AUTHORIZED DOCUMENTATION

Driver for PeopleSoft* 5.2 Implementation Guide

Novell®
Identity Manager

3.6

July 23, 2008

www.novell.com

Legal Notices

Novell, Inc. makes no representations or warranties with respect to the contents or use of this documentation, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

Further, Novell, Inc. makes no representations or warranties with respect to any software, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to make changes to any and all parts of Novell software, at any time, without any obligation to notify any person or entity of such changes.

Any products or technical information provided under this Agreement may be subject to U.S. export controls and the trade laws of other countries. You agree to comply with all export control regulations and to obtain any required licenses or classification to export, re-export or import deliverables. You agree not to export or re-export to entities on the current U.S. export exclusion lists or to any embargoed or terrorist countries as specified in the U.S. export laws. You agree to not use deliverables for prohibited nuclear, missile, or chemical biological weaponry end uses. See the Novell International Trade Services Web page (http://www.novell.com/info/exports/) for more information on exporting Novell software. Novell assumes no responsibility for your failure to obtain any necessary export approvals.

Novell, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed on the Novell Legal Patents Web page (http://www.novell.com/company/legal/patents/) and one or more additional patents or pending patent applications in the U.S. and in other countries.

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

Online Documentation: To access the latest online documentation for this and other Novell products, see the Novell Documentation Web page (http://www.novell.com/documentation).

Novell Trademarks

For Novell trademarks, see the Novell Trademark and Service Mark list (http://www.novell.com/company/legal/trademarks/tmlist.html).

Third-Party Materials

All third-party trademarks are the property of their respective owners.

Contents

About This Guide

Introduction

The Identity Manager Driver for PeopleSoft (PeopleSoft driver) provides a solution for synchronizing data between Novell® Identity Vault and PeopleSoft.

This guide provides an overview of the driver's technology as well as configuration instructions.

Audience

This guide is intended for consultants, administrators, and IS personnel who need to install, configure, and maintain the PeopleSoft driver.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comments feature at the bottom of each page of the online documentation, or go to Novell's Feedback Web site (http://www.novell.com/documentation/feedback.html) and enter your comments there.

Documentation Updates

For the most recent version of this document, see the Identity Manager 3.6 Drivers Documentation Web site (http://www.novell.com/documentation/idm36drivers/index.html).

Additional Documentation

For documentation on using Novell Identity Manager and the other drivers, see the Identity Manager 3.6 Documentation Web site (http://www.novell.com/documentation/idm36).

Documentation Conventions

In Novell documentation, a greater-than symbol (>) is used to separate actions within a step and items within a cross-reference path.

A trademark symbol (®, ™, etc.) denotes a Novell trademark. An asterisk (*) denotes a third-party trademark.

When a single pathname can be written with a backslash for some platforms or a forward slash for other platforms, the pathname is presented with a backslash. Users of platforms that require a forward slash, such as Linux* or UNIX*, should use forward slashes as required by your software.

Overview

1

PeopleSoft applications are some of the most popular Enterprise Resource Planning (ERP) systems available. The Identity Manager 3.6 Driver for PeopleSoft 5.2 (PeopleSoft driver) enables you to create and manage Identity Vault (Novell® eDirectory™) objects by using data you receive from a PeopleSoft application. It's a powerful solution to maintain, propagate, and transform your data.

This driver can integrate any PeopleSoft component with the Identity Vault. Using Novell Identity Manager technology, you can share and synchronize authoritative PeopleSoft data with other enterprise applications, databases, or directories. As new records are added, modified, disabled, or deactivated in PeopleSoft, tasks associated with these events can be processed automatically using Identity Manager.

Because Identity Manager is a bidirectional data management solution, you can also synchronize authoritative data from other systems to PeopleSoft components. This dynamic, business-specific solution allows you to manage and integrate information however you desire.

This section contains the following topics:

1.1 Prerequisites

- ❑ PeopleSoft Application Server with PeopleTools version 8.17 or later, 8.20 or later, or 8.41 or later.
- ❑ The appropriate version of the PeopleTools `psjoa.jar` client to match the PeopleTools version of the target PeopleSoft Application Server.
- ❑ A `.jar` file containing the compiled Java* Component Interface APIs for the desired integration component. For the default PSA components, the file containing the interfaces is named `dirxmlcomps.jar`. For more information on creating Component Interface APIs, refer to Chapter 6, "Customizing the Driver," on page 51.

1.2 Driver Concepts

The following sections explain the concepts you should understand before attempting to implement the PeopleSoft driver in your environment:

1.2.1 Driver Components

The driver includes the following components:

- Driver shim

  The driver shim (`psoftshim.jar`) enables communication between PeopleSoft and Identity Vault. It bidirectionally reports object change events and applies object modification commands between these systems.

- Driver Configuration

  `PeopleSoft50-IDM3_6_0-V2.xml` is a driver configuration that is imported through Designer or iManager. It contains configuration parameters and policies that enable Identity Manager to handle the synchronization and data object manipulation between PeopleSoft and the Identity Vault.

  This file is not intended for use in a production environment. It acts as a template that contains most of the common synchronization tasks performed in typical integration scenarios. You should configure your policies based on your own business processes and integration points. For more information, refer to Chapter 4, "Creating a New Driver," on page 41 and Section 6.3, "Modifying Driver Policies," on page 56.

- PeopleSoft Service Agent

  The PeopleSoft Service Agent (PSA) is a collection of PeopleSoft application objects developed for use with the driver shim and default driver configuration. Because all of the objects (fields, records, pages, components, component interfaces) are specifically named with a DirXML identifier, the PSA can be deployed onto a PeopleSoft application server without affecting existing PeopleSoft applications and objects.

  The various pieces of the PSA provide examples of how data can be integrated between Identity Vault and PeopleSoft. Examples of PSA uses include the following:

  - Implementation of an intermediate staging table. The synchronization between the Novell sample Personnel Application and the staging table shows the best practices of PeopleSoft internal application integration using PeopleCode Component Interfaces.
  - Integration can be accomplished directly to the sample Personnel Application by simply changing the driver's configuration.
  - PeopleCode is provided to show how database events on the sample Personnel Application can be reported to the driver shim via the transaction record interface required by the driver shim.
  - PeopleCode is provided to show how to implement a Delete method on a Component Interface.

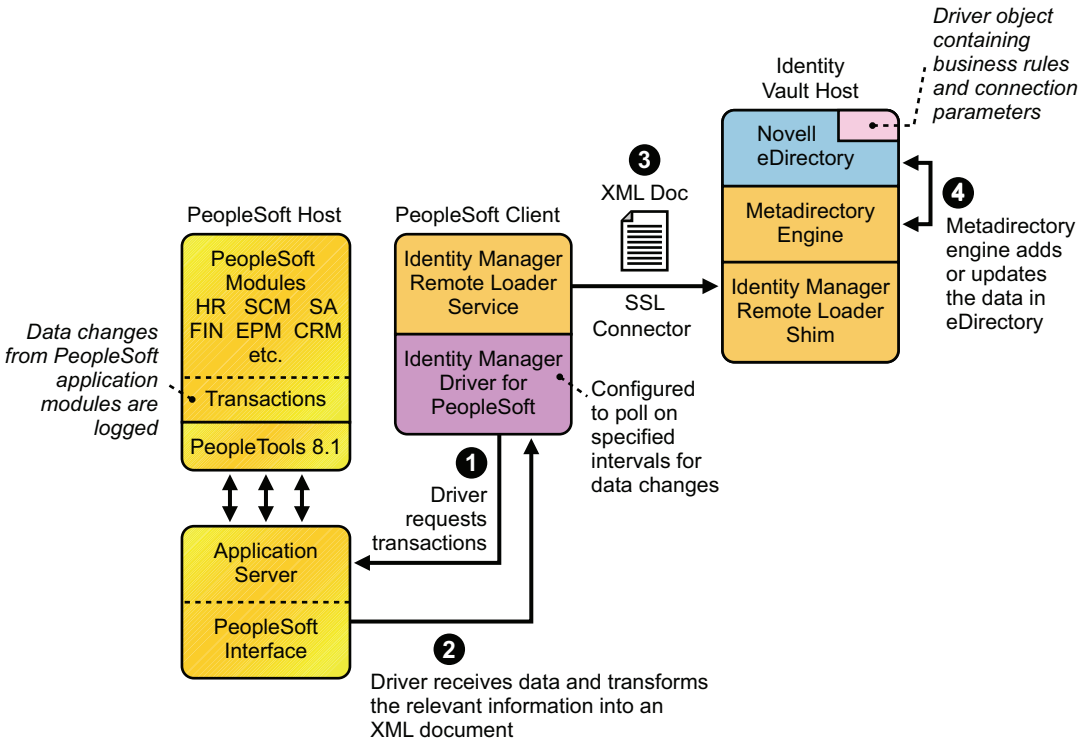
  As with the driver configuration, the PSA is not intended for use in a production environment. It acts as a template that contains most of the common synchronization tasks needed in typical integration scenarios. You should configure your policies based on your own business processes and integration points. For more information, refer to Chapter 3, "Configuring Your PeopleSoft Environment," on page 19.

1.2.2 How the Driver Works

The following section describes the basic functions of the driver. It uses the Remote Loader configuration as an example; however, it does not require the use of the Remote Loader. For more information, refer to Chapter 2, "Installing the Driver Files," on page 17.

The Publisher Channel

Figure 1-1 *The Publisher Channel*



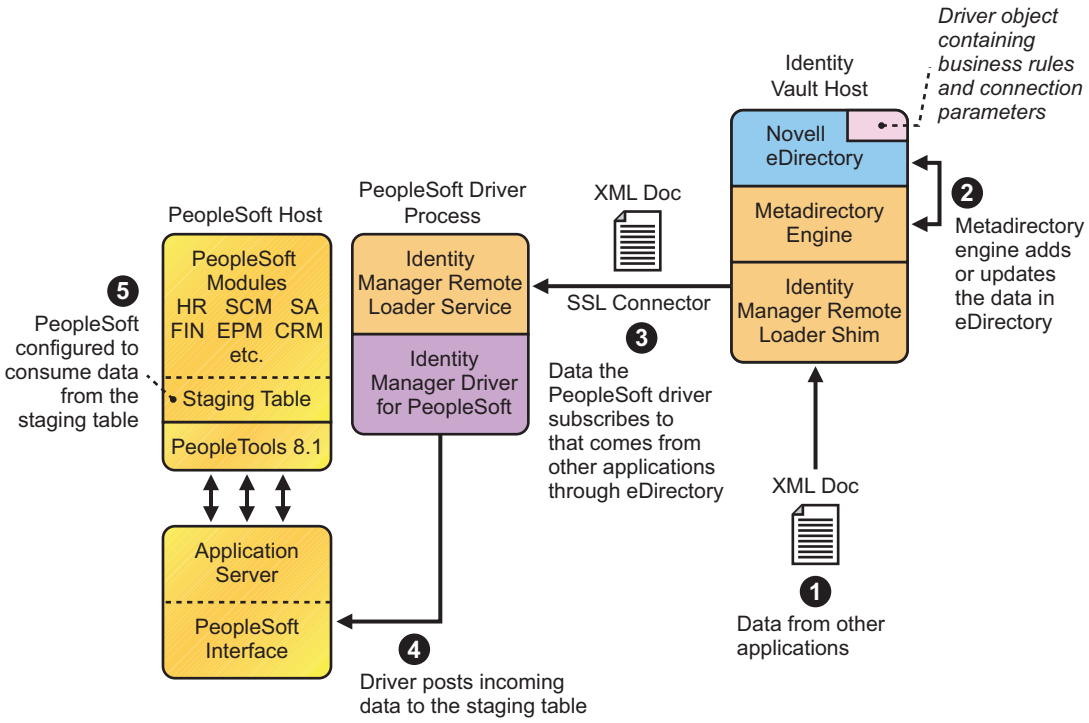
The Publisher channel synchronizes data from PeopleSoft to Identity Vault. As events occur within PeopleSoft, transactions are placed into a transaction table. These transactions are usually written to the table through PeopleCode (you can use other methods such as Batch SQL, COBOL, SQR, and so forth.) Component Interface (CI) objects enable the driver to access transactions within the PeopleSoft system, and to query for relevant data associated with an individual transaction type. These CI objects are included as part of the PeopleSoft Service Agent (PSA).

The driver accesses the PeopleSoft environment by connecting through the Component Interface at the Application Server level. The driver periodically requests transactions that are waiting to be processed by driver subtype (such as Employee, Student, or Customer.) It processes only those transactions that have an available status and a transaction date and time less than or equal to the current date and time.

The driver then constructs an XML document from the data it retrieves and passes it to the Metadirectory engine for processing. When the Metadirectory engine finishes processing the transaction, the driver updates the transaction with the status and any applicable messages in the transaction table inside of PeopleSoft. When events occur within Identity Vault, the driver connects to the appropriate CI and updates the PeopleSoft staging table accordingly. You can also configure the Driver to poll the Application server for event changes.

The Subscriber Channel

Figure 1-2 *The Subscriber Channel*



The Subscriber channel synchronizes data from other applications via Identity Vault to PeopleSoft.

As events occur within eDirectory, the driver receives an XML document from the Metadirectory engine and updates PeopleSoft. By configuring the filter on the Subscriber channel, you can specify what data you want updated in PeopleSoft. The driver uses the Schema Component Interface (CI) and updates a staging table inside the PeopleSoft environment.

If you want to move the data from the staging table into PeopleSoft, you can create and apply the necessary PeopleCode to handle this transaction. (All PeopleSoft objects that can interact with the transaction table, application data, as well as the CI, are delivered with the sample project.)

1.2.3 Configuring Your PeopleSoft Environment

You must configure your PeopleSoft application to:

- Trap events that occur within PeopleSoft and place transactions into a transaction table.
- Expose the transactions and any other desired data to the driver.

For detailed instructions regarding how to configure these processes, refer to Chapter 3, “Configuring Your PeopleSoft Environment,” on page 19.

1.2.4 Configuring Your Identity Manager System

The driver interacts with PeopleSoft at the PeopleTools level by using Component Interface (CI) technology. By using existing CI definitions within the PeopleSoft modules, along with a collection of the driver's preconfigured CI objects, you can do the following:

- Create eDirectory objects as new data is synchronized from PeopleSoft.
- Synchronize data bidirectionally between a PeopleSoft application and Identity Vault.
- Enable bidirectional object Create and Delete events.
- Transform eDirectory events (such as Delete, Rename, or Move events) into different events in PeopleSoft.
- Maintain publication authority over data.
- Establish Group, Role, or other relationships in Identity Vault based on relationships defined within the PeopleSoft application.
- Provide notifications based on various events or required approval processes.
- Adhere to enterprise business processes and policies.
- Share data with other systems involved in your enterprise provisioning solution.

For more information on configuring your Identity Manager system, refer to Chapter 6, "Customizing the Driver," on page 51.

1.3 Support for Standard Driver Features

The following sections provide information about how the PeopleSoft driver supports these standard driver features:

1.3.1 Local Platforms

A local installation is an installation of the driver on the Metadirectory server. The PeopleSoft driver can be installed on the operating systems supported for the Metadirectory server.

For information about the operating systems supported for the Metadirectory server, see "Metadirectory Server" in "System Requirements" in the *Identity Manager 3.6 Installation Guide*.

NOTE: The support of both local and remote platforms depends on the supported platforms of the PeopleTools Client (PSJOA) software.

1.3.2 Remote Platforms

The PeopleSoft driver can use the Remote Loader service to run on a server other than the Metadirectory server. The PeopleSoft driver can be installed on the operating systems supported for the Remote Loader.

For information about the supported operating systems, see “Remote Loader” in “System Requirements” in the *Identity Manager 3.6 Installation Guide*.

NOTE: The support of both local and remote platforms depends on the supported platforms of the PeopleTools Client (PSJOA) software.

1.3.3 Entitlements

The PeopleSoft driver does not have entitlement functionality defined with the default configuration files. The driver does support entitlements, if there are policies created for the driver to consume.

Installing the Driver Files

2

By default, the PeopleSoft driver files are installed on the Metadirectory server at the same time as the Metadirectory engine. The installation program extends the Identity Vault's schema and installs the driver shim and the driver configuration file. It does not create the driver in the Identity Vault (see Chapter 4, "Creating a New Driver," on page 41) or upgrade an existing driver's configuration (see Chapter 5, "Upgrading an Existing Driver," on page 49).

To ensure that you have the driver installed in the right location, and to finish the installation by manually copying required PeopleSoft files to the driver's directory, refer to the following sections:

- Section 2.1, "Installing the Driver Files," on page 17
- Section 2.2, "Copying the PeopleSoft Middleware Library File (psjoa.jar) to the Driver Directory," on page 17

2.1 Installing the Driver Files

The PeopleSoft driver must be located on the same server as the PeopleSoft application. If the driver is not on that server, you have the following options:

- Install the Metadirectory server (Metadirectory engine and drivers) to the PeopleSoft server. This requires Novell® eDirectory™ to be installed on the server. See the instructions in "Installing the Metadirectory Server" in the *Identity Manager 3.6 Installation Guide*.
- Install the Remote Loader (required to run the driver on a non-Metadirectory server) and the PeopleSoft driver files to the PeopleSoft server. This assumes that you already have a Metadirectory server installed on another server in your environment. See "Installing the Remote Loader" in the *Identity Manager 3.6 Installation Guide*.

As part of the installation, select the Utilities option and install the PeopleSoft Components. This installs the PeopleSoft Server Agent files and the Component Tester program. If you've already installed the driver files but did not install the PeopleSoft Components, you can run the installation program again to install only the PeopleSoft Components.

2.2 Copying the PeopleSoft Middleware Library File (psjoa.jar) to the Driver Directory

The PeopleSoft middleware library is located in the `web/psjoa` directory of the PeopleTools software distribution.

You must copy the `psjoa.jar` file, and any additional Component Interface API class libraries that your solution requires, to the driver's `\lib` subdirectory. By default, this is `Novell\NDS\lib` for local driver installation or `Novell\RemoteLoader\lib` for a remote installation.

