IBM WebSphere Portal Server

Product Architecture

V1.2

May 7, 2001

Abstract
Portals provide a secure, single point of access to diverse information and applications, personalized to the needs of their users. WebSphere Portal Server provides an open and extensible framework plus the flexible and scalable infrastructure needed for many types of portals supporting B2C, B2B, and B2E usage models. WebSphere Portal Server provides the base on which to build enterprise, marketplace, consumer, and workspace portals accessible from a wide variety of desktop and mobile devices.

This paper gives an in-depth view of the architecture and design of WebSphere Portal Server, including its presentation and portlet framework, security, user management, personalization, content management, performance and scalability, search, and enterprise information connectivity features.

Table of Contents

Introduction

Portals provide a secure single point of access to diverse information and applications, personalized to the needs of their users. WebSphere Portal Server provides an open, flexible, and scalable infrastructure for creating and deploying many categories of portals that are accessible from a wide variety of desktop and mobile devices.

In some respects, enterprise information portals, business-to-business marketplaces, employee workspaces, and public web portals have common requirements. All require scalable infrastructure, a flexible and powerful presentation framework, and a framework for building portal components easily. All require a high degree of personalization so that the most relevant information is delivered to the user, enabling a more productive interactive experience and encouraging user loyalty to the portal.

Of course the various types of portals do have some unique requirements. Depending on the nature and sensitivity of the information, some portals may require a greater degree of security, including specialized forms of authentication and access control. Depending on the size of the user base, some portals may require very high availability and scalability. Consumer portals generally allow users to enroll themselves and to manage their own accounts. Conversely, enterprise portals often require integration with existing user databases or enrollment systems. WebSphere Portal Server offers a framework with the depth and flexibility to meet all these needs.

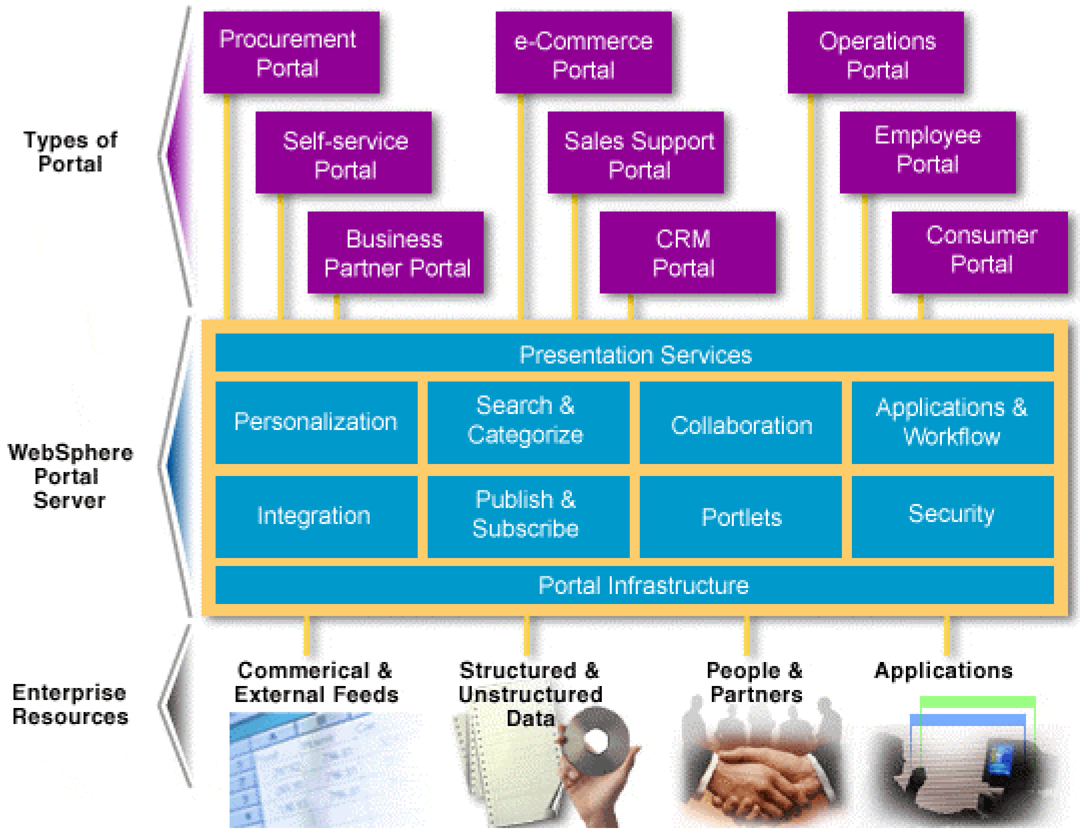


Figure 1. Portal Solution Architecture

The portal framework simplifies the development and maintenance of portal sites.

- The page structure is defined only once
- Portlets are defined independently
- Portlets can be changed without impacting the overall page design
- Targeting multiple browsers and mobile devices is made easier.

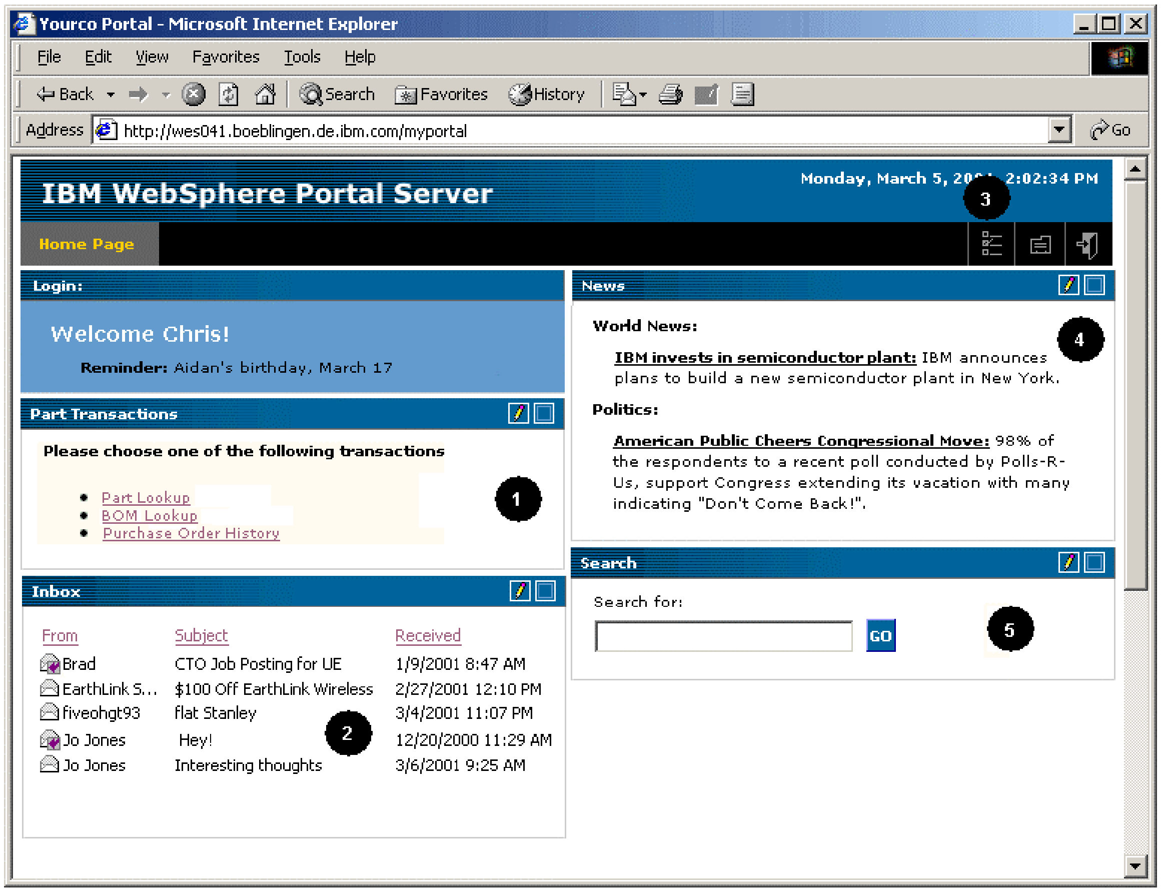
Figure 1 shows the overall architecture of the portal solution. The remainder of this paper will explore technical details about each section of this architecture diagram, including WebSphere Portal Server's Presentation services, its Connectivity Services, and its core capabilities for personalization, search, collaboration, workflow and application integration, content publishing and subscriptions, portlets, security, and more.

Presentation Services

The purpose of the portal framework is to produce a customized, personalized home page for its users, where the home page contents are assembled (or "aggregated") from a variety of content and application data sources. Although desktop browsers are the typical portal access point, other devices might also be used.

While every portal is unique, some popular elements include those shown in Figure 2.

1. Custom applications for business intelligence and enterprise resource planning deliver timely and relevant information.
2. Collaboration portlets keep users in touch with colleagues or customers in real time and centralize access to e-mail, calendar, address books, and to-do list.
3. The Customize link lets users adjust their profile and modify their home page contents.
4. The News portlet shows the latest news headlines from a syndicated content provider such as Reuters, Dow Jones, Newswire, or Business Wire.
5. The Search portlet gives quick access to both Internet content and local documents for searching.



http://wes041.boeblingen.de.ibm.com/myportal

IBM WebSphere Portal Server

Monday, March 5, 2001 2:02:34 PM

Login:

Welcome Chris!

Reminder: Aidan's birthday, March 17

Part Transactions

Please choose one of the following transactions

- Part Lookup
- BOM Lookup
- Purchase Order History

Inbox

From	Subject	Received
Brad	CTO Job Posting for UE	1/9/2001 8:47 AM
EarthLink S...	$100 Off EarthLink Wireless	2/27/2001 12:10 PM
fiveohgt93	flat Stanley	3/4/2001 11:07 PM
Jo Jones	Hey!	12/20/2000 11:29 AM
Jo Jones	Interesting thoughts	3/6/2001 9:25 AM

News

World News:

IBM invests in semiconductor plant: IBM announces plans to build a new semiconductor plant in New York.

Politics:

American Public Cheers Congressional Move: 98% of the respondents to a recent poll conducted by Polls-R-Us, support Congress extending its vacation with many indicating "Don't Come Back!".

Search

Search for:

Figure 2. Typical Home Page

Portal Engine

WebSphere Portal Server provides a pure Java portal engine, which runs on many hardware platforms and operating systems. The portal engine’s main responsibility is to aggregate content from different sources, and to serve the assembled content to multiple devices. Because each content area (or “portlet”) is developed and maintained as a discrete component, it is faster and easier to develop the overall site. In effect, the presentation details of the portlets are decoupled from those of the overall page.

