



IBM Global Services

Technical Portal Forum 2005 – Prague

Best Practices for a Successful WebSphere Portal Implementation

Dusan Smolej / IT Architect

IBM Certified Solution Developer

IBM Certified System Administrator

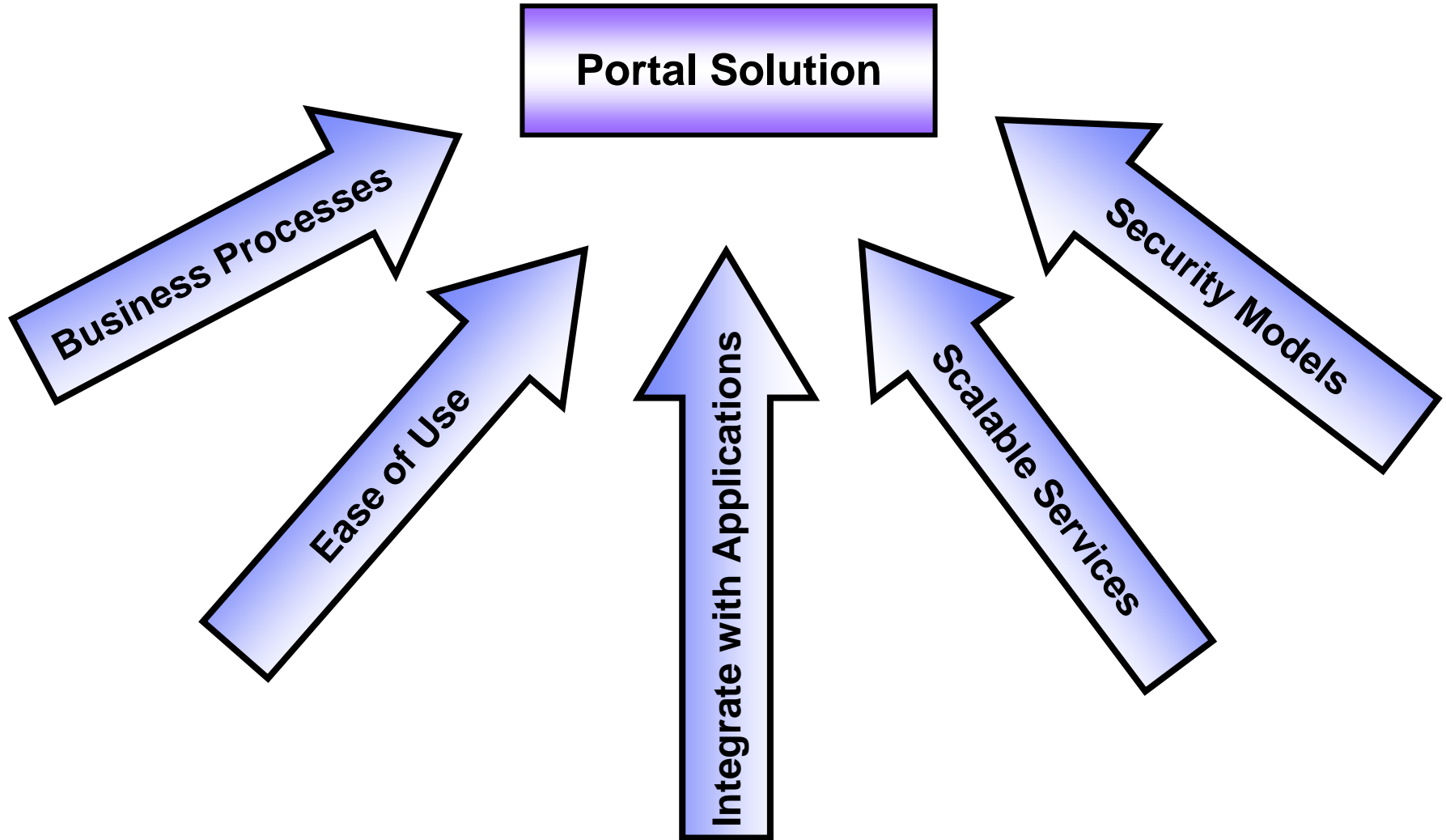
for WebSphere Portal for Multiplatforms



Agenda

- **Installation Scenarios**
- **High Availability Design**
- **Project Management**
- **Portal Deployment**
- **Portal Navigation**
- **Performance**
- **Security**
- **Caching**
- **Portlet Development**

Common Implementation Themes



Portal Solution Components

Products and components that are packaged in each WebSphere Portal offering



Product or component	Enable	Extend
WebSphere Portal	X	X
WebSphere Translation Server	X	X
WebSphere Application Server products	X	X
IBM Lotus Domino Enterprise Server		X
IBM Lotus Extended Search		X
IBM Lotus Instant Messaging and Web Conferencing		*
IBM Lotus Team Workplace		*
IBM Workplace Web Content Management	X	X
IBM Rational Application Developer	X	X
IBM Tivoli Directory Server	X	X
IBM Tivoli Web Site Analyzer		X
Cloudscape Database	X	X
Document Manager	X	X
DB2 Universal Database Enterprise Edition	X	X
Lotus Collaboration Center	X	X
Lotus Collaborative Services	X	X
Member Manager	X	X
Portal Personalization	X	X

*Lotus Instant Messaging and Web Conferencing and Lotus Team Workplace are limited portal use only.

Installation Scenarios

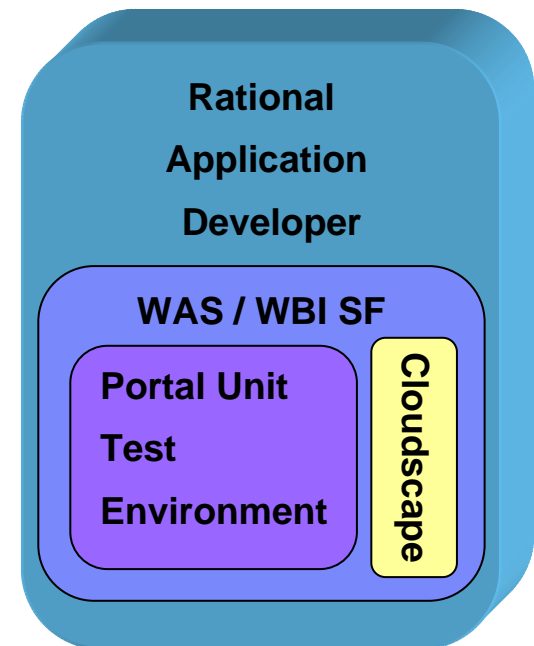
- **Single machine**
 - Proof-of-concept
 - Development
- **Multi-machine**
 - DMZ/ HTTP server separation
 - Separation of LDAP, database
 - Search, Content Manager, Business Process Management
 - Collaborative configurations
- **Special considerations**
 - Security architecture (Enterprise Security manager)
 - High Availability requirements (HA)

Environment Preparation

- **Design a development environment**
- **Design a test environment**
- **Design a Q/A or integration test environment**
- **Design a production environment considering:**
 - Scalability
 - Failover
 - Workload distribution
- **Plan for security**
- **Establish a backup and recovery strategy**

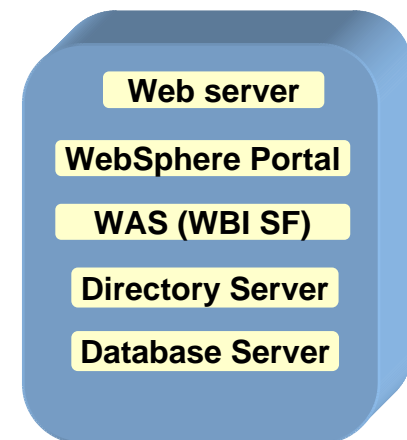
Development Environment

- **All necessary components are installed and configured on one physical machine**
 - WebSphere Studio
 - WebSphere Portal Test Environment
 - Cloudscape Database

