Novell exteNd Composer™

5.2.1

www.novell.com

CONNECT FOR PEOPLESOFT* USER'S GUIDE

Novell, Inc.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.

www.novell.com

Novell exteNd Composer 5.2.1 *Connect for PeopleSoft User's Guide*
March 2005

Online Documentation: To access the online documemntation for this and other Novell products, and to get updates, see www.novell.com/documentation.

Indiana University Extreme! Lab Software License

JDOM.JAR

Phaos

W3C

Contents

About This Book

Purpose

This guide describes how to use the Novell exteNd Composer Connect for PeopleSoft®. This product has design-time as well as runtime executables and uses Java Connector Architecture technology to provide integration capability.

Audience

This book is for developers and systems integrators who are planning to use Novell exteNd Composer to develop PeopleSoft®-aware services and components.

Prerequisites

This book assumes prior familiarity with exteNd Composer's work environment and deployment options. Some familiarity with PeopleSoft® is also assumed. Familiarity with Java Connector Architecture is helpful but not required.

Software Versions

This guide assumes that you are using Novell exteNd Composer (Enterprise or Professional) version 5.2 (or higher) and that you are deploying your applications to a J2EE 1.3 application server or equivalent.

The Connect for PeopleSoft® is designed to work with PeopleSoft® 8.1 or 8.4.

NOTE: You will need to obtain **psjoa.jar** from PeopleSoft in order to complete the installation of your software. This file is proprietary to PeopleSoft and is not shipped nor installed as part of the Novell exteNd Suite. If this file is not already on your file system as part of your PeopleSoft installation, contact your PeopleSoft representative to obtain it.

Additional documentation

For the complete set of Novell exteNd documentation, see the Novell Documentation Web Site (http://www.novell.com/documentation-index/index.jsp).

1 Welcome to Novell exteNd Composer Connect for PeopleSoft®

Welcome to the *Novell exteNd Composer Connect for PeopleSoft® User's Guide*. This Guide is a companion to the *Novell exteNd Composer User's Guide*, which details how to use all the features of Composer except for the Connect Component Editors. So, if you haven't looked at the *Composer User's Guide* yet, please familiarize yourself with it before using this Guide.

Novell exteNd Composer provides separate Component Editors for each Connect, including the Connect for PeopleSoft®. The special features of each component editor are described in separate Guides like this one.

If you have been using exteNd Composer and are familiar with the core component editor (the XML Map Component Editor), then this Guide should be enough to get you started with the PeopleSoft Component Editor.

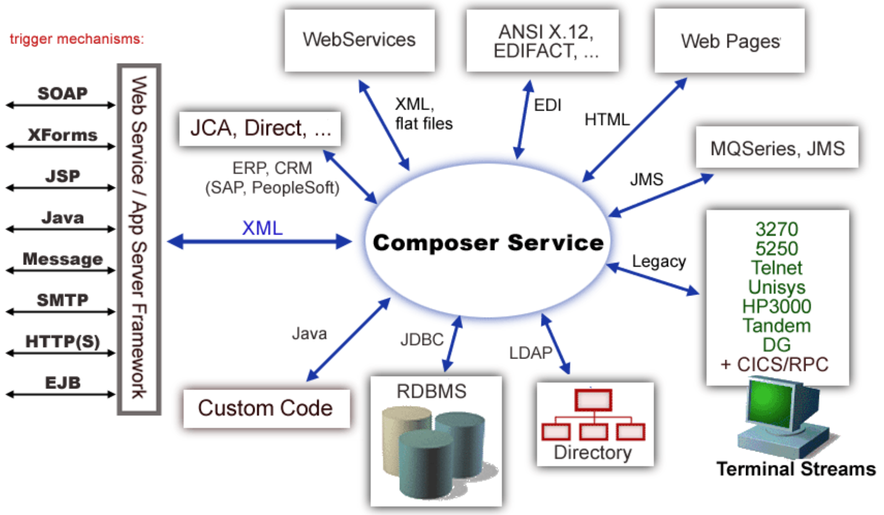
NOTE: To be successful with this Component Editor, you should already be familiar with PeopleSoft and basic XML integration concepts.

About Novell exteNd Composer™

Novell exteNd Composer is the XML integration-broker piece of the Novell exteNd suite. It encompasses a set of design tools for building XML integration applications and web services, plus a runtime engine that enables execution and administration of the services you build. The applications and services you build with Composer can be deployed to any popular J2EE application server or servlet container. (Supported servers include JBoss, IBM WebSphere and BEA WebLogic in addition to the Novell exteNd application server. Tomcat is also supported. Be sure to consult the Novell web site for latest platform-support information.)

At the core of Composer is a robust XML transformation engine capable of performing a wide range of data transformations, including joining of multiple documents, decomposition of documents, and/or creation of entirely new documents on the fly. The underlying enabling technologies include not only XSLT but XPath, ECMAScript, and Java. Composer's design environment offers a rich, intuitive graphical user interface, making it possible for you to specify XML transformations and mappings visually, using wizards, dialogs, and drag-and-drop gestures. You never have to write raw XSL or Java code.

Composer supports numerous kinds of data-source connectivity, through individual adapters called Connects. Using the functionality exposed in the various Connects, you can design EAI applications and/or web services that pull data in from or push data out to a variety of different kinds of back-end systems, using a variety of transport protocols and technologies, ranging from 3270 and 5250 terminal data streams to Telnet, HP3000, Unisys T27 (and UTS), Tandem, Data General, etc., in addition to HTML screen-scraping, JMS messaging, and CICS RPC transactions. You can also take advantage of JDBC, LDAP, and other mechanisms to reach back-end data repositories and systems that might or might not natively understand XML. Composer Connects allow you to connect to these systems inobtrusively, so as to marshall non-XML data into XML form or vice versa without any need to modify host-system setups or code.



In addition to legacy data-stream and protocol-specific Connects, Composer has Connects for ERP and CRM systems, including Baan, PeopleSoft, SAP, Siebel, Lawson, JD Edwards, and Oracle Financials. As with other Connect solutions, the ERP and CRM Connects are fully integrated into Composer's design-time environment so that you can use intuitive visual tools to create powerful custom integration solutions, eliminating the need to write Java code or edit raw XML or schemas by hand. You can also test the components you build against live PeopleSoft connections, using the design environment's animation facility (step-through debugger). As part of the design and debug process, your PeopleSoft-aware components can call other Composer Components (such as XML Map Components, JDBC Components, etc.) and make use of any of the core actions that Composer defines (such as Map, Function, Log, and other actions).

About the Composer Connect for PeopleSoft®

The Novell exteNd Composer Connect for PeopleSoft® allows you to leverage PeopleSoft's Component Interface technology to build powerful XML-based integration solutions.

The Composer Connect for PeopleSoft® enables you to access a *PeopleSoft component* using the Component Interface mechanism provided by PeopleSoft. (Here, "PeopleSoft component" means a component defined by the PeopleSoft API, not a Composer xObject.)

You do not need to manually generate or install PeopleSoft XML schemas in order to use the Composer Connect for PeopleSoft®. The Connect will generate schemas for you, as needed, automatically.

You also don't have to take any steps to preinstall RARs (resource adapter archives, defined by the Java Connector Architecture) on the target application server ahead of time. Composer handles RAR deployment for you automatically when you deploy any Composer-built service that utilizes the Composer Connect for PeopleSoft®.

NOTE: Some one-time setup and configuration steps *are* required in order to use the Composer Connect for PeopleSoft®. The steps in question are described in detail under "Setup and Configuration" in the next chapter.

About J2EE Connector Architecture

The J2EE™ Connector Architecture defines a standard architecture for connecting elements of the J2EE platform to a heterogeneous Enterprise Information System (EIS). Examples of EIS components include (typically) Enterprise Resource Planning (ERP) systems, Customer Relationship Management (CRM), Supply Chain Management (SCM), mainframe transaction processing, database systems, and legacy applications that are not written in the Java programming language. By defining a common set of scalable, secure, transactional mechanisms reachable via a standard set of APIs, J2EE™ Connector Architecture enables the integration of an EIS with an application server and enterprise applications.

The J2EE Connector Architecture permits an EIS vendor to provide a standard resource adapter for its EIS. The resource adapter plugs into an application server, providing connectivity to an EIS, and integrating it with the rest of the enterprise. If an application server vendor has extended its system to support J2EE™ Connector Architecture, it is assured of seamless connectivity to multiple Enterprise Information Systems.

Before J2EE™ Connector Architecture, most EIS vendors offered vendor-specific architectures to provide connectivity between applications and their software; each program interacting with an EIS needed to be hand-tooled by someone with a detailed knowledge of the peculiarities of the target EIS. Custom software to do this across multiple systems was time consuming to develop, debug, and maintain.

By providing a standard set of APIs and contracts for managing connectivity, exposing EIS APIs, and using application-server services (like transaction control and connection pooling), J2EE™ Connector Architecture greatly reduces the need for custom programming. Developers can focus on business logic rather than connectivity and transaction-related logic and a variety of "plumbing issues."

How J2EE™ Connector Architecture Works

The "major participants" in J2EE™ Connector Architecture include these components:

- Application server
- Resource adapter (RAR)
- Application

The application server is not strictly required. Certain services like connection pooling and transaction control will not be available in a "server" that is just a servlet container. But J2EE™ Connector Architecture resource adapters can still operate.

The RAR represents the interests of the underlying EIS.

The *application* interacts with the *resource adapter* using what J2EE™ Connector Architecture calls standard contracts. Standard contracts define what interactions are to take place and how they are exposed. The contract between the application and the resource adapter is called the Common Client Interface (CCI). The resource adapter, in turn, interacts with the application server under the Service Provider Interface (SPI), which defines how the management of resource adapter interactions occurs. The aspects of this include:

- Connectivity management
- Transaction demarcation
- Event listening (listeners can receive notification of significant events; for example, a connection failure)
- Pooling of connections and other resources

In the normal course of events, the application uses a *naming service* to locate the appropriate resource adapter. The application server supplies the naming service, and so it recognizes that a request is being made to locate a resource adapter. In such a case, the application server interposes a resource-adapter-supplied intermediate object that interacts between the resource adapter and the application server. Through this intermediating object, the application server manages the items within the SPI contract below the awareness of the application.

For more information about J2EE™ Connector Architecture, visit **http://java.sun.com/j2ee/connector/**.

About iWay Technology

Novell exteNd Composer uses licensed J2EE™ Connector Architecture adapter technology from iWay Software (a division of Information Builders, Inc.) to mediate EIS interactions in the Composer Connect for PeopleSoft®. A leader in the J2EE™ Connector Architecture technology space, iWay Software provide resource adapters and connectivity solutions across a wide array of EIS and other systems.

For more information about iWay, see **http://www.iwaysoftware.com**.

What Kinds of Applications Can You Build Using the Composer Connect for PeopleSoft®?

With Composer Connect for PeopleSoft®, you can build any kind of web service or integration application that needs to push data into or pull data from a PeopleSoft-based data store using XML as the interchange format. Your integration application can be deployed to a J2EE application server and run as a public web service, or it can be used in "behind the firewall" scenarios. It can be triggered by a servlet, JSP, EJB, e-mail, timer, file arrival, JMS message arrival, or programmatically via your own custom code. It can also run standalone or as part of a workflow built using Composer Enterprise Edition's Process Manager. (For more information on deployment options, see the deployment chapter of the *Composer User's Guide*.)

2 Getting Started with the Composer Connect for PeopleSoft®

This chapter discusses software setup and configuration issues relevant to using the Composer Connect for PeopleSoft®, as well as the steps needed to create a Composer Connection Resource for a PeopleSoft® Component.

Setup and Configuration

A number of steps must be taken before you can use the Composer Connect for PeopleSoft®. You must, at a minimum:

1. Put copies of the PeopleSoft Java Object Adapter (**psjoa.jar**) on design-time *and* runtime (application server) machines. See "Putting psjoa.jar in the Classpath" on page 16.
2. Deploy the supplied remote interfaces (Component Interfaces). See "Deploying the PeopleSoft Component Interface Files" on page 17.
3. Generate PeopleSoft Component Interface API classes, package them in a JAR, and put the JAR on design-time and runtime machines. See "Building PeopleSoft Component Interface API Classes" on page 26.
4. Update classpath entries in **xconfig.xml** (design environment) as well as on the server, to reflect addition of new JARs. See (again) "PeopleSoft Java Object Adapter" on page 15.
5. Update the license for your Composer Connect for PeopleSoft® installation, as needed. See "Updating and/or Activating Your License(s)" below.
6. Create new component interfaces for PeopleSoft services that you wish to expose. See "How to Create a New Component Interface" on page 63 in Appendix B of this guide.

The steps necessary to accomplish these requirements are discussed in detail in the following sections.

PeopleSoft Java Object Adapter

Before attempting to use the Composer Connect for PeopleSoft® (which is installed automatically as part of the default exteNd suite installation process), you must complete the installation by obtaining and installing the **psjoa.jar** file, which comes from PeopleSoft®. *This file is proprietary to PeopleSoft® and is not shipped by Novell nor installed as part of the Novell exteNd Suite.*

The **psjoa.jar** file may already exist on your PeopleSoft Application Server under the **PS_HOME/Web/psjoa** directory. If not, contact your PeopleSoft representative to obtain a copy of this file.

When you have located the **psjoa.jar** file, you must add it to your design-time classpath as well as your runtime-environment classpath (typically the application server classpath). The procedures for doing this are given below.

Putting psjoa.jar in the Classpath

The PeopleSoft Java Object Adapter (psjoa.jar) must be added to your design-time and runtime environments before you can use the Composer Connect for PeopleSoft®.

➤ To add psjoa.jar to your design-time configuration

1. Determine the version number of your existing PeopleSoft system. It should conform to version 8.1 or 8.4. These are the only versions currently supported by Novell exteNd Composer.
2. If Composer is running, shut it down before proceeding.
3. Obtain **psjoa.jar** from PeopleSoft® if it is not already included in your preexisting PeopleSoft installation®.
4. Copy **psjoa.jar** to the **/Common/lib** folder of the exteNd installation directory on your design-time machine.
   - In this folder, you should also see files called **pstools.properties** and **iwpeoplesoft.jar**. If these files are not present, contact Novell customer support to obtain them.
5. Ensure that the JAR files are on your design-time CLASSPATH. To do this: Locate your **xconfig.xml** file under **/Composer/Designer/bin** and open it in a text editor. Scroll to the bottom. Within the <RUNTIME> element, you should see many <JAR> entries. Add an additional <JAR> element as follows:

   ```
   <JAR>..\..\..\Common\lib\psjoa.jar;..\..\..\Common\lib\iwpeoplesoft.jar;..\..\..\C
   ommon\lib\psGenComp.jar;..\..\..\Common\lib\pstools.properties</JAR>
   ```

   This entry essentially tells the classloader where it can find four files:
   - psjoa.jar—This is PeopleSoft's own Java Object Adapter jar (which ships with PeopleSoft)
   - iwpeoplesoft.jar—Comes with Novell exteNd installation.
   - pstools.properties—This file is not an absolute requirement but will prevent certain exceptions from being thrown in the background.
   - **psGenComp.jar** —You will need to generate this JAR yourself according to the procedure described at "Compiling the Component Interface Files" on page 28 further below.
6. In **xconfig.xml**, verify that a <JAR> entry exists, referring to **iwpsci81.jar** if you are using PeopleSoft 8.1, or **iwpsci84.jar** if you are using PeopleSoft 8.4. (These files are installed as part of the Novell exteNd suite installation.)

   CAUTION: *Only **one** of these JARs should be on the classpath. Do not allow both JARs to be referenced in **xconfig.xml**, as this will cause unpredictable behavior.*

7. Save **xconfig.xml** and close it.

➤ To add psjoa.jar to your Novell application server environment

1. Obtain **psjoa.jar** from PeopleSoft® if not already included in your preexisting PeopleSoft installation®.
2. Copy **psjoa.jar** to a suitable location on your application server. The exact location doesn't matter as long as you create a CLASSPATH entry pointing to it, as described in the next step.
3. Update the application server CLASSPATH. For the Novell exteNd application server, locate the **AgJars.conf** file under the **AppServer/bin** directory. Under "MODULE COMMON" create a new pair of entries:

   ```
   $SS_LIB ../../Common/lib/psjoa.jar
   $SS_LIB ../../Common/lib/psGenComp.jar
   ```

   (This example assumes that you have placed the JAR files under **/Common/lib**. Edit as necessary to reflect the actual target directory.)

   NOTE: For application servers other than Novell exteNd, follow the application server vendor's instructions for updating the classpath.

Also verify that an entry exists for **iwpsci81.jar** *or* **iwpsci84.jar** (as appropriate to the version of PeopleSoft you are using). Do not allow *both* JARs to have entries, as this will cause unpredictable behavior. These files are installed as part of the Novell exteNd suite installation. You do not need to obtain them from a third party.

Deploying the PeopleSoft Component Interface Files

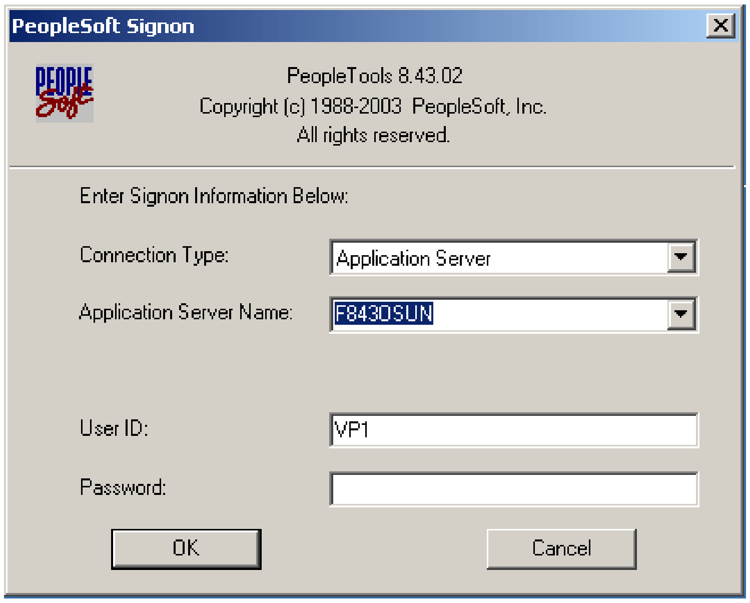
Before you can use the Component Interface APIs discussed in "Building PeopleSoft Component Interface API Classes" on page 26, you must expose the supplied component interface files to your PeopleSoft system. This is not done by the exteNd installer program (because the location of your PeopleSoft system and the file-write permissions in that system are not known at install time, and in any case this step requires manipulation of the files using PeopleSoft's Application Designer). The files that you need to deploy are contained in a zip archive called **iwpsci84.zip** (if you are using PeopleSoft 8.4) or **iwpsci81.zip** (if you are using PeopleSoft 8.1), which are included in the Composer install in the <install directory>\Novell\exteNd5\Composer\Designer\lib directory. The discussion that follows applies to PeopleSoft 8.4.

You will need to deploy the files in this archive to PeopleSoft (not to the application server) in order to enable introspection of the PeopleSoft Component Interfaces by the middle-layer adapters that Composer relies on to support interaction with PeopleSoft. The following procedure describes how to do this.

Importing and Building the Component Interfaces

➤ **To import and build the Component Interface files**

1. Copy the **iwpsci84.zip** or **iwpsci81.zip** from <install directory>\Novell\exteNd5\Composer\Designer\lib to an empty directory of your choice (e.g., C:\temp\PSProjects), then expand the zip file. This creates a sub-folder called IWY_CI_84 (which contains 2 files) or IWY_CI_81 (which contains 21 files).
2. Launch the PeopleSoft Application Designer in two-tier mode.
3. From the PeopleSoft Login Screen, select the Connection Type dropdown and select the appropriate database for your system (e.g., Oracle).
4. Log on using your ID/password.



PeopleSoft Signon

PeopleTools 8.43.02
Copyright (c) 1988-2003 PeopleSoft, Inc.
All rights reserved.

Enter Signon Information Below:

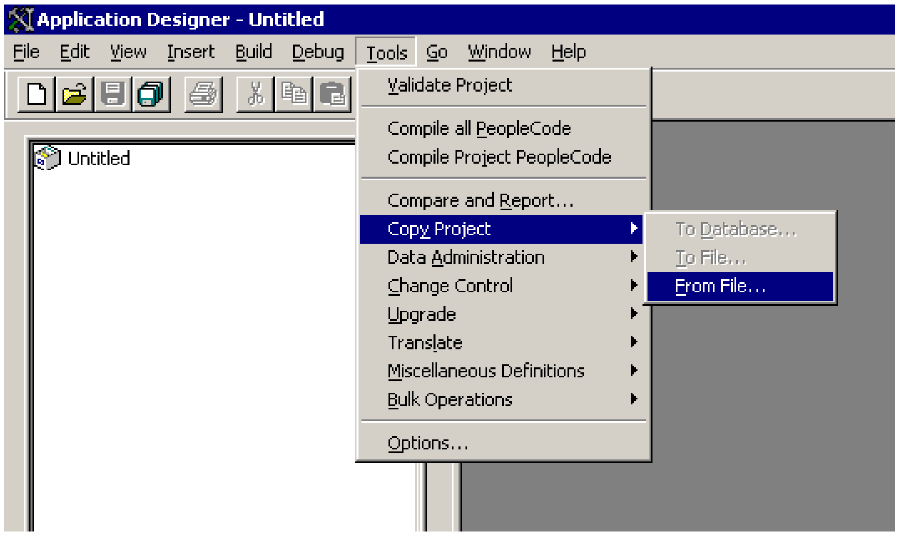
Connection Type: Application Server

Application Server Name: F843OSUN

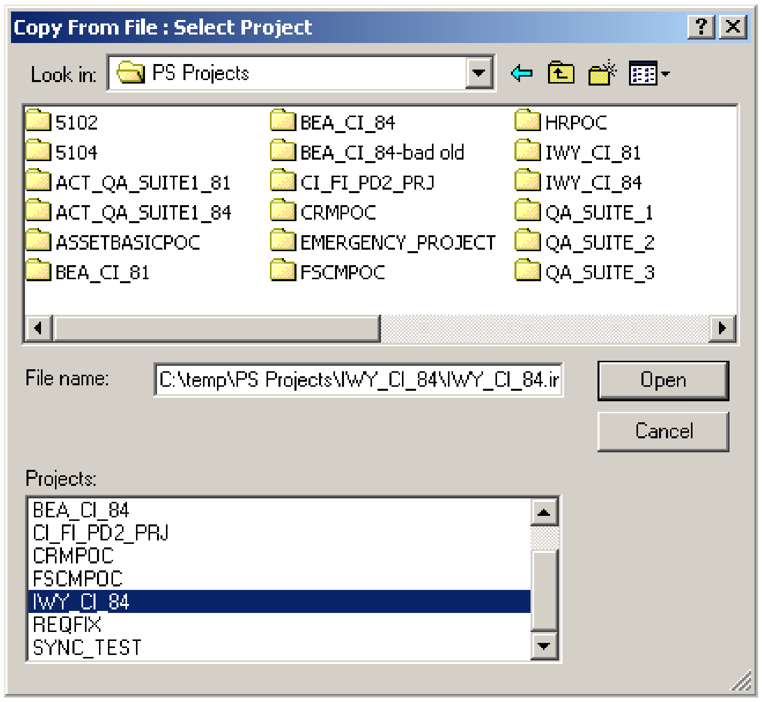
User ID: VP1

Password:

5 From the **Tools** menu, choose **Copy Project > From File**.



6 In the file-browsing dialog that appears (see the following illustration), navigate to your unzipped project directory and select IWY_CI_84 or IWY_CI_81.


Copy From File : Select Project

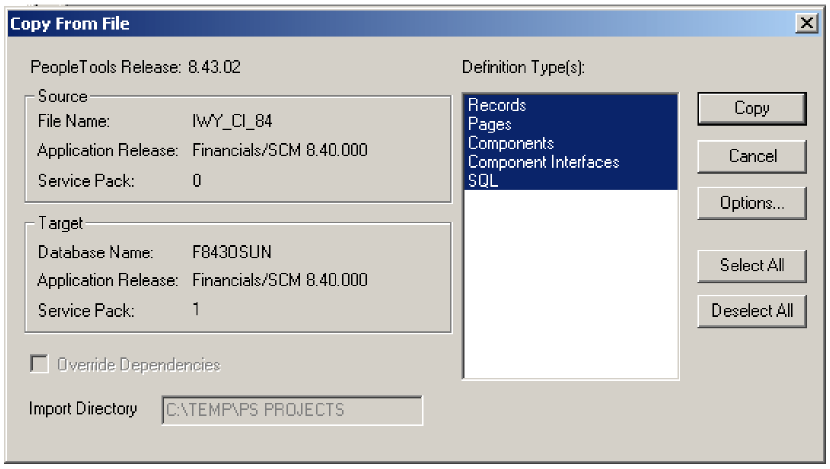
Look in: PS Projects

5102, 5104, ACT_QA_SUITE1_81, ACT_QA_SUITE1_84, ASSETBASICPOC, BEA_CI_81, BEA_CI_84, BEA_CI_84-bad old, CI_FI_PD2_PRJ, CRMPOC, EMERGENCY_PROJECT, FSCMPOC, HRPOC, IWY_CI_81, IWY_CI_84, QA_SUITE_1, QA_SUITE_2, QA_SUITE_3

File name: C:\temp\PS Projects\IWY_CI_84\IWY_CI_84.ir

Projects:
BEA_CI_84
CI_FI_PD2_PRJ
CRMPOC
FSCMPOC
IWY_CI_84
REQFIX
SYNC_TEST

7 Click the **Open** button. The project appears in the Project Workspace and there is output in the Output Window. Make sure that all the objects listed under Object Types are highlighted, and then click **Copy**.


Copy From File

PeopleTools Release: 8.43.02

Source
File Name: IWY_CI_84
Application Release: Financials/SCM 8.40.000
Service Pack: 0

Target
Database Name: F8430SUN
Application Release: Financials/SCM 8.40.000
Service Pack: 1

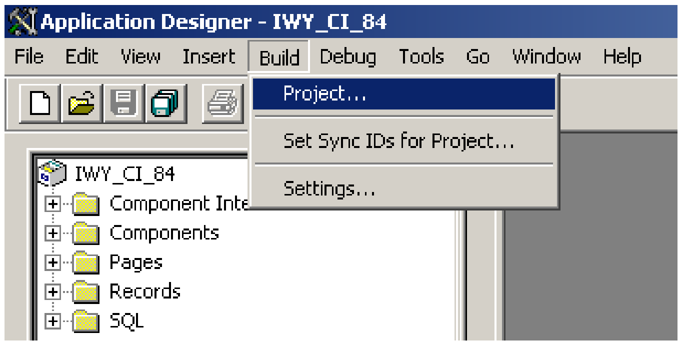
Override Dependencies

Import Directory: C:\TEMP\PS PROJECTS

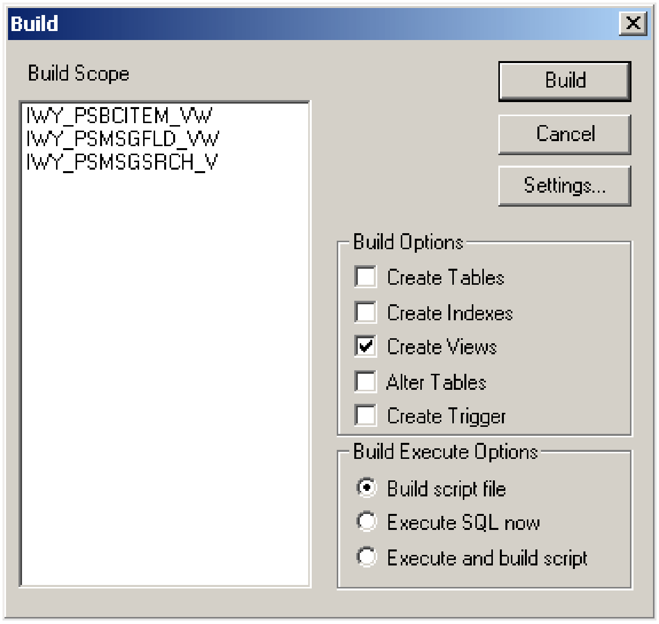
Definition Type(s):
Records
Pages
Components
Component Interfaces
SQL

The Application Designer displays an “Upgrade Copy ended” message in the Output Window to indicate successful completion.

