



# Portlets and WSRP

**Rickard Öberg**  
**Product Architect**  
**Senselogic**



# Agenda

- **web.next**
- **Portlets**
- **WSRP**
- **Why not?**
- **Conclusions**



# Part 1, web.next



# Web service ecology

- **Websites expose data through webservices**
  - Amazon
  - Retailers
- **Other websites gather data, combines and refines it, and produce new services on top**
- **How do they expose their web services?**
  - Services use (X)HTML and should be consumed by real users



## Example: Bob's Car Shop

- **Car shop has internal web portal where servicemen can search for cheap parts**
- **Parts can come from many places**
- **Search is provided by external ASP**
- **How should it be integrated?**



# Common integration techniques

- **Links**
- **iframe**
- **Web clipping**
- **Problem: these are all mostly transport-oriented solutions. What about security and business relationships? What about design issues?**



## Web components from n vendors

- **How do you integrate web components from many vendors?**
- **Apps typically utilize the entire page**
- **Look & feel issues**



# Security

- **How do you handle security without common rules?**
- **How does application provider know whether client may access it or not?**



# Personalisation and profiling

- **Users want websites the way they want it (personalisation)**
- **Service owner want users to get the information they want (profiling)**



## Scaling it up

- **When is the Internet sleeping? Always on!**
- **How many users do you have? One today, a thousand tomorrow, ...**
- **In a web service ecology, how do you handle inter-DMZ communication?**



# Part 2, Portlets



# Container for web components

- **Portlet API is based on servlet model in Java EE**
- **Portlet class + XML descriptor**
- **Portal provides secure sandbox and layouting of portlets**
- **Portlets deliver the actual functionality**



# Security

- **Today's apps require a LOT in terms of security**
  - **Secure transport (HTTPS)**
  - **Single Sign On (reverse proxies)**
  - **Network topology issues (DMZ vs internal nets)**
  - **Administration of “who gets to do what and where”**
  - **Account management (use LDAP!!!)**



# Security management

- **Portals are typically very good at those things**
- **Application developers are typically very bad at those things**
- **Building a good security framework takes a lot of time and know-how**



# Portal and portlet security integration

- **Portal handles logins**

- More and more common with strong authentication using OTP or certificates

- **Portal ensures that portlet is accessed only by those with proper rights**

- **Portlet can access user info and role info through Portlet API**

- Map with user info is available (name, phone, email, etc.)
- Can also use JAAS if available



# Personalisation and profiling

- **Users sometimes want to customize parts of portal**
  - **Example: customize how forums are presented**
- **Administrators sometimes want to profile parts of portal for groups of users**
  - **Example: show extra system status messages on intranet if you are a member of the IT admin group**



## The portal as a desktop

- **Roles and rights can be used to restrict what portlets are available**
- **Typically simplifies life for the user**
  - “See what you need, skip what you don't need”
- **Allowing user to modify layout is typically a bad idea**
  - Support and training nightmare



# Scaling it up

- **Portals can be made highly scalable**
  - both vertically and horizontally
- **Since portal control configuration data it can optimize how clustering and replication works**
- **Portlet API supports caching**
  - Currently rather simplistic however



## What's in Portlet API 2.0?

- **Add access to CC/PP data via JSR188 API**
- **Introduction of portlet filters**
- **Inter-portlet communication as defined in WSRP 2.0**
- **Public render parameters as defined in WSRP 2.0**
- **Enhance caching support**
- **Align with WSRP versions 2.0**



# Part 3, WSRP



# What is WSRP?

- **Web Services for Remote Portlets**
- **Transport for web components**
- **Uses SOAP**
- **Has both business and technical concepts**
  - **Consumer registration**



# Security

- **WSRP leverages WS-Security and SAML for user authentication and identification**
- **May provide extra user profile information in call, to be used for security and output customization**
- **WSRP fixes some security issues with running portlets**
  - Using external resources
  - Crashing portlet, crashing portal



## Personalisation and profiling

- **Users can create their own instances of remote portlets**
- **Remote portlet invocation may include user profile information**



# Scaling it up

- **Using a loadbalanced Web Service infrastructure**
- **Cache is king**
- **Using non-HTTP based SOAP transfer**
  - **Especially if Consumer and Producer are of the same package**



# What's in WSRP 2.0?

- **Inter-portlet communication**
  - Send events between portlets
- **Better defined security**
  - WS-Security and SAML



## So how do I use it?

- **Roll your own**
  - May be expensive
- **Use a portal product (OpenSource or commercial)**
- **Integrate with a pluggable WSRP implementation**
  - WSRP4J
  - OpenWSRP



# Part 4, Why not?



## Why not use Portlets?

- **Has lacked MVC frameworks**
  - WebWork and Spring
- **Portals software often made by developers for developers**
  - 99% of our users and admins are not developers(!)
- **Many portals have lots of legacy baggage**
- **Many portals erringly use the “portal desktop” metaphor**
- **Some technical issues – many are resolved in Portlet 2.0 API**



## Why not use WSRP?

- **Performance is critical**
- **Using AJAX may not work**
- **Require features not yet available**
- **Interoperability**
- **Don't want to tie your app to a particular way of delivering it**



## Part 5, Conclusions

- **Portlets provide a good way to create web components**
- **Portals can be a big help in packaging your functionality**
- **WSRP as an enabling technology will boost the next web**
- **Many issues to overcome**
- **No good alternatives available**



# Questions