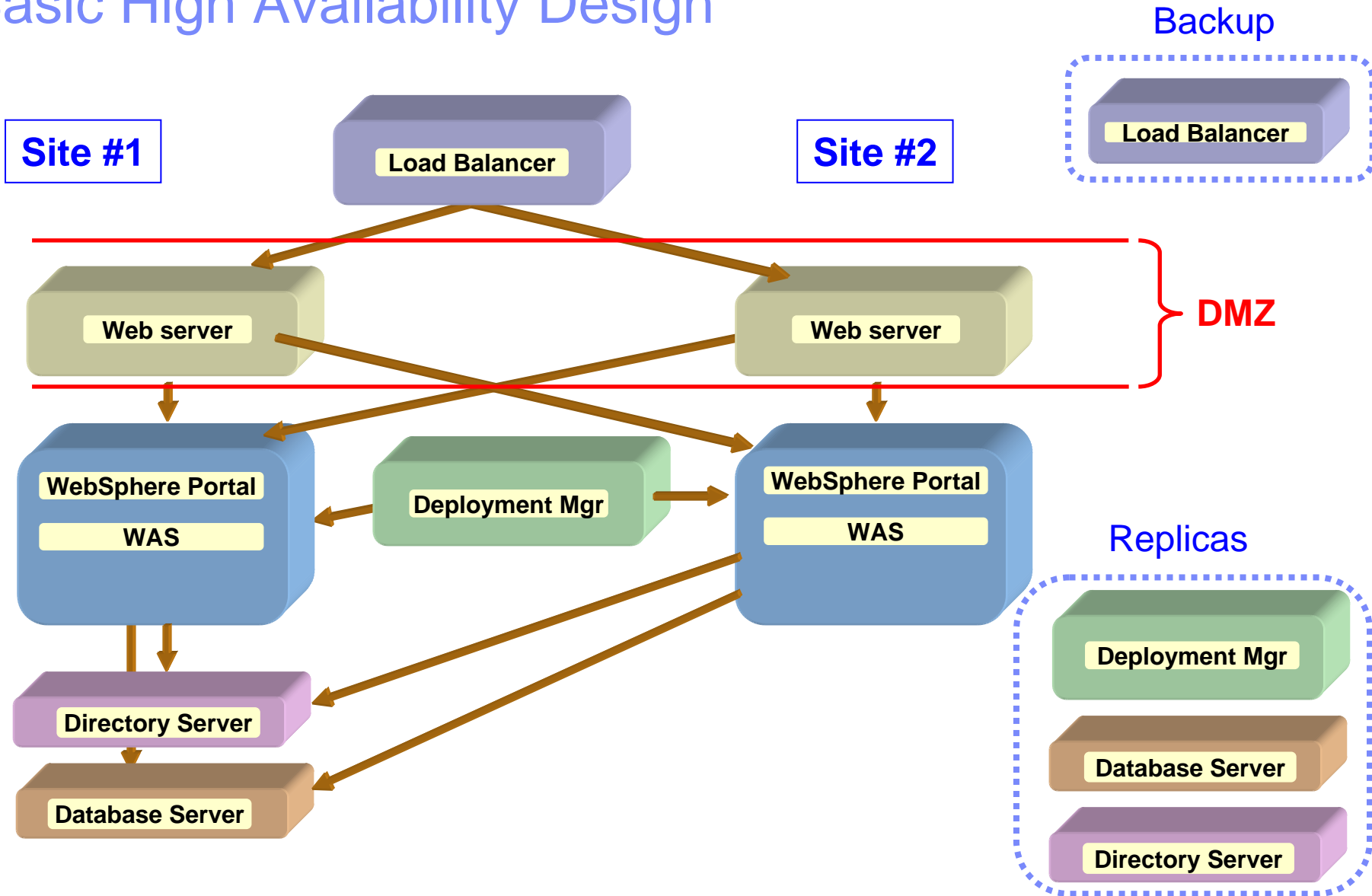


Sandbox Environment

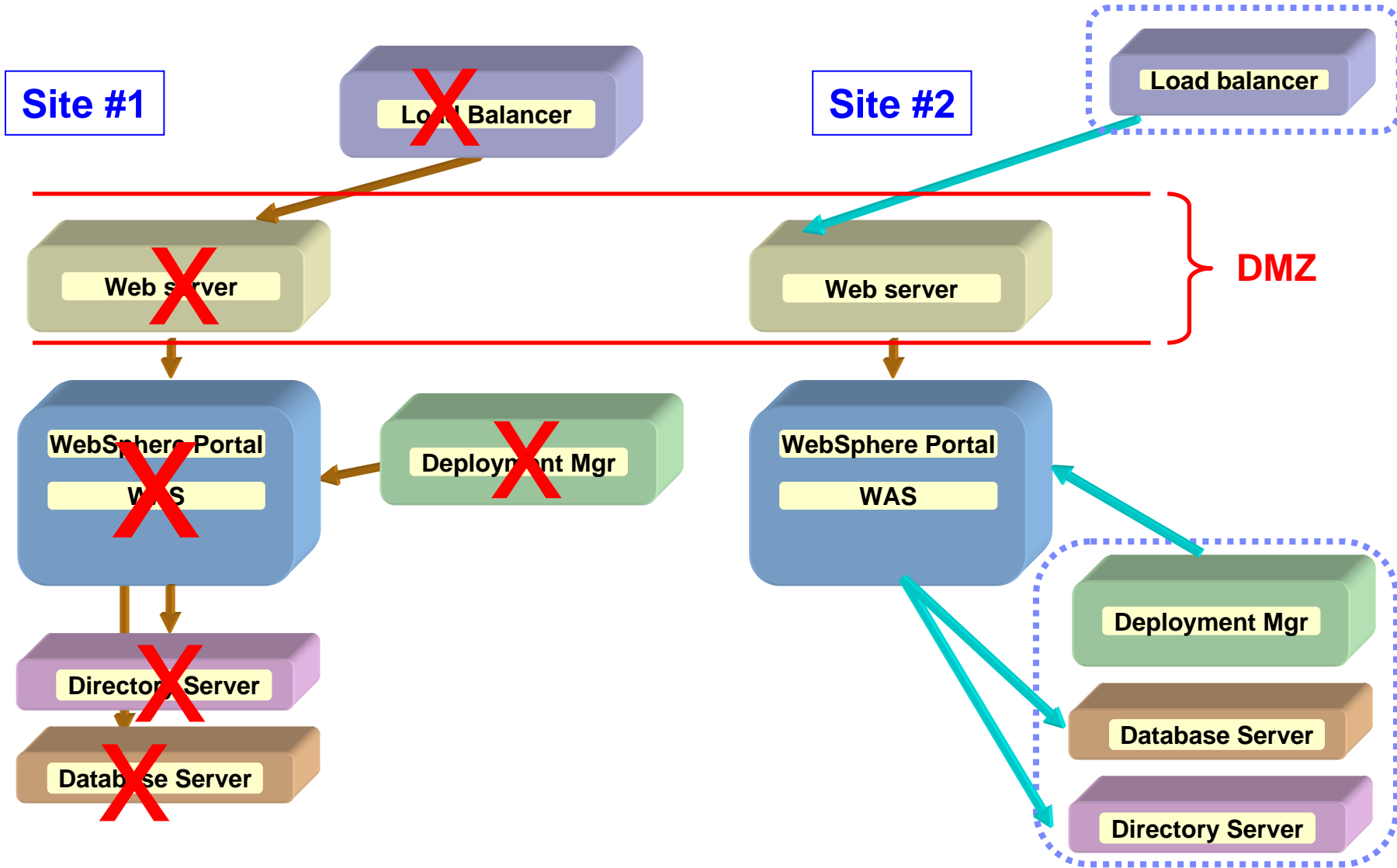
- **All necessary components are installed and configured on one physical machine**
 - WebSphere Portal
 - WebSphere Application Server (WBISF)
 - IBM HTTP Server
 - IBM Directory Server
 - IBM DB2 Universal Database



