Novell Nsure™ Identity Manager Driver for PeopleSoft*

5.0

www.novell.com

IMPLEMENTATION GUIDE

December 15, 2004

Novell®

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.

You may not use, export, or re-export this product in violation of any applicable laws or regulations including, without limitation, U.S. export regulations or the laws of the country in which you reside.

Copyright © 2000-2004 Novell, Inc. All rights reserved. No part of this publication may be reproduced, photocopied, stored on a retrieval system, or transmitted without the express written consent of the publisher.

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 at 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

Identity Manager Driver for PeopleSoft Implementation Guide
December 15, 2004

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

Novell Trademarks

DirXML is a registered trademark of Novell, Inc., in the United States and other countries.
eDirectory is a trademark of Novell, Inc.
Novell is a registered trademark of Novell, Inc., in the United States and other countries.
Nsure is a trademark of Novell, Inc.

Third-Party Trademarks

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

Contents

About This Guide

The Identity Manager Driver 5.0 for PeopleSoft provides a solution for synchronizing data between Novell® eDirectory™ and PeopleSoft.

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

The guide contains the following sections:

Additional Documentation

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

Documentation Updates

For the most recent version of this document, see the Drivers Web site (http://www.novell.com/documentation/lg/dirxmldrivers/index.html).

Documentation Conventions

In this 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.

User Comments

We want to hear your comments and suggestions about this manual and the other documentation included with Novell Nsure Identity Manager. To contact us, send e-mail to proddoc@novell.com.

1 Introducing the Identity Manager Driver 5.0 for PeopleSoft

PeopleSoft applications are some of the most popular Enterprise Resource Planning (ERP) systems available. The Identity Manager Driver 5.0 for PeopleSoft enables you to create and manage Novell® eDirectory™ objects 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 eDirectory. Using Novell Nsure™ 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:

Driver Components

The driver includes the following components:

- Driver shim

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

- Driver Configuration

  PeopleSoft50.xml is a driver configuration that is imported using Novell iManager. It contains configuration parameters and policies that enable Identity Manager to handle the synchronization and data object manipulation between PeopleSoft and eDirectory.

  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 3, "Installing and Configuring the Driver," on page 35 and "Customizing the Driver by Modifying Driver Policies" on page 45.

- 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 eDirectory and PeopleSoft. Examples of PSA uses include the following:

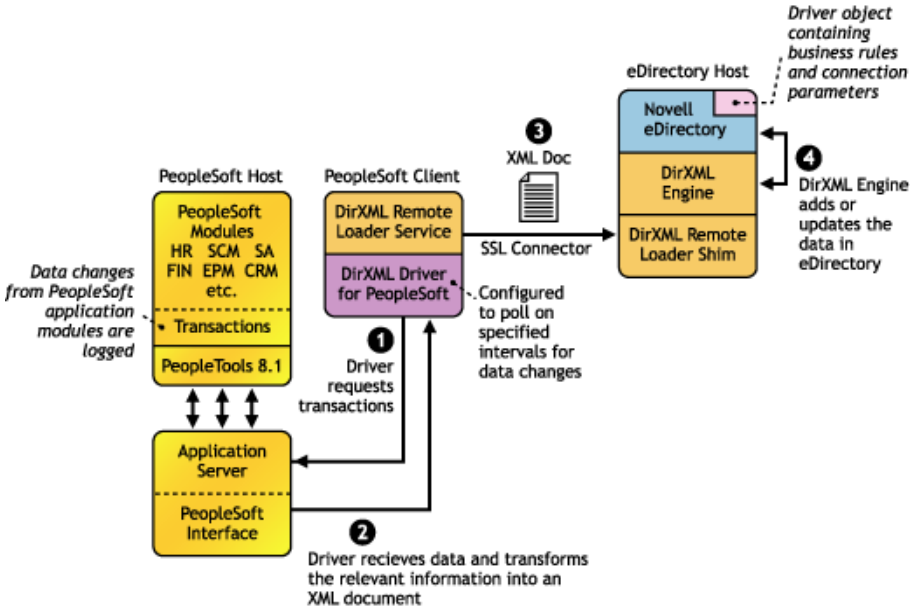
- Implementation of an intermediate staging table. The synchronization between the Novell-provided sample Personnel Application and the staging table shows the best practices of PeopleSoft internal application integration using PeopleCode Component Interfaces.
- Integration can be accomplished directly 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 "Using the PeopleSoft Service Agent" on page 15.

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 3, "Installing and Configuring the Driver," on page 35.

The Publisher Channel

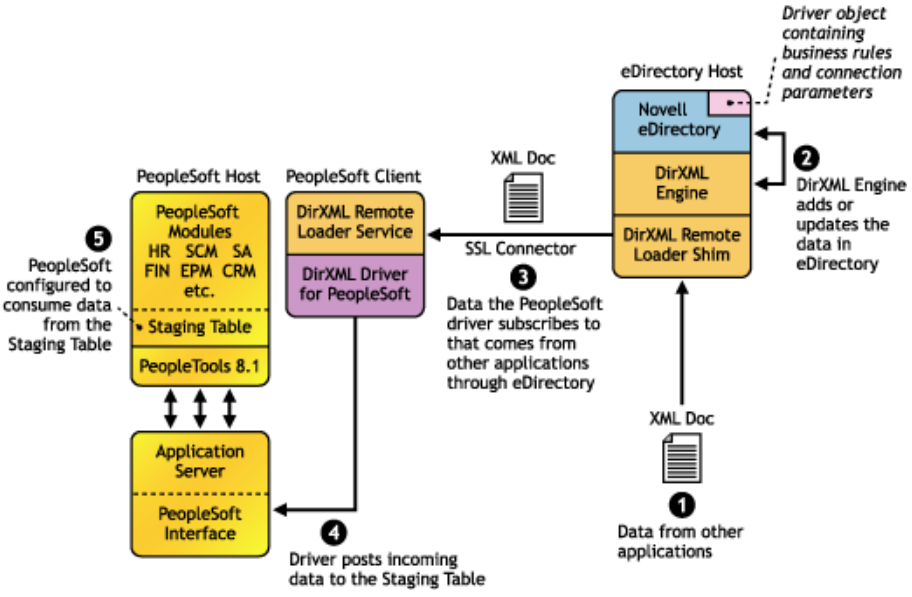


The Publisher channel synchronizes data from PeopleSoft to eDirectory. As events occur within PeopleSoft, transactions are placed into a transaction table. These transactions are usually written to the table through PeopleCode (you can use other methods such as Batch SQL, COBOL, SQR, and so forth.) Component 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 the subtype of 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 DirXML engine for processing. When the DirXML 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 eDirectory, 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



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

As events occur within eDirectory, the driver receives an XML document from the DirXML 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 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.)

