

# Development of Java™ Portlets using Java Portlet 2.0 API, Java Server Faces (JSF 1.2 & 2.0) and Ajax

Trayan Iliev

IPT – Intellectual Products & Technologies

e-mail: [tiliev@iproduct.org](mailto:tiliev@iproduct.org)

web: <http://www.iproduct.org>

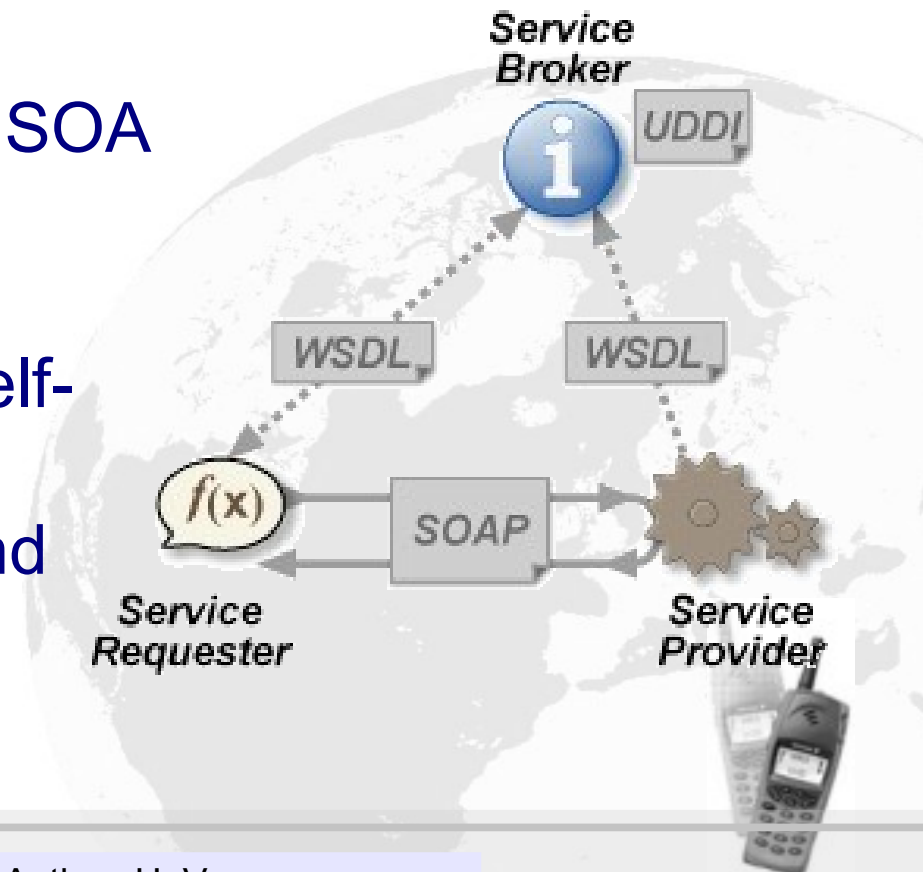
Oracle®, Java™ and EJB™ are trademarks or registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.