Configuring Your PeopleSoft Environment

3

The PeopleSoft Server Agent (PSA) is a set of PeopleTools objects and code that enables you to define the integration between PeopleSoft applications and the Identity Vault. The following sections explain how the PSA works and how to configure your PeopleSoft environment:

3.1 Using the PeopleSoft Service Agent

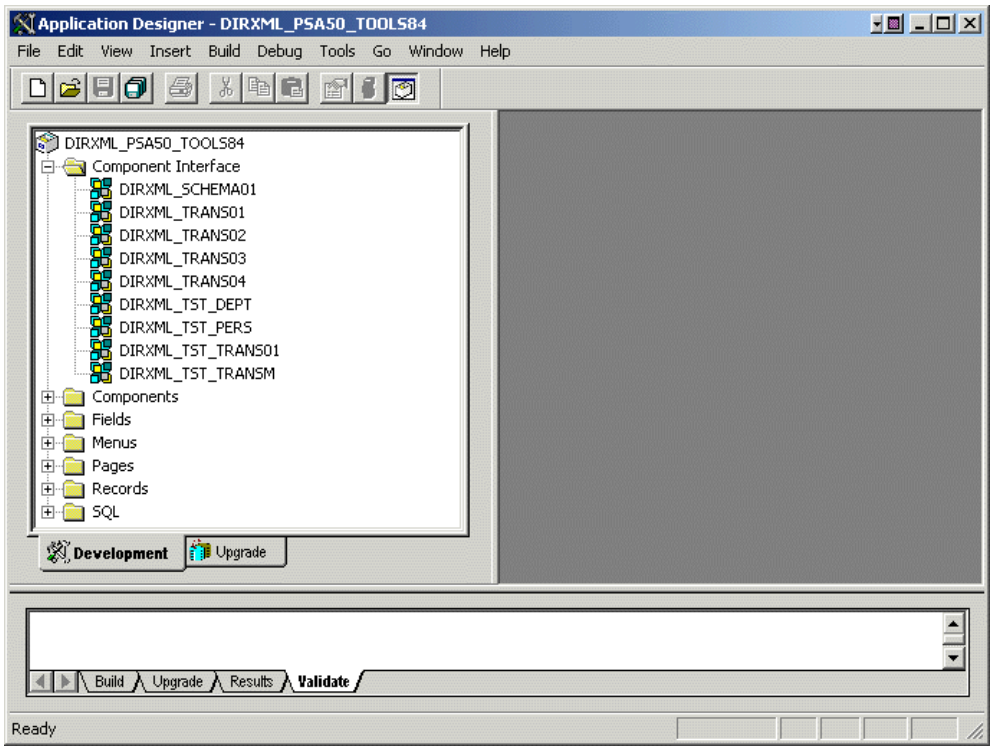
The PeopleSoft Server Agent (PSA) includes all of the components for a sample Personnel application, a staging table for moving data between the Sample application and the driver, a Transaction record interface for recording data events of interest to the driver, and utilities for managing the Transaction record interface. Using the sample data and code in the PSA, you can quickly model and implement an Identity Manager solution that is not intrusive to the existing functions and applications on your PeopleSoft system.

This version of the PSA works with any PeopleSoft database on the required release level of PeopleTools. Before you can install the PSA, you need access to a PeopleSoft user ID and password with Administrator or appropriate developer rights. You can create a unique user ID and password for implementing these objects.

The PSA contains SQLExec statements in the PeopleCode for the various Table and View records. There is no guarantee that all of these statements are compatible with the underlying database software. If you encounter problems, refer to Chapter 8, “Troubleshooting the Driver,” on page 71 for specific issues and consult with your DBMS/PeopleSoft Database administrator for additional assistance.

3.1.1 The Component Interface Infrastructure for Identity Manager

Figure 3-1 *The DirXML Component Interface*



You can use the Component Interface infrastructure and PeopleCode function calls to specify the type and content of Transaction records that are generated in relation to PeopleSoft Component events. You can also decide if the driver processes events and how they are processed. For example, a new row event generated by the sample application generates a slightly different event than a new row event generated by the driver's Subscriber channel.

For more information, refer to "Configuring PeopleCode to Trigger Transactions" on page 32.

3.1.2 The Sample Application

The PSA project has sample Personnel Applications that you can install on your PeopleSoft 8.1*x*, 8.2*x,* or 8.4*x* systems for configuration and testing purposes.

Depending on your business requirements, you should configure internal processes to trigger events into transaction tables, synchronize with other PeopleSoft tables, etc. either by replicating the provided PeopleCode or by merging the components within your PeopleSoft environment. When synchronizing data internally between application tables, you should always try to use the tools that provide the highest degree of data integrity checking. (For example, the Staging CI to Application CI synchronization in the PSA utilizes the PeopleCode CI interface to ensure proper syntax, translate table usage, related fields, etc. as it has been defined on the Application record.) For more information, refer to "Understanding the Architecture of the PSA Sample Project" on page 23.

3.2 Installing the PSA Sample Project

Complete the following tasks to install the sample project for testing and configuration purposes:

3.2.1 Installing the PSA Files

If you did not install the PSA during the initial driver installation, locate the product CD or download, go to the PeopleSoft application server, and run `install.exe`. You should see the following PSA files on your target server:

- `DIRXML_PSA50_TOOLS81.exe`
- `DIRXML_PSA50_TOOLS84.exe`

These files are self-extracting zip files that contain the PSA project folder and files for the respective 8.1*x* and 8.4*x* versions of PeopleTools. Extract the appropriate file onto the file system of your PeopleSoft Application Designer host (`c:\psa`). Use the `DIRXML_PSA50_TOOLS81.EXE` file for 8.2*x* PeopleTools deployments.

To ensure that the PSA can be imported into the PeopleSoft Application server, make sure that the PSA files have write access enabled. For example, in Windows*, you should turn off read-only file properties.

3.2.2 Importing the PSA Project into the PeopleSoft Database

After the PSA files have been installed in an accessible file system location, they must be imported into the PeopleSoft Database via the PeopleSoft Application Designer tool.

PeopleSoft 8.1*x* and 8.2*x*

1. Connect to the PeopleSoft database as administrator in two tier mode.
2. In the Application Designer, select *File* > *Copy Project From File*.
3. Click *Browse*, then select the PSA project directory: `c:\psa\psa-psa8`.
4. Click *Open*.
5. With all object types selected, click *Copy* to copy all project components into the PeopleSoft database.

PeopleSoft 8.4*x*

1. Connect to the PeopleSoft database as administrator in two tier mode.

2 In the Application Designer, select *Tools* > *Copy Files*> *From File* or *Tools*> *Copy Project*> *From File*.

3 Click *Browse* and select the PSA project directory: `c:\psa\DIRXML_PSA50_TOOLS84`.

4 Click *Open*.

5 With all object types selected, click *Copy* to copy all project components into the PeopleSoft database.

3.2.3 Building Project Record Definitions

After you have imported the project into the PeopleSoft database, you should build project record definitions and project views.

1 Log in to the PeopleSoft Application Designer by using an administrator username that has administrative and development rights.

2 In the Application Designer, select *Build* > *Project*.

3 From Build Options, click *Create Tables and Execute SQL Now*. After project tables are created, click *Close* to close the Build Progress window.

4 Click *Build* to create sample project tables.

You must create project tables before creating the views. Views are created with information from table fields.

5 In the Application Designer, select *Build* > *Project*.

6 In Build Options, click *Create Views and Execute SQL Now*.

7 Click *Build* to create the sample project views. After views are created, click *Close* to close the Build Progress window.

3.2.4 Applying Security to the PSA

In order for the driver to access PeopleSoft transaction tables, you need to apply security to the PSA. You accomplish this by creating the DirXML® Administrator role and then assigning it to the administrative user. You can assign the role to an existing account or create a new account specifically for PSA security.

- "PeopleSoft 8.1x and 8.2x" on page 22
- "PeopleSoft 8.4" on page 23

PeopleSoft 8.1*x* and 8.2*x*

1 In the Application Designer, click *Go* > *PeopleTools* > *Maintain Security*.

2 Click *Use* > *Roles* > *General* > *Add*.

3 In the *Add Role* field, type `DirXML Administration`, then click *OK*.

4 In the *Description* field, type `DirXML Administration`.

5 Click the *Permission Lists* tab, then click the drop-down arrow.

6 For the *Permission List* value, type `DirXML`, then click *OK*.

The *Description* field populates automatically.

7 Click *Save*.

8 Click *Use* > *User Profiles* > *General* > *Update/Display*.

9 Type your administrative user username as the User ID, then click *OK*.

10 Click the *Roles* tab, then click in the last row to add data.

11 Add the *DirXML Administration 4* role to this user, then click Save.

12 Close and restart the PeopleSoft clients and applications.

PeopleSoft 8.4

1 Log in to the PeopleSoft portal.

2 Click *PeopleTools* > *Security* > *Permissions & Roles* > *Roles*.

3 Click *Add a New Value*, then specify a role name (for example, *DirXML Administrator*).

4 Type a description for the role.

5 (Optional) Type a long description for the role.

6 Click the *Permissions List* tab.

7 Search for and select the DirXML permissions list, then click *Save*.

8 Assign *DirXML Administrator* role to your administrative user by clicking *PeopleTools* > *Security* > *User Profiles* > *User Profiles*.

9 Click *Search* to locate the User Profile that you want to add the DirXML Administrator role to, then click the *User ID*.

10 Click the *Roles* tab, then click one of the + buttons to add a new role.

11 Search and select the role, click *DirXML Administrator* to add it, then click *Save*.

3.2.5 Understanding the Architecture of the PSA Sample Project

The PSA Sample project is intended to provide recommended PeopleSoft integration scenarios through the use of very comprehensive instructions and examples. The various elements of the PSA are completely independent of any existing PeopleSoft application software, so there is no risk of data table corruption, extension, or modification.

The objects of the PSA include:

- Field definitions
- Record definitions
- Page definitions
- Component definitions
- Component Interface definitions
- Menu definitions
- SQL code

All of these objects are named with a prefix of DIRXML_ so that they cannot be confused with existing objects.

- “Sample Application” on page 24
- “Staging Table” on page 24

Sample Application

This application is intended to simulate the data and functions of an HR or other type of Person provisioning application. The base Record definitions for the application are:

- DIRXML_S_PERS: Provides basic HR field data
- DIRXML_S_DEPT: Sample Department codes table
- DIRXML_S_PHONES: Phone Numbers table

The data is accessed through the DIRXML_ADMINISTRATOR menu options. The menu provides access to a *DirXML Sample People* component and a *DirXML Sample Department* component. These applications simulate the standard methods for adding and updating Department and Person data into the PeopleSoft database. The driver default configuration does not directly access any of these tables or components. Additionally, the data provided by this application is not passed directly to the driver from these components. The actual transfer of data takes place through staging table components.

Staging Table

The staging table interface is designed to insulate the PeopleSoft Application data from direct manipulation by the driver. This allows an interface to be designed to:

- Combine access to the data from multiple data tables and applications through a single interface.
- Prevent the driver from viewing or modifying sensitive application data.
- Provide storage for external data that does not easily fit into standard PeopleSoft applications.

The Record definition that represents all of the data that can be published or subscribed by the driver is called DIRXML_STAGE01. This record aggregates most of the data fields from the three Application data records into a single access point. There are also additional fields not in the Application records that are used to contain references to the synchronized eDirectory™ objects.

For PeopleSoft users, the data in the staging table can be accessed via the DirXML Schema 01 component of the DIRXML_ADMINISTRATOR menu. The driver accesses the data via the DIRXML_SCHEMA01 Component Interface.

Transaction Table

Every time a modification is made to the Application Data Records, transaction records are placed in the DIRXML_TRANS01 table. The PSA places identifiers indicating the key of the data row being modified, the time of the event, the type of event, and various other pieces of transaction related data. Any change made to a data row via the DIRXML_ADMINISTRATOR application interfaces is recorded with an NPSDriver1P identifier that shows the data is being published by a PeopleSoft administrator. Changes made via the Component Interface API are recorded with an NPSDriver1S identifier that shows the data was subscribed into the application tables programmatically.

In addition to providing an audit trail of database modifications, the transaction table is utilized by the driver to facilitate Publisher channel activities. The driver polls the transaction table for records in the *Available* state, reads the related application data record, and processes the data through the Publisher channel. The transaction records are then updated with the processing status.

In addition to the transaction tables and interfaces, the PSA includes utilities to monitor, maintain, archive, and remove transaction records.

PSA Best Practices

Data moves between the various PeopleSoft Components and tables through PeopleCode. Each of the application data records in the PSA contains PeopleCode that performs the basic functions of moving data between the Staging table and Application tables, ensuring the integrity of the data and data transfers, and generating Transaction table records at the appropriate time and with the appropriate data. PeopleCode is very powerful and is capable of performing a wide variety of tasks, some of which are potentially destructive to your data.

IMPORTANT: Only personnel who have completed PeopleTools and PeopleCode training should modify the elements of the PSA.

These guidelines might prove helpful when implementing changes to the PSA.

- Whenever possible, always provide a Component Interface (CI) to affected data tables. In the PSA, the DIRXML_S_PERS data rows are created and updated with PeopleCode via the DIRXML_TST_PERS CI. The CI guarantees the integrity of the data as defined by the designer of the application. It ensures valid Translate values, proper data format, required fields are present. Most importantly, the CI can restrict the data fields that can be accessed on a particular record or record set. This is a very important aspect of data security.
- The DIRXML_SCHEMA01 CI has been extended with a *Delete* method that enables removal of data schema records via the driver's Subscriber channel. Using this functionality is not required. The method can be removed from the CI or the driver can be configured to not use it.
- Make sure that the staging table record contains the same required fields that exist on the target Application records. This helps ensure successful record data synchronization.
- It is important to generate transactions whenever the application data table records are created or updated, even if the changes are made by the driver. Although data loopback can occur, the generation process ensures that Translate table values and related field values generated by the changes are properly synchronized.
- If you are using SQLExec() statements to update tables or create records, use great care to ensure that you are not violating the logic and rules of the applications overlying the tables. SQL is the easiest and most powerful, and therefore most destructive, method for updating data.
- Do not generate Transaction records until after you have successfully updated the application tables.
- If desired, it is possible to completely bypass the staging table interface in your synchronization scenario. The driver can be directed to interact with any CI. Make sure that the PeopleCode generating the transaction records is updated to specify the new Application data CI and is triggered appropriately. Also ensure that the same CI methods are implemented and enabled.

- The driver is delivered with a Java archive (JAR) file that contains the compiled Java interfaces for all of the CI defined in the PSA. If the driver is to be configured to use different application CIs, it is necessary to build and JAR those interfaces. For more information, refer to Section 6.2, "Changing the Data Schema Component Interface," on page 53.
- Test everything thoroughly.

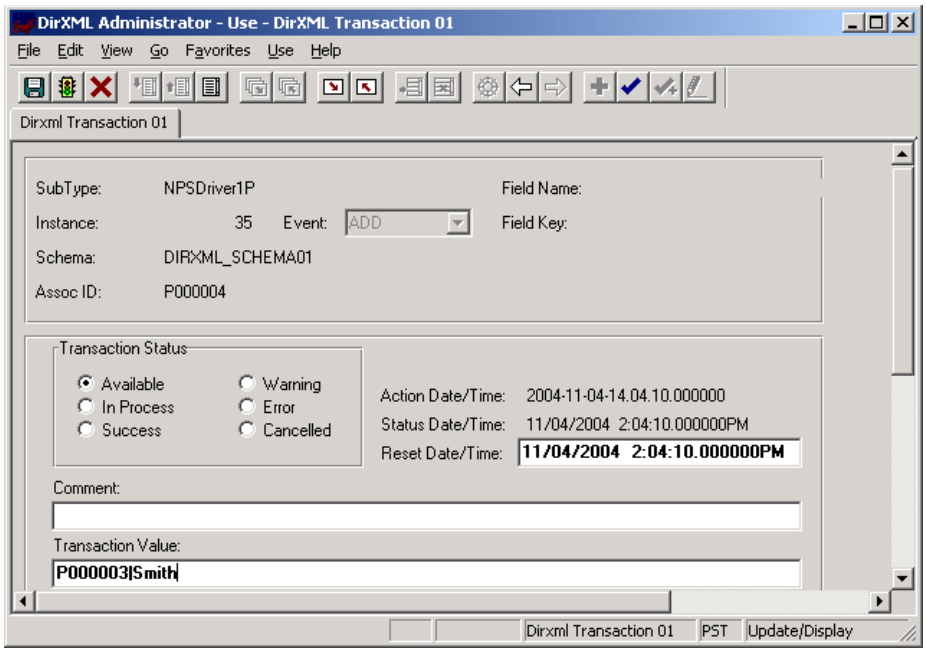
3.2.6 Testing Sample PeopleSoft Applications

You can test to ensure that transactions are created by entering a new person through the PeopleSoft Identity Manager sample application. This example uses Departments, so you need to create a sample department and then add a person (assigning him or her to that department) to validate that the application works. The following information explains how to test your sample applications.

PeopleSoft 8.1*x* and 8.2*x*

1. In the Application Designer, select *Go* > *DirXML Administrator*.
2. In the *DirXML Administrator* menu, select *Use* > *DirXML Sample Department*.
3. Click an empty *Department* field row to add sample department and description values.
4. Click *Save* to add the Department.
5. From the *DirXML Administrator* menu, select *Use* > *DirXML Sample People* > *Add*.
6. Type data into the various fields for the new person, then click *Save*.

   Asterisks represent required fields.
7. Verify that an *ADD* transaction was created by selecting *Use* > *DirXML Transaction 01*.
8. Click the *Search* button.
9. Verify that the transaction was created and select the transaction.

