Using JEMS to Deliver a US Navy Enterprise Portal

Doug Schnelzer
Applied Engineering Management (AEM)

Agenda

- Navy Portal Objectives and Overview
- Extending JBoss Portal Security
- Portal Design and Layout
- Portlet Development Best Practices
- Integrating Business Process Management

Navy Portal Objectives and Overview

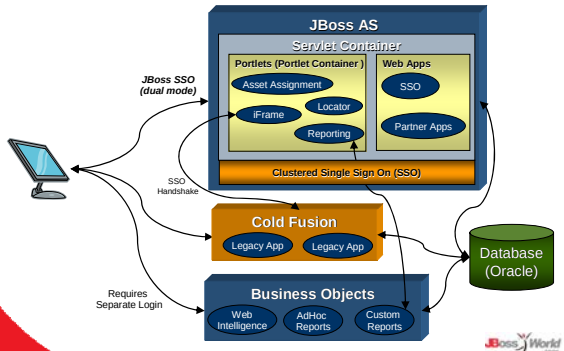
Navy Enterprise Portal Objectives

- Existing System
  - ✓ Manages personnel moves and tracking of worldwide assets
  - ✓ Mix of Cold Fusion and Servlet applications
  - ✓ Portal used globally by ~5000 users
  - ✓ Data Warehouse and enterprise reporting for business intelligence and official reporting
  - ✓ Used to continually revise long term budgets and planning
- Enterprise Portal Objectives
  - ✓ Establish architecture for consolidating and personalizing information and applications
  - ✓ Specify standards for new applications

Why We Selected JBoss Portal

- JBoss Portal passed "must have" requirements
  - ✓ Custom JAAS
  - ✓ Struts bridge
  - ✓ Ease of development/customization
  - ✓ Solid platform
- Good support in JBoss Portal User Forums
- Availability of Professional Support
- Complementary J2EE & JEMS components
  - ✓ JBoss Application Server
  - ✓ Hibernate
  - ✓ JBoss jBPM
  - ✓ JBoss Cache
  - ✓ J2EE → JMS, EJB3, Web Services,

Navy Portal System Architecture

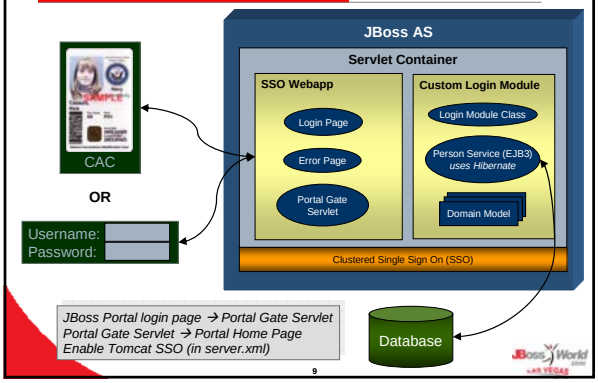


Extending JBoss Portal Security

Handling Complex Security

- JBoss Portal Security – "Out of the Box"
  - ✓ Uses a custom JAAS security domain
    - Registered in `jboss-portal.sar\conf\login-config.xml`
    - Login page `jboss-portal.sar\portal-server.war\login.jsp`
    - Error page `jboss-portal.sar\portal-server.war\error.jsp`
- What about unique security requirements?
  - ✓ Extend Single Sign On (SSO) for other web applications
  - ✓ PKI (X.509) certificates
  - ✓ Integration with LDAP, Windows Active Directory, etc.
- Our approach
  - ✓ Create a separate authentication web application
  - ✓ Develop custom login module (start with JBoss login module)
  - ✓ Modify JBoss Portal to use our JAAS Login Module (`jboss-portal.sar\portal-server.war\WEB-INF\jboss-web.xml`)

Our Authentication Approach



Ramifications of Our Solution

Pros

- Complete control over Authentication and Authorization
- Can specify Principle class
- Integrate LDAP, Windows AD, or custom RDBMS schema
- Architecture uses Servlet role based security
- Easily upgrade to new versions of JBoss Portal