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 2, “Configuring Your PeopleSoft Environment,” on page 15.

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 and initial passwords as new data is synchronized from PeopleSoft.
- Synchronize data bidirectionally between a PeopleSoft application and eDirectory.
- 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 eDirectory 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 5, "Customizing the Driver," on page 41.

Prerequisites

- ❑ Novell Nsure Identity Manager 2.0.1.
- ❑ PeopleSoft Application Server with PeopleTools version 8.17 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 5, "Customizing the Driver," on page 41.

Driver Features

This section explains the new features available in this version of the driver.

Driver Features

The driver contains some significant changes designed to make it easier to deploy in customer environments. The primary changes include:

- An enhanced PeopleSoft Service Agent (PSA) that provides sample integration components and objects using the best practices of PeopleSoft integration.
- The PSA components now support Subscriber channel Add and Delete commands. This enables full bidirectional driver functionality.
- The PSA now includes utilities for managing the transaction record database. This allows easier archive, delete, reset, etc. of transactions.
- The driver shim is written in Java. This allows the driver to be deployed on any supported PeopleTools platform.
- The driver shim can be either a Local or Remote Loader driver.
- The driver can handle queries using any Level-0 fields as search criteria.
- The driver uses a default "explicit" search value criterion. The driver supports a "starts with" search value matching option if the search value is contained within single quotes (such as 'P00000').
- The driver can coexist with prior releases in PeopleTools 8.1*x* and 8.4*x* environments, so it provides the benefit of parallel testing. After you have implemented this version of the driver in production, you can delete all older objects within PeopleSoft that are attached to previous versions of the driver.

The driver delivers XSLT templates to facilitate upgrades in 4.0.*x* driver deployments.

The new architecture and driver configuration enable you to quickly implement the driver into your testing environment. After you have configured and tested the driver in you lab environment, you can then customize the implementation to meet your business needs, work with your application data, and then move it into production.

As is the case with any implementation, when the level of complexity in your business processes increases, so does the amount of time and effort required to implement your solution. For highly complex implementations, you might need to obtain external consulting assistance.

Identity Manager Features

Identity Manager includes new features. For more information, refer to the "What's New in Identity Manager 2?" in the *Nsure Identity Manager 2 Administration Guide* (http://www.novell.com/documentation/lg/dirxml20/admin/data/alxnk27.html).

2 Configuring Your PeopleSoft Environment

This section discusses how the PeopleSoft Service Agent (PSA) works and how to configure your PeopleSoft environment.

Using the PeopleSoft Service Agent

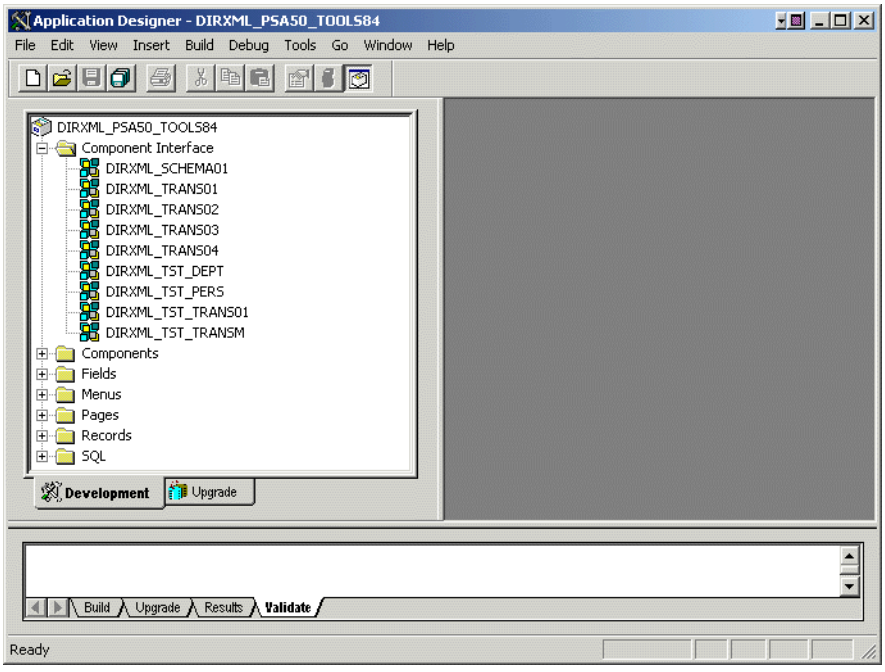
The PeopleSoft Server Agent (PSA) is a set of PeopleTools objects and code that enables you to define the integration between PeopleSoft applications and Novell® eDirectory®. It includes all of the components for a sample Personnel application, a staging table for moving data between the Sample application 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.

In this section, you will find installation and configuration information for the two main PSA components: "The Component Interface Infrastructure for Identity Manager" on page 16 and "The Sample Application" on page 16.

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.

NOTE: 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 6, "Troubleshooting the Driver," on page 57 for specific issues and consult with your DBMS/PeopleSoft Database administrator for additional assistance.

The Component Interface Infrastructure for Identity Manager



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 27.

The Sample Application

The PSA project has sample Personnel Applications that you can install on your PeopleSoft 8.1*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 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 19.

