

# Web Services For Remote Portlets

## Vision for user-facing web services

Jana Polgar (Monash University)  
 Tony Polgar (Sensis Pty. Ltd)  
 Ashvin Ramanjooloo (Monash University)

# Outline

- What are portals and portlets?
  - WebSphere Portal 5.1
  - JSR168 and JSR286
- Web services
  - Data oriented web services
  - Presentation oriented web services (WSRP)
- WSRP architecture
  - Concept
  - Walk through some APIs
  - Standards WSRP 1.0 and 2.0 draft
- Demonstration
  - Dynamic Services for Remote portlets capability assessment on the WebSphere Portal 5.1
    - The tool developed by Monash University to support out-of-band WSRP registration

# Portals and Portlets

What are portals and portlets?  
 • WebSphere Portal 5.1  
 • JSR168 and JSR286

- Portals
  - Integrated view of multiple applications
- Portlets
  - Managed by a portal container
  - Generate a piece of markup called "fragment" that area aggregated to a web page
    - Adhere to certain rules such as no <html> tags
  - Can use configuration and personalisation



•Portals are considered as the future desktops for industry integration efforts

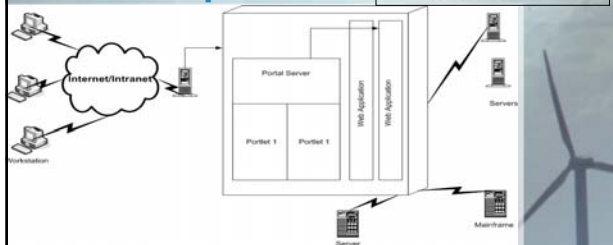
# Portal Solutions

What are portals and portlets?  
 • WebSphere Portal 5.1  
 • JSR168 and JSR286

- The portals aim at
  - Provide the end user with convenient access to interact with enterprise applications, people, content, processes
  - Provide site users with a *single point of access (SSO)* to
    - Multiple types of information,
    - Legacy applications
    - Local and remote web services Deliver personalized access
  - Organize end-user view of business processes
  - Personalise and manage end-user own profiles
  - Publish and share documents
  - Provide support for multiple e-business applications over the Web

# Portal concept

What are portals and portlets?  
 • WebSphere Portal 5.1  
 • JSR168 and JSR286

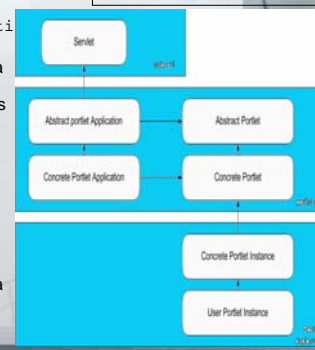


- Portals are composed of portal configuration and multiple portlet applications
- All portlets in the application share the same context root
- Portlet Applications are packaged in web modules (WAR file)

# Configuration and personalisation

What are portals and portlets?  
 • WebSphere Portal 5.1  
 • JSR168 and JSR286

- Concrete Portlet Application
  - A portlet application parameterised with API - `PortletApplicationSettings`
  - Allows for multiple copies of a portlet application using different configuration settings
- Concrete Portlet
  - A portlet parameterised with `PortletSettings`
- Concrete Portlet Instance
  - A portlet placed on a page
  - Parameterised with `PortletData`
  - Allows for multiple copies of a portlet using different *user settings*



## Modes, states and events

What are portals and portlets?

- WebSphere Portal 5.1
- JSR168 and JSR286

➤ **PORTLET MODES**

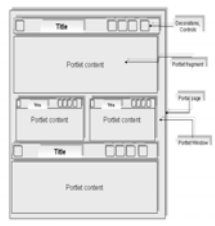
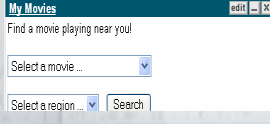
- View
- Edit
- Configure
- Help

