Web development for mere portals

Benjamin Paillerau

Solution Linux

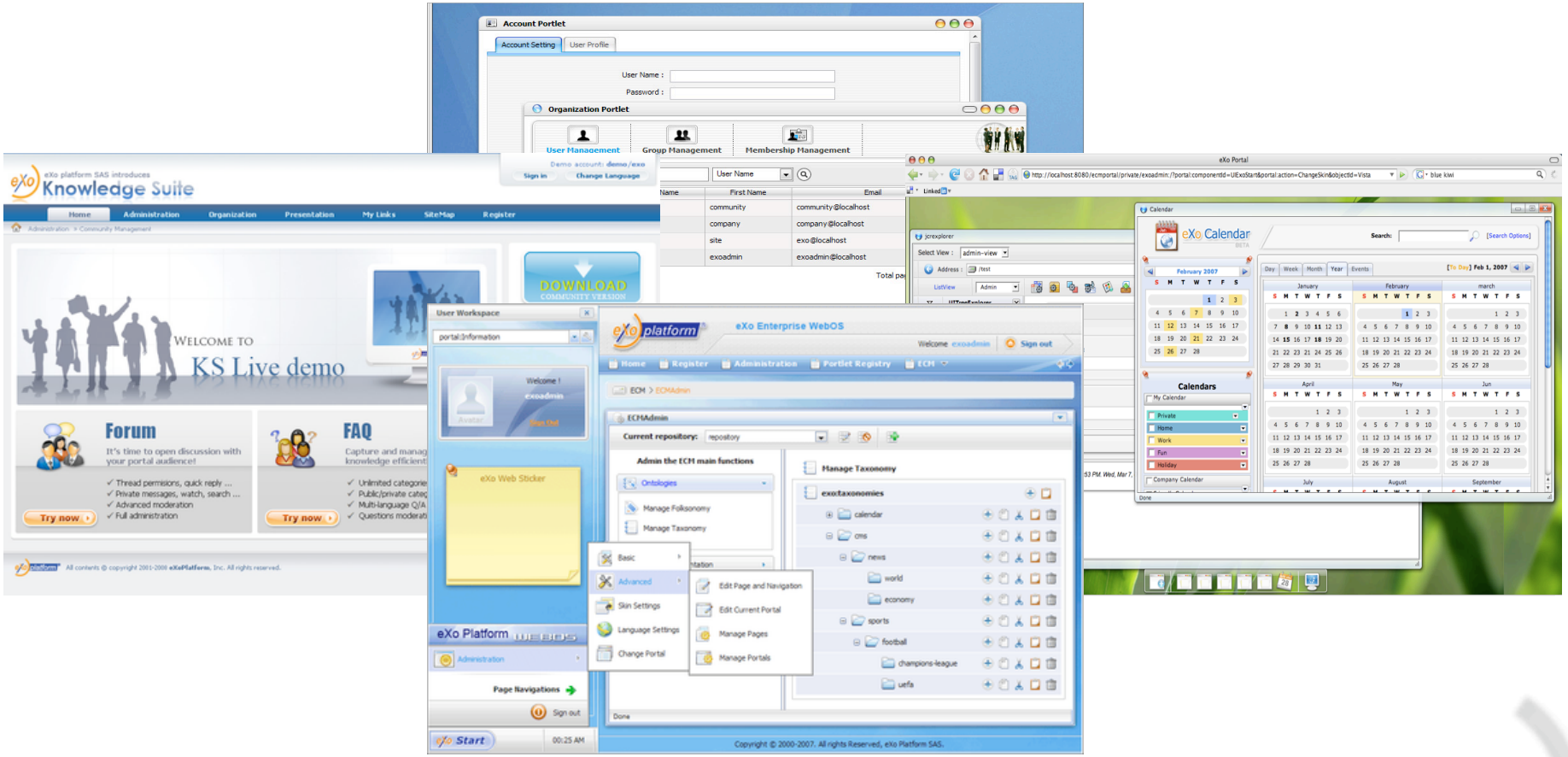
eXo platform

OW2 Annual Conference

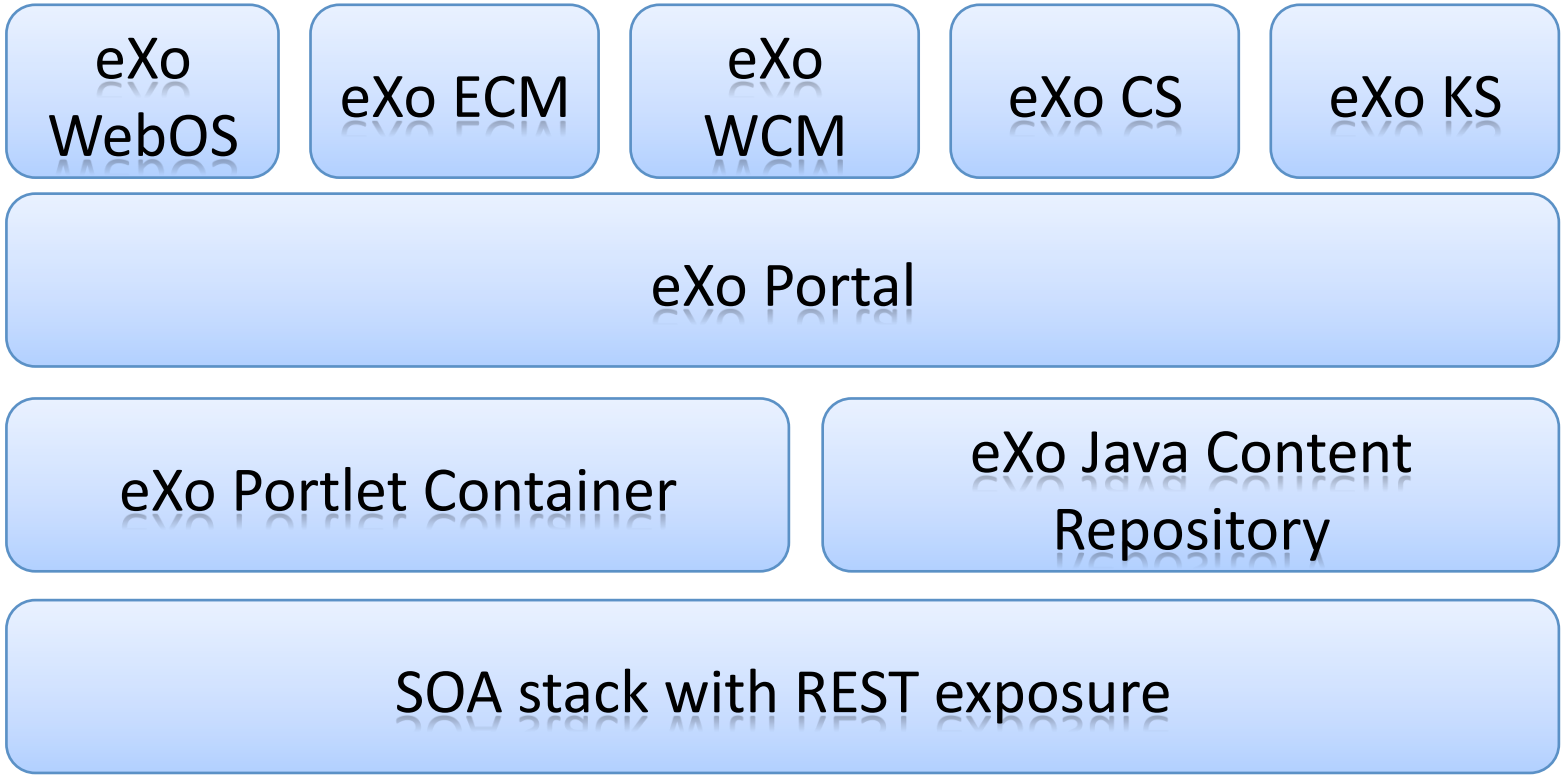
About me

- Benjamin Paillereau
  - eXo WCM Product Manager