8 Next, we will build the views in the project by choosing **Build > Project** from the main menu.



9 Check the **Create Views** checkbox from Build Options. (See the following illustration.) Under **Rule Execute Options,** select the Build Script File radio button.


Build

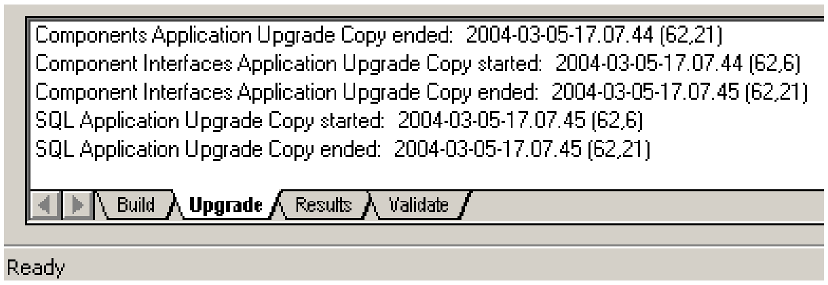
Build Scope
IWY_PSBCITEM_VW
IWY_PSMSGFLD_VW
IWY_PSMSGSRCH_V

Build Options
- [ ] Create Tables
- [ ] Create Indexes
- [x] Create Views
- [ ] Alter Tables
- [ ] Create Trigger

Build Execute Options
- ● Build script file
- ○ Execute SQL now
- ○ Execute and build script

You may want to click on the **Settings...** button to verify that “Recreate view if it already exists” is checked, then return to the Build dialogue box.

10 Click **Build**.



```
Components Application Upgrade Copy ended:  2004-03-05-17.07.44 (62,21)
Component Interfaces Application Upgrade Copy started:  2004-03-05-17.07.44 (62,6)
Component Interfaces Application Upgrade Copy ended:  2004-03-05-17.07.45 (62,21)
SQL Application Upgrade Copy started:  2004-03-05-17.07.45 (62,6)
SQL Application Upgrade Copy ended:  2004-03-05-17.07.45 (62,21)
```

11 Verify in the Upgrade tab (at the bottom of the main window) that the view built correctly.

NOTE: If the view has not been generated correctly, close the dialog and double-click the SQL Build log.

Configuring Component Interface Security

You must ensure that all Application Explorer users have access to the Component Interfaces that you deployed and built in the previous sections. As with all PeopleSoft objects, security is assigned at the Permission List level. Review your site security requirements to determine which users are going to work with Composer, and then set Component Interface security for each distinct Permission List belonging to those users.

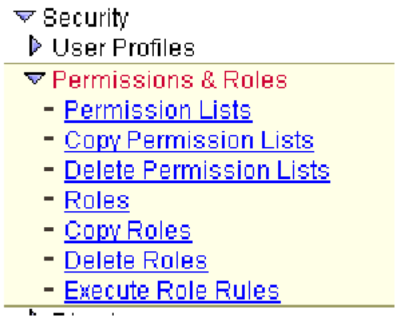
NOTE: These Component Interfaces have only Get and Find access and cannot be used to update your PeopleSoft database. This minimizes any possible security exposure.

In PeopleSoft release 8.1, you may set security in 2, 3, or 4-tier mode; in release 8.4 and higher, you may set security 4-tier mode only.

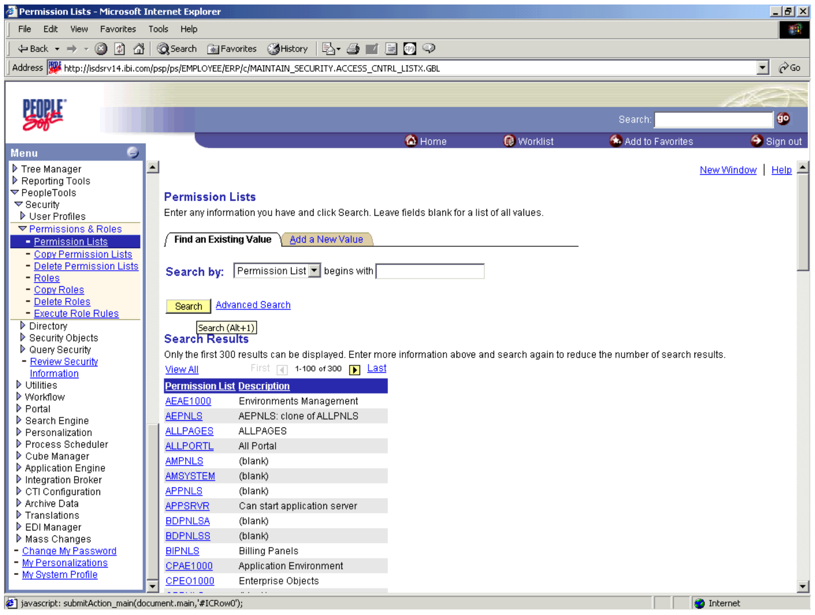
The following procedure describes how to configure security for all supported releases of PeopleSoft in all supported modes. The figures shown in the steps are from PeopleSoft release 8.4 in 4-tier mode.

➢ **To configure security for each Component Interface:**

1 Choose **PeopleTools**, **Security**, **User Profiles**, **Permissions & Roles**, and then **Permission Lists**.



2 Select **Search** and select the relevant Permission List. The Permission List pane opens on the right.



Permission Lists

Enter any information you have and click Search. Leave fields blank for a list of all values.

Search by: Permission List begins with

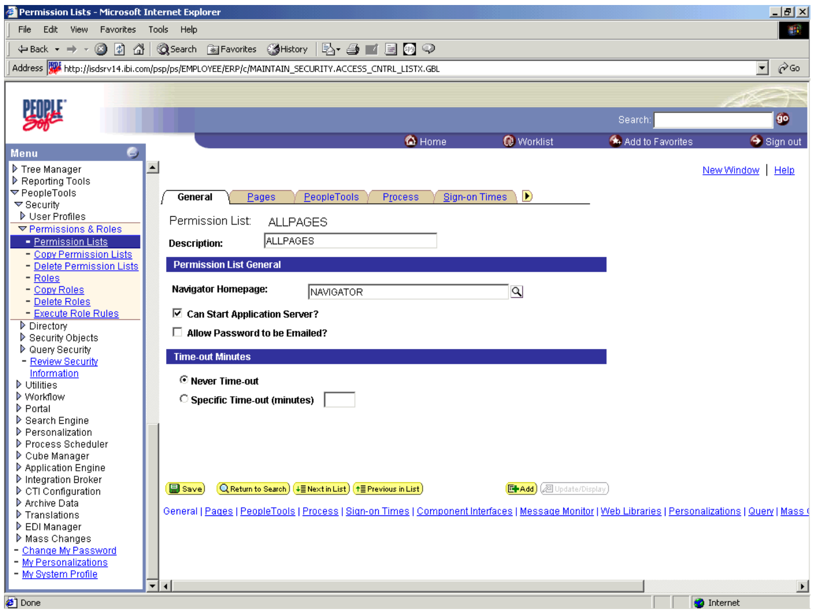
Search Results

Only the first 300 results can be displayed. Enter more information above and search again to reduce the number of search results.

Permission List	Description
AEAE1000	Environments Management
AEPNLS	AEPNLS: clone of ALLPNLS
ALLPAGES	ALLPAGES
ALLPORTL	All Portal
AMPNLS	(blank)
AMSYSTEM	(blank)
APPNLS	(blank)
APPSRVR	Can start application server
BDPNLSA	(blank)
BDPNLSS	(blank)
BIPNLS	Billing Panels
CPAE1000	Application Environment
CPEO1000	Enterprise Objects

javascript: submitAction_main(document.main,'#ICRow0');

3 Click the right arrow next to the **Sign-on Times** tab to display the Component Interfaces tab.



Permission List: ALLPAGES

Description: ALLPAGES

Permission List General

Navigator Homepage: NAVIGATOR

☑ Can Start Application Server?

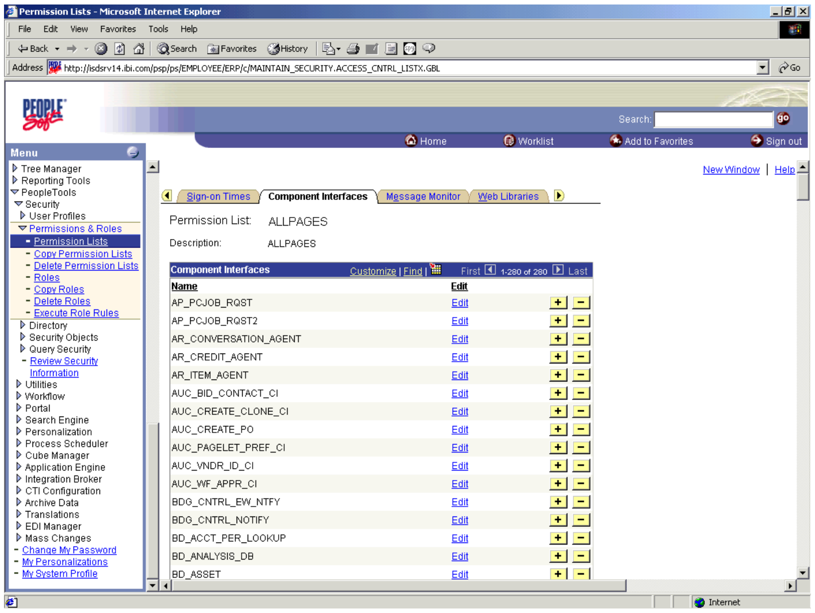
☐ Allow Password to be Emailed?

Time-out Minutes

◉ Never Time-out

○ Specific Time-out (minutes)

4 Click the **Component Interfaces** tab.


Permission Lists - Microsoft Internet Explorer
http://isdsrv14.ibi.com/psp/ps/EMPLOYEE/ERP/c/MAINTAIN_SECURITY.ACCESS_CNTRL_LISTX.GBL
Home
Worklist
Add to Favorites
Sign out
Menu
Permission Lists
Sign-on Times
Component Interfaces
Message Monitor
Web Libraries
New Window
Help

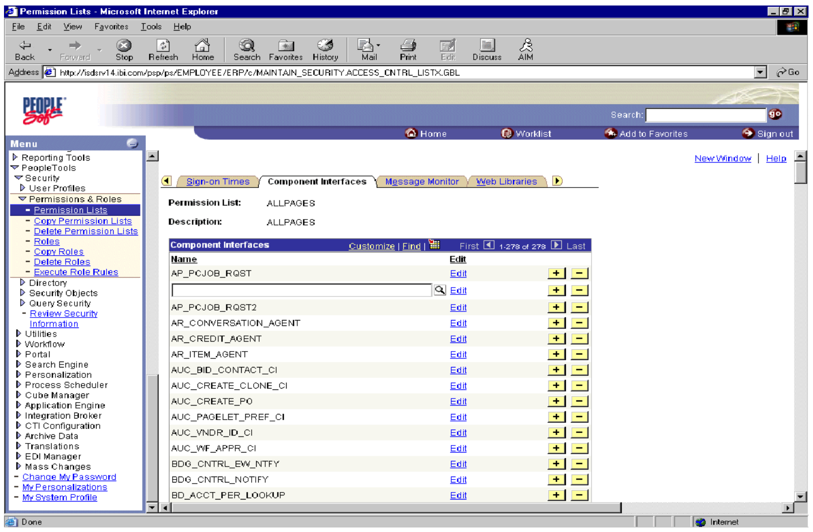
Permission List: ALLPAGES

Description: ALLPAGES

Component Interfaces — Customize | Find | First 1-280 of 280 Last

Name	Edit
AP_PCJOB_RQST	Edit
AP_PCJOB_RQST2	Edit
AR_CONVERSATION_AGENT	Edit
AR_CREDIT_AGENT	Edit
AR_ITEM_AGENT	Edit
AUC_BID_CONTACT_CI	Edit
AUC_CREATE_CLONE_CI	Edit
AUC_CREATE_PO	Edit
AUC_PAGELET_PREF_CI	Edit
AUC_VNDR_ID_CI	Edit
AUC_WF_APPR_CI	Edit
BDG_CNTRL_EW_NTFY	Edit
BDG_CNTRL_NOTIFY	Edit
BD_ACCT_PER_LOOKUP	Edit
BD_ANALYSIS_DB	Edit
BD_ASSET	Edit

5 To add a new row to the Component Interfaces list, select the plus sign (+).


Permission Lists - Microsoft Internet Explorer
http://isdsrv14.ibi.com/psp/ps/EMPLOYEE/ERP/c/MAINTAIN_SECURITY.ACCESS_CNTRL_LISTX.GBL
Home
Worklist
Add to Favorites
Sign out
Menu
Permission Lists
Sign-on Times
Component Interfaces
Message Monitor
Web Libraries
New Window
Help

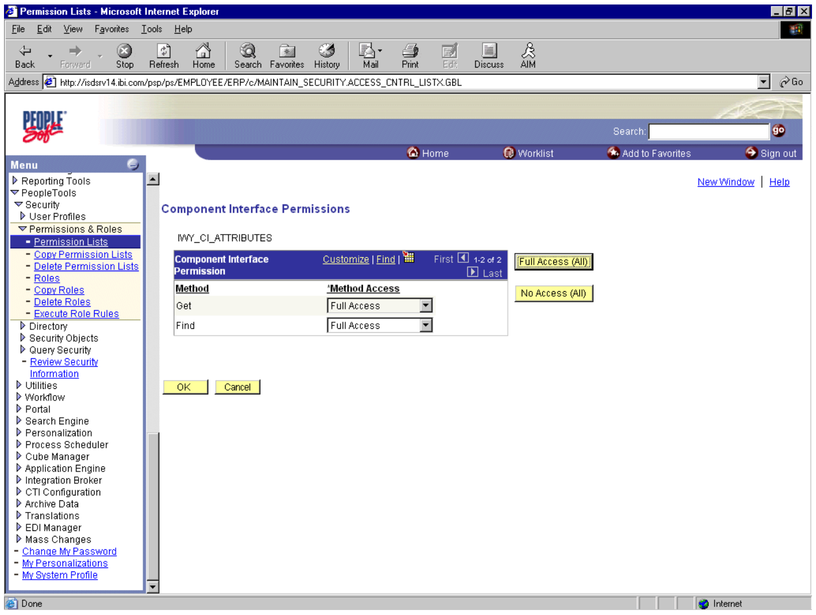
Permission List: ALLPAGES

Description: ALLPAGES

Component Interfaces — Customize | Find | First 1-279 of 279 Last

Name	Edit
AP_PCJOB_RQST	Edit
	Edit
AP_PCJOB_RQST2	Edit
AR_CONVERSATION_AGENT	Edit
AR_CREDIT_AGENT	Edit
AR_ITEM_AGENT	Edit
AUC_BID_CONTACT_CI	Edit
AUC_CREATE_CLONE_CI	Edit
AUC_CREATE_PO	Edit
AUC_PAGELET_PREF_CI	Edit
AUC_VNDR_ID_CI	Edit
AUC_WF_APPR_CI	Edit
BDG_CNTRL_EW_NTFY	Edit
BDG_CNTRL_NOTIFY	Edit
BD_ACCT_PER_LOOKUP	Edit

6 Enter or select the IWY_CI_ATTRIBUTES Component Interface and select **Edit**.



Component Interface Permissions

IWY_CI_ATTRIBUTES

Component Interface Permission

Method	*Method Access
Get	Full Access
Find	Full Access

7 To set the Get and Find methods to Full Access, select **Full Access (All)**.

8 Select **OK**.

9 Repeat steps 5 through 8 for the IWY_CI_MESSAGES Component Interface.

10 Scroll down to the bottom of the Component Interfaces window and click **Save**.

You have finished configuring security for the Component Interfaces delivered with Composer. To test these Component Interfaces, see "Testing the Component Interfaces" on page 23.

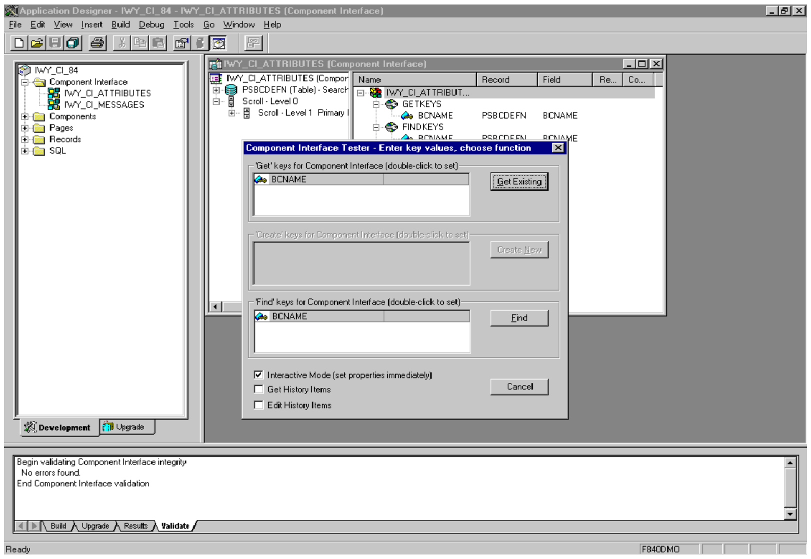
Testing the Component Interfaces

You must test each of the Component Interfaces that were created and secured in the previous sections before using them.

➢ To test the Component Interfaces:

1 In PeopleSoft Application Designer, open the IWY_CI_ATTRIBUTES Component Interface.

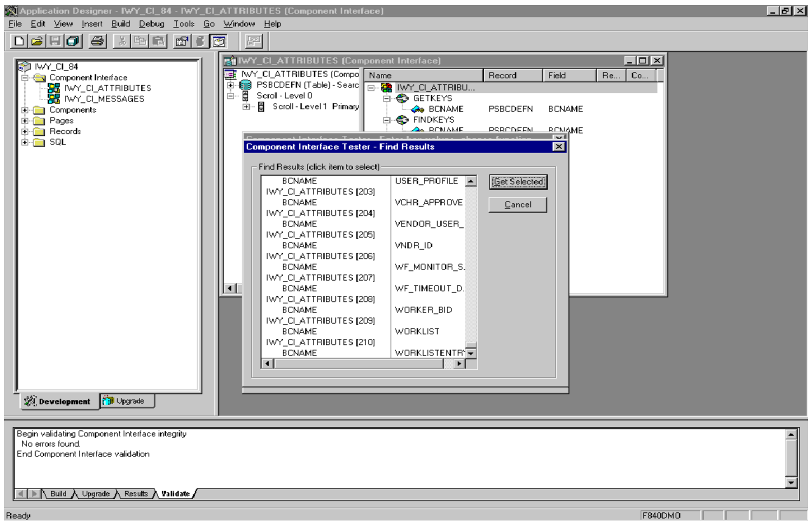
2 Select **Tools**, and then **Test Component Interface**. The Component Interface Tester dialog box is displayed.


```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

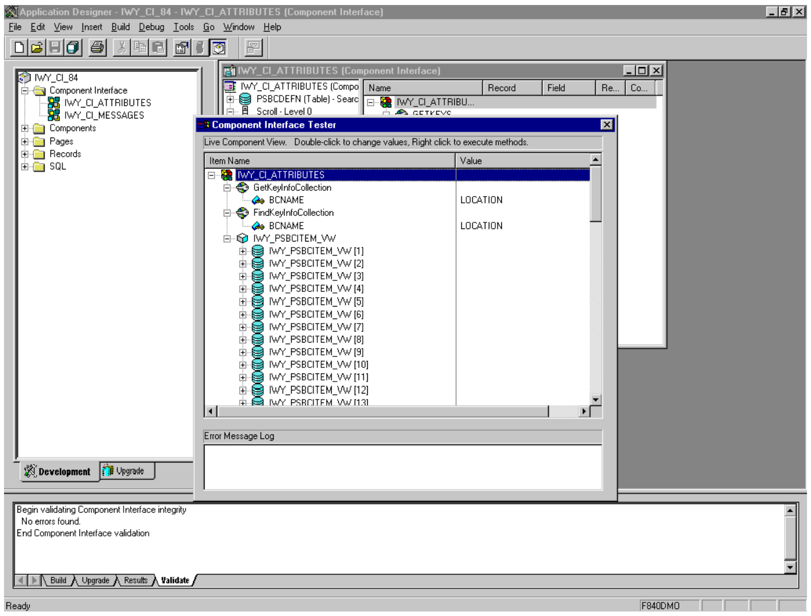
NOTE: The **Create New** option is disabled because the Add method is not applicable to this Component Interface.

3 Click **Find**. Entries for the underlying component are displayed.

NOTE: A message may appear stating that display is limited to a certain number of entries; this is not a problem.


```
Find Results (click item to select)
      BCNAME                     USER_PROFILE
IWY_CI_ATTRIBUTES [203]
      BCNAME                     VCHR_APPROVE
IWY_CI_ATTRIBUTES [204]
      BCNAME                     VENDOR_USER_
IWY_CI_ATTRIBUTES [205]
      BCNAME                     VNDR_ID
IWY_CI_ATTRIBUTES [206]
      BCNAME                     WF_MONITOR_S.
IWY_CI_ATTRIBUTES [207]
      BCNAME                     WF_TIMEOUT_D.
IWY_CI_ATTRIBUTES [208]
      BCNAME                     WORKER_BID
IWY_CI_ATTRIBUTES [209]
      BCNAME                     WORKLIST
IWY_CI_ATTRIBUTES [210]
      BCNAME                     WORKLISTENTRY
```
```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

4 Highlight one of the lines with its corresponding key in the Find Results window and click **Get Selected**. The relevant data for the selected key is displayed.

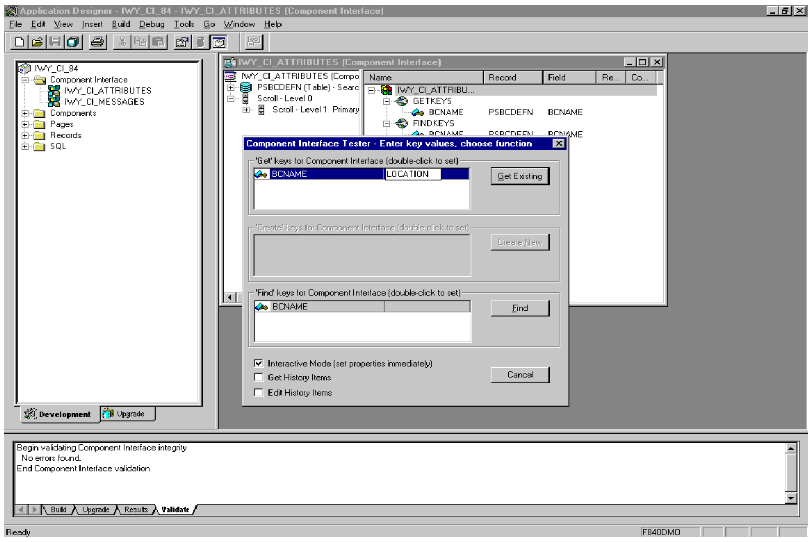

Application Designer - IWY_CI_84 - IWY_CI_ATTRIBUTES (Component Interface)
File Edit View Insert Build Debug Tools Go Window Help
Component Interface Tester
Live Component View. Double-click to change values, Right click to execute methods.
Error Message Log
Development
Upgrade
Build
Upgrade
Results
Validate
Ready
F840DMO

Item Name	Value
IWY_CI_ATTRIBUTES	
GetKeyInfoCollection	
BCNAME	LOCATION
FindKeyInfoCollection	
BCNAME	LOCATION
IWY_PSBCITEM_VW	
IWY_PSBCITEM_VW [1]	
IWY_PSBCITEM_VW [2]	
IWY_PSBCITEM_VW [3]	
IWY_PSBCITEM_VW [4]	
IWY_PSBCITEM_VW [5]	
IWY_PSBCITEM_VW [6]	
IWY_PSBCITEM_VW [7]	
IWY_PSBCITEM_VW [8]	
IWY_PSBCITEM_VW [9]	
IWY_PSBCITEM_VW [10]	
IWY_PSBCITEM_VW [11]	
IWY_PSBCITEM_VW [12]	
IWY_PSBCITEM_VW [13]	

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

If this window is displayed, the Component Interface has been successfully tested for the Find method.

5 Click the Get button. For the Get method, an existing key must be entered.


Application Designer - IWY_CI_84 - IWY_CI_ATTRIBUTES (Component Interface)
File Edit View Insert Build Debug Tools Go Window Help
Component Interface Tester - Enter key values, choose function
Get Existing
Create New
Find
Cancel
Development
Upgrade
Build
Upgrade
Results
Validate
Ready
F840DMO

'Get' keys for Component Interface (double-click to set)

BCNAME	LOCATION

'Create' keys for Component Interface (double-click to set)

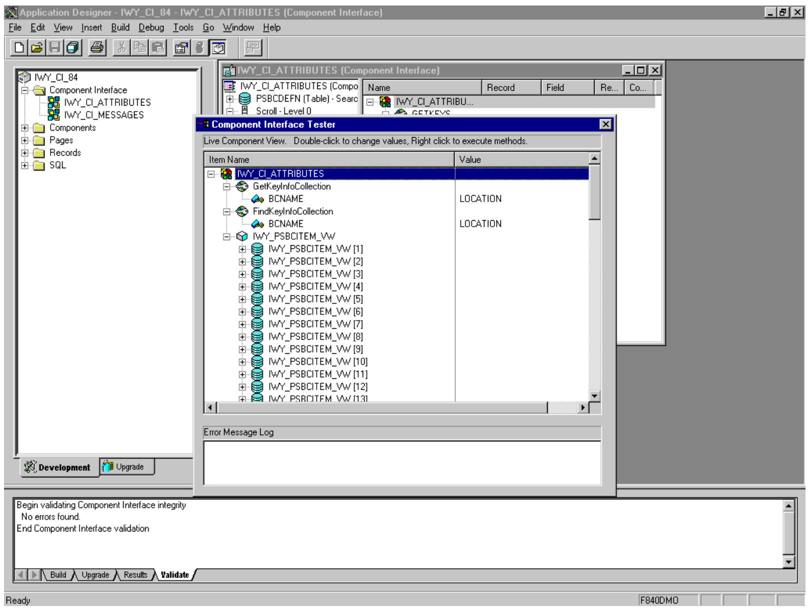
'Find' keys for Component Interface (double-click to set)

BCNAME	

☑ Interactive Mode (set properties immediately)
☐ Get History Items
☐ Edit History Items

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

The exposed properties for the key that is entered are returned.



Live Component View. Double-click to change values, Right click to execute methods.

Item Name	Value
IWY_CI_ATTRIBUTES	
GetKeyInfoCollection	
BCNAME	LOCATION
FindKeyInfoCollection	
BCNAME	LOCATION
IWY_PSBCITEM_VW	
IWY_PSBCITEM_VW [1]	
IWY_PSBCITEM_VW [2]	
IWY_PSBCITEM_VW [3]	
IWY_PSBCITEM_VW [4]	
IWY_PSBCITEM_VW [5]	
IWY_PSBCITEM_VW [6]	
IWY_PSBCITEM_VW [7]	
IWY_PSBCITEM_VW [8]	
IWY_PSBCITEM_VW [9]	
IWY_PSBCITEM_VW [10]	
IWY_PSBCITEM_VW [11]	
IWY_PSBCITEM_VW [12]	
IWY_PSBCITEM_VW [13]	

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

If this window is displayed, the Component Interface has been successfully tested for the Get method.

6 Repeat this process for the IWY_CI_MESSAGES Component Interface.

You have finished testing the Component Interfaces.

Generating Component Interface APIs

You must create a PeopleSoft API to enable communications with the PeopleSoft application. The API is a collection of Java class files that reside on the client machine and mediate between the client application layer and PeopleSoft. You also must apply security to the component interface and test the component interface (see "Securing a Component Interface" on page 66 and "Testing a Component Interface" on page 71).

Building PeopleSoft Component Interface API Classes

Before you can build working integration applications with the Composer Connect for PeopleSoft®, you must create Java classes for the particular PeopleSoft services you wish to utilize, and make these classes available to Composer (at design time as well as runtime). These classes serve as the *API bindings* that allow your Composer components to perform Add, Update, and other operations on the target PeopleSoft system.