Installing the PSA Sample Project

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

1. “Installing the PSA Files” on page 17
2. “Importing the PSA Project into the PeopleSoft Database” on page 17
3. “Building Project Record Definitions” on page 18
4. “Applying Security to the PSA” on page 18
5. “Understanding the Architecture of the PSA Sample Project” on page 19
6. “Testing Sample PeopleSoft Applications” on page 21

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).

To ensure that the PSA can be imported into the PeopleSoft Application server, make sure of the following:

- Make sure that the PSA files have write access enabled. For example, in Windows, you should turn off read-only file properties.

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.

For PeopleSoft 8.1

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.

For PeopleSoft 8.4

1. Connect to the PeopleSoft database as administrator in two tier mode.
2. In the Application Designer, select 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.

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 PeopleSoft using an administrator username that has administrative and development rights.
2. From 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 using 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.

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.

For PeopleSoft 8.1

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.

For 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 the 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 Role tab, then click Add to add a new role.
11. Search and select the role, click DirXML Administrator to add it, then click Save.

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 direction 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

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 using 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 will be recorded with an NPSDriver1P identifier that shows the data is being published by a PeopleSoft administrator. Changes made via the Component Interface API are be 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 to affected data tables. In the PSA, the DIRXML_S_PERS data rows are created and updated with PeopleCode via the DIRXML_TST_PERS Component Interface. The Component Interface 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, etc. Most important is the fact that the Component Interface 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 Component Interface has been extended with a Delete method that enables removal of data schema records via the driver's Subscriber channel. Use of 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, this 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 Component Interface. 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 Component Interfaces 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 Chapter 3, "Installing and Configuring the Driver," on page 35.
- Test everything thoroughly.

Testing Sample PeopleSoft Applications

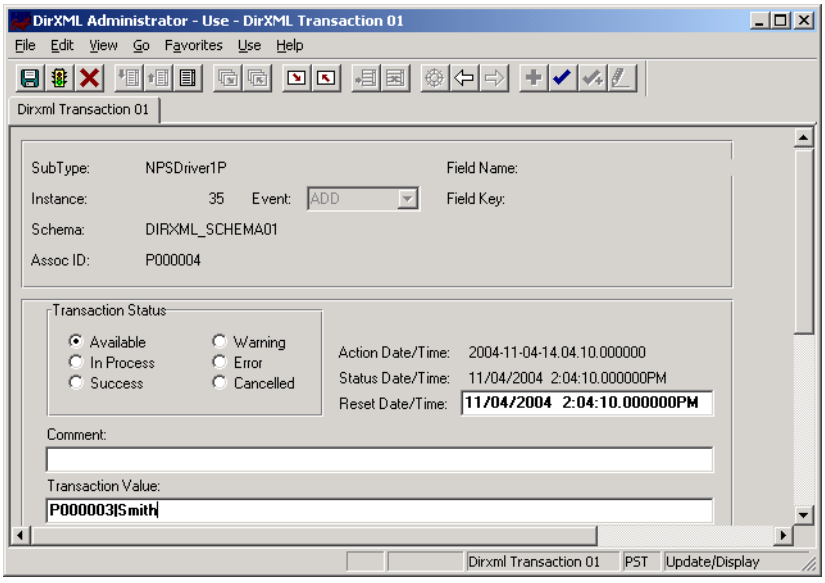
You can test to ensure that transactions are created by entering a new person using the PeopleSoft Identity Manager sample application. This example uses Departments, so you need to create a sample department and then add a person (assigning him or her to that department) to validate that the application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

For PeopleSoft 8.1

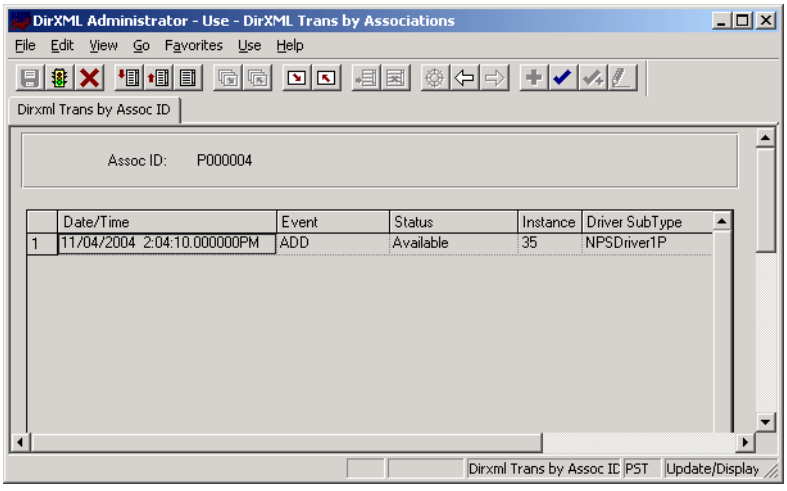
1. In the Application Designer, select Go > DirXML Administrator.
2. In the DirXML Administrator menu, select Use > DirXML Sample Department.
3. Click an empty Department field row to add sample department, description, and DirXML DN values.

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. Validate 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.

