

Oracle WebCenter Suite
Integrating
Secure Enterprise Search

An Oracle White Paper

January 2007

Oracle WebCenter Suite

Integrating Secure Enterprise Search

INTRODUCTION

As organizations continually reinvent themselves and strive for higher levels of efficiency and productivity, the demands on the information worker are continually increasing. To meet these ever growing demands, the information worker needs a better, more productive work environment. This new environment must be role- and task-focused so that all elements of a task are provided directly in context for the user. Collaboration and communication tools must also be directly integrated into these task-oriented applications because people rarely work in a vacuum and frequently need to work together to complete a given task. Furthermore, the work environment must go beyond the browser to include all of the desktop tools and mobile devices with which the user is familiar. And perhaps most importantly, the information worker must have the ability to tailor and evolve the environment based on their own preferences and the needs of their organization.

Oracle WebCenter Suite injects new capabilities into the standard JavaServer Faces development environment that allow developers to create context-rich applications that satisfy these needs. WebCenter also provides the natural user interaction environment for your SOA applications and allows you to leverage all types of services in creating a better, more effective user experience.

One of these services is Oracle Secure Enterprise Search (SES), Oracle's enterprise-grade search solution that allows you to search all your content sources across the enterprise.

INTEGRATING SECURE ENTERPRISE SEARCH

Oracle WebCenter applications combine the capabilities of Oracle Application Development Framework (ADF) and WebCenter-specific extensions, such as support for portlets and runtime customization. As such, a WebCenter application can be considered a regular Java application and therefore any concept that can be used in a Java application can also be used here.

Oracle Secure Enterprise (SES) exposes its functionality via a Web service that can be integrated into any Java application. This can be done using a Java proxy for the Web service to call its methods or, in the case of a WebCenter application, by leveraging the declarative approach using the Web service Data Control.

You can access the WSDL definition for the SES Web Service using the following URL:

```
http://myServer:myPort/search/query/OracleSearch?WSDL
```

The WSDL describes all the available methods of the Web service and its parameters. Note that the endpoint URL for the Web service is shown as

```
http://myserver:7777/search/query/OracleSearch
```

We will need to make a note of this, as we will need to modify the URL endpoint later in our example application.

INTEGRATING SES USING THE WEB SERVICE DATA CONTROL

The Web service Data Control enables you to access Web services in a declarative manner and subsequently bind operations and results of those Web services to JSF view components, such as tables, buttons, etc.

In this example, we want to add a search box with a button that submits a search to SES and returns the result in an ADF table.

Creating the SES Data Control

In JDeveloper, begin by opening the *New Gallery*. Under *Business Tier > Web Services* locate the Web Service Data Control. You might need to expand your technology scope by selecting “All Technologies” from the drop-down list at the top.

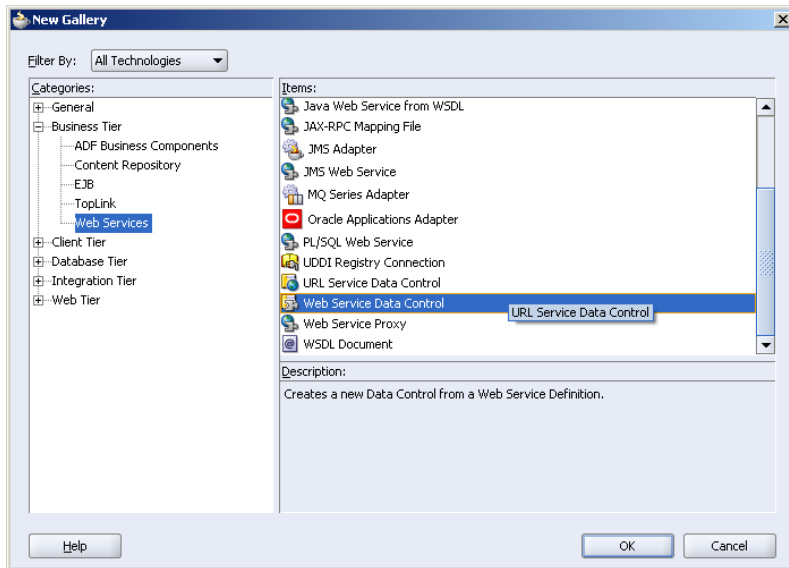


Figure 1 - New Gallery

Click OK to invoke the “Create Web Service Data Control” wizard. In Step 1 of the wizard, we specify a name for the Data Control and the URL for the WSDL that describes the service, that is, the SES Web service.

