Red Hat Enterprise Portal Server: Architecture and Features

By: Richard Li and Jim Parsons

March 2003

Abstract

This whitepaper provides an architectural overview of the open source Red Hat Enterprise Portal Server and describes key distinguishing features of the product. The paper outlines the technologies used in Portal Server and lists the requirements that Portal Server was designed to address.

Copyright ©2003 Red Hat, Inc.

Table of Contents

Introduction

Red Hat Enterprise Portal Server brings a mature, open source solution architected on the J2EE standard to the enterprise portal space. Originally designed and built to provide a composite presentation for the Red Hat Web Application Framework, Portal Server has undergone significant design and development enhancements in recent months that broaden the types of content that may be displayed on a portal.

Portal Server aggregates both local and remote content within a configurable and personalizable framework that supports multiple languages in its user interface. In addition, its rendering pipeline supports multiple modes of output such as WAP and XHTML. The configuration options available for portal administration were designed so that portals could be built and targeted for the individual, work groups or teams; people with a common set of interests; and large corporations and organizations.

Portal Server is inherently XML in nature, so integration with most web services and WSRP sites is a simple matter. Portal Server also integrates with the entire suite of applications built with the Red Hat Web Application Framework, including the Red Hat Enterprise Content Management System, document manager, forums, and bookmarks applications. The Portal Server API is fully documented for those users interested in building portlets to provide views into their own applications.

Features

The Red Hat Enterprise Portal Server includes a complete set of out of the box functionality, including:

- Creating and naming portals
- Administering people associated with a portal, including setting read, write, modify, delete, and administrator privileges.
- Providing a scaleable, highly configurable permissions hierarchy for managing portal access
- Adding, deleting, and reordering tabs in a portal to help organize content and available applications into separate portal views.
- Adding and removing portlets within a portal tab.
- Customizing a portal by choosing a layout and arranging the portlets where desired.
- Personalizing a portal by choosing look and feel elements such as color scheme, background images, and other rendering attributes.
- Globalizing all admin user interface labels for multiple language support.
- Providing a high-performance, scaleable caching system for portlets

In addition to the features listed above, the Portal Server integrates directly with the following applications built using the Red Hat Web Application Framework:

- Discussion Forum
- Document Manager
- Task manager
- Shared bookmarks
- Chat
- RSS display
- Content Management System

Architecture

Portal Server is written in Java and built on top of the Red Hat Web Application Framework. The Web Application Framework and Portal Server run in any standards-compliant servlet container that supports the Servlet 2.2 and JSP 1.1 specifications. Data persistence is achieved via an object-relational persistence layer; out-of-the-box, Portal Server supports Oracle 8i, Oracle9i, and PostgreSQL 7.2.x.

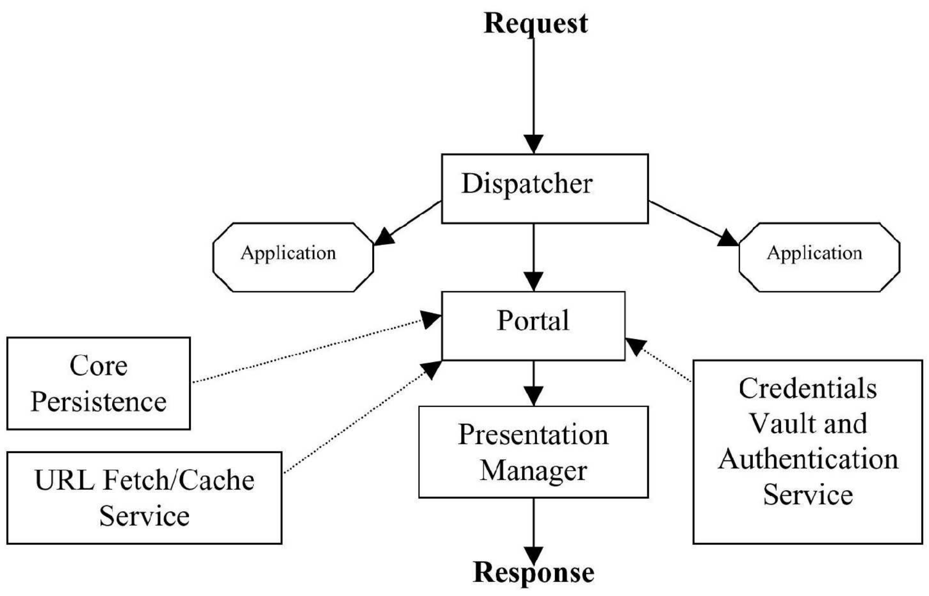

Figure 1: Principle components of Red Hat Enterprise Portal Server

Portal Server deployments usually run parallel with applications from the Red Hat Enterprise Applications family. In fact, application instances on the server may be instantiated and mounted from within the portal administration area. It is the dispatcher's responsibility to direct requests to the proper servlet handler. After the dispatcher determines that a request is for a portal instance, the request is handed off to the portal's dispatcher, which checks the user against the database and, pending authorization, assembles the portal requested by the user. The assembled portal is then passed to the Presentation Manager for final styling and transformation before being delivered to the requesting device.

Presentation Rendering Pipeline

One of the unique architectural features of Portal Server is its central reliance on XML and XSLT for rendering views of a portal. The portal framework aggregates content to be viewed in the portal as a number of *portlets* arranged in a tree data structure, each of which contains a unique *portlet renderer* that produces the portlets' output in the form of an XML element. At render time, the portal creates an empty Document Object Model (DOM) instance, adds a root element, and passes the DOM to each portlet in the render tree. At each stop on the tree, the portlet's element is added to the DOM instance. When this aggregation stage is finished, the DOM is passed to the presentation manager, where it is styled and transformed by either the Xalan or Saxon XSLT processor.