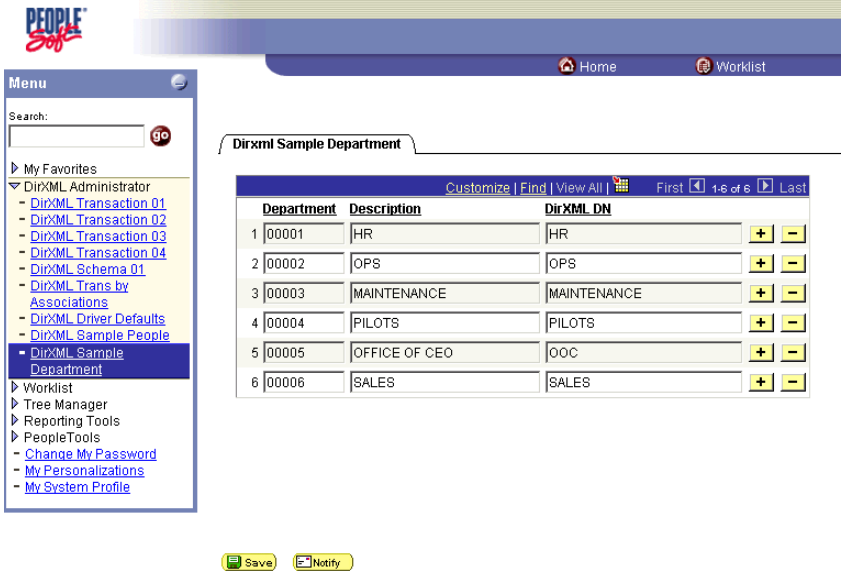
For PeopleSoft 8.4

1 Log in to the PeopleSoft portal.

2 Click DirXML Administrator from the left-side 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.

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

Component Interfaces

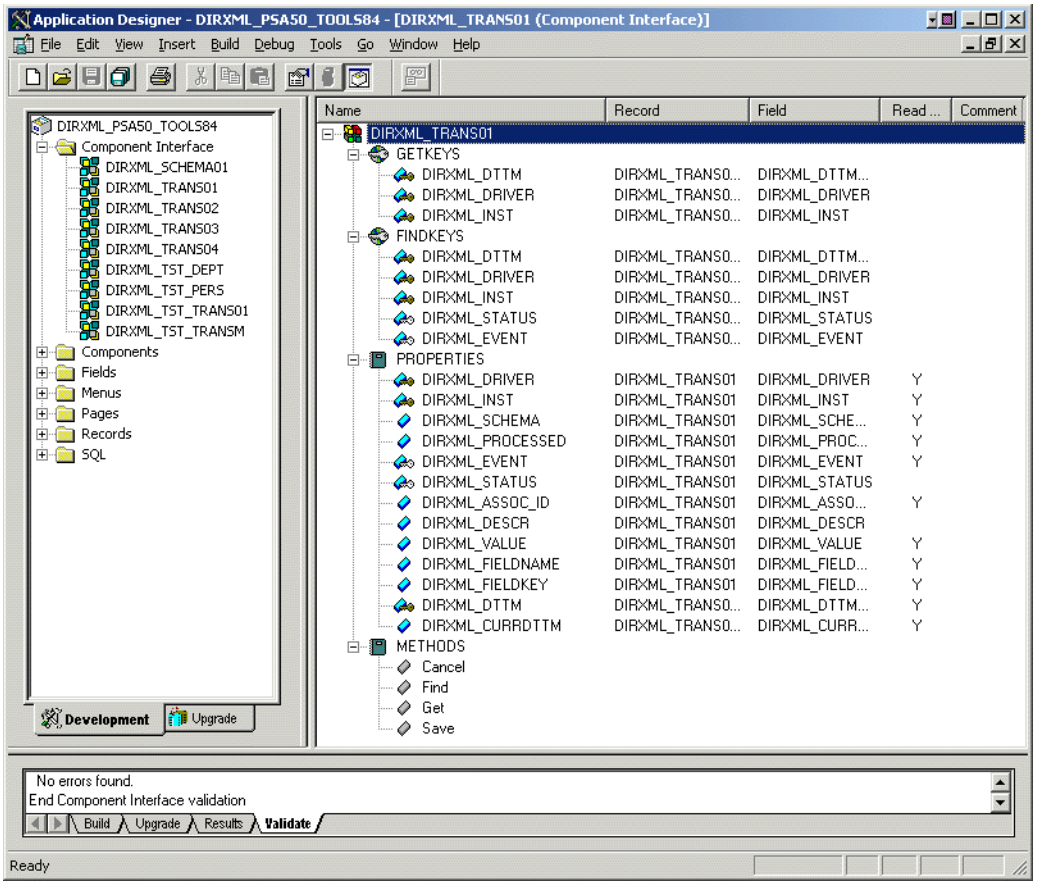
- "Accessing Transactions and Data through Component Interfaces" on page 24
- "Configuring the Transaction Record SQL Date/Time Format" on page 26
- "Configuring PeopleCode to Trigger Transactions" on page 27

Accessing Transactions and Data through Component Interfaces

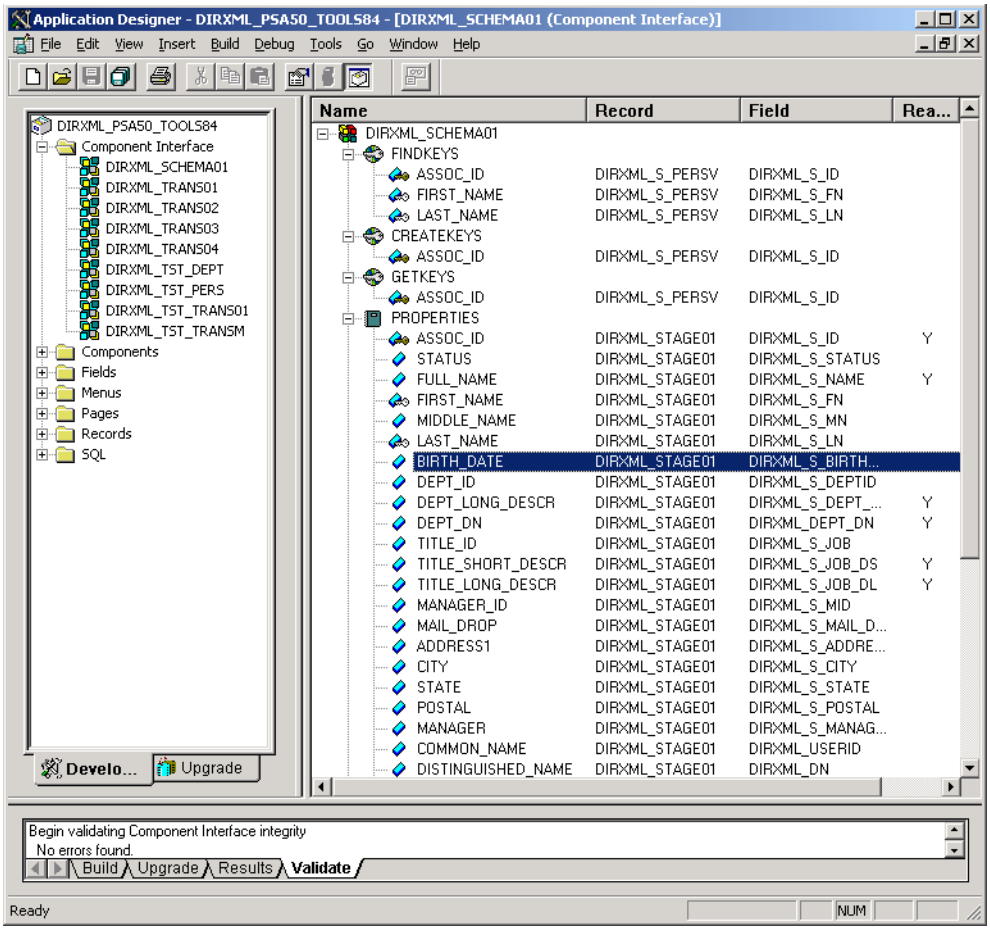
The driver accesses transactions waiting to be processed from the transaction table via the Component Interface (CI) object that is defined within PeopleTools. Each Component Interface maps to a particular component. Components are built in order to access transaction tables and schema data. Schema objects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