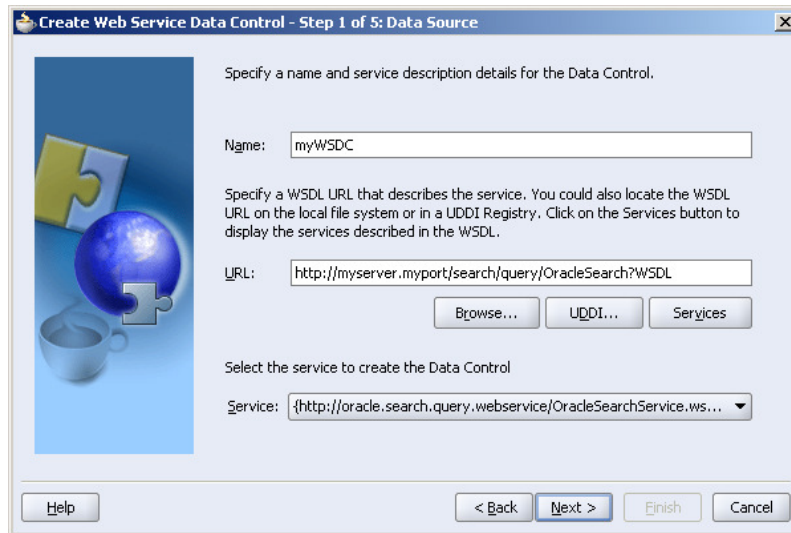


Figure 2 - Web Service Data Control Wizard

Next, we click the “Services” button. The wizard will query the service and populate the “Service” field. In the case of SES, there is only one service described by this WSDL and therefore, no further action needs to be taken. Click “Next >” to proceed to the next page of the wizard.

In Step 2, the “Data Control Operations” page, we select the methods of the Web service we want to expose through the Data Control. For this example, all we want to expose is “OracleSimpleSearch”, so we select it and shuttle it to the right side.

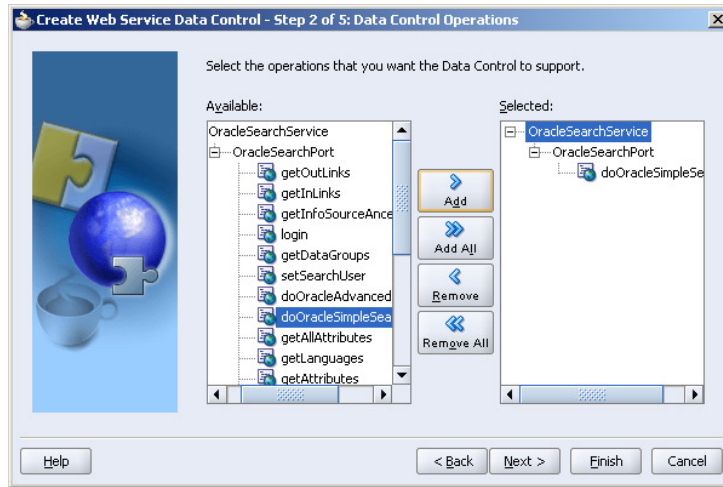


Figure 3 - Data Control Operations Dialog

The remaining steps of the wizard are not relevant in this example, so we can click “Finish” to complete the wizard and create the Data Control.

Correcting the Endpoint URL

Recall that we need to modify the URL endpoint for our Data Control. Currently, the URL points to the default one in the WSDL. We can modify the URL endpoint by modifying the Data Control definition.

In your project, expand the “Application Source” node and locate a package with the same name as your project. In the package, you should find a file called `DataControls.cpx`. When you select it, the Structure Pane will show the Data Controls described by this file, including the Data Control we just created.

Right-click the Data Control in the Structure Pane and select “Edit Web Service Connection ...” from the context menu.