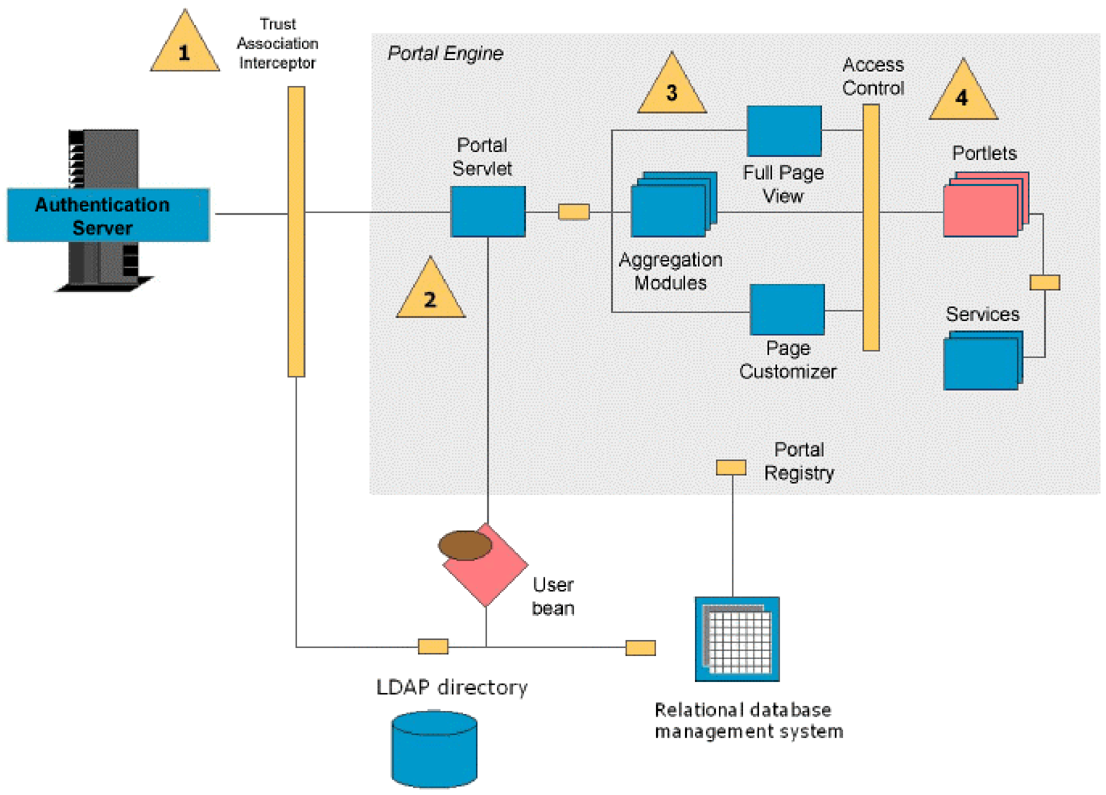


Figure 3. Portal Engine Components

1. In front of the portal engine is an authentication component such as standard WebSphere security or a third-party authentication proxy server.

2. The central component in the portal engine is the *portal servlet*. It examines the URL and header fields of each request and invokes the appropriate handler. The request is handled in two phases. In the first phase, portlets have an opportunity to send event messages to other portlets. (For example, portlets might send events in order to update data that will be rendered in the next phase).

3. In the second phase, the appropriate aggregation module for the user's device renders multiple portlets in a single page. The aggregation modules accumulate information from each portlet, put standard decorations (e.g. a title bar, edit button, and enlarge button), around the portlet, position it on the page, and generate the overall page markup.

4. Access to portlets is controlled by checking access rights during page aggregation, page customization, and other access points such as viewing the portlet in its maximized state.

Currently, WebSphere Portal Server has three aggregation modules. The HTML Aggregation component produces pages for desktop computers and other devices with HTML browsers. The WML aggregation component produces

WML content for WAP devices, which are typically mobile phones. The iMode aggregation component produces cHTML markup (e.g. for mobile devices in the NTT DoCoMo network).

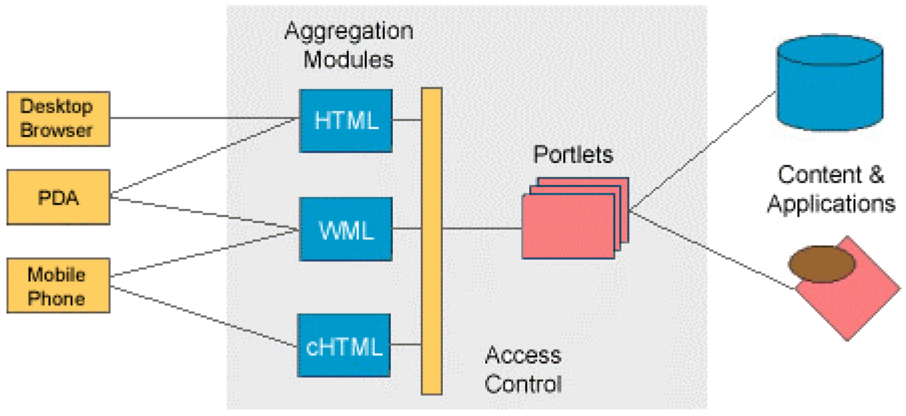


Figure 4. Page Content Aggregation

In future releases, IBM will provide additional aggregation components, including a voice aggregation module for devices with Voice XML browsers and a PDA aggregation module for Personal Digital Assistant devices.

Each user can customize a unique home page for each device, selecting the content and applications that are most useful on the device. When the home page is requested, page aggregation works by first detecting the type of device that is making the request, and then assembling the portlets which each render their contents in the appropriate markup language.

Multi-device Portlets

When a user customizes the home page for a particular device, the portlet selection list only shows portlets that can actually produce markup appropriate for that device. Thus, the list of available portlets for each device depends on what the portlets can actually do.

Some portlets may be available for all the supported devices, while others may be available only on a single device. The user interface design of each portlet also varies from device to device. Thus, the user's home page and each of the portlets might be very different on a mobile phone, as shown in Figure 5.

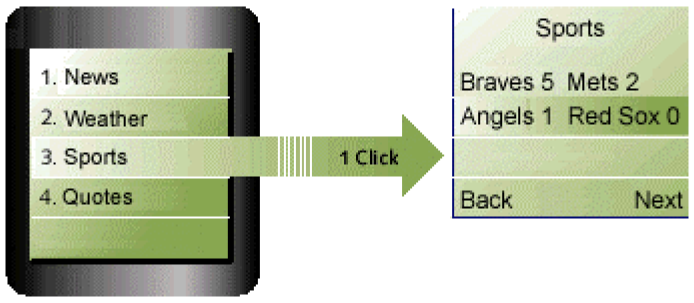


Figure 5. Mobile Access

In general, WebSphere Portal Server produces platform and browser-neutral markup, which will work most popular browsers, including Microsoft's Internet Explorer (4.0 and higher) as well as Netscape Navigator (4.5 and higher), running on Windows, Macintosh, and other operating systems. The new Opera browser from Opera Software is also supported, along with several mobile phone browsers. PDA devices running HTML or WML browsers will also work. Most browsers that support HTML 3.2, WML 1.1 or 1.2, and iMode 1.0 will work, though specific devices should always be verified.

However, depending on the needs of a particular portal or set of portlets, it is equally possible and valid for portlets to generate markup that is specifically targeted to newer browsers, such as Internet Explorer 5 or Navigator 6.

Constructing the Home Page

The structure of the page and its navigation areas is defined in page template files, using Java Server Pages markup. The arrangement and the look of these areas can be changed easily, so that it is possible to make the portal pages reflect your unique style or brand image.

Navigation areas can be placed anywhere, such as a masthead area at the top of the page or a navigation area on the left. Cascading style sheets, images, and other visual elements are used to further define the look of the page. Using these techniques, the contents of the navigation areas are determined by the portal site developer. The precise details differ for each mark-up. For example, WML decks are also structured using JSP templates, but of course they do not use all the same graphic elements that HTML pages have.

When building the body of the page where the portlets are displayed, WebSphere Portal Server uses page layout templates, which are also based on Java Server Pages.

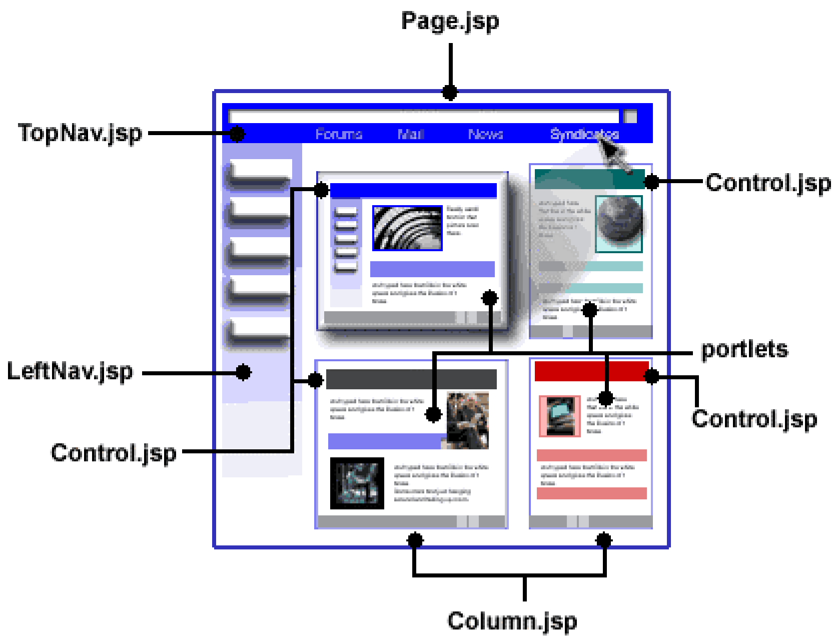


Figure 6. Page Layout Templates

The process to aggregate a page works like this:

- When the aggregation module is invoked, it obtains the page layout information for the current page of the current user.
- The page layout information is parsed and generates a tree with nodes referencing rows, columns, portlet decoration elements, and finally the actual portlets.
- As the tree is traversed, JSP templates corresponding to each node are included in the output stream. Each node representing a row or column includes the appropriate row or column template. Each node that represents a portlet decoration includes another JSP template. For the portlet nodes, the actual markup for that portlet is included.