Basic High Availability Design

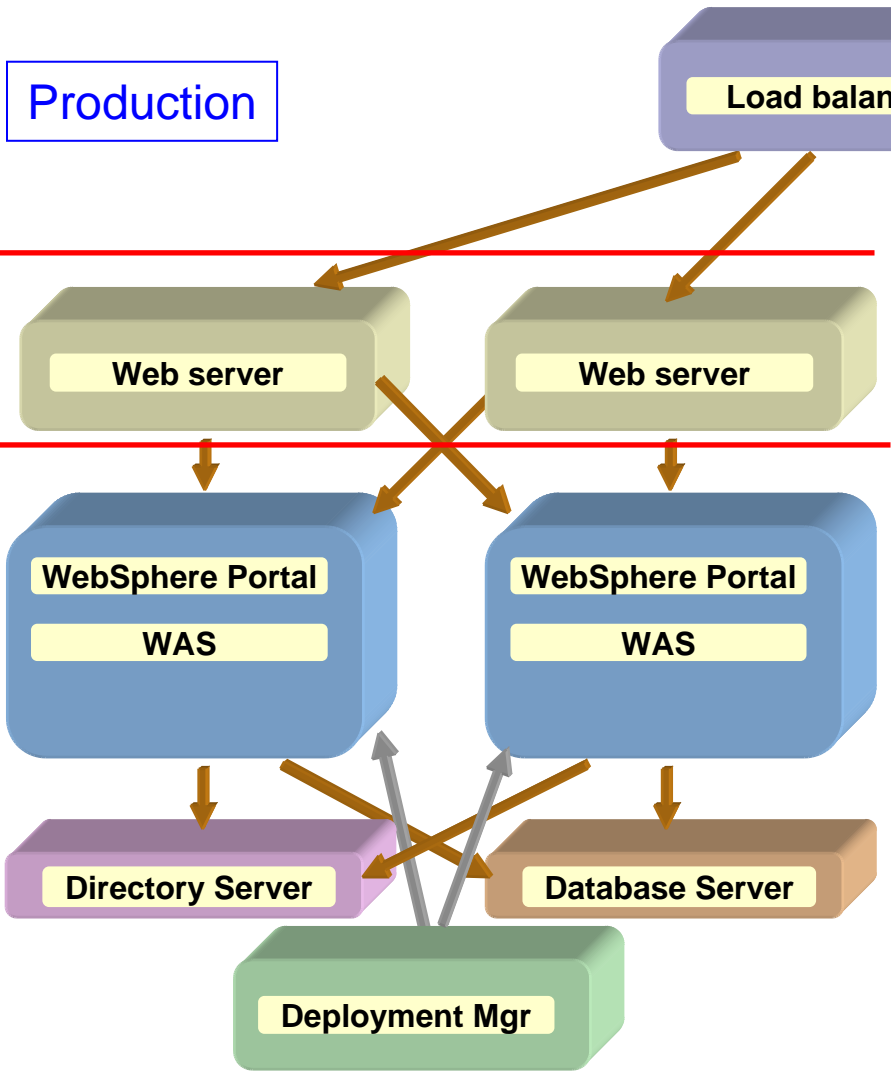


Basic High Availability Design - Failover

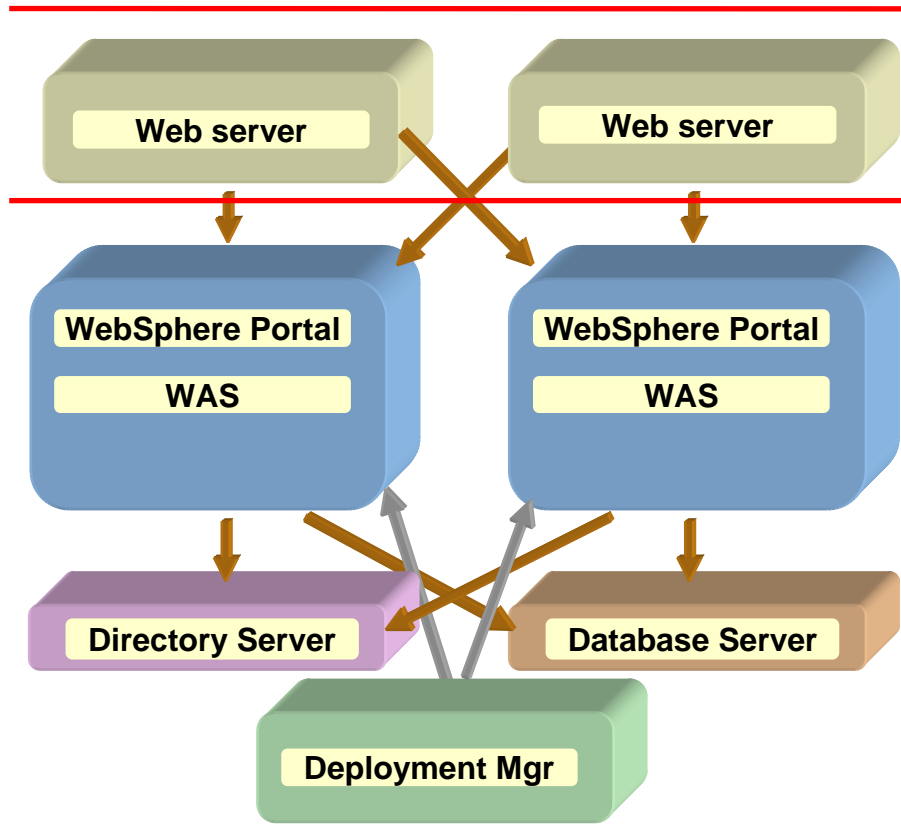


High Availability - Multiple Personalities

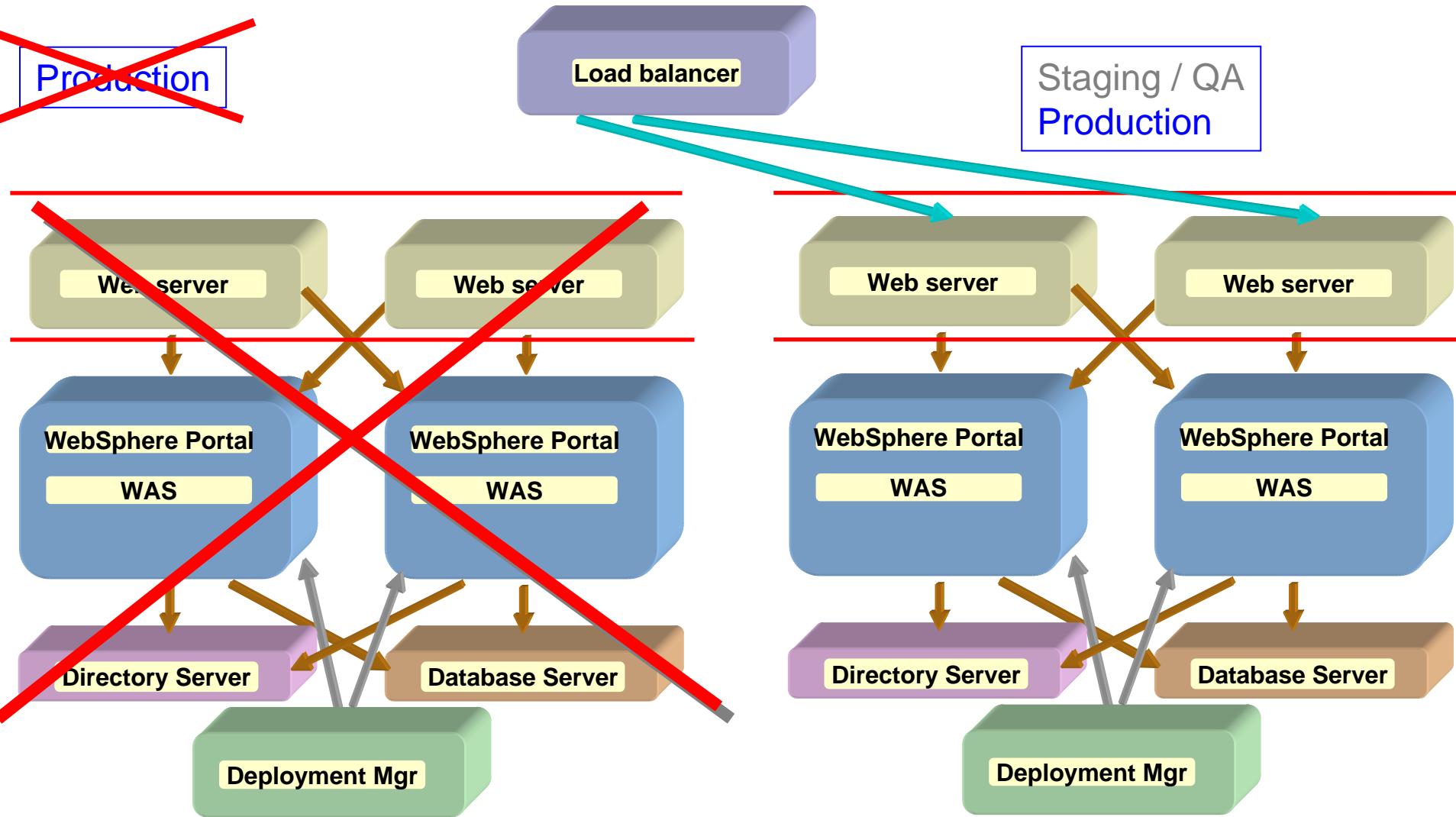
Production



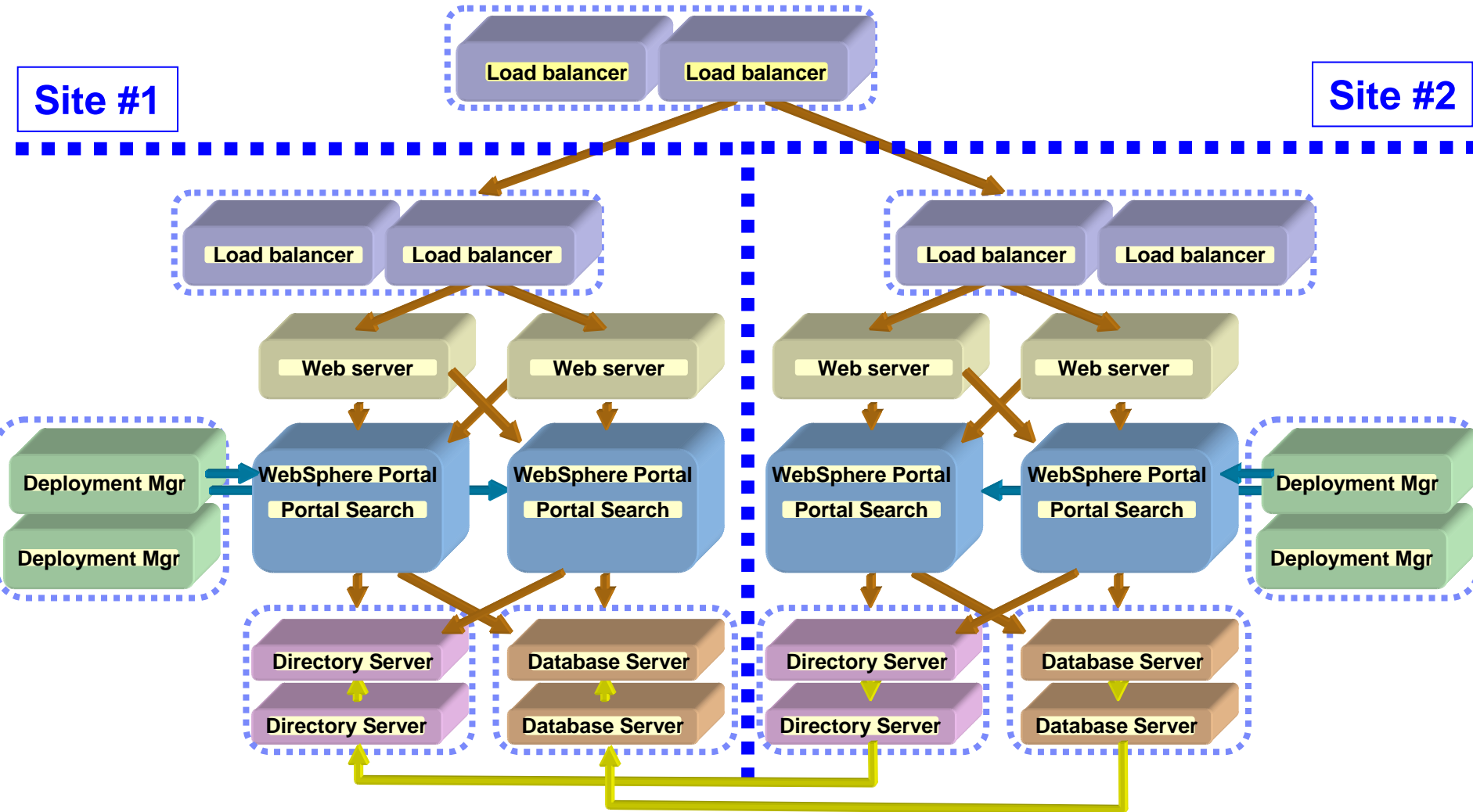
Staging / QA



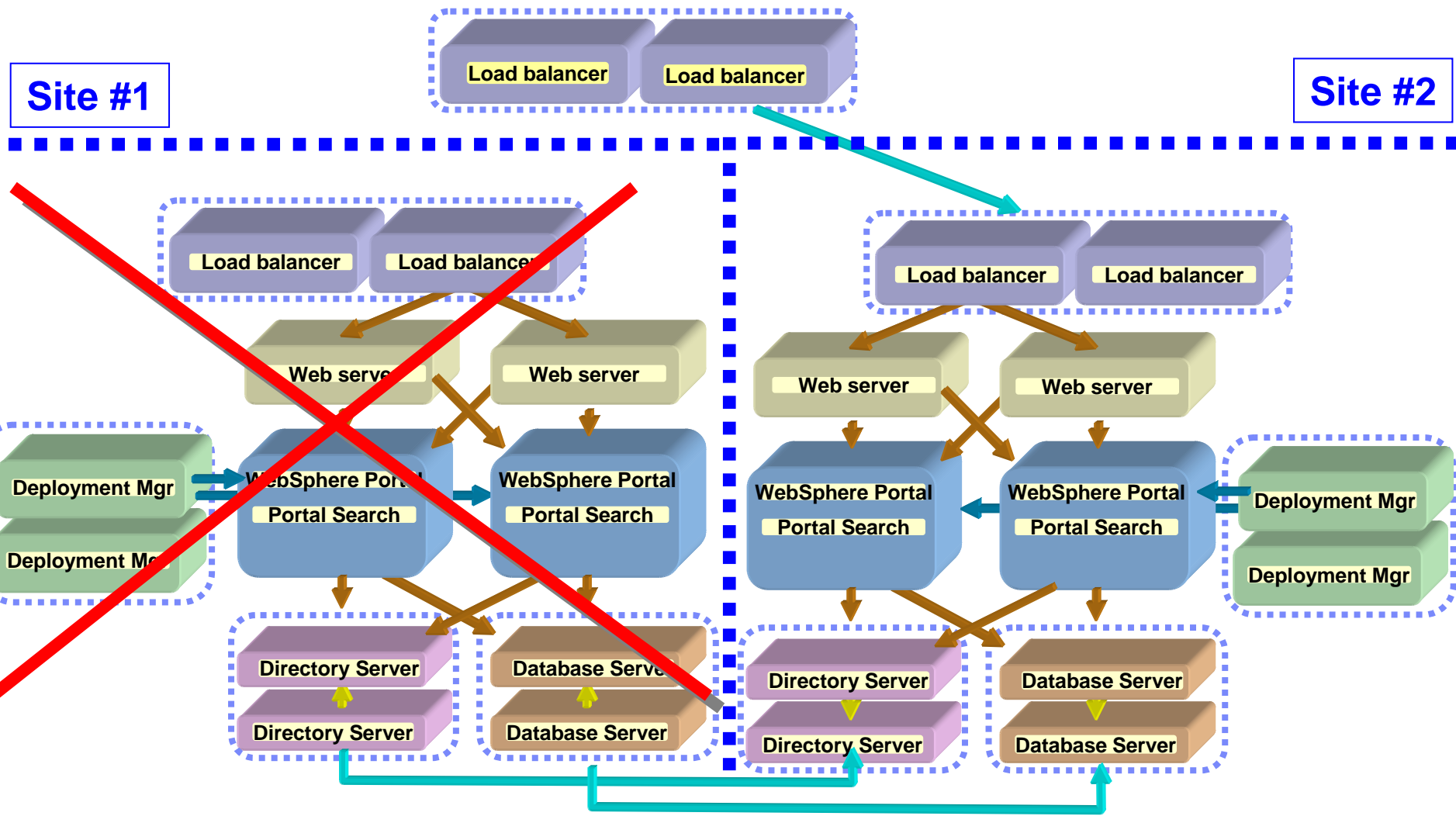
High Availability - Multiple Personalities - Failover



High Availability Gold Standard



High Availability Gold Standard - Failover

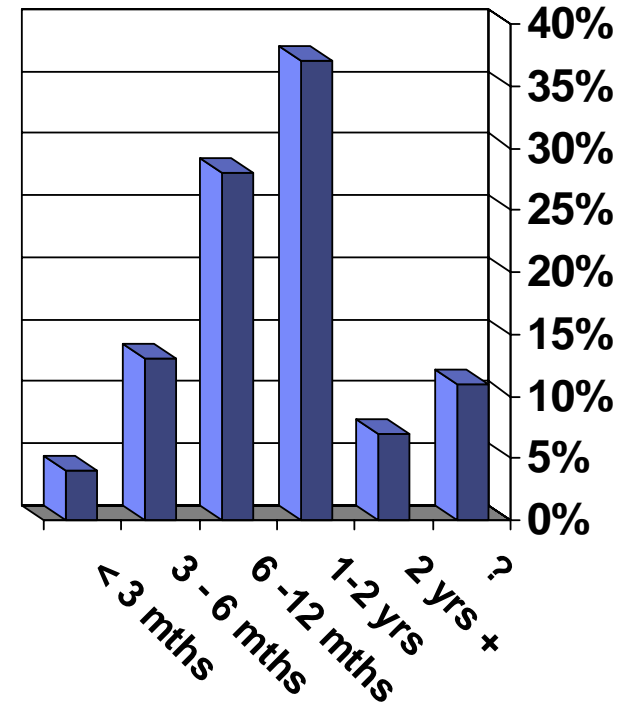


Best Practices – Project Management

Be Realistic

- Most projects take a year!
- Need multiple phases that each a layer of complexity
- Start small. Build from the base components first and install added functionality later (plan four months minimum for the first pilot release)
- Iterate ...
- Allocate minimally 2 months for performance testing

Portal Project Length



Best Practices - Project Management - continued

WebSphere Portal must be treated like any other complex software project

- A defined methodology should be chosen that supports a rapid iterative process
 - Use a visual methodology (screenshots, wireframes, and so forth)
 - Consider an agile development methodology such as XP
- Work must be scoped based on defined requirements
 - Hold an architecture workshop
 - White board the project to get user community to grasp concept and scope and to close the requirements
- Allocate and track resources
 - Plan for vacation and Murphy's Law
- Use Quality Metrics
- Update your plan. It is a working document.

Identifying Business Requirements

What are the business drivers and how are they being measured?

- Improved access to resources
- Reduction in cost of delivery for business processes
- Enable people to work more effectively
- Support collaboration and improve access to training

Where are the technology drivers for IT and how are they being measured?

- Simplified purchase, use, and deployment of technology
- Reduce complexity and cost of managing content, applications, and standalone tools
- What are your availability and reliability requirements?

What criteria will your sponsors use to measure this success?

- What are their expectations?
- What would be necessary to exceed their expectations?

Identifying Content and Document Management Requirements

- **What content do you want to display on WP?**
 - Where is the content stored?
 - Who needs to create and change it and who views it?
- **What documents do you want to store?**
 - How many documents will be created each day?
 - How large is each document?
 - How many users will be accessing them (concurrently and total) and what will they be doing with them?
 - What roles will they be assigned?
 - What will the workflow look like?
- **What are your expectations regarding time to access, view, and store a document?**
- **Personalization**
 - Do you want customization or personalization?

Client, Network and Server Architecture

Need to define a network and server architecture for

- Development
 - Usually need a single server and Rational Application Developer V6
- Test and Staging
 - Staging should be the same as production including routers and firewalls
 - Same OS Version including patches
- Disaster Recovery
- Do clustering last. Do not initially install more than one instance of Portal on one machine/partition unless it is for migration.
- WebSphere Portal loves RAM. The more RAM, the better.

