



Development of standards-based grid portals, Part 3: WSRP and the future of grid portals

Level: Intermediate

Xiaobo Yang (x.yang@dl.ac.uk), Software Developer, Consultant

Xiao Dong Wang (x.d.wang@dl.ac.uk), Senior Software Developer, Consultant

Robert Allan (r.j.allan@dl.ac.uk), Group Leader, Consultant

19 Apr 2007

Built on top of grid middleware, grid portals act as gateways to the grid because they smooth the learning curve of using grid. In the first of this three-part "[Development of standards-based grid portals](#)" series, we give an overview of grid portals, focusing on today's standards-based (JSR 168 and Web Services for Remote Portlets (WSRP) V1.0) second-generation grid portals. In Part 2, we develop three portlets to illustrate how a grid portal can be built using JSR 168-compliant portlets. And here in Part 3, we discuss the application of WSRP and the future of grid portals.

Investigation of WSRP for use in grid portals

When we talked about adoption of SOA for portal development in [Part 1](#), one of the key concepts is that presentation logic can be integrated with business logic and provided as a service. The WSRP V1.0 specification standardises how portlets can be exposed to remote portlet containers using Web services technology. In WSRP, a producer is modelled as a portlet container within which portlets reside. Such a producer acts as a service provider, while at the same time, various clients are able to consume this service through which portlets are evoked. To clearly understand how WSRP works, read the WSRP V1.0 Primer, in addition to the V1.0 specification itself (see [Resources](#)).

WSRP4J: Open source WSRP V1.0 implementation

Although the Apache WSRP4J project initiated by IBM® is still in its incubation phase, the WSRP4J producer is, in our experience, more practical than WSRP producers from other open source portal frameworks, such as the eXo platform and Liferay. Therefore, we selected WSRP4J to illustrate how grid portlets we developed in this series can be reused in other systems, allowing WSRP-equipped consumers to benefit from having grid tools without local development.

The WSRP4J producer uses Pluto, the Reference Implementation of JSR 168 provided by Apache, as its portlet container. It provides four interfaces defined by the WSRP V1.0 specification: ServiceDescription, Markup, Registration (optional), and PortletManagement (optional). A consumer can contact the ServiceDescription interface to retrieve descriptions of a producer, including information like the portlets it holds and locale support. The consumer can then contact the producer to access a particular portlet. The Markup interface is responsible for accepting requests and returning markup fragments (output of the portlet).

Four interfaces defined by WSRP V1.0

ServiceDescription interface is a required interface that defines an operation for acquiring a producer's metadata, enabling a consumer to discover services provided by that producer.

Markup interface is the second required interface to define operations for getting the markup from a portlet and processing user interactions with that markup. It also handles HTTP cookies.

Registration interface is an optional interface, which enables a consumer to register at the producer by defining operations for establishing, updating, and destroying a registration.

Publish and consume example grid portlets

This article will not describe in detail how to install and configure a WSRP4J producer. In general, to deploy our portlets inside a WSRP4J producer, you first need to copy the compiled war file to Tomcat, within which the WSRP4J producer is deployed. Remember to rename the war file to grid.war. You are required to register the portlets inside a file under the WSRP4J producer directory called portletentityregistry.xml, as shown in Listing 1. (See [Resources](#) for more information.)

PortletManagement interface is another optional interface, which covers life cycle and properties of portlets. It defines operations for getting portlet metadata, cloning portlets for further customisation, and interacting with the property interface.

Listing 1. Registering portlets inside the WSRP4J producer

```
<application id="grid">
  <definition-id>grid</definition-id>
  <portlet id="LdapBrowserPortlet">
    <definition-id>grid.LdapBrowserPortlet</definition-id>
  </portlet>
  <portlet id="JobSubmissionPortlet">
    <definition-id>grid.JobSubmissionPortlet</definition-id>
  </portlet>
  <portlet id="ProxyManagerPortlet">
    <definition-id>grid.ProxyManagerPortlet</definition-id>
  </portlet>
</application>
```

Once portlets are deployed inside the WSRP4J producer, a consumer is required for accessing those portlets. To publish a remote portlet inside uPortal, you need to define the WSRP4J producer interfaces introduced above, together with the portlet handle, which is the same as `definition-id` above. Axis TCP Monitor is a useful tool to monitor communications between a WSRP consumer and producer. Figures 1a and 1b are screenshots that illustrate consumption of the LDAP Browser and Proxy Manager portlets inside uPortal and Sakai, respectively. (Sakai is an open-source e-Learning framework, see [Resources](#) for more information.) uPortal has its own WSRP consumer, while Sakai is equipped with a WSRP consumer we developed based on the WSRP4J consumer ProxyPortlet.