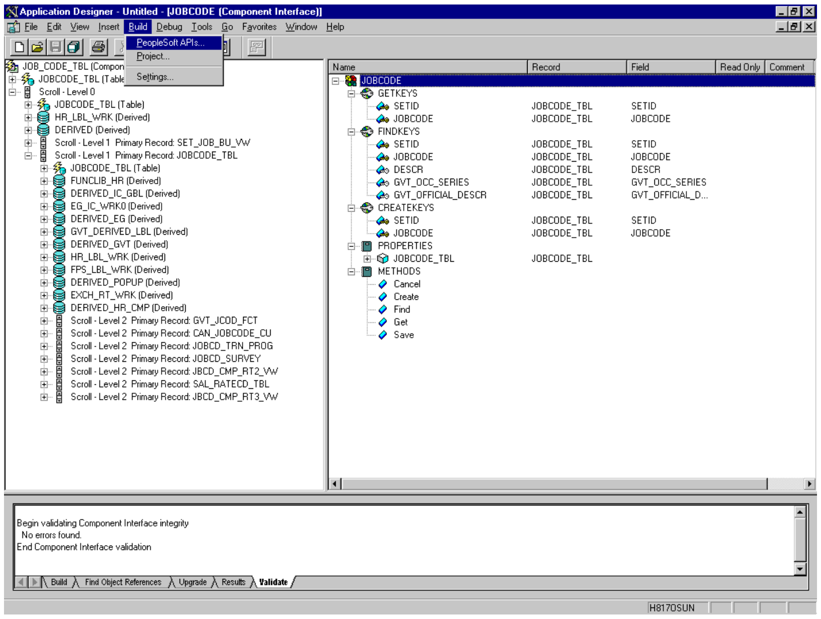
To create the necessary Java classes, you will:

- Determine which PeopleSoft services (methods) you wish to call from Composer.
- Create a Component Interface for each service using the PeopleSoft Application Designer.
- Use the Application Designer to generate Java source files representing the API bindings for the particular Component Interface(s) you want to use (which could be just a few, or potentially many).
- Use **javac.exe** (in your JDK) to compile the generated Java sources into class files.
- Use **jar.exe** (from the JDK) to assemble the class files into a Java archive (JAR).
- Put the JAR on your design-time and runtime machines and make appropriate classpath updates.

The detailed procedure is as follows.

➢ To create Component Interface Classes (and JAR)

1 Launch your copy of PeopleSoft Application Designer.


Application Designer - Untitled - [JOBCODE (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
PeopleSoft APIs...
Project...
Settings...

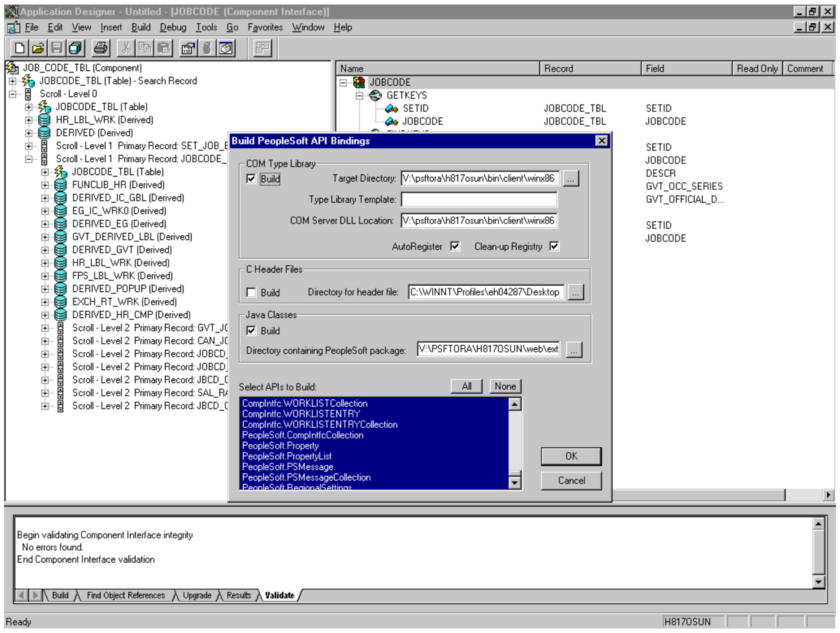
Name	Record	Field	Read Only	Comment
JOBCODE				
GETKEYS				
SETID	JOBCODE_TBL	SETID		
JOBCODE	JOBCODE_TBL	JOBCODE		
FINDKEYS				
SETID	JOBCODE_TBL	SETID		
JOBCODE	JOBCODE_TBL	JOBCODE		
DESCR	JOBCODE_TBL	DESCR		
GVT_OCC_SERIES	JOBCODE_TBL	GVT_OCC_SERIES		
GVT_OFFICIAL_DESCR	JOBCODE_TBL	GVT_OFFICIAL_D...		
CREATEKEYS				
SETID	JOBCODE_TBL	SETID		
JOBCODE	JOBCODE_TBL	JOBCODE		
PROPERTIES				
JOBCODE_TBL	JOBCODE_TBL			
METHODS				
Cancel				
Create				
Find				
Get				
Save				

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

2 From the PeopleSoft Application Designer, open a component interface.

3 Click the right-hand pane to give it focus. Then select **PeopleSoft APIs** from the **Build** menu. (See the previous illustration.)

The Build PeopleSoft API Bindings dialog box opens and prompts you for the types of bindings to create.



COM Type Library
- [✓] Build — Target Directory: V:\psftora\h817osun\bin\client\winx86
- Type Library Template:
- COM Server DLL Location: V:\psftora\h817osun\bin\client\winx86
- AutoRegister [✓] Clean-up Registry [✓]

C Header Files
- [ ] Build — Directory for header file: C:\WINNT\Profiles\eh04287\Desktop

Java Classes
- [✓] Build
- Directory containing PeopleSoft package: V:\PSFTORA\H817OSUN\web\ext

Select APIs to Build:
- CompIntfc.WORKLISTCollection
- CompIntfc.WORKLISTENTRY
- CompIntfc.WORKLISTENTRYCollection
- PeopleSoft.CompIntfcCollection
- PeopleSoft.Property
- PeopleSoft.PropertyList
- PeopleSoft.PSMessage
- PeopleSoft.PSMessageCollection
- PeopleSoft.RegionalSettings

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

4 Ensure that **Java Classes Build** is selected (deselect "COM Type Library Build" if necessary), then select a directory on your local machine where the generated Java files are to be placed.

5 Select the APIs for which you wish to generate Java source files. (Click the None button first, to clear any selections.)

NOTE: In addition to the APIs for the selected component interface, you also must generate the API files for the following generic component interface properties:

- CompIntfcPropertyInfo
- CompIntfcPropertyInfoCollection

6 Click **OK**. PeopleSoft generates the files. This may take from a few seconds to several minutes, depending on how many selections you made. After the process is complete, a message appears in the PeopleSoft Application Designer output window.

Compiling the Component Interface Files

The files you generated in the previous steps must be compiled into classes, and the classes packaged into a JAR, before you can use them.

Note that PeopleSoft places the Java sources to be compiled in the directory called

```
[targetFolder]\PeopleSoft\Generated\CompIntfc
```

where **targetFolder** is the directory you specified during the build process.

There are two Java files for every API file that you selected when you generated the source files.

Before you compile the Java sources, you should confirm that you have a copy of the PeopleSoft Java Object Adapter (the **psjoa.jar** file that resides on your PeopleSoft Application Server under the PS_HOME\Web\psjoa directory). Note its location.

➢ How to Compile the generated PeopleSoft API Java sources

NOTE: Before beginning, ensure that you have a copy of Sun's JDK (1.4.x or later) on your machine.

1 If you are compiling on a machine that does not have a local copy of the **psjoa.jar** file, bring a copy onto the local machine. You will need to have this JAR in the classpath in order for compilation to occur without errors.

2 Compile the files. To do this, first open an operating-system console and navigate to the target folder you specified in the Java Target Build step (step 4) of the previous procedure. From there, run a batch file or shell script that invokes the **javac.exe** program in your JDK to compile the sources. A typical Windows batch file that does this might look like:

```
@echo off
set JAVA_HOME=<my-java-home>
set PATH=%JAVA_HOME%\bin;%PATH%
set CLASSPATH=%JAVA_HOME%\lib\tools.jar;psjoa.jar;%CLASSPATH%
javac -classpath %CLASSPATH% .\PeopleSoft\Generated\CompIntfc\*.java
```

where:

<my-java-home>

is the fully qualified path name of your Java home directory.

3 Compress the class files into a JAR file. Again, start from the target directory you specified in the Java Target Build step of the previous section's procedure. A Windows batch file similar to the following will create a JAR file named **psGenComp.jar**:

```
@echo off
set JAVA_HOME= my-java-home
set PATH=%JAVA_HOME%\bin;%PATH%
set CLASSPATH=%JAVA_HOME%\lib\tools.jar;%CLASSPATH%
jar cvf psGenComp.jar .\PeopleSoft\Generated\CompIntfc\*.class
```

Substitute the appropriate path for *my-java-home*. (e.g., c:/jdk142.)

NOTE: Java is case-sensitive. Be sure to use the correct case when creating the JAR file since the Java classloader mechanism depends on this.

4 Place the resulting JAR file (in this example, **psGenComp.jar**) into your exteNd installation's **Common/lib** directory. This applies to both design-time and runtime machines, if they are different.

Updating and/or Activating Your License(s)

You will need to activate your design-time *and* runtime copies of the Novell exteNd Composer Connect for PeopleSoft® before using them. Follow the procedure outlined below.

➢ **To update the license on your design-time software**

1 Obtain a valid license string from your Novell representative.

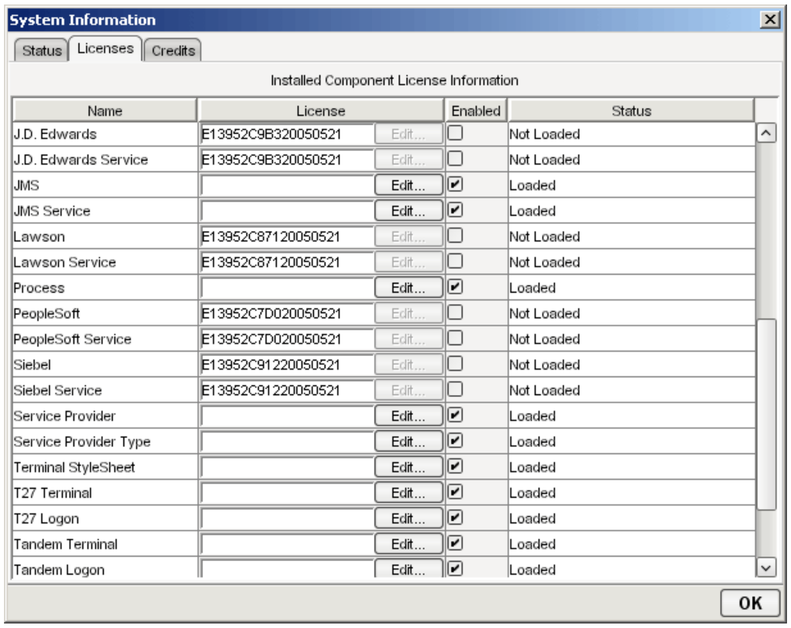
NOTE: The same license string will be used to activate both the design-time and runtime versions of the software.

2 Launch Composer.

3 Use the **About Composer** command on the **Help** menu, on Composer's main menu. An "About" screen appears.

4 At the bottom of the "About" dialog, click the **System** button. A System Information dialog appears.

5 Near the top of the System Information dialog, click the **Licenses** tab to bring it forward.



Installed Component License Information

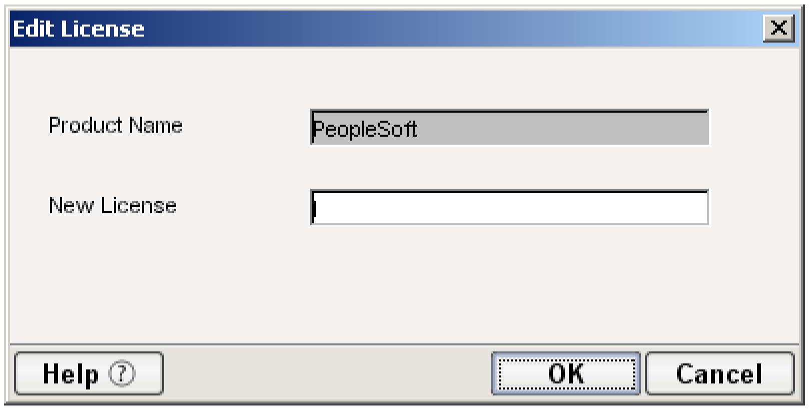
Name	License		Enabled	Status
J.D. Edwards	E13952C9B320050521	Edit...	☐	Not Loaded
J.D. Edwards Service	E13952C9B320050521	Edit...	☐	Not Loaded
JMS		Edit...	☑	Loaded
JMS Service		Edit...	☑	Loaded
Lawson	E13952C87120050521	Edit...	☐	Not Loaded
Lawson Service	E13952C87120050521	Edit...	☐	Not Loaded
Process		Edit...	☑	Loaded
PeopleSoft	E13952C7D020050521	Edit...	☐	Not Loaded
PeopleSoft Service	E13952C7D020050521	Edit...	☐	Not Loaded
Siebel	E13952C91220050521	Edit...	☐	Not Loaded
Siebel Service	E13952C91220050521	Edit...	☐	Not Loaded
Service Provider		Edit...	☑	Loaded
Service Provider Type		Edit...	☑	Loaded
Terminal StyleSheet		Edit...	☑	Loaded
T27 Terminal		Edit...	☑	Loaded
T27 Logon		Edit...	☑	Loaded
Tandem Terminal		Edit...	☑	Loaded
Tandem Logon		Edit...	☑	Loaded

6 Scroll down until you see the table row for PeopleSoft.

NOTE: The Connect ships with an Evaluation license string which may be used for 90 days.

7 To use the evaluation license string, select the check box in the **Enabled** column and skip to Step 10. To enter a different license string, select the check box in the **Enabled** column. The **Edit** button in the Siebel row is enabled.

8 Select **Edit** in the PeopleSoft row. The **Edit License** dialog box is displayed.



Product Name: PeopleSoft

New License:

9 Type the license string for the Composer Connect for PeopleSoft in the **New License** field.

10 Scroll down until you see the row for PeopleSoft Service.

11 Repeat Step 7 through Step 9 for the PeopleSoft Service row.

12 Click **OK**.

13 Exit out of all dialogs using the **OK** button.

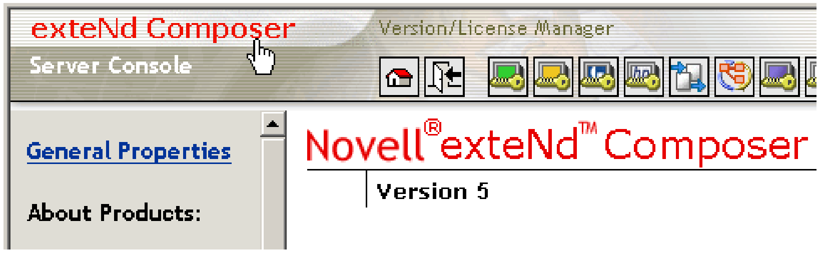
➤ To update the runtime license

1 Start the application server if it is not already running. (Novell exteNd Composer Enterprise Server should already be installed. It will start when the application server starts.)

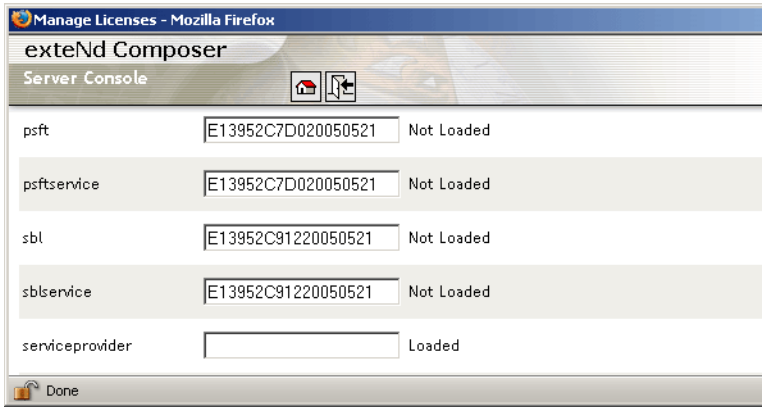
2 Open a browser window and navigate to the Novell exteNd Composer main administrative console. Typically, this is at:

```
http://localhost/exteNdComposer
```

3 In the upper left corner of the console window, click the **exteNd Composer** logo immediately above the words "Server Console." See the following illustration.



4 In the content area of the main frame, near the bottom, click the **Licenses** button. A new window will open, showing the license status of every Composer Connect. See the following illustration.



psft	E13952C7D020050521	Not Loaded
psftservice	E13952C7D020050521	Not Loaded
sbl	E13952C91220050521	Not Loaded
sblservice	E13952C91220050521	Not Loaded
serviceprovider		Loaded

5 Scroll down to the entry labeled "psft".

NOTE: The Connect ships with Evaluation license strings which may be used for 90 days.

6. Type a license string in the text field in the “psft” row; then select **Enabled** in the “psft” row. To use the evaluation license string, just select **Enabled.**
7. Select **Update** in the “psft” row.
8. Scroll down to the entry labeled “psftservice”.
9. Type a license string in the text field in the “psftservice” row; then select **Enabled** in the “psftservice” row. To use the evaluation license string, just select **Enabled.**
10. Select **Update** in the “psftservice” row.

3 Creating a PeopleSoft Component

To create a Component that utilizes the Composer Connect for PeopleSoft, you will generally need to do three things:

- Create a Connection Resource (to allow your component to connect to a PeopleSoft system)
- Create any necessary XML Templates
- Create the Component itself (containing your business logic)

Each of these processes is discussed in detail in this chapter.

Creating Connection Resources

Before creating a component that interacts with a PeopleSoft system, you need to create a Connection Resource, which is a lightweight Composer object (xObject) that encapsulates basic connection information (parameter values) associated with a connection to a back-end system.

In addition to a connection resource, a PeopleSoft Component requires that you have already created XML templates so that you have sample input and output documents for use in designing your component. For more information, see “Creating an XML Template” in the *Novell exteNd Composer User’s Guide.*

If your component design calls for any other resources, such as custom scripts, XSL, XSD, etc., it is best to create these before creating the PeopleSoft Component. For more information, see *“*Creating Custom Scripts” in the *Novell exteNd Composer User’s Guide*.

Types of Connection Resources

You create different types of connection resource depending on the type of interaction with the PeopleSoft system that is desired. If your application needs to initiate PeopleSoft business events, you need to create a PeopleSoft Connection resource. If your application needs to process data when a business event occurs within PeopleSoft, you need to create an inbound connection resource. Inbound connection resources include file, HTTP, and TCP connection types.

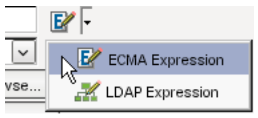
About Constant-Driven and Expression-Driven Connection Parameters

You can specify Connection parameter values in one of two ways: as Constants or as Expressions. A constant-driven parameter uses the value you type in the Connection dialog every time the Connection is used. An expression-driven parameter allows you to set the value using a programmatic expression, which can result in a different value each time the connection is used at runtime. This allows the Connection’s behavior to be flexible and vary based on runtime conditions each time it is used.

For instance, one very simple use of an expression-driven parameter in a Connection would be to define the User ID and Password as PROJECT Variables (e.g. PROJECT.XPATH("USERCONFIG/MyDeployUser"). This way when you deploy the project, you can update the PROJECT Variables in the Deployment Wizard to values appropriate for the final deployment environment. At the other extreme, you could have a custom script that queries a Java business object in the Application Server to determine what User ID and Password to use.

➤ **To switch a parameter from Constant-driven to Expression driven:**

1. Click the right mouse button in the parameter field you are interested in changing.
2. Select **Expression** from the context menu and the editor button will appear or become enabled.

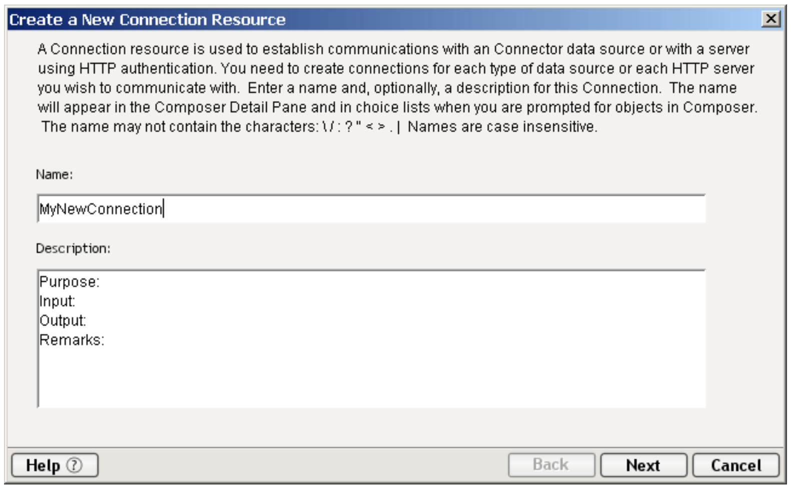


3. Click on the button and then create an expression that evaluates to a valid parameter value at runtime. (Strings should be wrapped in double-quotes.)

Creating a PeopleSoft Connection Resource

➤ **To create a PeopleSoft connection resource:**

1. Select **File > New > xObject** and select the **Resource** tab. Click on **Connection**. The "Create a New Connection Resource" Wizard appears.



A Connection resource is used to establish communications with an Connector data source or with a server using HTTP authentication. You need to create connections for each type of data source or each HTTP server you wish to communicate with. Enter a name and, optionally, a description for this Connection. The name will appear in the Composer Detail Pane and in choice lists when you are prompted for objects in Composer. The name may not contain the characters: \ / : ? " < > . | Names are case insensitive.

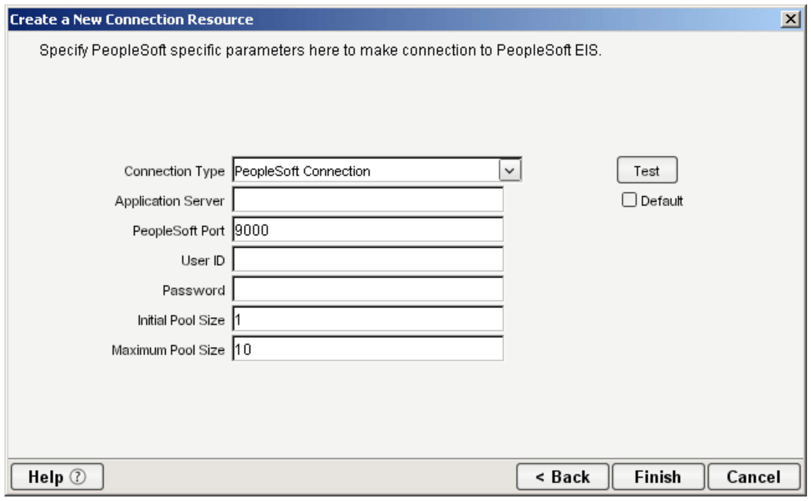
Name:

MyNewConnection

Description:

Purpose:
Input:
Output:
Remarks:

2. Type a **Name** for the connection object.
3. Optionally, type **Description** text.
4. Click **Next**. The parameters panel appears.



Specify PeopleSoft specific parameters here to make connection to PeopleSoft EIS.

Connection Type	PeopleSoft Connection
Application Server	
PeopleSoft Port	9000
User ID	
Password	
Initial Pool Size	1
Maximum Pool Size	10

☐ Default

5 Select **PeopleSoft Connection** from the **Connection Type** pull down menu.

6 In the **Application Server** field, enter the host name or IP address for the PeopleSoft server that you'll be using.

NOTE: This parameter, and all subsequent parameters in this dialog, can be dynamically set using Expressions. See the discussion "About Constant-Driven and Expression-Driven Connection Parameters" later in this chapter.

7 In the **PeopleSoft Port** field, enter the appropriate port number.

8 Enter a valid **User ID** to sign on to the selected PeopleSoft server.

9 Enter a valid **Password**.

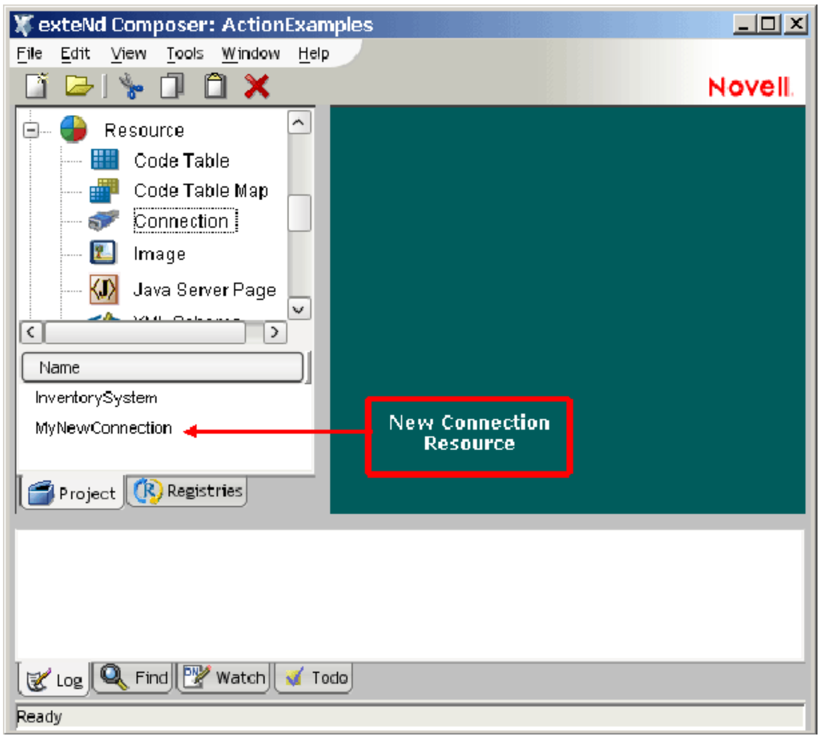
10 In the **Initial Pool Size** field, enter the desired J2EE™ Connector Architecture pool size. (Not applicable for non-managed environments.)

11 Enter a **Maximum Pool Size** as desired. Again, this is a J2EE™ Connector Architecture setting. (Not applicable for non-managed environments.)

12 If desired, select **Test** to see if your connection parameters and current network environment will allow you to create a live connection. (The connection is discarded immediately after the test.) A "success" or "failure" message dialog will appear. *You can continue working with the resource, even if your connection fails.*

NOTE: This does not test the connection pool (if defined).

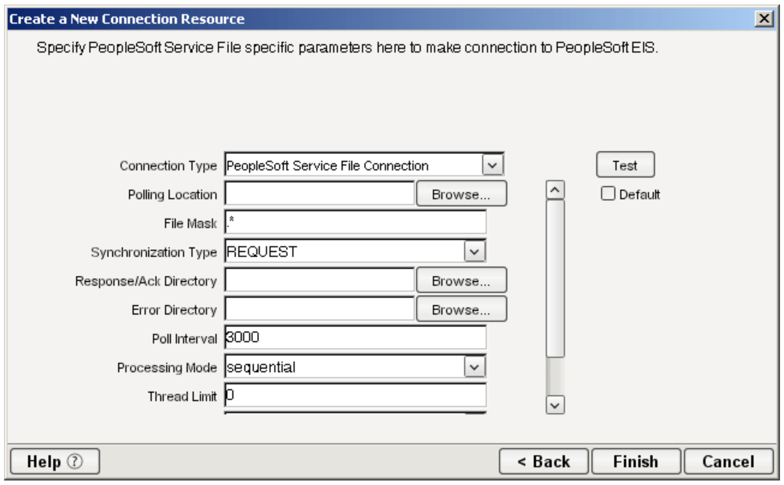
13 Click **Finish**. The newly-created resource connection object appears in the Composer Connection Resource detail pane.



Creating a File Service Connection Resource

To Create a PeopleSoft File Service Connection Resource:

1. Select **File** > **New** > **xObject**. The New xObject dialog box is displayed.
2. Select the **Resource** tab.
3. Double-click on **Connection**. The "Create a New Connection Resource" wizard is displayed.
4. Type a name for the connection object in the **Name** field.
5. Optionally, type a description of the connection object in the **Description** field.
6. Select **Next**. A connection parameters panel is displayed.


Create a New Connection Resource

Specify PeopleSoft Service File specific parameters here to make connection to PeopleSoft EIS.

