Portals act as a gateway between users and services.
- Portal standards have evolved to enable separation of business functions from application server.
- Portlets are the driving concept to deliver reusable web functionality.
Portal standards

- **JSR 168 Portlet API ratified August 2003**
  - Similar to Servlet API in providing reusable web applications
  - Ratified by vendors including BEA, Sun, IBM, Oracle, Plumtree and others...

- **WSRP (Web Services for Remote Portlets) ratified by OASIS committee**
  - Specifies how web services can be consumed by standards compliant portals

- **Java Server Faces ratified**
  - Specifies an event based user interface for web presentation development
Portlets

- Standardized packaging model makes it easier to share portlet applications among portal vendors.
- Supports window states and mode settings like desktop environment.
Portlet Implementations

- Current vendors offer compliant portals or fix-it packs...
  - IBM WebSphere
  - Oracle 10g Portal
  - Sun Application Server
- Jakarta Pluto
  - open-source implementation of Portlet API
  - Used by Jetspeed, uPortal, wsrp4j
- Jetspeed 2 still not out...
- Emerging open source projects Exo and Liferay
- And GridSphere...
GridSphere 2.0 Feature List

- Portlet API passed Sun TCK and is 100% JSR 168 compliant
- Additional Portlet API implementation nearly fully compatible with IBM's WebSphere 4.2. (GridSphere 2.0)
- Support for the easy development and integration of new portlet applications
- Higher-level model for building complex portlets using visual beans and the GridSphere User Interface (UI) tag library.
- Flexible XML based portal presentation description can be easily modified to create customized portal layouts.
- Built-in support for Role Based Access Control (RBAC) separating users into guests, users, admins and super users.
- Sophisticated portlet service model that allows for creation and reusability of new business logic with support for persistence of data
GridSphere 2.0 Feature List ...

- Persistence of data provided using Hibernate OQL for database support
- Integrated Junit/Cactus unit tests for complete server side testing of portlet services including the generation of test reports.
- GridSphere core portlets:
  - Login, Logout, Locale settings
  - Profile personalization and Layout customization
  - Administration portlets for creation of users, groups, portlet management and portal layout customization
- Localization support in the Portlet API implementation and portlets support French, English, German, Czech, Polish, Hungarian and Italian.
- Open-source and 100% free! :-}
The GridSphere portlet container is implemented as a web application and requires a hosting environment such as the Jakarta Tomcat container.

Many additional libraries are used and deployed to the servlet container during installation.
The GridSphere *portlet container* loads and instantiates portlet classes.
Portlet, GenericPortlet
- All portlets must implement Portlet interface or subclass from GenericPortlet that define portlet lifecycle methods

PortletConfig
- Provides portlet with its initial configuration

RenderRequest, ActionRequest
- Encapsulates the a request sent by client

RenderResponse, ActionResponse
- Represents response to client

PortletContext
- Defines portlets view of the portlet container in which portlet is running

PortalContext
- Provides vendor information and portal properties
PortletRequest additions

- PortletPreferences
  - Contains user-specific persistent data
- PortletSession
  - Holds user-specific information needed to provide personalized view
- PortletWindow
  - Defines “window” in which portlet is displayed
- PortletMode
  - Defines “mode” in which portlet is operating
- User information defined in request attribute as a map of key value pairs
Portlet Modes

- **View**
  - The standard view of a portlet on a Web page

- **Edit**
  - Allows portlet to capture user-specific parameterization, which leads to personalized view of the portlet

- **Help**
  - A portlet should provide online-help with this mode

- **Configure (not required by spec. but supported)**
  - Allows a portlet to display its own configuration mode
  - Generally, you want to restrict access to this mode to portal administrators
A Portlet Deployment Descriptor provides the portlet container with portlet configuration information.

- Defines a collection of portlet definitions as an XML schema e.g. portlet.xml.

- A portlet definition defines configuration information valid for a single portlet/servlet for all users:
  - Specifies portlet deployment attributes.
  - Supported markups, portlet modes and window states.
  - Display name and title information.
  - Security role information.
  - Supported locales.
  - Time for portlet to expire cached output.
<portlet>
  <description xml:lang="en">A TextMessagingPortlet</description>
  <portlet-name>TextMessagingPortlet</portlet-name>
  <display-name xml:lang="en">TextMessaging</display-name>
  <portlet-class>org.gridlab.gridsphere.extras.portlets.textmessaging.TextMessagingPortlet</portlet-class>
  <expiration-cache>60</expiration-cache>
  <supports>
    <mime-type>text/html</mime-type>
    <portlet-mode>help</portlet-mode>
  </supports>
  <supports>
    <mime-type>text/wml</mime-type>
    <portlet-mode>edit</portlet-mode>
    <portlet-mode>help</portlet-mode>
  </supports>
  <supported-locale>en</supported-locale>
  <portlet-info>
    <title>TextMessaging</title>
    <short-title>TextMessaging</short-title>
    <keywords>textmessaging</keywords>
  </portlet-info>
  <security-role-ref>
    <role-name>GUEST</role-name>
    <role-link>auth-user</role-link>
  </security-role-ref>
</portlet>
GridSphere Lifecycle

client request

init()

action()

render()

Core services

Layout Engine

processAction()

doView()

TabbedPane
Tab
Frame
TitleBar
etc...