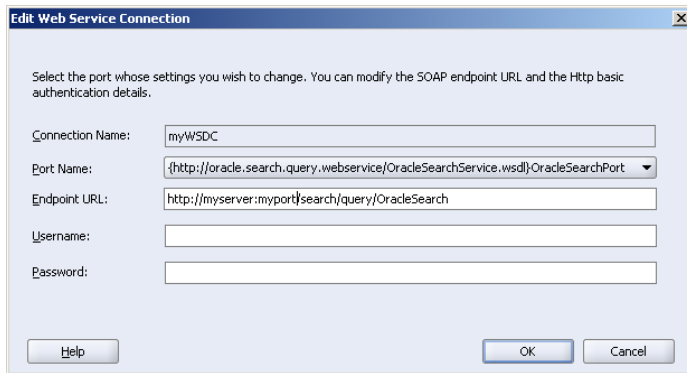


Figure 4 - Edit Web Service Connection Dialog

In the “Endpoint URL” field, specify the actual endpoint URL for the SES Web service. Click OK to save the changes.

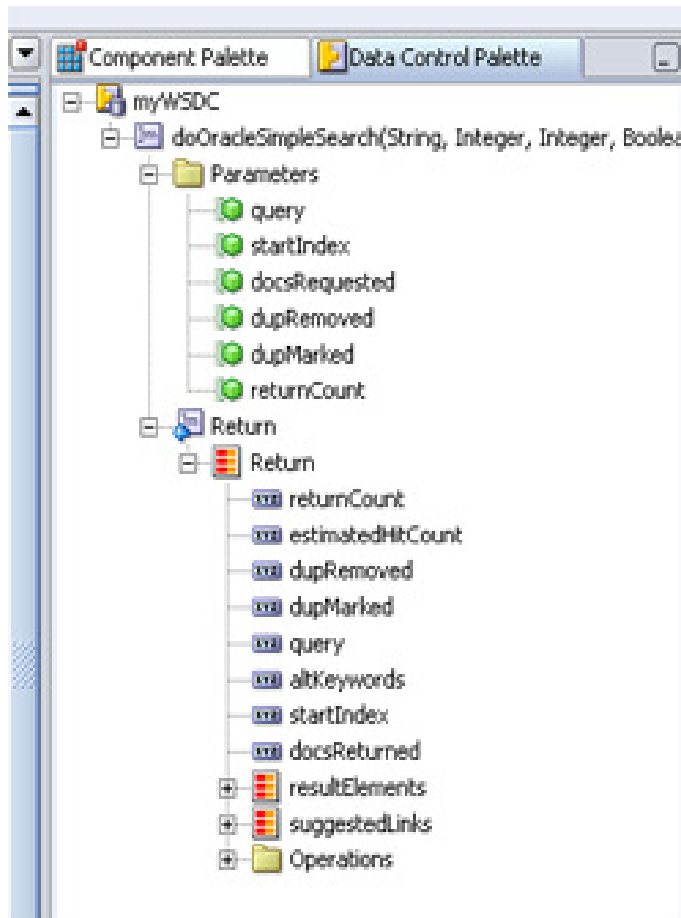


Figure 5 - Data Control Palette

Let's take a look at the Data Control and its parameters and methods.

The service accepts the following input parameters:

- query: the string to search for
- startIndex: the first record of the result set that should be returned
- docsRequested: the number of rows in the result set that should be returned
- dupRemoved and dupMarked: how to handle duplicates
- returnCount: indicates to the service whether the total number of results should be calculated

The return node contains several attributes, two collections that represent the result set of the search, as well as a list of suggestions based on the search. For more information about the returned attributes and the concept behind the suggestions, please consult the documentation for Secure Enterprise Search.

In our example, the important elements are the “OracleSimpleSearch” operation, the “query” parameter, and the “resultElements” collection.

Creating the Search Form

Now it is time to create the search form and the table for the result set. First, you need to create a JSPX page, if you have not done that already. Next, select the “OracleSimpleSearch” node in the Data Control Palette and drop it onto your page. Select *Create > Methods > ADF Command Button* from the pop-up menu. The Action Binding Editor displays.

