Testable and Maintainable JQuery Composite Applications using Oracle Service Bus and PeopleSoft

A University of Colorado Case Study

University of Colorado

Boulder | Colorado Springs | Denver | Anschutz Medical Campus

Your Presenters

Jason Armbruster

Enterprise Architect & Integrations Manager

Jason.Armbruster@cu.edu

Twitter: @jarmbruster74

linkedin.com/jasonarmbruster

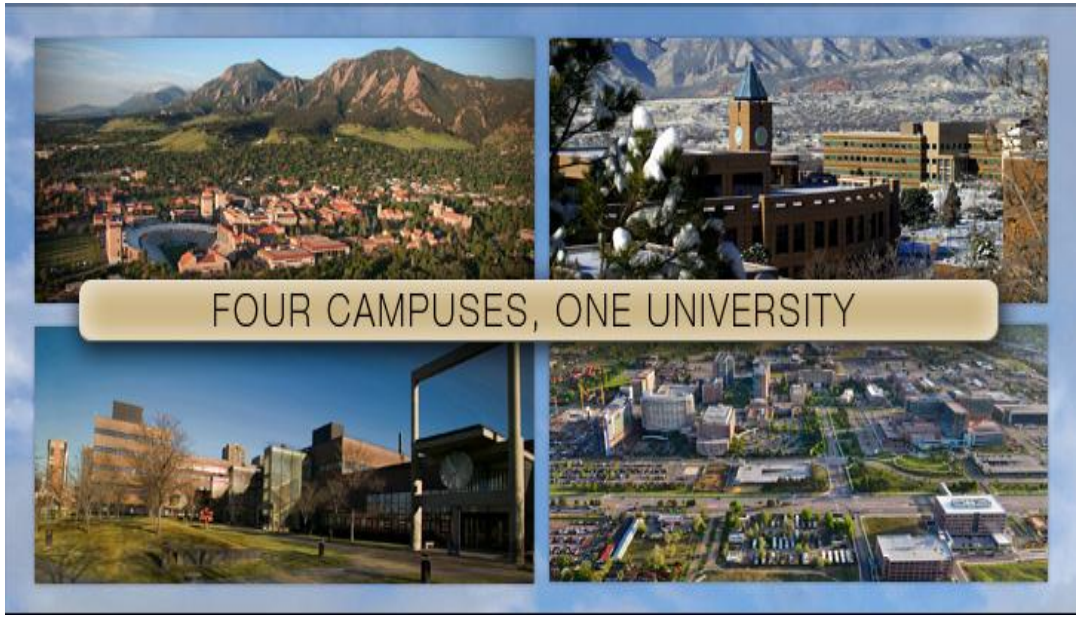
Pamela Song

SOA Technical Lead

Pamela.Song@cu.edu

University of Colorado

- Public Research University
- Four Campuses
- 60,000 Students
- 4,000 Faculty



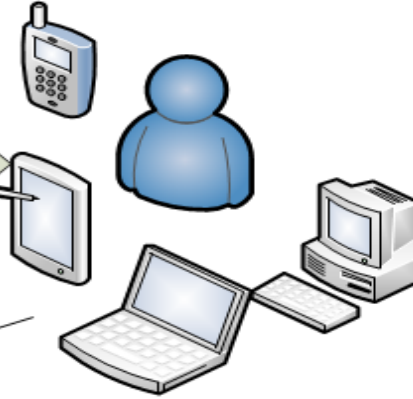
Agenda

- The Problem
- Why JSON/REST? … And Why Not?
- The Tools
- Technical Detail (the guts)
  - Jquery and AJAX
  - JSON & JSONP
  - Oracle Service Bus
  - Transformation XML->JSON, JSON->XML
  - PeopleSoft CI's & Web Services
  - Testing Web Services with SOAPUI
- Putting it all together
- Making it better – future considerations
- Q&A

Our use case

- “Fiscal Certification” - annual requirement for all officers and directors with budget authority to certify that their budget has been managed responsibly and within established policies
- Functional Requirements –
  - Changes every year (and sometimes more often) to the content side of the application (new policies, enhanced help, etc)
  - Underlying business process is pretty static
  - Data requirements are pretty static from year to year
  - Data storage in the Financial GL system allows for best use of the data for reporting purposes
- Non-Functional Requirements
  - Desire for a future mobile version
  - Desire for all the capabilities of a modern web page (collapsable sections, hover text, etc)

- Content Changes
- Design Changes
- HTML5
- Mobile & Tablets
- Rich Internet Applications



REST/JSON

OSB

SOAP/Web Services

- Modifications
- Enhancements
- Bundles
- PeopleTools
- Regulatory Changes

Why JSON/REST…and Why Not?

JSON/REST

- Easy to parse and use within a browser context
- Lightweight
- Supported by both Javascript and Native Mobile frameworks