Connection Type	PeopleSoft Service File Connection
Polling Location	
File Mask	.*
Synchronization Type	REQUEST
Response/Ack Directory	
Error Directory	
Poll Interval	3000
Processing Mode	sequential
Thread Limit	0

7. Select **PeopleSoft File Service Connection** from the **Connection Type** list.
8. Use the **Browse** button next to the **Polling Location** field to select the target file system location for the PeopleSoft XML file.

9 In the **File Mask** field type the file name to be used for the output file generated as a result of this operation. The default is “.*”.

10 Select REQUEST, REQUEST_RESPONSE, or REQUEST_ACK from the **Synchronization Type** list. Select the option that is correct for the Workflow with which you are integrating (e.g., if your Workflow is set up to process a response, select REQUEST_RESPONSE). The Synchronization Type selected also determines the tabs that are displayed in the Service Request Native Environment pane (e.g., if REQUEST_RESPONSE is selected, both Request and Response tabs are displayed).

11 Use the **Browse** button next to the **Response/Ack Directory** field to select the directory to which responses or acknowledgements from your application will be written.

12 Use the **Browse** button next to the **Error Directory** field to select the directory to which error information will be written.

13 In the **Poll Interval** field, type the interval (in milliseconds) in which the Polling Location is checked for input. The default is 3000 (3 seconds).

14 Select Threaded or Sequential from the **Processing Mode** list. Threaded indicates processing of multiple requests simultaneously. Sequential indicates serial processing of requests.

15 If Threaded processing mode is selected in the Processing Mode list, use the **Thread Limit** field to specify the maximum number of requests that can be processed simultaneously.

16 From the **Metadata Connection** list, select a PeopleSoft connection resource from the list of defined connection resources. This connection is used for acquiring metadata from the PeopleSoft system.

17 Type the desired maximum pool size in the **Maximum Pool Size** field. This is a JCA setting that specifies the maximum number of connections to be allowed in the connection pool (not applicable for non-managed environments).

Creating an HTTP Service Connection Resource

➢ **To Create a PeopleSoft HTTP Service Connection Resource:**

1 Select **File > New > xObject**. The New xObject dialog box is displayed.

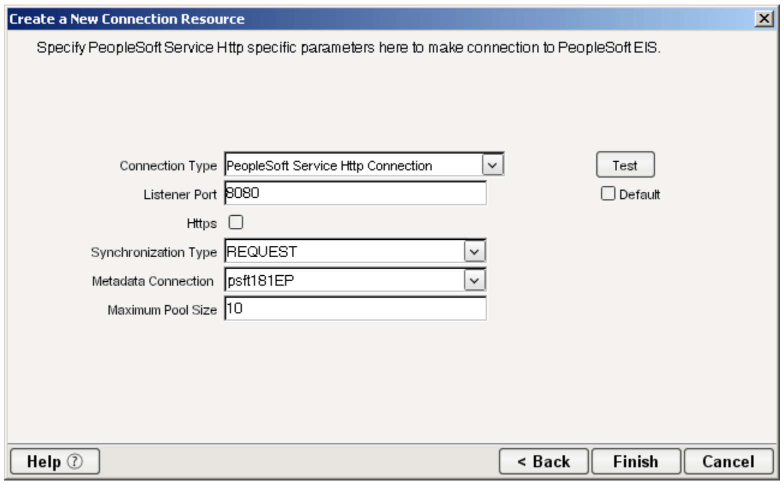
2 Select the **Resource** tab.

3 Double-click on **Connection**. The “Create a New Connection Resource” wizard is displayed.

4 Type a name for the connection object in the **Name** field.

5 Optionally, type a description of the connection object in the **Description** field.

6 Select **Next**. A connection parameters panel is displayed.



Specify PeopleSoft Service Http specific parameters here to make connection to PeopleSoft EIS.

Connection Type: PeopleSoft Service Http Connection
Listener Port: 8080
Https: ☐
Synchronization Type: REQUEST
Metadata Connection: psft181EP
Maximum Pool Size: 10
☐ Default

7 Select **PeopleSoft HTTP Service Connection** from the **Connection Type** list.

8 In the **Listener Port** field, type the number of the port on which the application will listen for PeopleSoft event data. The default port is 8080.

9 To use HTTPS for this connection, select **HTTPS.**

10 Select REQUEST, REQUEST_RESPONSE, or REQUEST_ACK from the **Synchronization Type** list. Select the option that is correct for the Workflow with which you are integrating (e.g., if your Workflow is set up to process a response, select REQUEST_RESPONSE). The Synchronization Type selected also determines the tabs that are displayed in the Service Request Native Environment pane (e.g., if REQUEST_RESPONSE is selected, both Request and Response tabs are displayed).

11 From the **Metadata Connection** list, select an outbound PeopleSoft connection resource from the list of defined connection resources. This connection is used for acquiring metadata from the PeopleSoft system.

12 Type the desired maximum pool size in the **Maximum Pool Size** field. This is a JCA setting that specifies the maximum number of connections to be allowed in the connection pool (not applicable for non-managed environments).

Creating a TCP Service Connection Resource

➢ To Create a PeopleSoft TCP Service Connection Resource:

1 Select **File** > **New** > **xObject**. The New xObject dialog box is displayed.

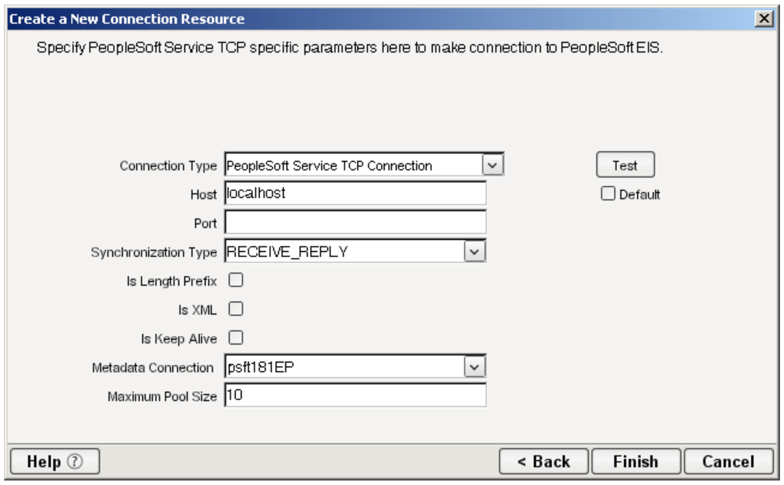
2 Select the **Resource** tab.

3 Double-click on **Connection**. The “Create a New Connection Resource” wizard is displayed.

4 Type a name for the connection object in the **Name** field.

5 Optionally, type a description of the connection object in the **Description** field.

6 Select **Next**. A connection parameters panel is displayed.



Create a New Connection Resource

Specify PeopleSoft Service TCP specific parameters here to make connection to PeopleSoft EIS.

Connection Type	PeopleSoft Service TCP Connection
Host	localhost
Port	
Synchronization Type	RECEIVE_REPLY
Is Length Prefix	☐
Is XML	☐
Is Keep Alive	☐
Metadata Connection	psft181EP
Maximum Pool Size	10

7 Select **PeopleSoft Service TCP Connection** from the **Connection Type** list.

8 In the **Host** field, specify the host name or URL of the system on which the PeopleSoft system is installed.

9 In the **Port** field, type the name of the port on which the host is listening.

10 Select RECEIVE_REPLY, RECEIVE_ACK, or RECEIVE from the **Synchronization Type** list:.

- Select RECEIVE_REPLY if the PeopleSoft application expects a reply sent back to it.
- Select RECEIVE_ACK when a TCP/IP acknowledgement (ACK) is sent back to the PeopleSoft application.

- Select RECEIVE if the PeopleSoft application does not expect a return.

11 Select **Is Length Prefix** if the PeopleSoft application sends data that is not in XML format. The TCP/IP event application must prefix the data with a 4-byte binary length field when writing the data to the TCP/IP port. When this option is selected, deselect the **Is XML** option.

12 Select **Is XML** if the PeopleSoft application sends data in XML format. When this option is selected, deselect the **Is Length Prefix** checkbox.

13 Select **Is Keep Alive** to maintain continuous communication between the PeopleSoft application and your application.

14 From the **Metadata Connection** list, select an PeopleSoft Connection resource from the list of defined connection resources. This connection is used for acquiring metadata from the PeopleSoft system.

15 Type the desired maximum pool size in the **Maximum Pool Size** field. This is a JCA setting that specifies the maximum number of connections to be allowed in the connection pool (not applicable for non-managed environments).

XML Templates for PeopleSoft Components

Before creating a PeopleSoft Component, you should create any XML templates that might be needed. This basically means telling Composer what XML samples to use. (For more information, see *Creating a New XML Template* in the separate *Composer User's Guide.*) Once you've specified the XML templates, you can create a component, using the template's sample documents to represent the inputs and outputs processed by your component.

Creating PeopleSoft Components

Also, as part of the process of creating a PeopleSoft Component, you can select a preexisting PeopleSoft connection resource, or you can create a new one. If you create the connection beforehand (as outlined in the previous chapter), then it is available to for use by any PeopleSoft Components in the current project. *If you have not already created at least one PeopleSoft connection resource in the current project, you will be prompted to do so when you try to create a PeopleSoft Componen*t.

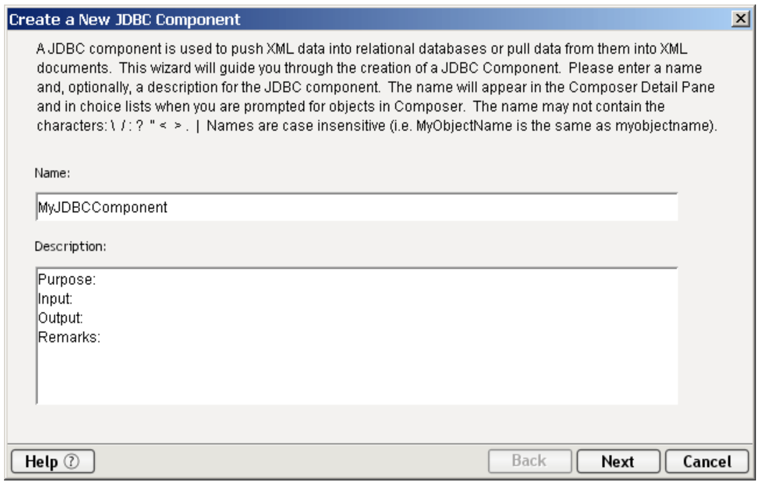
NOTE: You can still create a Component without first creating a Connection Resource, but you won't be able to use any debug features that depend on a live connection.

➢ **To create a new PeopleSoft Component:**

1 Select **File>New>xObject**. In the dialog that appears, select the **Component** tab and then the **PeopleSoft** component type.

NOTE: Alternatively, under **Component** in the Composer Navigator pane (explorer view), you can highlight **PeopleSoft**, click the right mouse button, then select **New**.

2 The "Create a New PeopleSoft Component" Wizard appears.



Create a New JDBC Component

A JDBC component is used to push XML data into relational databases or pull data from them into XML documents. This wizard will guide you through the creation of a JDBC Component. Please enter a name and, optionally, a description for the JDBC component. The name will appear in the Composer Detail Pane and in choice lists when you are prompted for objects in Composer. The name may not contain the characters: \ / : ? " < > . | Names are case insensitive (i.e. MyObjectName is the same as myobjectname).

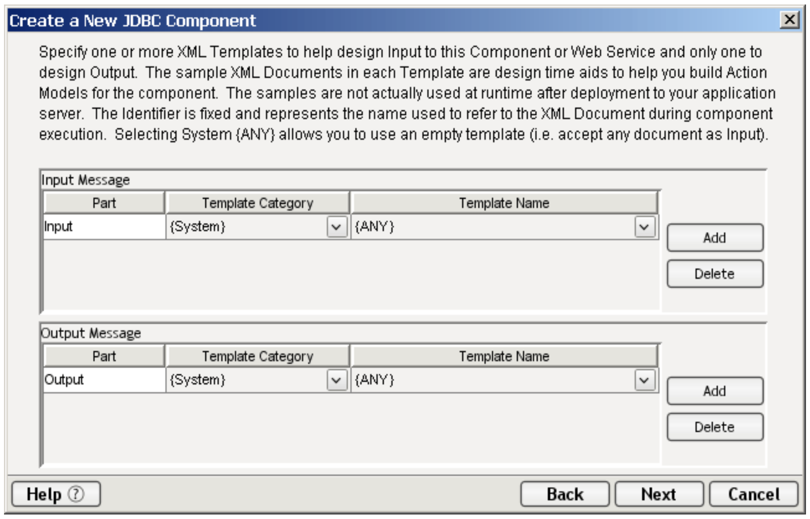
Name: MyJDBCComponent

Description:
Purpose:
Input:
Output:
Remarks:

3 Enter a **Name** for the new PeopleSoft Component.

4 Optionally, type **Description** text.

5 Click **Next**. The XML Input/Output Property Info panel of the New PeopleSoft Component Wizard appears.



Create a New JDBC Component

Specify one or more XML Templates to help design Input to this Component or Web Service and only one to design Output. The sample XML Documents in each Template are design time aids to help you build Action Models for the component. The samples are not actually used at runtime after deployment to your application server. The Identifier is fixed and represents the name used to refer to the XML Document during component execution. Selecting System {ANY} allows you to use an empty template (i.e. accept any document as Input).

Input Message

Part	Template Category	Template Name
Input	{System}	{ANY}

Output Message

Part	Template Category	Template Name
Output	{System}	{ANY}

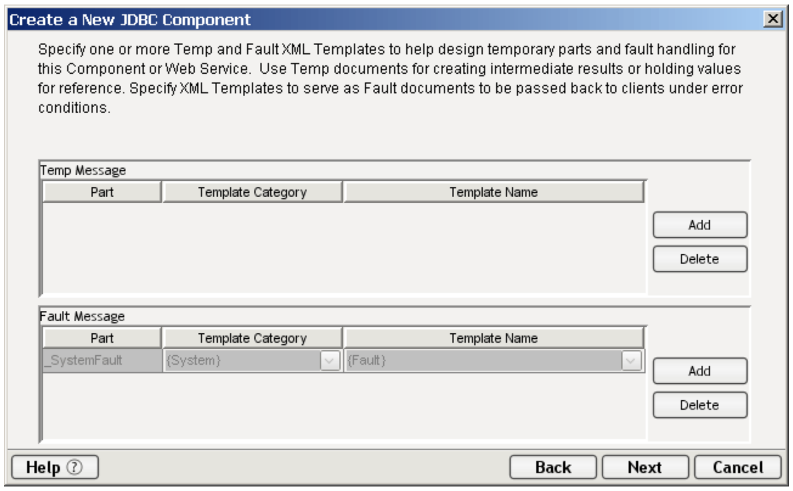
6 Specify the Input and Output templates as follows.

- Type in a name for the template under **Part** if you wish the name to appear in the DOM as something other than "Input".
- Select a **Template Category** if it is different than the default category.
- Select a **Template Name** from the list of XML templates in the selected **Template Category**.
- To add additional input XML templates, click **Add** and choose a **Template Category** and **Template Name** for each.
- To remove an input XML template, select an entry and click **Delete**.

7 Select an XML template for use as an Output DOM using the same steps outlined above.

NOTE: You can specify an input or output XML template that contains no structure by selecting {System}{ANY} as the Input or Output template. For more information, see *Creating an Output Document without Using a Template* in the *Composer User's Guide*.

8 Click **Next**. The Temp and Fault XML Template panel appears.


Create a New JDBC Component

Specify one or more Temp and Fault XML Templates to help design temporary parts and fault handling for this Component or Web Service. Use Temp documents for creating intermediate results or holding values for reference. Specify XML Templates to serve as Fault documents to be passed back to clients under error conditions.

Temp Message

Part	Template Category	Template Name

Fault Message

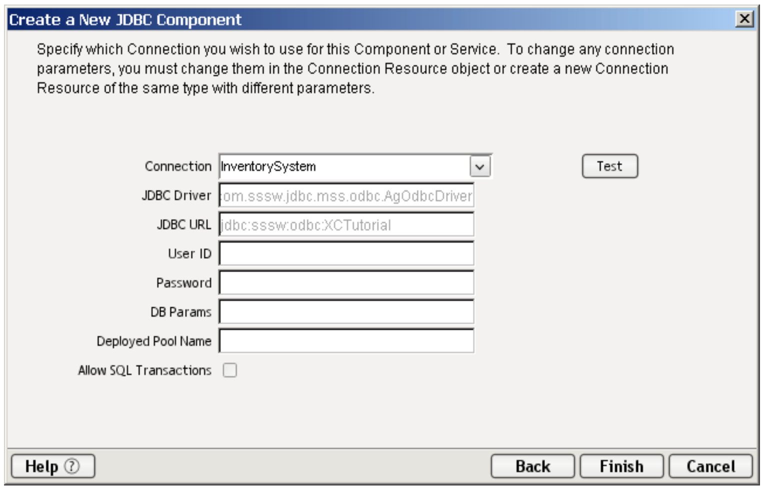
Part	Template Category	Template Name
_SystemFault	{System}	{Fault}

9 If desired, specify a template to be used as a scratchpad under the “Temp Message” pane of the dialog window. This can be useful if you need a place to hold scratchpad values that will be used temporarily during the execution of your component. Select a **Template Category i**f it is different than the default category. Then select a **Template Name** from the list of XML templates in the selected Template Category.

10 Under the “Fault Message” pane, select an XML template to be used to pass back to clients when an error condition occurs.

11 As above, to add additional input XML templates, click **Add** and choose a Template Category and Template Name for each. Repeat as many times as desired. To *remove* an input XML template, select an entry and click **Delete**.

12 Click **Next.** The Connection Info panel of the “Create a New PeopleSoft Component” Wizard appears.


Create a New JDBC Component

Specify which Connection you wish to use for this Component or Service. To change any connection parameters, you must change them in the Connection Resource object or create a new Connection Resource of the same type with different parameters.

Connection	InventorySystem
JDBC Driver	om.sssw.jdbc.mss.odbc.AgOdbcDriver
JDBC URL	jdbc:sssw:odbc:XCTutorial
User ID	
Password	
DB Params	
Deployed Pool Name	
Allow SQL Transactions	☐

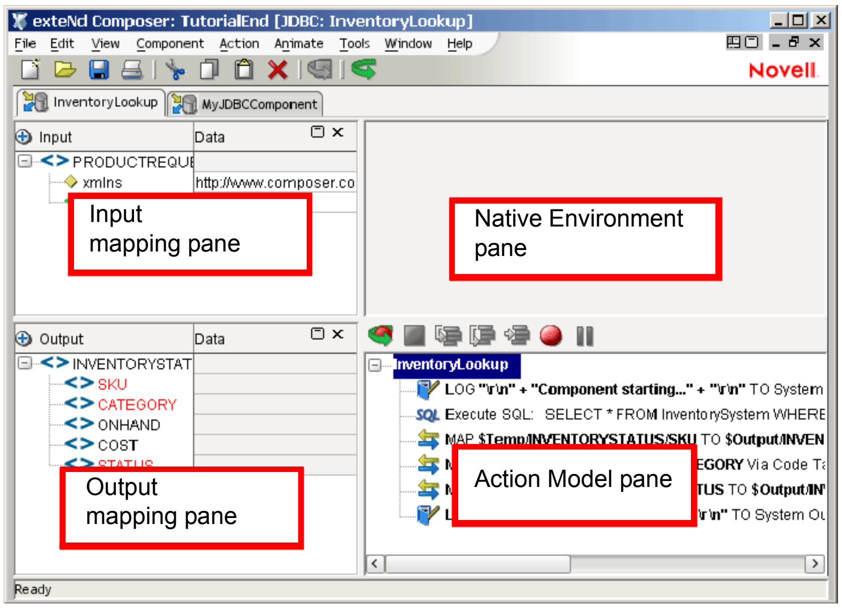
13 Select a **PeopleSoft Connection** from the **Connection** list.

14 Click **Finish**. The component is created and the PeopleSoft Component Editor appears.

About the PeopleSoft Component Editor Window

The PeopleSoft Component Editor includes all the functionality of the XML Map Component Editor. It contains mapping panes for Input and Output XML documents as well as an Action Model.

The PeopleSoft Component Editor also includes a Native Environment Pane, which appears as a grey pane until you create a PeopleSoft Request action, at which time it will show a tabbed Request/Response pane with corresponding XML trees.



(The Native Environment Pane shown here is blank, because no PeopleSoft Request action has yet been created.)

Creating Actions in the Component Editor

You can create all the normal Composer actions in the Action Model of your PeopleSoft component (e.g., XML Map, Function, Log, Send Mail, and so on). In addition, you can create a PeopleSoft Request action.

The PeopleSoft Request action communicates requests (XML request documents) to your PeopleSoft system and fetches responses back from the same system.

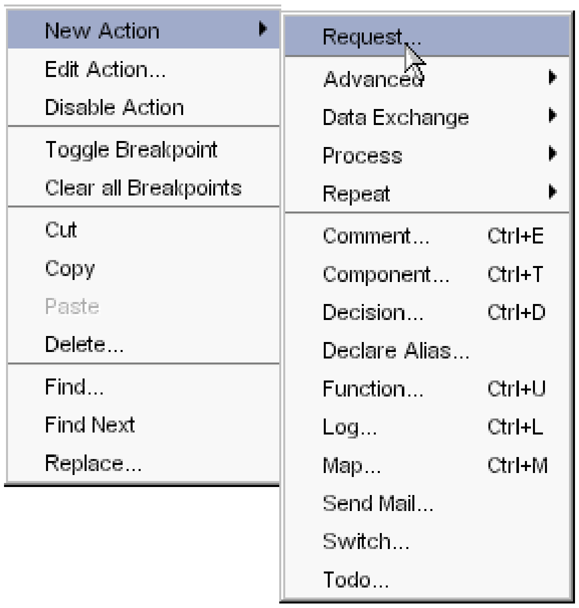
The Composer GUI for creating a PeopleSoft Request action allows you to

- specify the type of PeopleSoft function (or “service”) you want to invoke
- automatically generate XML schemas (request and response) for the particular Component Interface API you wish to use
- use the generated schemas to fine-tune the exact structure of the actual request document you want to use (a request document that conforms to the schema but reflects your own “setup” choices and initialization parameters, etc.)
- automatically generate the corresponding XML request and response documents that conform to the schemas and choices you’ve made

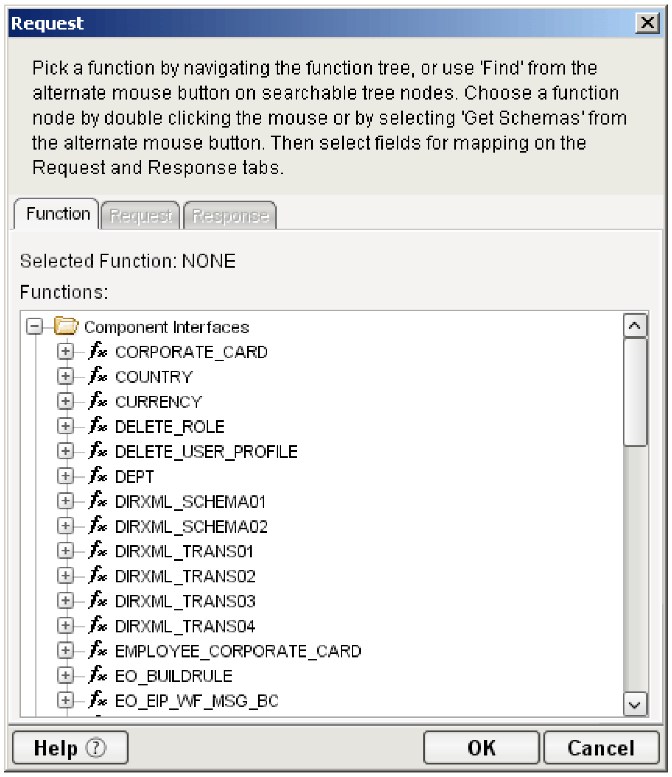
You will see how this works in the procedure given below.

➢ To create a PeopleSoft Request action

1 Right-click in the Action Model at the location where you wish to create a new PeopleSoft Request action, then (in the context menu that appears) choose **New Action** > **Request**.



2 In the dialog that appears (see below), give focus to the Functions tab and expand the topmost node (Component Interfaces) in the tree-view shown.



Pick a function by navigating the function tree, or use 'Find' from the alternate mouse button on searchable tree nodes. Choose a function node by double clicking the mouse or by selecting 'Get Schemas' from the alternate mouse button. Then select fields for mapping on the Request and Response tabs.

Selected Function: NONE

Functions:

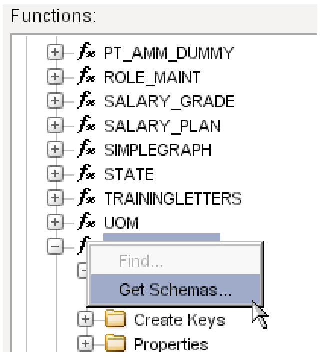
- Component Interfaces
  - CORPORATE_CARD
  - COUNTRY
  - CURRENCY
  - DELETE_ROLE
  - DELETE_USER_PROFILE
  - DEPT
  - DIRXML_SCHEMA01
  - DIRXML_SCHEMA02
  - DIRXML_TRANS01
  - DIRXML_TRANS02
  - DIRXML_TRANS03
  - DIRXML_TRANS04
  - EMPLOYEE_CORPORATE_CARD
  - EO_BUILDRULE
  - EO_EIP_WF_MSG_BC

You will see a list of functions corresponding to those for which Component Interface classes were built (see "Deploying the PeopleSoft Component Interface Files" on page 17 and "How to Create a New Component Interface" on page 63).

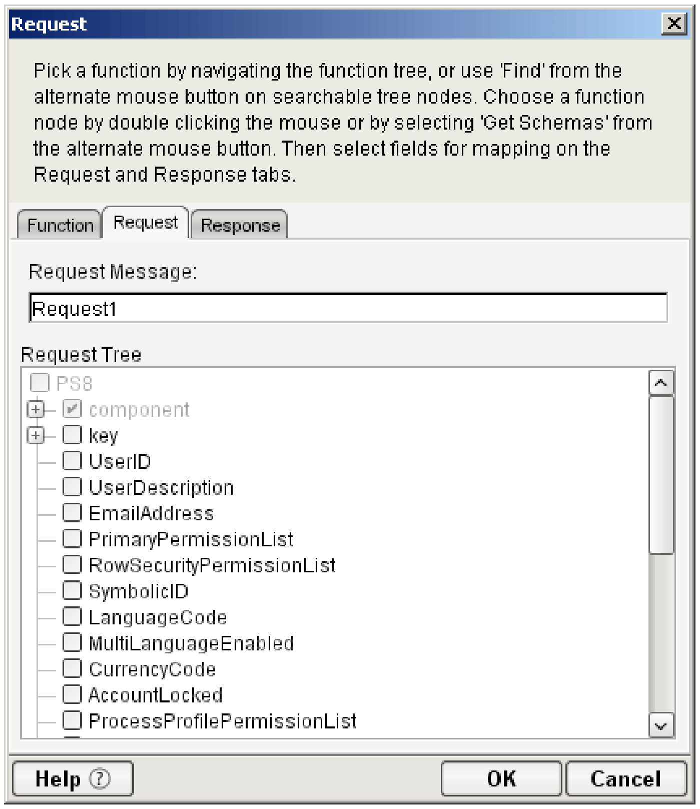
- All nodes have tooltips specifying the node type:

- Folder and Searchable Folder nodes: “type – Folder”
- Function nodes: “type – EventOp”, “type – Operation”, “type - Message”

3 Right-click on the function of interest. Select **Get Schemas** from the context menu that appears. (If the command is not available, it means no Component Interface API classes are available for this Function type.)



4 After a few seconds, the Request and Response tabs of the dialog should become enabled. When this has occurred, select the Request tab to bring it forward.



Pick a function by navigating the function tree, or use 'Find' from the alternate mouse button on searchable tree nodes. Choose a function node by double clicking the mouse or by selecting 'Get Schemas' from the alternate mouse button. Then select fields for mapping on the Request and Response tabs.