Product Offering



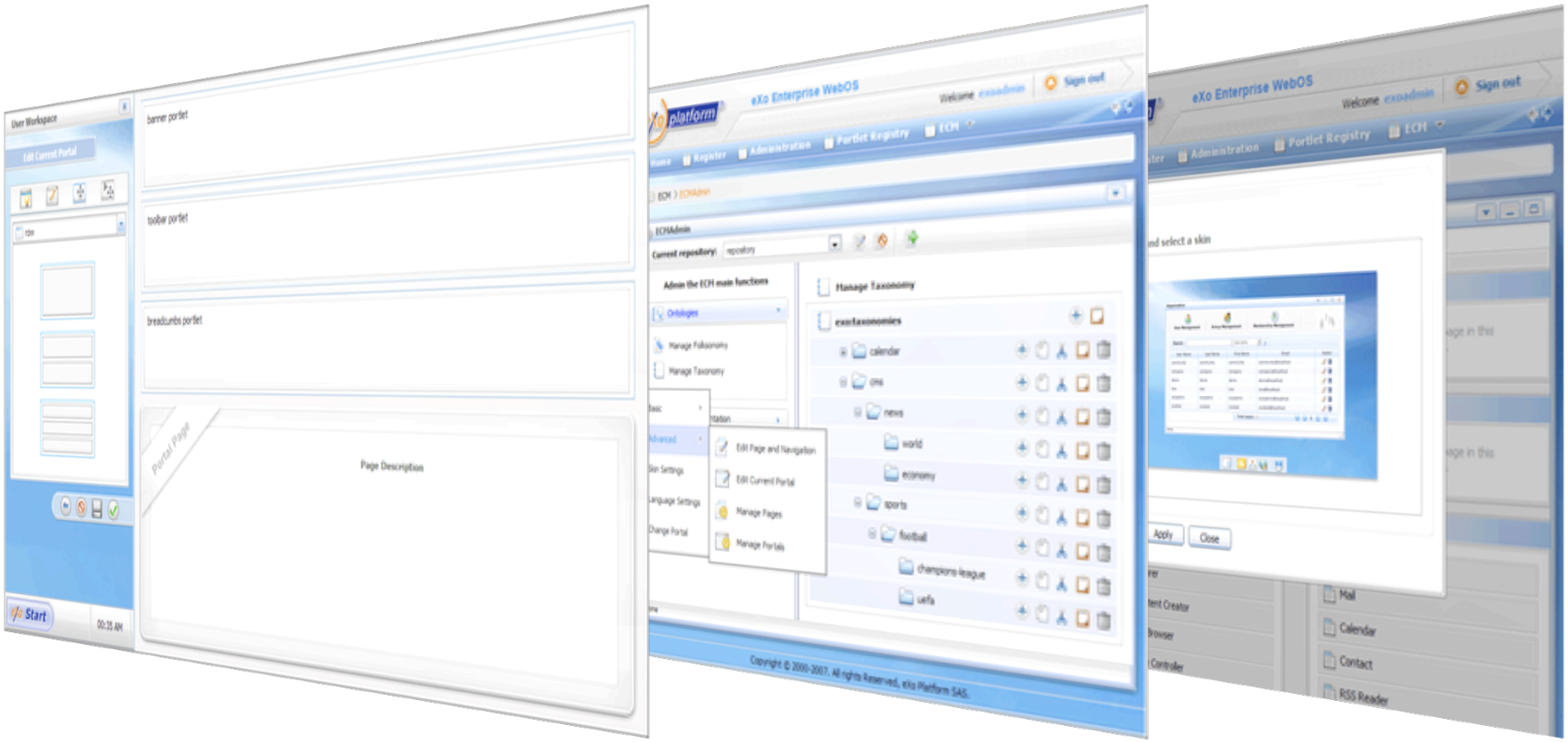
The eXo Platform Suite



eXo Portal 2.5

- Facilitate Enterprise Data and Applications Access
  - Personalization / Profiling
  - Flexible Layouts & Themes
  - Leverage AJAX to improve user experience
- Standard Based
  - Leverage JavaEE Platform
  - Java Portlets API (JSR-168 & JSR-286)
  - Remote Portlets (WSRP 1&2)
  - Java Content Repository (JSR-170)
  - Google Gadgets
- Extensible
  - REST Services
  - Portlet Bridges to facilitate IS integration