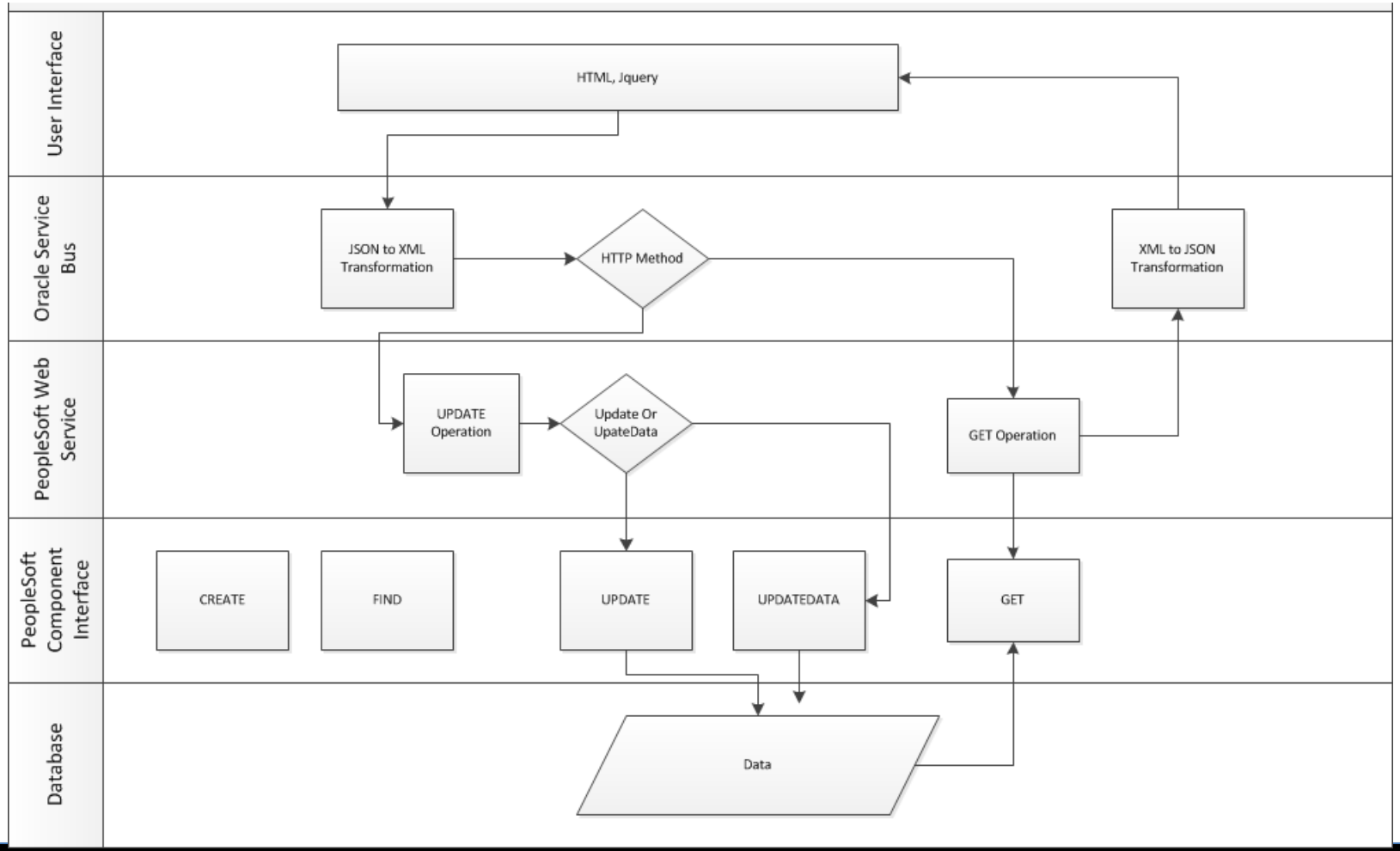
Why Not?

- Schema-less (for now)
  - Implicit Schema
  - No way to validate correctness of responses
  - Errors often detected only at runtime
- Not well supported by PeopleTools (for now)

The Tools

- PeopleTools 8.51 Portal (wrapper)
- Apache Web Server (basic content)
- PeopleTools 8.49 for Finance 8.4 (backend)
- Oracle Service Bus 11g
- Oracle Database 11g

Technical Detail – The Guts



User Interface Jquery and AJAX

- User Interface – HTML and JQuery
- JQuery – Javascript library that simplifies how to traverse HTML documents, handle events, perform animations, and add AJAX, Use Jquery, write less code, get more functionalities
- AJAX = Asynchronous JavaScript and XML. With AJAX, applications can send data to, and retrieve data from, a server asynchronously. Save or load data in the background and display data on the webpage, without reloading the whole page.
  - Note: doesn't really require the data transferred to be XML

Read JSON using Javascript

- Same Domain - Javascript code in the client's browser retrieves JSON data by performing an AJAX request to the same server that served the active page.

```
$.getJSON("https://www.cu.edu/proxyServiceURL?", inputData,
 function(data) {
 // process your data here.
 }
);
```

On OSB side, we will see something like this.

https://www.cu.edu/proxyServiceURL?parameter=parameter_value

Read JSON file in Javascript

- Cross Domain - retrieve the JSON file points to a domain other than the domain that served the page that includes the Javascript code that performs the request.

- For security reasons, the implementations of the Javascript interpreter in those browsers do not allow inter-domain request of json files.

- A solution - JSONP or “JSON with padding” provides a method to request data from a server in a different domain. JSONP is script tag injection, passing the response from the server in to a user specified function

How to use JSONP

- You need to make both request and response to handle JSONP
- In your request, append JSONP callback parameter to the proxy service URL.
- The value of that callback parameter will be generated dynamically (eg. jQuery16406684299709741026_1357613568428)

```
$.getJSON("https://www.cu.edu/proxyServiceURL?callback=?", inputData,
        function(data) {
        // process your data here.
        }
);
```

On OSB side, we will see something like this.
https://www.cu.edu/proxyServiceURL?callback=jQuery16406684299709741026_1357613568428&otherParameter=..
...