Figure 1a. LDAP Browser portlet running inside uPortal through WSRP

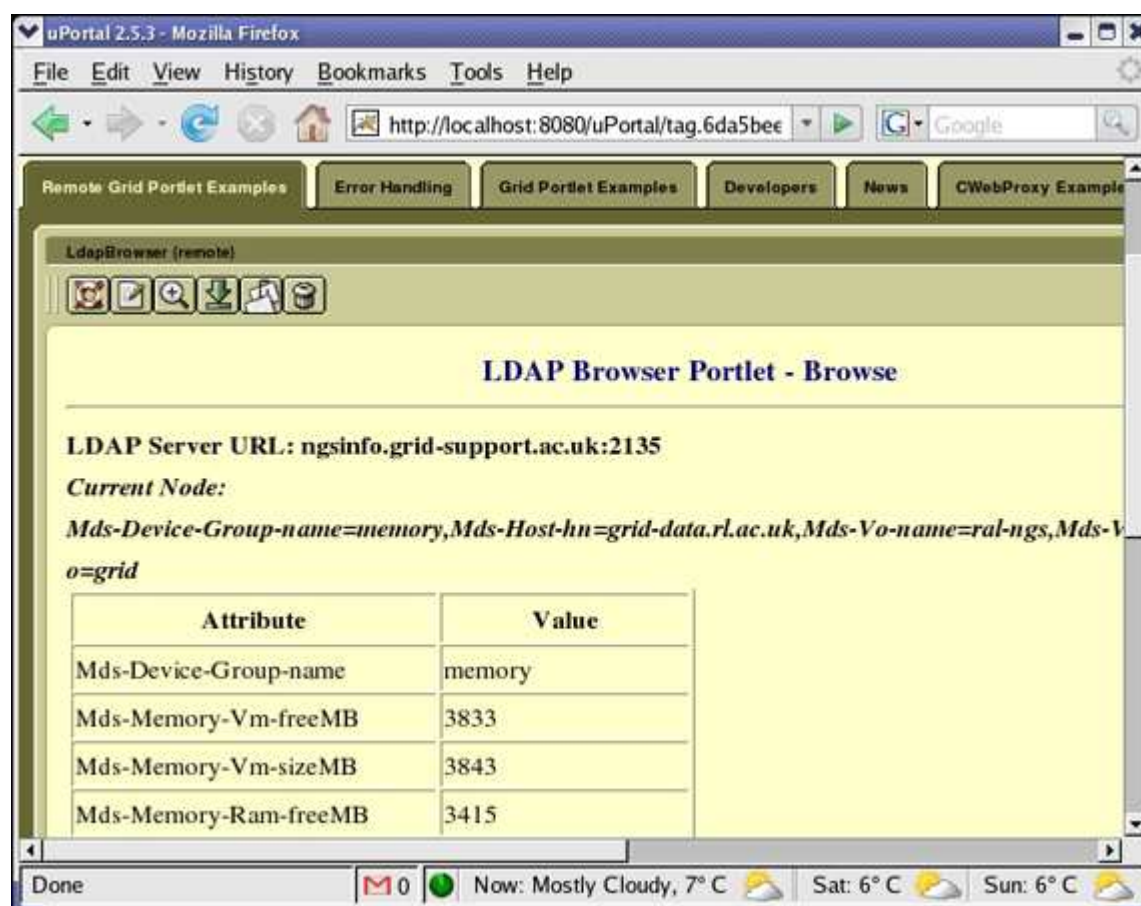
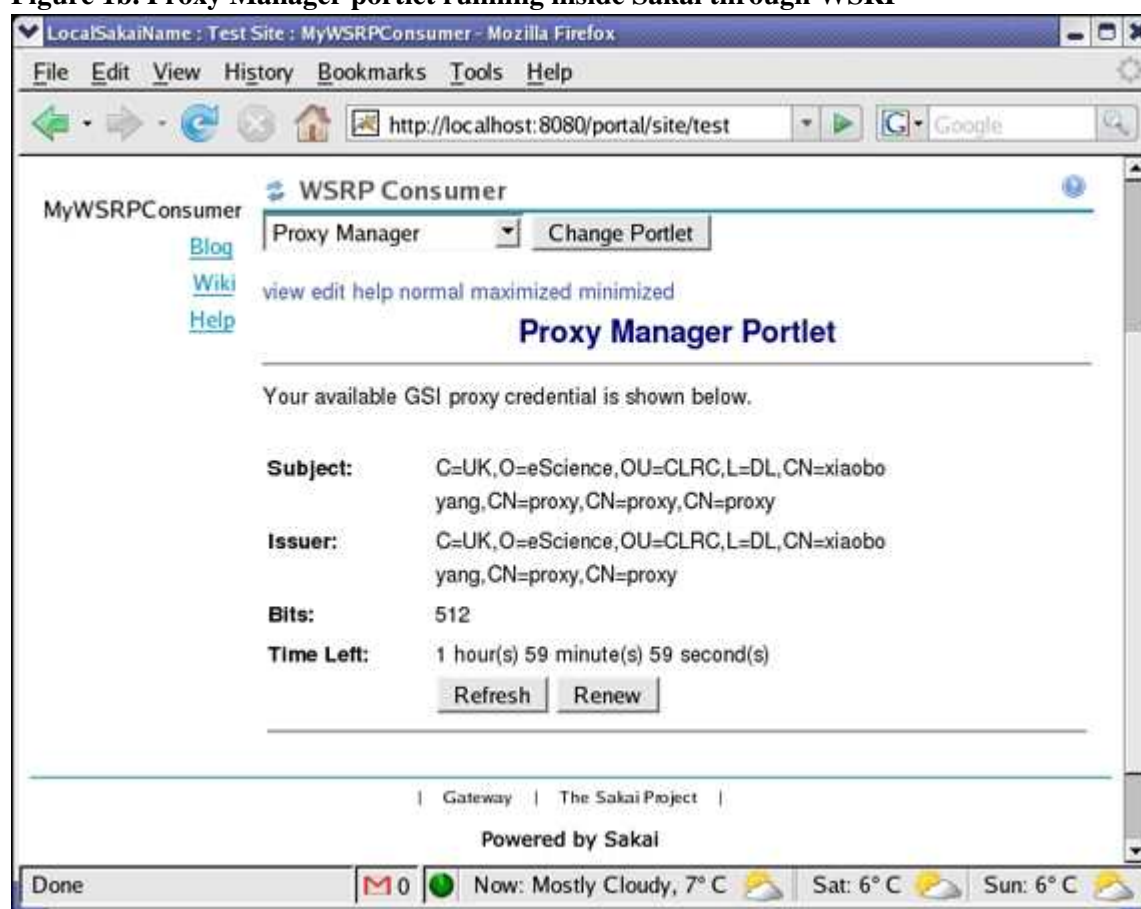