Users
Layout
Portlets

GridSphere Servlet
Portal uses header and double layer tabbed pane to organize content

Portal layout specified as XML descriptor:

```xml
<portlet-tab>
  <title>Examples</title>
  <portlet-tabbed-pane style="sub-menu">
    <portlet-tab>
      <title lang="en">Hello</title>
      <portlet-panel>
        <grid-layout>
          <portlet-frame>
            <portlet-class>org.gridlab.gridsphere.portlets.examples.HalloWelt.1</portlet-class>
          </portlet-frame>
        </grid-layout>
      </portlet-panel>
    </portlet-tab>
  </portlet-tabbed-pane>
</portlet-tab>
```
Presentation Components

- Layout component library similar to Java AWT or Swing
- Basic layout components:
  - PortletTabbedPane
  - PortletTab
  - PortletFrame
  - PortletTitleBar
  - PortletPanel
  - PortletGridLayout
- Some components act as containers for other components e.g. PortletPanel
- All components have `init`, `action` and `render` lifecycle methods
- Follows the Composite Design Pattern
- Components are marshalled/unmarshalled to XML using Castor libraries
GridSphere provides value-added UI JSP tag library

Goal is to minimize HTML usage

- UI tags can provide platform independence e.g. support HTML and WML

Example tag usage in JSP:

```jsp
<%@ taglib uri="/portletUI" prefix="ui" %>
<%@ taglib uri="http://java.sun.com/portlet" prefix="portlet" %>
<portlet:defineObjects/>

<ui:form action="login">
  <ui:inputfield name="username" size="8" maxlength="20"/>
  <ui:passwordfield name="password" size="8" maxlength="20"/>
</ui:form>
```
Additional “container” tags make it possible to quickly create interfaces that hide CSS/HTML from presentation design

```html
<%@ taglib uri="/portletUI" prefix="ui" %>
<%@ taglib uri="http://java.sun.com/portlet" prefix="portlet" %>

<portlet:defineObjects/>
  <ui:panel>
    <ui:messagebox beanId=displayMsg"/>
    <ui:frame>
      <ui:tablerow>
        <ui:tablecell width="50%">
          <ui:actionlink action="doSomething"/>
        </ui:tablecell>
        <ui:tablecell/>
      </ui:tablerow>
    </ui:frame>
  </ui:panel>
```
Visual Bean Model

- For each visual tag, there is a visual bean counterpart that provides access to the visual component from the portlet.

- A visual bean is obtained from a FormEvent object

```java
public void doViewUserFiles(RenderFormEvent event) throws PortletException {
    log.debug("in LoginPortlet: doViewUser");
    RenderRequest request = event.getRenderRequest();
    ListBoxBean lb = event.getListBoxBean("filelist");
    lb.clear();
    String userid = (String)request.getAttribute("userid");
    String[] list = userStorage.getFileList(userid);

    ...
}
```
A “portlet service” moves logic from a portlet to a reusable service that may be used by other portlets.

**PortletService** is the base interface for all portlet services.

PortletService instances are created by a **PortletServiceFactory**.

Similar to Portlets, PortletService objects are configured at initialization with a **PortletServiceConfig** object.

A service may be created per instance or cached.

PortletServices can make use of GridSphere provided persistence classes.

Similar to Spring service framework.
<portlet-services>
  <service>
    <name>Portlet Manager Service</name>
    <description>Provides Administration Capabilities for Portlet Web Applications</description>
    <interface>org.gridlab.gridsphere.services.core.registry.PortletManagerService</interface>
    <implementation>org.gridlab.gridsphere.services.core.registry.impl.PortletManagerServiceImpl</implementation>
  </service>

  <service>
    <name>Login Service</name>
    <description>Provides Login Capabilities</description>
    <interface>org.gridlab.gridsphere.services.core.user>LoginService</interface>
    <implementation>org.gridlab.gridsphere.services.core.user.impl>LoginServiceImpl</implementation>
  </service>
</portlet-services>
GridSphere Framework includes basic support for persistent objects using PersistenceManager singleton.

PersistenceManager uses open-source Hibernate libraries which provides mechanisms for mapping objects to SQL and an object query language (OQL).

Hibernate supports many databases including hsqldb, MySQL, Postgres, Oracle, etc.

Every portlet app manages its own database keeping data independent.
GridSphere Security

- Access control based on 4 defined roles within a group:
  - Guest < User < Admin < Super
- A guest is anyone that has not logged in
- An admin has permissions to manage users in the group and edit group layout
- Super is the portal administrator
- A group defines a set of deployed portlets with access restrictions
- Users can be in multiple groups, but can only add portlets that they have access to
Core User Portlets

- Login/Logout portlet
  - Enables user to logon/logout
  - Allows user to refresh password if forgotten
  - Configurable option enables new users to request an account.

- Locale portlet
  - Simple locale chooser in the portal banner offers support for 7 languages

- User profile portlet
  - Enables users to configure personal information e.g. name, email, locale, preferences

- Layout configuration portlet
  - Enables users to customize their layout by creating new tabs which portlets can be easily added to.
Core Administrative Portlets

- **User Manager Portlet**
  - Enables admins to create/delete/edit portal users

- **Group Manager Portlet**
  - Enables admins to add/remove users to/from portlet groups
  - Enables admins to select whether a group is public or private (public means anyone can join, private requires an administrator approval)

- **Portlet Manager Portlet**
  - Enables admins to start, stop, or redeploy a portlet application

- **Layout Manager Portlet**
  - Enables admins to edit existing group layouts
GridSphere provides “extras” portlet application:

- Text Messaging portlet communicates to IM users
- Photo album portlet allows users to upload and display photos
- Poll portlet to create polls and display results
- Chart portlets demonstrating a chart service that uses JFreeChart to display plots, timeseries graphs, etc.
- Commander portlet manages secure portal filesystem to upload, download and transfer files
Future directions

- Integrate JSF into portlet development
- Develop a WSRP “consumer” portlet using WSRP4J
- Support JSR 170, content management
- Further improve upon the GridSphere portlet development model
- Continue seeking collaborations with other groups requiring portal frameworks