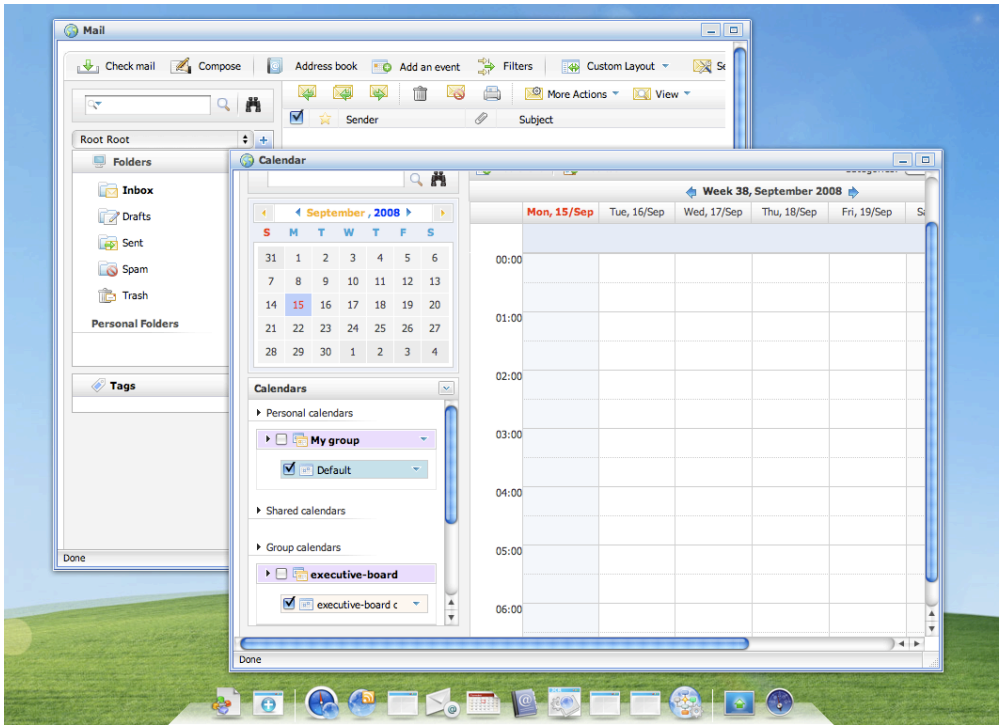
eXo Portal 2.5

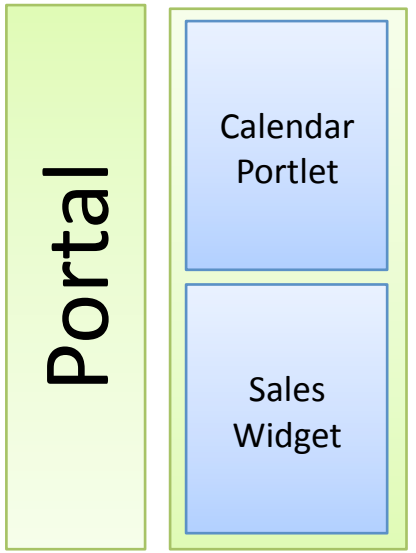


Agenda

- Enterprise portal introduction
- Portlets and widgets explained
- Web frameworks for portal

Enterprise portals aggregate your applications, contents and services





Widgets

- Widgets are browser components
  - An emdeddable chunck of Javascript using web standards
  - Many implementations, many names: gadget, badge, module, webjit, capsule, snippet, …

Portlets

- Portlets are server components
  - Similar to servlets
  - Designed for aggregation and personalization
  - Portlet 2.0 standard
  - Designed to integrate with the Java EE ecosystem but not part of it

The MVC paradigm

- Portlets interactions are divided into phases reflecting the MVC paradigm
- Three phases
  - **render** phase retrieves markup to update the view
  - **action** phase updates state and acts as controller
  - **event** (Portlet 2.0+) phase coordinates portlets, it is an extension of the action phase to other portlets on the same page