Identify the clients you are going to support

- Remember some don't support all the JavaScript features or CSS2
- Note the limitations of WAP/PDA clients

More Information

Creating a new portal, Part 1: Getting started

Taking the first steps toward a successful portal project

- http://www-128.ibm.com/developerworks/websphere/library/techarticles/0508_bernal/0508_bernal.html

Creating a new portal, Part 2: Conducting a portal workshop

Getting the team together to plan the portal

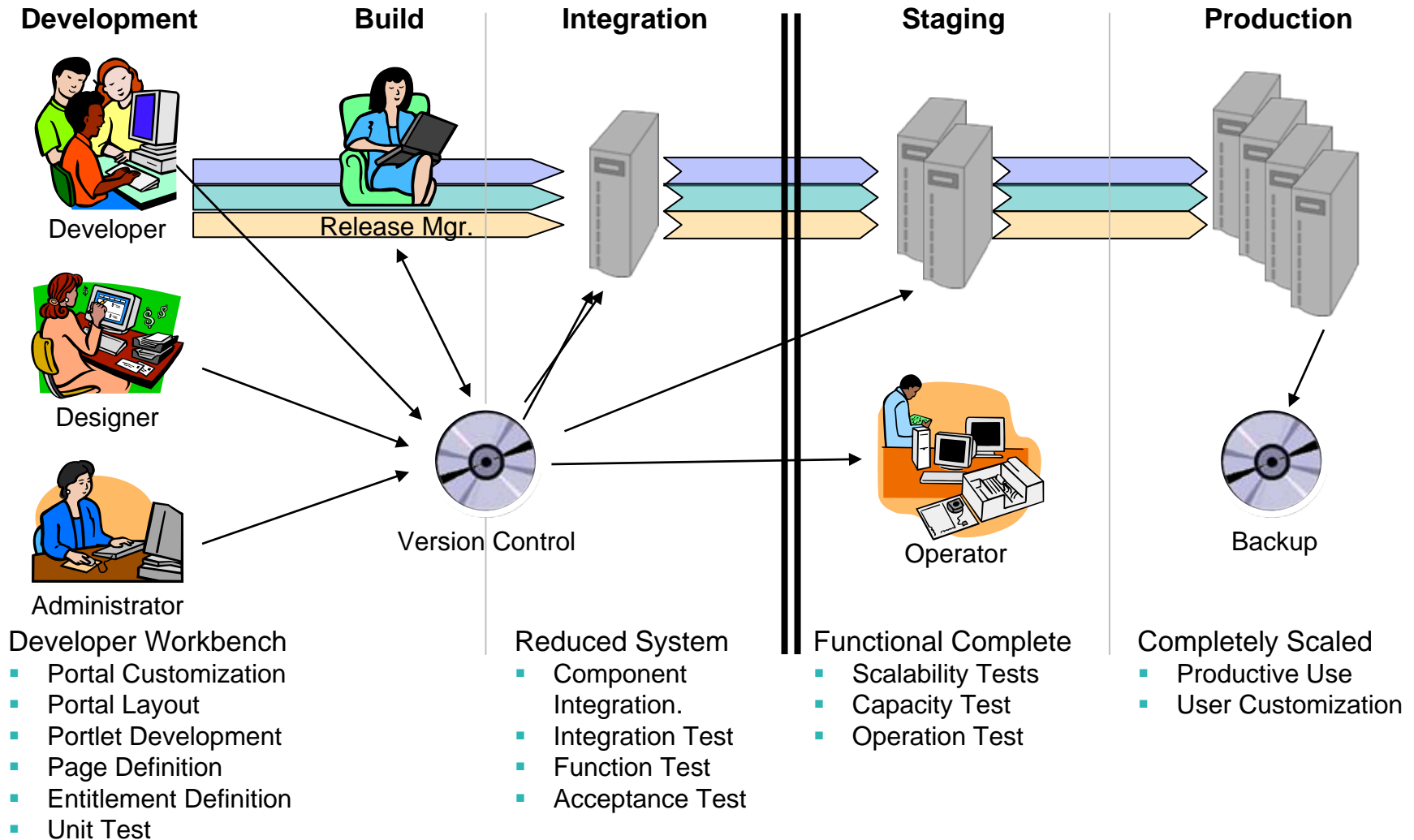
- http://www-128.ibm.com/developerworks/websphere/library/techarticles/0509_bernal/0509_bernal.html

Creating a new portal, Part 3: Estimating and tracking

See how to estimate project tasks and then how to keep your portal project on track.

- http://www-128.ibm.com/developerworks/websphere/library/techarticles/0511_bernal/0511_bernal.html

Best Practices – Portal Deployment



Build and Deployment Process Example

Portlet Developer

- Create portlets and portlet applications on a development system.
- Perform unit tests of portlets and portlet applications on a development system.
- Deliver portlets and portlet applications into the version control system (VCS).
- Create portal artifacts on a development system.
- Perform unit tests of portal artifacts on a development system.
- Deliver portal artifacts into the VCS

Portal Designer

- Create portal look and feel artifacts (Themes, Skins, Screens, Help Pages) on a development system.
- Perform unit tests of portal look and feel artifacts on a development system.
- Deliver portal look and feel artifacts into the VCS.

Build and Deployment Process Example

Portal Administrator

- Select ready made portlets and portal artifacts for use.
- Deliver ready made portlets (including administrative portlets) and portal artifacts into the VCS.
- Check out all artifacts from the VCS.
- Install artifacts onto development system (content master).
- Configure WebSphere Portal on the integration system. This includes global portal settings and other configurations, such as property file entries.
- Deliver documentation on global settings and configurations, for example, a description file, into the VCS.
- Create the WebSphere Portal solution configuration including content hierarchy, page layouts, and portlet configurations, on the development system.
- Export release configuration (export-release-only) from content master using the XML configuration interface . Export the content topology as well as all portlets.
- Deliver the portal solution release configuration into the VCS.

Build and Deployment Process Example

Release Manager

- Place the version artifacts and configurations as a portal solution release in the VCS.
- Check out all artifacts for the portal solution release from the VCS.
- Install the artifacts to the integration system.
- Check out all configurations for the portal solution release from the VCS.
- Import configurations into the integration system using the XML configuration interface.
- **Test Team:** Perform integration tests of the release on the integration system.
- Notify the *portal operator* about the availability of the initial portal solution release.

Build and Deployment Process Example

Portal Operator

- Check out all artifacts, including administrative portlets for initial portal solution release from VCS.
- Install artifacts to production system. (for portlet installation use the XML configuration interface export of all portlets).
- Check out the description of global settings and configurations for the initial portal solution release from the VCS.
- Apply the portal global settings and configurations to the production system.
- Check out all configurations for the initial portal solution release from the VCS.
- Import the configurations into the production system using the XML configuration interface. Preserve the object IDs.
- Backup the production system (data base and file system).
- **Test Team:** Perform regression tests of the release on the production system.
- Use the portal solution and customize the portal configuration, creating user data.

Components of a Portal solution

Configuration stored in database

- general config settings (DB, WebSphere Portal, configuration of custom applications)
- Portlet application configurations / settings
- Portlet configurations
- Portlet data / preferences
- ACL (Roles, ActionSets)
- Themes / Skins
- content tree / navigation / pages
- layouts
- URL Mappings
- Credential data

Configuration stored in filesystem

- general config settings / properties (Edge server, HTTP server, WAS, WebSphere Portal)

Portal artifacts

- Portlet applications (WAR files)
- Portal customization (Java classes)
 - ✓ Portlet services
 - ✓ Credential Vault Adapters
 - ✓ WMM Adapters / Custom User Registry
 - ✓ Custom portal extensions
- Themes / Skins (JSP files and stylesheets)

→ **Each component has to be considered during a Staging process**

Best Practices - Portal Navigation

- **Levels Of Navigation**
- **Avoid Using More Than 3 or 4 Levels**
 - Create A Clear and Distinct Hierarchy Between The Levels
 - Create A Clear and Distinct Relationship Between The Levels
- **Recommendation to use navigation portlets**
 - L1 based on theme navigation, L2 and more by navigation portlets
- **Location And Orientation**
 - Stick With Industry Standards
 - Primary Navigation Top Horizontal or Left Vertical
- **Style**
 - Avoid Using Drop Down Menus for Site Navigation
 - Use Tabs For Horizontal Navigations Only
 - Use Tree-like Mechanisms for Vertical Navigations

Best Practices – Page Design

- **Amount of Content and Complexity of Layout**

- While It's Impossible To Limit The Amount Of Portlets Which May Be Added to the Page,
- Be Realistic In The Amount Of Portlets Users Have Access To
- Keep The Layout Simple!
- Avoid Placing Portlets in Rows, Use Columns Only
- Keep lightweight portlets on pages everybody accesses. Heavier portlets go on pages users select.