➤ **WINDOW STATES**

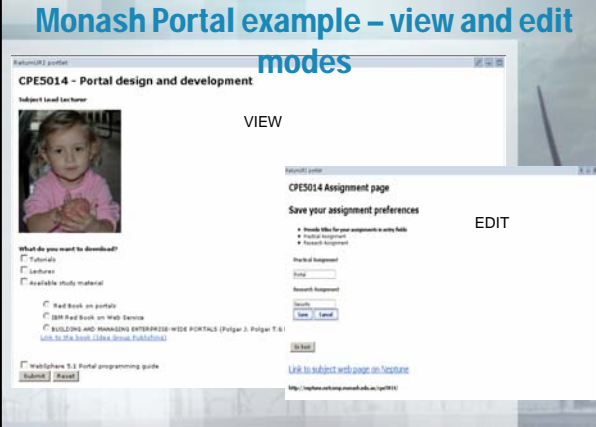
- Maximized
- Minimized
- Normal
- Solo

➤ **EVENTS**

- Action
- Message
- Window

## Monash Portal example – view and edit modes



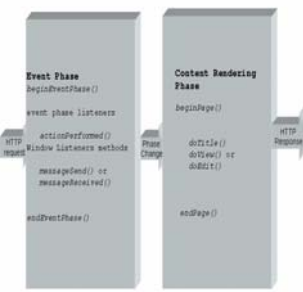
## Page aggregation

What are portals and portlets?

- WebSphere Portal 5.1
- JSR168 and JSR286

➤ **Two phase process**

- Events processed
  - ActionListener
  - MessageListener
  - WindowListener
- Content rendering
  - Starts with `beginPage()`
  - TitleListener is processed
  - Ends with `endPage()`
  - Aggregated content (response) is returned to the portal server



## Standards

What are portals and portlets?

- WebSphere Portal 5.1
- JSR168 and JSR286

➤ **JSR 168**

- Finished in October, 2003
- Co-led by IBM and Sun
- Web Service for Remote Portlets (WSRP), defined at OASIS, aligned with JSR 168 ([http://www.oasisopen.org/committees/tc\\_home.php?wg\\_abbrev=wsrp](http://www.oasisopen.org/committees/tc_home.php?wg_abbrev=wsrp))
- Standard defines
  - Portlet API and portlet container
  - Contract between the API and the container
  - Deployment unit: portlet application
- Out of scope are
  - Aggregation, layout management
  - Page personalization and configuration engines
  - Portal administration and configuration

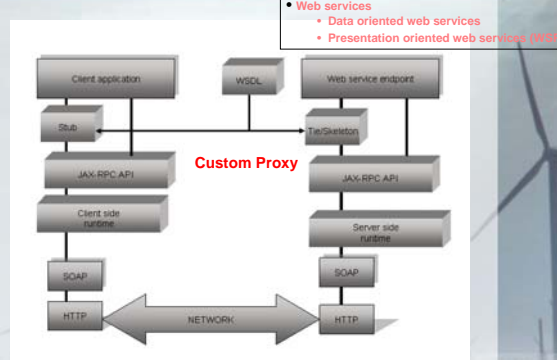
➤ **JSR 286**

- Version 2.0 of the Portlet Specification plans to align with J2EE 1.4, integrate other new JSRs relevant for the portlet, and align with the WSRP specification V 2.0
- Started 29th Nov 2005
- Led by IBM
- JSR 286 will introduce a loosely coupled event paradigm
  - A portlet can declare events it wants to receive, and events it wants to emit
  - The portal / portlet container will act as broker and distribute the events accordingly
  - Allows wiring of portlets at runtime
  - Portlets can share session related data across web application boundaries

## RPC based Web Services – data oriented

Web services

- Data oriented web services
- Presentation oriented web services (WSRP)



## Vision for presentation oriented web services

Web services

- Data oriented web services
- Presentation oriented web services (WSRP)

➤ **WSRP is a protocol between consumer and producer**

- Producer provides remote portlets
- Consumer uses remote portlets

➤ **The goal of WSRP - to promote business integration** by providing a framework for sharing web service presentation components

## WSRP concept

- WSRP architecture
  - Concept
  - Walk through some APIs

➤ The web service interfaces exposed by the **producer** to the **consumer** are described using WSDL extensions for portlets

➤ Optionally, **consumer** can be registered in a **producer's** portal

➤ The local portal detects the remote portlet (or remote portlet proxy) on its page and sends a `getMarkup()` message to the **producer**

➤ In response, it receives a HTML fragment from the **producer**

➤ The **consumer** portal aggregates the fragment into the portal page

➤ Optional functionality is provided by the portlet management, which defines operations (an API) for cloning, customising and deleting portlets.