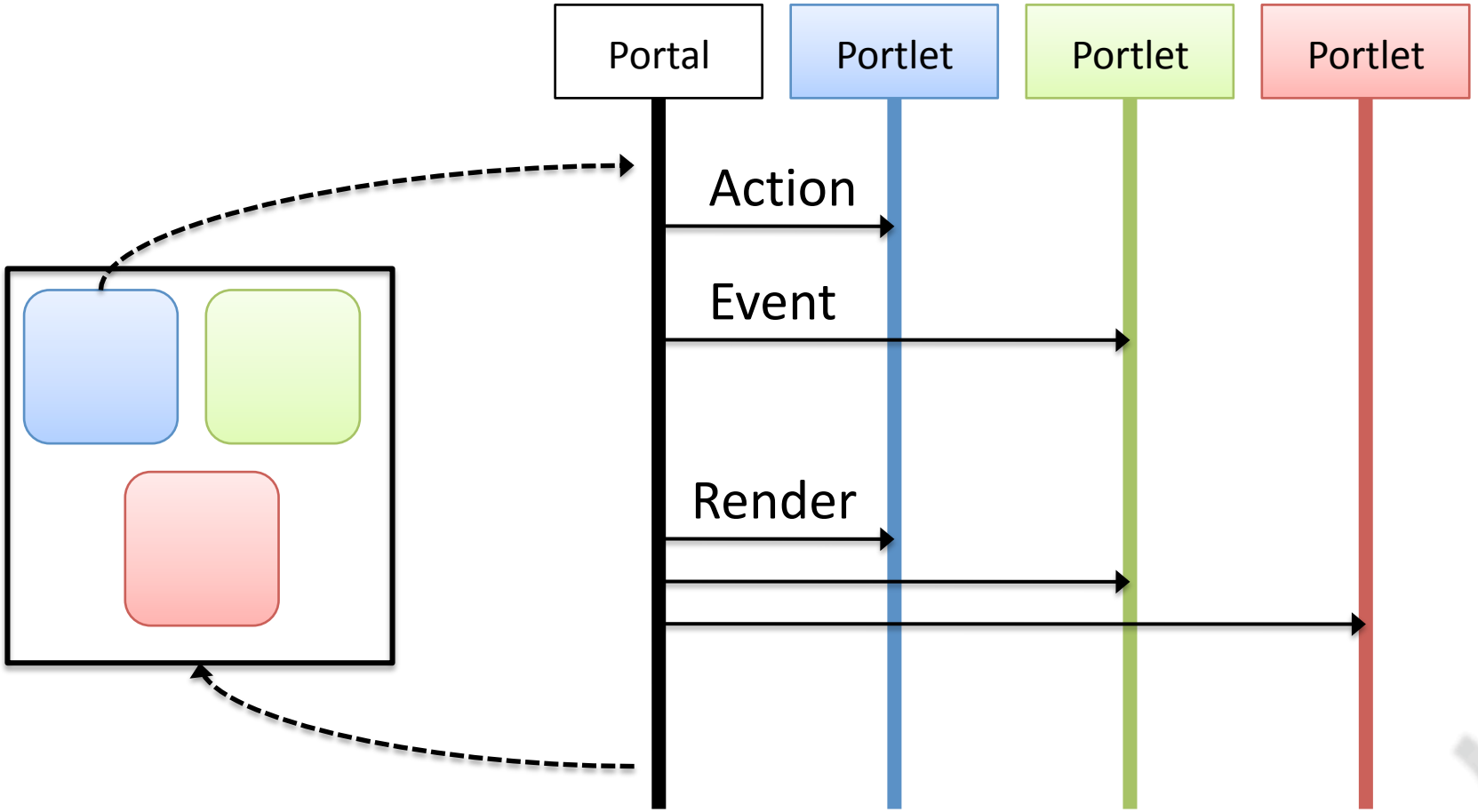
Portal http GET

- Renders a portal page
  - The render phase reads the state and create markup

Portal http POST

- Triggered by a portlet to trigger the execution of its action phase
  - The action updates the state
    - Database
    - Session
    - Preferences
    - Parameter
  - The optional event phase signals other portlet that something happened
  - Finally page rendering occurs again