How to use JSONP

- In response, pad the response data.
- Capture this callback value in OSB and then pad it to the pure JSON object in the response pipe line using XQuery function.
- Example: fn:concat(fn:string($callback),"(", fn:string($xmlToJSONResult), ")")
- Response JSONP will be:

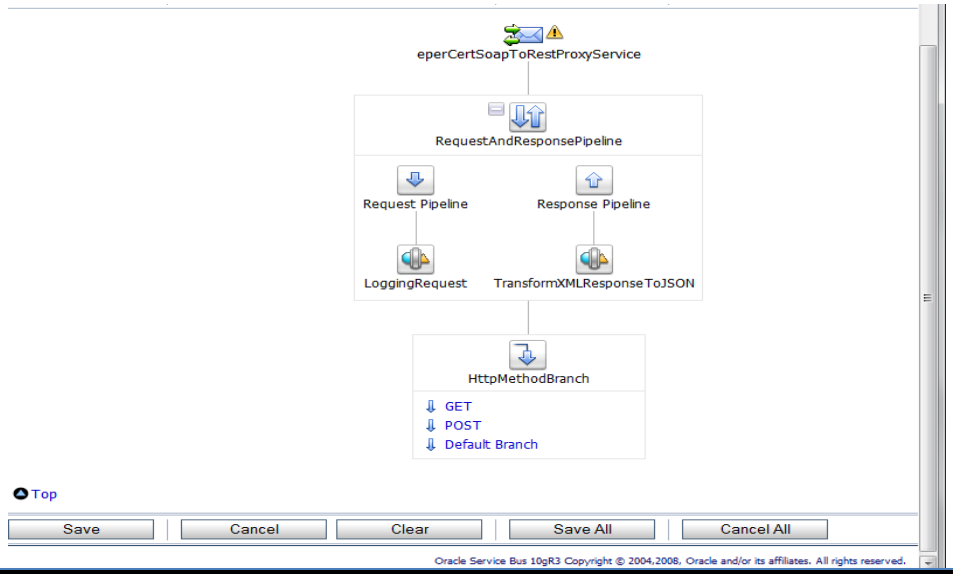
  jQuery16406684299709741026_1357613568428 ({“key1": "999999", “year": 2013, "data": “here is the response data”});
- jQuery takes care of interpreting this result and generate the javascript data structure finally returned.

Security Considerations

- JSONP can only be use for GET requests
- It is more suitable for consumption of public data feeds.
- For POST requests, HTML/jQuery pages need to be in the same domain as the service serving the data.
- In our application we used security token validation and make sure the users have permissions to view the data and they requested.

Oracle Service Bus

- JSON to XML transformation - transform JSON input to XML request for SOAP web service
- Conditional Branching based on the HTTP method
- XML to JSON transformation -transform SOAP web service response XML to JSON



Transformation – XML->JSON

- XML to JSON transformations using XSLT
  - This one works well

    http://code.google.com/p/xml2json-xslt/source/browse/trunk/xml2json.xslt?r=30
- Have not found a robust XSLT/Xquery to do JSON to XML transformation, especially for complicated JSON objects
  - JSON to XML transformation – Use Java Callout

Using Java Callout

These Java libraries are used in our Java class. Either directly imported by our Java class or used by one or more library here.

Dependency libraries

- json-lib-2.2.3-jdk15.jar (http://json-lib.sourceforge.net/) (library for trans forming beans, maps, collections, java arrays and XML to JSON and back again to beans and DynaBeans)
- xmlbeans-2.3.0.jar (http://xmlbeans.apache.org/ - Library for access XML in java friendly way)
- commons-logging-1.1.1.jar (http://commons.apache.org/logging/ -)

- Json-lib depens on these folloiwng java fiels
- commons-beanutils-1.7.0.jar (http://commons.apache.org/beanutils/- provides easy-to-use wrappers around reflection and introspect capabilities)
- commons-collections-3.2.jar (http://commons.apache.org/collections/- interfaces, implementations and utilities for collection handling)
- commons-lang-2.4.jar (http://commons.apache.org/lang/ - helper utilities for the java.lang API)
- commons-logging-1.1.1.jar (http://commons.apache.org/logging/ -)
- ezmorph-1.0.2.jar (http://ezmorph.sourceforge.net/- library for transforming an Object to another Object.)
- xom-1.1.jar (http://www.xom.nu - Dual Streaming/Tree API for Processing XML)

To add to OSB: Upload library jar files to $DOMAIN_DIR/lib directory

The jar files located in this directory will be picked up and added dynamically to the end of the server classpath at server startup.

(Note: Need to restart server after adding new library files)

Transformation JSON to XML

Here is the java callout method that will transform a JSON string to a XML Object

```
public static XmlObject Json2XmlObject(String jsonString)
	{

		JSONObject json = JSONObject.fromObject(jsonString);
		XMLSerializer xmlSerializer = new XMLSerializer();
		xmlSerializer.setTypeHintsEnabled(false);
		xmlSerializer.setForceTopLevelObject(false);
		String xml = xmlSerializer.write( json );
		XmlObject xmlObject = null;
		try
		{
			xmlObject = org.apache.xmlbeans.XmlObject.Factory.parse(xml);
		}
		catch (XmlException ex)
		{
			xmlObject = null;
			Log logger = LogFactory.getLog("edu.cu.integration.JSONToXMLConverter");
			logger.error("Couldn't create xml from Json String. " + ex.getMessage());
		}
		return xmlObject;
	}
```