Figure 1b. Proxy Manager portlet running inside Sakai through WSRP



Whilst grid portlets can be deployed inside uPortal directly, we couldn't deploy them in Sakai because at the time this article was written, there was no JSR 168 support in Sakai. This clearly demonstrates the power of WSRP.

Without redeployment of those grid portlets, uPortal and, in particular, Sakai, are now able to provide grid tools maintained in a remote WSRP4J producer. The only requirement is for a generic WSRP consumer. This approach eliminates the need for local deployment of grid tools, which, in some cases, is not desirable.

The future of grid portals

Grid portals are bridging the Web and grid. Today, second-generation grid portals are based heavily on Java™ standards. JSR 168 helps portal developers write reusable Web components, while WSRP V1.0 defines a Web service approach to reuse them. Grid portals are becoming more complex in response to demands from end users. Currently, grid portals are focusing on the integration of services and resources, among which visualisation and workflow are two important aspects.

Visualisation and workflow

Visualisation is essential for experimental and numerical studies in scientific research. Complex visualisation systems typically require special hardware and software that are not easy to integrate within Web portals. Currently, visualisation is limited to the use of techniques like Java 2D/3D, Java applet, and Java Web Start, and it is relatively independent of grid portals. The grid also brings distributed data sources, which challenge visualisation. And sometimes, interactive visualisation of temporary results is required during numerical simulations so they can be steered, paused, and resumed with feedback from the visualisation.