Cons

- Loose JBoss Portal permission management
- Doesn't support out-of-the-box self registration
- Must replicate new users to JBP_USER table to support user preferences

Quick Sidetrack – AJAX Login

- The separate Login Web App can also be a portlet and use AJAX
  - ✓ Login portlet displays login form
  - ✓ Portlet submit button triggers AJAX (XMLHttpRequest) to request protected resource in Login web application
  - ✓ If login successful, AJAX function reloads portal page
  - ✓ If login not successful, display error without reloading page
- This approach allows users to login without needing to display a separate login form

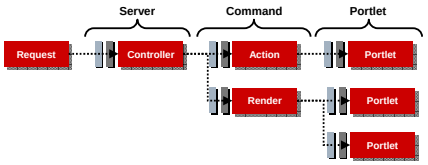
Extending Security to Non-Portlets

- Developed iFrame portlet to host web apps that can't quickly be made JSR-168
  - ✓ Quickly integrated 30+ web applications into Portal
  - ✓ Custom XML handshake establishes session with Cold Fusion server
  - ✓ Avoids complex URL rewriting and JavaScript issues
  - ✓ Note: whenever page refreshes iFrame goes to page one
- Servlet web apps can use JBoss SSO for authentication and authorization
  - ✓ Servlet web apps on JBoss AS can use the iFrame Portlet and JBoss SSO for quick integration into JBoss Portal

Portal Design and Layout

Customizing Portal Pages in 2.0.x

- Navigation Tabs
  - ✓ Custom Tag Library for navigation tabs using role security
- Our Window State Interceptor
  - ✓ Remembers normal/minimized state (user preferences)
  - ✓ Checks user permission to access portlet
  - ✓ If not, sets window title to "inaccessible"
- Use interceptors to modify JBoss Portal



Portal Navigation in 2.2.x

- Enhanced navigation requirements
  - ✓ Controlling tab order
  - ✓ Hidden portal pages – *"a page without a tab"*
  - ✓ Customizing tab generated HTML
- JBoss Portal 2.2.x Navigation Portlet
  - ✓ Create custom version of JBoss Portal 2.2.x Nav Portlet
    - org.jboss.portal.core.portlet.catalog.NavigationPortlet
    - Add new DeclaredProperties in extended portlet

```
<page>
    <page-name>Home</page-name>
    <properties>
      <property>
        <name>page.order</name>
        <value>1</value>
      </property>
...
```


JBoss Portal Strategy

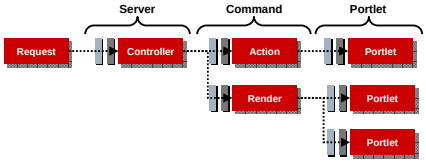
- Using a Strategy
  - ✓ Role security can restrict access for a portal window
  - ✓ How to display a window when user is not authenticated
  - ✓ Add a Strategy that checks request for user principle

```
public StrategyResponse evaluate(StrategyContext context) throws
        StrategyException {
    StrategyResponse response = context.createResponse();
    if (context.getHttpServletRequest().getUserPrincipal() != null) {
        List<PortletContext> portletContexts = context.getPortletContexts();
        for (PortletContext portletCtx : portletContexts) {
            String windowName = portletCtx.getWindowContext().getWindowName();
            if (windowName.contains("loggedInOnly")) {
                response.addNoRender(portletCtx);
            }
        }
    }
    return response;
}
```


Portlet Development Best Practices

Key Portlet vs. Servlet Difference

- JSR-168 defines two phase request processing



- Servlet apps define request/response model

Choosing a MVC Framework?