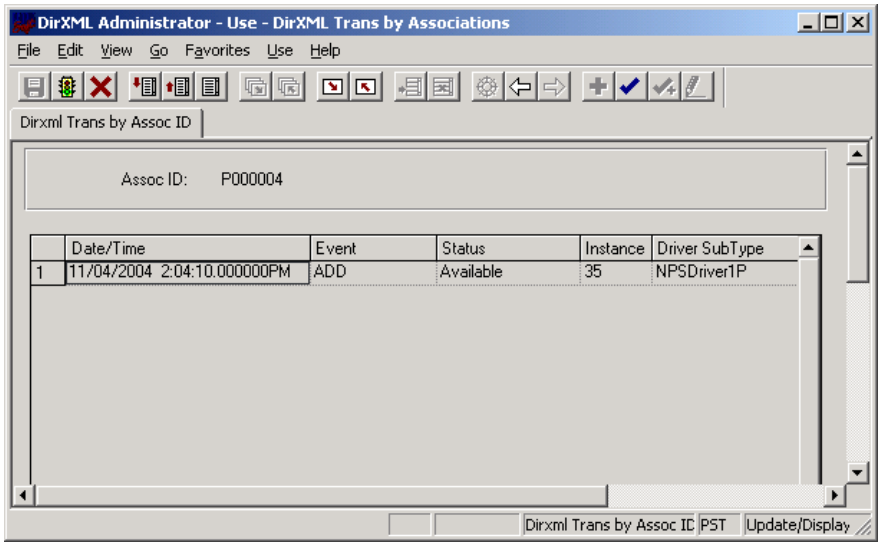


10 Click *Use* > *DirXML Schema 01* > *DirXML Schema01A*.

11 Verify the Schema data on the first tab (*Schema 01 A*).

12 Verify that you can update the fields on the second tab (*DirXML Schema 01 B*).

13 Click *Use* > *DirXML Trans by Associations* and verify that you can view the data.



14 Click *Use* > *DirXML Driver Defaults* and verify that you can view the sequence of transactions.

15 Verify that other Transaction table applications work by clicking *Use* > *DirXML Transaction 02* (*03*, *04*, etc.), and *DirXML Trans Maintenance*.

You can create additional transaction tables (Transaction 05, Transaction 06, and so forth.) The delivered sample application is configured to use only *Transaction01*.

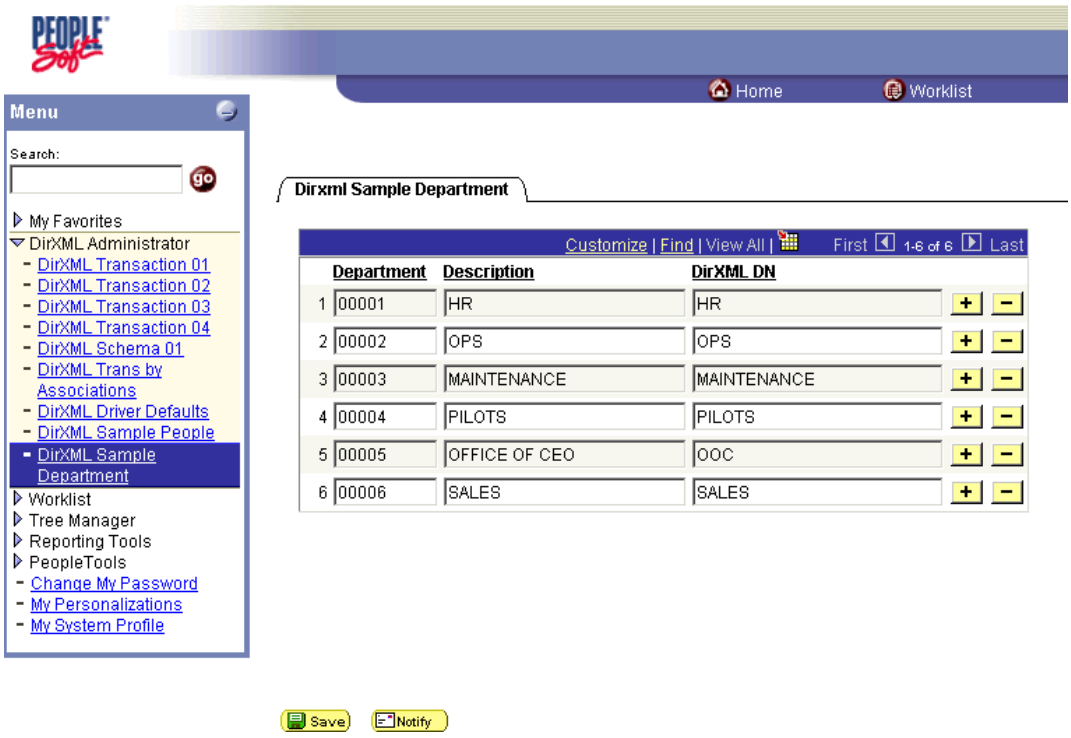
PeopleSoft 8.4*x*

1 Log in to the PeopleSoft portal.

2 Click *DirXML Administrator* from the left menu.

If the *DirXML Administrator* menu doesn't appear, you should delete the Application Server cache and reboot the Application Server.

3 Click *DirXML Sample Department*.



4 Specify a sample department, then click *Save*.

5 Click *DirXML Sample People*.

6 Enter values for a sample user, then click *Save* and verify that the user's data appears in the Transaction01 table. You do this by searching in the DirXML Transaction01 application.

7 Verify that other delivered applications work by selecting them from the *DirXML Administrator* menu.

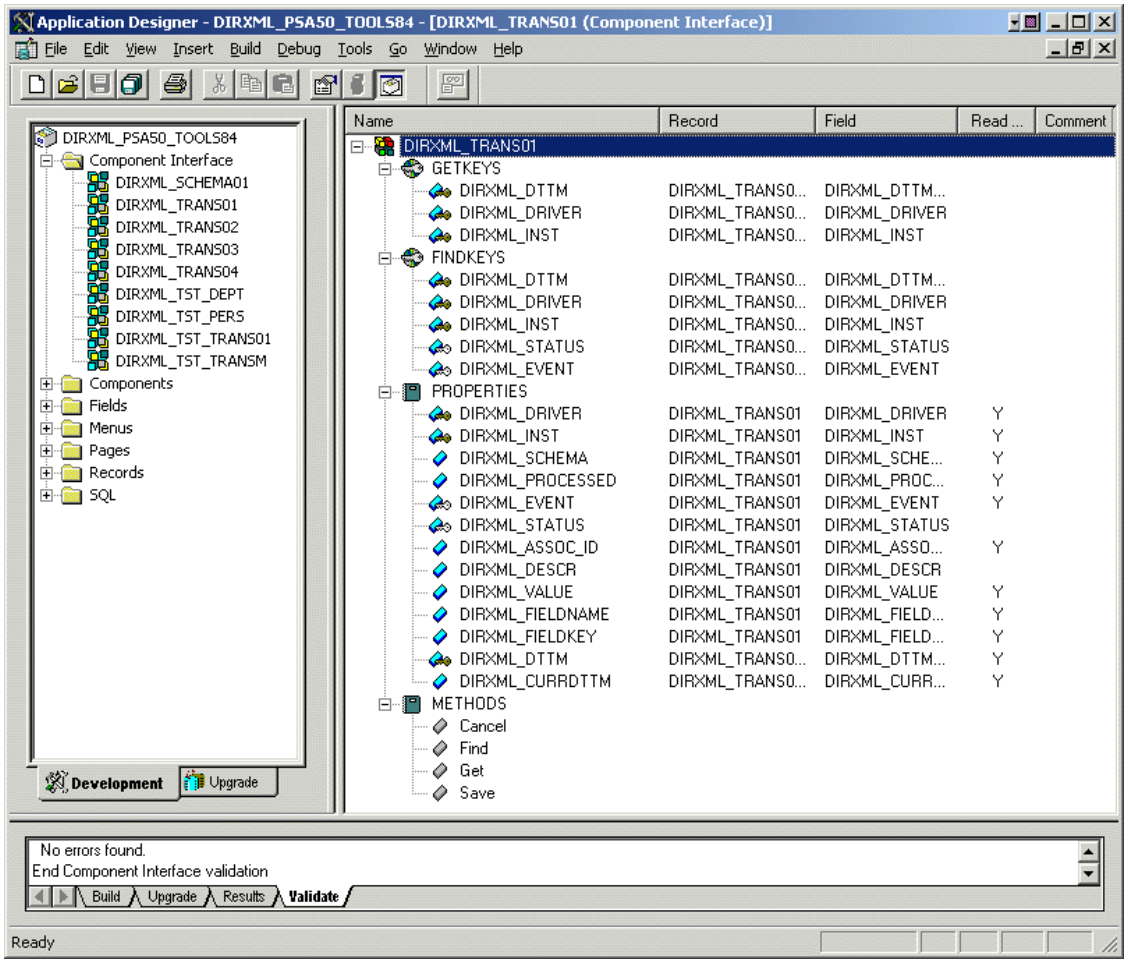
3.3 Component Interfaces

3.3.1 Accessing Transactions and Data through Component Interfaces

The driver accesses transactions waiting to be processed from the transaction table via the Component Interface (CI) object that is defined within PeopleTools. Each CI maps to a particular component. Components are built in order to access transaction tables and “schema” application object data. Schema objects represent all the necessary fields and methods that need to be exposed for data synchronization to the driver. These objects also enable the driver to update PeopleSoft data.

Each driver uses only one Transaction CI to access transactions. Every transaction is assigned to one default Schema CI. In the driver’s parameters, you must specify the Transaction CI object name as defined in PeopleSoft. This CI object maps to a predefined component that enables the driver to access transactions from one transaction table. The following represents the CI for a transaction table:

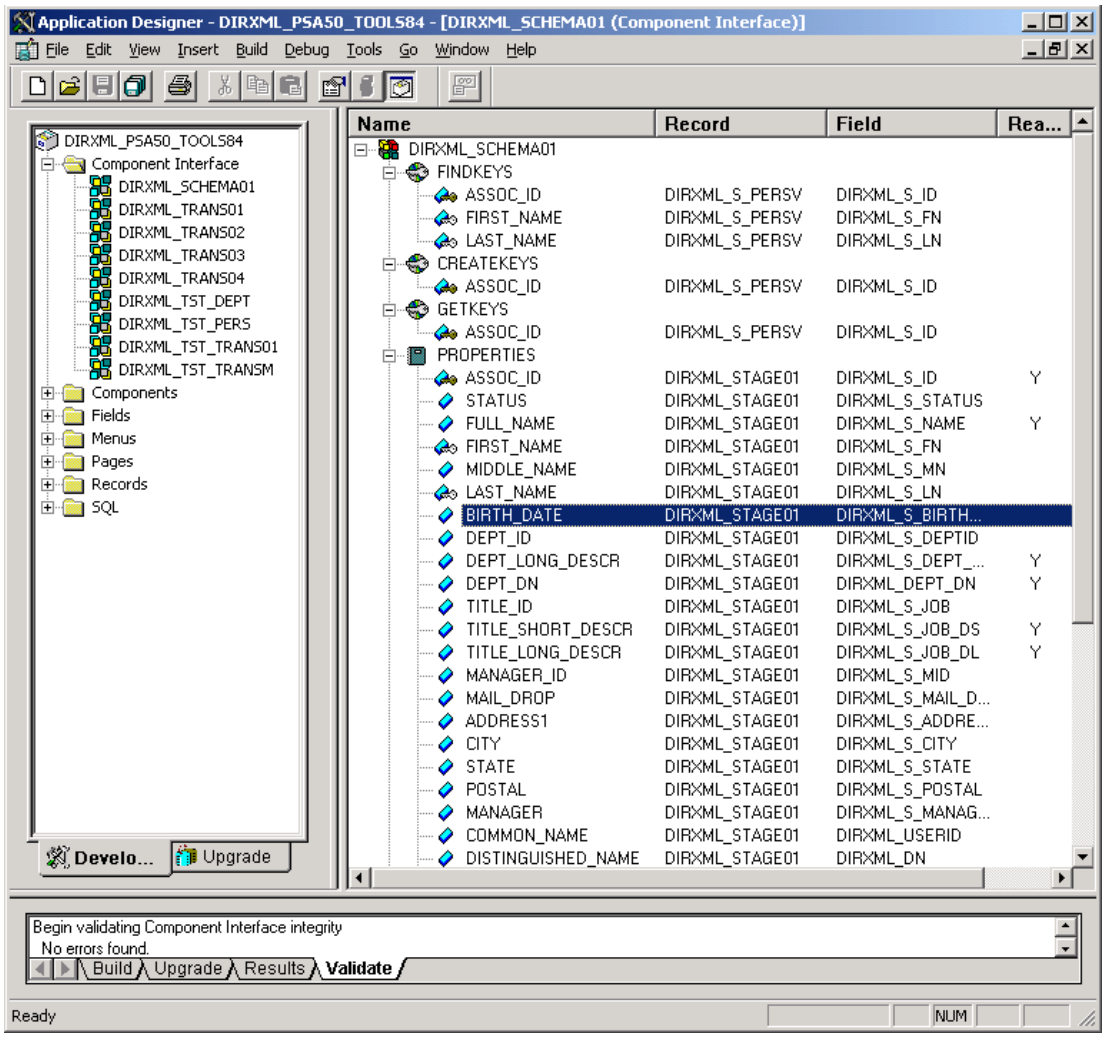
Figure 3-2 *Transaction Table Component Interface (CI)*



In addition to the Transaction CI name, the driver configuration contains a parameter that can specify a particular subset of the transactions that are available for processing. This allows a single Transaction CI to interface with multiple drivers, which may be synchronizing different sets of object data or different object types. This subset identifier is maintained in the DIRXML_DRIVER field of the Transaction CI.

The following figure represents the CI for a Schema Component:

Figure 3-3 *Schema Component*



The PeopleSoft developer can specify these values when configuring the PeopleCode function calls to trigger a transaction online, or when creating transactions via a batch process. The PSA provides PeopleCode function DirXML_Trans and is responsible for generating transaction events. The PSA contains several function calls which you may use to guide customization.

The PeopleCode DirXML_Trans schema calls should always be placed in the SavePostChange PeopleCode on the record definitions. This ensures that the data is committed prior to the transaction being generated.

Changes to Field Names in PeopleSoft 8.41

With new releases of PeopleTools, changes are made to the policies regarding field names. With PeopleTools 8.41, there were two significant changes:

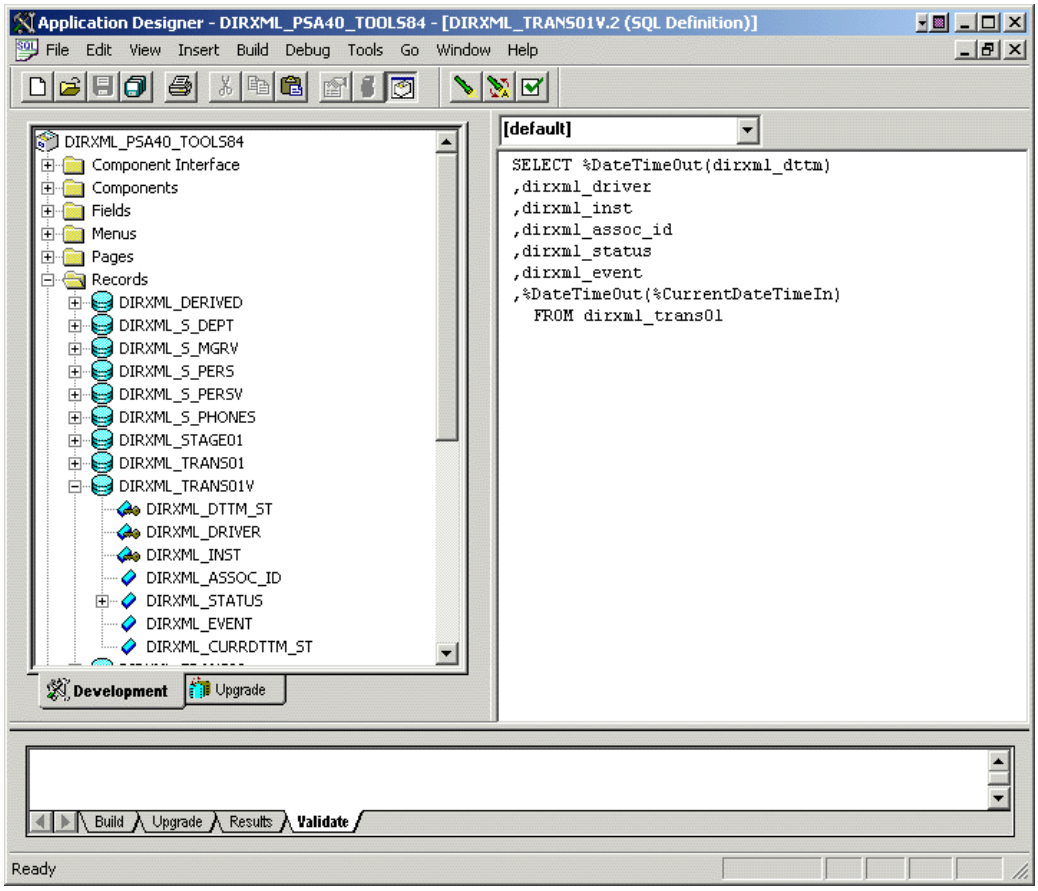
- Spaces are no longer allowed in CI field names.

- There are now case sensitivity issues in the CI API. Field names and field name values no longer align because of case-sensitive sorting. For example, a field named CN is sorted prior to a field named City. The result of trying to access the value of City returns the value for CN. The default schema of the CIs used by the driver now uses naming conventions that eliminate this unusual sorting error. This issue is particularly important for any field name modifications or additions customized for a prior implementation of the driver.
- Use the standard ALL-CAPS format to avoid any field name issues.

3.3.2 Configuring the Transaction Record SQL Date/Time Format

The proper functioning of the driver depends on the Date/Time strings in the Transaction View record to determine processing availability and relative event order of Transaction data rows. The Date/Time fields in the Transaction data rows are converted to formatted strings in the Transaction Views by using the PeopleCode Meta-SQL%DateTimeOut() function. The following image shows the default SQL View code for the DIRXML_TRANS01V record:

Figure 3-4 *SQL View of Code for the DirXML_TRANS01V Record*



Unfortunately, the format of the strings presented by %DateTimeOut() might differ depending on the underlying DBMS software. To make sure a date and time string is generated in a consistently increasing lexicographic format, the following format is recommended:

- The date should be presented first in *YYYY-MM-DD* format.