The driver uses only one Transaction CI in order to access transactions. Every transaction is assigned to one default Schema CI, although processing that is defined within a policy might request a query to other Schema CIs that are defined in the driver's Schema Mapping policy.

In the driver's parameters, you must specify the 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:



The following figure represents the CI for a Schema Component:



The driver uses a defined parameter to determine the subtype of transactions it needs to process. As a PeopleSoft developer, you determine this value when configuring a PeopleCode function call to trigger a transaction online, or when creating a transaction via a batch process. If the parameter does not exist, the driver processes all transactions available through the CI. If the parameter exists, the driver limits processing to the transaction type specified in the subtype string.

The PeopleCode DIRXML_TRANS function calls should always be placed in the SavePostChange PeopleCode on the record definition. Also, remember to declare the function prior to calling it.

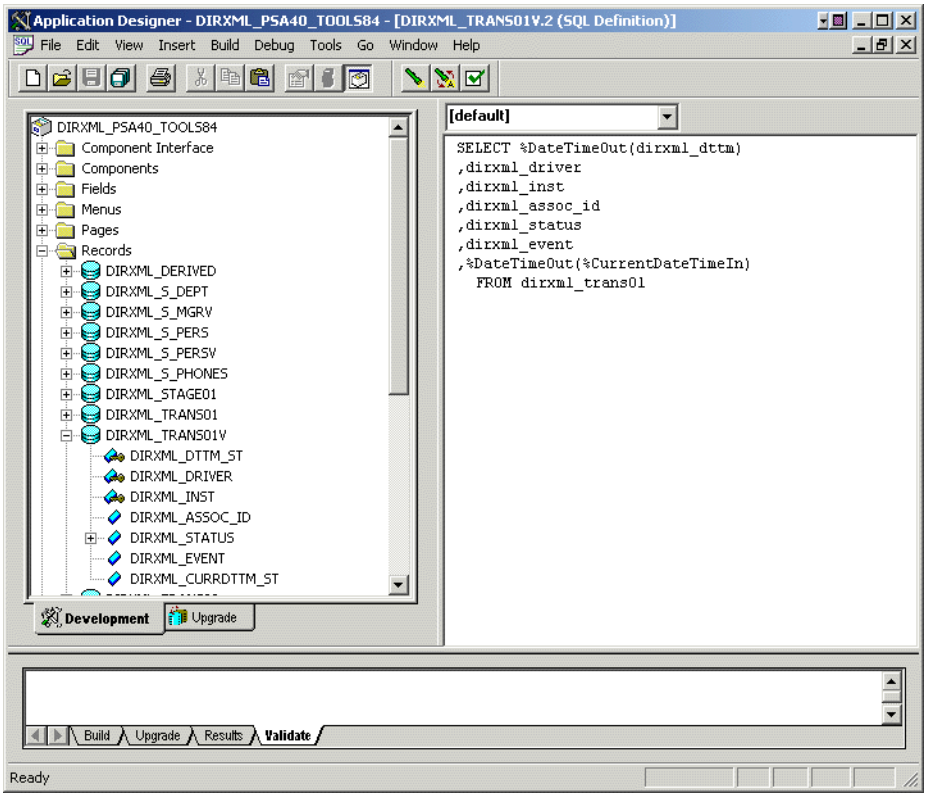
Changes to Field Names in PeopleSoft 8.41

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 Component Interface 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 Component Interfaces 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.

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 using the PeopleCode Meta-SQL %DateTimeOut function. The following image shows the default SQL View 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 is 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/Time that the Transaction data row was created and the current processing Date/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.

NOTE: Since 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.

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 Component Interface 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 DirXML 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 DirXML 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.

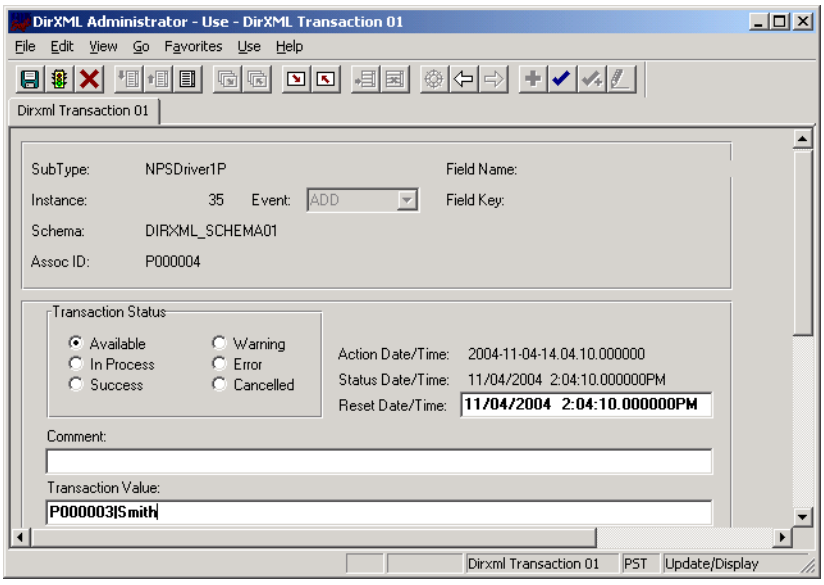
The driver requires the Transaction and Schema Component to process transactions. For more information on calling the PeopleSoft function that creates transaction records, please refer to “Customizing the PSA by Triggering Transactions” on page 41.