5 Enter a request-message (document) mapping name, or accept the default as shown.

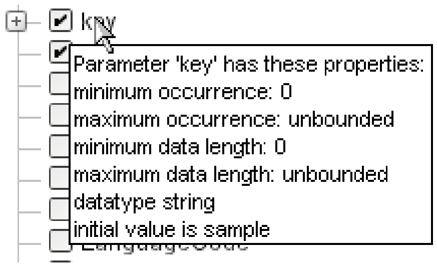
6 Select the elements that you want to occur in the Request document, by checking or unchecking checkboxes in the node tree. Some nodes are greyed out (disabled) while others are not. This is because the document’s schema determines which nodes you can edit.

Composer will not let you specify an invalid request document, because the things you *should not edit* are greyed out (disabled), and the editable items for which customization is allowed have RMB context-menus that non-customizable nodes do not have, etc. For example:

- *Mandatory* elements and attributes are marked *selected* and *disabled*.
- All child nodes are left unselected and disabled if the parent node is not selected.

- A parent element, once selected, will autoselect mandatory elements and attributes underneath it.
- When a parent node is selected, you will need to select *optional* element *manually.*

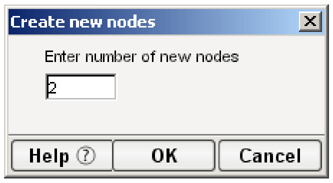
Tool tips are an important aid in using the Request and Response panes, because they display document-structure rules that come straight from the corresponding Component Interface request/response document schema. You should take time to familiarize yourself with the tooltip feature by letting the mouse hover over various nodes. An example is shown below.



Parameter 'key' has these properties:
minimum occurrence: 0
maximum occurrence: unbounded
minimum data length: 0
maximum data length: unbounded
datatype string
initial value is sample

NOTE: The tooltips that appear in this tree are also made available in the Native Environment pane.

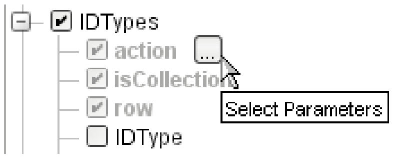
7 If a node has "maximum occurrence" of more than one, then a RMB menu command called "Create new..." is available. Choosing that command brings up the following dialog:



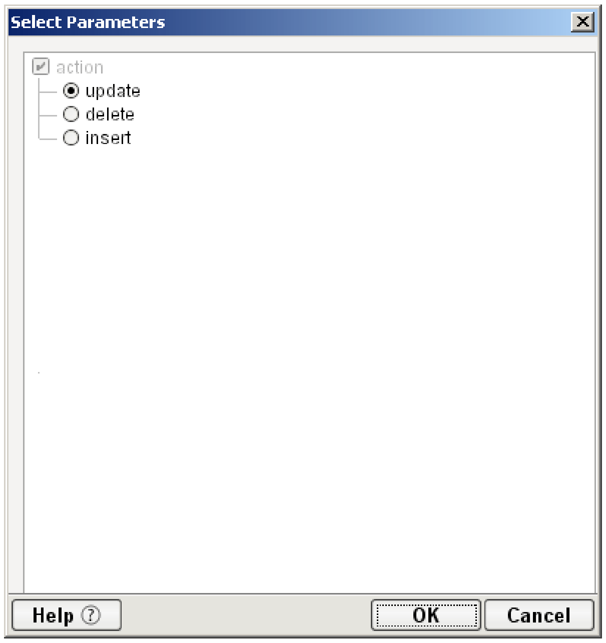
Enter the number of additional instances of this node-type that you want to insert in the document. Then click **OK**. New nodes are added to the document structure.

CAUTION: *There is currently no Undo for this.*

If an Enumeration Button appears to the right of a given node, you can click it to choose from the list of appropriate (schema-allowed) values for that enumeration.

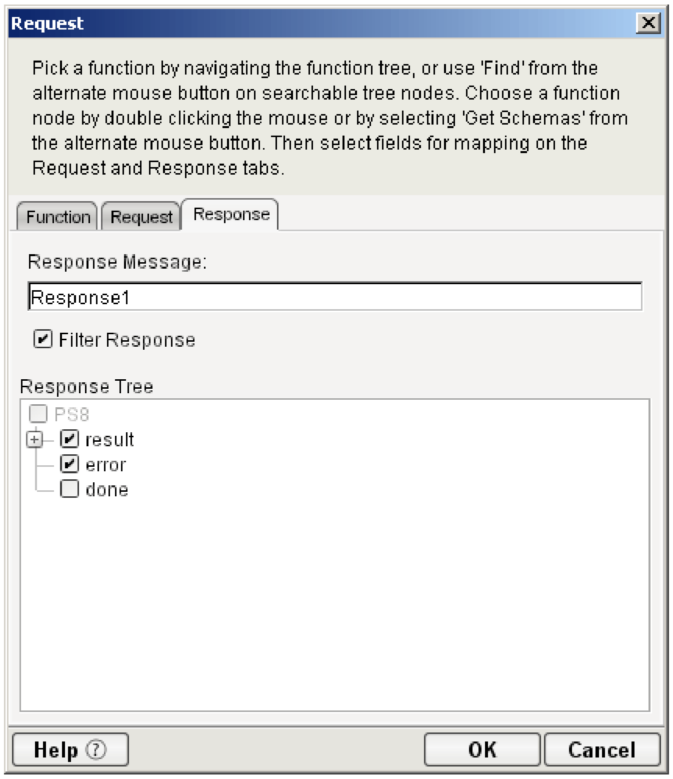


8 A common situation is that an enumeration offers choices of Update, Insert, or Delete. An example of this is shown in the following illustration, which shows the dialog that appears when you click an Enumeration Button.



9 Visit all of the nodes in the request tree that are of interest to you, and use the various Composer UI features to customize the request document's structure.

10 When you are satisfied with the structure, click the Response tab at the top of the dialog to bring the Response document pane forward. This pane shows a tree view of the response document, similar to the one shown for the Request document.


Request

Pick a function by navigating the function tree, or use 'Find' from the alternate mouse button on searchable tree nodes. Choose a function node by double clicking the mouse or by selecting 'Get Schemas' from the alternate mouse button. Then select fields for mapping on the Request and Response tabs.

Function | Request | Response

Response Message: Response1

[✓] Filter Response

Response Tree
PS8
[✓] result
[✓] error
[ ] done

11 Check or uncheck **Filter Response** as desired. It is checked by default. If this box is unchecked, the tree will be disabled and greyed (no longer editable), meaning that a default schema configuration will be used. When the box is checked, you can customize the structure of the request document by specifying various types of node filtering.

12 Visit all of the nodes in the Response document tree and customize the settings as desired. (See previous steps, applicable to the Request document, for an explanation of the UI semantics.)

13 Click **OK**. (Otherwise, use **Cancel**.) The dialog goes away and a new action appears in the Action Model of your component. At the same time, the Native Environment Pane changes to show the Request and Response tabs along with corresponding tree views of the fully customized document(s).

14 In the normal component editor view, drag and drop data from the Input document pane to the Request document of the Native Environment Pane, as desired. This will create mappings from your Input document to the Request doc that will be submitted to the PeopleSoft server. The mappings will appear (automatically) as new XML Map Actions in the Action Model.

15 Repeat the drag-and-drop mapping process for the Response document, dragging nodes from the Native Environment Pane to the component's Output document tree view. Again, this will automatically cause new Map Actions to appear in your Action Model. (You can cut/copy/delete the actions once they appear in the Action Model pane.)

16 Add or remove actions from the Action Model as needed to create the business logic you need.

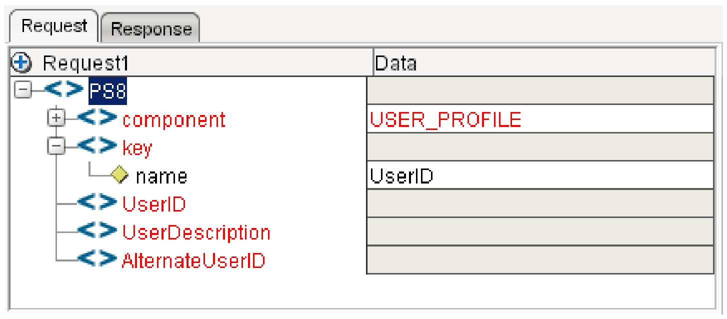
Returning to Schema-Edit Mode

Once you have created a PeopleSoft Request Action in your action model (using the above procedure), you can go back and revisit the Request and Response schema trees (and change your customizations) simply by doubleclicking the Request Action in your action model. Doubleclicking will cause the PeopleSoft Request Action dialog to reappear. You can use the Request and Response tabs to navigate the request- and response-doc schemas and change your customizations as desired. When you exit out of the dialog, your changes will be reflected in the Native Environment Pane.

NOTE: If you make hand-edits to the Request or Response documents in your Native Environment Pane, you will not be able to reopen the PeopleSoft Request Action dialog. (See discussion further below.) Therefore, try to avoid manually editing request/response documents. Instead, make modifications through careful use of the XML Map Action or by using Request.createXPath( ) and/or DOM methods in a Function Action.

Request and Response Documents

After you have created the PeopleSoft Request Action using the foregoing procedure, the Native Environment Pane portion of the Composer editor view will show Request and Response documents in tree-view form.

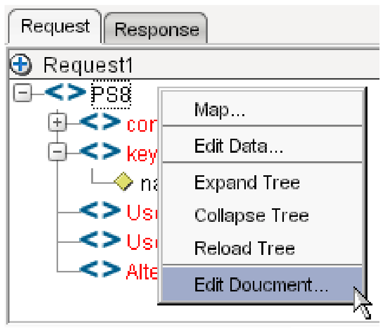


Request1	Data
PS8	
component	USER_PROFILE
key	
name	UserID
UserID	
UserDescription	
AlternateUserID	

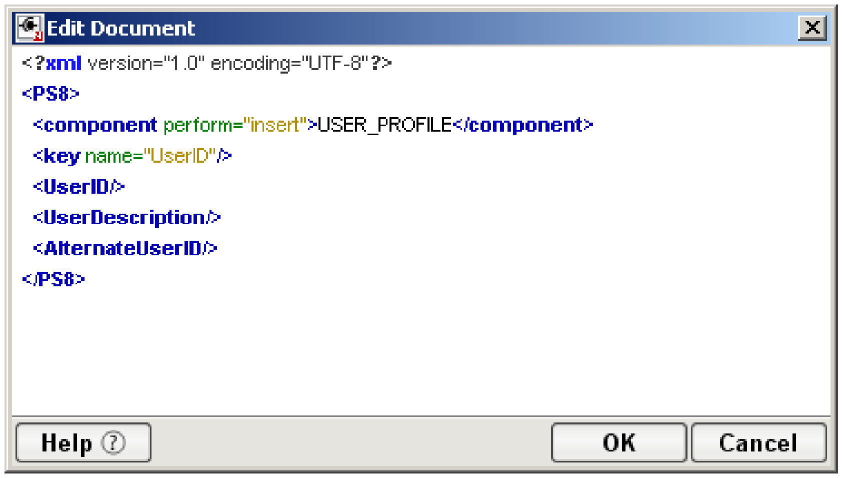
As mentioned earlier, you can drag and drop data from your component's Input document to the Request document to create XML Map actions. (First, put focus on the target Request-document node by clicking in it. Then go to the Input document and click-drag any data you want to map over to the target node in the Request document.) You can use drag-and-drop mapping to map from Response to Output, as well. In the illustration shown above, some nodes are shown in red. The red entries represent nodes to which data have been mapped.

Manually Editing the Request and Response Docs

If you right-click anywhere inside the Request or Response tree views in the Native Environment Pane, you will get a context menu.



The **Edit Document** command in this menu will open the document in a text-editor view as shown below.

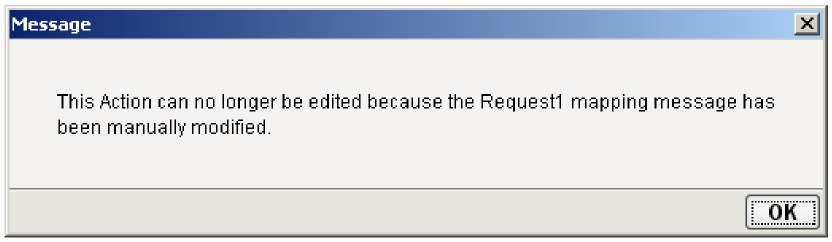


```
<?xml version="1.0" encoding="UTF-8"?>
<PS8>
  <component perform="insert">USER_PROFILE</component>
  <key name="UserID"/>
  <UserID/>
  <UserDescription/>
  <AlternateUserID/>
</PS8>
```

In this window, you can hand-edit the Request or Response document as desired.

CAUTION: *If you make hand-edits to the document, it may no longer conform to the original schema. As a result, you will no longer be able to reopen it in the PeopleSoft Request Action dialog.*

Note that if you've made hand-edits to a Request or Response document, doubleclicking on the corresponding PeopleSoft Request Action in the action model will cause the following dialog to appear:



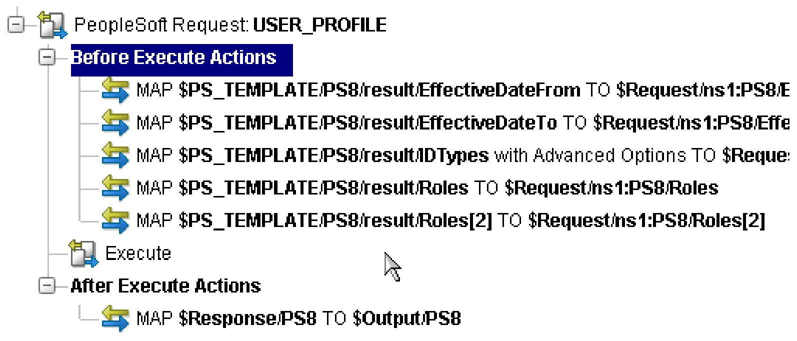
This Action can no longer be edited because the Request1 mapping message has been manually modified.

If this dialog appears, you will need to recreate the action in order to see the original Request or Response document. (Any hand edits will be lost, however.)

To prevent this situation from occurring, *avoid making manual edits to Request or Response documents*. Instead use XML Map Actions (and/or Function actions) to modify documents or node contents. Be aware that doing so may violate schema constraints. This will not necessarily cause runtime problems for your component when executing in the Composer Enterprise Server environment, because Composer does not validate the request document against the schema at runtime, on the server. However, your PeopleSoft system may validate incoming requests, in which case a hand-edited Request document could be the source of hard-to-troubleshoot errors.

"Before Execute" and "After Execute" Actions

You will notice that whenever a PeopleSoft Request action is created in your Action Model, two additional lines are added to the action list: Before Execute Actions and After Execute Actions. These are header labels (grouping labels) for blocks of actions that you want to occur *before* the request is sent to the PeopleSoft system or *after* a response document has come back, respectively. Typically you will want to map data fields from Input to Request and have those actions be grouped under the "Before Execute Actions" header. Likewise, any map actions or other action logic you need to perform on the Response document after the PeopleSoft system executes you request should be grouped under the "After Execute Actions" heading.



```
PeopleSoft Request: USER_PROFILE
  Before Execute Actions
    MAP $PS_TEMPLATE/PS8/result/EffectiveDateFrom TO $Request/ns1:PS8/E
    MAP $PS_TEMPLATE/PS8/result/EffectiveDateTo TO $Request/ns1:PS8/Effe
    MAP $PS_TEMPLATE/PS8/result/IDTypes with Advanced Options TO $Reque
    MAP $PS_TEMPLATE/PS8/result/Roles TO $Request/ns1:PS8/Roles
    MAP $PS_TEMPLATE/PS8/result/Roles[2] TO $Request/ns1:PS8/Roles[2]
  Execute
  After Execute Actions
    MAP $Response/PS8 TO $Output/PS8
```

Creating PeopleSoft Services

This section describes how to use the exteNd Composer™ Connect for PeopleSoft to create a PeopleSoft Service component to connect to PeopleSoft and listen for events. As part of the process of creating a PeopleSoft Service, you can select an existing PeopleSoft connection resource, or you can create a new one. If you create the connection beforehand, it is available for use by any PeopleSoft Services in the current project. *If you have not already created at least one PeopleSoft connection resource in the current project, you will be prompted to do so when you try to create a PeopleSoft Service.*

NOTE: You can still create a PeopleSoft Service without first creating a connection resource, but you won't be able to use any debug features that depend on a live connection.

➢ **To Create a New PeopleSoft Service:**

1 Select **File** > **New** > **xObject**. The New xObject dialog box is displayed.

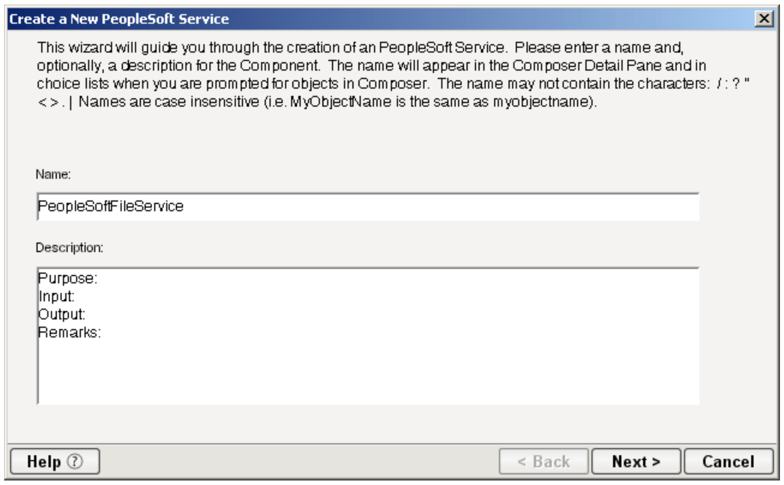
NOTE: Alternatively, under **Service** in the Composer Navigator pane (Explorer view), you can highlight **PeopleSoft Service**, click the right mouse button, then select **New**.

2 Select the **Process/Service** tab.

3 Double-click on "PeopleSoft Service."

If you have not previously defined a connection resource, you are prompted to do so now (see "Creating Connection Resources" on page 33).

If you have already defined a connection resource, the “Create a New PeopleSoft Service” Wizard is displayed.



This wizard will guide you through the creation of an PeopleSoft Service. Please enter a name and, optionally, a description for the Component. The name will appear in the Composer Detail Pane and in choice lists when you are prompted for objects in Composer. The name may not contain the characters: / : ? " < > . | Names are case insensitive (i.e. MyObjectName is the same as myobjectname).

Name:

PeopleSoftFileService

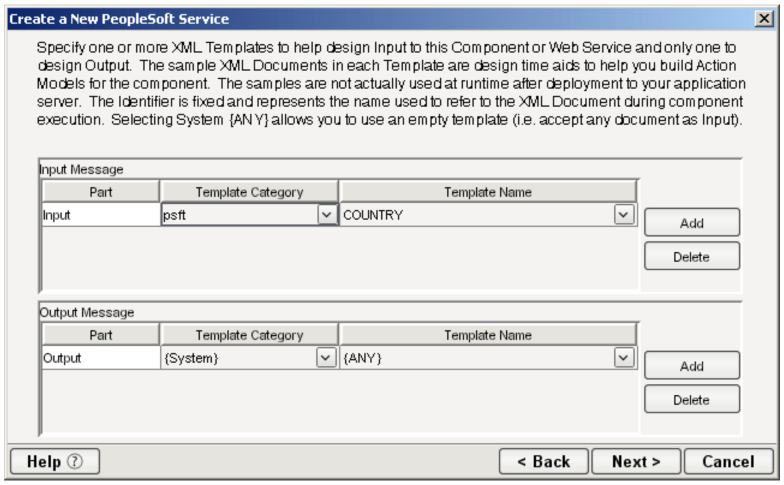
Description:

Purpose:
Input:
Output:
Remarks:

4 Type a **Name** for the new PeopleSoft Service.

5 Optionally, type **Description** text.

6 Select **Next**. The XML Input/Output Message Property Info panel of the New PeopleSoft Service Wizard is displayed.



Specify one or more XML Templates to help design Input to this Component or Web Service and only one to design Output. The sample XML Documents in each Template are design time aids to help you build Action Models for the component. The samples are not actually used at runtime after deployment to your application server. The Identifier is fixed and represents the name used to refer to the XML Document during component execution. Selecting System {ANY} allows you to use an empty template (i.e. accept any document as Input).

Input Message

Part	Template Category	Template Name
Input	psft	COUNTRY

Output Message

Part	Template Category	Template Name
Output	{System}	{ANY}

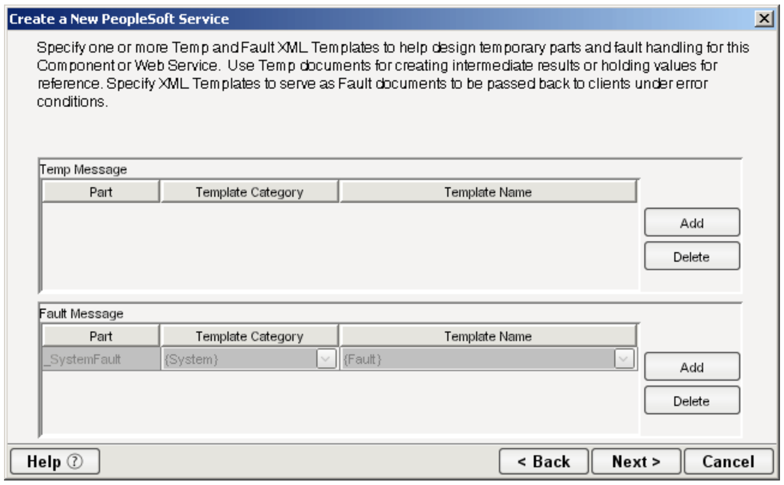
7 Specify the Input and Output templates as follows:

- Type a name for the template under **Part** if you wish the name to appear in the DOM as something other than “Input”
- Select a **Template Category** if it is different than the default category.
- Select a **Template Name** from the list of XML templates in the selected **Template Category**.
- To add additional input XML templates, click **Add**, then choose a **Template Category** and **Template Name** for each.
- To remove an input XML template, select an entry and click **Delete**.

8 Select an XML template for use as an Output DOM using the procedure described in the previous step.

NOTE: You can specify an input or output XML template that contains no structure by selecting {System}{ANY} as the Input or Output template. For more information, see “Creating an Output Document without Using a Template” in the *Novell exteNd Composer User’s Guide*.

9 Select **Next**. The Temp and Fault XML Template panel is displayed.


Create a New PeopleSoft Service

Specify one or more Temp and Fault XML Templates to help design temporary parts and fault handling for this Component or Web Service. Use Temp documents for creating intermediate results or holding values for reference. Specify XML Templates to serve as Fault documents to be passed back to clients under error conditions.

Temp Message

Part	Template Category	Template Name

Fault Message

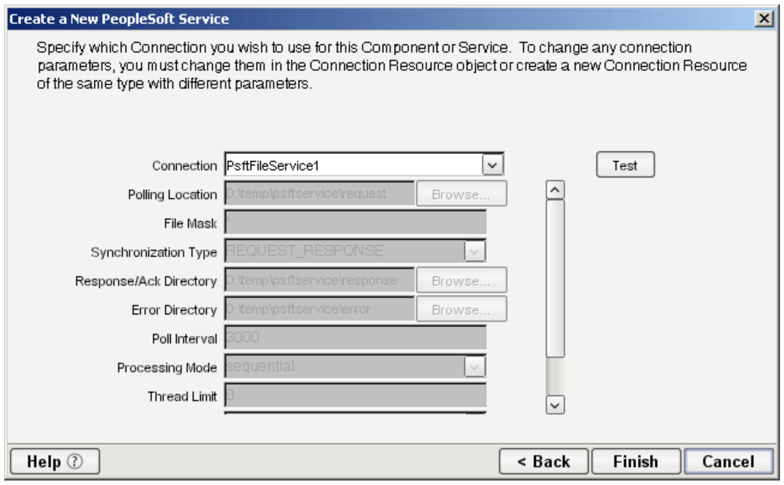
Part	Template Category	Template Name
_SystemFault	{System}	{Fault}

10 If desired, specify a template to be used as a scratch pad under the “Temp Message” pane of the dialog box. This can be useful if you need a place to hold values that will be used temporarily during the execution of your component. Select a **Template Category** if it is different than the default category. Then select a **Template Name** from the list of XML templates in the selected Template Category.

11 Under the “Fault Message” pane, select an XML template to be used to pass back to clients when an error condition occurs.

12 To add additional input XML templates, click **Add** and choose a **Template Category** and **Template Name** for each. Repeat as many times as desired. To remove an input XML template, select an entry and click **Delete**.

13 Select **Next**. The Connection Info panel of the Create a New PeopleSoft Service wizard is displayed.


Create a New PeopleSoft Service

Specify which Connection you wish to use for this Component or Service. To change any connection parameters, you must change them in the Connection Resource object or create a new Connection Resource of the same type with different parameters.

Connection	PsftFileService1
Polling Location	D:\temp\psftservice\request
File Mask	*
Synchronization Type	REQUEST_RESPONSE
Response/Ack Directory	D:\temp\psftservice\response
Error Directory	D:\temp\psftservice\error
Poll Interval	3000
Processing Mode	sequential
Thread Limit	3

14 Select a PeopleSoft File, HTTP, or TCP Service connection from the **Connection** list. The Connection list displays the names of the PeopleSoft connection resources that have been defined in this project.

15 Select **Finish**. The component is created and the PeopleSoft Service Editor is displayed. The features of this window are identical to the features described in “About the PeopleSoft Component Editor Window” on page 41. The following sections describe the steps required to create a PeopleSoft Service Action.

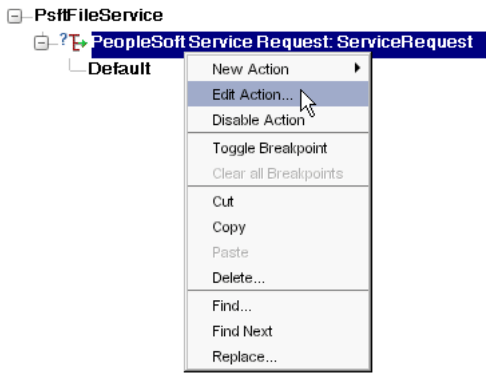
Creating PeopleSoft Service Actions

You can create all the normal Composer actions in the Action Model of your PeopleSoft Service (e.g., XML Map, Function, Log, Send Mail). In addition, you can create a PeopleSoft Service Request action.

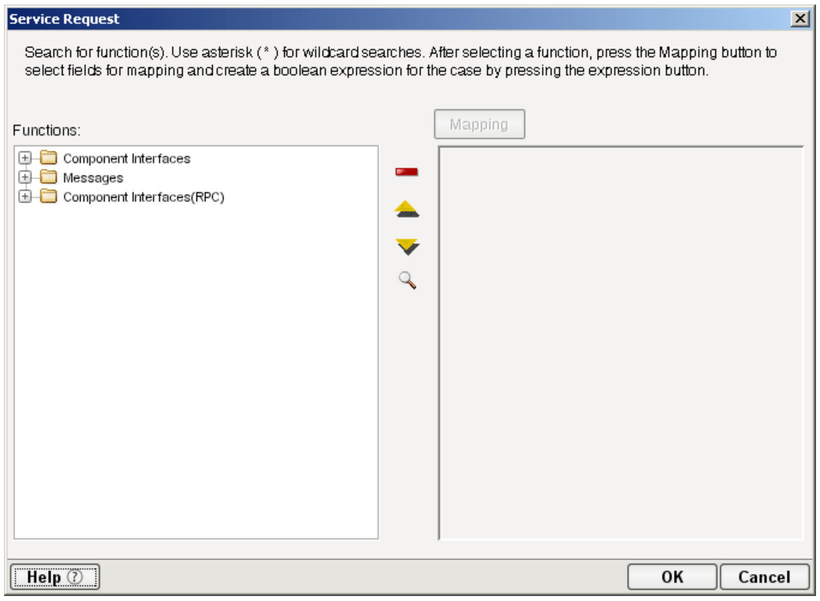
The PeopleSoft Service Request action is similar to a Switch action (see "The Switch Action" in the in the *Novell exteNd Composer User's Guide).* A Switch action allows program control to branch to a block of actions based on a match between an input value and a Case value. In a PeopleSoft Service Request, Composer receives a DOM, and compares a series of cases against the DOM. If an exact match occurs between the DOM and a case, execution branches to the actions listed underneath the case in the Action model. Cases are tested in the order listed; and once a match is found, execution of the match logic precludes execution of any other logic in the PeopleSoft Service Request.