- The time should be presented in 24-hour form with *HH:MM:SS* format (Additional information concatenated to this string, such as <milliseconds> is acceptable)
- These two strings should be placed together in “date-time” format.

The characters used to delimit the numerical values are not important as long as they are consistent. Examples of a well-formed, lexicographically ordered format are:

2004-08-26-14.44.33.000000 (ODBC Canonical style 121)

or

2004/08/26-14:44:33 (Generic)

The fields used by the driver are DIRXML_DTTM_ST and DIRXML_CURRDTTM_ST. These fields represent the date and time that the Transaction data row was created and the current processing date and time of the transaction.

If you are using a driver trace level of 2 or above, the driver traces the CurrDate and ActionDate of each Transaction row that it processes. If the format shown does not match the criteria specified, edit the SQL of the desired Transaction View record to perform the appropriate conversions on these fields. Make sure that you use the *Build* > *Create Views* option after making any modifications to the SQL definition of the View Record.

Because the configuration of the Date and Time format varies depending on the DBMS being utilized, changing this format should be done by the DBMS/PeopleSoft Database Administrator or other qualified personnel.

3.3.3 Configuring PeopleCode to Trigger Transactions

The PSA contains a number of PeopleTools objects that enable PeopleSoft to trap events into a transaction table. The driver then accesses the transaction tables through CI objects. The driver periodically requests transactions that need to be processed-based on their driver subtypes. It processes only those transactions that have a transaction date or time less than or equal to the current date or time value, along with an available status. Also, the driver processes transactions one at a time from the transaction table before getting a new transaction.

The driver then constructs an XML document from the data it retrieved and passes this to the Metadirectory engine for processing. It updates the transaction status and any applicable messages on the transaction table inside of PeopleSoft after processing is completed by the Metadirectory engine. When events occur within eDirectory, the driver connects to the appropriate CI and updates the PeopleSoft staging table as appropriate.

You trigger transactions using PeopleCode within the PeopleSoft application.This document assumes that you know how to write PeopleCode. If you need further assistance, refer to PeopleSoft documentation for more information.

The driver requires a Transaction CI and a Schema CI to process transactions. For more information on calling the PeopleSoft function that creates transaction records, please refer to “Customizing the PSA by Triggering Transactions” on page 51.

- “Transaction Component” on page 33
- “Schema Component” on page 33

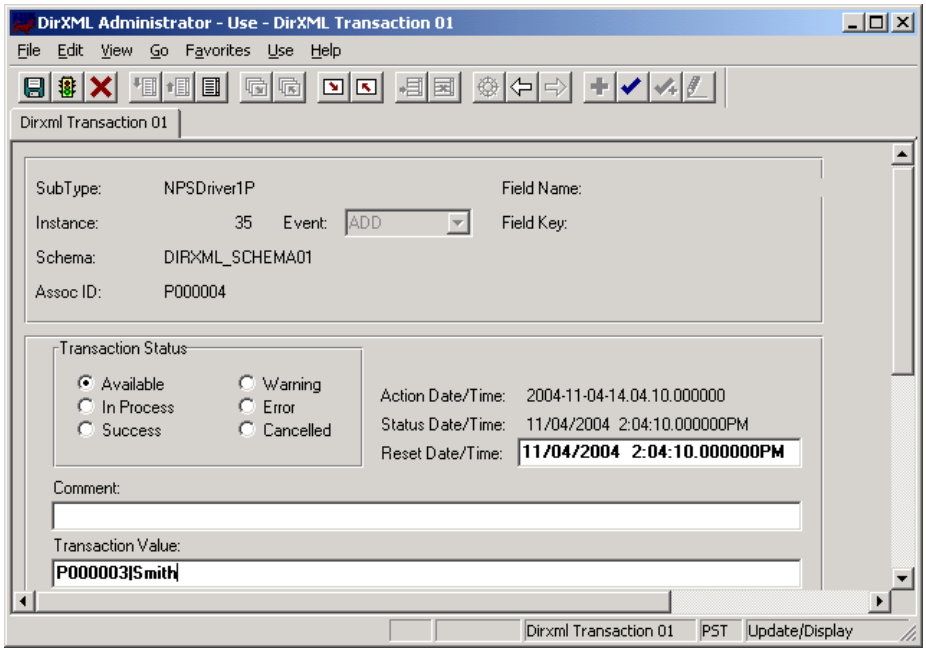
Transaction Component

The Transaction Component interface enables the driver to request transactions by driver subtype, date and time, and event type. The driver requests a single transaction for processing and obtains the object ID for the record being processed.

When the driver selects the first transaction to process, it sets the status of the transaction to In Process. The driver then retrieves the Event Name (DIRXML_EVENT), which it uses to create an Add, Modify, or Delete XML document. The driver uses the Schema ID (DIRXML_SCHEMA) and the Associate ID (DIRXML_ASSOC_ID) values to access the appropriate CI Schema and appropriate record information associated with the object.

After the transaction has been processed by the Metadirectory engine, the driver updates the status of the transaction (DIRXML_STATUS) and updates the *Comment* field (DIRXML_DESCR) if an error or warning message is applicable.

Figure 3-5 *DirXML Transaction01*



Schema Component

The Schema Component interface lets the driver retrieve data for a particular record and update the PeopleSoft staging table for that record. After the driver retrieves the Association ID (DIRXML_ASSOC_ID) and Schema name (DIRXML_SCHEMA), it accesses the appropriate Schema object.

The driver also uses Schema CI as a Class identifier for object type matching. When the driver accesses the Schema CI, it uses the value it received in the Association ID (DIRXML_ASSOC_ID) as the key value to retrieve the data from the PeopleSoft environment. It also uses this CI to update PeopleSoft records.

For example, assume you want to process transactions for an employee with the DIRXML_ASSOC_ID field (key) value = P000001. The driver accesses the Schema CI with a key value of P000001. It retrieves all of the configured component elements that have been defined for

that employee. The driver then converts the data collection into XML documents to be consumed by the Metadirectory engine. If there is a write-back command processed, or when data is written on the Subscriber channel, the driver uses this CI to update the staging table with the appropriate information into PeopleSoft for this particular employee.

3.3.4 Testing Component Interfaces

The Component Tester program (`CITester.class`) is included as part of the driver package. The program validates the proper installation, configuration, and revision of the PeopleSoft PeopleTools client software on the computer hosting the Identity Manager Driver for PeopleSoft. The program also validates a selected Transaction CI and Schema CI. Therefore, successful operation of CITester helps ensure the proper client functionality for the driver.

The CITester completes the following checks during four phases:

- Phase I: Ensures that a PeopleSoft client session can be created.
- Phase II: Ensures that connection and authentication parameters to the PeopleSoft Application Server are correct.
- Phase III: Verifies that the Transaction CI required fields and keys are present.
- Phase IV: Verifies that the Schema CI required fields and keys are present.

If you are not using the default Schema CI, it is necessary to build the APIs for the desired CI. See Section 6.2, “Changing the Data Schema Component Interface,” on page 53 for information on building custom CI API JAR files.

There might be variations of the error message data depending on the PeopleTools release. The CITester program runs all platforms supporting Java 1.3.1 or later and uses the Java PeopleSoft Component Interface (`psjoa.jar`) package from the PeopleTools distribution.

Important Considerations

When you run the test, you must either:

- Set the CLASSPATH environment variable to include the path of the `CITester.class`, `psjoa.jar` and `dirxmlcomps.jar` files. If a custom CI API JAR file is required, include it in CLASSPATH.
- Set the *-classpath* option on the Java command line to include the `CITester.class`, `psjoa.jar` and `dixmlcomps.jar` files and any required custom CI API JAR files.

It is also important to note that the `java.exe` for JRE*/JDK* version 1.3.1 or later must be installed and accessible via the PATH environment variable or be explicitly called out from the Java command line.

How Do I Run the Test?

From a command shell, execute the `CITester.class` test file. A sample `CITester.bat` file is provided as a reference that indicates the correct syntax and class files required to execute the test and the driver. To accept the test's default values, press Enter. In Phase II, you are required to enter a value for the Application Server Name.

The test writes output to the screen and to `CITesterOutput.txt`. The output file is written to the location where `CITester.class` resides.

Phase I: Creating a PeopleSoft Client Session

If the test program establishes a session with the PeopleSoft client, you see the following message:

```
** PeopleSoft client session established successfully.
```

You might encounter the following errors during the test.

Table 3-1 *Phase I Errors and Solutions*

Error Message	Solution
`Exception in thread "main" java.lang.NoClassDefFoundError: psft/pt8/util/PSProperties.`	You must add a path to the 8.1x or 8.4x `psjoa.jar` file to the environment CLASSPATH variable or set the path to the `psjoa.jar` file in the Java command line. This error also occurs if an invalid version of the JVM* (JRE/JDK) is being used. Refer to the Important Considerations at the beginning of this section for more information.

Phase II: Authenticating to the PeopleSoft Client

1. Specify the Application Server Name or IP address. Forward slashes are required when entering the Application Server Name (for example, //255.255.255.255).
2. Specify the Application Server Jolt Port Number.
3. Specify the PeopleSoft UserID
4. Specify the PeopleSoft UserID Password.

If the test program verifies the connection and authentication parameters, you see the following message indicating success:

```
** The Connection and Authentication Parameters are verified to be correct.
```

You might encounter the following errors during the test.

Table 3-2 *Phase II Errors and Solutions*

Error Message	Solution
ERROR: Failed Connection to the PeopleSoft Application Server. Please make sure you entered your authentication information correctly. PeopleSoft Error/Warning Messages Pending. Number of Messages: 1 Message 1: Connect Failed: No additional information available (90, 01)	The target Application Server generates error and warning messages. This error indicates that you entered the wrong Application Server Name or Application Server Port Number. Ensure that the Server Name or address you entered contains a leading double slash (//) and that the address and name data is correct. Also, verify that you entered the Jolt port configured on the Application Server.
ERROR: Failed Connection to the PeopleSoft Application Server. Please make sure you entered your authentication information correctly. PeopleSoft Error/Warning Messages Pending. Number of Messages: 1 Message 1: DOWNbea.jolt.ServiceException: Invalid Session	This message indicates an invalid Application Server Name or Port Number. In some instances, if an invalid port number is specified, the CITester program hangs and requires a manual interrupt.
PeopleSoft Error/Warning Messages Pending Number of Messages: 2 Message 1: PeopleTools release (8.<num>) from web server ’’ is not the same as Application Server PeopleTools release (8.<num>) Access denied.	This message indicates that the PeopleTools version of the specified `psjoa.jar` does not match the version of the target PeopleSoft Application Server. PeopleTools requires a version match of the client and server.
PeopleSoft Error/Warning Messages Pending. Number of Messages: 3 Message 1: <UserID>@<Client computer> is an Invalid User ID, or you typed the wrong password. User ID and Password are required and case-sensitive. Make sure you’re typing in the correct upper and lower case. Message 2: Failed to execute GetCertificate request Message 3: Invalid certificate for user <User ID>	The target Application Server generates the error and warning messages. Either the User ID or User ID Password are incorrect or have been entered by using incorrect case.

Phase III: Authenticating to the PeopleSoft Client

The driver uses a Component Interface (CI) to access application modification transaction records from the Application Server. The field definitions of this interface must be identical to the DIRXML_TRANS01 CI delivered with the driver. This test phase validates the field definitions of the named Transaction CI.

Enter the Transaction CI name or press Enter to validate DIRXML_TRANS01.

If the test program retrieves and validates that all required fields and elements are present, you see the following message:

```
** Retrieval of Transaction Component Interface "DIRXML_TRANS01" succeeded.

- Property 'DIRXML_ASSOC_ID' is present.
- Property 'DIRXML_CURRDTTM' is present.
- Property 'DIRXML_DESCR' is present.
- Property 'DIRXML_DRIVER' is present and validated as key field.
- Property 'DIRXML_DTTM' is present.
- Property 'DIRXML_EVENT' is present.
- Property 'DIRXML_FIELDKEY' is present.
- Property 'DIRXML_FIELDNAME' is present.
- Property 'DIRXML_INST' is present and validated as key field.
- Property 'DIRXML_PROCESSED' is present.
- Property 'DIRXML_SCHEMA' is present.
- Property 'DIRXML_STATUS' is present.
- Property 'DIRXML_VALUE' is present.

** Transaction Component Interface element validation succeeded.
```

You might encounter the following errors during the test.

Table 3-3 *Phase III Errors and Solutions*

Error Message	Solution

`PeopleSoft Error/Warning Messages Pending.`
`Number of Messages: 4Message 1: Cannot find Component Interface {<Transaction CI Name>} (91,2)`
`Message 2: Initialization Failed (90,7)`
`Message 3: Not Authorized (90,6)`
`Message 4: Failed to execute PSSession request`
 	
`ERROR: Retrieval of Component Interface <Transaction CI Name> failed.`
	The target Application Server generates error and warning messages. This error indicates that the Transaction CI Name specified does not exist or cannot be found by the Application Server. Ensure that you specified the correct name.

`-Property '<property field name>' is not required.`	This is an advisory message. It indicates that an additional field or fields that are not required by the driver have been defined in the specified Component Interface.

Error Message	Solution
`ERROR: Transaction Component Interface element validation failed. Required Fields are not all present.`	The Transaction Component Interface specified does not contain all of the fields required by the driver. Verify that you entered the proper Transaction Component Interface name and validate that all fields contained in the default DIRXML_TRANS01 Component Interface are present.
`ERROR: Property '<Key field name>' is not defined as key field.`	A field in the Transaction Component Interface is present, but is not properly configured as a key field. The Transaction Component Interface DIRXML_DRIVER and DIRXML_INST fields must be specified as key fields.

Phase IV: Retrieving the Schema Component Interface

The Schema CI defines the application data that is to be synchronized via the driver. The specified Schema CI must contain a primary key field that is specified via the Data Record ID field name.

To test the Schema CI, type the Schema CI name or press Enter to retrieve DIRXML_SCHEMA01. Enter the Data Record ID field name. If the test program retrieves and validates that all required fields and elements are present, you see the following message:

```
** Retrieval of Schema Component Interface "DIRXML_SCHEMA01" succeeded.

- Property 'ASSOC_ID' is present and validated as key field.
- Property 'STATUS' is present.
- Property 'FULL_NAME' is present.
- Property 'FIRST_NAME' is present.
- Property 'MIDDLE_NAME' is present and validated as key field.
- Property 'LAST_NAME' is present.
- Property 'BIRTH_DATE' is present.
- Property 'DEPT_ID' is present.
- Property 'DEPT_LONG_DESCR' is present.
- Property 'DEPT_DN' is present.
- Property 'TITLE_ID' is present.
- Property 'TITLE_SHORT_DESCR' is present.
- Property 'TITLE_LONG_DESCR' is present.
- Property 'MANAGER_ID' is present.
- Property 'MAIL_DROP' is present.
- Property 'ADDRESS1' is present.
- Property 'CITY' is present.
- Property 'STATE' is present.
- Property 'POSTAL' is present.
- Property 'MANAGER' is present.
- Property 'COMMON_NAME' is present.
- Property 'DISTINGUISHED_NAME' is present.
- Property 'DESCRIPTION' is present.
- Property 'EMAIL_ID' is present.
- Property 'PHONE_BUSN' is present.
- Property 'PHONE_CELL' is present.
- Property 'PHONE_HOME' is present.
- Property 'PHONE_PGR' is present.

** Schema Component Interface element validation succeeded.
```

```
** All expected platform support is verified correct.
```

You might encounter the following errors during the test.

Table 3-4 *Phase IV Errors and Solutions*

Error Message	Solution
PeopleSoft Error/Warning Messages Pending. Number of Messages: 4 Message 1: Cannot find Component Interface {<Schema CI Name>} (91,2) Message 2: Initialization Failed (90,7) Message 3: Not Authorized (90,6) Message 4: Failed to execute PSSession request ERROR: Retrieval of Component Interface <Schema CI Name> failed.	The target Application Server generates error and Warning messages. This error indicates that the Schema CI Name specified does not exist or cannot be found by the Application Server. Ensure that you specified the correct name.
ERROR: Specified Schema Component Interface Data Record ID Field '<Data Record ID Field Name>' not found.	The field name specified as the key field of the Schema Component Interface is not in the Component Interface definition. Verify that you entered the proper field name.
ERROR: Property '<Data Record ID Field Name>' is not defined as key field.	The field name specified as the key field of the Schema Component Interface is present but is not properly defined as the key field. Validate the Component Interface definition or verify that the proper field name was specified.

Summary

At the completion of the test, the program provides a summary containing the results of the test. The validated parameters are shown below in the summary.

```
Component Interface Test Summary
--------------------------------
Full Component Interface Functionality has been verified.
The following parameters may be used for PeopleSoft 5.0 Driver Configuration

Authentication ID           : PSADMIN
Authentication context     : //255.255.255.255:9000
Application Password      : PSADMIN
Schema CI Name             : DIRXML_SCHEMA01
Data Record ID Field       : ASSOC_ID
Transaction CI Name       : DIRXML_TRANS01
```

Creating a New Driver

4

After the PeopleSoft driver files are installed on the server where you want to run the driver (see Chapter 2, “Installing the Driver Files,” on page 17) and you’ve configured your PeopleSoft environment (see Chapter 3, “Configuring Your PeopleSoft Environment,” on page 19), you can create the driver in the Identity Vault. You do so by importing the basic driver configuration file and then modifying the driver configuration to suit your environment. The following sections provide instructions:

4.1 Creating a PeopleSoft Account

The driver requires an administrative account for the PeopleSoft system. The driver uses this account to authenticate to PeopleSoft and make changes. You can use an existing administrative account; however, we recommend that you create an administrative account exclusively for the driver.

4.2 Creating the Driver in Designer

You create the PeopleSoft driver by importing the driver’s basic configuration file and then modifying the configuration to suit your environment. After you’ve created and configured the driver, you need to deploy it to the Identity Vault and start it.