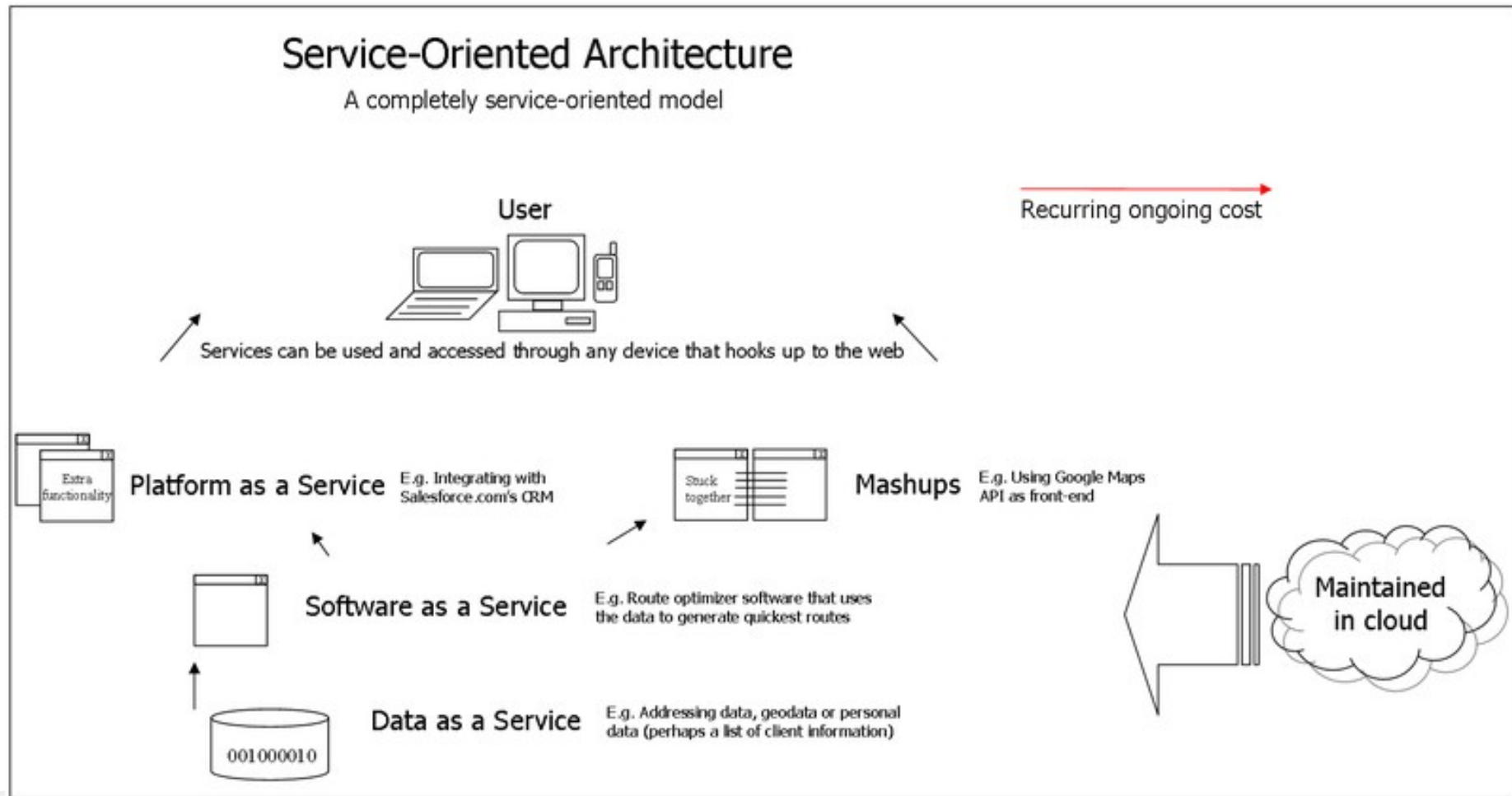
## Web Services

### Web Services are:

- components for building distributed applications in SOA
- communicate using open protocols
- are self-descriptive and self-content
- can be searched and found using UDDI or ebXML registries



# Service Oriented Architecture (SOA)



## Java™ EE 6 / Java™ SE Standards APIs

According to Java™ EE Specification:

- **Web Services**
  - Java API for XML Web Services (JAX-WS)
  - Java Architecture for XML Binding (JAXB)
  - SOAP with Attachments API for Java (SAAJ)
  - Java API for XML Registries (JAXR)
  - ~~Java API for XML-based RPC (JAX-RPC)~~
- **RESTful Web Services**
  - Jersey – RESTful Web Services - JAX-RS

## Web Services Support

- XML -based web services:
  - Simple Object Access Protocol (SOAP)
    - XML-based envelope
    - XML-based encoding rules
    - XML-based request and response convention
  - Web Services Description Language (WSDL)
  - Universal Description, Discovery and Integration (UDDI) and ebXML Registries integration

## SOAP Example (1)

- SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>  
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">  
  <S:Header/>  
  <S:Body>  
    <ns2:add xmlns:ns2="http://calculator.me.org/">  
      <i>5</i>  
      <j>12</j>  
    </ns2:add>  
  </S:Body>  
</S:Envelope>
```

## SOAP Example (2)

- SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?>  
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">  
  <S:Body>  
    <ns2:addResponse xmlns:ns2="http://calculator.me.org/">  
      <return>17</return>  
    </ns2:addResponse>  
  </S:Body>  
</S:Envelope>
```

## Representational State Transfer (REST)

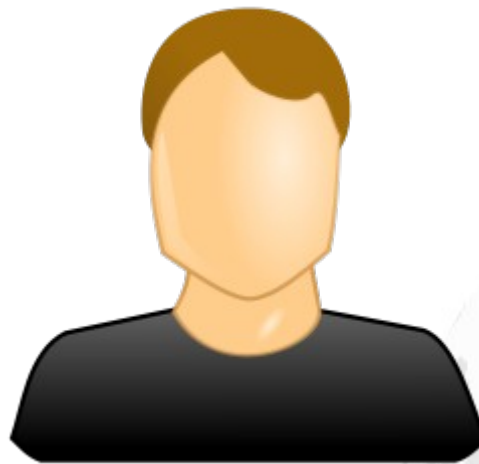
- REpresentational State Transfer (REST) is a distributed hypermedia access web-service architecture.
- The resources are identified by URIs and are manipulated using an HHTTP interface
- Information is exchanged using representations of these resources
- Lightweight alternative to SOAP+WSDL -> HTTP

## Question

What is missing from the SOA picture so far?