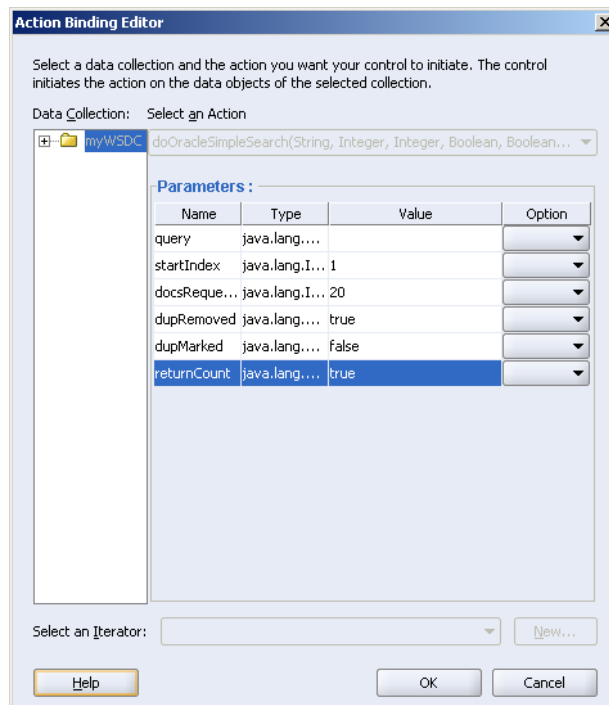


Figure 6 - Action Binding Editor for Command Button

In the Action Binding Editor, we specify the necessary values for the service parameters, as seen in Figure 6. We can leave the parameter query empty as we are going to create an inputText component for it.

Now drag and drop the “query” parameter onto your page and select *Create > Texts > ADF Input Text w/ Label* from the pop-up menu.

Finally, select the “ResultElements” collection and drop it onto the page. Select *Create > Tables > ADF Read-only Table* from the pop-up menu.

Due to a known issue with the Web Service Data Control, we need to remove the reference to the “last modified” column. Simply select the row in the “Edit Table Columns” dialog and click the “Delete” button.

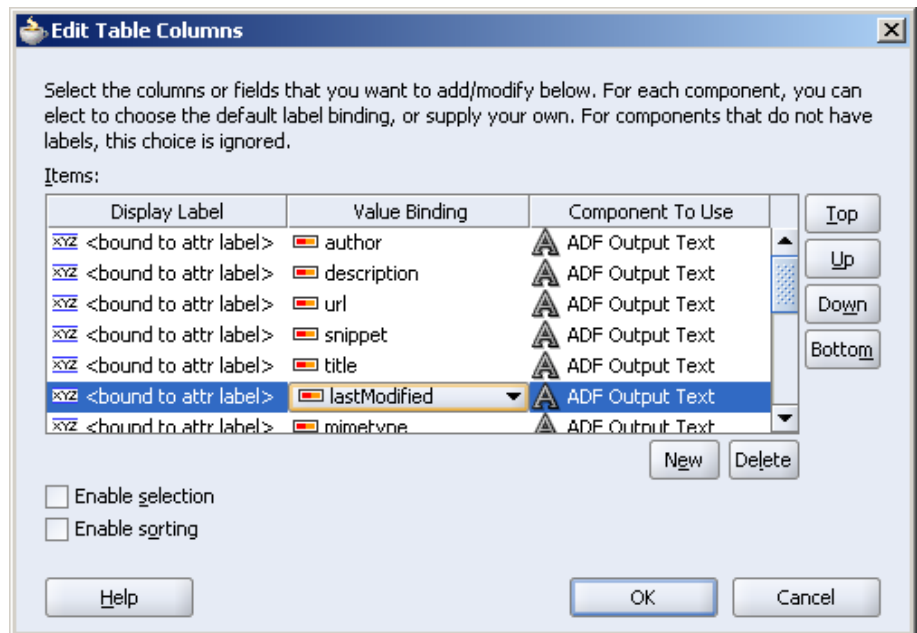


Figure 7 - Edit Table Columns Dialog

The two remaining tasks that need to be done are in the Page Definition. Right-click the page and select “Goto Page Definition” from the context menu.

First, we need to specify a default value for the search term. This can be anything you want.

```
<variable Type="java.lang.String"
  Name="doOracleSimpleSearch_query"
  IsQueryable="false"
  DefaultValue="oracle"/>

```

Second, we need to remove the reference to “LastModified” from the table model in the Page Definition.

Look for the tag

```
<Item Value="lastModified"/>
```

and delete it.

Now we are ready to test the search. Go back to the JSPX page, right-click the page, and select “Run” from the context menu.

ENHANCING THE SEARCH EXPERIENCE

There are a couple of simple things you can do to enhance the experience of your search page.