- **Skin Options**

- Only Associate Skins with Themes They Are Designed to Work With

- **What Should Be Editable and What Should Be Fixed?**

- Use Container and Portlet Locks To Restrict Users from Changing Areas of the Page You Wish To Preserve or Control

Best Practices – XMLaccess Visualisation

Vlastnosti stránek a portletů - Page and portlets attributes

[zpět / back](#)

My Portal / Můj portál

cs: Homepage	A	wpsadmin, wpsadmins, tisadmin, tisadmin, wpsadmins, tisadmin, wpsadmins, wpsadmin	cs: TESCO Banner zip 1 (RSD)	A	wpsadmin, wpsadmins, tisadmin, tisadmin, wpsad
en: Homepage	SA		en: TESCO Banner zip 1 (RSD)	SA	
Theme: TescoCorporate	D		Active	D	
Active	M			M	
	E			E	tiscoadmin, tissaadmin, tissoeditor
	PU			PU	
	U	all authenticated portal users		U	all authenticated portal users, all authenticated p

cs: O společnosti	A	wpsadmin, wpsadmins, tisadmin, tisadmin, wpsadmins, tisadmin, wpsadmins, wpsadmin	cs: Filozofie Tesca (RS0003)	A	wpsadmin, wpsadmins, tisadmin
en: About Tesco	SA		en: Tesco Philosophy (RS0003)	SA	
Theme: Default	D		Active	D	
Active	M			M	
	E				
	PU				
	U	all authenticated portal users			

Aplikace a jejich klony - Applications and their clones

[zpět / back](#)

HotInfo.war

Česky	English
Hot Info - MASTER	Hot Info - MASTER
Hot Info - uživatelé z obchodů	Hot Info - Store users
Hot Info - pro editory	Hot Info - for editors

Document Register.war

Česky	English
Document Register (MASTER)	Document Register (MASTER)
Titulek chybí!	Title is missing!
Prohlížeč dokumentů (RS0084)	Document Viewer (RS0084)

Performance Requirements

- **Identify Performance Requirements before you order the hardware**
 - Projected active users
 - Pages/Visit which consists of number of portlets processed plus number of logon pages times logons
 - Page rate required
 - Think time in minutes
 - Response time per page view in seconds
 - Processor utilization in percentage
 - Percent contingency factor
 - Response time calculations in percentile
 - Type of architecture (how many tiers)

Tivoli Performance Viewer

- **Java client used to monitor application servers**
 - Shipped and installed with WebSphere Application Server
- **View data in real time**
 - Chart
 - Tabular
- **Minimizes the performance impact on servers by:**
 - Polls servers at intervals set by administrator
 - All data manipulation is performed at the client
 - Client may run on a separate machine
- **May monitor single WebSphere Application Server or servers in a Network Manager deployment**

Tivoli Performance Viewer Console Layout

Tivoli Performance Viewer
_ □ ×

File Logging Setting Help

⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹

Data Collection

- ⊕ Performance Advisor
- ⊖ Viewer
 - ⊖ student1
 - ⊖ WebSphere_Portal
 - ⊕ Alarm Manager
 - ⊕ Enterprise Beans
 - ⊕ Dynamic Caching
 - ⊕ JDBC Connection Pools
 - ⊕ JVM Runtime
 - ⊕ Object Pool
 - ⊕ ORB
 - ⊕ Scheduler Service
 - ⊕ Servlet Session Manager
 - ⊕ Thread Pools
 - Transaction Manager
 - ⊕ Web Applications
 - ⊕ Web services
 - ⊕ Workload Management
 - ⊕ nodeagent
 - ⊕ server1

Resource Selection

View Data
View Chart

Counter View

Legend

- Free Memory (KB)
- Used Memory (KB)
- ▲ Total Memory (KB)
- ◆ Total Memory (KB) (Avg)

Name	Description	Value	Select	Scale
Total Memory (KB)	Total memory (KBytes) in JVM runtime	330889.0 (...)	<input checked="" type="checkbox"/>	1.0E-4
Free Memory (KB)	Free memory (KBytes) in JVM runtime	167246	<input checked="" type="checkbox"/>	1.0E-4
Used Memory (KB)	Used memory (KBytes) in JVM runtime	163643	<input checked="" type="checkbox"/>	1.0E-4
JVM Up Time (seconds)	The amount of time (in seconds) the JVM has b...	5452	<input type="checkbox"/>	0.01

Counter Selection

Refresh rate: 10 sec Buffer size: 40 View Data As: Raw Value Logging: OFF

Performance Tuning

- **Use the WebSphere Portal Tuning Guide**
- **Allow one week per integration point for performance tuning**
- Monitor *everything*
 - Backends (content servers, databases, security, and so forth)
 - Mid-tier (WP)
 - Front end (HTTP servers, caching proxies, networks, test clients)
- **Focus on database capacity, throughput, response time, and sessions created**
- **See InfoCenter for Tuning and Performance Improvements**
 - Caching, Parallel rendering, Lightweight Themes and Skins, High Performance Skins, Nested Groups ...

More Information

Located in the WebSphere Portal Documentation Library:

[http://www.ibm.com/developerworks/websphere/zones/portal/pr
oddoc.html](http://www.ibm.com/developerworks/websphere/zones/portal/pr
oddoc.html)

- WebSphere Portal Performance Tuning Guide 5.1.0.1
- WebSphere Portal Performance Tuning Guide 5.1
- WebSphere Portal Performance Tuning Guide 5.0.x

Best Practices - Security

- Portal is very dependent on LDAP. Make sure you have a good LDAP implementation
- Do not externalize your authorizations for portal resources
- WebSphere Portal should be behind a firewall
- Change the default passwords and establish a password expiration policy.
- Remove unused portlets and components (self-registration, samples portlets)
- Carefully plan use of encryption
- Do not run WebSphere Portal as Root
- Give “others” only read access to Websphere Portal installation directory, WebSphere Server, and the Web Server
- Also limit access to the logs
- IDS and AV-Proxy is recommended

Best Practices – Access Control List

Permission assigning for Users

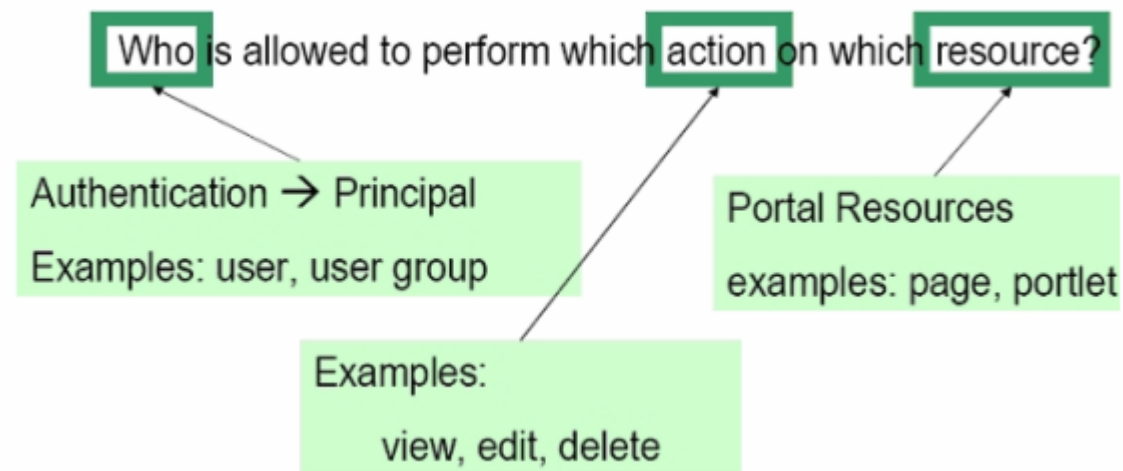
User Group: User@Pages-Content Root-My Portal-Page

User Group: User@Portlets-Concrete Portlets on Page

Permission assigning for Department Editor

User Group: Editor@Pages-Content Root-My Portal-Page

User Group: Administrator@Portlets-Concrete Portlets on Page



Best Practices – Access Control List

Permission assigning for Department Administrators

User Group: Security Administrators+Editor@Pages-Content Root-My Portal-Page (can not delete page)

User Group: Administrator@Portlets-Concrete Portlets on Page

User Group: Security Administrator@User Groups-Department Editors Group

User Group: Security Administrator@User Groups-Department Users Group

(User Group: Security Administrator@User Groups-TISAuthUsers)

User Group: Editor@Virtual Resources-Portlet Applications

(Department Administrator can copy portlet for using them, and modify parameters, but Release manager has to be asked for activating and assigning Department Administrator to Administrator Role for copied portlet)

Best Practices – Access Control List

Permission assigning for Department Administrators

User Group: User@Pages-Content Root-Page Customizer (Edit Page link)

User Group: User@Pages-Content Root-Page Properties (New Page link)

User Group: User@Pages-Content Root-Administration-Portal User Interface-
Manage Pages