## Answer



THE User

Only server side technologies are not enough ...



## What are Portlets for?

- In 2002 the Java™ community started a pioneering effort for standardization of the **web-based presentation services** as well
- Can be easily combined and integrated into **enterprise portals** based on a common standard defining their interaction – **JSR 186: Java Portlet Specification**
- In 2005 it was followed by another more advanced specification – **JSR 286: Portlet Specification 2.0**





# IPT - Intellectual Products & Technologies

Knowledge makes you free ∞

Welcome!

- VOD
- SemanticContent
- UGC
- Welcome
- Temp
- Admin
- Semantic Demo
- MyChat2
- JSF 2.0 Jobs
- JFS 2.0 Events
- JSF 2.0 Ajax Push

**VOD Chooser**

A-Z New Top 10 Most Seen Search Cart FAQ Help

**Rubrics**


- IPT Ads and Intros
- Black Sea Moments
- Black Sea Panorama
- Black See Views
- Veleka River
- IPT Announcements

**Videos**

- A Boulder in the Water**  
*There is a Boulder in the Sea near*  
[read more ...](#)
- Sunset 1**  
*The Sunset on the Butamiata Beach*  
[read more ...](#)
- Wild Sea 1**  
*Wild Black Sea near Veleka Beach*  
[read more ...](#)

**IPT VOD Player**

IPT Video On Demand



Add to Cart Visited 1 times ★★★★★

Video Metadata

## Why to Use Portals? (1)

- Portals offer many advantages over other software applications (<http://portals.apache.org/>):
  - provide a **single access point** for all employees, partners, and customers
  - provide access to business functionality transparently from any device in virtually **any location**
  - portals are highly **flexible** - they can exist in the form of B2E intra-nets, B2B extra-nets, or B2C inter-nets

## Why to Use Portals? (2)

- Portals offer many advantages over other software applications (<http://portals.apache.org/>):
  - portals can be combined to form a **portal network** that can span a company's ecosystem
  - because portals provide **front end for different web services** they can easily integrate existing heterogeneous software systems and are **future-proof**

## Evolution of Portals (1)

- Portlets became popular – allowing easily to share and combine web applications developed by different organizations and individuals in a **personalized enterprise portal**
- A new "**portlet-based**" style of web application development has emerged. **Portlet Apps (PA)** are more distributed, flexible and agile, compared to older style, monolithic web applications we know for years.

## Evolution of Portals (2)

- **PA** can use asynchronous data requests and can be dynamically updated in response (and sometime in anticipation) to user's needs.
- According to latest **Portlet Specification 2.0**:
  - **PA** are typically consisting from several different portlets
  - Different **Portlets** can communicate using **shared parameters** or **publish/subscribe events** and can query server resources dynamically (**AJAX**)

## Future: Web Services for Remote Portlets (WSRP)

- WSRP v1 as an OASIS standard in September, 2003
- WSRP v2 was approved by OASIS on April 1st, 2008
- WSRP v2 supports Web 2.0 technologies, such as **AJAX** and **REST**
- Allows the portlets to be accessed remotely and combined easily by third party portals and consumers
- Supported by all of the portal market's major players, including **Oracle®**, **IBM®**, **Microsoft®**
- The ultimate goal of WSRP is to bring the benefits of **Service-Oriented Architecture** to the **end-user...**



## JSR 286: Java™ Portlet 2.0 API Details

- `init()` - called after the portlet is instantiated by the container
- `destroy()` - called before the container destroys the portlet
- `processAction()` - called when user changes the portlet state
- `render()` - called on each re-drawing of the portlet by the portlet window
- `doView()` - called by `render()` when in View portlet mode
- `doEdit()` - called by `render()` when in Edit portlet mode
- `doHelp()` - called by `render()` when in Help portlet mode
- Portlet window states: **MINIMIZED, NORMAL, MAXIMIZED**

## JSR 314: JavaServer Faces 2.0 (1)

- In order portlet technology to become even more successful are **needed developer frameworks and tools** that support rapid portlet application development and reuse of components
- As part of Java™ Enterprise Edition standardization process and as a recommended technology for web based presentation in the latest version of Java™ EE 6, **Java Server Faces (JSF) Model-View-Controller (MVC)** framework became most promising candidate for such an enabling technology.