Phase sequence



Portlet features you won't find in a servlet

- Decorations
  - Portlet mode: view, edit, help, admin
  - Window state: minimized, normal, maximized
- Personalization
- Access to the user profile
- Support for internationalization
- Categorization

Portlet / Servlet similarities

`PortletRequest`	`ServletRequest`
`PortletResponse`	`ServletResponse`
`PortletSession`	`ServletSession`
`PortletContext`	`ServletContext`
`PortletFilter`	`ServletFilter`
`PortletRequestDispatcher`	`RequestDispatcher`

Portlet application packaging

- Portlets are packaged in a war file and the portlet container discovers them
- The `portlet.xml` deployment descriptor configures and describes the portlets

Ajax portlets

- Since Portlet 2.0 there is API support for Ajax portlets

Portlets and Widgets compared

Portlet	Widget
Portlets use the javax.portlet interfaces and classes	Standard runtime per design (js/ dom/css) but each implementation has its own API
Requires a build (compile/package/ deploy)	No development process required
Supports Browser Back Button (BBB)	No support for BBB
Supports bookmarkability	No support for bookmarkability

Portlets and Widgets compared

Portlet	Widget
Request/response server interactions	Live user interface
State is shared between the portal and the portlet container	State is shared between the browser DOM and the portal No development process required
Use container services (Database, EJB, ESB)	Use Programmable Web, usually REST Web Services
Standardized coordination	Coordination is a feature of the widget API, usually not interoperrable

The web framework reality check

- Gives you productivity
- You should not develop an application with the Portlet API
  - low level
  - not productive

Web framework for portals

- The Portlet Bridge
  - An adapter for a web framework to the portlet container runtime
  - Works ideally with framework that do not expose the servlet container
  - Limited support for the full portlet API
- The Portlet Framework
  - The web framework approch to the portlet world
  - Leverage your knowledge of a framework
  - The full portlet API can be leveraged

Portlet Bridges

- Modern web frameworks have an abstraction (SPI) over the servlet API
  - The bridge implements the SPI by delegating to portlet container
- Legacy web frameworks relies on the servlet API
  - The bridge executes the target framework in its servlet container context

Modern Portlet Bridges

- The framework action is executed during the portlet action phase
- The framework rendering is executed during the portlet render phase
- The best example is JavaServer Faces
  - The JSR 301 specifies the semantics of the bridge

Legacy Portlet Bridges

- Provides a servlet container context
  - The bridge emulates the Servlet API

  or

  - The bridge performs a request dispatch to the web framework servlet
- Markup must be filtered for aggregation
- URLs must be rewritten to work with the portlet container and the portal