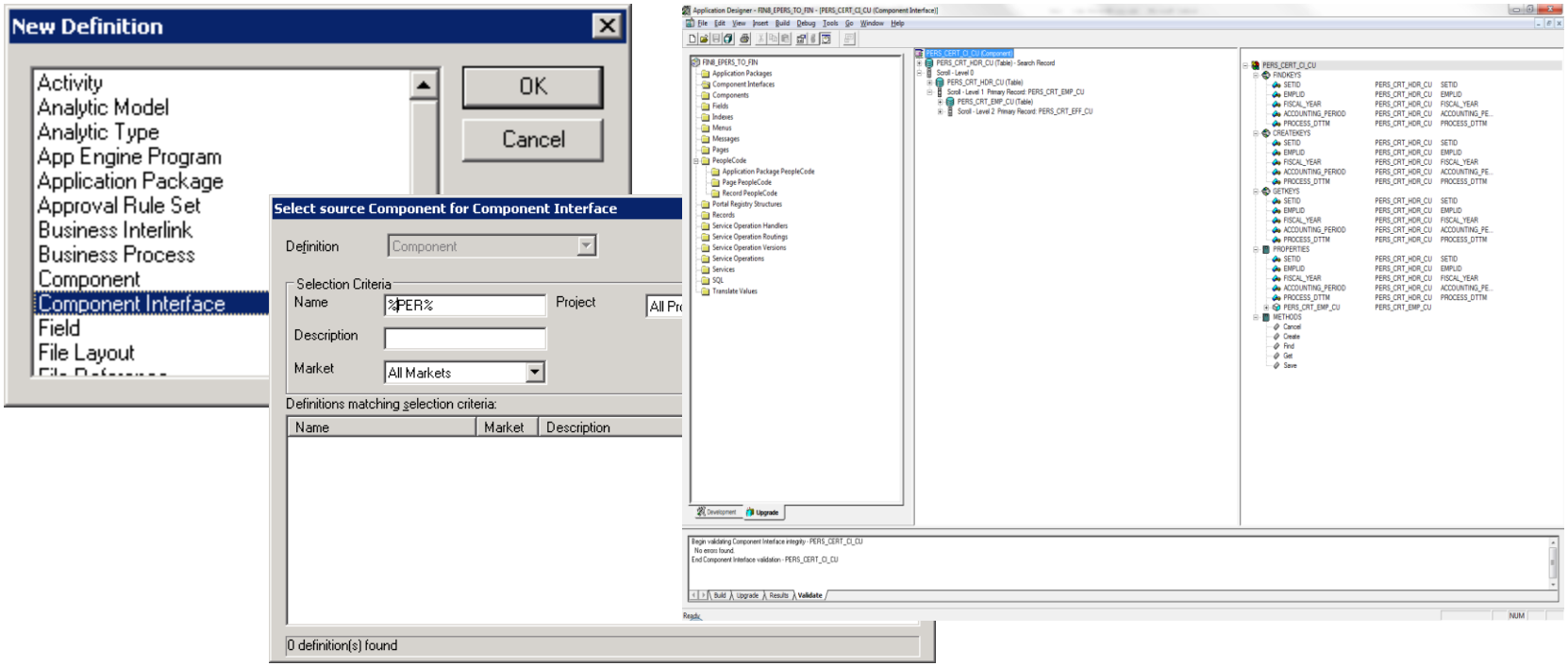
Transform - XML to JSON

- Here is the java callout method that will transform a XML object to a JSON string

```
public static String xmlToJson2(XmlObject xml)
{
        XMLSerializer xmlSerializer = new XMLSerializer();
        xmlSerializer.setSkipNamespaces( true );
        xmlSerializer.setTrimSpaces( true );
        xmlSerializer.setRemoveNamespacePrefixFromElements(true);
        JSON json = xmlSerializer.read( xml.toString() );
         return json.toString();
}
```