If you look closer at the “Snippet” column, you will discover that SES adds `...` tags around the occurrences of the search term. Unfortunately, the normal ADF `OutputText` view component escapes those tags such that the HTML does not take effect but instead, the tags are displayed in the browser as text. To change that, you can convert the `OutputText` view component into an `OutputFormatted` component that will then render the search term in bold.

Simply return to JDeveloper and right-click the view component that renders `row.snippet`. Choose “Convert” from the context menu to invoke the Convert dialog window.

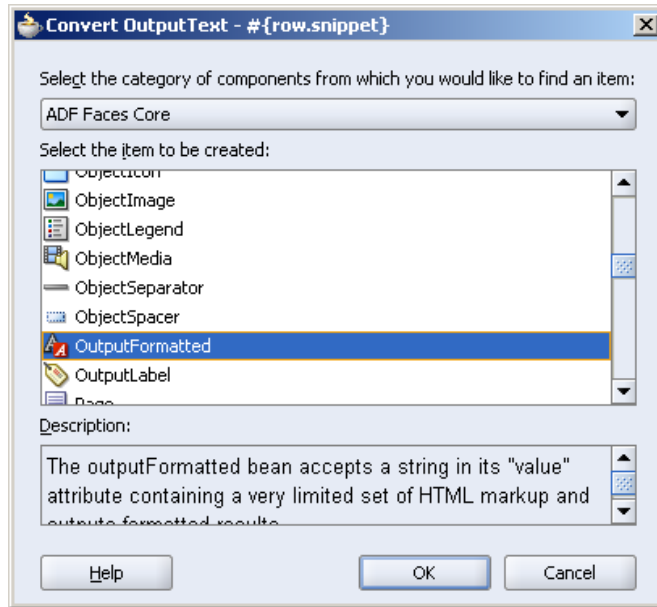


Figure 8 - Convert Dialog

Make sure “ADF Core” is selected in the drop-down list and look for `OutputFormatted`. If you do not see “ADF Core” in the drop-down list, exit the dialog by clicking `Cancel` and make sure that you selected the view component and not the surrounding table cell. The easiest way to ensure this is to use the Structure Pane, not the visual editor, to make your selection.

Once you click `OK`, the component will be converted. The next time you run the page, the search term will appear in bold.

	snippet
	Yahoo! - Yahoo! Bangalore Yahoo! R&D India The Yahoo R&D center's core activity includes developing innovative technologies and global product platforms... ? Yahoo
	maintained by Roger Ford (email: roger.ford@ oracle .com) Recent access count

Figure 9 - Highlighted Search Terms

Another way to improve the experience is to make the page title a hyperlink. You can do this by simply converting the OutputText component rendering “row.title” into a goLink. Right-click the component, select “Convert” from the context menu, and look for “GoLink” in the list of “ADF Core” components. You will have to confirm the conversion this time, since not all attributes of OutputText can be carried over to GoLink, but in our example, it does not matter.

Using the Structure Pane, locate the GoLink that you just created, right-click it, and select “Properties” from the context menu.

In the Properties window, provide the binding for Text and Destination of the GoLink.



Figure 10 - GoLink Property Dialog

If you are not sure what the exact attribute name is, click the “Bind” button to bring up the Binding Editor. Expand the JSP Objects node and look for “row”. Expand “row” and you will see all valid attributes. Double-clicking one attribute will create the necessary expression on the right hand side of the editor. Click “OK”.

	title	mim
Yahoo! Bangalore	Yahoo!	text/
India The Yahoo	Bangalore	
's core activity		
veloping		
echnologies and		
platform		

Figure 11 - Title Column as Hyperlink

Run the page again. When you click the link, you will now navigate to the document that contained the search term.

LEVERAGING A CUSTOM DATA CONTROL

There is another method for integrating SES in your WebCenter application. When you want to create more complex scenarios or if you require authentication, you can leverage the Data Control mechanism that ADF provides and create a Data Control based on a custom Java class.

Create your custom Java code

In your project, you create the Java class representing the business logic you want to expose. You could, for example, perform login or other operations before or after the actual search execution.

The class would contain one or more methods returning `OracleSearchResult`, which represents the result of the search operation.