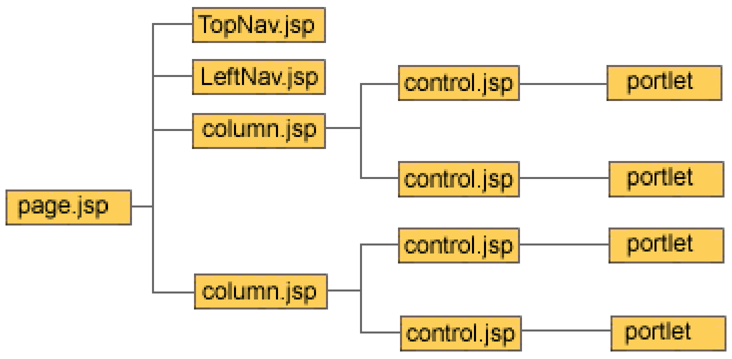


Figure 7. JSP-based Aggregation

Just as the page layout and navigation areas can be changed easily, so can the JSP row, column, and decoration templates. This makes it easy to change the look and feel of all aspects of the portal.

End-user Page Customization

The portal engine's customizer component allows users to modify the content and layout of their portal pages. End users can define one or more home page tabs, and then decide how the portlets are arranged inside the portlet display area of each tab. There are several predefined column styles and portlets can be placed or rearranged in each column.

Globalization and Accessibility

WebSphere Portal Server loads page templates and portlets based on the browser settings or based on the preferred language specified in the user's profile.

The page mark-up produced by WebSphere Portal Server meets U.S. federal guidelines and Web Accessibility Initiative guidelines for ensuring accessibility to people with disabilities.

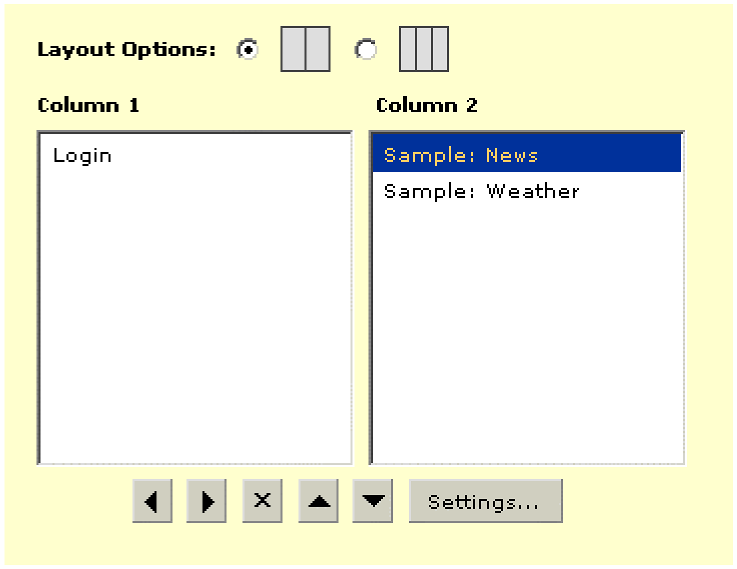


Figure 8. Page Customization

Portlets

Portlets are the visible components that users see on their portal pages. Portlets can be as simple as an e-mail inbox or as versatile as a sales forecast from an ERP application. From a technical point of view portlets are very similar to Java servlets, except that they only return a subset of the output page.

Portlets are the way that software vendor products or in-house custom applications can snap into the portal framework. They can be written in a variety of ways. The simplest portlets might use static HTML or WML mark-up, or perhaps Java Server Pages syntax. Intermediate portlets could use Java beans or servlets, or perhaps XML/XSL transformations. More complex portlets involve writing custom Java code.

Portlets will be provided by IBM as well as by third-party software vendors and business partners. Standard portlets provided by IBM include portlets for personal productivity applications, such as Lotus Notes e-mail and calendar, plus many application enabling and utility portlets.

Standard Portlets
WebSphere Portal Server provides a rich set of standard portlets. Specifically, it includes the following:

- RSS portlet: formats Rich Site Summary data, commonly used for news feeds
- XSL portlet: transforms any XML using XSLT
- JSP portlet: renders any Java Server Pages file or servlet
- HTML and WML portlet: renders any URL
- Notes portlets: includes portlets that access Lotus Notes e-mail, calendar, address book and to-do list.
- Exchange portlets: includes portlets that access Microsoft Exchange e-mail, calendar, address book and to-do list.
- Sametime portlet: gives access to the SameTime instant messaging server
- QuickPlace portlet: gives access to a QuickPlace team room
- Syndicated Content portlets: provide news and other information from ScreamingMedia, iSyndicate, Factiva and Hoovers.

Portlets can be structured to inherit configuration settings from other portlets. In this way, many new portlets can be created without writing any code. For example, a general news portlet might be defined to work with any Rich Site Summary data feed. Two specific instances of the portlet might point to CNN and BBC news.

Portlet API
For cases where custom coding is required, WebSphere Portal Server includes an open standard Java API, called the *portlet invocation API*. The API provides a stable, high performance, scalable interface for portlet writers. This API is independent of the portal engine to allow interoperability of portlets among future portal engines. The portlet invocation API is supplemented by plug-in services, giving vendors the ability to provide value-add functions without requiring API changes from release to release of the portal server.

The portlet invocation API is very similar to the Java servlet API. The *Portlet* class corresponds to the *Servlet* class, with method signatures matching the *init*, *service*, and other key methods. Similarly, the *PortletConfig* class corresponds to *ServletConfig, PortletContext* corresponds to *ServletContext*, *PortletRequest* corresponds to *ServletRequest*, and so on.

Programmers who already know the servlet API will be very comfortable with writing portlets. Best practices for writing portlets are similar to those for writing servlets, since they have similar operational characteristics. Portlets, like servlets, are singletons; meaning that there is only one instance of each portlet class shared by all requesters. This means that instance variables and class variables in portlets must not be used; instead, you should store information in the user's session. It is important to make the portlet code as fast as possible to minimize its impact on the overall page performance.

Services, Events, and Access Control

The portlet API allows for the definition and registration of services implementing particular interfaces. This enables a stable API core that can be extended by services as required. Standard services provided by WebSphere Portal Server include the *UserInfoService* (for getting user data like name, address, etc.) and the *PersistenceService* (for storing per-user and per-portlet settings). These services in turn, call the User bean interfaces that will be described later in this paper.

To enable portlet-to-portlet communication, the PortletContext object has a send method. This method allows a portlet to send a message to another portlet through the framework. The target portlet will then receive a message event and can then retrieve the message. In addition to sending messages, portlets also can share data through dynamic attributes attached to their context object.

The home page customizer, aggregation modules and views invoke portlets via the *portlet invoker*. The portlet invoker uses the *access control interface* to determine whether the current user is permitted to access the portlet.

Portlet Development

WebSphere Portal Server includes several example portlets with source code. You can use these examples to learn about portlet programming techniques or as a starting point for further portlet development. Many of the portlets can be used as-is, or can be specialized by changing a few parameters rather than by writing code.

Portlets include both visual elements and processing logic. A typical portal could include class files, web pages, images, and other related assets. All of these assets are packaged together into a jar file format, called a portlet archive file (PAR).

When writing custom portlets, a model-view-controller design is recommended, as shown in Figure 9.

1. The *controller* is the class responsible for rendering the portlet by calling upon the appropriate view.
2. *Views* are usually implemented as Java Server Pages. Portlets may have several different views, including their *standard view* (which renders the portlet on the home page), a *maximized view* (which renders the portlet in its maximized state), and an *edit view* (which displays a page for changing the portlet settings).
3. *Models* are data bean classes, which hold the internal settings for the portlet. This data bean also supplies data to the edit view.

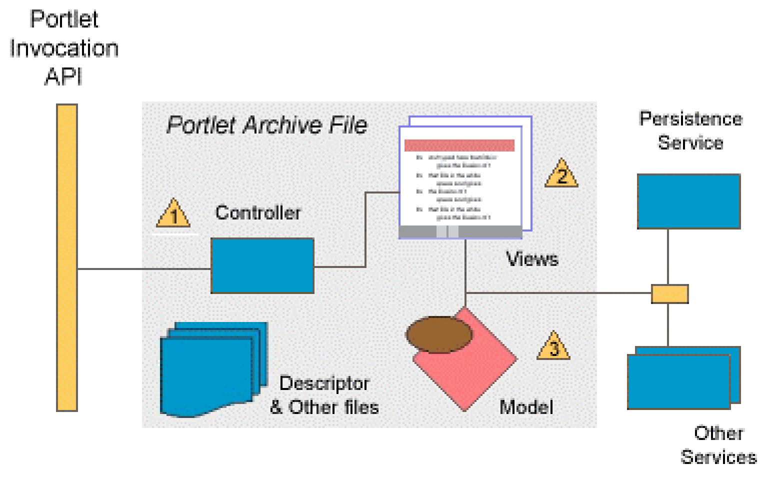


Figure 9. Portlet Structure

To make the job easier, WebSphere Portal Server provides a generic *MVCPortlet* class that sets up the structure for you. It accepts parameters for the JSP views and the data beans. This enables simple portlets to be implemented just by creating the JSP files and beans.

Tools
Both IBM and third party web application development tools can be used in portlet development. IBM offers VisualAge for Java, which is useful as a programming and test environment for portlets. IBM's WebSphere Studio is useful for creating and publishing Java Server Pages, images, and other portal assets. The WebSphere Everyplace Suite SDK includes additional tools for developing and testing mobile content and applications.

There are several steps involved in creating and deploying a portlet:

1. Implement the portlet classes using VisualAge for Java
2. Implement the Java Server Pages using WebSphere Studio
3. Develop other portlet resources, such as images, data connections, multi-media content, etc.
4. Publish the classes, content, and JSP files in the portal directory structure, using WebSphere Studio
5. Configure the portlet in the portal registry
6. Test the portlet by adding it to a user's home page.

Steps 4 and 5 are simplified by using the Portlet Archive (PAR) file. This is very similar to a web application archive file. It contains an XML descriptor for configuring the portlet, plus all the related file assets such as images and JSP files. When a PAR file is deployed, it copies all file assets into the appropriate folder and adds the portlet configuration information to the portal registry.

Portal Services

Personalization

WebSphere Portal Server allows users to customize the appearance of their portal pages according to personal preferences. The customization is accomplished partly through administrative setup, which defines the default settings and access rights to portlets. Further customization is accomplished through explicit user actions to change the contents and layout of the portal home page.

Attract and Retain Visitors

Personalization is an important functionality for all kinds of portals. It is a mechanism for attracting and retaining users.