4.2.1 Importing the Driver Configuration File

1. In Designer, open your project.
2. In the Modeler, right-click the driver set where you want to create the driver, then select *New* > *Driver* to display the Driver Configuration Wizard.
3. In the Driver Configuration list, select *PeopleSoft50*, then click *Run*.
4. On the Import Information Requested page, fill in the following fields:

   Driver Name: Specify a name that is unique within the driver set.

   Active Users Container: Specify the name of the Organizational Unit object in the Identity Vault where Active users from PeopleSoft are placed. You can modify this parameter through a global configuration value (GCV) after installation.

Inactive Users Container: Specify the name of the Organizational Unit in the Identity Vault where Inactive users from PeopleSoft are placed.You can modify this parameter through a GCV after installation.

Active Employee Group Object: Specify the name of the Group object in the Identity Vault to which Active Employee users from PeopleSoft are added. You can modify this parameter through a GCV after installation.

Active Manager Group Object: Specify the name of the Group object to which Active Manager users from PeopleSoft are added. You can modify this parameter through a GCV after installation.

Driver is Local/Remote: Select *Local* if this driver will run on the Metadirectory server without using the Remote Loader service. Select *Remote* if you want the driver to use the Remote Loader service, either locally on the Metadirectory server or remotely on another server.

PeopleSoft User ID: Specify an administrative account through which the driver can authenticate to PeopleSoft.

Connection to PeopleSoft Server: The hostname or IP address and port number for connecting to the appropriate PeopleSoft application server. This is typically referred to as the PeopleSoft application server connection string. The default port is 9000.

The connection string uses the following format:

```
<hostname or IP address>:<Jolt Port Number>
```

Example: `//PSServer:9000`

To enable failover and load balancing, you can supply multiple server connection strings separated by a comma.

Example: `//PSServer:9000,//111.222.3.4:9000`

5 (Conditional) If you chose to run the driver remotely, click *Next*, then fill in the fields listed below. Otherwise, skip to Step 6.

Remote Host Name and Port: Specify the host name or IP address of the server where the driver's Remote Loader service is running.

Driver Password: Specify the driver object password that is defined in the Remote Loader service. The Remote Loader requires this password to authenticate to the Metadirectory server.

Remote Password: Specify the Remote Loader's password (as defined on the Remote Loader service). The Metadirectory engine (or Remote Loader shim) requires this password to authenticate to the Remote Loader

6 Click *Next* to import the driver configuration.

At this point, the driver is created from the basic configuration file. To ensure that the driver works the way you want it to for your environment, you must review and modify the driver's default configuration settings.

7 To review or modify the default configuration settings, click *Configure*, then continue with the next section, Configuring the Driver.

or

To skip the configuration settings at this time, click *Close*. When you are ready to configure the settings, continue with Configuring the Driver.

4.2.2 Configuring the Driver

After importing the driver configuration file, you need to configure the driver before it can run. You should complete the following tasks to configure the driver:

- **Ensure that the driver can authenticate to PeopleSoft:** Make sure that you've established a PeopleSoft administrative account for the driver (see Section 4.1, "Creating a PeopleSoft Account," on page 41) and that the correct authentication information, including the User ID and password, is defined for the driver parameters (see Section A.1.3, "Authentication," on page 78).
- **Configure the driver parameters:** There are many settings that can help you customize and optimize the driver. The settings are divided into categories such as Driver Configuration, Engine Control Values, and Global Configuration Values (GCVs). Although it is important for you to understand all of the settings, your first priority should be to configure the driver parameters located on the Driver Configuration page. For information about the driver parameters, see Section A.1.5, "Driver Parameters," on page 80.
- **Configure the driver policies and filter:** Modify the driver policies and filter to implement your business policies. For instructions, seeSection 6.3, "Modifying Driver Policies," on page 56.
- **Configure password synchronization:** The basic driver configuration is set up to support password synchronization through Universal Password. If you don't want this setup, see "Configuring Password Flow " in the *Identity Manager 3.6 Password Management Guide*.

After completing the configuration tasks, continue with the next section, Deploying the Driver.

4.2.3 Deploying the Driver

After a driver is created in Designer, it must be deployed into the Identity Vault.

1. In Designer, open your project.
2. In the Modeler, right-click the driver icon  or the driver line, then select *Live* > *Deploy*.
3. If you are authenticated to the Identity Vault, skip to Step 5, otherwise, specify the following information:
   - **Host:** Specify the IP address or DNS name of the server hosting the Identity Vault.
   - **Username:** Specify the DN of the user object used to authenticate to the Identity Vault.
   - **Password:** Specify the user's password.
4. Click *OK*.
5. Read through the deployment summary, then click *Deploy*.
6. Read the successful message, then click *OK*.
7. Click *Define Security Equivalence* to assign rights to the driver.

   The driver requires rights to objects within the Identity Vault. The Admin user object is most often used to supply these rights. However, you might want to create a DriversUser (for example) and assign security equivalence to that user. Whatever rights that the driver needs to have on the server, the DriversUser object must have the same security rights.

   7a Click *Add*, then browse to and select the object with the correct rights.

   7b Click *OK* twice.

8 Click *Exclude Administrative Roles* to exclude users that should not be synchronized.

You should exclude any administrative User objects (for example, Admin and DriversUser) from synchronization.

8a Click *Add*, then browse to and select the user object you want to exclude.

8b Click *OK*.

8c Repeat Step 8a and Step 8b for each object you want to exclude.

8d Click *OK*.

9 Click *OK*.

4.2.4 Starting the Driver

When a driver is created, it is stopped by default. To make the driver work, you must start the driver and cause events to occur. Identity Manager is an event-driven system, so after the driver is started, it won't do anything until an event occurs.

To start the driver:

1 If you are using the Remote Loader with the driver, make sure the Remote Loader driver instance is running. For instructions, see "Starting the Remote Loader" in the *Identity Manager 3.6 Remote Loader Guide*.

2 In Designer, open your project.

3 In the Modeler, right-click the driver icon or the driver line, then select *Live* > *Start Driver*.

4 Continue with Section 4.4, "Activating the Driver," on page 48.

4.3 Creating the Driver in iManager

You create the PeopleSoft driver by importing the driver's basic configuration file and then modifying the configuration to suit your environment. After you've created and configured the driver, you need to start it.

4.3.1 Importing the Driver Configuration File

1 In iManager, click to display the Identity Manager Administration page.

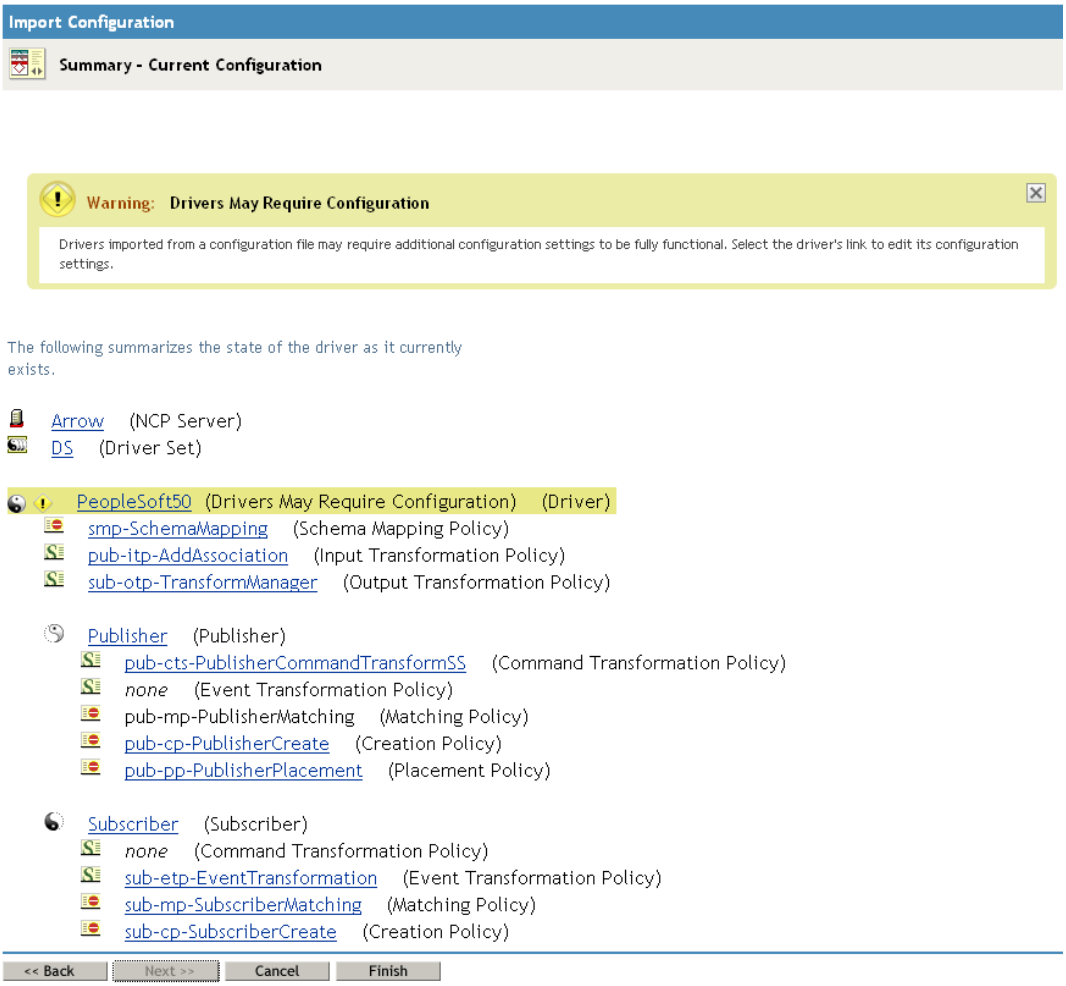
2 In the Administration list, click *Import Configuration* to launch the Import Configuration Wizard.

3 Follow the wizard prompts, filling in the requested information (described below) until you reach the Summary page.

Prompt	Description
Where do you want to place the new driver?	You can add the driver to an existing driver set, or you can create a new driver set and add the driver to the new set. If you choose to create a new driver set, you are prompted to specify the name, context, and server for the driver set.
Import a configuration into this driver set	Use the default option, *Import a configuration from the server (.XML file)*. In the *Show* field, select *Identity Manager 3.6 configurations*. In the *Configurations* field, select the PeopleSoft50 file.
Driver name	Type a name for the driver. The name must be unique within the driver set.
Active Users Container	Specify the name of the Organizational Unit container in the Identity Vault where Active users from PeopleSoft are placed. You can modify this parameter through a global configuration value (GCV) after installation.
Inactive Users Container	Specify the name of the Organizational Unit container in the Identity Vault where Inactive users from PeopleSoft are placed. You can modify this parameter through a GCV after installation.
Active Employees Group	Specify the name of the Group object in the Identity Vault to which Active Employee users from PeopleSoft are added. You can modify this parameter through a GCV after installation.
Active Managers Group	Specify the name of the Group object in the Identity Vault to which Active Manager users from PeopleSoft are added. You can modify this parameter through a GCV after installation.
Driver is Local/Remote	Select *Local* if this driver will run on the Metadirectory server without using the Remote Loader service. Select *Remote* if you want the driver to use the Remote Loader service, either locally on the Metadirectory server or remotely on another server.
PeopleSoft User ID	Specify an administrative account through which the driver can authenticate to PeopleSoft.
Connection to PeopleSoft Server	Specify the hostname or IP address and port number for connecting to the appropriate PeopleSoft application server. This is typically referred to as the PeopleSoft application server connection string. The default port is 9000. The connection string uses the following format: `<hostname or IP address>:<Jolt Port Number>` Example: `//PSServer:9000` To enable failover and load balancing, you can supply multiple server connection strings separated by a comma. Example: `//PSServer:9000,//111.222.3.4:9000`
Remote Host Name and Port	This applies only if the driver is running remotely. Specify the host name or IP address of the server where the driver's Remote Loader service is running.

Prompt	Description
Driver Password	This applies only if the driver is running remotely. Specify the driver object password that is defined in the Remote Loader service. The Remote Loader requires this password to authenticate to the Metadirectory server.
Remote Password	This applies only if the driver is running remotely. Specify the Remote Loader's password (as defined on the Remote Loader service). The Metadirectory engine (or Remote Loader shim) requires this password to authenticate to the Remote Loader
Define Security Equivalences	The driver requires rights to objects within the Identity Vault. The Admin user object is most often used to supply these rights. However, you might want to create a DriversUser (for example) and assign security equivalence to that user. Whatever rights that the driver needs to have on the server, the DriversUser object must have the same security rights.
Exclude Administrative Roles	You should exclude any administrative User objects (for example, Admin and DriversUser) from synchronization.

When you finish providing the information required by the wizard, a Summary page, similar to the following is displayed.



At this point, the driver is created from the basic configuration file. To ensure that the driver works the way you want it to for your environment, you must review and modify the driver's default configuration settings.

4 To modify the default configuration settings, click the linked driver name, then continue with the next section, Configuring the Driver.

or

To skip the configuration settings at this time, click *Finish*. When you are ready to configure the settings, continue with Configuring the Driver.

4.3.2 Configuring the Driver

After importing the driver configuration file, you need to configure the driver before it can run. You should complete the following tasks to configure the driver:

- **Ensure that the driver can authenticate to PeopleSoft:** Make sure that you've established a PeopleSoft administrative account for the driver (see Section 4.1, "Creating a PeopleSoft Account," on page 41) and that the correct authentication information, including the User ID and password, is defined for the driver parameters (see Section A.1.3, "Authentication," on page 78).

- **Configure the driver parameters:** There are many settings that can help you customize and optimize the driver. The settings are divided into categories such as Driver Configuration, Engine Control Values, and Global Configuration Values (GCVs). Although it is important for you to understand all of the settings, your first priority should be to configure the driver parameters located on the Driver Configuration page. For information about the driver parameters, see Section A.1.5, "Driver Parameters," on page 80.
- **Configure the driver policies and filter:** Modify the driver policies and filter to implement your business policies. For instructions, see Section 6.3, "Modifying Driver Policies," on page 56.
- **Configure password synchronization:** The basic driver configuration is set up to support password synchronization through Universal Password. If you don't want this setup, see "Configuring Password Flow " in the *Identity Manager 3.6 Password Management Guide*.

After completing the configuration tasks, continue with the next section, Starting the Driver.

4.3.3 Starting the Driver

When a driver is created, it is stopped by default. To make the driver work, you must start the driver and cause events to occur. Identity Manager is an event-driven system, so after the driver is started, it won't do anything until an event occurs.

To start the driver:

1. If you are using the Remote Loader with the driver, make sure the Remote Loader driver instance is running. For instructions, see "Starting the Remote Loader" in the *Identity Manager 3.6 Remote Loader Guide*.
2. In iManager, click  to display the Identity Manager Administration page.
3. Click *Identity Manager Overview*.
4. Browse to and select the driver set object that contains the driver you want to start.
5. Click the driver set name to access the Driver Set Overview page.
6. Continue with Section 4.4, "Activating the Driver," on page 48.

4.4 Activating the Driver

If you created the driver in a driver set where you've already activated the Metadirectory engine and service drivers, the driver inherits the activation. If you created the driver in a driver set that has not been activated, you must activate the driver within 90 days. Otherwise, the driver stops working.

For information on activation, refer to "Activating Novell Identity Manager Products" in the *Identity Manager 3.6 Installation Guide*.

Upgrading an Existing Driver

5

The following sections provide information to help you upgrade an existing driver to version 3.6:

5.1 Supported Upgrade Paths

You can upgrade from any 3.*x* version of the PeopleSoft driver. Upgrading a pre-3.*x* version of the driver directly to version 3.6 is not supported.

5.2 What’s New in Version 3.6

Version 3.6 of the driver does not include any new features that are not already in the 3.5.1 version. However, the process for creating a new driver has changed. For detailed information, see Chapter 4, “Creating a New Driver,” on page 41.

5.3 Upgrade Procedure

The process for upgrading the PeopleSoft driver is the same as for other Identity Manager drivers. For detailed instructions, see “Upgrading” in the *Identity Manager 3.6 Installation Guide*.

Customizing the Driver

6

This section covers how you can customize the driver by triggering transactions through the PSA via PeopleCode.

6.1 Customizing the PSA by Triggering Transactions

Transaction record creation is triggered by PeopleCode associated with modifications and additions to data on the Schema Data record (DIRXML_S_PERS) and row Delete events on the Staging record (DIRXML_SCHEMA01). If desired, the PeopleSoft administrator or consultant can use these examples to trigger transactions based on other events within the PeopleSoft application.

NOTE: The D Event Type operation has been redefined for the 5.0*x* PeopleSoft Service Agent (PSA). The D Event Type is now generated for application object Delete events instead of object Disable events. All Modification events now generate M (Modify) transaction events.

The default Transaction creation function is defined on the DIRXML_DRIVER field of the DIRXML_DERIVED Record definition:

A PeopleCode function call would be as follows:

```
DirXML_Trans( Transaction Table Name,
    Transaction Channel Type,
          Schema CI Name,
          Event Type,
          Schema Record Key Value,
          Transaction Date and Time,
          Transaction Miscellaneous Info,
          Collection Row Delete Field Name,
          Collection Row Delete Field Key Value,
          Transaction Status);
```

An example of sample Modification event transaction from the PSA looks like this:

```
DirXML_Trans(“DIRXML_TRANS01”,
      &channel,
        “DIRXML_SCHEMA01”,
        “A”,
        DIRXML_S_PERS.DIRXML_S_ID,
        %DateTime,
        &tValue,
        “”,
        “”,
        “A”);
```

Table 6-1 *Function Call Parameter Definitions*

Parameter	Description	Default Value
Transaction Table	The name of the table where transactions are written. This table is built within PeopleTools and the field elements should be consistent with the delivered DIRXML_TRANS01 table.	DIRXML_TRANS01
Transaction Channel Type	The name used to identify the driver that processes the transactions and the channel that created the transactions.	The NPSDriver1S transaction was caused by a Subscriber channel event and is processed by driver NPSDriver1. The NPSDriver1P transaction was caused by a Publisher channel event and is processed by driver NPSDriver1.
Schema CI Name	The name of the Schema CI object that the transaction type is connected to. The driver uses the name of this object to query for the data connected to the transaction type.	DIRXML_SCHEMA01
Event Type	The type of XML event that is written to the transaction table. This can be 1 of 4 values as listed.	A=ADD M=MODIFY D=DELETE R=ROW DELETE
Schema Record Key Value	The identifier that is used to associate a particular record within PeopleSoft to an eDirectory™ object. It could be the EMPLID value for employees, STUDENTID value for students, DEPTID for departments, ACCTID for account codes, and so forth. Key elements must be identified for the Transaction Schema.	ASSOC_ID
Transaction Date Time	The date/time element used to determine when the transaction is processed.	%Datetime
Transaction Miscellaneous Info	The parameter contains 1...n values that the developer wants to pass to the driver during processing. This value might not be available via the Schema object when a transaction is processed by the driver.	ASSOC_ID | "|" | LAST_NAME
Collection Row Delete Field Name	The field name of the scroll level attribute in the application record.	DIRXML_S_PHONES
Collection Row Delete Field Key	The key field value of the deleted data row (CELL, PGR, BUSN).	