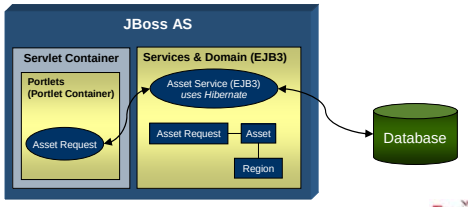
- Initially tried using Struts
  - ✓ In a portlet container, Struts apps need a Struts bridge
  - ✓ Had to customize bridge for required Struts functionality
  - ✓ Only use for making legacy Struts apps into Portlets
- Decided to use JSF instead
  - ✓ Not the only alternative, but probably the best supported
  - ✓ JavaServer Faces (JSF) spec requires support for portlets
  - ✓ Two common JSF implementations
    - MyFaces (Apache)
    - Reference Implementation (Sun)
  - ✓ JBoss developers wrote portlet functionality in MyFaces
    - Better support for MyFaces than JSF RI
  - ✓ We have JSF portlets in production → working well
- Spring is another viable alternative

Some Benefits of MyFaces

- Tomahawk components
  - ✓ File Upload
  - ✓ Calendar
  - ✓ Tabbed Pane
- Some Tomahawk components didn't initially work in portlets
  - ✓ Usually because of a cast to HttpServletRequest
  - ✓ Because MyFaces is Open Source we could find the problem, fix it, and then contribute back to MyFaces
  - ✓ JBoss Professional Services helpful here
- The MyFaces sandbox also has some JSF AJAX components in development

Using EJB3 from JBoss Portlets

- If you like Hibernate, you'll really like EJB3
- Let's take a look at
  - ✓ Using EJB3 Stateless Session Beans and Entities
  - ✓ Advantage of an injected Entity Manager
  - ✓ Accessing a EJB3 Session Bean from a Portlet



An EJB3 Session Bean

- Annotations to make a POJO an EJB3 Session Bean

```
@Stateless
@Local({ApprovalService.class})
public class ApprovalServiceBean implements ApprovalService {
    @PersistenceContext(unitName = "approval")
    private EntityManager entityManager;
    …
}
…
```

- Service method uses the injected EntityManager

```
public ApprovalStatus getStatus(String year) {
    String queryString = "select status from ApprovalStatus status " +
            "where status.fiscalYear = :fiscalYear";
    Query query = entityManager.createQuery(queryString);
   query.setParameter("fiscalYear", year);
    return (IuApprovalStatus) query.getSingleResult();
}
```


Using EJB3 Services from Portlets

- A portlet invokes the Session Bean by

```
InitialContext context = new InitialContext();
ApprovalService service =
    context.lookup("approval/ApprovalServiceBean/local");
ApprovalStatus status = service.getApprovalStatus(year);
```

- When deployed in EAR the JNDI name will be

```
<ear name>/<ejb-name>/local or remote
```

- Keep business logic in Services and Entities
  - ✓ Business logic in web apps creates spaghetti code
  - ✓ EJB3 Entities are POJOs allowing unit testing → JUnit

Our Next Steps for Portlets…

- Currently Prototyping JBoss Seam
- Seam is a JBoss framework that acts as the glue between JSF and EJB3
- Should enhance developer productivity and reduce bugs
- Supported in recent JBoss Portal release
- See Seam Tutorial in Seam Reference Doc

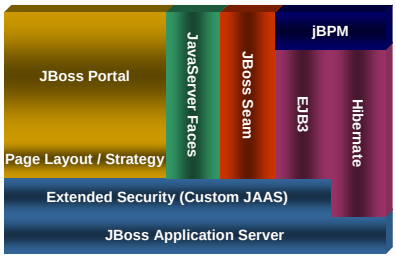
JBoss World 2006 LAS VEGAS

Integrating Business Process Management

Using jBPM in Portlets?

- Navy apps had common workflow requirements
- Ability to deploy new workflow versions without redeploying application/portlet
  - ✓ Process Instances are tied to a Process Definition
  - ✓ By default, Process Instances use latest Process Definition
- Removing workflow logic simplifies portlets
  - ✓ Shortens portlet development
  - ✓ Less code to write reduces bugs & increases productivity
- Template Pattern for common functionality
  - ✓ Common Service Template class
  - ✓ Common JSF Backing Bean
  - ✓ Common jBPM AssignmentHandler

Putting The JBoss Pieces Together…



We have been running this architecture (minus Seam) in production since June 2005