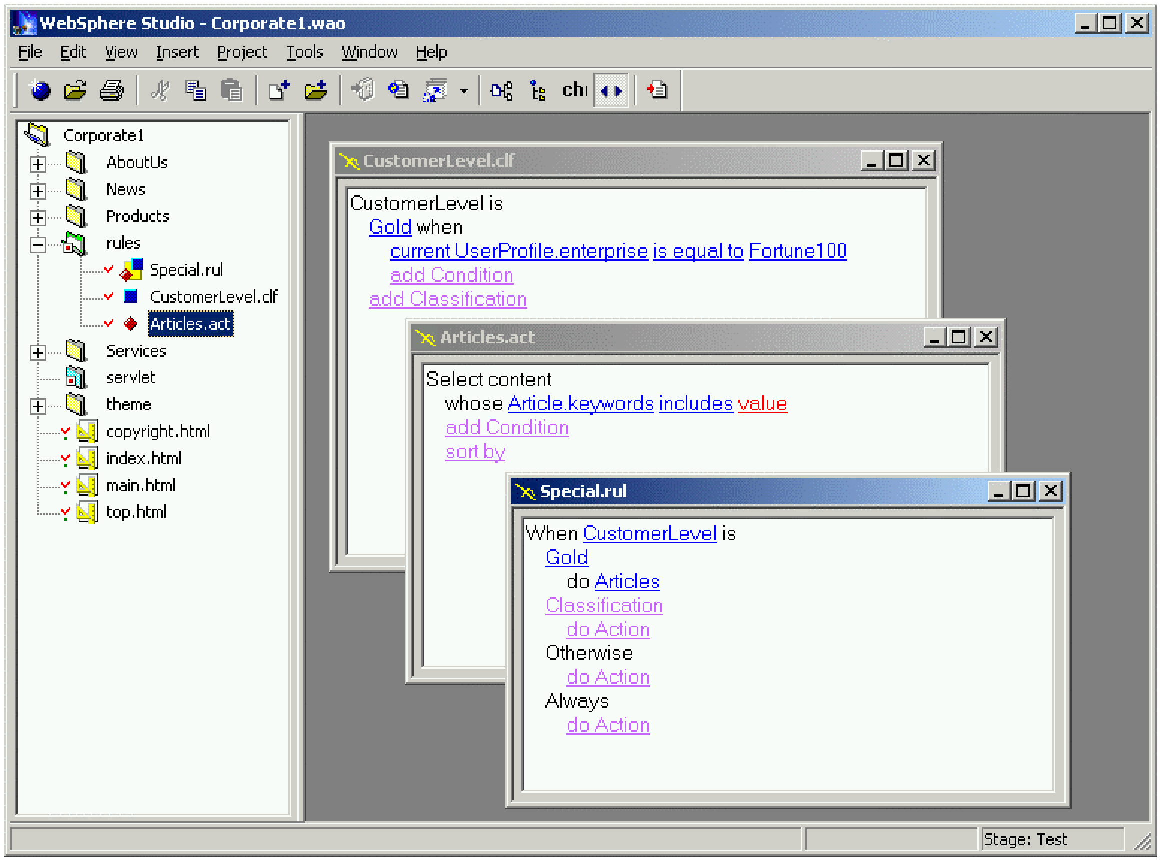
Personalized content makes users more productive while visiting the portal by making the most relevant information readily available. Personalized interaction distinguishes a portal from an ordinary web site.

IBM's WebSphere Personalization product is integrated into and included with WebSphere Portal Server, so that advanced levels of personalization can be achieved. For example, personalization can be based on business rules and user profile information, in addition to explicit user preferences. The WebSphere Personalization server goes beyond simple home page customization, and supports targeting information to specific users. It offers two advanced kinds of personalization techniques:

- The rules engine uses business logic to select content for the user. For example, a rule might display special discounts to gold customers, but only during the summer months.
- The recommendation engine uses collaborative filtering technology to select content based on common interests or behaviors. It observes click streams that can subsequently be examined for trends. This technique is often used in commerce portals for cross-selling products.

The portal server, the rules engine and the recommendation engine share user profile and content repositories. In other words, the User bean class of the portal server is already enabled for use in WebSphere Personalization Rules. Additionally, content can be stored in any data repository and is accessed through classes implementing the WebSphere Personalization Resource interface methods.

A portlet's JSP views can use WebSphere Personalization rules and recommendations in the same way that any JSP page does. This allows the content within the portlet to be personalized, based on the rules and recommendations. Rule and recommendations can also be used in the layout JSP templates or in the page customizer JSP to provide more advanced personalization of the portal.



CustomerLevel is
Gold when
current UserProfile.enterprise is equal to Fortune100
add Condition
add Classification

Select content
whose Article.keywords includes value
add Condition
sort by

When CustomerLevel is
Gold
do Articles
Classification
do Action
Otherwise
do Action
Always
do Action

Figure 10. Rule Editor for WebSphere Personalization

Customers who have already used WebSphere Personalization to create personalized web page can take advantage of the JSP portlet to reuse their JSP pages. The content model and the user profile are also reusable. If there is not already an existing user profile resource class, the one supplied with the portal server is already enabled for use with WebSphere Personalization. If there is an existing user profile resource class, then using the same LDAP repository for both the personalization and portal servers ensures that the user identity information is consistent. Alternatively, the user profile resource class can be replaced by what the portal server provides.

Content Management

WebSphere Portal Server works with content management tools such as Interwoven TeamXpress, Vignette Content Management Server, FatWire UpdateEngine 5, or Documentum 4i WCM, and IBM Content Manager.

For example, the Interwoven products work with WebSphere Portal Server to help customers develop presentation templates and portlets and also display and deploy content from the content management repository to the portal.

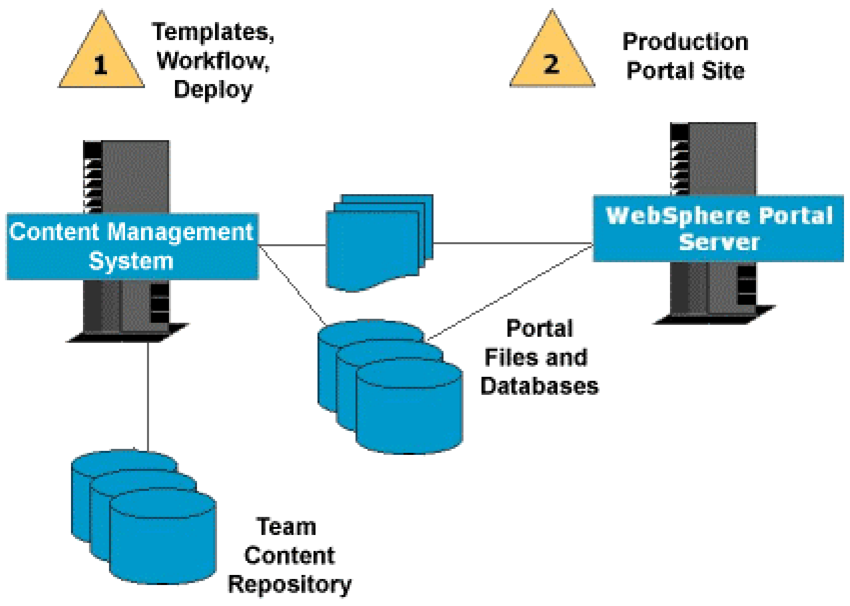

1
Templates, Workflow, Deploy
2
Production Portal Site
Content Management System
WebSphere Portal Server
Portal Files and Databases
Team Content Repository

Figure 11. Content Management

Content Publication and Subscriptions

WebSphere Portal Server provides the infrastructure to facilitate connections to virtually any content or application source. The portal server provides built-in support for many common content standards, including Rich Site Summary (RSS), Open Content Syndication (OCS), News Markup Language (NewsML), News Industry Text Format (NITF), and virtually any XML or browser markup.

Content Suppliers

Content suppliers offer a large variety of content for portals. Some providers offer live data through a URL (e.g. Moreover and XMLTree), and others require client software that stores content locally, either as files or in a database (e.g. ScreamingMedia and iSyndicate). Pricing, subscription models, and content formats vary widely.

Content Sources for Portlets

News and financial content can easily be displayed through portlets that have been developed by IBM's content partners. These companies offer trial content feeds and various content offerings that you can purchase for use in your portal.

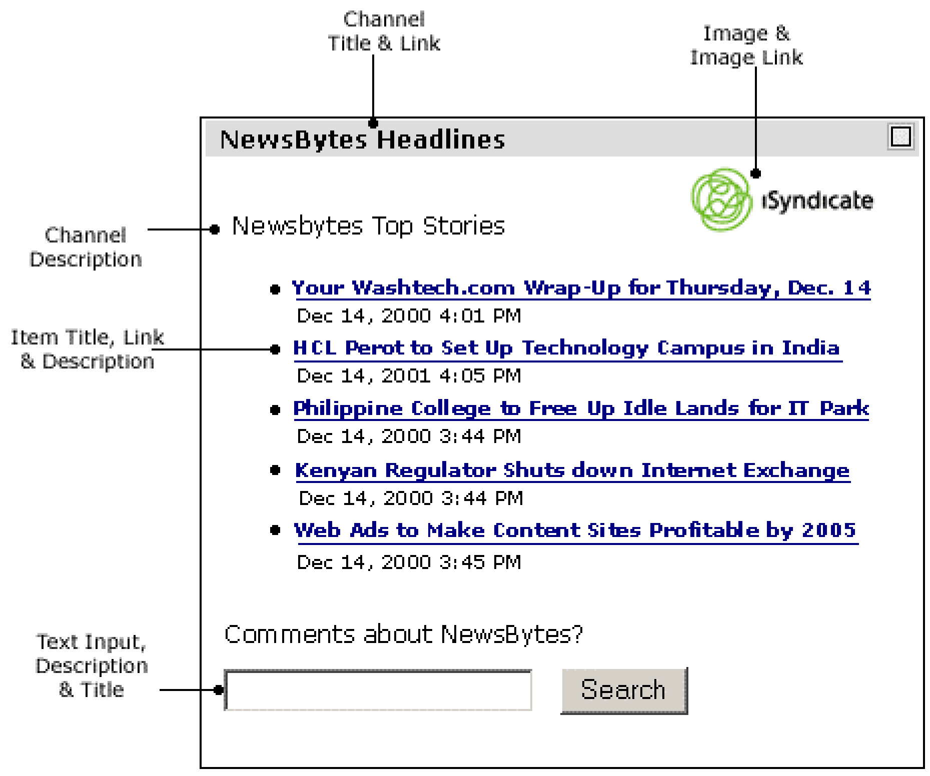
- Factiva, a Dow Jones & Reuters company, provides world-class global news and business information, offering access to up to 7000 highly respected global, multi-language sources such as The Wall Street Journal, The New York Times, Le Monde, The Times of London, and the Dow Jones, Reuters, and AP newswires. For more information, see http://www.factiva.com.
- Hoover's, Inc, aggregates business information on 64,000 companies, including public, private, and non-US businesses. Hoover's capsules and profiles consist of company overviews, products, operations, officers, competitors, financials, and more. For more information, see http://www.hoovers.com.
- iSyndicate, Inc. is a leading provider of syndication services for enterprises and other Internet-connected business seeking to leverage the Web for information collection and delivery. iSyndicate offers one of the largest selections of syndicated content available, including wireless content, from over 1200 leading brands and niche providers around the globe. For more information, see http://www.isyndicate.com.