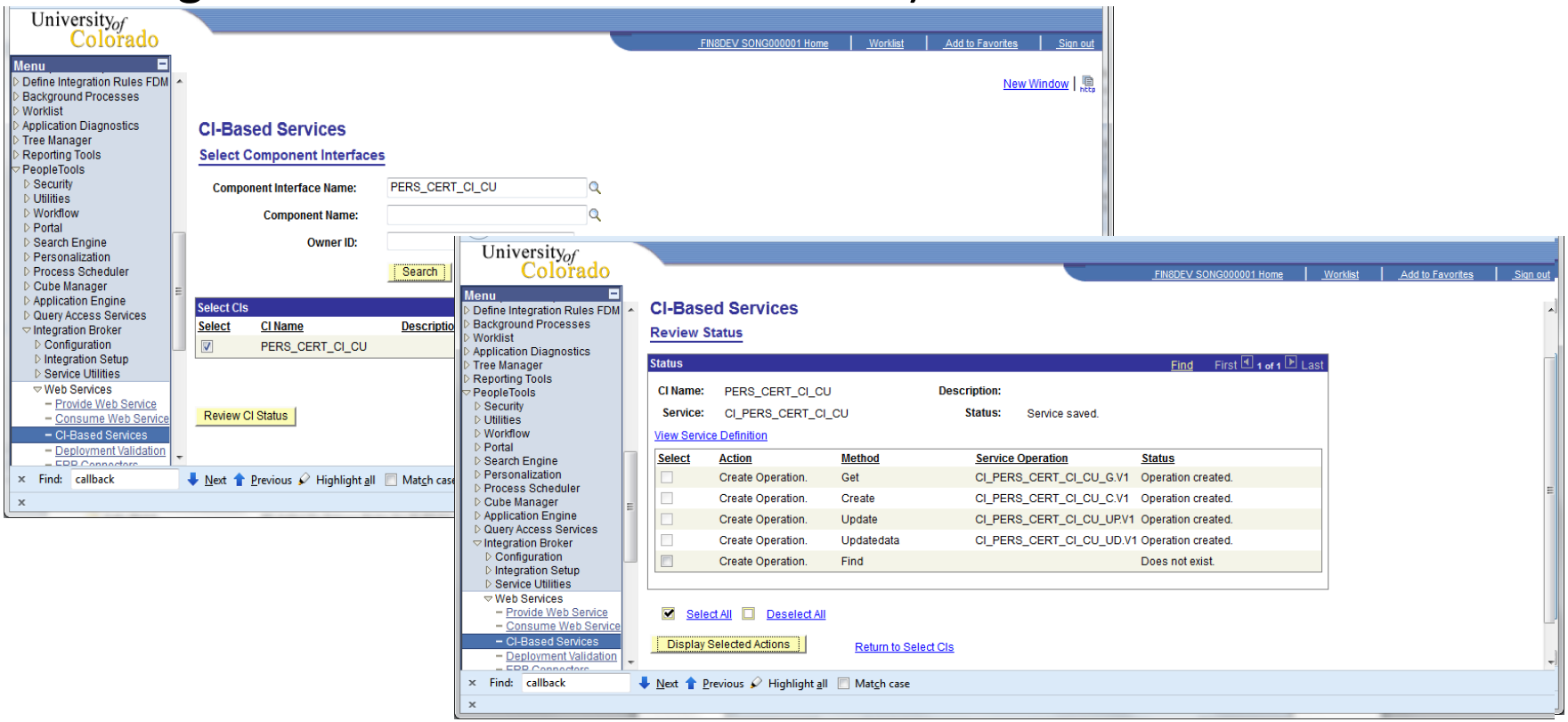
PeopleSoft – CI's and Web Services

- Create Component Interface using the existing Component – leverage the business logic in component.



PeopleSoft – CI's and Web Services

- Expose Component Interface as web service. (PeopleTools->Integration Broker->Web Services->)



PeopleSoft – CI's and Web Services

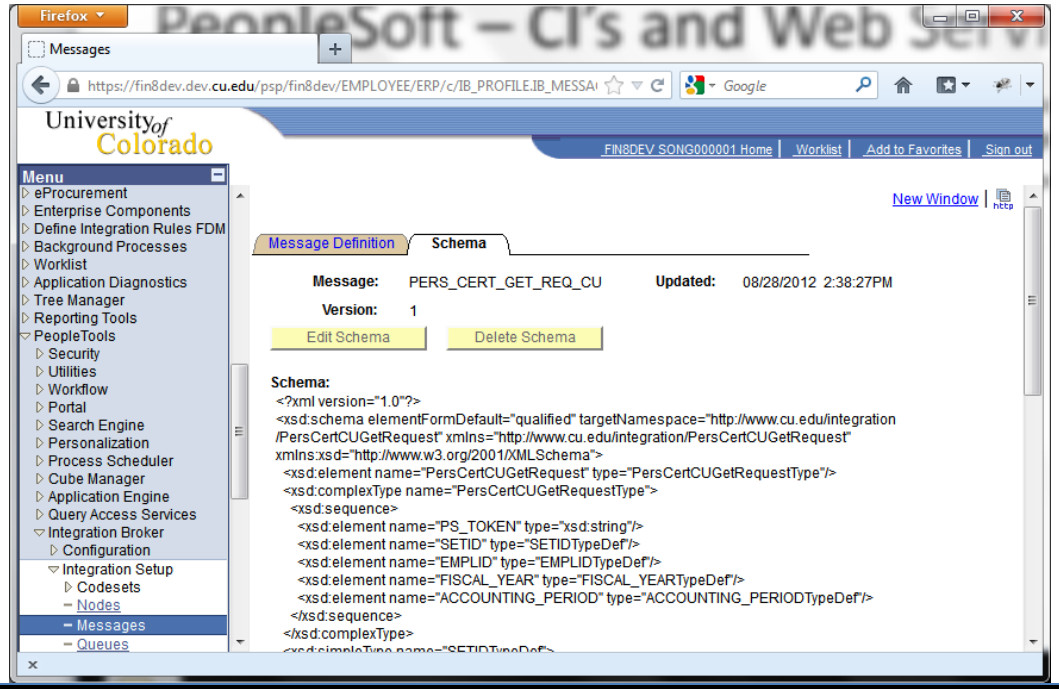
- Business logic needed to determine the correct service operation to call (UPDATE or UPDATEDATA) in our application.
  - UPDATE operation - For Inserting new Values into PeopleSoft.
  - UPDATEDATA operation – For Updating already existing Value in PeopleSoft.

- Oracle Service Bus – Data transformation, service routing only. no business logic (Design Principle)
- Front end – light weight, no business logic.
- Need for Wrapper Web Services based on CI instead of using Exposed CI web service.
- Other Advantages: simplified schema, additional layer of separation.

PeopleSoft – CI's and Web Services

1. Create request and response message schemas (we build new schema based on schema generated by exposing CI as webservice – use xml editor - Oxygen)
2. Create request and response messages – Nonrowset-based message type

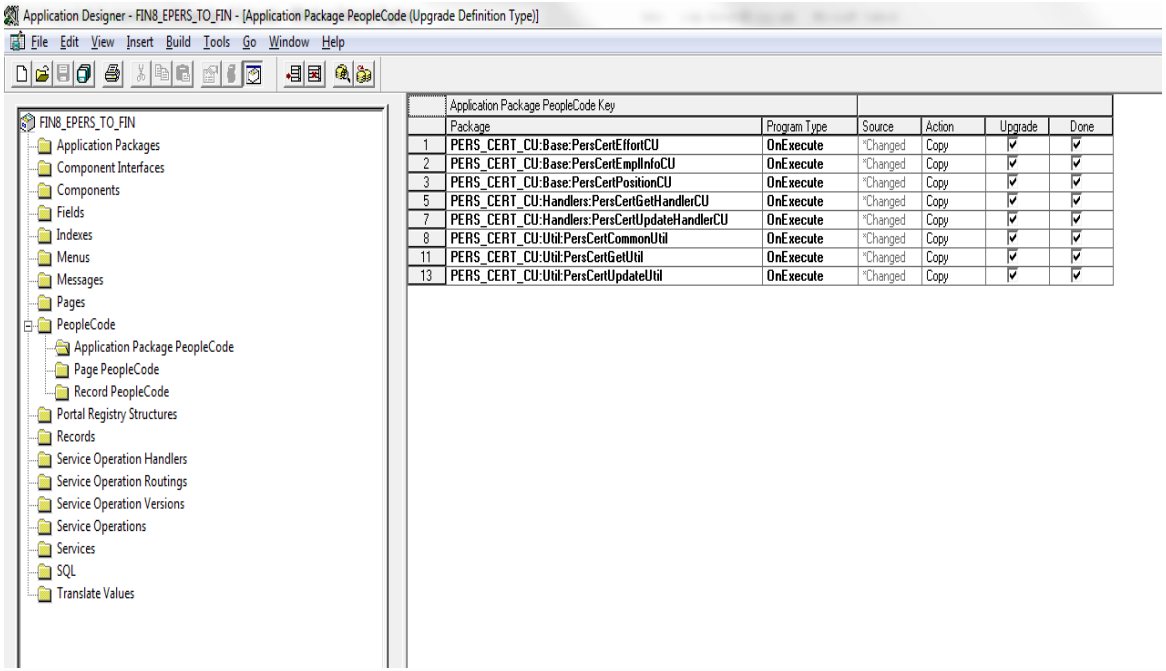
   (PeopleTools->Integration Broker->Integration Setup -> Messages)*