13

## WSRP protocol interfaces

- WSRP architecture
  - Concept
  - Walk through some APIs

➤ **Service Description Interface (required)**

- producer** advertises services and its capabilities to consumers

➤ **Markup Interface (required)**

- consumer** interacts with a remotely running portlet supplied by the **producer**

➤ **Registration Interface (optional)**

- a mechanism for opening a dialogue between the **producer** and **consumer**
  - Out-of-band mechanism to ensure compatibility**
  - In-band mechanism to register with the producer**

➤ **Portlet Management Interface (optional)**

- gives the **consumer** control over the life cycle methods of the remote portlet
  - Clone or destroy the portlets

14

## WSRP versus Web Services

- WSRP architecture
  - Concept
  - Walk through some APIs

a) Data Oriented Web Services

b) Presentation Oriented Web Services

15

## WSRP – UDDI

- WSRP architecture
  - Concept
  - Walk through some APIs

Portal clients access portals via the HTTP protocol. Two different kinds of portlets are available:

- Local Portlets** that run on the consumer portal server
- Remote Portlets** run as Web services on remote servers

16

## Basic scenario of WSRP - Monash Portal

- WSRP architecture
  - Concept
  - Walk through some APIs

17

## Service Description API

- WSRP architecture
  - Concept
  - Walk through some APIs

The Campus wishes to know the capabilities of FIT and list of portlets available for the subject CPE5014 (tutorials, lecture notes, lab setup notes). It sends the request for description with the question if registrations is required.

➤ Request (`getServiceDescription()`) contains parameters:

- `RegistrationContext`
- `desiredLocale`

Returns `ServiceDescription` structure

18

## Service Description API

- WSRP architecture
- Concept
- Walk through some APIs

*In response from FIT portal, the Campus receives metadata, supported capabilities and list of available portlets for CPE5014*

`getServiceDescriptionResponse()` contains

- Metadata
  - Registration is required (true/false)
- Capabilities
  - Window States, locale, portlet handles
    - » Some states may not be supported by the Consumer portal
  - Supported locales, MIME types, description and title
- List of portlets in this application in ServiceDescription and for each portlet
- PortletDescriptions structure
  - `portletHandle` to FIT home
- Note that Metadata can change any time, resulting in
  - Consumer portlets may not function properly
  - Some data are only optional
  - Is this similar to a web service proxy change?

19

## Registration is contract

- WSRP architecture
- Concept
- Walk through some APIs

*FIT courses are mostly private property and only Campus can view the FIT metadata - Registration is required. As a large University, each Campus may have different version of portal. FIT is running WebSphere 5.1 and Gippsland Campus only WebSphere 4.0*

- The Campus registers **out-of-band** with FIT
  - In most cases, person-to-person **contract agreement** between administrators
  - Campus is required to provide unique numbers for campus ID (CID) and service ID (SID)
    - These values are part of the service description as pairs
      - `property_type/property_value`
- The Campus now can use **in-band** registration
  - Sends registration request to the FIT portal with CID and SID
  - In response it receives
    - `registrationContext` object to use in future communications

20

## Registration Interface

- WSRP architecture
- Concept
- Walk through some APIs

**WSRP standard does not specify or restrict the registration**

- **Consumer** can use the registration to let know what capabilities he can support
- **Producer** can keep track of portlet used as WSRP
  - E.g. associate persistent state with the Consumer registration
- **Producer** attempts to meet the **Consumer** requirements and tailors the portlets on the Consumer by Consumer basis
- **Producer** may restrict some capabilities
  - use of `PortletData` (personalisation), cloning