Parameter	Description	Default Value
Transaction Status	The initial processing status of the transaction. Generally this value is set to A for Available. For Subscriber delete events, it is not desirable for the driver to process the transaction event. Therefore, delete event transactions generated by the default PSA are assigned a status of S for Success.	

6.2 Changing the Data Schema Component Interface

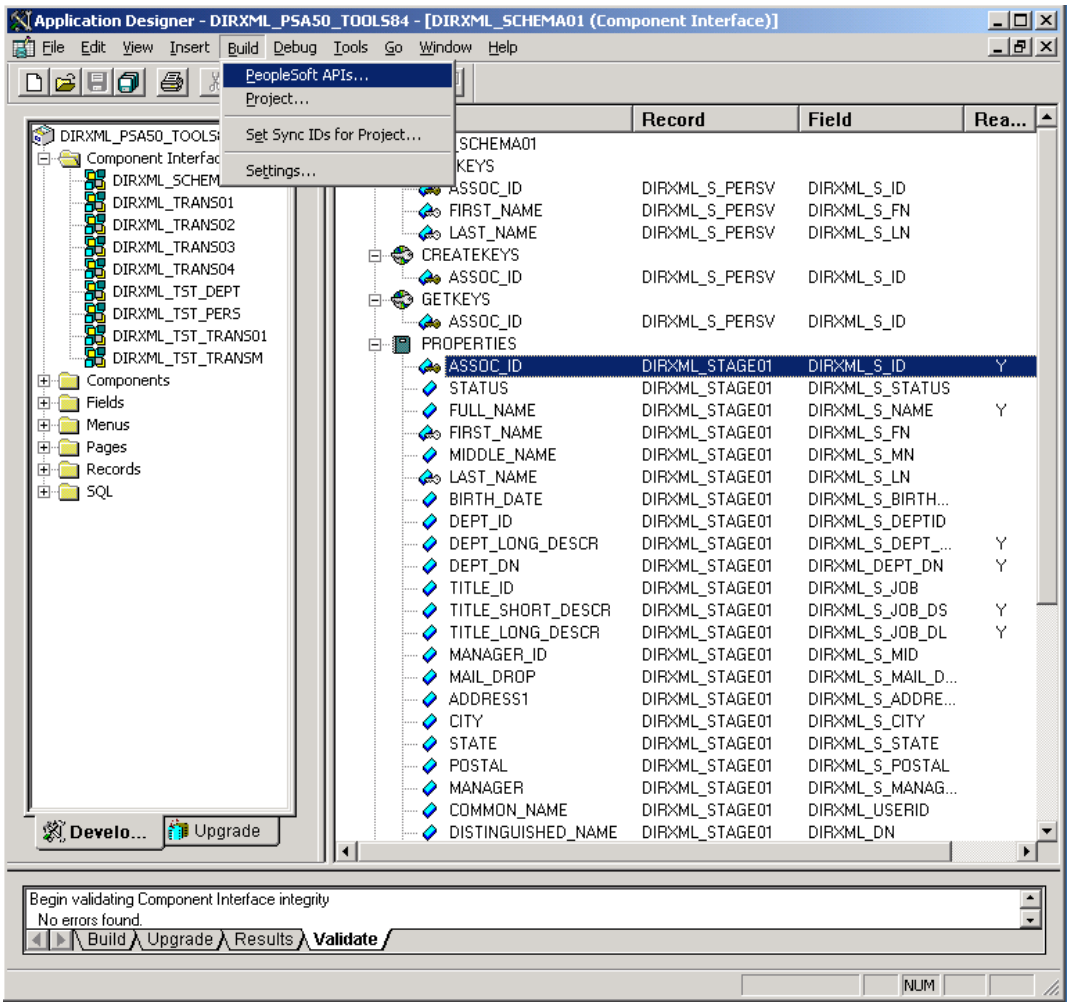
The driver is preconfigured to use the Component Interfaces (CIs) defined in the PSA. The APIs for these CIs have been compiled and combined into the `dirxmlcomps.jar` file. At run time, the driver imports the interfaces required to interact with the appropriate CI.

If the driver is configured to use different data schema CIs, the Java APIs for these CIs also need to be built, compiled, and archived into a `.jar` file. The directions for building the CI APIs is documented in the *PeopleBooks* > *PeopleSoft Component Interfaces* > *Programming Component Interfaces in Java* > *Building APIs in Java* section of the PeopleTools documentation. (It is not necessary to rebuild all of the Java CI APIs, only those that are associated with your new schema CI.) The compiled and archived `.jar` file should be placed in the DirXML `/lib` subdirectory with the `psoftshim.jar` file.

- Section 6.2.1, “Building the PeopleSoft Java Component Interface API,” on page 53
- Section 6.2.2, “Compiling the Java CI API,” on page 55
- Section 6.2.3, “Building the CI API JAR File,” on page 55

6.2.1 Building the PeopleSoft Java Component Interface API

1. From the PeopleTools Application Designer, select the Schema CI you want to build.
2. Click *Build* > *PeopleSoft APIs*. In this example, the *DirXML_SCHEMA01 CI* is used.



3 From the Build PeopleSoft API Bindings dialog box, select the *build* option for the Java classes.

4 Select a target directory for the Java CI APIs (For our example, `C:\newci`).

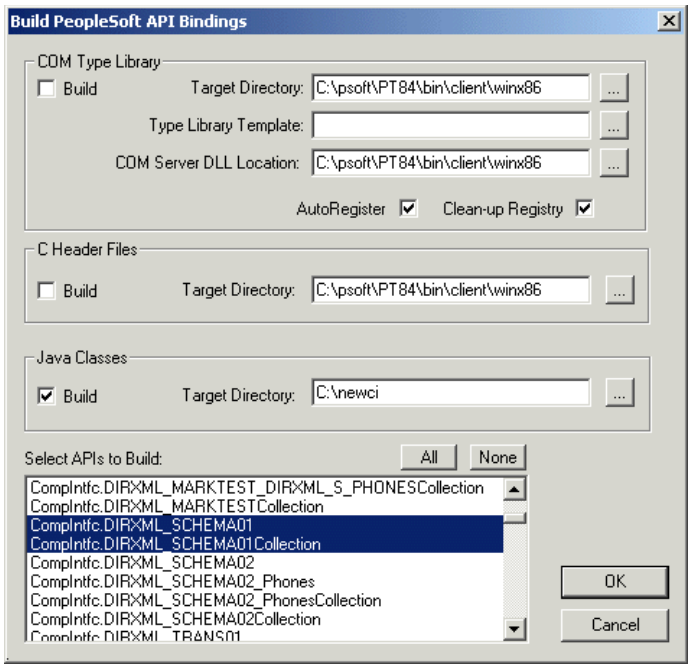
5 For the *Select APIs to Build* prompt, click *None* to deselect all CIs.

6 Using the scroll area, select the CIs for which you wish to generate an API. Make sure you select all CIs that begin with the name of the desired interface. In this example we selected CompIntfc.DIRXML_SCHEMA01 and CompIntfc.DIRXML_SCHEMA01Collection.

IMPORTANT: In addition to your schema CIs, you must always select the CompIntfc.CompIntfcPropertyInfo and CompIntfc.CompIntfcPropertyInfoCollection APIs. They are required in order to compile the schema APIs.

---

7 Click *OK* to generate the CI APIs. You might be prompted to create the target directory you specified.



The Application Designer status window should show a `Generating API Wrappers` message with the selected CIs, and then a `Done` message.

6.2.2 Compiling the Java CI API

After the APIs are generated, they must be compiled. The files are in `C:\newci\PeopleSoft\Generated\CompIntfc`. For our example, there should be eight Java files present in this directory, including the four selected CI API files and four associated Java Interface files. Change directories to the file location and compile the Java files, using an appropriate JVM (suggested version 1.3.1 and higher) with a classpath argument specifying the PeopleTools `psjoa.jar` file.

An example command line is:

```
javac -classpath c:\psoft\pt84\class\psjoa.jar *.java
```

After a successful compile, there is a `.class` file for each `.java` file in the directory.

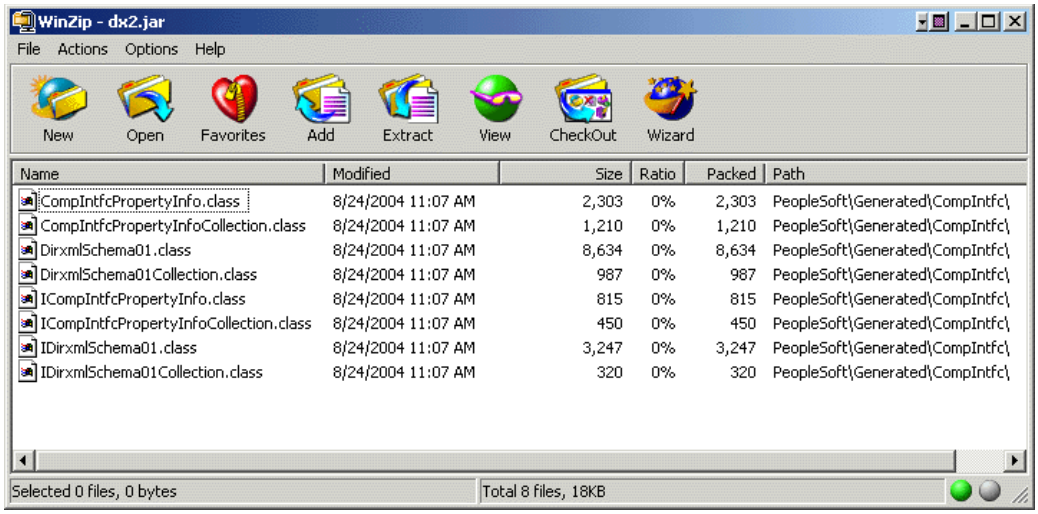
6.2.3 Building the CI API JAR File

The final step of the build process is the generation of a `.jar` file containing the compiled `.class` files. It is important that the full class path be generated with the JAR file, so the process must begin at the root of the CI API directory, `C:\newci`. From this directory, use the following command line:

```
jar cvf0M newci.jar PeopleSoft/Generated/CompIntfc/*.class
```

This command builds a JAR file called `newci.jar` that is comprised of the previously compiled `.class` files and contains the full class paths. The contents of the file can be verified by using WinZIP or another appropriate tool.

Figure 6-1 *Building a New .jar File*



If the driver is currently running, it must be stopped. If you are using the Remote Loader, the driver must be shut down. If you are using a local driver, it is also necessary to shut down eDirectory.

Copy the new JAR file to the same lib location with the driver components `psoftshim.jar`, `dirxmlcomps.jar`, and `psjoa.jar`. Restart eDirectory (if required) and the driver and Remote Loader.

6.3 Modifying Driver Policies

The following sections contains information to help you understand how the driver's policies are implemented, as well as information about how you can modify these objects:

6.3.1 Modifying the Driver Mapping Policy

The Mapping policy is a Novell® eDirectory object that defines the relationship between data fields defined in the PeopleSoft application and eDirectory attributes. The Mapping policy is located in the driver object container and is used by both the Publisher and Subscriber channels of the driver.

A preconfigured default Mapping policy is delivered with the driver product. The mappings defined in the Mapping policy are designed in coordination with the preconfigured PeopleSoft application Component Interface (CI) that is also delivered with the driver product.

The default Data Schema CI is called DIRXML_SCHEMA01 and represents a set of employee data. The data fields in this CI are mapped to similar attributes of the eDirectory User object.

The following is a short sample of the delivered Mapping policy in `.xml` format. The nds-name represents the name of the class or attribute in eDirectory and the app-name represents the class or field name of the PeopleSoft CI.

```
<?xml version="1.0" encoding="UTF-8"?> <attr-name-map>
     <class-name>
          <nds-name>User</nds-name>
          <app-name>DIRXML_SCHEMA01</app-name>
     </class-name>      <attr-name classname="User">
          <nds-name>Given Name</nds-name>
          <app-name>FIRST_NAME</app-name>
     </attr-name>      ... </attr-name-map >
```

When you modify or create a Mapping policy, verify that the PeopleSoft field names appear identically (spelling and capitalization) in the Mapping policy and PeopleSoft CI definition. If you use the Identity Manager Mapping policy editor, correct mapping behavior is ensured. It is also important to note that if new attributes are included or removed from the Mapping policy, the new attribute set should be reflected in the respective Publisher and Subscriber filters. If mapped attributes are not included in the filters, they cannot be synchronized.

6.3.2 Using the Schema Query to Refresh the PeopleSoft Schema Component Interface

By default, the schema that the driver reads from PeopleSoft consists of the data fields defined on the DIRXML_SCHEMA01 CI. If the data elements are modified, the CI is renamed, or additional Data Schema CIs are added to the driver configuration, it is necessary to refresh the PeopleSoft application schema definition and re-map affected attributes.

6.3.3 Publisher Channel Objects

The Publisher object contains a filter and a set of policies. Policies are necessary for converting data from the PeopleSoft CI into XDS format. The driver then submits the data to the Metadirectory engine. The engine applies the Publisher filter to the data and applies the business logic defined by the Publisher policies prior to submitting the data to Identity Vault.

6.3.4 Understanding the Publisher Filter

The Publisher filter is a logical component of the driver filter object. The filter specifies the object classes and accompanying attributes that are passed from the PeopleSoft CI to eDirectory. The filter is defined by using eDirectory attribute naming, so it is applied after schema mapping takes place.

For example, if the User class is specified in the Publisher filter with only the Surname and Given Name attributes, the Metadirectory engine allows changes to only these attributes to be passed to the Publisher policies from the PeopleSoft Driver. If a Telephone Number attribute is modified, the Publisher filter removes this data from the event document because the Telephone Number attribute is not in the filter. Other attribute values can be created by the Publisher policies according to the integration scenario.

You can configure the Publisher filter to include attributes required by your environment and allowable by eDirectory access controls. Configure it to include the following:

- Object classes you want to synchronize.
- Attributes on those objects you want to synchronize.
- Attributes that are required by your Publisher policies. These attributes are not synchronized, but are required to perform some defined business logic that is to be applied to the synchronized data. These are known as “notify” attributes.

The Publisher filter specifies a set of data to be considered as authoritative from the PeopleSoft database. Based on business logic, there might be multiple authoritative sources of specific data (such as another driver or eDirectory itself). The configuration of the filters in your environment determines data ownership and authority.

6.3.5 Publisher Filter Attributes

By default, PeopleSoft applications are considered to be highly authoritative. Therefore, most of the field names in the default DIRXML_SCHEMA01 Component Interface are passed through the Publisher filter. As noted earlier, the names of attributes in the filter are eDirectory attribute names that have been mapped via the Mapping policies.

The Publisher channel attributes listed below are included in the default filter:

Table 6-2 *Publisher Channel Attributes*

departmentNumber	OU
employeeStatus	pager
Full Name	Physical Delivery Office Name
Given Name	Postal Code
homePhone	S
Initials	SA
isManager	Surname
jobCode	Telephone Number
mailstop	Title
managerWorkforceID	workforceID
mobile	

Most of the attributes in the Driver Filter are configured for bidirectional synchronization. This is done for sample purposes to allow the driver to perform add, modify, and delete operations on both the Subscriber and Publisher channels. In most installations the driver policies and filter are configured to function in either a predominant Subscriber or Publisher mode.

6.3.6 Securing the Data

PeopleSoft applications, as with many other applications, contain sensitive data that must be highly secured by organizations. There are two ways to ensure that secure data is not published from the PeopleSoft application:

- Remove it from the synchronized data CI definition.
- Remove it from the Publisher filter.

The first method guarantees that the data does not leave the PeopleSoft application. The second method guarantees that the data is not be synchronized to Identity Vault via the driver.

6.3.7 Publisher Object Policies

Policies contain rules or templates that perform specific operations that can manipulate data, query either Identity Vault or PeopleSoft for additional data required for processing, create new attributes based on values of other attributes, or even discard entire data events. The following sections explain each policy and describe the operations of each policy or template. Because XML, DirXML® Script, and XSLT allow for great flexibility, all policies can be modified to meet the individual needs of your organization. The Mapping policy has been previously described and is addressed in this section under "Modifying the Driver Mapping Policy" on page 57.

The Publisher channel object, by default, contains or uses the following policies:

- Section , "Input Transformation Policy," on page 59
- Section , "Matching Policy," on page 60
- Section , "Create Policy," on page 60
- Section , "Placement Policy," on page 60
- Section , "Command Transformation Policy," on page 61

Input Transformation Policy

The Input Transformation policy is implemented as a default policy for the PeopleSoft Driver. Although the Input Transformation policy is not exclusively used by the Publisher channel, it performs a publishing role because it is used to transform the data format of any XDS document received from the PeopleSoft Driver shim, regardless of which channel generated the submission of the document. That is, the Subscriber channel can issue object queries to the driver shim. All data returned in the response is processed through the Input Transformation policy. An example of data transformation contained in this policy is the transformation of character attributes in PeopleSoft to Boolean attributes in eDirectory.