* Non Rowset also simplifies transform into JSON

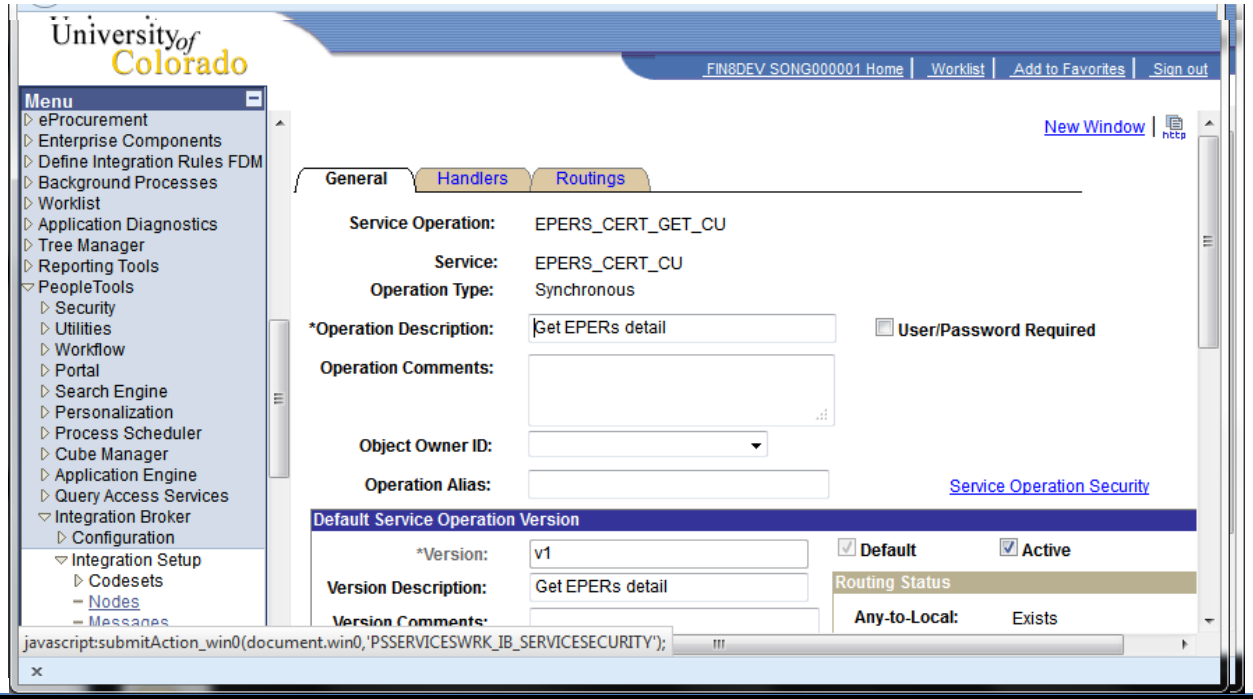
PeopleSoft – CI's and Web Service

3. Develop application packages/classes
4. Create Handler classes that implement PS_PT:Integration:IRequestHandler interface -implement OnRequest and OnError method (Use PeopleTool generated code to access CI)