➢ To Create a PeopleSoft Service Request Action

1 In the Action Model, right-click on PeopleSoft Service Request:

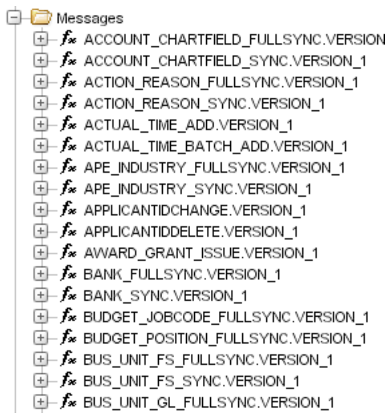


2 Select **Edit Action**. The Service Request dialog box is displayed.

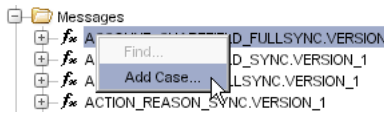


Search for function(s). Use asterisk (*) for wildcard searches. After selecting a function, press the Mapping button to select fields for mapping and create a boolean expression for the case by pressing the expression button.

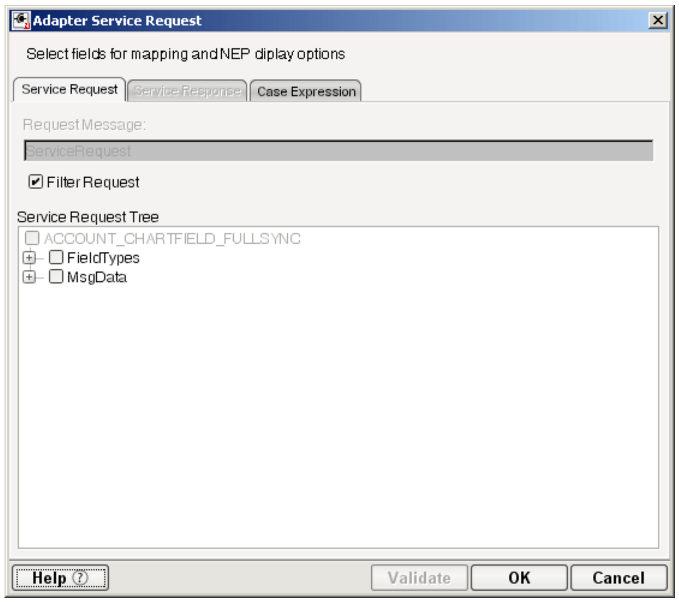
3 In the Function tree, click on the plus (+) sign to the left of the Messages node to expand the tree view for that node.


Messages
ACCOUNT_CHARTFIELD_FULLSYNC.VERSION
ACCOUNT_CHARTFIELD_SYNC.VERSION_1
ACTION_REASON_FULLSYNC.VERSION_1
ACTION_REASON_SYNC.VERSION_1
ACTUAL_TIME_ADD.VERSION_1
ACTUAL_TIME_BATCH_ADD.VERSION_1
APE_INDUSTRY_FULLSYNC.VERSION_1
APE_INDUSTRY_SYNC.VERSION_1
APPLICANTIDCHANGE.VERSION_1
APPLICANTIDDELETE.VERSION_1
AWARD_GRANT_ISSUE.VERSION_1
BANK_FULLSYNC.VERSION_1
BANK_SYNC.VERSION_1
BUDGET_JOBCODE_FULLSYNC.VERSION_1
BUDGET_POSITION_FULLSYNC.VERSION_1
BUS_UNIT_FS_FULLSYNC.VERSION_1
BUS_UNIT_FS_SYNC.VERSION_1
BUS_UNIT_GL_FULLSYNC.VERSION_1

4 Right-click on the message name that you want to use for input and select **Add Case,** or double-click on the message name.



The Adapter Service Request dialog box is displayed. This dialog box is similar to the PeopleSoft Request dialog box (see "Creating Actions in the Component Editor" on page 42).


Adapter Service Request

Select fields for mapping and NEP diplay options

Service Request | Service Response | Case Expression

Request Message:
ServiceRequest

☑ Filter Request

Service Request Tree
☐ ACCOUNT_CHARTFIELD_FULLSYNC
☐ FieldTypes
☐ MsgData

5 Select or deselect **Filter Request**, as desired. Filter Request is selected by default. If deselected, the tree will be disabled and greyed (no longer editable), indicating that a default schema configuration will be used. If selected, you can customize the structure of the request document by selecting the nodes to be included in the request document.

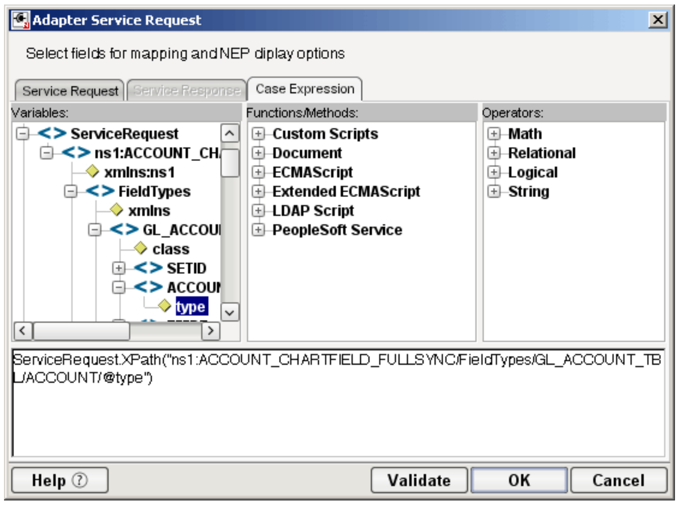
6 Expand the nodes of the Service Request Tree by clicking on the plus signs to the left of the nodes.

7 Select the elements that you want to include in the Service Request document by selecting or deselecting the check boxes in the node tree. Some nodes are greyed out (disabled). This is because the document schema determines the nodes that you can edit.

8 If the **Service Response** tab is enabled (this depends on the Synchronization Type setting in the Connection resource), click the **Service Response** tab at the top of the dialog box to bring the Service Response document pane forward. This pane shows a tree view of the Response document, similar to the one shown for the Request document. The same user interface features that applied to the Request document pane apply to the Response document pane.

9 Visit all of the nodes in the Response document tree and customize the settings as desired.

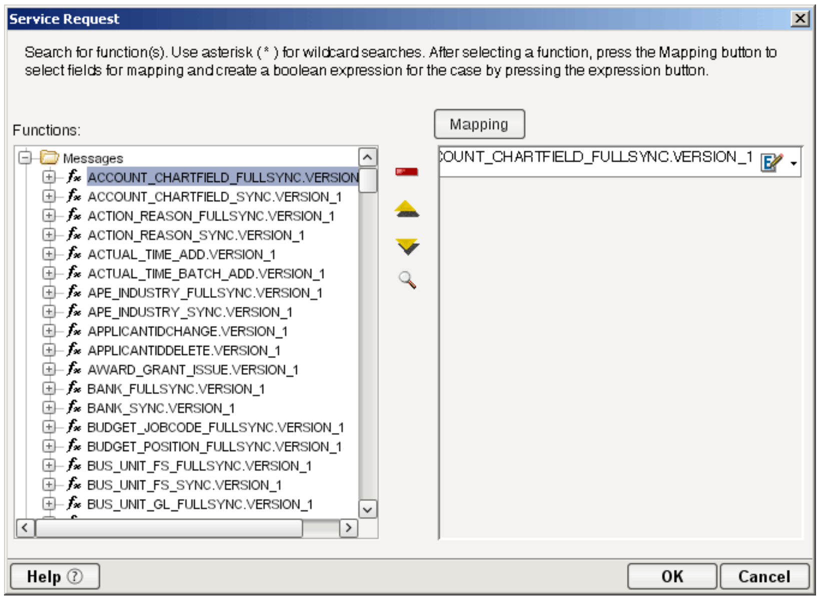
10 Click the **Case Expression** tab at the top of the dialog box to bring the Case Expression pane forward. This pane provides an Expression Builder that you use to build a case expression.



ServiceRequest.XPath("ns1:ACCOUNT_CHARTFIELD_FULLSYNC/FieldTypes/GL_ACCOUNT_TBL/ACCOUNT/@type")

11 Type the static string values or the ECMAScript expressions that will be checked against the input DOM.

12 Select the OK button. The Service Request dialog box is displayed.


Service Request
Functions:
Mapping
Messages
ACCOUNT_CHARTFIELD_FULLSYNC.VERSION
ACCOUNT_CHARTFIELD_SYNC.VERSION_1
ACTION_REASON_FULLSYNC.VERSION_1
ACTION_REASON_SYNC.VERSION_1
ACTUAL_TIME_ADD.VERSION_1
ACTUAL_TIME_BATCH_ADD.VERSION_1
APE_INDUSTRY_FULLSYNC.VERSION_1
APE_INDUSTRY_SYNC.VERSION_1
APPLICANTIDCHANGE.VERSION_1
APPLICANTIDDELETE.VERSION_1
AWARD_GRANT_ISSUE.VERSION_1
BANK_FULLSYNC.VERSION_1
BANK_SYNC.VERSION_1
BUDGET_JOBCODE_FULLSYNC.VERSION_1
BUDGET_POSITION_FULLSYNC.VERSION_1
BUS_UNIT_FS_FULLSYNC.VERSION_1
BUS_UNIT_FS_SYNC.VERSION_1
BUS_UNIT_GL_FULLSYNC.VERSION_1
COUNT_CHARTFIELD_FULLSYNC.VERSION_1
Help
OK
Cancel

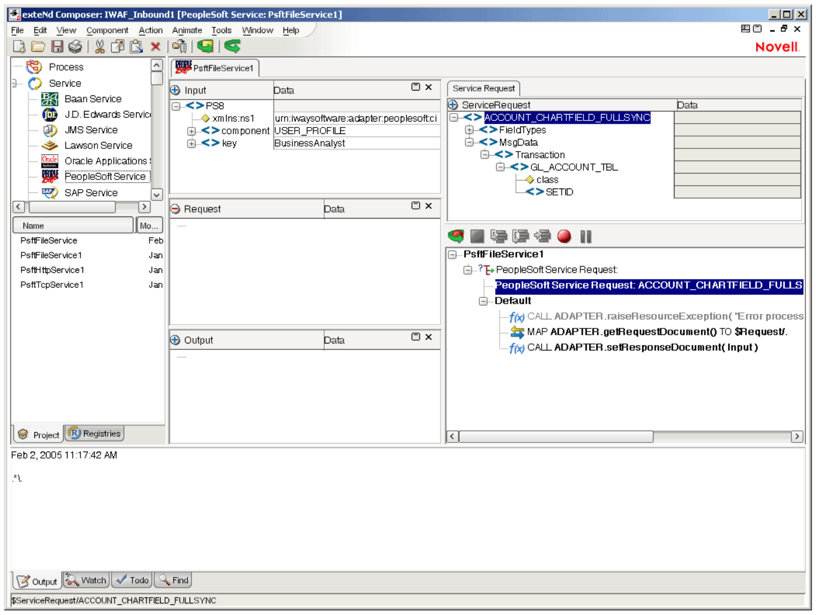
Search for function(s). Use asterisk (*) for wildcard searches. After selecting a function, press the Mapping button to select fields for mapping and create a boolean expression for the case by pressing the expression button.

13 To add another case, repeat Step 3 through Step 12. The cases are listed from top to bottom in the order in which they are evaluated. Each Case value will be checked in turn, in the order you list them.

- For optimal performance, list the most likely matches first.
- You can change the order in which the cases are listed by clicking on the name of a case, then clicking the up or down triangle icons.

- To edit a case, select the name of the case, then press the **Mapping** button. You can also select the name of the case, then select the Expression Builder button.

14 When you have finished adding cases, select **OK.** The Service Request dialog box closes and a new action appears in the Action Model of your component. The Native Environment Pane changes to show the Request and Response (if the Synchronization Type is set to REQUEST_RESPONSE in the Connection resource) tabs along with tree views of the request and response documents.



Name	Mo...
PsftFileService	Feb
PsftFileService1	Jan
PsftHttpService1	Jan
PsftTcpService1	Jan

Input	Data
PS8	
xmlns:ns1	urn:iwaysoftware:adapter:peoplesoft:ci
component	USER_PROFILE
key	BusinessAnalyst

Request	Data

Output	Data

ServiceRequest	Data
ACCOUNT_CHARTFIELD_FULLSYNC	
FieldTypes	
MsgData	
Transaction	
GL_ACCOUNT_TBL	
class	
SETID	

```
PsftFileService1
  PeopleSoft Service Request:
    PeopleSoft Service Request: ACCOUNT_CHARTFIELD_FULLS
    Default
      CALL ADAPTER.raiseResourceException( "Error process
      MAP ADAPTER.getRequestDocument() TO $Request/.
      CALL ADAPTER.setResponseDocument( Input )
```

Feb 2, 2005 11:17:42 AM

.*\.

$ServiceRequest/ACCOUNT_CHARTFIELD_FULLSYNC

15 In the Action Model, add actions to be performed to each case to create the desired business logic.

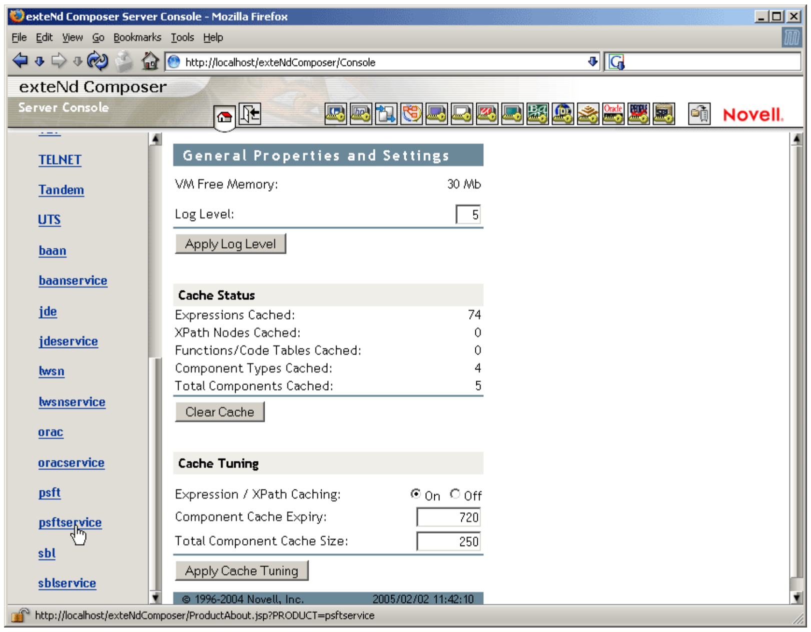
- To map a node of the Input document to a node in the Request document, drag a node from the Input pane and drop it on a node in the Request tab of the Native Environment pane. This automatically creates a new XML Map Action in the Action Model.
- To map a node of the Response document to a node in the Output document, drag a node from the Response tab of the Native Environment pane and drop it on a node in the Output pane. This automatically creates a new XML Map Action in the Action Model. You can cut, copy, or delete the actions once they appear in the Action Model pane.
- When all of the Actions for a case have been evaluated, Composer sets the Service Response DOM.
- The final case in the Action model is always labeled Default. This case is generated automatically and cannot be removed. Actions placed under Default are executed if and only if the none of the other cases are matched. While you are not required to place actions under Default, it is good programming practice to have at least some kind of fallback logic for the Default case, even if it's only a Log action or a Raise Error action.
- The Request and Response document displays for Service Requests work similarly to the displays for Requests (see "Request and Response Documents" on page 47).

Managing Deployed PeopleSoft Services

Once a project containing PeopleSoft Services has been deployed (see "Deploying Your Project" in the *Novell exteNd Composer User's Guide)*, the Listener objects actively listen for messages each time you start your server. To manually start and stop these services you need to use the exteNd PeopleSoft Services Console. This browser-based console allows you to see the list of PeopleSoft Services, the status of each service (running or not running), the running tally (Count) of messages received, other administrative information, and a Start/Stop button.

➢ **To display the exteNd PeopleSoft Services Console:**

1. Make sure that your application server is running.
2. From the Windows Start menu, select **Programs** > **Novell Extend 5.2** > **Composer** > **Composer Server Console**. A dialog box for entering the application server administrator ID and password is displayed.
3. Type your administrator ID and password, then select **OK**. The Composer Server Console page is displayed.
4. Scroll down the **About Products** list on the left side of the page until you see the link titled "psftservice".


exteNd Composer Server Console - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/exteNdComposer/Console
exteNd Composer
Server Console
Novell.
TELNET
Tandem
UTS
baan
baanservice
jde
jdeservice
lwsn
lwsnservice
orac
oracservice
psft
psftservice
sbl
sblservice
Apply Log Level
Clear Cache
Apply Cache Tuning
http://localhost/exteNdComposer/ProductAbout.jsp?PRODUCT=psftservice

General Properties and Settings

VM Free Memory:	30 Mb
Log Level:	5

Cache Status

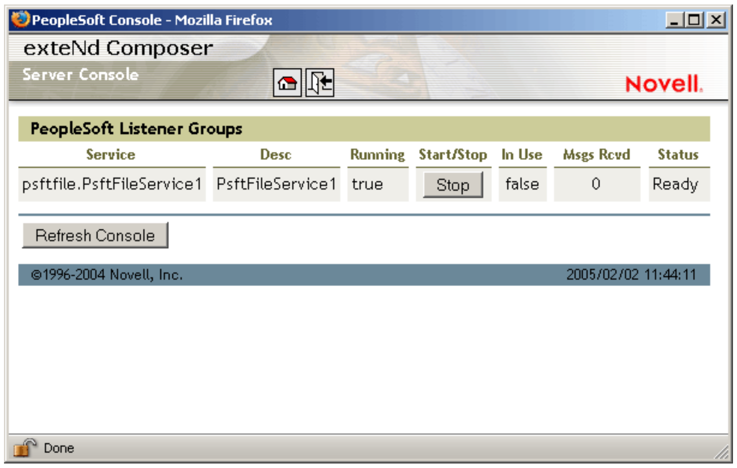
Expressions Cached:	74
XPath Nodes Cached:	0
Functions/Code Tables Cached:	0
Component Types Cached:	4
Total Components Cached:	5

Cache Tuning

Expression / XPath Caching:	[✓ On] Off
Component Cache Expiry:	720
Total Component Cache Size:	250

© 1996-2004 Novell, Inc. 2005/02/02 11:42:10

5. Select the "psftservice" link. A page is displayed that provides information (e.g., version, license number, copyright) about the Connect. It also displays a Console button.
6. Select the Console button. The PeopleSoft Service console page is displayed. This page lists any PeopleSoft Services that have been deployed.



PeopleSoft Listener Groups

Service	Desc	Running	Start/Stop	In Use	Msgs Rcvd	Status
psftfile.PsftFileService1	PsftFileService1	true	Stop	false	0	Ready

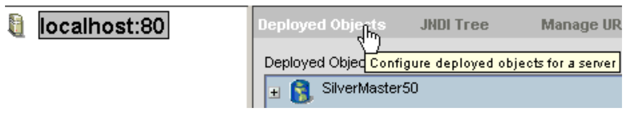
©1996-2004 Novell, Inc. 2005/02/02 11:44:11

7. To stop a PeopleSoft Service, select the appropriate **Stop** button (the button will then change to Start).

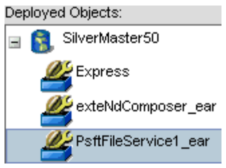
NOTE: If messages are being handled by a service at the time of the **Stop** command, there may be some delay before the service actually exits. Select the **Refresh** button periodically until the "Running" column of the console says No for the service.

➢ **To undeploy a PeopleSoft Service:**

1. Make sure that your application server is running.
2. From the Windows Start menu, select **Programs > Novell Extend 5.2 > AppServer > Server Management Console**.
3. Login as an administrator (**File > Login**).
4. Select the **Deployment** icon ( )from the toolbar.
5. Select **Deployed Objects**:



6. Expand the database containing the deployed objects that you want to manage:



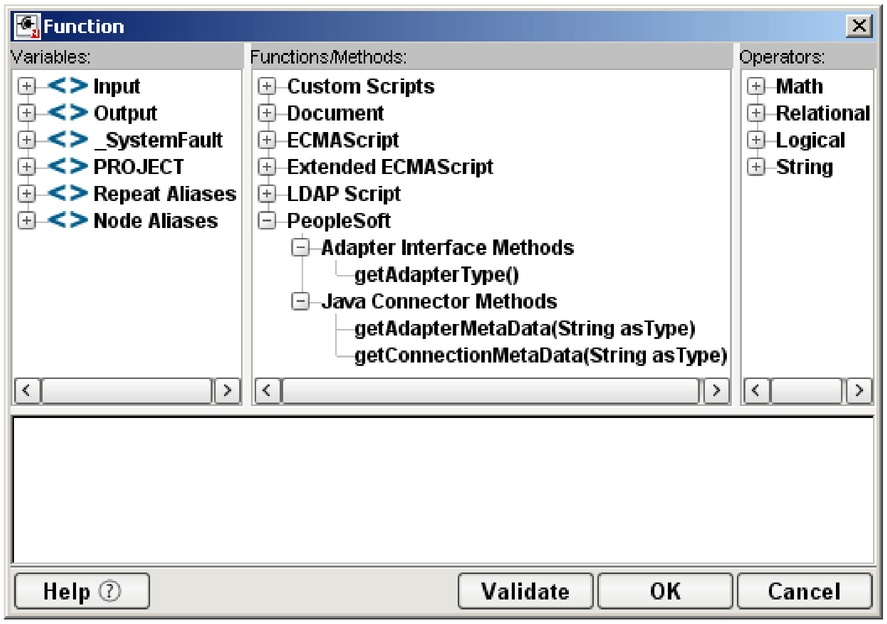
7. Select the object that you want to undeploy.
8. Select the **Undeploy** button.

ECMAScript Extensions

See Appendix A to this guide for a list of ECMAScript extension methods that can be used with the PeopleSoft Connect.

A ECMAScript Methods

The J2EE Connector framework of the exteNd Composer Connect for PeopleSoft implements a number of ECMAScript extension methods that are available for use through the Expression Builder (which is available in Function Actions and elsewhere). You will find the methods listed in the Functions/Methods pane of the Expression Builder



Hover-help is available for each method.

For more information on working with the Expression Builder, see the Scripting chapter of the *Composer User's Guide*.

Adapter Interface Methods

getAdapterType()

This method calls returns the adapter type name (e.g., "psft") as a String.

Syntax:
getAdapterType()

Connection Interface Methods

The following CCI API metadata-wrapped methods are published through the Expression Builder.

getAdapterMetaData()

This method wraps calls on the javax.resource.cci.ResourceAdapterMetaData class.

Syntax:
getAdapterMetaData(String *parameter*)

Parameter	Description
NAME	Returns the name of the adapter (e.g., “iWay JCA adapter”).
EXECUTEWITHI	Returns true if the implementation class for the Interaction interface implements public Record execute method; otherwise the method returns false.
EXECUTEWITHIO	Returns true if the implementation class for the Interaction interface implements public boolean execute method; otherwise the method returns false.
VENDOR	Returns the name of the vendor that has provided the resource adapter (e.g., “iWay Software, Inc.”.
VERSION	Returns the version of the resource adapter (e.g., “5.5”).
INTERACTIONSPECS	Returns the fully-qualified name of the InteractionSpec type supported by this resource adapter (e.g., “com.ibi.afjca.IWAFInteractionSpec”).
SPECVERSION	Returns a string representation of the version of the connector architecture specification that is supported by the resource adapter (e.g., “1.0”).
LOCALTX	Returns true if the resource adapter implements the LocalTransaction interface and supports local transaction demarcation on the underlying EIS (Enterprise Information System) instance through the LocalTransaction interface.

getConnectionMetaData()

This method wraps calls on the javax.resource.cci.ConnectionMetaData class.

Syntax:
getConnectionMetaData(String *parameter*)

Parameter	Description
EISPRODUCTNAME	Returns the product name of the underlying EIS instance (e.g., “PeopleSoft”).
EISPRODUCTVERSION	Returns product version of the underlying EIS instance (e.g., “Supports PeopleSoft 8.”).
USER	Returns the user name for an active connection as known to the underlying EIS instance.

Additional Methods

The following ECMA-wrapped methods are not displayed by the Expression Builder user interface, but you can use them in the Expression Builder by typing them directly into expressions.

getWarnings()

Returns line separated list of `javax.resource.cci.ResourceWarning` for the javax.resource.cci.Interaction.

clearWarnings()

Calls `javax.resource.cci.Interaction.clearWarnings().`

getLastError()

Returns last error, such as a `javax.resource.ResourceException`.

B Using Component Interfaces

This appendix describes how to create new component interfaces—and how to modify existing component interfaces—for use with PeopleSoft 8 and Novell exteNd Composer.

After generating and securing a component interface as described in this Appendix, you must generate its API, as described in “Building PeopleSoft Component Interface API Classes” on page 26 in this guide.

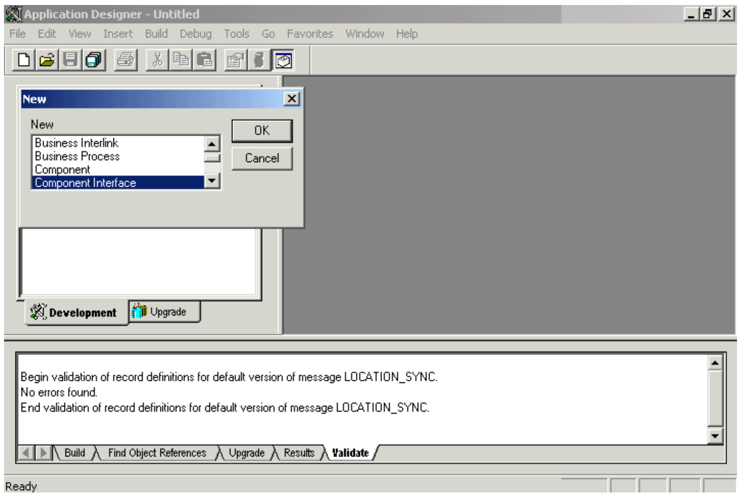
NOTE: This appendix is not a substitute for PeopleSoft documentation. For complete and up-to-date information about PeopleSoft integration APIs, see the PeopleSoft Online Library for your PeopleSoft system.

How to Create a New Component Interface

➤ **To create a component interface:**

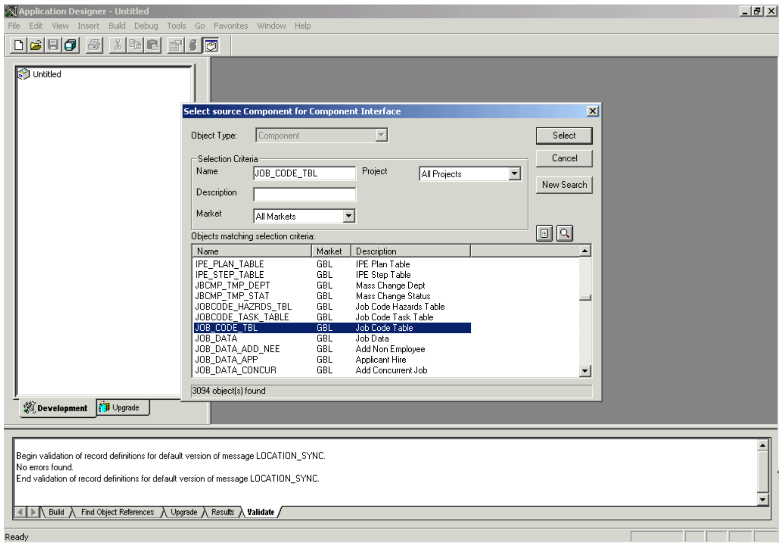
1. Launch the PeopleSoft Application Designer.
2. Choose **New** from the File menu.