Manager Flag Data Transformation Template

This template, contained in the Input Transformation policy, converts the Y or N data values in the PeopleSoft Manager attribute into True or False Boolean values to reflect the data format of the Identity Vault's Manager attribute.

Matching Policy

The Matching policy is used by the Metadirectory engine to apply criteria to determine if a matching data object already exists in Identity Vault. The Matching policy is applied to all Add documents received from the PeopleSoft Driver shim. If a match is found by this policy, the Add event is automatically converted to a Modify event by the Metadirectory engine. If a matched object in eDirectory is not currently associated with the PeopleSoft application, an association is created.

The Matching policy should provide criteria that are guaranteed to produce 0 or 1 match. More than one policy can exist, and the Metadirectory engine applies them in the order that they are defined. Any policy producing 0 or more than one match is skipped and the next policy is applied. Processing finishes when one match is found or after the last policy has been processed.

The default Publisher Matching policy is a DirXML Script policy that attempts to match eDirectory User objects containing the same value in the workforceID attribute (mapped from the DIRXML_SCHEMA01 attribute). A secondary policy attempts to match using the Surname and Given Name attribute.

Create Policy

The Create policy is used to specify the criteria for creating a new object after the Matching policy has failed to find a match. This policy performs various tests and transformations based on the requirements for object creation in eDirectory and the business logic being applied.

The default PeopleSoft Create policy is an XML policy that asserts that the <add> document is for a User object and that it must contain a Surname and Given Name attribute. The Surname attribute is mandatory in eDirectory, and the business logic used for object naming requires the existence of the Given Name attribute. If this criteria is met, a secondary XSLT style sheet policy is called to create the eDirectory Name attribute. A final policy is provided to append a default password to new User objects.

Placement Policy

The Placement policy defines where an object is placed in the eDirectory tree when the object is created. This placement can be determined based on the presence (or absence) of attributes, particular values of attributes, etc. Placement can also be determined by the Create policy and passed to the Placement policy.

In a typical PeopleSoft HR environment, an employee is hired within PeopleSoft, a notification is sent to the IS department, and an IS administrator determines the location of the new User object in the eDirectory tree. Before defining location policies in the Placement policy, analyze your organization's current business process.

The default PeopleSoft Placement policy is a DirXML Script policy that handles placement of User objects based on the employeeStatus attribute. It uses the following order of processing: If the employeeStatus value is A, the employee is placed in the organizational unit specified by the Active Users Container GCV value. If the employeeStatus is I, the employee’s User object is placed in the organizational unit as specified by the Inactive Users Container GCV value. If the employeeStatus attribute is not present, the employee’s User object is placed in the organizational unit as specified by the Inactive Users Container GCV value.

Command Transformation Policy

The Command Transformation policy is the final transformation policy to be processed prior to submission of a Publisher document to Identity Vault. To demonstrate this functionality, the default PeopleSoft Driver configuration implements an unusual example of business logic that demonstrates the flexibility and power of Identity Manager.

The business logic scenario is a requirement to maintain the object CN attribute and full distinguished name DN of each User in the PeopleSoft application. The CN attribute is generated on new objects when they are created in eDirectory. The DN is not a true attribute, but a concatenation of the directory path and CN of a User. The DN changes based on the employeeStatus attribute of an object, so it is set on User Add events and Delete events that are transformed into Move events.

Because this data is known during the processing of the Command Transformation policy, the CN and DN data is placed into an `<operation-data>` element appended to the document, causing the object to be added or moved. After Identity Manager applies the data to Identity Vault, a status document is returned. The Output Transformation policy (documented in the Subscriber channel) monitors status documents that are returned and transforms successfully processed documents with attached `<operation-data>` elements into modification documents that are applied to the PeopleSoft application. This is known as *write-back functionality*.

As the final transformation policy, the Command Transformation policy provides an excellent location to define operations that must be applied without the risk of further event transformation, thus allowing complicated policy processing to be programmed in one location. the bulk of the business logic transformations in the Publisher channel are implemented in this policy.

The following templates exist in the default Command Transformation policy. In addition to the listed templates, all Identity Manager policies contain identity-transform templates that allow the copying of XML attributes and elements that are passed through unmodified. The default configuration only handles documents related to User objects.

- “match <add> element” on page 62
- “match <modify> element” on page 62
- “get-empl-status” on page 63
- “get-empl-isManager” on page 63
- “get-empl-CN” on page 63
- “get-empl-managerWorkforceID” on page 63
- “get-empl-ID” on page 63
- “set-manager-on-user” on page 63
- “set-manager-on-direct-reports” on page 63
- “clear-manager-on-direct-reports” on page 63
- “set-directReports-on-manager” on page 63
- “clear-directReports-on-manager” on page 63
- “set-directReports-on-user” on page 64

match <add> element

This template does the following:

- Tries to find the User's manager. If the manager is found and the manager's employeeStatus attribute is set to A, the template sets the manager and managerWorkforceID attribute on the User.
- Sets the Login Disabled attribute based on employeeStatus. If the status is A, Login Disabled is set to False. If the status is I, then Login Disabled is set to True.
- Adds or removes the Group Membership value based on the employeeStatus attribute value. All active employees with the isManager attribute set to False are placed into an Employee Group. All active employees with the isManager attribute set to True are placed into a Manager Group. The Group Membership attribute and associated links on the group objects are cleared if the employeeStatus is set to I.
- Determines placement of an object based on the employeeStatus attribute value. Active User objects are placed in an Active organizational unit. Inactive User objects are placed in an InActive organizational unit.
- Adds the manager attribute to any other active User objects in the directory whose managerWorkforceID attribute specifies this new User.
- Adds the directReports attribute value to any active User object in the directory that is specified by this User's managerWorkforceID attribute.
- Generates a write-back `<operation-data>` element to facilitate the addition of the CN and DN attributes in PeopleSoft.

match <modify> element

This template does the following:

- Tries to find the User's manager. If the manager is found and the manager's employeeStatus attribute is set to A, the template sets the manager and managerWorkforceID attribute on the User.
- Sets the Login Disabled attribute based on employeeStatus. If the status is A, Login Disabled is set to False. If the status is I, then Login Disabled is set to True.
- Adds or removes the Group Membership value based on the employeeStatus attribute value. All active employees with the isManager attribute set to False are placed into an Employee Group. All active employees with the isManager attribute set to True are placed into a Manager Group. The Group Membership attribute and associated links on the group objects are cleared if the employeeStatus is set to I.
- Adds the manager attribute to any other active User objects in the directory whose managerWorkforceID attribute specifies this new User.
- Adds the directReports attribute value to any active User object in the directory that is specified by this User's managerWorkforceID attribute.
- If the employeeStatus is changing from I to A, generates a Move event with a write-back event-id to facilitate the modification of the DN attribute in PeopleSoft. This event moves the User object from the InActive organizational unit to the Active organizational unit.

get-empl-status

This template requests the value of the employeeStatus attribute from a specified User object in eDirectory.

get-empl-isManager

This template requests the value of the isManager attribute from a specified User object in eDirectory.

get-empl-CN

This template requests the value of the CN attribute from a specified User object in eDirectory.

get-empl-managerWorkforceID

This template requests the value of the managerWorkforceID attribute from a specified User object in eDirectory.

get-empl-ID

This template requests the value of the Identity Manager WorkforceID attribute from a specified User object in eDirectory.

set-manager-on-user

This template queries Identity Vault to determine if the passed-in managerWorkforceID parameter references an active User object in eDirectory. The name of the manager-User object is set in the User's manager attribute if the manager is active.

set-manager-on-direct-reports

This template receives a manager-User object ID parameter. A query is sent to Identity Vault for a list of all active Users who have the specified manager-User object ID in the managerWorkforceID attribute. The manager attribute of all Users in the list is set with the name of the manager-User.

clear-manager-on-direct-reports

This template receives a manager-User object ID parameter. A query is sent to Identity Vault for a list of all active Users who have the specified manager-User object ID in the managerWorkforceID attribute. The manager attribute of all Users in the list is removed.

set-directReports-on-manager

This template receives a manager-User object ID parameter. A query is sent to Identity Vault to find an active User who has the specified manager-User object ID in the workforceID attribute. The directReports attribute of the manager-User object is modified to include the DN of the User object specified in the source document.

clear-directReports-on-manager

This template receives a manager-User object ID parameter. A query is sent to Identity Vault to find an active User who has the specified manager-User object ID in the workforceID attribute. The directReports attribute of the manager-User object is modified to remove the DN of the User object specified in the source document.

set-directReports-on-user

This template receives a User object ID parameter. A query is sent to Identity Vault to find a list of active Users who have the specified User object ID in the managerWorkforceID attribute. The directReports attribute of the User object is modified to include the DN of all User objects in the list.

6.3.8 Subscriber Channel Objects

The Subscriber object contains a filter and a set of policies. These policies are necessary for converting data from Identity Vault to the PeopleSoft Driver.

The Identity Vault sends filtered data modification events to PeopleSoft through the Metadirectory engine. The engine applies the business logic defined by the Subscriber policies prior to submitting the data to the PeopleSoft Driver, which converts the data from the Identity Manager XDS format into PeopleSoft Component Interface format. The PeopleSoft Driver then submits the data to the PeopleSoft application and updates the staging table.

6.3.9 Understanding the Subscriber Filter

The Subscriber filter is a logical component of the driver filter object. The filter specifies the object classes and accompanying attributes that are passed from eDirectory to the PeopleSoft Component Interface. The filter is defined using eDirectory attribute naming, so it is applied before schema mapping takes place.

For example, if the User class is specified in the Subscriber filter with only the mobile and pager attributes, the filter allows changes to only these attributes to be passed to the Metadirectory engine. If a Telephone Number attribute is modified, the Subscriber filter removes this data because the Telephone Number attribute is not in the filter. Other attribute values can be created by the Subscriber policies according to the integration scenario.

You can configure the Subscriber filter to include attributes required by your environment and allowable by eDirectory access controls. Configure it to include the following:

- Object classes you want to synchronize.
- Attributes on those objects you want to synchronize.
- Attributes that are required by your Subscriber policies. These attributes cannot be synchronized, but are required to perform some defined business logic that is to be applied to the synchronized data. These are known as “notify” attributes.

The Subscriber filter specifies a set of data to be considered as authoritative from Identity Vault or any other authoritative application that might have written the data to Identity Vault. Based on business logic, there might be multiple authoritative sources of specific data (such as another driver or eDirectory itself). The configuration of the filters in your environment determines data ownership and authority.

Subscriber Filter Attributes

Most of the attributes in the Subscriber filter are configured for bidirectional synchronization. This is done for sample purposes to allow the driver to perform add, modify, and delete operations on both the Subscriber and Publisher channels. In most installations the driver policies and filter are configured to function in either a predominant Subscriber or Publisher mode.

The attributes listed below are included in the default Subscriber filter.

Table 6-3 *The Subscriber Channel Attributes*

CN	managerWorkforceID
departmentNumber	mobile
Description	pager
employeeStatus	Physical Delivery Office Name
Given Name	Postal Code
homePhone	S
Initials	SA
Internet EMail Address	Surname
jobCode	Telephone Number
mailstop	workforceID

As mentioned previously, the Subscriber synchronization attributes in the Driver filter are present for demonstration purposes. It is very important to note that Subscriber filter and policies should be set to match the requirements of the PeopleSoft CI being utilized. If you want the ability to Add objects on the Subscriber channel, make sure all required attributes in the CI are allowed to pass through the filter. Also make note of which fields in the CI are related display fields or translate values that cannot be synchronized, such as, Title, OU, and Full Name. By implementing and enforcing the CI application restrictions in your filter and policies, you encounter fewer synchronization errors and achieve higher throughput.

6.3.10 Securing the Data

If there is sensitive data that should not be shared with the PeopleSoft application, it should be removed from the Subscriber filter.

6.3.11 Modifying the Filter

A properly configured Subscriber filter promotes a secure environment and secures data sharing from Identity Vault to PeopleSoft. Make sure that attributes that are required for Subscriber policies processing (such as workforceID) are present in the filter even if they won't be synchronized to the PeopleSoft application.

6.3.12 Subscriber Object Policies

Policies contain templates that perform specific operations that can manipulate data, query either Identity Vault or PeopleSoft for additional data required for processing, create new attributes based on values of other attributes, or even discard entire data events. The following section explains each policy and provides a description of the operations each policy performs. Because XML, DirXML Script, and XSLT allow for great flexibility, all policies can be modified to meet the individual needs of your organization. The Schema Mapping policy has been previously described and is not be addressed in this section. For information on the Schema Mapping policy, refer to “Modifying the Driver Mapping Policy” on page 57.

The Subscriber object, by default, contains or uses the following policies:

- “Event Transformation Policy” on page 66
- “Matching Policy” on page 66
- “Create Policy” on page 67
- “Output Transformation Policy” on page 67

Event Transformation Policy

The Event Transformation Policy is used to remove or change received event types into different events. The following templates exist in the default Event Transformation Policy:

- “Match <rename> Element” on page 66
- “Match <move> Element” on page 66
- “Match <delete> Element” on page 66

Match <rename> Element

PeopleSoft does not allow the rename or modification of primary key values. This policy transforms a User Rename event into a Modify event that resets the CN and DISTINGUISHED_NAME fields in the CI.

Match <move> Element

PeopleSoft does not support object containment or hierarchy. This policy transforms User Move events into Modify events that reset the DISTINGUISHED_NAME field in the CI.

Match <delete> Element

This template is commented out by default to allow delete events to be passed to the driver. This policy remains for reference for non-delete scenarios. Previous versions of the driver did not support Delete events, so this template transforms them into Modify events that remove only Subscriber channel fields in the CI (CN, DISTINGUISHED_NAME, Internet Email Address, and Description).

Matching Policy

The Matching policy is used by the Metadirectory engine to apply criteria to determine if a matching data object already exists in the PeopleSoft application. The Matching policy is applied to all documents received from Identity Vault that contain User objects that are not currently associated with PeopleSoft objects. If a match is found by this policy, the Add event is converted into an object merge operation. This merge queries PeopleSoft for all attributes in the Publisher filter and applies them to Identity Vault, and queries Identity Vault for all attributes in the Subscriber filter and applies them to the PeopleSoft application. The process is finalized when an association value is written on the eDirectory User object.

The Matching policy should provide criteria that are guaranteed to produce a 0 or 1 match. More than one policy can exist, and the Metadirectory engine applies them in the order that they are defined. Any policy producing 0 or more than one match is skipped and the next policy is applied. Processing finishes when one match is found or after the last policy has been processed.

The default Subscriber Matching policy is an XML policy that attempts to find a PeopleSoft DIRXML_SCHEMA01 object that contains an ASSOC_ID attribute that matches the eDirectory User object’s workforceID attribute. If that policy fails, the driver uses another policy to locate a

PeopleSoft DIRXML_SCHEMA01 secondary object with a matching Given Name and Surname. The policy is defined in eDirectory class and attribute names because schema mapping has not yet been applied.

Create Policy

The Create policy specifies the criteria for creating a new object after the Matching policy has failed to find a match. This policy performs various tests and transformations based on the requirements for object creation in the DIRXML_SCHEMA01 CI in PeopleSoft and the business logic being applied.

The default Subscriber Create policy is a DirXML Script policy that asserts that the Add document is for a User object and that it must contain a value for Given Name, Surname, employeeStatus, departmentNumber, and jobCode. These fields are required because the default PSA has defined these fields as required in the DIRXML_SCHEMA01 CI. (Additional required fields, such as BirthDate and Manager, have defined default values in the CI and are thus not required here.) By making sure all required fields are present before submitting the Add event to the driver, synchronization performance and integrity is enhanced.

This default policy does not assert particular values that are required for employeeStatus, departmentNumber, and jobCode in the PSA, but such assertions can be added to the Subscriber policies if desired.

Output Transformation Policy

The Output Transformation policy is implemented as both a DirXML Script and an XSLT policy by default for the PeopleSoft Driver. Although the Output Transformation policy is not exclusively used by the Subscriber channel, it performs a subscribing role because it is used to transform the data format of any XDS document received from Identity Vault, regardless of which channel generated the submission of the document. An example of data transformation contained in this policy is the transformation of single-value eDirectory attributes into structured, multivalue scroll elements in PeopleSoft.

The following templates exist in the default Output Transformation policy. In addition to the listed templates, all Identity Manager policies contain identity-transform templates that allow the copying of XML attributes and elements that are passed through unmodified. Multiple instances of each listed template exist for each type of document or data element that can be received by the policy.

- “Write-back” on page 67
- “From-merge Write-back” on page 68
- “Manager Flag Data Transformation” on page 68
- “Add DN Value to Subscribe Adds” on page 68

Write-back

As documented in the Publisher channel Command Transformation policy, this template monitors all status document responses from Identity Vault. If a status document with a Success value is received and it contains <operation-data> with peoplesoft-cn and peoplesoft-dn values, the template holds the status document and issues a Modify document with the CN and DN values to the PeopleSoft application. When the write-back command completes, the original status document is returned to the Publisher channel.

From-merge Write-back

This policy behaves similarly to the one described above, but it is applied to modify documents that are generated when the Metadirectory engine merges data on matched or resynchronized objects.

Manager Flag Data Transformation

This template converts the Boolean True and False values of the Identity Vault isManager attribute into character Y and N values of the *PeopleSoft Manager* field.

Add DN Value to Subscribe Adds

When objects are added to PeopleSoft, this policy adds the DN of the IDV object to the event attributes.

Managing the Driver

7

As you work with the PeopleSoft driver, there are a variety of management tasks you might need to perform, including the following:

- Starting and stopping the driver
- Viewing driver version information
- Using Named Passwords to securely store passwords associated with the driver
- Monitoring the driver's health status
- Backing up the driver
- Inspecting the driver's cache files
- Viewing the driver's statistics
- Using the DirXML® Command Line utility to perform management tasks through scripts
- Securing the driver and its information

Because these tasks, as well as several others, are common to all Identity Manager drivers, they are included in one reference, the *Identity Manager 3.6 Common Driver Administration Guide*.

Troubleshooting the Driver

8

This section contains potential problems and error codes you might encounter while configuring or using the driver.