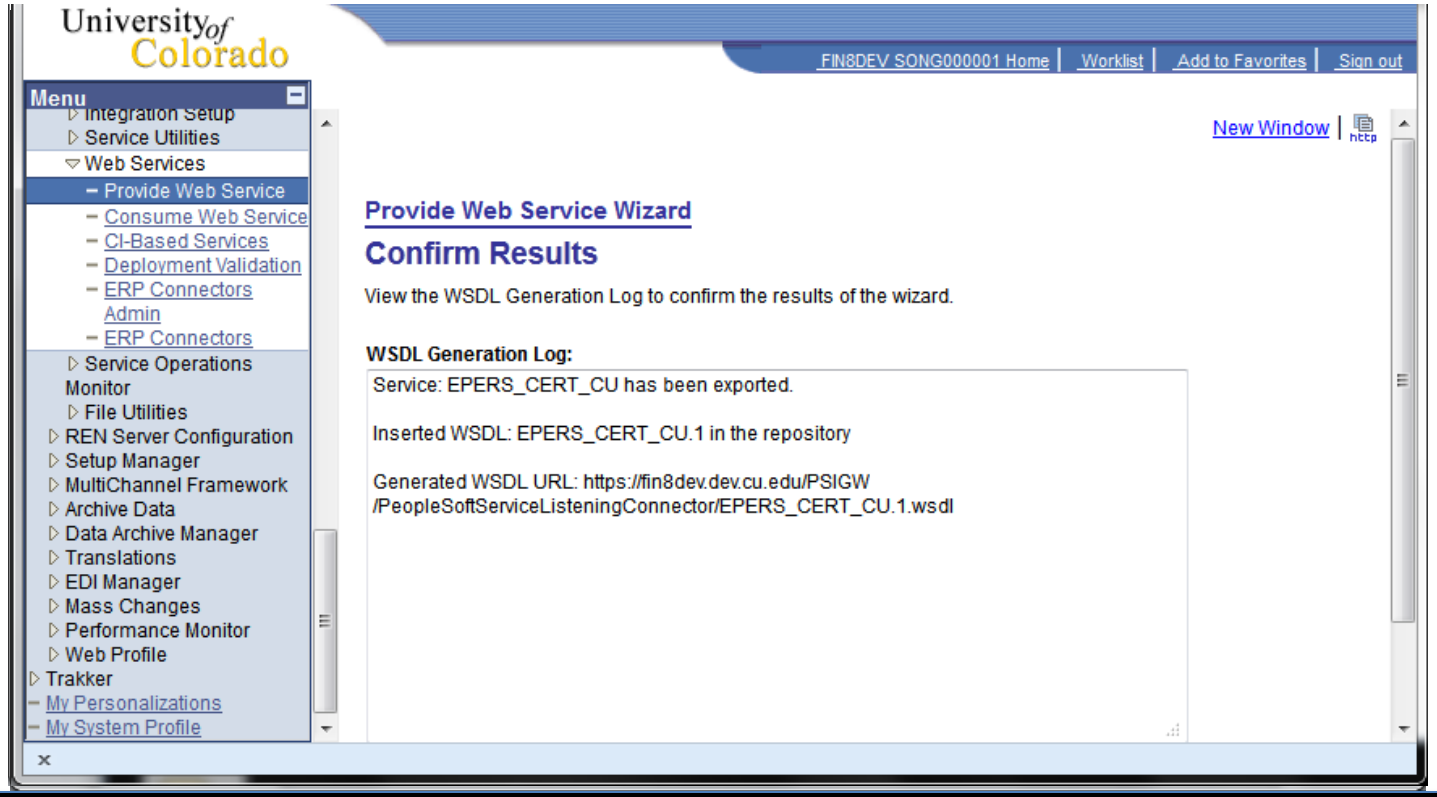
PeopleSoft – CI's and Web Service

5. Add new service (PeopleTools->Integration Broker->Integration Set up->services)
6. Add service operations (use defined req/resp msgs, hander classes, generate any to local routing)
7. Set up Service Operation Security