Workflows are becoming increasingly important to grid portals, especially because grid middleware, such as GT4, is incorporating the Service-Oriented Architecture (SOA). With more services being developed, workflow systems are widely investigated to offer the ability of service integration. For example, a typical scientific research procedure in bioinformatics may involve a set of services executed in a particular sequence. This can be managed well by workflow systems like Taverna, a component of the United Kingdom's myGrid project.

Similar to visualisation, workflow systems normally have their own GUIs, which make it hard to integrate them in grid portals -- though approaches like the Java applet make integration possible. Efforts are ongoing to integrate workflow systems in grid portals. For instance, a JSR 168 portlet has been developed and tested in GridSphere to provide myGrid workflow via a portal interface.

Whilst integration of visualisation and workflow in grid portals is ongoing, based on our experience, grid portals are still not as attractive as desktop applications for these purposes. This explains the lack of portal users. Most of all, communications between portal developers and end users need to be enhanced so portals can meet requirements from end users. Next, we discuss how to attract end users from a technical point of view.

Web 2.0

In the Web design and development communities, Web 2.0 has become a precise buzzword. While it's not easy to define, Web 2.0 includes content-hosting technologies such as blogs, wikis, RSS feeds, etc. These technologies are all aiming to provide a more collaborative and interactive Web environment for sharing information. Users are no longer acting only as information receivers but they are becoming information providers.

In addition to the community-based techniques, Asynchronous JavaScript And XML (Ajax) is used to build rich, interactive Web applications by making use of idle time during interactions with a Web browser. Google is a good example of how end users experience the power of Ajax. Gmail, Google Suggest, and Google Maps all make use of Ajax to enhance interactivity with users. For example, Google Suggest pops up words to complete input when a user is typing a search term.

Google Maps allows users to drag the map to different views without waiting for a response from the server -- and

there is no need to refresh the whole Web page. Traditional online map providers, such as [Multimap.com](http://www.multimap.com), require users to wait for a response from the servers of each action to refresh the whole page.

Ajax can be applied to JSR 168 portlets because they are JavaScript-based. For example, portlets within the Google portal can be dragged and dropped so users can place portlets easily according to their preferences. Figure 2 provides a screenshot with a portlet called "Quote of the Day" being dragged to a new place. The dashed box in Figure 2 indicates the portlet's current position.

Figure 2. Google portal: A portlet is being dragged



Obviously, Ajax can be used by portal developers to design more interactive portlets, which in some senses is key to attracting end users. But based on our initial investigation, due to security concerns, Ajax does not work properly within remote portlets, though a workaround does exist.

There are also requirements for collaboration tools, such as instant messaging and online discussion forums, for exchanging ideas, organising project meetings during research activities, etc. A grid portal can act as a gateway for an e-learning or e-research system equipped with some collaboration tools. We believe these grid portals would attract more users.

Reusing existing Web applications

The lack of open source portlets that can be shared is a real issue now. Thus, it is important to convert existing applications to portlets. For example, Struts and JavaServer Framework (JSF) are two major Web development frameworks on the market. Struts is a widely adopted Web development framework in the Java world, while JSF, its successor, is now part of Java EE 5, acting as one of its key presentation technologies. JSF simplifies Web development by leveraging concepts like Inversion of Control (IoC) and UI components. These UI components have their own state -- event plus input conversion and validation. They can be written and distributed to other Web developers.

Techniques like Struts and JSF should be supported in portlet development because they are now in mainstream

Web development. There are many Struts-based Web applications, and the new JSF is gaining interest. Struts and JSF bridges from Apache Portals are attractive because they promise the deployment of Struts and JSF applications as portlet applications with little or no change. In addition to Struts and JSF, Apache Portals provide bridges for PHP, Perl, and Velocity.

Shibboleth

Acting as service clients, portals naturally communicate with local or remote services. Credential delegation becomes key in this scenario when building secure portals and services. Among techniques like XML Signature, XML Encryption, WS-Security, SAML, and XACML, work with Shibboleth is focusing on building federations within which Web resources can be shared. Shibboleth is a good example of single sign-on (SSO) in grid portals. In the United Kingdom, Shibboleth is being deployed in higher education as a single point of authentication for access to distributed digital resources. These institutions and universities have their own authentication systems for local users, which can be used in Shibboleth-enabled Web applications.