   The New dialog box opens.



```
Begin validation of record definitions for default version of message LOCATION_SYNC.
No errors found.
End validation of record definitions for default version of message LOCATION_SYNC.
```

3. Select Component Interface and click **OK**. The Select Source Component for Component Interface dialog opens.


Application Designer - Untitled
File Edit View Insert Build Debug Tools Go Favorites Window Help
Untitled
Development
Upgrade
Build
Find Object References
Upgrade
Results
Validate
Ready

Select source Component for Component Interface

Object Type: Component

Selection Criteria
Name: JOB_CODE_TBL
Project: All Projects
Description:
Market: All Markets

Select | Cancel | New Search

Objects matching selection criteria:

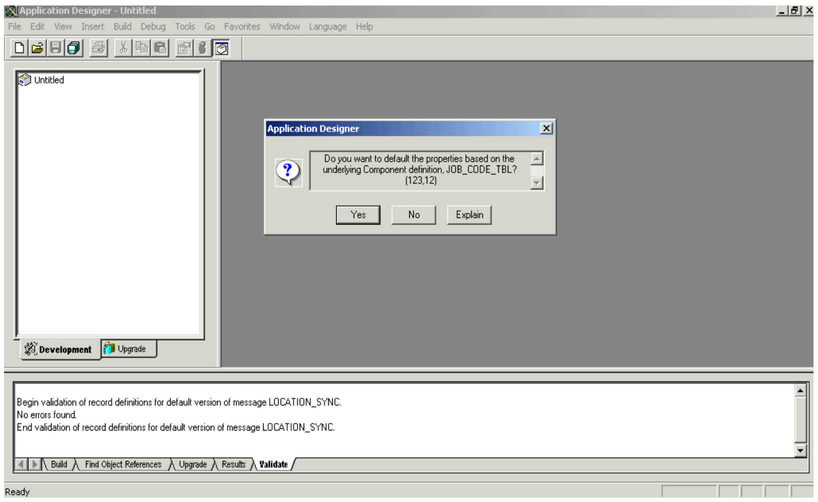
Name	Market	Description
IPE_PLAN_TABLE	GBL	IPE Plan Table
IPE_STEP_TABLE	GBL	IPE Step Table
JBCMP_TMP_DEPT	GBL	Mass Change Dept
JBCMP_TMP_STAT	GBL	Mass Change Status
JOBCODE_HAZRDS_TBL	GBL	Job Code Hazards Table
JOBCODE_TASK_TABLE	GBL	Job Code Task Table
JOB_CODE_TBL	GBL	Job Code Table
JOB_DATA	GBL	Job Data
JOB_DATA_ADD_NEE	GBL	Add Non Employee
JOB_DATA_APP	GBL	Applicant Hire
JOB_DATA_CONCUR	GBL	Add Concurrent Job

3094 object(s) found

```
Begin validation of record definitions for default version of message LOCATION_SYNC.
No errors found.
End validation of record definitions for default version of message LOCATION_SYNC.
```

4 Select the component to use as a basis for a Component Interface.

5 Click the **Select** button.The Application Designer dialog opens.



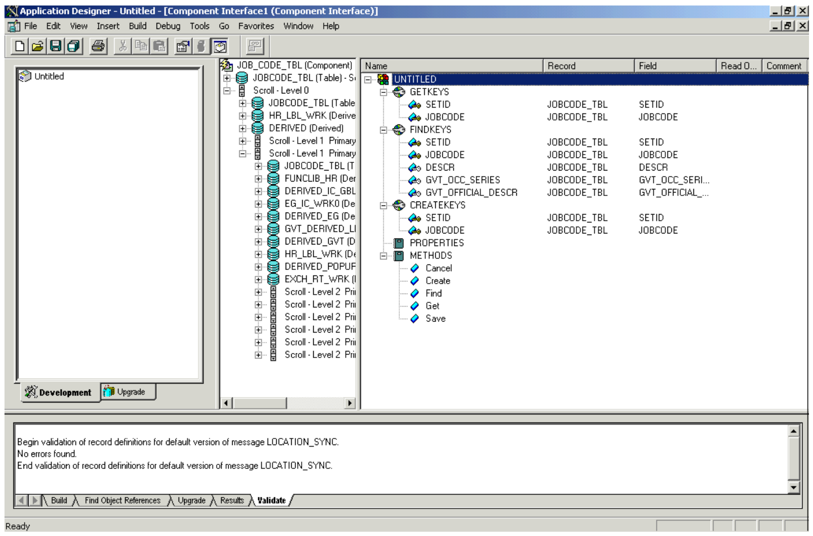
Do you want to default the properties based on the underlying Component definition, JOB_CODE_TBL? (123,12)

```
Begin validation of record definitions for default version of message LOCATION_SYNC.
No errors found.
End validation of record definitions for default version of message LOCATION_SYNC.
```

6 To create the component interface without displaying individual properties for one-by-one selection, click **No**. Otherwise, to expose component properties manually, click **No** and *drag the relevant fields from the left pane to the right pane*.

NOTE: If the component interface is large, it is recommended that you expose the component properties manually as just described.

See your PeopleBooks documentation for additional options and their meanings.


Application Designer - Untitled - [Component Interface1 (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
Untitled
Development
Upgrade
Build
Find Object References
Upgrade
Results
Validate
Ready

Name	Record	Field	Read O...	Comment
UNTITLED				
GETKEYS				
SETID	JOBCODE_TBL	SETID		
JOBCODE	JOBCODE_TBL	JOBCODE		
FINDKEYS				
SETID	JOBCODE_TBL	SETID		
JOBCODE	JOBCODE_TBL	JOBCODE		
DESCR	JOBCODE_TBL	DESCR		
GVT_OCC_SERIES	JOBCODE_TBL	GVT_OCC_SERI...		
GVT_OFFICIAL_DESCR	JOBCODE_TBL	GVT_OFFICIAL_...		
CREATEKEYS				
SETID	JOBCODE_TBL	SETID		
JOBCODE	JOBCODE_TBL	JOBCODE		
PROPERTIES				
METHODS				
Cancel				
Create				
Find				
Get				
Save				

```
Begin validation of record definitions for default version of message LOCATION_SYNC.
No errors found.
End validation of record definitions for default version of message LOCATION_SYNC.
```

7 Designate the methods you wish to use in this Component Interface as described in the next procedure.

Methods

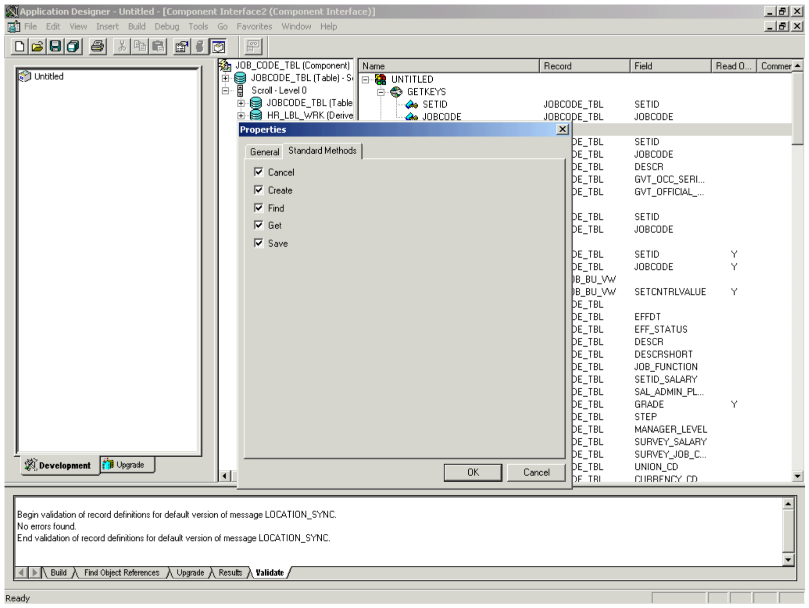
The standard methods created for a component interface are:

- Create
- Find
- Get
- Save

Only those methods actually implemented in the underlying component are available. For example, if the underlying component does not contain Add capabilities, Create is not available.

➢ To designate Methods to use in the Component Interface

1 Open the Component Interface Properties dialog box (continuing from the procedure described previously).


Application Designer - Untitled - [Component Interface2 (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
Properties
General
Standard Methods
OK
Cancel
Development
Upgrade
Ready

☑ Cancel
☑ Create
☑ Find
☑ Get
☑ Save

```
Begin validation of record definitions for default version of message LOCATION_SYNC.
No errors found.
End validation of record definitions for default version of message LOCATION_SYNC.
```

2 Select the desired methods.

Properties

You may add properties from the records in the component view. You can also *delete* a property in the component interface that you do not want to expose. You can rename properties by clicking the property and then, clicking again until you can type a new name. If you rename a property, it can be referenced in the component interface only by the new name, not by the underlying component name.

Properties may have various icons adjacent to them. For example, EMPLID has an icon indicating that it is a key field from the underlying record. NAME has an icon indicating that it is an alternate key field from the underlying record. For a complete list of property icons, see the PeopleBooks documentation.

You can right-click both panes to select various functions to perform depending on which pane is active. For a complete list of functions, see the PeopleBooks documentation.

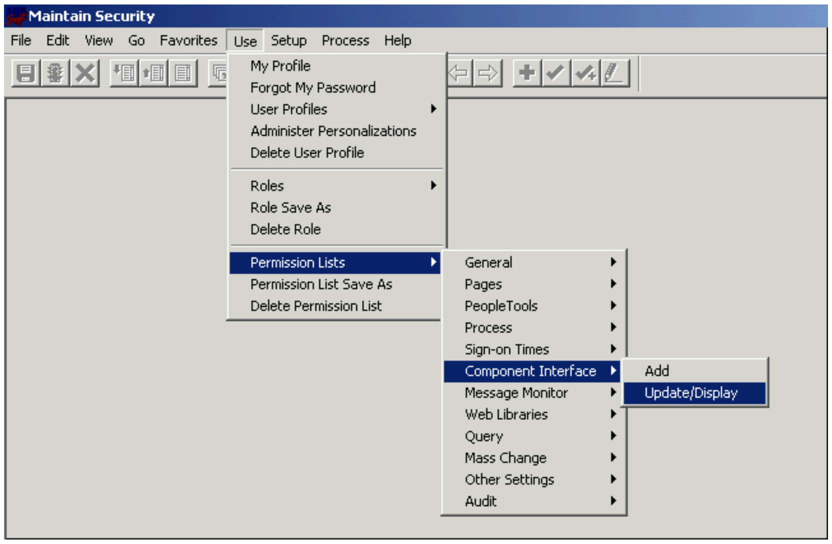
Securing a Component Interface

You must set up security for the component interface before you can begin testing.

Configuring Component Interface Security for PeopleSoft Version 8.1x

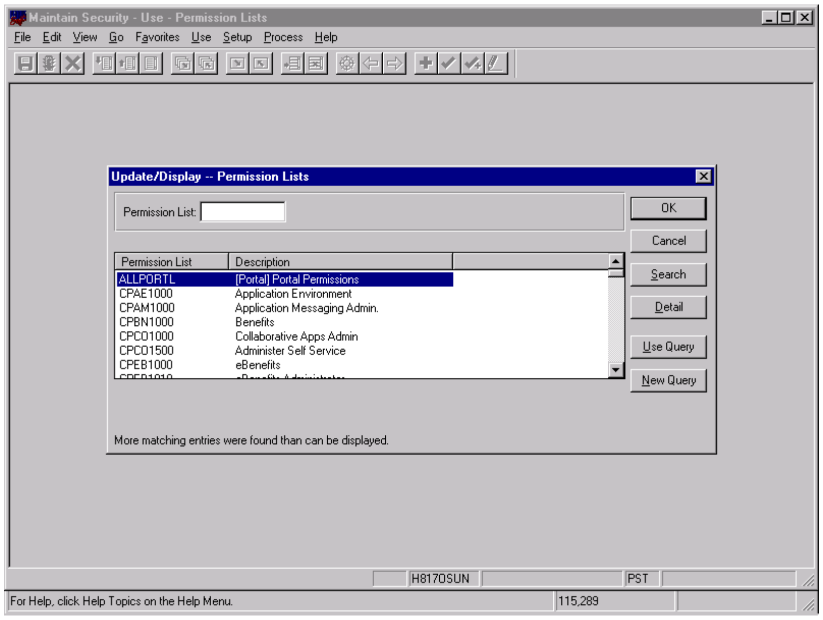
The following procedure describes how to configure component interface security for PeopleSoft Version 8.1 in 2- and 3-tier mode.

➤ **To configure component interface security:**



1 From the Use menu, select *Permission Lists*, *Component Interface*, and then click *Update/Display.*

Before Security can be set, the Permission List(s) must be identified.


Maintain Security - Use - Permission Lists
File Edit View Go Favorites Use Setup Process Help
Update/Display -- Permission Lists
OK
Cancel
Search
Detail
Use Query
New Query

Permission List:

Permission List	Description
ALLPORTL	[Portal] Portal Permissions
CPAE1000	Application Environment
CPAM1000	Application Messaging Admin.
CPBN1000	Benefits
CPCO1000	Collaborative Apps Admin
CPCO1500	Administer Self Service
CPEB1000	eBenefits

More matching entries were found than can be displayed.

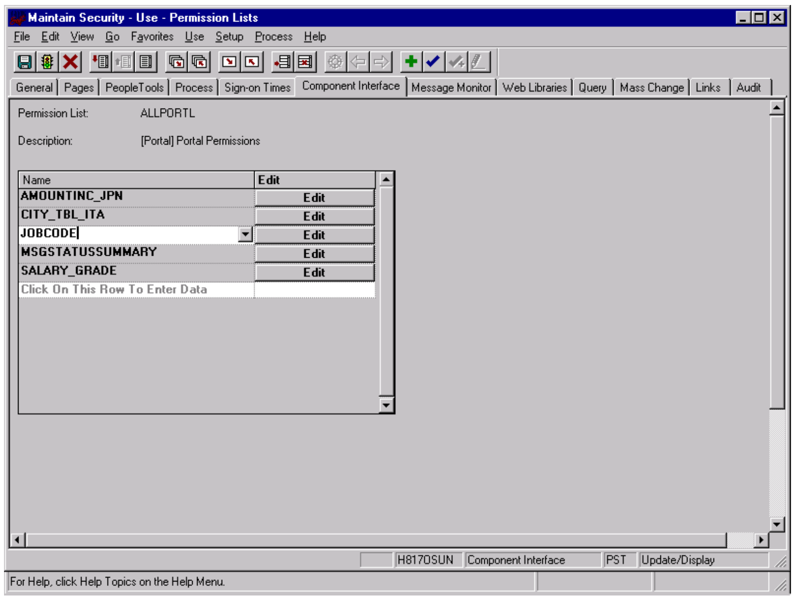
H817OSUN PST

For Help, click Help Topics on the Help Menu. 115,289

2 Select the relevant Permission list.

NOTE: For more information on Permission Lists, see the PeopleBooks documentation.

The following pane opens.


Maintain Security - Use - Permission Lists
File Edit View Go Favorites Use Setup Process Help
General Pages PeopleTools Process Sign-on Times Component Interface Message Monitor Web Libraries Query Mass Change Links Audit

Permission List: ALLPORTL

Description: [Portal] Portal Permissions

Name	Edit
AMOUNTINC_JPN	Edit
CITY_TBL_ITA	Edit
JOBCODE	Edit
MSGSTATUSSUMMARY	Edit
SALARY_GRADE	Edit
Click On This Row To Enter Data	

H817OSUN Component Interface PST Update/Display

For Help, click Help Topics on the Help Menu.

3 Insert the new component interface that you created.

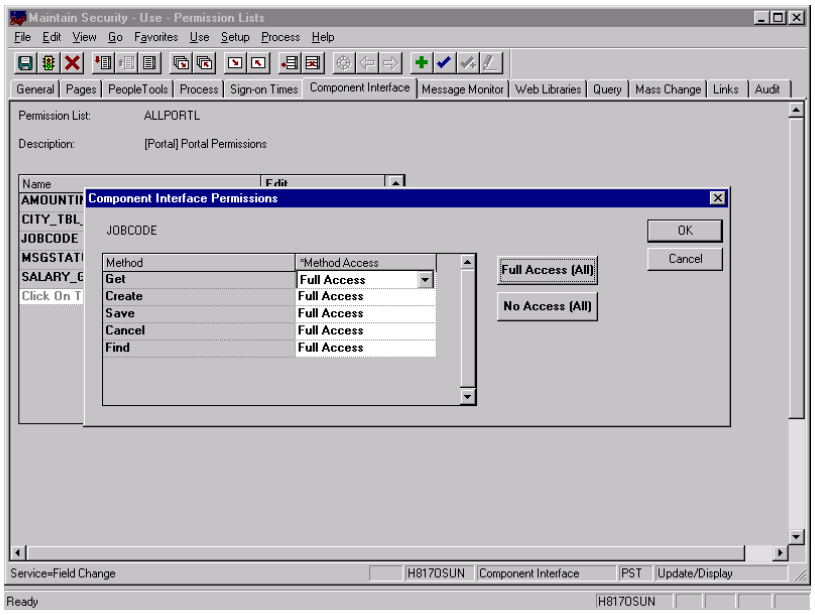
4 Click **Edit**.

NOTE: When you select the component interface, all available methods appear, including user-defined methods. This enables you to specify whether this particular Permission List must have Full or Partial Access.

5 Select the desired level of access.

6 Click **OK**.

In the following example, the ALLPORTL Permission List has Full Access to all methods.


Maintain Security - Use - Permission Lists
File Edit View Go Favorites Use Setup Process Help
General Pages PeopleTools Process Sign-on Times Component Interface Message Monitor Web Libraries Query Mass Change Links Audit
Component Interface Permissions
OK
Cancel
Full Access (All)
No Access (All)

Permission List: ALLPORTL

Description: [Portal] Portal Permissions

Name	Edit
AMOUNTII	
CITY_TBL_	
JOBCODE	
MSGSTATI	
SALARY_G	
Click On T	

JOBCODE

Method	*Method Access
Get	Full Access
Create	Full Access
Save	Full Access
Cancel	Full Access
Find	Full Access

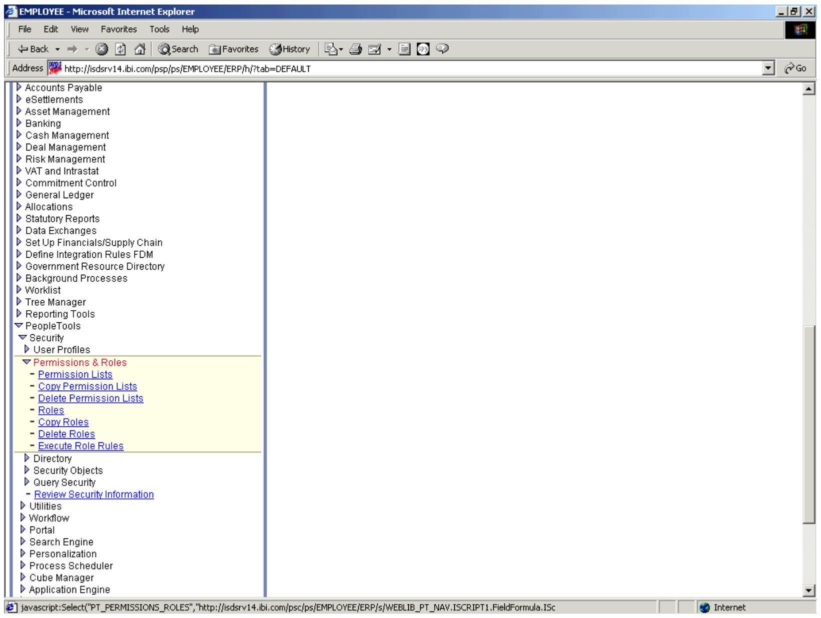
Service=Field Change | H8170SUN | Component Interface | PST | Update/Display

Ready | H8170SUN

Configuring Component Interface Security for PeopleSoft Version 8.4+

The following sample procedure describes how to configure component interface security for PeopleSoft Version 8.4 or higher.

➢ To configure interface security:



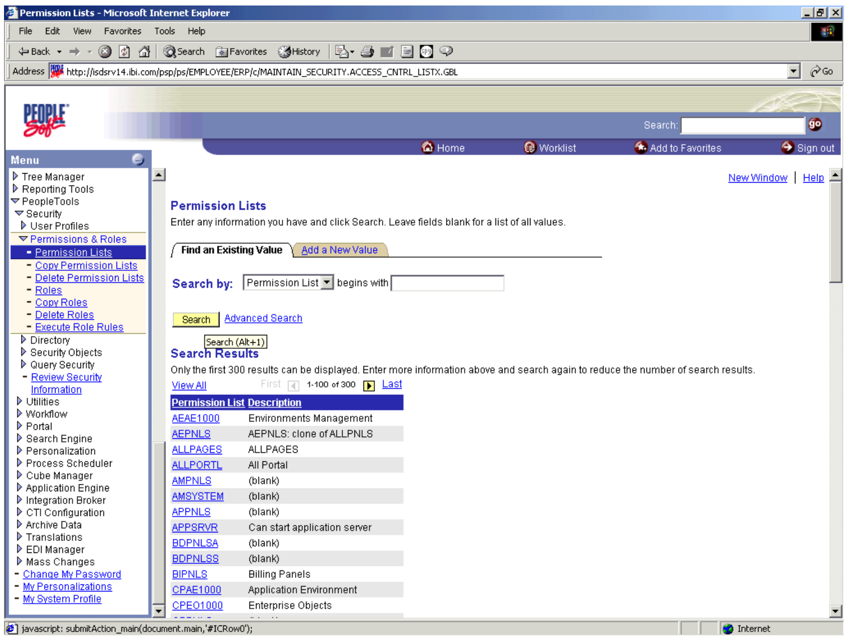
Address: http://isdsrv14.ibi.com/psp/ps/EMPLOYEE/ERP/h/?tab=DEFAULT

javascript:Select("PT_PERMISSIONS_ROLES","http://isdsrv14.ibi.com/psc/ps/EMPLOYEE/ERP/s/WEBLIB_PT_NAV.ISCRIPT1.FieldFormula.ISc

1 Expand *PeopleTools*, *Security*, *User Profiles*, and *Permissions & Roles* and then, click *Permission Lists*.

2 Click *Search*.

The Permissions List search window opens.



Permission Lists

Enter any information you have and click Search. Leave fields blank for a list of all values.

Search by: Permission List begins with

Search Results

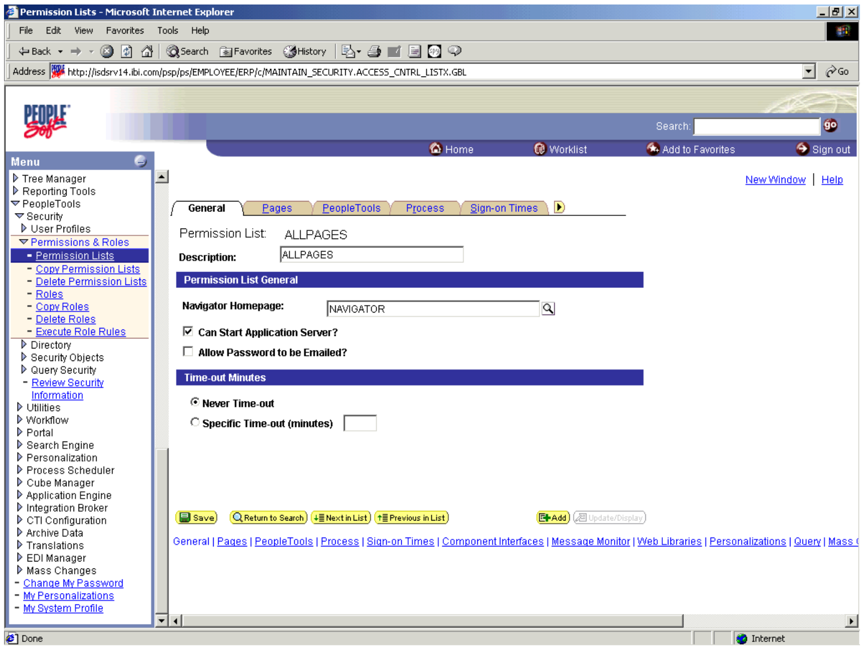
Only the first 300 results can be displayed. Enter more information above and search again to reduce the number of search results.

Permission List	Description
AEAE1000	Environments Management
AEPNLS	AEPNLS: clone of ALLPNLS
ALLPAGES	ALLPAGES
ALLPORTL	All Portal
AMPNLS	(blank)
AMSYSTEM	(blank)
APPNLS	(blank)
APPSRVR	Can start application server
BDPNLSA	(blank)
BDPNLSS	(blank)
BIPNLS	Billing Panels
CPAE1000	Application Environment
CPEO1000	Enterprise Objects

javascript: submitAction_main(document.main,'#ICRow0');

3 Select the relevant Permission List.

The Permissions List window opens.


Permission Lists - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://isdsrv14.ibi.com/psp/ps/EMPLOYEE/ERP/c/MAINTAIN_SECURITY.ACCESS_CNTRL_LISTX.GBL
Search: go
Home Worklist Add to Favorites Sign out
Menu
Tree Manager
Reporting Tools
PeopleTools
Security
User Profiles
Permissions & Roles
Permission Lists
Copy Permission Lists
Delete Permission Lists
Roles
Copy Roles
Delete Roles
Execute Role Rules
Directory
Security Objects
Query Security
Review Security Information
Utilities
Workflow
Portal
Search Engine
Personalization
Process Scheduler
Cube Manager
Application Engine
Integration Broker
CTI Configuration
Archive Data
Translations
EDI Manager
Mass Changes
Change My Password
My Personalizations
My System Profile
New Window | Help
General Pages PeopleTools Process Sign-on Times
Save Return to Search Next in List Previous in List Add Update/Display
General | Pages | PeopleTools | Process | Sign-on Times | Component Interfaces | Message Monitor | Web Libraries | Personalizations | Query | Mass
Done
Internet

Permission List: ALLPAGES

Description: ALLPAGES

Permission List General

Navigator Homepage: NAVIGATOR

☑ Can Start Application Server?

☐ Allow Password to be Emailed?

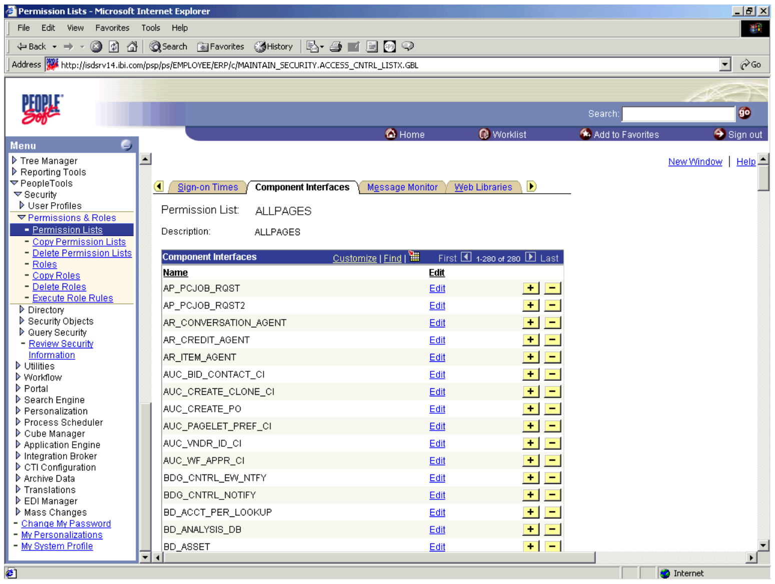
Time-out Minutes

◉ Never Time-out

○ Specific Time-out (minutes)

4 Click the right arrow next to the Sign-on Times tab.

The Component Interfaces tab becomes available.