- ScreamingMedia, Inc. is a leading global provider of content solutions: content infrastructure, syndication, and services. ScreamingMedia aggregates licensed content, such as news, features, photos, video, stock quotes, audio, and weather reports, and then filters, delivers, and precisely integrates it into its customers' Web sites instantaneously. For more information, see http://www.screamingmedia.com.

Content Syndication Standards

Publication and subscription of content is supported in WebSphere Portal Server through Open Content Syndication (OCS). The Open Content Directory Format provides a concise, machine-readable listing of syndicated channels. The directory format is capable of supporting multiple sites, each with multiple channels. Each channel can have multiple formats such as RSS (Rich Site Summary) versions 0.90 or 0.91, plain text, WML or Scripting News format as well as separate publishing schedules or languages.

WebSphere Portal Server also supports Rich Site Summary (RSS) to either broadcast changes in the portal pages or receive changes made in other channels. Rich Site Summary (RSS) is a lightweight XML format designed for sharing headlines and other web content. Many content providers have been adopting RSS as a simple means of distributing headlines and links to new stories on their sites. RSS is becoming a vital "What's New" mechanism that helps attract users on the Web to the provider's web site. Some examples of web sites supporting RSS are: CBS, ZDNet, BBC, CNET, Rolling Stone, Forbes, USA Today, CNN, Disney, AltaVista, Moreover, and thousands more sites.



Newsbytes Top Stories

- Your Washtech.com Wrap-Up for Thursday, Dec. 14
  Dec 14, 2000 4:01 PM
- HCL Perot to Set Up Technology Campus in India
  Dec 14, 2001 4:05 PM
- Philippine College to Free Up Idle Lands for IT Park
  Dec 14, 2000 3:44 PM
- Kenyan Regulator Shuts down Internet Exchange
  Dec 14, 2000 3:44 PM
- Web Ads to Make Content Sites Profitable by 2005
  Dec 14, 2000 3:45 PM

Comments about NewsBytes?

Figure 12. Rich Site Summary Format

Search

WebSphere Portal Server offers a search service that supports distributed, heterogeneous searches across many data sources. It searches across all the data sources in parallel and combines the results into a unified list of matching documents. This federated search service is called *Extended Search*.

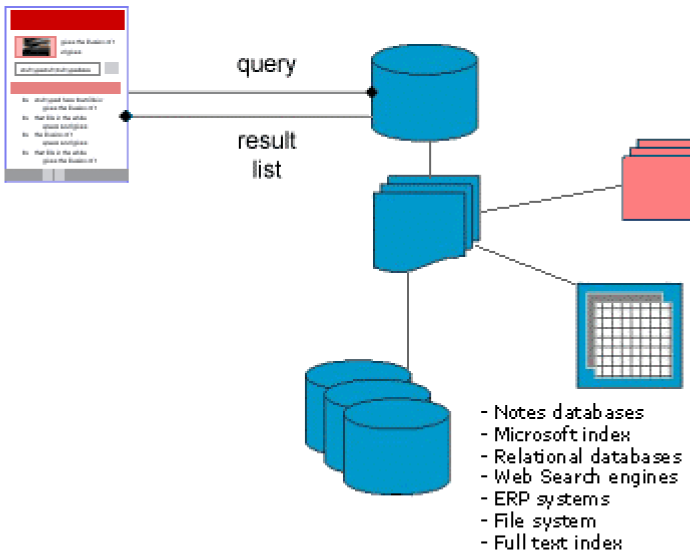


Figure 13. Search and Indexing

Extended Search

With Extended Search, there is no need to create or maintain a central index. Extended Search translates each query into the native search syntax of the target data sources, thus hiding the complexity of the various query languages from the end user. All of the data sources are searched in parallel.

Searches can query and retrieve documents from repositories that include Lotus Notes 4.*x* and 5.x, and Domino.Doc. Extended Search also supports file systems, popular web search engines, Microsoft Index Server, Microsoft Site Server, LDAP Directories, IBM Enterprise Information Portal databases, and relational databases such as IBM DB2, Oracle, and other ODBC-compliant databases.

In summary, with Extended Search users can:

- Search in parallel across Notes domains, legacy databases, local file systems, and popular web search sites
- Get aggregated results presented as a single, ranked list of hits
- Save, reuse, and share searches
- Search across content in various languages and develop your applications in any language
- Refine search results to exclude documents that do not meet certain conditions (e.g. exclude documents that were not created before a specific date or exclude documents that were not created by a specific author).

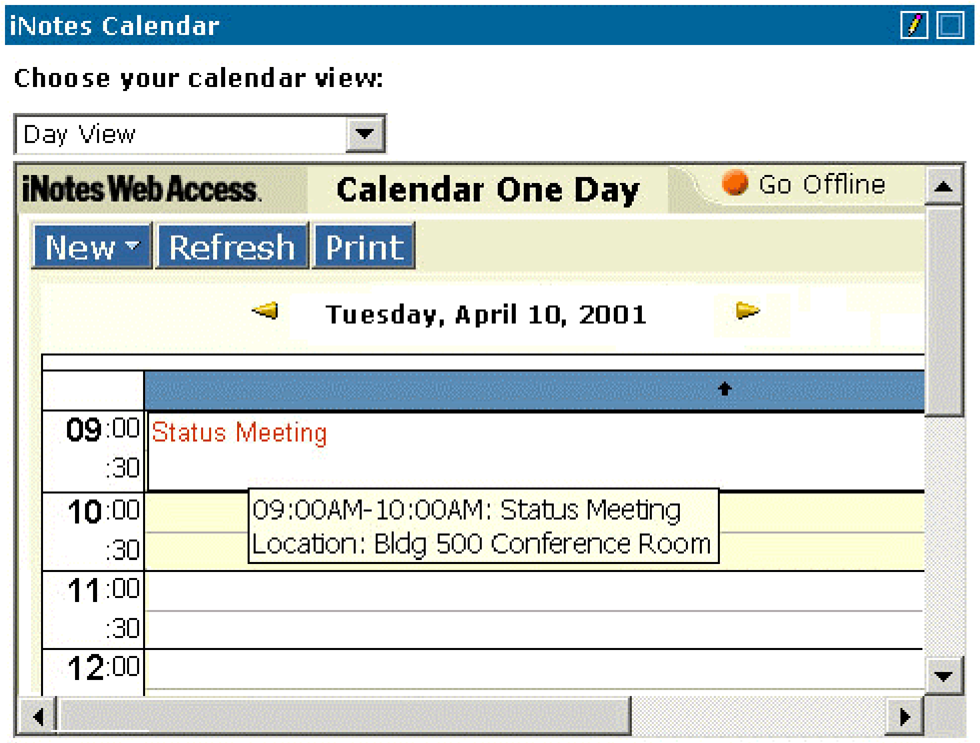
Collaboration

Collaboration starts as e-mail access and grows into team rooms, chat rooms and communities of interest. Collaboration is enabled in WebSphere Portal Server through a rich set of portlets, including portlets for Notes, Exchange, Sametime, and QuickPlace.

The Notes portlets include e-mail, calendar, address book and to-do list. Both iNotes and standard Notes servers are supported. (Lotus iNotes Web Access is a new product that provides the next generation mail and calendar client for web browsers.) Exchange portlets give access to Microsoft Exchange e-mail, calendar, address book and to-do list.

The Sametime portlet offers real-time collaboration through instant awareness, communication, and screen sharing capabilities. In addition, Sametime includes a comprehensive application development toolkit that will enable you to embed real-time capabilities, like live expert links or real-time help features, into your existing applications.

The QuickPlace portlet provides a team workspace in the portal. Teams use QuickPlace to share and organize ideas, content and tasks around any project or ad-hoc initiative. QuickPlace provides a central on-line workspace structured for productivity.


iNotes Calendar

Choose your calendar view:

Day View

iNotes Web Access. Calendar One Day

Tuesday, April 10, 2001

09:00 Status Meeting

09:00AM-10:00AM: Status Meeting
Location: Bldg 500 Conference Room

Figure 14. Lotus Notes Portlet

Enterprise Application Integration

WebSphere Portal Server readily supports the integration of new applications or legacy applications. Portlets written in Java or Java Server Pages can easily integrate with elements of the WebSphere Application Server programming model. This programming model includes comprehensive application development and integration services:

- J2EE is the glue between the application server and various enterprise applications. It offers broad functionality in the areas of application packaging, object services, transaction services, programming standards, and Java messaging services.
- XML provides a simple and nearly universal data representation. WebSphere has strong XML support, including XML parsing, XML style sheet transformation, and SAX event based XML parsing.
- Middleware connectors provide access to many enterprise application systems through Java bean interfaces. MQ Application Integrator and WebSphere's Java Connector Extension (J2C) provides a full suite of application integration connectors, including connectors for CICS ECI/EPI, Encina DE-Light, IMS ITOC, MQSeries, Host-on-Demand, and SAP R/3.

Web applications that already use these middleware services can be reused directly in the portal via the URL portlet. In other words, portlets are able to display existing web page markup such as HTML or XML. In this way, WebSphere Portal Server easily enables reuse of existing web assets.

New portlets can also use IBM's middleware to obtain content for the portal. For example, MQSeries Application Integrator might translate order information from a point of sale system to XML and from the backend, then translate the XML into a formation understood by the inventory and ordering systems, and where it is ultimately displayed in a portlet.