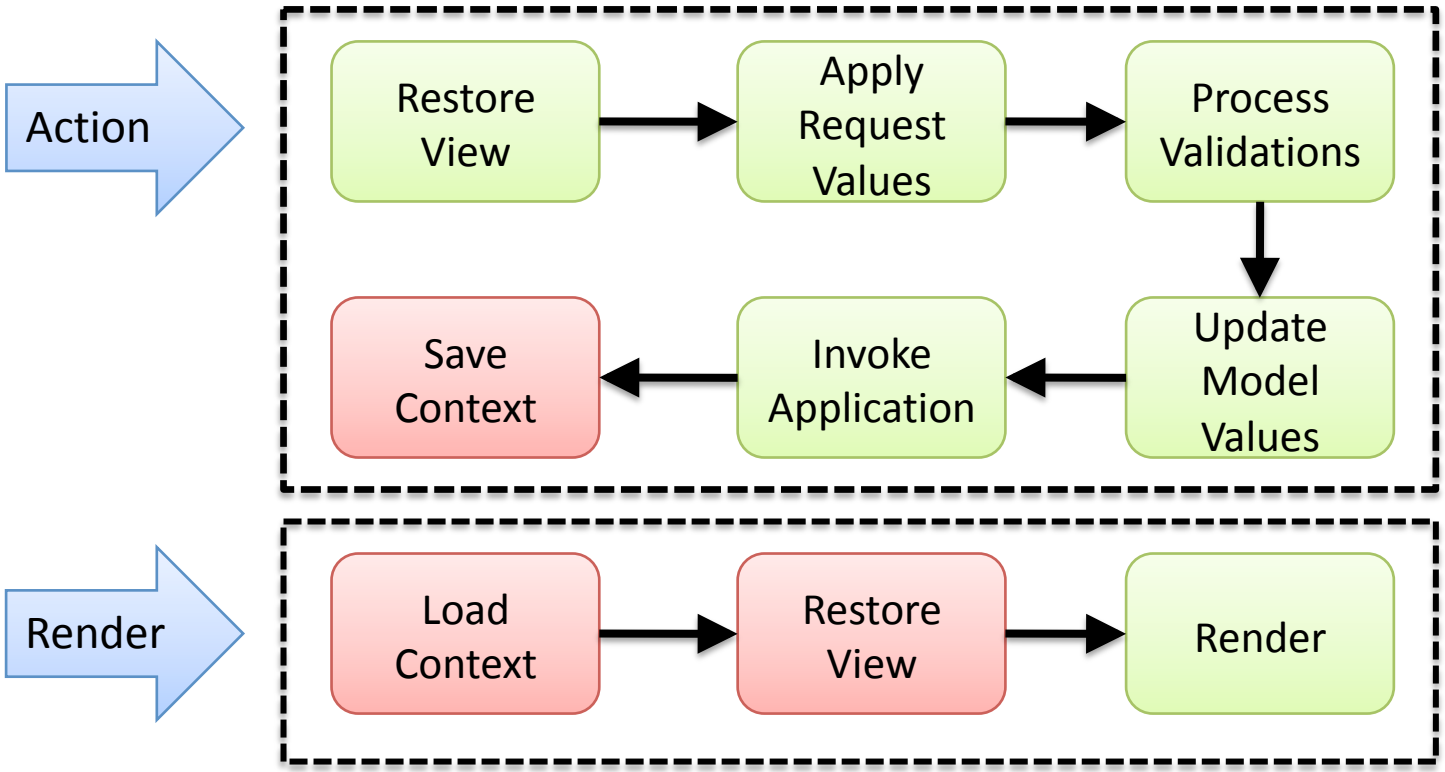
JSF Bridge

- Historically there were three opensource bridges
  - No consistent interoperability
- The Portlet Bridge Specification for JavaServer Faces (JSR 301)
  - The main goal is to define how a JSF application executes in a portlet environment
  - Additionally it specifies how advanced portlet features can be used

JSF Bridge Specification

- Status
  - Splitted into two specifications
    - Portlet 1.0 Bridge Specification for JavaServer Faces 1.2
    - Portlet 2.0 Bridge Specification for JavaServer Faces 1.2
- At least three opensource implementations
  - Apache MyFaces Portlet (RI)
  - JBoss PortletBridge
  - OpenPortal JSF PortletBridge

JSF Bridge Lifecycle



JSF Bridge Features

- Clean room specification
- Maintain action request attributes
- Handle identifier namespacing
- Handle redirects during the render phase
- Supports portlet mode changes
  - Manage a view per mode
  - Navigation between modes
    - `/edit.jspx?javax.portlet.faces.PortletMode=edit`
    - `#{sessionScope['javax.portlet.faces.viewIdHistory.view']}`

JSF Bridge known issues

- The bridge extends JSF and there can be conflicts or ordering issues with other JSF extensions

Struts 1.x Bridge

- A legacy bridge that supports Struts 1.x applications
- Limitations
  - Applications must be configured properly and not make abuse of Struts
  - Struts taglib must be used

Struts 1.x Bridge processing

- To properly execute the Struts application must be configured to separate action and view processing
- Model 2 paradigm is adapted to the portlet two phase protocol
  - The action phase executes the Struts action and freeze the processing just before the view is executed
    - ActionForm
    - ActionMessages
    - ActionErrors
  - The render phase resumes the Struts sequence and continues the processing

Struts 1.x Bridge render phase

- The bridge restores the render context and proceeds to the inclusion of the JSP page
  - The current page is set as a portlet parameter and translated into the request path in the emulated `HttpServletRequest`
  - The application must be configured to use a replacement TLD that generates correct urls