8.1 The Driver Is Not Processing Available Transactions or Is Processing Them Out of Order

- Set the driver trace level to 5 and verify that the DIRXML_DTTM and DIRXML_CURRDTTM values of the Transaction records being processed are in proper lexicographic format.
- If the records are not in the correct format, refer to Chapter 3, “Configuring Your PeopleSoft Environment,” on page 19.
- If the records are in the correct format, verify that Transaction date and time field values are correct and correspond to the system date and time.

8.2 Error Trying to Obtain Data Record

The following are typical reasons for this error:

- The Data record identified in a Transaction record was deleted from the PeopleSoft server before the Transaction was processed.

- The Data record identified in a query or Subscriber channel operation has been deleted from the PeopleSoft server.
- Through a database error or bad configuration, multiple Data records with the same primary key value exist in the PeopleSoft database.

  Verify the reason for the problem by using either an SQL tool, the PSA DirXML® Schema 01 sample application, or the PeopleSoft Application Designer's Test Component Interface tool (see Chapter 3, "Configuring Your PeopleSoft Environment," on page 19.) Correct any errors that might exist.

8.3 Error: joltServiceException: Invalid Session

This error is generated whenever the driver cannot access the target PeopleSoft Application Server. Typical reasons for the error are:

- The Application Server address and port number are incorrect or incorrectly specified (the format must be: //*address:port number*).
- The Application Server is down.

If this error occurs on the Publisher channel, the driver retries transaction processing in 60 seconds or the configured poll interval, whichever is greater. If the error occurs on the Subscriber channel, the Identity Manager engine schedules event retries.

8.4 The Driver Does Not Start

- Verify that `psoftshim.jar`, `dirxmlcomps.jar`, `psjoa.jar`, and the required CI API jar files are present in the DirXML `lib` subdirectory.
- Verify that the connection parameters are set correctly.
- Ensure that the configured CI names are valid.

8.5 Attributes Are Not Refreshed on the Data Schema Object

Verify that the Component Interfaces are working correctly by using PeopleSoft Application Designer tool. Refer to Chapter 3, "Configuring Your PeopleSoft Environment," on page 19.

8.6 Data Does Not Show Up in the Identity Vault on the Publisher Channel

- Verify that the Mapping policy and filters are configured correctly.
- Verify that the APIs are working correctly and data is being produced by them.

8.7 Error: Check Application Server IP Address and Jolt Port Number

Run the CITester utility to verify that proper connection and authentication parameters are set. Refer to Chapter 3, "Configuring Your PeopleSoft Environment," on page 19.

8.8 Data Does Not Update in PeopleSoft on the Subscriber Channel

- Verify that the Mapping policy and filters are configured correctly.
- Verify that the APIs are working correctly.
- If you are using <add> event functionality, verify that the Create method is available on the target schema CI.
- If you are using <delete> event functionality, verify that the Delete method is available on the target schema CI.

8.9 No Transactions Are Coming across the Publisher Channel

- Verify that there are active transactions in the queue ready for processing.
- Ensure that driver parameters are pointing to the correct PeopleSoft database. For example, transactions do not process if they are in the PROD database, and the driver is still pointing to the test database (which is configured to run with the driver, but holds no transactions).

8.10 Transactions Are Not Placed in the PeopleSoft Queue

Verify that PeopleCode is working properly.

8.11 Transactions Are Left in the “Process” State and Not Processed

- Verify that all of the CI objects can be processed and that the status can be updated to a Success (S), Warning (W), or Error (E) state.

  If e-mail is configured in PeopleSoft and the SMTP gateway is down, an error can occur, causing the update of the transaction to fail. You should verify that all online processing of the application works correctly. PeopleCode attached to the update might sometimes fail, causing the transaction to fail. If system connectivity is lost, the database or application server goes down during processing and causes the driver to abandon the transaction. The transaction is left in the selected state with a status of I.

  NOTE: If notification processing is required, we recommend using the Identity Manager Notification Service instead of using SMTP processing as configured in PeopleSoft.

8.12 Errors on the Publisher Channel When Processing a Transaction

The following list gives a sampling of errors and what they represent:

- Operation vetoed by the Create policy

  Possible required data missing in the Create policy or other criteria in the Create policy have an error.

- generateKeyPair: -216 DSERR_PASSWORD_TOO_SHORT

  The attribute used for the initial password does not comply with the policy; however, the user object is still created.

- Unable to read current state of 8101

  No association exists for this identity.

- nameToID: -601 ERR_NO_SUCH_ENTRY

  Possible Placement policy error with an invalid container object designated.

- No DN generated by Placement policy

  Possible missing or invalid data causing no valid DN to be created.

8.13 Component Interface Relationships Are Not Functioning

If data does not show up in the attributes, or isn't getting posted into PeopleSoft, or data is missing, you should begin looking at the Component Interface relationships.

First, verify that the API is getting the data from the PeopleSoft buffer.

After all of the CIs have been tested completely with validation of all processes that the driver is configured to do, there should be no issues regarding the driver accessing PeopleSoft through the CIs. Other problem areas include:

- Connectivity IP address and port for the application server
- ID and password
- Correct naming of all activities in the parameters for the driver.

For troubleshooting these problems, try three basic tests:

1. Manually test all of the processes online using the PeopleSoft applications as configured.
2. Test all of the processes using the Component Interfaces.
3. Test the driver connection to the API through the Component Interfaces.

8.14 SQL Error When Saving "Sample Person" Records

Error text example:

```
SQL error.Function: SQLExec
Error Position:  0
 Return:  8601 - [Microsoft][ODBC SQL Server Driver][SQL Server]FOR
 UPDATE cannot be specified on a READ ONLY cursor.
```

The DirXML_DERIVED.DIRXML_DRIVER.FieldFormula PeopleCode contains an SQLExec command that performs an exclusive lock on selected rows returned from a query for the current sequential transaction number in the DIRXML_TRANSNXT table. the command line is:

```
SQLExec("Select DIRXML_INST from DIRXML_TRANSXT Where DIRXML_DRIVER =:1
FOR UPDATE", &Driver, &InstanceID);
```

The “FOR UPDATE” locking clause is not valid for all flavors of SQL (such as SQL Server.) The clause can be safely removed to allow the PSA to function. However, if there is a possibility of simultaneous administrative access to the Transaction row creation functionality, the code should be modified by a qualified DBMS/PeopleSoft Database administrator to appropriately serialize the “Select” and “Insert or Update” of the DIRXML_TRANSNXT table.

8.15 Troubleshooting Driver Processes

Viewing driver processes is necessary to analyze unexpected behavior. To view the driver processing events, use DSTrace. You should only use it during testing and troubleshooting the driver. Running DSTrace while the drivers are in production increases the utilization on the Identity Manager server and can cause events to process very slowly. For more information, see “Viewing Identity Manager Processes” in the *Identity Manager 3.6 Common Driver Administration Guide*.

Driver Properties

A

This section provides information about the Driver Configuration and Global Configuration Values properties for the PeopleSoft driver. These are the only unique properties for drivers. All other driver properties (Named Password, Engine Control Values, Log Level, and so forth) are common to all drivers. Refer to "Driver Properties" in the *Identity Manager 3.6 Common Driver Administration Guide* for information about the common properties.

The information is presented from the viewpoint of iManager. If a field is different in Designer, it is marked with an  icon.

A.1 Driver Configuration

In iManager:

1. Click  to display the Identity Manager Administration page.
2. Open the driver set that contains the driver whose properties you want to edit:
   - **2a** In the *Administration* list, click *Identity Manager Overview*.
   - **2b** If the driver set is not listed on the *Driver Sets* tab, use the *Search In* field to search for and display the driver set.
   - **2c** Click the driver set to open the Driver Set Overview page.
3. Locate the driver icon, then click the upper right corner of the driver icon to display the *Actions* menu.
4. Click *Edit Properties* to display the driver's properties page.

   By default, the Driver Configuration page is displayed.

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver icon or line, then select click *Properties* > *Driver Configuration.*

The Driver Configuration options are divided into the following sections:

A.1.1 Driver Module

The driver module changes the driver from running locally to running remotely or the reverse.

Table A-1 *Driver Module*

Option	Description
Java	Used to specify the name of the Java class that is instantiated for the shim component of the driver. This class can be located in the `classes` directory as a class file, or in the `lib` directory as a `.jar` file. If this option is selected, the driver is running locally. The Java class name is: `com.novell.nds.dirxml.driver.psoftshim.PSOFTDriverShim`
Native	This option is not used with the PeopleSoft driver.
Connect to Remote Loader	Used when the driver is connecting remotely to the connected system. Designer includes two suboptions: ◆ *Driver Object Password*: Specifies a password for the Driver object. If you are using the Remote Loader, you must enter a password on this page. Otherwise, the remote driver does not run. The Remote Loader uses this password to authenticate itself to the remote driver shim. ◆ *Remote Loader Client Configuration for Documentation*: Includes information on the Remote Loader client configuration when Designer generates documentation for the driver.

A.1.2 Driver Object Password (iManager Only)

Table A-2 *Driver Object Password*

Option	Description
Driver Object Password	Use this option to set a password for the driver object. If you are using the Remote Loader, you must enter a password on this page or the remote driver does not run. This password is used by the Remote Loader to authenticate itself to the remote driver shim.

A.1.3 Authentication

The Authentication section stores the information required to authenticate to the connected system.

Table A-3 *Authentication*

Option	Description
Authentication ID or *User ID*	Specify an administrative account that the driver can use to authenticate to PeopleSoft. For more information, see Section 4.1, “Creating a PeopleSoft Account,” on page 41. Example: `PSAdmin`

Option	Description
Authentication Context or *Connection Information*	Specify the IP address or name of the PeopleSoft server the driver should communicate with. The connection string uses the following format: `<hostname or IP address>:<Port Number>` Example: `//PSServer:9000` To enable failover and load balancing, you can supply multiple server connection strings separated by a comma. Example: `//PSServer:9000,//111.222.3.4:9000`
Remote Loader Connection Parameters or *Host name* Port *KMO* *Other parameters*	Used only if the driver is connecting to the application through the remote loader. The parameter to enter is `hostname=xxx.xxx.xxx.xxx port=xxxx kmo=certificatename`, when the hostname is the IP address of the application server running the Remote Loader service and the port is the port the Remote Loader is listening on. The default port for the Remote Loader is 8090. The `kmo` entry is optional. It is only used when there is an SSL connection between the Remote Loader and the Metadirectory engine. Example: `hostname=10.0.0.1 port=8090 kmo=IDMCertificate`
Driver Cache Limit (kilobytes) or *Cache limit (KB)*	Specify the maximum event cache file size (in KB). If it is set to zero, the file size is unlimited. Click *Unlimited* to set the file size to unlimited in Designer.
Application Password or *Set Password*	Specify the password for the user object listed in the *Authentication ID* field.
Remote Loader Password or *Set Password*	Used only if the driver is connecting to the application through the Remote Loader. The password is used to control access to the Remote Loader instance. It must be the same password specified during the configuration of the Remote Loader on the connected system.

A.1.4 Startup Option

The Startup Option section allows you to set the driver state when the Identity Manager server is started.

Table A-4 *Startup Option*

Option	Description
Auto start	The driver starts every time the Identity Manager server is started.

Option	Description
Manual	The driver does not start when the Identity Manager server is started. The driver must be started through Designer or iManager.
Disabled	The driver has a cache file that stores all of the events. When the driver is set to Disabled, this file is deleted and no new events are stored in the file until the driver state is changed to Manual or Auto Start.
Do not automatically synchronize the driver	This option only applies if the driver is deployed and was previously disabled. If this is not selected, the driver re-synchronizes the next time it is started.

A.1.5 Driver Parameters

The Driver Parameters section lets you configure the driver-specific parameters. When you change driver parameters, you tune driver behavior to align with your network environment.

The parameters are presented by category:

- Table A-5, “Driver Settings,” on page 80
- Table A-7, “Publisher Settings,” on page 81
- Table A-7, “Publisher Settings,” on page 81

Table A-5 *Driver Settings*

Option	Description
Schema CI Name	Allows you to list the Data Schema Component Interfaces that the driver synchronizes. Use the *Plus* icon to add an item, the *Minus* icon to delete an item, and the *Pen* icon to edit an item. The default is *DIRXML_SCHEMA01*.
Data Record ID Field	Specify the name of the data record definition that uniquely identifies a PeopleSoft object. The default is *Assoc_ID*.
Use Case-Sensitive Search	Controls whether the driver uses case-sensitive matching criteria while evaluating search attribute matches. The default is *No Case-Sensitive Matching*.

Table A-6 *Subscriber Settings*

Option	Description
Allow "add" events	Subscriber Add events are implemented by invoking the Component Interface Create method (if present). If you want the driver to allow Subscriber channel Add events, select *Allow Subscriber add*. The default is *Disallow Subscriber add*.
Data Record ID Field Default Value	Specify a default value for the Schema CI key field. This parameter is used only for Subscriber channel Add events. The default is *New*.

Option	Description
Allow "delete" events	Subscriber Delete events are implemented by invoking the Component Interface Delete method (if present). If you want the driver to allow Subscriber channel Delete events, select *Allow Subscriber delete*. The default is *Disallow Subscriber delete*.

Table A-7 *Publisher Settings*

Parameter	Description
Transaction CI Name	Specify the name of the PeopleSoft CI object that defines the set of fields required for the driver Transaction interface. The set of fields in the specified transaction CI must contain the same fields and keys that are identified in the default transaction CI in order for the driver to work. The default is *DIRXML_TRANS01.*
Driver Subset Identifier	Specify the name of the field in the Data Schema CI that uniquely identifies a PeopleSoft object. This name is used for All Data Schema CIs that are specified in the *Schema CI Name* parameter. The field identifies which transactions in the transaction CI are to be processed by the driver. When the driver reads a transaction CI record, it compares the values of the DIRXML_DRIVER field with this parameter value and only processes transactions that match. A match is determined by matching characters for the length of this parameter value. For instance, if this parameter is NPSDriver and the DIRXML_DRIVER field in a transaction is NPSDriver1, a match is made. This allows multiple drivers to utilize the same transaction CI, which in turn can be populated by multiple PeopleSoft applications or processes.
Publisher Polling Option	The PeopleSoft driver supports two options for Publisher Transaction record polling. To choose an interval of seconds between polls, select *Utilize Interval Polling*. To use a crontab format, select *Utilize crontab Format Polling*. If you select *Utilize Interval Polling*, fill in the *Queue Poll Interval* by specifying the number of seconds between checks for available transactions to process. The default is *5*. If you select *Utilize crontab Format Polling*, fill in the *Enter Queue Poll crontab Format String* by specifying the polling interval in crontab format. The default value of * * * * * generates a poll every minute.
Publisher Heartbeat Interval	Specify how many minutes of inactivity can elapse before this channel sends a heartbeat document. In practice, more than the number of minutes specified can elapse. That is, this parameter defines a lower bound.

A.2 Global Configuration Values

Global configuration values (GCVs) are values that can be used by the driver to control functionality. GCVs are defined on the driver or on the driver set. Driver set GCVs can be used by all drivers in the driver set. Driver GCVs can be used only by the driver on which they are defined.

The PeopleSoft driver includes several predefined GCVs. You can also add your own if you discover you need additional ones as you implement policies in the driver.

To access the driver's GCVs in iManager:

1. Click  to display the Identity Manager Administration page.
2. Open the driver set that contains the driver whose properties you want to edit.
   - **2a** In the *Administration* list, click *Identity Manager Overview*.
   - **2b** If the driver set is not listed on the *Driver Sets* tab, use the *Search In* field to search for and display the driver set.
   - **2c** Click the driver set to open the Driver Set Overview page.
3. Locate the driver icon, click the upper right corner of the driver icon to display the *Actions* menu, then click *Edit Properties*.

   or

   To add a GCV to the driver set, click *Driver Set*, then click *Edit Driver Set properties*.

To access the driver's GCVs in Designer:

1. Open a project in the Modeler.
2. Right-click the driver icon  or line, then select *Properties* > *Global Configuration Values.*

   or

   To add a GCV to the driver set, right-click the driver set icon  , then click *Properties* > *GCVs*.

Table A-8 *Global Configuration Values*

Option	Description
Identity Manager accepts passwords from application	If *True*, allows passwords to flow from the connected system to Identity Manager. In Designer, you must click the  icon next to an option to edit it. This displays the Password Synchronization Options dialog box, which has a better display of the relationship between the different GCVs. In iManager, you should edit the Password Management Options on the Server Variables tab rather than under the GCVs. The Server Variables page has a better display of the relationship between the different GCVs. For more information about how to use the Password Management GCVs, see "Configuring Password Flow " in the *Identity Manager 3.6 Password Management Guide*.
Publish passwords to NDS password	Use the password from the connected system to set the non-reversible NDS® password in eDirectory.
Publish passwords to Distribution Password	Use the password from the connected system to set the NMAS™ Distribution Password used for Identity Manager password synchronization.

Option	Description
Require password policy validation before publishing passwords	If *True*, applies NMAS password policies during publish password operations. The password is not written to the data store if it does not comply.
Notify the user of password synchronization failure via e-mail	If *True*, notify the user by e-mail of any Password Synchronization failures.
Connected System or Driver Name	The name of the connected system, application, or Identity Manager driver. This value is used by the e-mail notification templates.
Password Failure Notification User	Password Synchronization policies are configured to send e-mail notifications to the associated user when password updates fail. To send a copy to another user, such as an administrator, specify the DN of that user. Otherwise, leave this field blank.
Active Users Container	The name of the Organizational Unit container in the Identity Vault where Active users will be placed. Specify the DN of the OU object. Otherwise, leave this field blank.
Inactive Users Container	The name of the Organizational Unit container in the Identity Vault where the inactive users will be placed. Specify the DN of the OU object. Otherwise, leave this field blank.
Active Employees Group	The name of the Group object in the Identity Vault where Active Employee users will be added. Specify the DN of the object. Otherwise, leave this field blank.
Active Managers Group	The name of the Group object in the Identity Vault where Active Manager users will be added. Specify the DN of the object. Otherwise, leave this field blank.