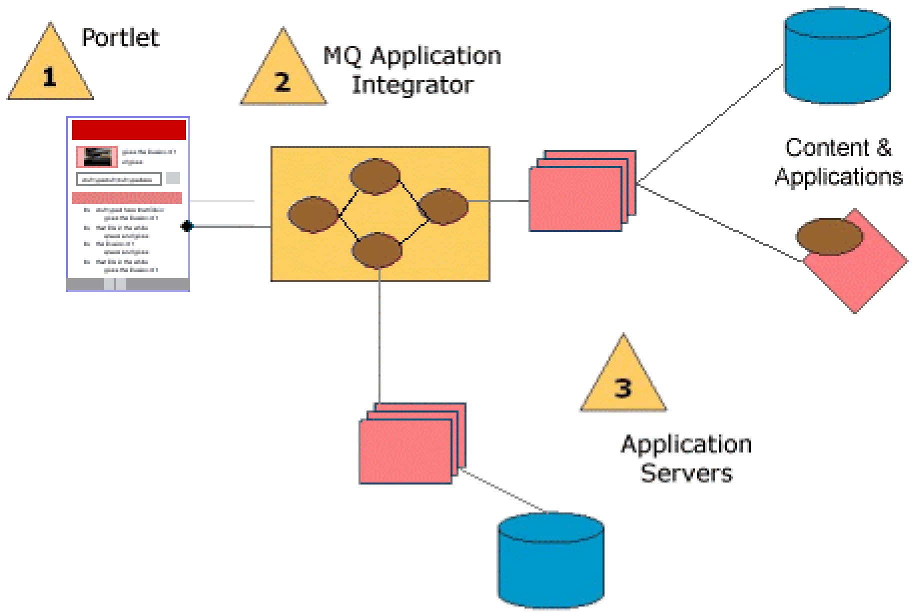


Figure 15. MQ Application Integrator

Web Services

A web service is an interface that describes a collection of network accessible operations. A web service is described using an XML description language, so that the service can be invoked without prior knowledge of the platform, language, or implementation design of the web service.

In a future release, WebSphere Portal Server will provide support for web services. Portlets will be able to use web services to perform their processing, and portal administrators will be able to bind remote portlets as web services, making the remote portlets available in the portal's registry dynamically.

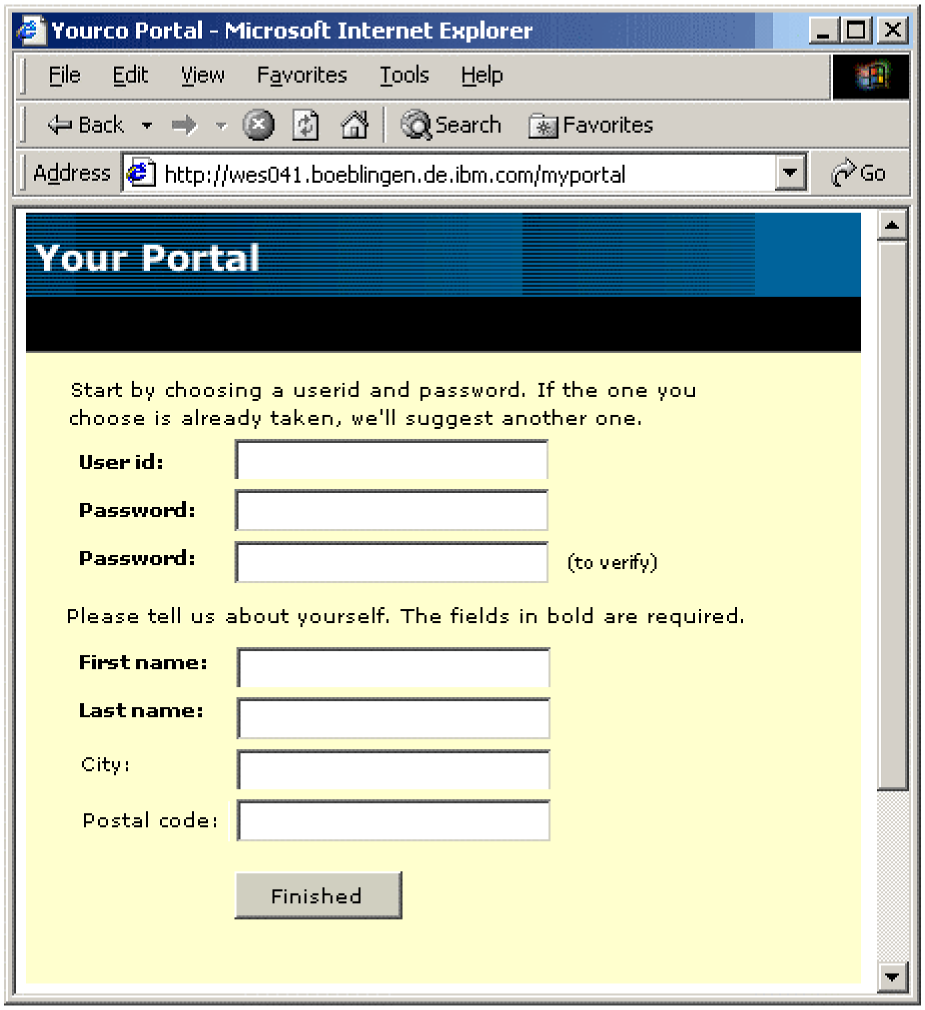
For example, a large corporation might have several different portals, such as an employee portal, a supplier portal and a human resources portal. Each of these portals may choose to publish some of its portlets as web services for access through other the portals.

Individual portlets can also bind to web services in delivering their functionality. For example, a search portlet might query the user for a search string, then use a search web service to search the internet. Or, calendar portlet might act as a front end, providing views for a calendar web service.

Portal Infrastructure

User and Group Management

WebSphere Portal Server provides web pages that allow users to enroll at the portal and to self-manage their own preferences and account information. Alternatively, enterprises can integrate the portal with existing user directories, and may choose to disable the self-enrollment pages.


Yourco Portal - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address http://wes041.boeblingen.de.ibm.com/myportal

Your Portal

Start by choosing a userid and password. If the one you choose is already taken, we'll suggest another one.

User id:

Password:

Password: (to verify)

Please tell us about yourself. The fields in bold are required.

First name:

Last name:

City:

Postal code:

Finished

Figure 16. Self-enrollment Page

Federated User Profile

WebSphere Portal Server provides connectivity between the portal and information in various user directories.

Directory Standards

By default, the LDAP directory stores user-specific data as entries that are associated with the *inetOrgPerson* schema.

- User-specific data; such as the user name, user ID, and password is stored in a lightweight directory access protocol (LDAP) directory. The Java Naming and Directory Interface (JNDI) enables read/write interoperability between WebSphere Portal Server and the LDAP directory.
- Portal-specific data, such as home page settings and portlet settings, is stored in a relational database management system (RDBMS). WebSphere Portal Server supports IBM DB2 and Oracle 8*i*.

WebSphere Portal Server provides a Java bean interface for accessing user information. The User bean acts as an interface to a stateless session EJB, which in turn, acts as a consolidation interface to multiple back-end EJB classes, each responsible for retrieving a portion of the user data. Figure 16 shows the general structure.

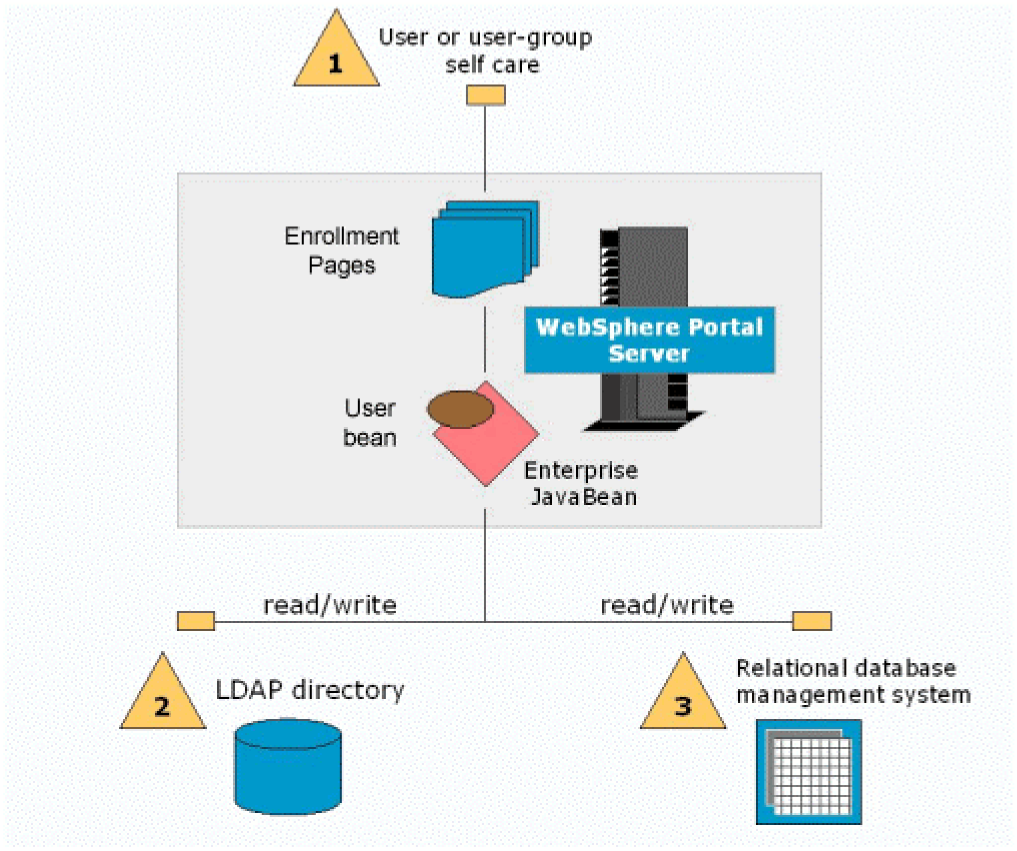


Figure 17. Federated User Profile

User, Group, and Access Beans

The portal server provides a Java bean interface for users, group memberships, and access control.

The back-end implementations can be replaced to support existing infrastructure or third-party products.

Referring to figure 17:

1. The user and group self care interface begins with the portal's enrollment pages. These pages can be modified to collect additional attributes or to allow the user to specify group membership information.
2. The User bean class calls the backing EJB to store or retrieve the basic user information (such as name or city) in the LDAP directory.
3. The bean also calls another EJB to store or retrieve the portal-specific settings for the user, such as the user's list of portlets and their settings.

The EJB implementation classes can be exchanged transparently, so that third-party LDAP servers, alternate schemas, and other external data stores can be easily integrated into the portal's user management system. Each customer can replace the implementation classes to match where their data is stored. The source code for WebSphere Portal Server's default implementation is provided as an example.