Apache Struts 2

- A Struts 2 plugin provides support for Portlet 1.0
- Struts 2 is a clean MVC implementation and does not require much configuration as opposed to Struts 1.x
- No specific support for portlet features

Spring Portlet MVC

- Leverages the Spring Framework for Portlets
  - Part of the core framework since Spring 2.0
  - Apply concepts of Spring Web MVC
  - File upload support
  - Annotation support since Spring 2.5
  - No support for Portlet 2.0

Spring Portlet MVC dispatching

- The portlets distinct phases are mapped to the controller and the view
- The `DispatcherPortlet` orchestrates the controller and the view
  - The action phase is mapped to a specific controller
  - The render phase emulates a servlet container and allow the reuse of the view technologies of Spring Web MVC (`ViewRendererServlet`)

Spring Portlet MVC controller

- The action phase delegates to a portlet controller integrated with the Spring Framework

```
public interface Controller {

   ModelAndView handleRenderRequest(
      RenderRequest request,
      RenderResponse response)

   void handleActionRequest(
      ActionRequest request,
      ActionResponse response);

}
```

Spring Portlet MVC annotation support

- `@Controller` annotates a controller class
- Method dispatching
  - `@RequestMapping` annotates a method and maps a render or action request
    - Supports portlet mode matching
    - Supports parameter matching
  - Rich population of method arguments
    - Portlet API objects
    - Spring Framework objects (`Model`, `View`, `ModelAndView`)
    - `@RequestParam` annotates request parameters
    - Etc…

Apache Wicket Bridge

- Wicket is a component based web framework
- The bridge is available since Wicket 1.3
- It requires the Portlet 2.0 API as it relies on the resource serving feature
- There is no support for advanced portlet features

Tapestry Bridge

- Tapestry has been rearchitected in version 4 to support portlets natively
- It provides support for portal features such as
  - Map window state and portlet mode to specific pages
  - Map user attributes to page properties

Grails Portlet

- Grails aims to bring “coding by convention” paradigm to Groovy
  - Heavily based on Spring and Hibernate
- The Grails Portlet is a plugin for Grails
  - The plugin is an extension of the Spring DispatcherPortlet
  - View rendering is delegated to a Groovy ServerPage (gsp)
  - Transparent Portlet reloading
- Warning
  - No release of the plugin at the moment
  - Need to obtain trunk from SVN

Grails Portlet action phase

- Portlet action phase is mapped on a Groovy closure resolved
  1. Using request parameter “action” value
  2. Using the value composed by “action” + the portlet mode
  3. Using the “doAction” value

Grails Portlet render phase

- Portlet render phase is mapped on a Groovy closure and a Groovy ServerPage
  1. Resolve a Groovy closure
     A. Using request parameter “action” value
     B. Using the value composed by “action” + the portlet mode
     C. Using the “doAction” value
  2. Rendering occurs in the GSP resolved
     A. “/” + portlet name + “/” + (“action” parameter)
     B. “/” + portlet name + “/render”

Struts 1.x Bridge configuration

- A specific configuration file provides additional information that the bridges uses to correctly execute the application

```
<config>
  <render-context>
    <attribute name="errors"/>
    <attribute name="message" keep="true"/>
  </render-context>
  <portlet-url-type>
    <action path="/shop/add"/>
    …
    <resource path="/images/"/>
  </portlet-url-type>
</config>
```

Useful resources

- http://jcp.org/en/jsr/detail?id=301
- http://jcp.org/en/jsr/detail?id=286
- http://static.springframework.org/spring/docs/2.0.x/reference/portlet.html
- http://portals.apache.org/bridges/multiproject/portals-bridges-struts/index.html
- http://struts.apache.org/2.x/docs/portlet-plugin.html
- http://cwiki.apache.org/WICKET/portal-howto.html
- http://grails.org/Portlets+Plugin
- http://tapestry.apache.org/tapestry4/tapestry-portlet/index.html
- http://myfaces.apache.org/portlet-bridge/index.html
- http://www.jboss.org/portletbridge/

Conclusion

Q&A