Transaction Component

The Transaction Component 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 association 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, which it uses to create an Add, Modify, or Delete XML document. The driver uses the Schema ID and the Association ID values to access the appropriate CI Schema and appropriate record information associated with the Association ID object.

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



Schema Component

The Schema Component 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 and Schema CI name, it accesses the appropriate Schema object.

When the driver accesses the Schema Component, it uses the value it received in the Association ID and the Key value to retrieve the data from the PeopleSoft environment. It also uses this component in order to update PeopleSoft for the record owned by the Associated Key Value.

For example, assume you want to process transactions for employees with the Data Record ID field and key value = P000001.

The driver accesses the Schema Component with a key value of P000001. It retrieves all of the component elements that have been defined for that employee. The driver then converts the data collection into XML documents to be consumed by the DirXML engine. If there is a write-back command processed, or when an event is subscribed to on the Subscriber channel, the driver uses this component to update the staging table with the appropriate information into PeopleSoft for this particular employee.

When the driver creates a new object, it creates an association value that contains the Association ID. When Schema CIs are loaded by the driver, each Schema CI is treated as a schema class. The Mapping policy uses the class mapping as appropriate. This allows the driver to know what Schema object in PeopleSoft is used for any particular element, and how to update data back into PeopleSoft.

Testing Component Interfaces

The Component Tester program (CITester.class) is included as part of the download. This program ensures that your PeopleTools client software is installed and configured properly on the computer hosting the driver.

The program validates the proper installation, configuration, and revision of the PeopleSoft PeopleTools client software on the computer hosting the DirXML Driver for PeopleSoft. The program performs all of the validation procedures in the same manner as the driver. Therefore, successful operation of CITester ensures 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. *

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 for PeopleTools 8.4x.

*If you are not using the default Schema CI, it will be necessary to build the APIs for the desired CI. See "Changing the Driver by Changing Data Schema Component Interface" on page 43 for information on building custom CI API JAR files.

Important Considerations

When running this 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 files 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 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 will see the following message:

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

Phase I Errors

You might encounter the following errors during the test. Suggestions for resolving errors are provided below.

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

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

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

Phase II Errors

You might encounter the following errors during the test. Suggestions for resolving errors are provided below.

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 will hang and will require a manual interrupt.

Error Message	Solution
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/Warning messages. Either the User ID or User ID Password are incorrect or have been entered using improper case.

Phase III: Authenticating to the PeopleSoft Client

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

Enter the Transaction Component Interface name or press Enter to validate DIRXML_TRANS01.

If the test program retrieves and validates that all required fields and elements are present, you will 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.
```

Phase III Errors

You might encounter the following errors during the test. Suggestions for resolving errors are provided below.

Error Message	Solution
PeopleSoft Error/Warning Messages Pending. Number of Messages: 4 Message 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 only 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: 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 key field that is specified via the Data Record ID field name.

To test the Schema CI, type the Schema Component Interface 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 will see the following message:

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

- Property 'AssocID' is present.
- Property 'Status' is present.
- Property 'FullName' is present.
- Property 'FirstName' is present.
- Property 'MiddleName' is present and validated as key field.
- Property 'LastName' is present.
- Property 'BirthDate' is present.
- Property 'DeptID' is present.
- Property 'DeptLongDescr' is present.
- Property 'DeptDN' is present.
- Property 'TitleID' is present.
- Property 'TitleShortDescr' is present.
- Property 'TitleLongDescr' is present.
- Property 'ManagerID' is present.
- Property 'MailDrop' is present.
```

```
- Property 'Address1' is present.
- Property 'City' is present.
- Property 'State' is present.
- Property 'Postal' is present.
- Property 'Manager' is present.
- Property 'CommonName' is present.
- Property 'DistinguishedName' is present.
- Property 'Description' is present.
- Property 'Email' is present.
- Property 'BusinessPhone' is present.
- Property 'CellPhone' is present.
- Property 'HomePhone' is present.
- Property 'Pager' is present.

** Schema Component Interface element validation succeeded.


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

Phase IV Errors

You might encounter the following errors during the test. Suggestions for resolving errors are provided below.

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       : AssocID
Transaction CI Name       : DIRXML_TRANS01
```

3 Installing and Configuring the Driver

This section contains the following information:

Installation Instructions

The driver contains three components that can be installed within your environment on multiple systems and platforms. They include the driver shim, PeopleSoft Service Agent (PSA) and the driver policies.

Depending on your system configuration, you might need to run the installation program several times to install driver components on the appropriate systems. For example, you would install the driver shim where Identity Manager or Remote Loader exists, the PSA where the PeopleSoft Application Server exists, and the driver policies to your iManager server.

This section contains the following topics:

Pre-Installation Instructions (All Platforms)

There are three separate .jar files required to complete the driver installation:

- psjoa.jar: The PeopleSoft middleware library. This file is located in the web/psjoa directory of the PeopleTools software distribution. You should copy this file to the \lib subdirectory with the driver Java library files.

  This file, and any additional Component Interface API class libraries that your solution requires, should be placed together in the \lib subdirectory with the DirXML Java library files. By default this is Novell\NDS\lib for a local driver installation or Novell\RemoteLoader\lib for a remote installation.
- psoftshim.jar: This is the driver shim and is installed when you run the installation program.
- dirxmlcomps.jar: This library contains the compiled APIs for the DIRXML_SCHEMA01, DIRXML_TST_PERS, and DIRXML_TRANS*nn* Component interfaces delivered in the PSA. This library is installed when you run the installation program.