Existing Directories

Because many companies already have databases that contain user data, WebSphere Portal Server enables the federation of user data from multiple, existing repositories. For example, in many business-to-employee portals, a directory of user information already exists, so it is desirable to access user data in a read-only LDAP directory and then store additional user data in a relational database tables.

In this scenario, illustrated in Figure 17, user self-registration is disabled, so WebSphere Portal Server does not update the LDAP directory at all.

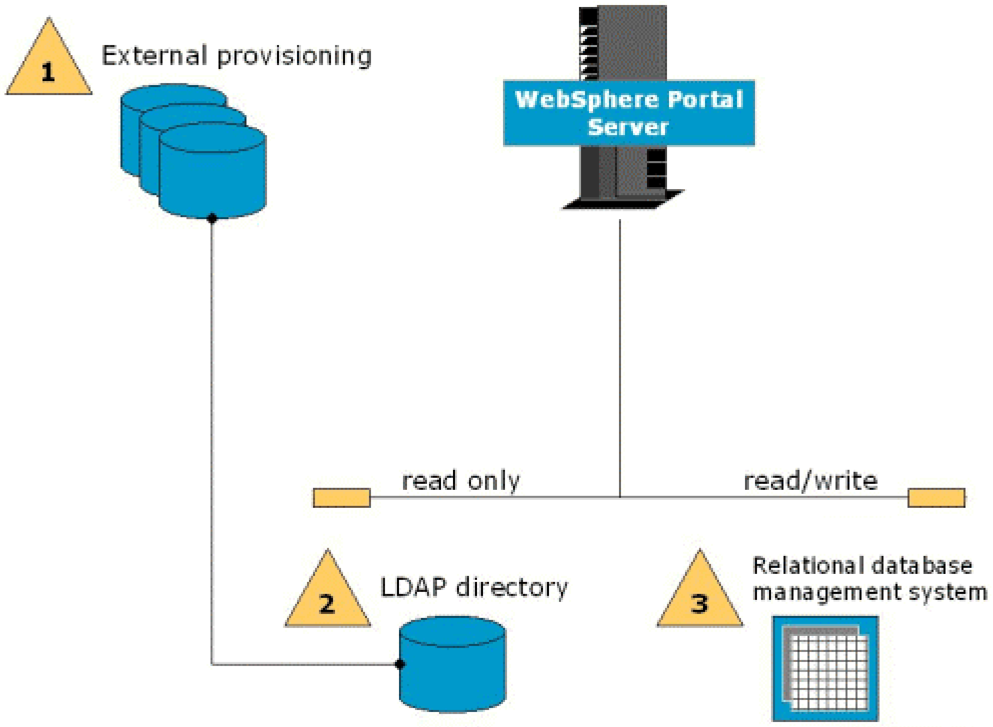


Figure 18. External Provisioning

Another important scenario involves the use of WebSphere Everyplace Suite, which includes Tivoli Personalized Services Manager, an integrated subscriber management system. When TPSM is used, the architecture is usually a hybrid of the two scenarios outlined so far.

The hybrid scenario uses self-enrollment pages to populate the TPSM repository. TPSM provisions the LDAP directory, and the portal server does not update LDAP at all. The implementation details of this scenario are isolated in the implementation classes of the user EJB, but the User Java bean interface remains unchanged.

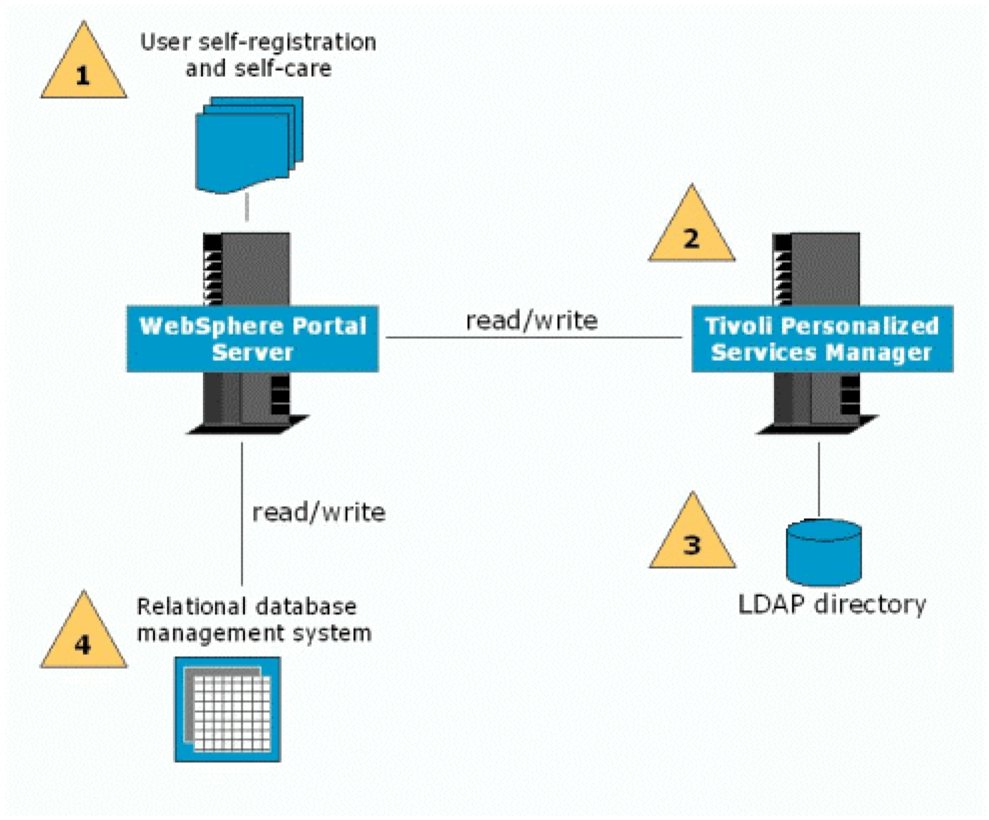

1
User self-registration and self-care
2
WebSphere Portal Server
read/write
Tivoli Personalized Services Manager
read/write
3
LDAP directory
4
Relational database management system

Figure 19. Advanced Subscriber Management

Group Management
The user bean class also gives access to group information. Users may be classified into one or more groups. Group membership information is stored in LDAP, and the group names are defined using the LDAP directory's administration interface.

Delegated administration of users is through the LDAP administration interface. Access control permissions are assigned through a policy specification portlet interface (or through the administration interface of a third party authorization server, if applicable). The portal server protects access to portlets, and the backing LDAP server protects access to users and groups.

Security

Authentication
The portal server should be configured so that incoming requests pass through an authentication component such as WebSphere Application Server security, WebSeal/Policy Director, Netegrity SiteMinder, or other authentication proxy servers. A separate alias is configured to allow anonymous requests for new users or for users who have not yet logged in.

Authentication proxy servers can be integrated with WebSphere Application Server through its Trust Association Interceptor APIs. This provides a highly secure and uniform interface to WebSphere Portal Server. Examples of authentication proxy servers that can be supported this way include IBM Policy Director's WebSeal and WebSphere Everyplace Suite's Authentication Server.

Authentication and Authorization Standards

WebSphere Portal Server uses standard Java Security APIs for its authentication, authorization, and single sign-on features.

WebSphere security and the IBM authentication proxy servers are configured to use the portal's LDAP directory to authenticate users. Once the authenticated user information is available, WebSphere Portal Server stores various credentials, including LTPA tokens, CORBA credentials, user id and password, etc. These credentials are available to portlets through a standard JAAS API interface, so that they can be passed to back-end applications to achieve single sign-on. This avoids having the user prompted again for authentication.

Another technique for integrating third-party authentication servers is to replace the portal server's login action class. The replacement login action would inspect the HTTP header information (depending on the details of the third-party product) to handle the login.

Authorization

The access control interface of the portal integrates with the user and group beans to find out which portlets a user is authorized to use. The aggregation modules and the page customizer use this information to filter the list of portlets that are displayed.

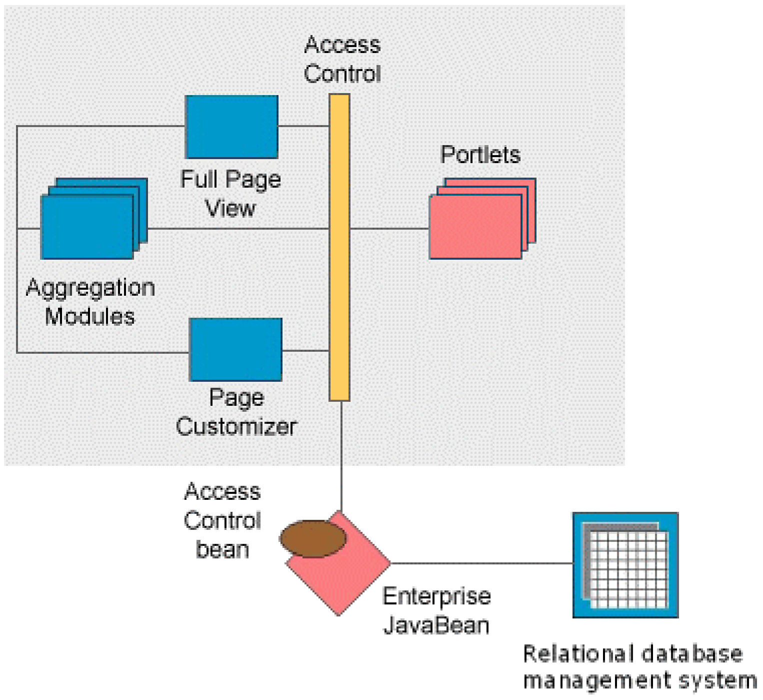


Figure 20. Access Control

The access control system consults the policy engine to determine access rights; access control lists are stored in database tables. Administrators can define access control list that manages access to each portlet. The administrator interface is itself a portlet whose access is managed through these same access control lists.

The access control model is additive, meaning that a user can be granted access to a resource either directly or indirectly (as a result of granting access to one of the user's groups). The priority or hierarchy of group membership has no bearing on access control rights; if a user is granted access by any means, then access is granted. For example, if an access control list is changed so that a portlet is removed from a higher-level group, this would not revoke a specific user's access rights if access had been granted to the user through some other group membership.

By replacing the access control system's implementation classes, third party access control servers (such as Tivoli Policy Director or Netegrity SiteMinder) may be used instead of the default WebSphere Portal Server access control implementation. Source code is provided to make this replacement easier. In such cases, the third party access control system would provide its own administration interface for managing the access control lists.