21

## Registration details

- WSRP architecture
- Concept
- Walk through some APIs

- **Producer** provides the **Consumer** with
  - `registrationContext` object
    - This object contains:
      - `registrationHandle` – it remains unchanged for the duration of registration
      - `registrationState` (optional)
- **Operations:**
  - `register()`
  - `modifyRegistration()`
  - `deregister()`

22

## Registration properties

- WSRP architecture
- Concept
- Walk through some APIs

**Consumer** may provide some additional information

- Consumer Name (Campus Name)
- Consumer Agent (name and version of the Consumer)
- support for method `GET`
  - The Consumer can aggregate markup containing form tags with method `GET`
- Supported **Consumer** modes and Window states
- Extensions

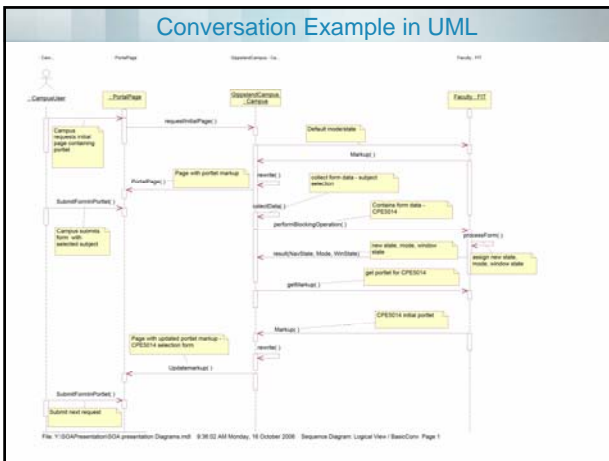
23

## Processing cycle

- WSRP architecture
- Concept
- Walk through some APIs

*The Campus requests the FIT home portlet and portlet with the list of available material for CPE5014 Access to the Subject Lead Lecturer*

24



### Markup `getMarkup()`

- WSRP architecture
- Concept
- Walk through some APIs

➤ **Two step protocol**

- User interacts with the portlet on a portal page
  - `getMarkup()` - requests the initial portlet (FIT home portlet)
    - page is generated with remote and local portlets
  - submits a form and forwards the interaction to the Producer
  - `performBlockingInteraction()` - processes user interaction, updates transient and persistent states in its response

### Markup Interface

- WSRP architecture
- Concept
- Walk through some APIs

➤ `initCookie()`

- Producer initializes cookies, and sends those cookies to the Consumer.
  - Producers may return cookies to Consumers for some implementation specific purposes (load-balancing)
    - Consumer must run the method
- `releaseSessions()`
  - portlets can initialize sessions
    - portlet creates a session
    - the Producer returns a `sessionID` to the Consumer in its response
    - Consumer uses the `releaseSessions()` operation to let the Producer know that those sessions expired

### State management

- WSRP architecture
- Concept
- Walk through some APIs

➤ **Transient state**

- Navigational state**
  - `performBlockingInteraction()` requests results in response to Consumers
    - `navigationalState` field is used to indicate the state
  - Consumer returns this navigational state to the Producer with next `getMarkup()` call
    - typically encapsulates state tracking data
- Session state (local at Producer)**
  - the Producer assigns a `sessionID` and returns it to the Consumer
  - Consumer returns this `sessionID` to the Producer with future requests

➤ **Persistent state**

- Consumer registration
- Persistent Portlet state
  - can be changed by Producers during interactions
  - Only Consumer knows if it is safe (`portletStateChange`)
    - `readOnly`
    - `cloneBeforeWrite`
    - `readWrite`

Only Portlet knows

### URL generation in remote portlets

- WSRP architecture
- Concept
- Walk through some APIs

All URLs embedded in the markup fragment returned by the remote **Producer** service must point back to the **Consumer** application

- Consumer**
  - Sends URL template as part of the invocation of the `getMarkup()` method with two variables: `navigationalState` and `sessionID`
  - `http://neptune.monash.edu.au/myApp?ns={navigationalState}&si={sessionID}`
- Producer**
  - Returns the template variables `navigationalState` and `sessionID` replaced with concrete values
  - `http://neptune.monash.edu.au/myApp?ns=page2&si=4AHH55A`