Windows Installation Instructions

1. Double-click ps50install.exe.
2. Follow the installation wizard to install the components on your systems.
3. Click Finish to close the installation program. Proceed to "Importing the Driver Configuration" on page 36.

UNIX Installation Instructions

1. To launch the installation program for your platform, type one of the following commands:

Platform	Command
Linux	/ps50_linux.bin
Solaris	/ps50_solaris.bin
AIX	/ps50_aix.bin

   Entering /ps50_linux.bin -i console launches the installation program in text mode.

2. Follow the installation wizard to install the components on your systems.
3. Click Finish to close the installation program. Proceed to "Importing the Driver Configuration" on page 36.

Importing the Driver Configuration

The Create Driver Wizard helps you import the basic driver configuration file. This file creates and configures the objects and policies needed to make the driver work properly.

1. In Novell® iManager, click DirXML Utilities > Create Driver.
2. Select a driver set.

   If you place this driver in a new driver set, you must specify a driver set name, context, and associated server.

3. Select Import a Driver Configuration from the Server, then select PeopleSoft50.xml.

   The driver configuration files are installed on the Web server when you install Identity Manager. During the import, you are prompted for the driver's parameters and other information.

4. Specify values for the following parameters:

Parameter Name	Parameter Description
Driver name	The actual name you want to use for the driver.
Active Users Container	The name of the Organizational Unit object where Active users are placed.
Inactive Users Container	The name of the Organizational Unit where Inactive users are placed.
Active Employees Group	The name of the Group Object to which Active Employee users are added.

Parameter Name	Parameter Description
Active Managers Group	The name of the Group Object to which Active Manager users are added.
PeopleSoft Connection String	The host name 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.
PeopleSoft User ID	The PeopleSoft User ID the driver uses for authentication to PeopleSoft.
PeopleSoft User Password	The PeopleSoft User password the driver uses for authentication to PeopleSoft.
Configure Data Flow	Dataflow can be configured to one of the following options: ◆ Bidirectional: PeopleSoft and eDirectory are both authoritative sources of the data synchronized between them. ◆ PS-to-eDirectory: PeopleSoft is the authoritative source. ◆ eDirectory-to-PS: eDirectory is the authoritative source.
Install Driver as Remote/Local	Configure the driver for use with the Remote Loader service by selecting the Remote option, or select Local to configure the driver for local use. (If you are using PeopleTools 8.4*x,* you must select a Remote installation. Local implementations are not supported.) If Local is selected, you can skip the remaining parameters.
Remote Host Name and Port	Specify the host name or IP address and port number for where the Remote Loader service has been installed and is running for this driver. The default port is 8090.
Driver Password	The driver object password is used by the Remote Loader to authenticate itself to the DirXML server. It must be the same password that is specified as the driver object password on the Identity Manager Remote Loader.
Remote Password	The Remote Loader password is used to control access to the Remote Loader instance. It must be the same password that is specified as the Remote Loader password on the Identity Manager Remote Loader.

The additional driver parameters are set to default values during the import process, but they can be modified in iManager (by clicking the Driver Configuration tab on the driver object.)

Parameter	Description	Default Value
Schema CI Name	The name of the PeopleSoft CI object that defines the set of data to be synchronized by the driver.	DIRXML_SCHEMA01
Data Record ID Field	The name of the field in the Data Schema CI that uniquely identifies a PeopleSoft object. The value in this field is used as the DirXML object association identifier.	DIRXML_ASSOC_ID
Use Case-Sensitive Search	Controls whether or not the driver evaluates search attribute matches using case-sensitive match criteria.	
Allow Add Events	When data flow is configured to allow Subscriber Channel synchronization, this parameter allows the administrator to allow or deny Add events on the Subscriber Channel.	
Data Record ID Field Default Value	Allows an administrator to specify the default value for the Schema CI key field. Only used for Subscriber Channel Add events.	NEW

Parameter	Description	Default Value
Allow Delete Events	When data flow is configured to allow Subscriber Channel synchronization, this parameter allows the administrator to allow or deny Delete events on the Subscriber Channel.	
Transaction CI Name	Contains the name of the PeopleSoft CI object that defines the set of fields required for the DirXML Transaction interface. The set of fields in the specified transaction CI must contain the same fields and keys identified in the default transaction CI in order for the driver to work.	DIRXML_TRANS01
Driver Subset Identifier	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.	NPSDriver
Queue Poll Interval (seconds)	This parameter specifies the number of seconds the driver waits between attempts to process transaction records. This poll interval is only applied when no transactions are available for processing.	5 seconds

5. Click Import.
6. When the import is finished, you can define security equivalences and exclude administrative roles from replication.

   The driver object must be granted sufficient eDirectory rights to any object it reads or writes. You can do this by granting Security Equivalence to the driver object. The driver must have Read/Write access to users, post offices, resources, and distribution lists, and Create, Read, and Write rights to the post office container. Normally, the driver should be given security equal to Admin.
7. Review the driver objects in the Summary screen, then click Finish.

Activating the Driver

Activation must be completed within 90 days of installation or the driver will not run.

For more information, refer to Activating Your Identity Manager Product (http://www.novell.com/documentation/lg/dirxml20/index.html) in the *Novell Nsure Identity Manager 2 Administration Guide*.

4 Upgrading the PeopleSoft Driver

This section discusses considerations you should take when upgrading from the 4.0 version of the PeopleSoft driver to the 5.0 version of the driver.

The install program places four subdirectories under the selected install root directory. The components include:

\lib
dirxmlcomps.jar
psoftshim.jar

\PSUtils
AssociationStyleTransform
CITester.class
CITester.bat

\PSA
DIRXML_PSA50_TOOLS81.exe
DIRXML_PSA50_TOOLS84.exe

\precfg
PeopleSoft50.xml
PeopleSoft50_fr.xlf

The following considerations reference files in these locations.