User Group: User@Pages-Content Root-Administration-Portlets-Manage
Applications

User Group: User@Pages-Content Root-Administration-Portlets-Manage Portlets

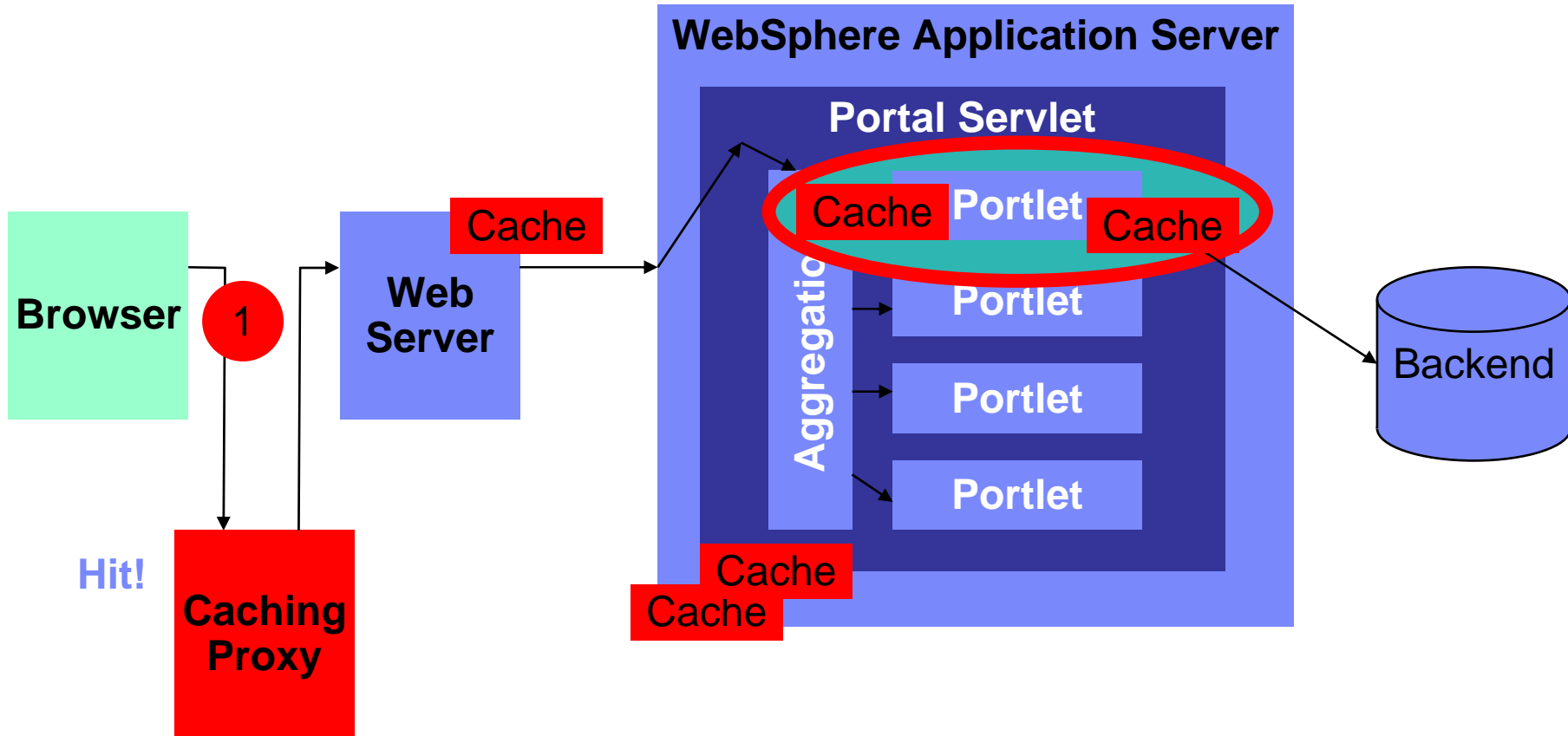
User Group: User@Pages-Content Root-Administration-Access-Resource
Permissions

Assigning into portal roles is recommended only for user groups not for individual users.

Best Practices - Caching

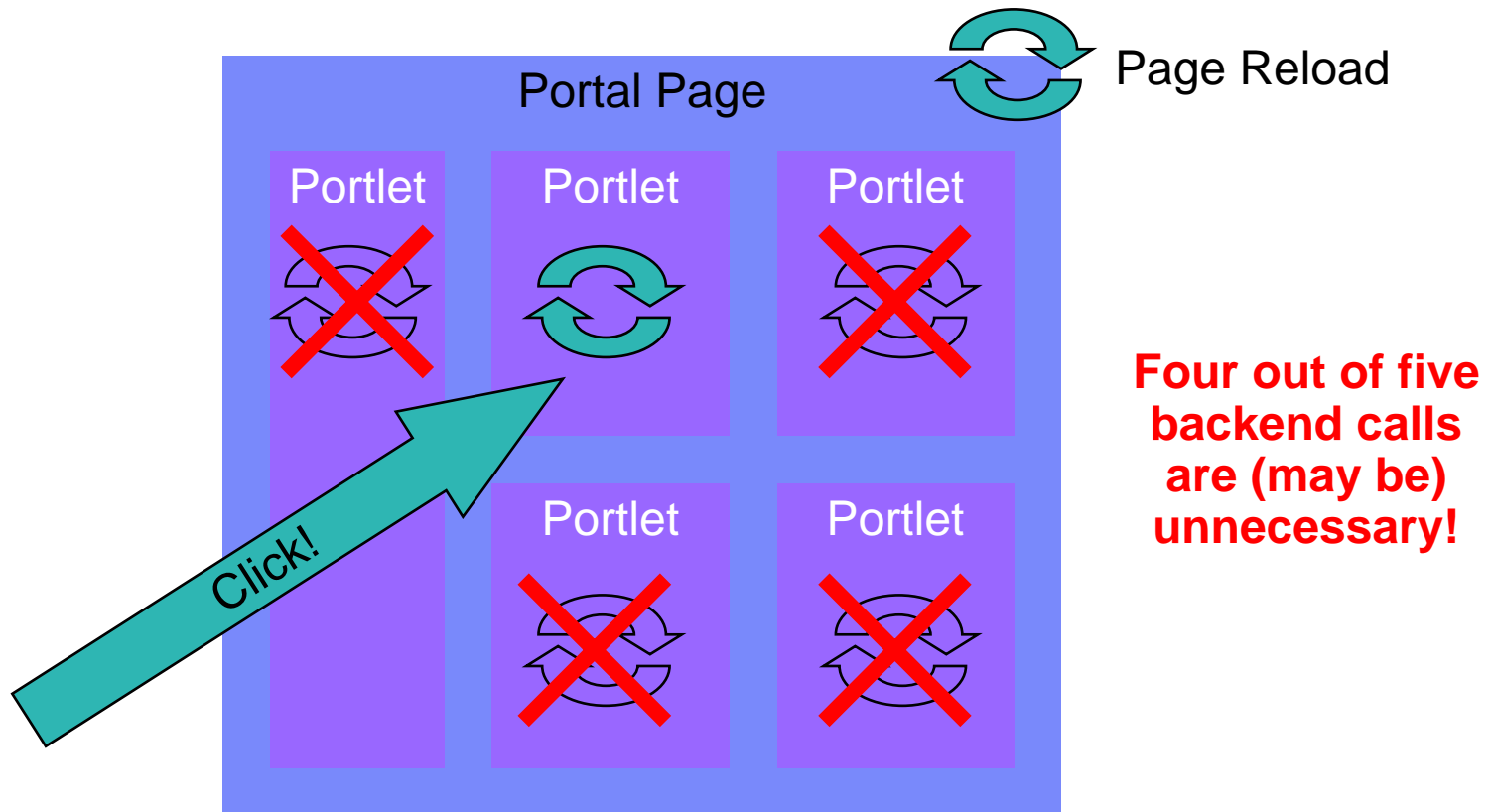
- Plan time to do caching. This will take about a month of time. Use normal portlet caching, learn dynacache and do advanced caching too for some portlets, finally learn the Command Cache and use it to store data for access in multiple portlets (use this instead of HttpSession)
- Proper or improper use of caching can make or break performance
- The more dynamic your portals, the less usefulness of caching
- Need to tune both the back-end and the front-end cache
- Enable Portlet Display cache in the portlet.xml file
- Use the command cache for portlet service
- Tune the cache expiration
- Tune the number and size of objects in the dynacache. Ensure that it doesn't overflow to disk

A Cache Can Help to Save Remote Calls



“Unnecessary” Reload of Portlets

Imagine a page with five Portlets:



Related Techniques

- **Portlet Fragment Caching**
 - Portal-internal facility
 - Configured in portlet.xml
- **Infrastructure level**
 - Edge of network caching, WAS-internal caches, and so forth
- **Command Cache**
 - Declarative cache
 - Follows Command Pattern
 - If applicable, almost no changes to application code

More Information

- **„Portlet filters“, WP 5.0.2 InfoCenter,**
<http://publib.boulder.ibm.com/pvc/wp/502/ent/en/InfoCenter/wps/adplftft.html>
- **„Using the DistributedMap interface for the dynamic cache“, WAS InfoCenter,**
http://publib.boulder.ibm.com/infocenter/wasinfo/topic/com.ibm.wasee.doc/info/ee/ae/tprf_distmap.html
- **Hines, Bill: “Static and dynamic caching in WebSphere Application Server V5”,** http://www-128.ibm.com/developerworks/websphere/techjournal/0405_hines/0405_hines.html
- **Bernal, Joey: “Using the command cache to improve portal application performance”,** http://www-128.ibm.com/developerworks/websphere/techjournal/0408_bernal/0408_bernal.html

Best Practices - Portlet Development (1 of 2)