### Modes and Window states

- WSRP architecture
- Concept
- Walk through some APIs

➤ Portlets in WSRP can use different **modes**:

- Specified in `PortletDescription`
  - Producer may let Consumer know
    - Can be modified after registration
- each mode treats a particular function
- treatment is the same as in local portlets
  - default functionality in `wsrp:view` mode
  - customization functions in `wsrp:edit` mode (persistent data)
  - Help functions in `wsrp:help`
  - `wsrp:preview` - example of VIEW rendering
  - Custom modes

➤ **Window states** allow the Consumer to indicate markup size of a portlet

- `wsrp:normal` - sharing page with other portlets
- `wsrp:minimized`
  - not visible markup but JavaScript is allowed
- `wsrp:maximized` - rich content is expected
- `wsrp:solo`
- `wsrp:custom`

## Management interface

- WSRP architecture
- Concept
- Walk through some APIs

- This functionality can be used by the **Consumer** administrators to modify portlets properties
  - Properties can be modified by administrators
    - `getPortletPropertyDescription()`
    - Then set up user-friendly interface to maintain properties
  - E.g. the Campus wishes to provide a page for administrator to view the description of FIT home portlet
    - This process involves a call to the FIT (Producer)
      - `getServiceDescription()`
      - `getPortletDescription()`
    - FIT returns for example
      - Mime type
      - State
      - Modes available
      - Locale
      - Title
    - Form representing each property can be designed to help with the customisation of properties
      - EDIT button for settings management for the portlet
      - We call the configured portlet "Consumer Configured Portlet"

31

## Fault handling and API extensions

- WSRP architecture
- Concept
- Walk through some APIs

- Each WSRP fault has an associated code and **fault\_string**
  - **Producer** returns both the code and **fault\_string** in the SOAP response
- API extensions
  - are implementation specific
  - may cause the **Consumer** not functioning properly
  - at the **Producer** site should be only optional in order to support cooperation with the **Consumers**

32

## WSRP Standards - V 1.0 and 2.0 draft

- WSRP architecture
- Standards

- **WSRP V1.0**
  - **Approved OASIS standard**
  - Based on language and platform independent technologies such as SOAP, WSDL, UDDI
  - Today, highly coupled with JSR 168 and Java
- **WSRP V2.0**
  - JSR 286 will align with WSRP V 2.0
  - Enables writing JSR 286 portlets and publish them as WSRP services
  - New features in WSRP 2.0
    - Coordination
    - Resource serving
    - Portlet management
    - Import / export of portlets
    - Runtime IDs, similar concept as portlet instance ID

33

## WSRP Extension to WSDL

- WSDL defines the following **portTypes** (normative definitions):
  - **WSRP\_v1\_Markup\_PortType:**
    - port on which the Markup Interface can be accessed. **All producers** must expose this portType.
  - **WSRP\_v1\_ServiceDescription\_PortType:**
    - port on which the Service Description Interface can be accessed. **All producers** must expose this portType.
  - **WSRP\_v1\_Registration\_PortType:**
    - port on which the Registration Interface can be accessed. Only **producers** supporting in-band registration of **consumers** need expose this portType.
  - **WSRP\_v1\_PortletManagement\_PortType:**
    - port on which the Management Interface can be accessed. Only **Producers** supporting the portlet management interface expose this portType. If this portType is not exposed, the portlets of the service cannot be configured by consumers.
- SOAP bindings for these **portTypes** are listed below:
  - **WSRP\_v1\_Markup\_Binding\_SOAP:**
    - All **producers** must expose a port with this binding for the **WSRP\_v1\_Markup\_PortType** (the Markup portType).
  - **WSRP\_v1\_ServiceDescription\_Binding\_SOAP:**
    - All **producers** must expose a port with this binding for the **WSRP\_v1\_ServiceDescription\_PortType** (ServiceDescription portType).
  - **WSRP\_v1\_Registration\_Binding\_SOAP:**
    - **Producers** supporting the Registration portType must expose a port with this binding for the **WSRP\_v1\_Registration\_PortType**.
  - **WSRP\_v1\_PortletManagement\_Binding\_SOAP:**
    - **Producers** supporting the PortletManagement portType must expose a port with this binding for the **WSRP\_v1\_PortletManagement\_PortType**.