PeopleSoft – CI's and Web Service

8. Provide web service (PeopleTools->Integration Broker->Web Services->)
9. Test with SOAPUI

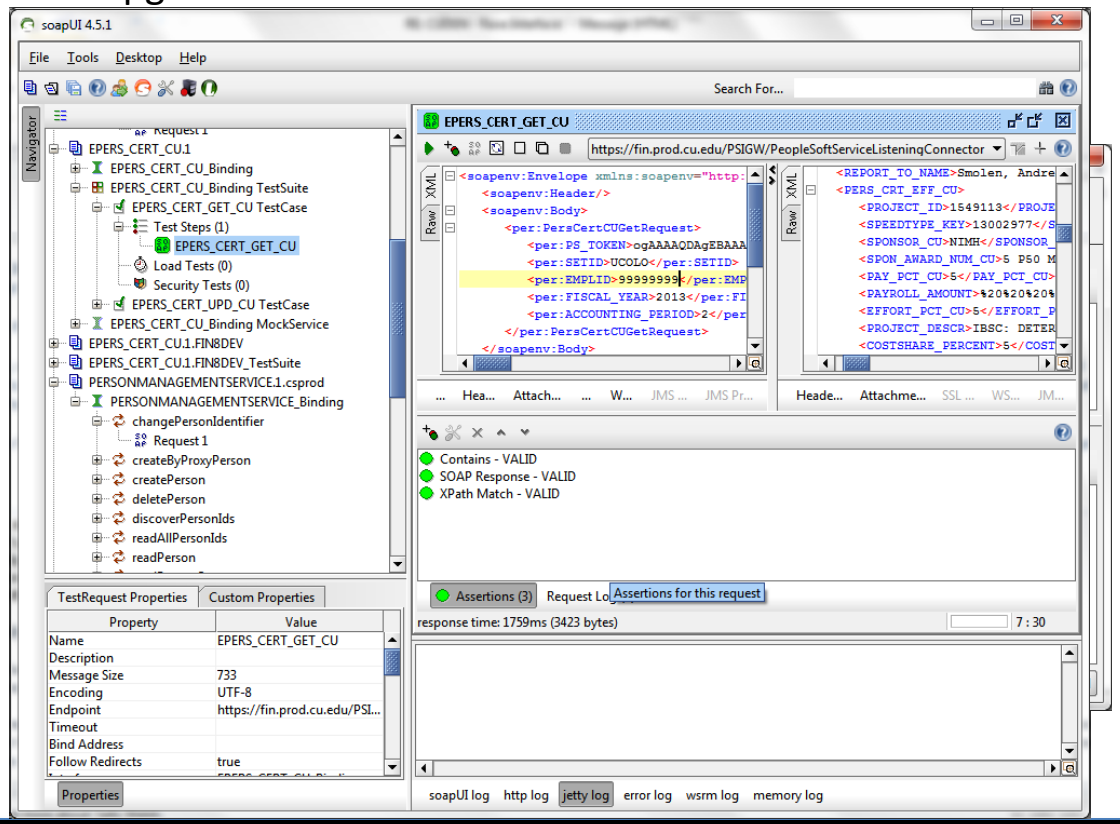


Testing PeopleSoft Web Services with SOAPUI

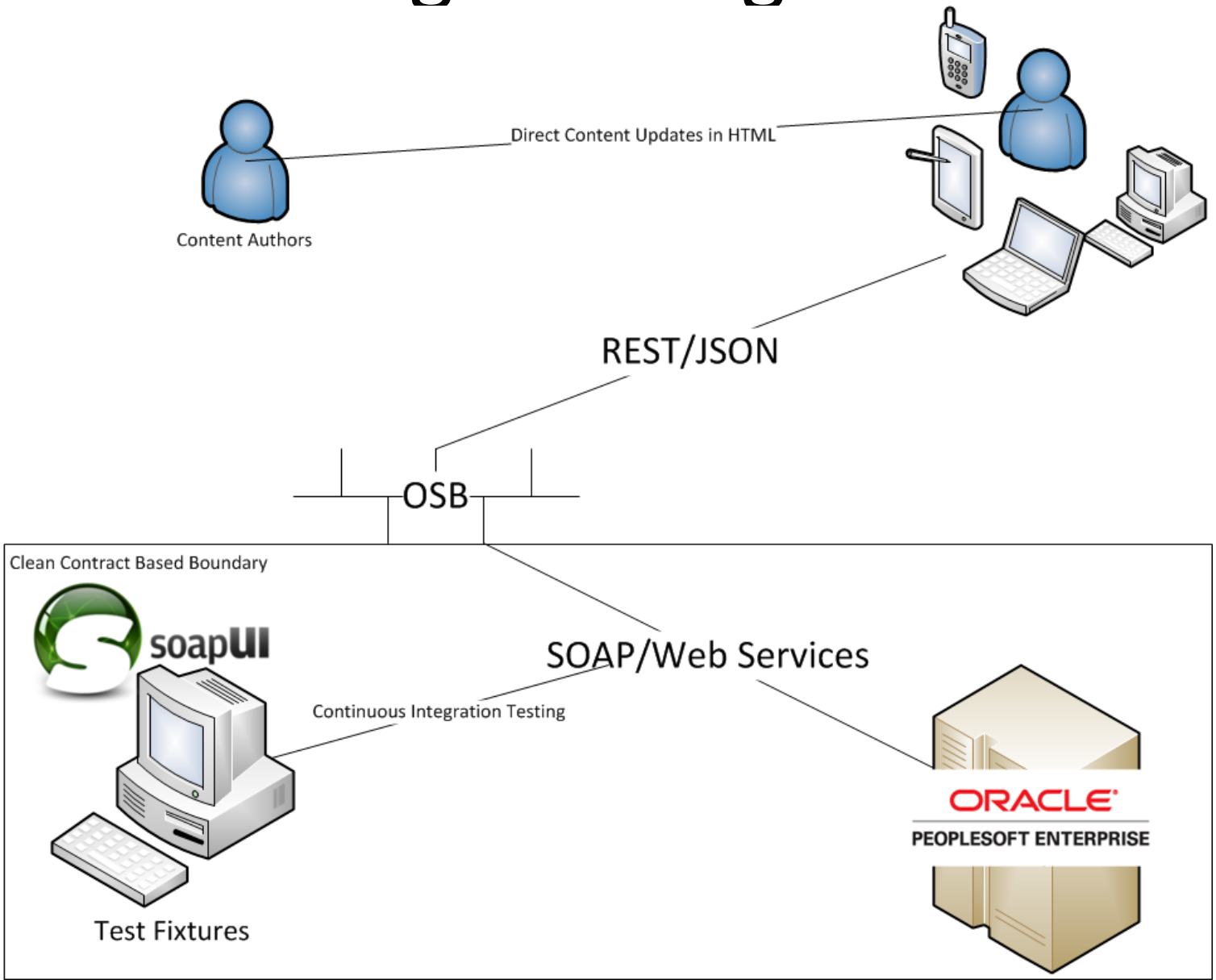
- SOAPUI – An Open Source Functional Testing tool for Web Service Testing (http://www.soapui.org/).
- SOAPUI - Test PeopleSoft web service during development, testing phase.
- SOAPUI – Validate web services after upgrade or maintenance.

Testing PeopleSoft Web Services with SOAPUI

- SOAPUI – An Open Source Functional Testing tool for Web Service Testing (http://www.soapui.org/).
- SOAPUI - Test PeopleSoft web service during development, testing phase.
- SOAPUI – Validate web services after upgrade or maintenance.



Putting it all together



Putting it all together

- Web Services in Peoplesoft are:
  - Well Supported
  - Schema based – easy to validate input/output
  - Maintainable – Impact analysis plus schemas
  - Based on components – no direct DB updates
- REST/JSON in OSB allows:
  - Flexible clients (Jquery, etc)
  - Business user or designer initiated application change
- A composite app that we can actually maintain over time

Future Considerations

- Transition Application to Mobile Web (Jquery Mobile) or Hybrid (PhoneGap)

  http://jquerymobile.com/

  http://phonegap.com/

- PeopleTools 8.52/3 support for REST/JSON

  http://www.oracle.com/us/products/applications/peoplesoft-enterprise/tools-tech/con9189-1872870.pdf

- JSON Schema

  http://json-schema.org/

Presentation Resources

- RMOUG website
- https://github.com/stonaker/cu-presentations
- Folder /RMOUG_Testable_Maintainable_JQuery_Composite_Applications
  - Presentation
  - Detailed Documentation
  - Sample Code
    - Java Transformation

Your Presenters

Jason Armbruster

Enterprise Architect & Integrations Manager

Jason.Armbruster@cu.edu

Twitter: @jarmbruster74

linkedin.com/jasonarmbruster

Pamela Song

SOA Technical Lead

Pamela.Song@cu.edu

Q&A