Portal federation

When we talk about SOA in grid portals, it's important to reuse not only services but also components like portlets. The latter is the key for building up future federated portals. Portals should be constructed using local and remote portlets, of which metadata are published on registry servers.

Summary

Today, grid portals play an important role as resource and application gateways in the grid community. Most of all, grid portals provide researchers with a familiar UI via Web browsers, which hide the complexity of computational and data grid systems. In this three-part series, we gave a general review of portals and discussed first- and second-generation grid portals. We built three grid portlets that demonstrate how a basic grid portal can be constructed using JSR 168-compliant portlets. We illustrated how these grid portlets are reused through WSRP and considered the future of grid portal development.

Share this...

-  [Digg this story](#)
-  [Post to del.icio.us](#)
-  [Slashdot it!](#)

JSR 168 and WSRP V1.0 are two specifications that aim to solve interoperability issues between portlets and portlet containers. In particular, today's grid portals are service-oriented. On one hand, portals are acting as service clients to consume traditional data-centric Web services. On the other hand, portals are providing presentation-centric services so federated portals can be easily built.

With basic grid related functions like proxy manager and job submission successfully implemented, advanced grid portals today are aimed at the integration of complex applications, including visualisation and workflow systems. Web 2.0 techniques were presented, and Ajax was recommended for portal development to make grid portals more interactive and attractive to users. In the future, grid portals should also aim to include existing Web applications and, as security techniques become more developed, credential delegation will play an important role in federation and sharing of grid services.

Resources

Learn

- [JSR 168](#) is a portlet specification developed by JCP to standardise communications between a portlet and a portlet container.

- WSRP V1.0 is a portlet specification developed by OASIS to standardise communications among portlet containers.
- The WSRP V1.0 Primer provides a tutorial-oriented explanation of the main concepts of the WSRP V1.0 specification.
- Be sure to read "Ajax: A New Approach to Web Applications," by Jesse James Garrett.
- Shibboleth is standards-based, open source middleware that provides Web single sign-on (SSO) across or within organisational boundaries.
- Browse all the grid computing content on developerWorks.
- To listen to interesting interviews and discussions for software developers, check out developerWorks podcasts.
- Stay current with developerWorks' Technical events and webcasts.
- Check out upcoming conferences, trade shows, webcasts, and other Events around the world that are of interest to IBM open source developers.
- Visit the developerWorks Open source zone for extensive how-to information, tools, and project updates to help you develop with open source technologies and use them with IBM's products.

Get products and technologies

- The Globus Toolkit is the de-facto standard implementation for enabling the grid.
- The myGrid Project aims to exploit the growing interest in grid technology, with an emphasis on the information grid.
- Pluto is the Reference Implementation of JSR 168 by Apache.
- The Sakai Project aims at providing a collaboration and learning environment for education.
- WSRP4J provides an implementation of the WSRP V1.0 specification by Apache.
- Download IBM product evaluation versions, and get your hands on application development tools and middleware products from DB2®, Lotus®, Rational®, Tivoli®, and WebSphere®.
- Innovate your next open source development project with IBM trial software, available for download or on DVD.

Discuss

- Check out the Grid computing forum on developerWorks.
- Participate in developerWorks blogs and get involved in the developerWorks community.

About the authors



Xiaobo Yang is a software developer working in the Grid Technology Group, CCLRC e-Science Centre, in the United Kingdom. He is interested in grid, collaborative virtual research environments,



and various Web technologies, including grid portals, and Service-Oriented Architecture (SOA).



Xiao Dong Wang is a senior software developer in the Grid Technology Group, CCLRC e-Science Centre, in the United Kingdom. His interests include grid middleware design and application, portal user interfaces and portlet development, JSP, and JSF. He has more than six years' experience in Java programming and Web and grid service development.



Robert Allan leads the Grid Technology Group, CCLRC e-Science Centre, in the United Kingdom. His background is as a physicist and -- since the mid-1980s -- developer of high-performance computing applications using the latest technologies. He has managed several large HPC and e-science projects in the United Kingdom. He is particularly interested in making the grid widely usable.