Permission Lists - Microsoft Internet Explorer

Address: http://isdsrv14.ibi.com/psp/ps/EMPLOYEE/ERP/c/MAINTAIN_SECURITY.ACCESS_CNTRL_LISTX.GBL

Permission List: ALLPAGES

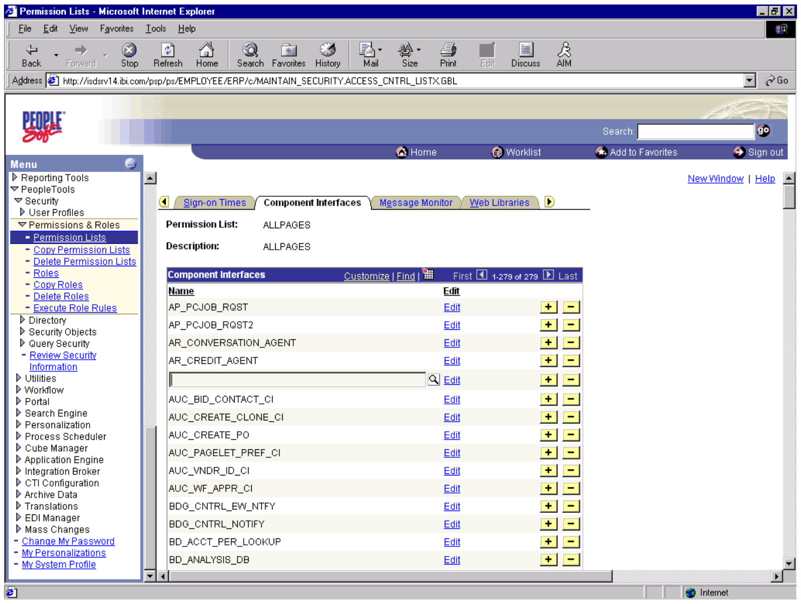
Description: ALLPAGES

Component Interfaces — Customize | Find | First 1-280 of 280 Last

Name	Edit
AP_PCJOB_RQST	Edit
AP_PCJOB_RQST2	Edit
AR_CONVERSATION_AGENT	Edit
AR_CREDIT_AGENT	Edit
AR_ITEM_AGENT	Edit
AUC_BID_CONTACT_CI	Edit
AUC_CREATE_CLONE_CI	Edit
AUC_CREATE_PO	Edit
AUC_PAGELET_PREF_CI	Edit
AUC_VNDR_ID_CI	Edit
AUC_WF_APPR_CI	Edit
BDG_CNTRL_EW_NTFY	Edit
BDG_CNTRL_NOTIFY	Edit
BD_ACCT_PER_LOOKUP	Edit
BD_ANALYSIS_DB	Edit
BD_ASSET	Edit

5 Click the *Component Interfaces* tab.

6 Click the + button to add a new row to the Component Interfaces list.


Permission Lists - Microsoft Internet Explorer

Address: http://isdsrv14.ibi.com/psp/ps/EMPLOYEE/ERP/c/MAINTAIN_SECURITY.ACCESS_CNTRL_LISTX.GBL

Permission List: ALLPAGES

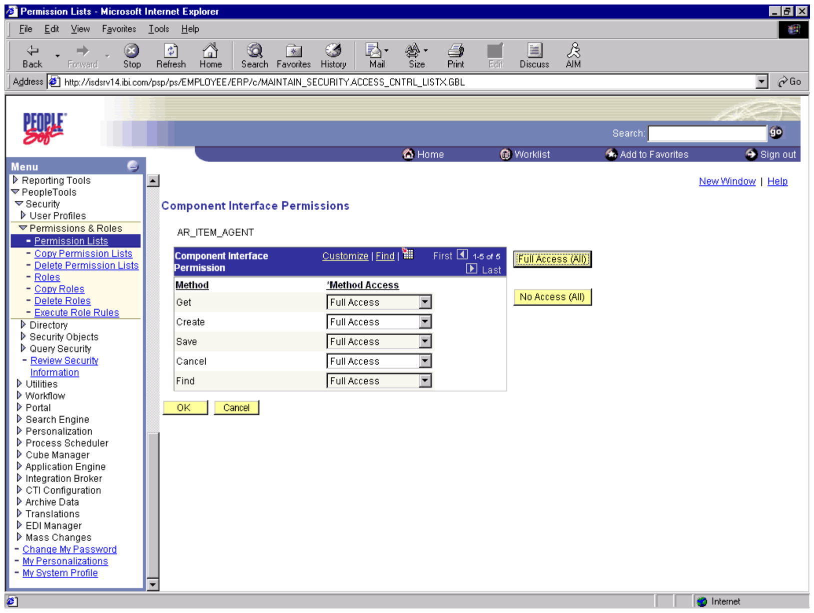
Description: ALLPAGES

Component Interfaces — Customize | Find | First 1-279 of 279 Last

Name	Edit
AP_PCJOB_RQST	Edit
AP_PCJOB_RQST2	Edit
AR_CONVERSATION_AGENT	Edit
AR_CREDIT_AGENT	Edit
	Edit
AUC_BID_CONTACT_CI	Edit
AUC_CREATE_CLONE_CI	Edit
AUC_CREATE_PO	Edit
AUC_PAGELET_PREF_CI	Edit
AUC_VNDR_ID_CI	Edit
AUC_WF_APPR_CI	Edit
BDG_CNTRL_EW_NTFY	Edit
BDG_CNTRL_NOTIFY	Edit
BD_ACCT_PER_LOOKUP	Edit
BD_ANALYSIS_DB	Edit

7 Type the component interface name and click **Edit**.

This example uses component interface AR_ITEM_AGENT.



Component Interface Permissions

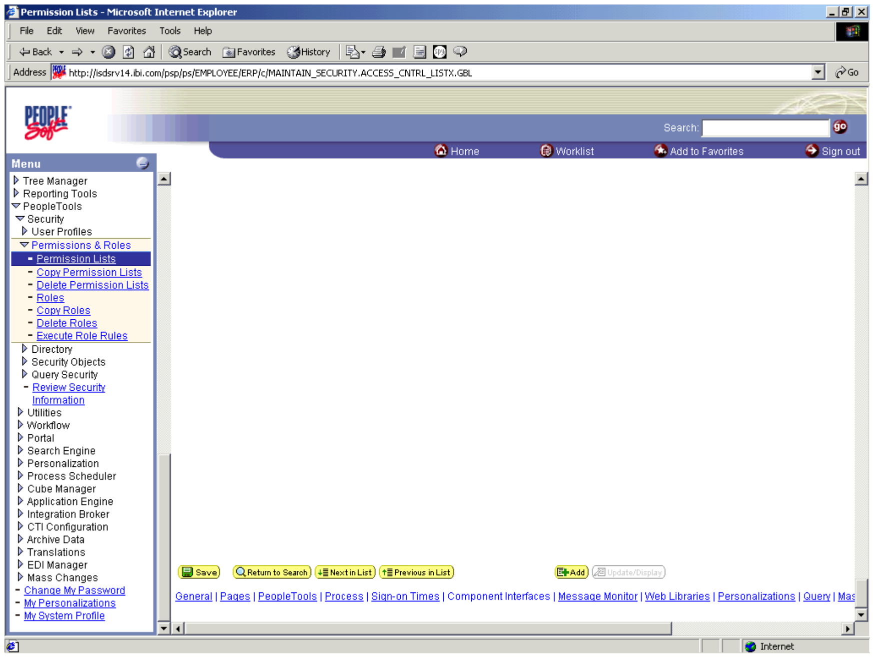
AR_ITEM_AGENT

Component Interface Permission

Method	*Method Access
Get	Full Access
Create	Full Access
Save	Full Access
Cancel	Full Access
Find	Full Access

8 Select the desired access from the drop-down lists.

9 Click **OK**.


Permission Lists - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://isdsrv14.ibi.com/psp/ps/EMPLOYEE/ERP/c/MAINTAIN_SECURITY.ACCESS_CNTRL_LISTX.GBL
Search: go
Home Worklist Add to Favorites Sign out
Menu
Tree Manager
Reporting Tools
PeopleTools
Security
User Profiles
Permissions & Roles
Permission Lists
Copy Permission Lists
Delete Permission Lists
Roles
Copy Roles
Delete Roles
Execute Role Rules
Directory
Security Objects
Query Security
Review Security Information
Utilities
Workflow
Portal
Search Engine
Personalization
Process Scheduler
Cube Manager
Application Engine
Integration Broker
CTI Configuration
Archive Data
Translations
EDI Manager
Mass Changes
Change My Password
My Personalizations
My System Profile
Save Return to Search Next in List Previous in List Add Update/Display
General | Pages | PeopleTools | Process | Sign-on Times | Component Interfaces | Message Monitor | Web Libraries | Personalizations | Query | Mas
Internet

10 Scroll down to the bottom of the Component Interface Permissions window and click *Save*.

Testing a Component Interface

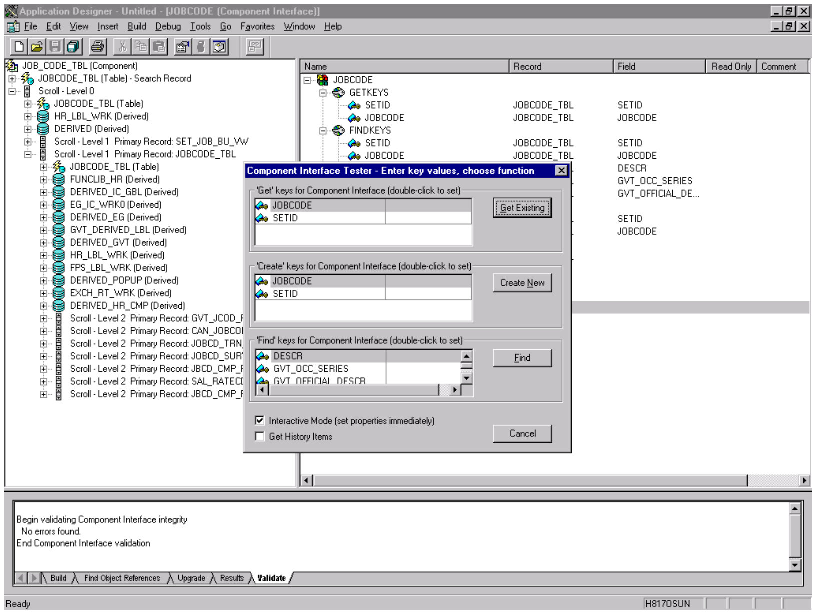
The iWay Adapter for PeopleSoft 8 uses PeopleSoft metadata and component interfaces, therefore, it can accommodate new or modified component interfaces. The adapter makes no assumptions about component interfaces except that they are logical and valid. Therefore, each component interface must be tested before being used as a source for the adapter.

If changes are made to the underlying application by the user or by means of a PeopleSoft upgrade, and these changes invalidate a component interface, the user must repair the invalid component interface before the adapter uses it.

➢ To test a component interface:

1 In Application Designer, select *Test Component Interface* from the Tools menu.

The Component Interface Tester dialog box opens.


Application Designer - Untitled - [JOBCODE (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
Name Record Field Read Only Comment
Component Interface Tester - Enter key values, choose function
'Get' keys for Component Interface (double-click to set)
Get Existing
'Create' keys for Component Interface (double-click to set)
Create New
'Find' keys for Component Interface (double-click to set)
Find
Interactive Mode (set properties immediately)
Get History Items
Cancel
Build Find Object References Upgrade Results Validate
Ready
H8170SUN

```
JOB_CODE_TBL (Component)
  JOBCODE_TBL (Table) - Search Record
  Scroll - Level 0
    JOBCODE_TBL (Table)
    HR_LBL_WRK (Derived)
    DERIVED (Derived)
    Scroll - Level 1  Primary Record: SET_JOB_BU_VW
    Scroll - Level 1  Primary Record: JOBCODE_TBL
      JOBCODE_TBL (Table)
      FUNCLIB_HR (Derived)
      DERIVED_IC_GBL (Derived)
      EG_IC_WRK0 (Derived)
      DERIVED_EG (Derived)
      GVT_DERIVED_LBL (Derived)
      DERIVED_GVT (Derived)
      HR_LBL_WRK (Derived)
      FPS_LBL_WRK (Derived)
      DERIVED_POPUP (Derived)
      EXCH_RT_WRK (Derived)
      DERIVED_HR_CMP (Derived)
      Scroll - Level 2  Primary Record: GVT_JCOD_F
      Scroll - Level 2  Primary Record: CAN_JOBCOI
      Scroll - Level 2  Primary Record: JOBCD_TRN
      Scroll - Level 2  Primary Record: JOBCD_SUR
      Scroll - Level 2  Primary Record: JBCD_CMP_F
      Scroll - Level 2  Primary Record: SAL_RATECI
      Scroll - Level 2  Primary Record: JBCD_CMP_F
```

Name	Record	Field
JOBCODE		
GETKEYS		
SETID	JOBCODE_TBL	SETID
JOBCODE	JOBCODE_TBL	JOBCODE
FINDKEYS		
SETID	JOBCODE_TBL	SETID
JOBCODE	JOBCODE_TBL	JOBCODE
		DESCR
		GVT_OCC_SERIES
		GVT_OFFICIAL_DE...
		SETID
		JOBCODE

'Get' keys: JOBCODE, SETID

'Create' keys: JOBCODE, SETID

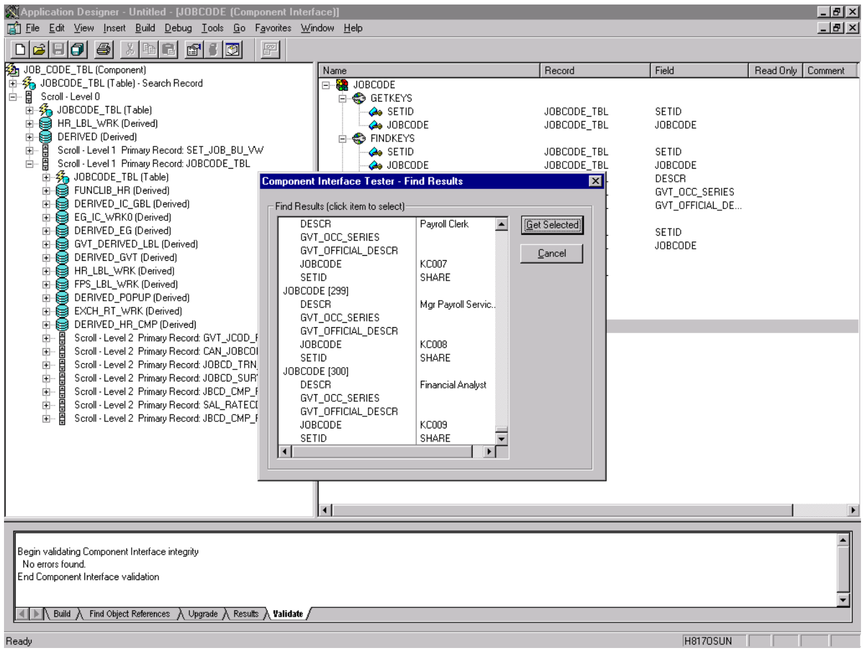
'Find' keys: DESCR, GVT_OCC_SERIES, GVT_OFFICIAL_DESCR

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

2 Click the *Component Interface Tester* dialog box to bring it to the foreground, if required.

3 To test the component interface using the Find method, click *Find*.

The Component Interface Tester - Find Results window opens, displaying all of the possible entries for the underlying component. If there are more than 300 entries, a message appears.

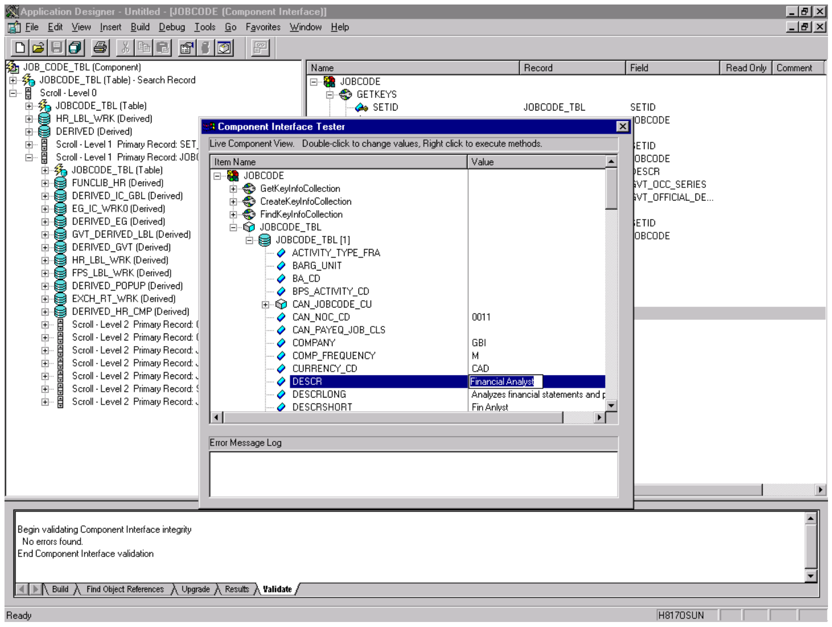

Application Designer - Untitled - [JOBCODE (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
Name Record Field Read Only Comment
Component Interface Tester - Find Results
Find Results (click item to select)
Get Selected
Cancel
Build Find Object References Upgrade Results Validate
Ready
H8170SUN

```
  DESCR                 Payroll Clerk
  GVT_OCC_SERIES
  GVT_OFFICIAL_DESCR
  JOBCODE               KC007
  SETID                 SHARE
JOBCODE [299]
  DESCR                 Mgr Payroll Servic..
  GVT_OCC_SERIES
  GVT_OFFICIAL_DESCR
  JOBCODE               KC008
  SETID                 SHARE
JOBCODE [300]
  DESCR                 Financial Analyst
  GVT_OCC_SERIES
  GVT_OFFICIAL_DESCR
  JOBCODE               KC009
  SETID                 SHARE
```

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

4 Select a field in the left pane of the Find Results window.

5 To display the relevant data for that particular key, click *Get Selected*.


Application Designer - Untitled - [JOBCODE (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
Component Interface Tester
Build Find Object References Upgrade Results Validate
Ready
H8170SUN

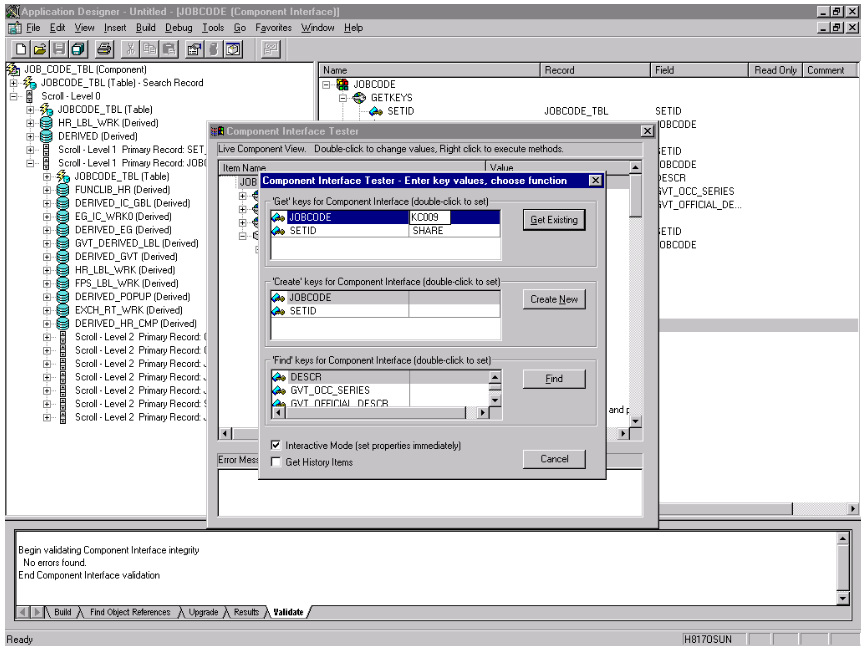
Live Component View. Double-click to change values, Right click to execute methods.

Item Name	Value
JOBCODE	
GetKeyInfoCollection	
CreateKeyInfoCollection	
FindKeyInfoCollection	
JOBCODE_TBL	
JOBCODE_TBL [1]	
ACTIVITY_TYPE_FRA	
BARG_UNIT	
BA_CD	
BPS_ACTIVITY_CD	
CAN_JOBCODE_CU	
CAN_NOC_CD	0011
CAN_PAYEQ_JOB_CLS	
COMPANY	GBI
COMP_FREQUENCY	M
CURRENCY_CD	CAD
DESCR	Financial Analyst
DESCRLONG	Analyzes financial statements and p
DESCRSHORT	Fin Anlyst

Error Message Log

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

If the security settings permit, you can change the values in the individual fields.


Application Designer - Untitled - [JOBCODE (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
Component Interface Tester
Component Interface Tester - Enter key values, choose function
Get Existing
Create New
Find
Cancel
Build Find Object References Upgrade Results Validate
Ready
H8170SUN

'Get' keys for Component Interface (double-click to set)

JOBCODE	KC009
SETID	SHARE

'Create' keys for Component Interface (double-click to set)

JOBCODE	
SETID	

'Find' keys for Component Interface (double-click to set)

DESCR	
GVT_OCC_SERIES	
GVT_OFFICIAL_DESCR	

[✓] Interactive Mode (set properties immediately)
[ ] Get History Items

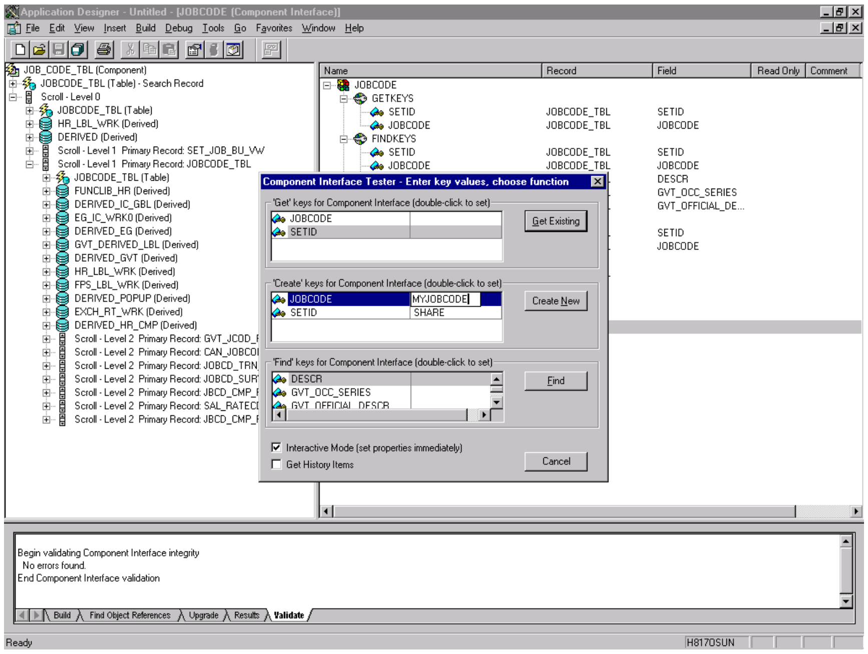
```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

6 To test the component interface using the Get method, enter the existing key(s).

7 Click *Get Existing*.

This returns the exposed properties for the key that you entered.

You can change values if Update access has been specified.


Application Designer - Untitled - [JOBCODE (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
Get Existing
Create New
Find
Cancel
Build
Find Object References
Upgrade
Results
Validate
Ready
H8170SUN

Component Interface Tester - Enter key values, choose function

'Get' keys for Component Interface (double-click to set)

JOBCODE	
SETID	

'Create' keys for Component Interface (double-click to set)

JOBCODE	MYJOBCODE
SETID	SHARE

'Find' keys for Component Interface (double-click to set)

DESCR	
GVT_OCC_SERIES	
GVT_OFFICIAL_DESCR	

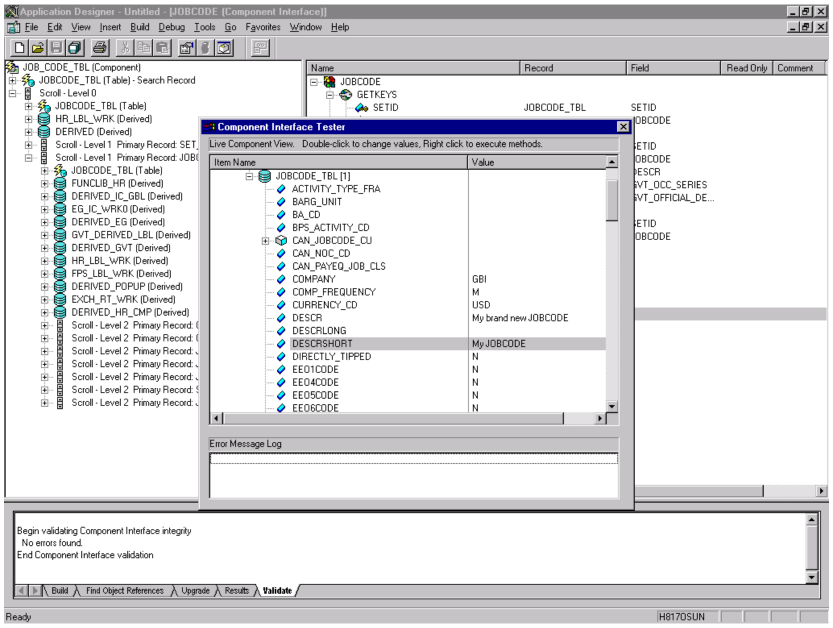
[✓] Interactive Mode (set properties immediately)
[ ] Get History Items

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

8 To test the component interface using the Create method, enter all required key values.

9 Click *Create New.*

When you enter valid values in ‘Create’ keys, the JOBCODE Data Display window opens after the Table name is expanded with default data in place.


Application Designer - Untitled - [JOBCODE (Component Interface)]
File Edit View Insert Build Debug Tools Go Favorites Window Help
Build
Find Object References
Upgrade
Results
Validate
Ready
H8170SUN

Component Interface Tester

Live Component View. Double-click to change values, Right click to execute methods.

Item Name	Value
JOBCODE_TBL [1]	
ACTIVITY_TYPE_FRA	
BARG_UNIT	
BA_CD	
BPS_ACTIVITY_CD	
CAN_JOBCODE_CU	
CAN_NOC_CD	
CAN_PAYEQ_JOB_CLS	
COMPANY	GBI
COMP_FREQUENCY	M
CURRENCY_CD	USD
DESCR	My brand new JOBCODE
DESCRLONG	
DESCRSHORT	My JOBCODE
DIRECTLY_TIPPED	N
EEO1CODE	N
EEO4CODE	N
EEO5CODE	N
EEO6CODE	N

Error Message Log

```
Begin validating Component Interface integrity
 No errors found.
End Component Interface validation
```

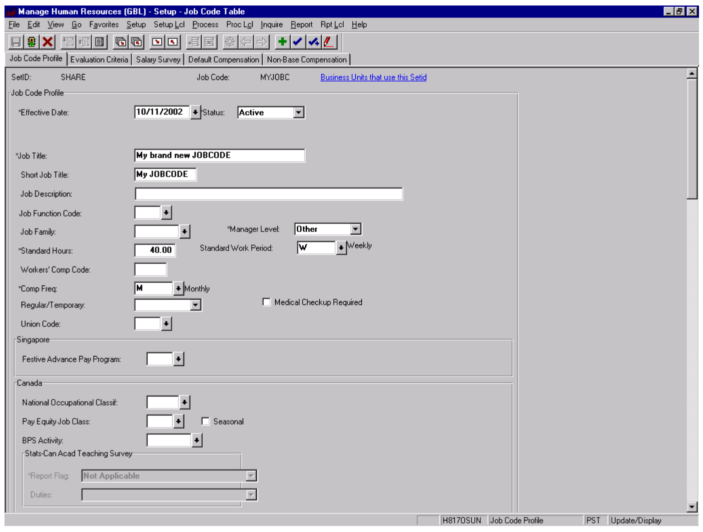
You can change fields at this point. Changes are validated against the component’s underlying business logic.

10 After you finish making changes, right-click the top item in the pane.

11 Click *Save* to save your changes.

The keys used to create the record can be used with the Get method for viewing data.

The data that were added can be viewed in the PeopleSoft Component as shown in the following example.


Manage Human Resources (GBL) - Setup - Job Code Table
File Edit View Go Favorites Setup Setup Lcl Process Proc Lcl Inquire Report Rpt Lcl Help
Job Code Profile
Evaluation Criteria
Salary Survey
Default Compensation
Non-Base Compensation
Business Units that use this Setid

SetID: SHARE Job Code: MYJOBC

Job Code Profile

*Effective Date:	10/11/2002	*Status:	Active
*Job Title:	My brand new JOBCODE		
Short Job Title:	My JOBCODE		
Job Description:			
Job Function Code:			
Job Family:		*Manager Level:	Other
*Standard Hours:	40.00	Standard Work Period:	W Weekly
Workers' Comp Code:			
*Comp Freq:	M Monthly		
Regular/Temporary:		☐ Medical Checkup Required	
Union Code:			

Singapore

Festive Advance Pay Program:	

Canada

National Occupational Classif:		
Pay Equity Job Class:		☐ Seasonal
BPS Activity:		

Stats-Can Acad Teaching Survey

*Report Flag:	Not Applicable
Duties:	

H817OSUN Job Code Profile PST Update/Display

Notice the Effective Date as one of the default values.

You have finished testing the component interface. Before using the component interface, you must generate its API, as described at "Building PeopleSoft Component Interface API Classes" on page 26.

Index