Page Transformation

As a separately purchased product, IBM offers the WebSphere Transcoding Publisher, which can transform the markup produced by WebSphere Portal Server to target additional devices. WebSphere Transcoding Publisher can be used in several different configurations:

- It can be installed in a proxy configuration, so that it transforms outbound markup before the markup is sent to the browser. This configuration is useful when making the portal accessible through personal digital assistant (PDA) devices, for example. It is also useful for WML/WAP devices, where the payload size is very limited, since WebSphere Transcoding Publisher can split a large WML deck into several smaller ones.
- Another useful way to use Transcoding Publisher is to develop a portlet that calls the transcoding service during page aggregation. This way, individual portlets can produce their mark-up automatically, rather than requiring custom JSP views or style sheets for each mark-up. This structure is particularly useful for rapidly changing content, such as news, or for clipping content from existing HTML pages. An advantage of this configuration is that only specific portlets perform the transformation step, rather than always processing the entire page.

In a very similar manner, WebSphere Translation Server can be configured to transform content into other human languages, such as transforming English to Spanish, for example.

Performance

Typical Configurations

The simplest configuration for prototyping or proof-of-concept installations of WebSphere Portal Server is quite minimal. An evaluation installation requires only WebSphere Application Server (Advanced Edition), IBM SecureWay LDAP Directory, and DB2, all running on a single server. The WebSphere Portal Server installation program installs everything else that you need, including WebSphere Personalization.

Larger installations would include the same basic components load balanced across several production servers for greater reliability and scaling.

Additional Infrastructure

WebSphere Everyplace Suite provides additional infrastructure for service providers and enterprises. Together WebSphere Everyplace Suite and WebSphere Portal Server deliver a portal to wireless devices such as phones and personal digital assistants operating over a variety of wireless networks such as GSM and GPRS. WebSphere Everyplace Suite provides a secure and scalable framework for network connection, authentication, device and subscriber management, data transformation, load balancing, and caching.

Figure 20 shows an example of a typical configuration.

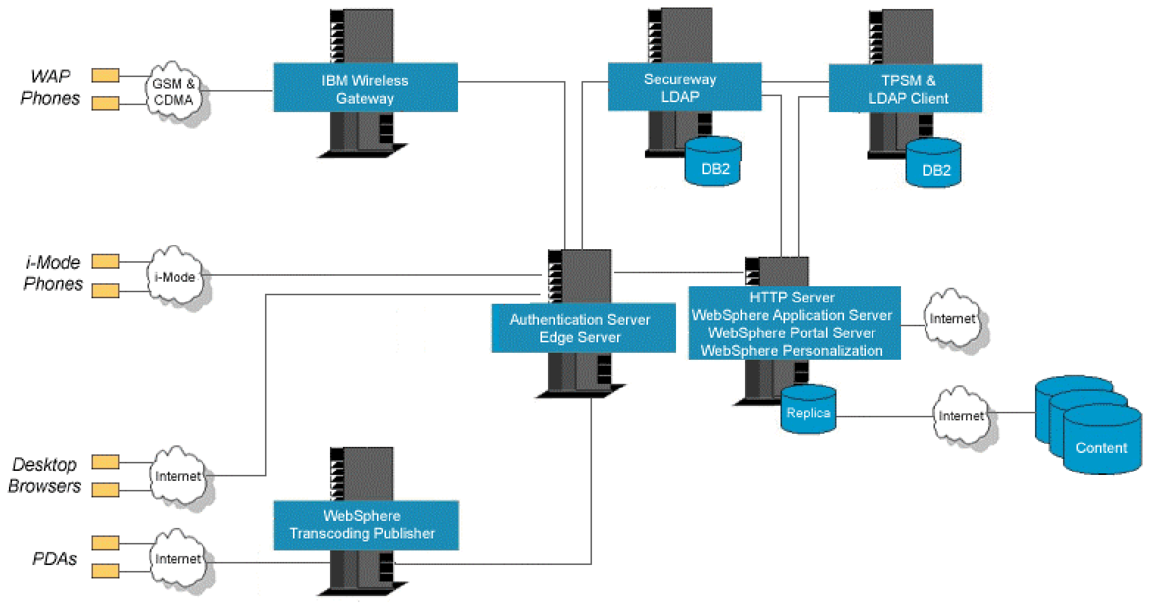


Figure 21. WebSphere Everyplace Suite Configuration

Content Caching

WebSphere Portal Server can work with live content from the Internet. For better performance, it uses a cache to keep local copies of remote documents. The disk cache contents are refreshed either at an interval set by the portal administrator, or based on an interval specified in the document's HTTP header. OCS channels are also cached. The channel entries specify the frequency and period for updating each channel's content.

Load Balancing

WebSphere Portal Server runs in a WebSphere Application Server cluster to achieve scalability and reliability.

The facilities of WebSphere Edge Server can also be used for additional load balancing and high availability. To achieve optimal fail-over, persistent sessions are used, storing all session data in a shared database. If one portal server fails, the Edge Server's network dispatcher component will detect the situation and will balance further requests between the remaining portal servers.

To achieve optimal performance, it is best to always route requests from a single client session to the same server, so that the session data will be retrieved more efficiently. In practice however, proxy servers interfere with the load balancing because they collapse IP addresses, replacing the client's true IP address with the address of the proxy server. The result is that all clients connecting via the same proxy would be routed to the same portal server all the time.

To achieve "sticky" sessions with reasonable load balancing, the network dispatcher component can be set up in a special way. It is configured for load balancing without sticky sessions on initial requests. When the first request from a client is received, the server redirects the client back to itself, thus bypassing the network dispatcher. The client directly communicates with the server and the session context need not be shared, as a particular client will always talk to the same server.

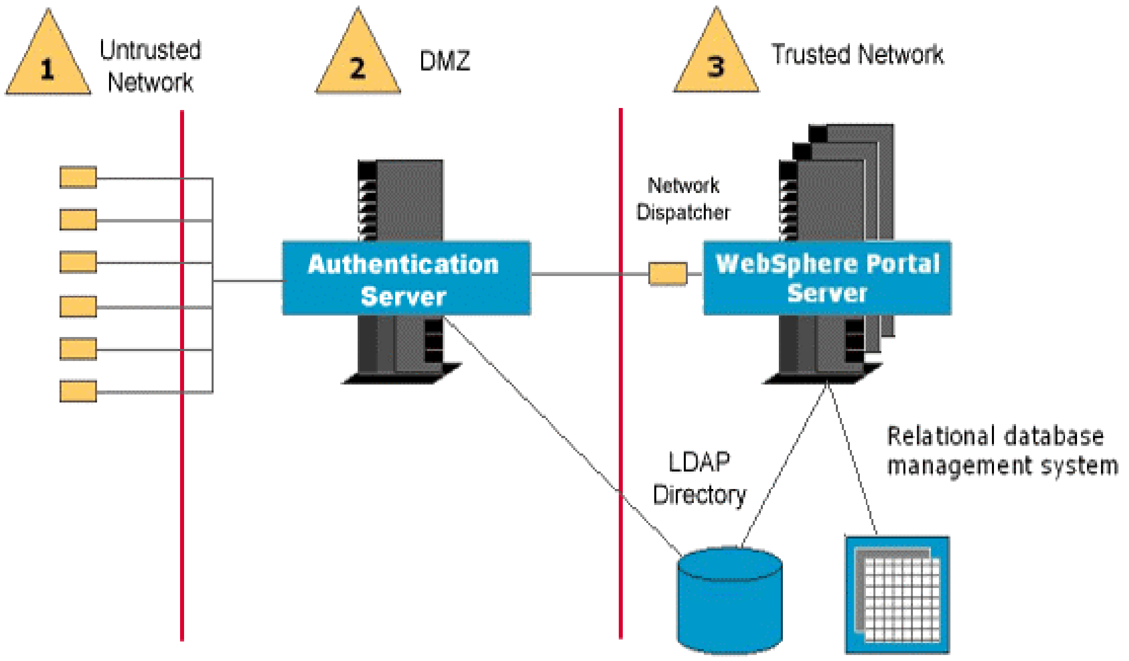


Figure 22. Load Balancing

Of course the direct communication between client and server will not work with firewalls and an authentication proxy in the demilitarized zone. So an alternative configuration has initial requests from clients being load balanced by the network dispatcher. Portal servers then force sticky sessions by redirecting the client to their own alias.

Summary

The WebSphere Portal Server framework simplifies many of the tasks involved in building complex portal web sites. It targets multiple devices and helps companies leverage and reuse existing their web assets.

WebSphere Portal Server provides a scalable, secure, and extensible infrastructure for building a wide range of portals, including business-to-consumer, business-to-employee, and business-to-business portals. IBM offers many other products that complement WebSphere Portal Server:

- Extended Search is included with WebSphere Portal Server, providing a high performance search capability across Internet document sources.
- Enterprise Information Portal provides tools for advanced searching, categorizing, and summarizing content. It offers federated search and the

ability to perform create, read, update, and delete operations across multiple data stores.

- **Interwoven TeamXpress** provides content management tools, including workflow, and presentation templates. TeamXpress synchronizes content from the content management repository to the production portal file system and databases.
- **Lotus iNotes Web Access** provides a web interface to Lotus Notes e-mail, calendar, address book and to-do lists. WebSphere Portal Server provides portlets for each of these functions.
- **Lotus Sametime** provides instant messaging, shared white boards, and application sharing for electronic meetings, which are accessible through portlets. **Lotus QuickPlace** provides team workspaces for sharing and organizing ideas, content and tasks.
- **WebSphere Everyplace Suite** includes infrastructure and services for large portal installations. It provides a secure and scalable framework for network connection, authentication, device and subscriber management, data transformation, load balancing, and caching. A few of the products included with the Everyplace Suite are:
    - **WebSphere Edge Server** includes load balancing and content caching services needed for highly scalable and fault-tolerant portal installations.
    - **WebSphere Transcoding Publisher** adapts and optimizes content for new mobile devices or other browser environments. In particular, it is a useful addition for customers who are using WebSphere Portal Server's mobile device support because optimizes the deck structure of WML portals.
    - **SecureWay Policy Director** provides security infrastructure, including authorization and authentication services.
- **WebSphere SiteAnalyzer** is useful for measuring the activity and effectiveness of the portal. It reports on structural information, such as broken links and page sizes. It also processes web server logs to reveal how the portal site is being used, who is using it, where they enter or exit and how they navigate within the site. The information is mined and stored in a database or displayed in reports.
- **WebSphere Studio** and **VisualAge for Java** are productive development tools for working with the portal.