- **Keep portlet designs simple**
- **Use design patterns**
 - Help to reduce conceptual complexities in code design
 - MVCA is fundamental to portlet development
 - Control functionality is the responsibility of the portlet
 - Model layer should be provided through JavaBeans or EJBs (via portlet services)
 - View should be handled through JSPs
 - Should be kept simple and render quickly
 - Break large view states up and handle transitions with the state pattern
 - Use custom tag libraries to reduce scripting
 - Access/persistence layer functionality should be provided through portlet services
- **Document your code**
 - Use comments and meaningful naming conventions
- **Use the portlet log to build debugging infrastructure into your code**

Portlet Development (2 of 2)

- **Understand PortletData behavior**
 - Page 'administrators' affect the default, common data configuration
 - Privileged users affect only their personal data
 - Once a privileged user customizes the data, they will not receive any future changes to the default configuration
- **Use the ContentAccessService to retrieve external content**
- **Ensure that all portlet applications and concrete portlet applications have a unique UID**
 - If there are UID conflicts, the application will not be installed
 - When updating an application, the UIDs must match
- **Conduct code reviews and inspect for:**
 - Resource usage (leaks, can local variables be reused)
 - Session and data management scoping/size
 - Namespace and URL encoding



Data Management

- **Avoid using mutable instance variables**
 - Data stored in instance variables are accessible across all requests and all users
- **Pass data to JSPs as a bean in the request object for rendering**
 - When the request is complete, the data falls out of scope
 - The JSP can use the bean data as properties through JSP syntax
- **Scope configuration settings and user data appropriately**
 - The configuration objects should only be used to define portlet behavioral settings and meta information
 - Use PortletSettings to store user-independent configuration data
 - Use PortletData to store user-dependent configuration data
 - Do not use PortletData to store application data
- **Consider caching PortletSettings and PortletData values**
 - If the data is complicated and requires complex parsing before use, it is a good idea to cache the data for quick retrieval later

Session Management

- **Limit the amount of data stored in the PortletSession**
 - There is considerable overhead in managing the session, both in CPU cycles as well as heap consumption
- **Do not use portlet sessions in portlets that will be accessible by anonymous users**
 - PortletSessions are bound to a user context
 - A portlet session can be requested by a portlet for an anonymous user, but it is a temporary session which only lasts for the duration of the request
- **Always request an existing portlet session**
 - Retrieve using `portletRequest.getPortletSession(false)`
- **Prevent temporary sessions from being generated in the JSP**
 - Use the JSP page directive `<%@ page session="false" %>` to prevent temporary sessions from being created by the JSP compiler if none already exist

JSP Coding (1 of 2)

- **JSP content should be fragment only**
 - Portlets are responsible for rendering only a fragment of the page
 - Must be well-formed content that can be rendered (for HTML) within a `<td> . . . </td>` structure
- **Design content to fit on a page with other portlet content**
 - Minimize the need for the user to scroll, particularly horizontally
 - Large images or significant textual content should only be rendered when the portlet window is in maximized or solo state
- **Use portal theme style classes, not portlet level style attributes**
 - Style classes are defined in the theme's cascading style sheet
 - Avoid defining stylesheets within the portlet
- **Minimize dependencies on JavaScript**
 - Implementations and behavior vary widely between browsers
 - The greater the use of JavaScript, the greater the difficulty in maintenance and porting to non-HTML markup
 - It is difficult to namespace encode resources and build portal URLs using JavaScript

JSP Coding (2 of 2)

- **Use tag libraries whenever possible**
 - Allows common view functions to be easily reused
 - Keeps the JSPs conceptually clean and makes them more like normal HTML pages
- **Use IFRAMEs with **extreme** caution!**
 - There are potential issues to consider when using an IFRAME
 - Content is filled based on a URL
 - The URL is retrieved by the browser (not by the Portal), and thus the target server URL must be directly accessible by the browser
 - Initial URL is provided by Portal causing the IFRAME to reset to the original URL when the page is re-rendered
 - CredentialVault, C2A, PortletMessaging and other Portal services can not be used
 - Not all browser levels support IFRAMEs
 - If the content is larger than the IFRAME region, then horizontal and vertical scrolling need to be enabled
 - Be careful of content which itself contains scrolling regions as it can be difficult to manipulate all embedded content, causing usability problem

Plan for Accessibility

- **Pages should be fully accessible**
- **JSPs need to support keyboard-only controls and other assistive technologies to enable users with disabilities to use your portal**
 - Use the ALT attribute with images to define descriptive text of the image content
 - Use tags to associate labels with form input controls, so that page readers will be able to associate prompts with inputs
 - Do not use color alone to denote state or information
 - Using red to emphasize text doesn't help those who are color blind
 - Use bold or italics, or use color in conjunction with graphics
 - Do not use voice or other sounds to convey information
- **Be careful with JavaScript!**
 - Not all browsers fully support JavaScript
 - May not be able to "click" (key press may be necessary)
 - Be careful with pop-up dialogs

Internationalization

- **Understand, and use, the built-in framework for supporting NLS**
- **Portlet strings should be fetched from resource bundles**
 - Bundles should be organized by language under the portlet's WEB-INF/classes directory
 - JSPs that use resource bundles should be language-independent
 - Render strings in the JSP using the tags provided by the Java Standard Tag Library (JSTL)
- **For complex layouts, use separate language specific JSPs**
 - Help files are almost always language specific JSPs
- **Be sensitive to cultural-specific formatting**
 - Countries represent certain information differently
 - Dates may be MMDDYYYY or YYYYMMDD
 - Decimal points may be a period or a comma

Security Considerations

- **Use the Credential Vault to store sensitive data**
 - It is designed to securely store “secrets” necessary for accessing integrated applications and avoiding multiple login prompts

- **Be aware of the data being passed to the client**
 - HTTP connections may not be secured
 - Certain browsers, especially mobile devices, do not secure transmitted data effectively

Packaging Portlet Applications

- **Isolate common functions and make them externally available to portlets**
 - Build a JAR file with the classes that encapsulate the common behavior and deploy using one of the following approaches
 - Place the JAR file in a location that is in each portlet's classpath, such as the AppServer/lib/app directory
 - Build a portlet service and access the functions by retrieving the portlet service

- **Combine portlets with similar functions into a single portlet**
 - Determine specific behavior through configuration settings
 - The common portlet code is specified as part of the abstract portlet application's abstract portlet definition
 - The concrete portlet definitions will have unique configuration parameters (PortletSettings) that are then used to customize the behavior of the common portlet code

Portlet Testing

- **Establish test cases for all portlets and their various behaviors**
- **Start testing early, and test often**
 - Don't wait for problems to appear in your production environment
 - Retest portlets continually
- **Always test portlets for performance**
 - Make sure to test portlet independently and with other portlets
 - Test to the breaking point

Portlet Performance is Critical

- **Identify poor performing portlets and correct**
- **General things to consider:**
 - Break the portlet's render screens into smaller display components
 - Parallel portlet rendering for portlets that access remote locations
 - Caching
 - Portlet content caching
 - WebSphere dynacache
 - CommandCache to store data for multiple portlets
 - Data caching in portlet services
- **Remember that Portal is a Framework for building portlet applications**
 - It is not an out-of-the-box simple solution!
- **Understand that Portal will eventually front - end everything in your enterprise**
 - this means you will be “blamed” for all IT issues!

Portlet Performance Tuning (1 of 2)

- **Do not spawn threads**

- Portlets are multi-threaded to begin with - spawning child threads can create problems with synchronization
 - If necessary, they should be used only briefly - nothing that outlasts a request
 - Do not use threads to access J2EE resources

- **Limit temporary (local) objects**

- Typically utilitarian and simple
 - Still takes CPU cycles to create/destroy and occupy space on the heap which must be maintained by the garbage collector.
- Reuse temporary variables as much as possible
- Declare temporary variables outside loops
- Use StringBuffer, which are optimized for parsing and string assembly, instead of Strings
- Declare collection classes (Vectors, Arrays) with an initial size that is the average maximum size, to prevent growth and reallocation of space on the heap

Portlet Performance Tuning (2 of 2)

- **Avoid synchronized methods**

- Causes a bottleneck in your portlet, slowing down the portal system and potentially causing deadlock

- **Avoid long-running loops**

- Long running loops consume a large number of CPU cycles and cause the portal page to wait for this one portlet to finish
- Attempt to accomplish through other means, such as breaking the large loop up into several shorter ones

- **Use JSPs instead of XML/XSLT**

- JSPs are **much** more efficient than XML/XSLT
- XSL processing causes hundreds of transient String objects to be created and destroyed
- If XML/XSLT is a requirement, make use of caching to reduce parsing and XSLT processing
 - Consider using a dedicated transformation server

Finally . . .

Know where to get help

- Redbooks:

www.redbooks.ibm.com

- WebSphere Portal developer zone

<http://www.ibm.com/developerworks/websphere/zones/portal/>

- WebSphere Portal Documentation Library

<http://www.ibm.com/developerworks/websphere/zones/portal/proddoc.html>

- WebSphere Portal Support Page

<http://www-306.ibm.com/software/genservers/portal/support/>

- Newsgroup:

<news://news.software.ibm.com/ibm.software.websphere.portal-server>

Thanks a lot for your attention

Dusan Smolej / IT Architect

IBM Global Services Czech Republic

The Park 2294/4, 148 00 Praha 4 Chodov

Phone : +420 272 131 768, Mobil: +420 737 264 279

Email : dusan_smolej@cz.ibm.com