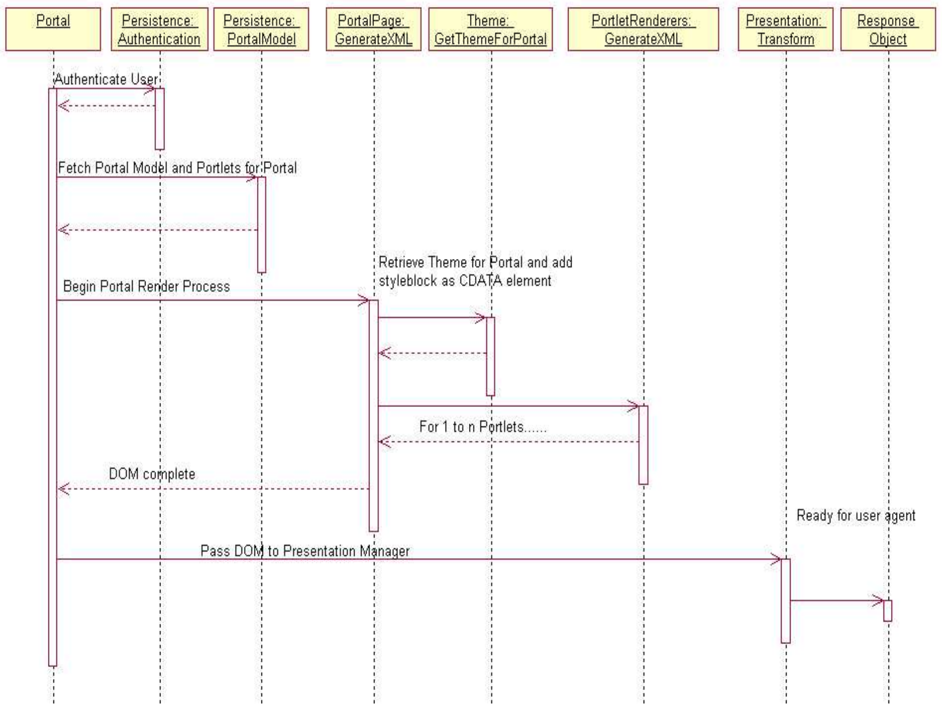
Portal
Persistence: Authentication
Persistence: PortalModel
PortalPage: GenerateXML
Theme: GetThemeForPortal
PortletRenderers: GenerateXML
Presentation: Transform
Response Object
Authenticate User
Fetch Portal Model and Portlets for Portal
Retrieve Theme for Portal and add styleblock as CDATA element
Begin Portal Render Process
For 1 to n Portlets......
DOM complete
Ready for user agent
Pass DOM to Presentation Manager

Figure 2 – Rendering process for a Red Hat Enterprise Portal Server deployment

The sequence diagram above shows the rendering process for Portal Server. Some detail is left out for the sake of clarity, however. For example, if a portlet within the portal is tasked with displaying external content from another server or a distributed database, then the portlet's renderer would register its external URL with the URL Fetcher service, which runs as a background process and maintains a cached copy of the external content ready for rendering within the portlet.

Portlet API

Portal Server is open source, and how-to documentation, as well as Javadoc, is available to assist users who wish to develop custom portlets. The process for developing a new portlet is designed to be quick and simple:

1. Build a small Persistence Definition Language (PDL) file that describes the data model for the portlet. Help with PDL is also available to developers.
2. Write a Java class for the portlet that extends the base portlet class, and includes accessors for any columns specified in the PDL file.
3. Write a Java class to render the new portlet that extends Portlet Renderer.
4. Write an Initializer class that extends the Red Hat Web Application Framework's Initializer class, and include an init entry in the server's initialization script.

Future Enhancements

Some Portal Server features targeted for future implementation include:

- **Standards support:** The JSR 168 and Web Services for Remote Portals (WSRP) standards are rapidly gaining support and maturity. Red Hat is committed to supporting open standards and expects to adopt these standards in future releases.
- **Jabber Integration:** The chat application is built around an in-house chat server project. Replacing the current chat server with jabber would enhance the Chat tool and provide a central messaging mechanism within Portal Server that would allow for message passing between applications and users (urgent requests, pending workflow tasks, etc.), as well as more sophisticated chat room portlets.
- **Credential Vault enhancements:** This allows for single sign on access to WSRP sites and distributed database sources.
- **URL Cache Service scalability:** This offers a more sophisticated UI for tuning the frequency of cache updates.
- **RSS Feed Portlet Enhancement:** This feature allows the user to open RSS links within the RSS portlet or in a new browser window.

Summary

Red Hat Enterprise Portal Server is an open-source, Java-based portal framework that is rapidly configurable to any conceived enterprise purpose. Portal pages can be built from ready-made templates. If the templates provided do not meet the needs of a deployment administrator, a custom template can be built and saved or portal pages can be built from the ground up. This allows a portal to be customized to the mission it serves - whether for a user's personal portal, a department within an organization, or for an enterprise-wide purpose.

Not only are Portals customizable, they are also *personalizable*. The look and feel of a portal can be adjusted to suit the taste of an individual user. For example, a portal built for an organizations' inside sales staff should offer the same access opportunities for all sales people. Each sales person has the option of making their view of the portal their own by choosing their own organizational scheme, colors, backgrounds, and even adding custom portlets that they have created. Personalization options, like most elements of Red Hat Enterprise Portal Server, are offered with fine-grained permissioning; server administers can choose who is allowed access to a portal and which aspects and features a portal member is allowed to alter.

For more information about Red Hat Enterprise Portal Server or other open source applications available from Red Hat please see our website at http://www.redhat.com/software/rhea or contact us at +1 866-273-3428 x41005.