## JSR 314: JavaServer Faces 2.0 (2)

Among the advantages of JSF are:

- easy construction of UI from a set of reusable components;
- clear separation between data and presentation using MVC design pattern;
- easy to use model for wiring client-generated events to server-side application code;
- UI components state managed automatically across client requests;
- separation of concerns between corporate developers and system programmers.

## JSR 314: JavaServer Faces 2.0 (3)

The latest version of JSF 2.0 provides additional advantages:

- more flexible and standard based presentation components using facelets;
- easy to use view/page templating;
- easy to create custom components without Java programming by composing existing components;
- seamless and unified integration with all different types of beans (ManagedBeans, POJO, EJB) using dependency injection annotations;

## JSR 314: JavaServer Faces 2.0 (4)

The latest version of JSF 2.0 provides additional advantages:

- new scopes (e.g. conversation scope, custom scopes);
- bookmarkable application states using view parameters;
- better ajax support using standard tags – no need to manually write JavaScript code;
- partial view processing and rendering during ajax requests;
- easier configuration, navigation and resource loading.

```
alt="jsf-sun"
url="/images/jsf-sun.gif">
</h:graphicImage>
```



```
src="/jsf-example/images/jsf-sun.gif"
alt="jsf-sun" />
```

### UIInput

#### inputText

```
<h:inputText id="address"
value="#{jsfexample.address}" />
```

```
<input type="text" name="jsftags:_idl"
value="123 JSF Ave" />
```

#### inputSecret

```
<h:inputSecret redisplay="false"
value="#{jsfexample.password}" />
```

```
<input id="jsftags:password"
type="password"
name="jsftags:password"
value="secret" />
```

#### inputHidden

```
<h:inputHidden id="hidden"
value="userPreference" />
```

No Renderer

```
<input id="jsftags:hidden"
type="hidden"
name="jsftags:hidden"
value="userPreference" />
```

#### inputTextArea

```
<h:inputTextArea id="textArea"
rows="4" cols="7"
value="Text goes here" />
```

```
<textarea id="jsftags:textArea"
name="jsftags:textArea"
cols="5" rows="3">
Text goes here
```

## JSR 329: Portlet 2.0 Bridge for JavaServer™ Faces 1.2 Specification (1)

- JSF technology provides many nice capabilities simplifying the development process, but there are some problems using JSF directly for portlet development
- The most important one is the **difference in lifecycles of portlets and JSF components** – portlets separate **action, event, resource and render requests**, while standard JSF servlet handles them in a **common request processing lifecycle**.
- That is why a bridge between two technologies is needed in order to combine their advantages: **JSR 301: Portlet 1.0 Bridge for JavaServer™ Faces 1.2**, and **JSR 329: Portlet 2.0 Bridge for JavaServer™ Faces 1.2 Specification**.

## JSR 329: Portlet 2.0 Bridge for JavaServer™ Faces 1.2 Specification (2)

- Now we are going to illustrate the details of Java™ portlets development using JSF 1.2 technology (including JSR 301 and JSR 329 open source implementations)
- And provide insight into ongoing initiatives for development of Portlet 2.0 to JSF 2.0 bridge (no specification available yet)
- Examples using open source technologies are demonstrated illustrating the practical value of the technologies ...

portletfaces.org Home

### Welcome to portletfaces.org



The portletfaces.org site hosts open source projects related to portlets and JavaServer™ Faces (JSF) technology and sponsored by [Mimacom](#) for support in Europe and [Triton](#) for support in North America.

### portletfaces tools™

The [portletfaces tools project](#) makes it easier to develop JSF portlets that run within Liferay Portal. [Read more...](#)



### portletfaces bridge™

The [portletfaces bridge](#) project enables JSF 2.0 and ICEfaces 2.0 portlets to run in Portlet 2.0 compliant portlet containers like Liferay 5.x/6.x -- [Read more...](#)



## Resources (1)

- SOAP Version 1.2 Part 0: Primer (Second Edition) W3C Recommendation - <http://www.w3.org/TR/soap12-part0/>
- Web Services Description Language (WSDL) Version 2.0 Part 0: Primer - <http://www.w3.org/TR/2007/REC-wsdl20-primer-20070626/>
- Apache portals - <http://portals.apache.org/>
- OASIS Web Services for Remote Portlets (WSRP) - <http://www.oasis-open.org/committees/wsrp/>
- JSR 286: Java™ Portlet 2.0 API - <http://jcp.org/en/jsr/detail?id=286>



## Resources (2)

- JSR 314: JavaServer Faces 2.0 - <http://www.jcp.org/en/jsr/detail?id=314>
- JSR 329: Portlet 2.0 Bridge for JavaServer™ Faces 1.2 Specification - <http://jcp.org/en/jsr/detail?id=329>



Thanks for Your Attention!

Questions?