In our simple example below, the only method that exists accepts one input parameter, the search term (`sSearchTerm`).

```
public class testSES {
    public static OracleSearchResult doSearch(String sSearchTerm) {
        OracleSearchService searchSvc;
        searchSvc = new OracleSearchService();
        OracleSearchResult searchResult;
        searchResult = new OracleSearchResult();

        searchSvc.setSoapURL("http://.../search/query/OracleSearch");
        try {
            searchSvc.login("", "");
        } catch (Exception e) {
            System.out.println(e);
        }
        try {
            searchResult =
                searchSvc.doOracleSimpleSearch(sSearchTerm,
                                                1,20, false, false,false);
        } catch (Exception e) {
            System.out.println(e);
        }
        return searchResult;
    }
}
```

Now that we have created the class with its methods, it is time to create our Data Control for this class. To do so, simply right-click the class in the Applications Navigator and select “Create Data Control” from the context menu.

JDeveloper will create the necessary information for you and when you display the Data Control Palette, you will find the new Data Control listed there.

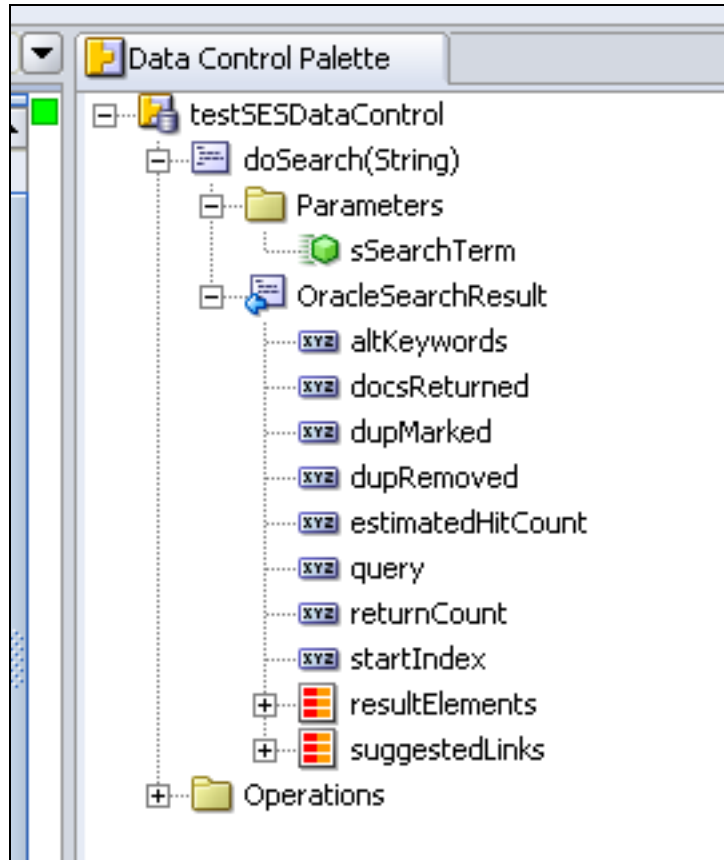


Figure 12 - Data Control Structure created

When you look at the Data Control and expand it to see its structure, you will find that it resembles the structure we observed with the Web service Data Control in the previous section. Under the Parameters node, however, you'll see only one item— sSearchTerm— which is the one we defined as the input parameter for our doSearch method in our Java class. The remaining steps are exactly the same as with the Web service Data Control.

Drag and drop the method to your page and select *Create > Method > ADF Command Button* from the pop-up menu. Then drag and drop “sSearchTerm” onto your page and select *Create > Texts > Input Text w/ label* from the context menu. Finally, drag and drop “resultElements” onto your page, and select *Create > Tables > ADF Read-only table*.

The biggest difference here is that instead of calling the search Web service directly, we are calling our Java class. Everything defined in the class will be performed which, in this example, is logging into Secure Enterprise Search and thereby performing a secure search on behalf of a particular user.

CONCLUSION

Oracle Secure Enterprise Search, one of the services of Oracle WebCenter Suite, can be easily integrated into any WebCenter application by either implementing a Web service proxy-based approach or by using the declarative methods of Data Controls.

With a few easy steps, you can integrate powerful search capabilities into your application, significantly improving your users' experience. ADF provides two possible approaches: using the generic Web service Data Control, or creating a custom Data Control from a Java class.



Oracle WebCenter Suite – Integrating Secure Enterprise Search
January 2007
Author: Philipp Weckerle

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Copyright © 2007, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission. Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.