34

## WSRP Registration tool developed at Monash

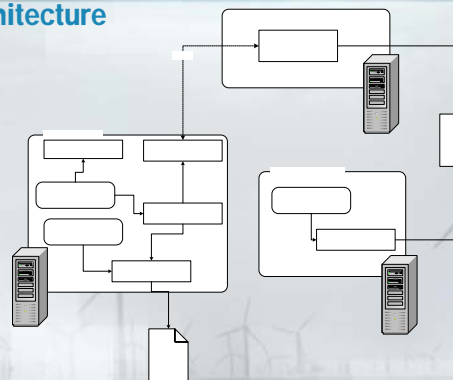
The **portal implementation of the producer** (the portlet API) may not be **compatible** with that of the consumer

- Some interfaces are only optional and not supported by all vendors
- `PortletDescription` and `ServiceDescription` is available too late in the process
- portlet API mismatch can seriously **destabilise** the consumer portal
- Implementing the access to the request parameters affects **portability**
- Any extensions implemented by the Producer may reduce the interoperability at Consumer
- Out-of-band registration is mostly verbal agreement between administrators
  - Significant testing is required before the Producer portlets can be cloned or used at the Consumer

35

## WSRP Registration Tool Architecture

- WSRP registration tool



36

## WSRP Registration Tool Components

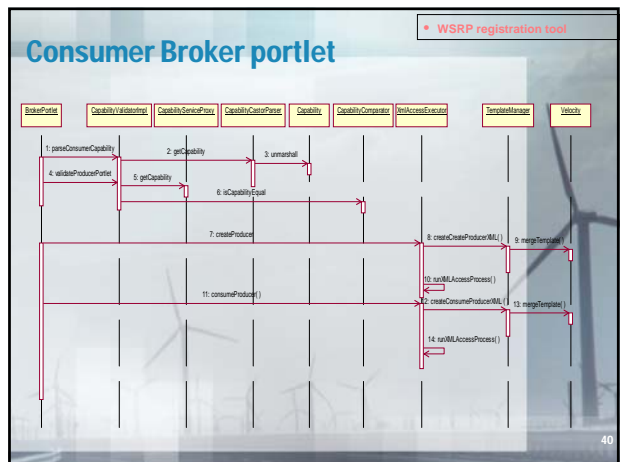
- Consumer and Producer define their capabilities in xml files based on **Capability.xsd**.
- The producer domain provides a webservice to return the capabilities of the producer portal.
- The consumer portal runs the broker application consisting of :
  - Capability portlet**
    - Allows the user to modify the capabilities of the portal. Makes use of **Castor** to convert XML to Objects and vice versa. The capabilities are encapsulated within Castor generated objects.
  - Broker portlet**
    - Two modules :
      - Validation module
      - Registration module

## WSRP Registration Tool Components

- Validation Module**
  - Consists of the web service client side of the capability service.
  - Uses **CapabilityComparator** Class to compare the capability Objects for the producer and the consumer. It actually compares the bean properties within the capability object by reflection.
  - As soon as properties are not equal, the process stops and the producer portal is deemed incompatible with the consumer.

## WSRP Registration Tool Components

- Registration Module**
  - Creates and consume the producer portlets
    - Uses **XML configuration interface**
    - Uses a **TemplateManager** which generates XML files for the XML configuration interface from templates. The TemplateManager makes use of **Velocity**.
    - Uses **XMLAccessExecutor** to execute the **XMLAccess.bat** with the generated XML files so as to create and consume the producer.
  - To create a producer, the **URL** of the producer WSDL service definitions and **unique name** are specified
  - To consume a producer portlets, the **group id** and **handle** of the remote portlet are required. Typically obtained from the administrator of the producer portal.



## WSRP Registration Tool Demo

## Discussion and demo follows

<http://neptune.netcomp.monash.edu.au/CPE5014/lectures>