- The driver shim is no longer a Win32 native shim. It is a pure Java shim that can run as either a local driver or a Remote Loader driver. The driver configuration or Remote Loader configuration must be changed to set the driver shim class path to com.novell.nds.dirxml.driver.psoftshim.PSOFTDriverShim
- The driver shim is psoftshim.jar. The compiled Java Component Interface (CI) classes used by the PSA are contained in dirxmlcomps.jar. These files must be placed in the \lib subdirectory of either the local or Remote Loader components. (By default, local drivers are placed in Novell\\NDS\lib and remote drivers are placed in Novell\RemoteLoader\lib on a Win32 host.)
- If you are using data or transaction CIs other than those delivered with the PSA, the Java APIs for your CIs need to be built, compiled, and made into .jar files as specified in “Changing the Driver by Changing Data Schema Component Interface” on page 43. This is a standard PeopleSoft procedure for using Java CIs in external programs. The new JAR file should be placed in the same \lib directory as the driver shim.
- The driver no longer requires PeopleSoft client or external client library packages. The only component required from the appropriate PeopleTools release is the web/psjoa/psjoa.jar file, which contains the PeopleSoft Java middleware interfaces. The psjoa.jar file must be placed with the driver JAR files in the appropriate \lib subdirectory.

- There is a new CITester.class file available to test the connectivity to the PeopleSoft Application Server and the validity of the specified transaction and schema CIs. To use the program on Win32 platforms, place the psjoa.jar file in the PSUtils directory and execute the CITester.bat file that is also in that directory. Refer to "Testing Component Interfaces" on page 28 for more information.
- When upgrading the driver shim, it is not necessary to use the 5.0 driver's PSA files. If you want to reference them and use the upgraded driver functionality, make sure you do not install the new PSA on the same application server that contains your current PSA. Many of the PSA fields, records, components, etc. have the same names as those used in the 4.0 PSA. Installing the 5.0 PSA overwrites them.
- An upgrade to the driver policies is not required. The 5.0 driver functions seamlessly with 4.0 policies, except for the format of the DirXML® association.

  The 4.0.*x* drivers used the AssocID field as the association value. However, this format does not uniquely identify objects that can exist in two different applications. The 5.0 driver's association format uses a *schema name/ASSOC_ID* format (For example, DIRXML_SCHEMA01/P000001.) To ease the upgrade process in an existing deployment, the new driver delivers two XSLT templates that convert between the 5.0 driver association format and the 4.0.*x* format (the association format exists on previously synchronized objects in eDirectory™.) These templates are in the AssociationStyleTransform file. If you want this conversion, copy and paste the templates into the InputTransformSS policy.

5 Customizing the Driver

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

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 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”);
```

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.	NPSDriver1S transaction was caused by a Subscriber Channel event and will be processed by driver NPSDriver1. NPSDriver1P transaction was caused by a Publisher Channel event and will be 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.	
Collection Row Delete Field Key	The data type of the attribute.	
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”.	

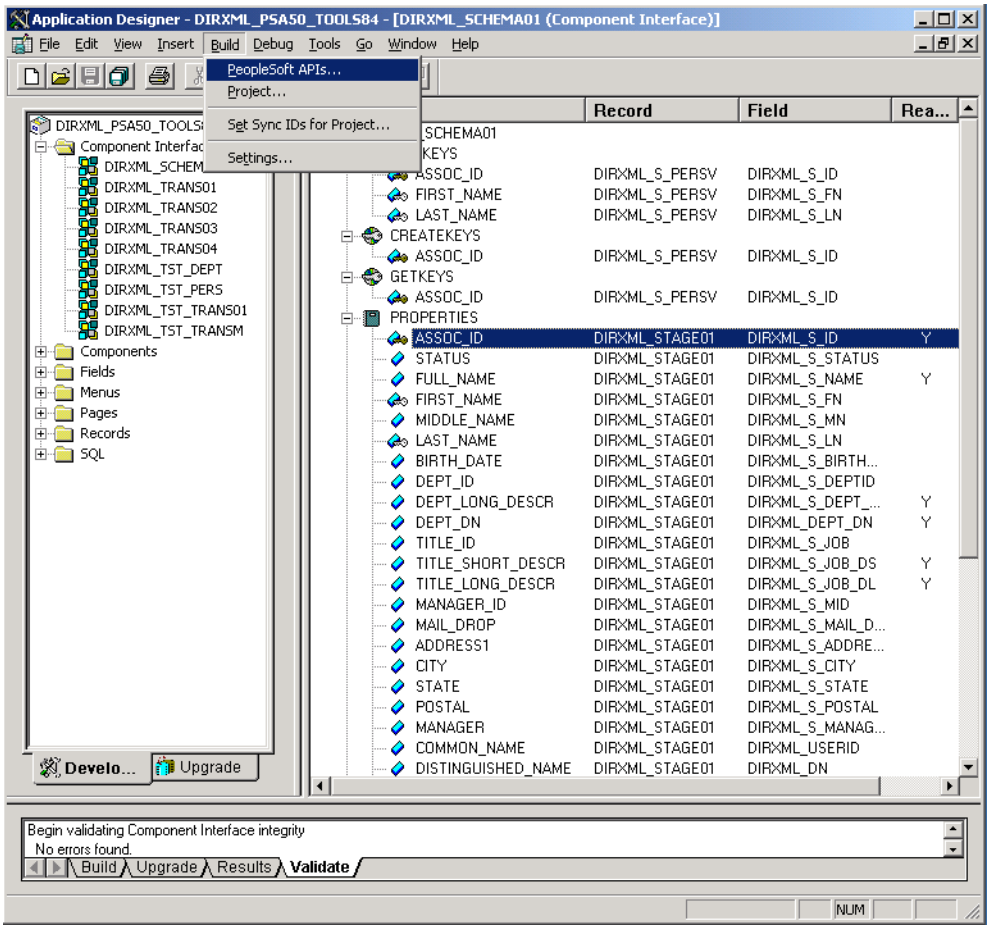
Changing the Driver by Changing Data Schema Component Interface

The driver is preconfigured to use the Component Interfaces 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.

Building the PeopleSoft Java Component Interface API

1. From the PeopleTools Application Designer, select the Schema Component Interface 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.

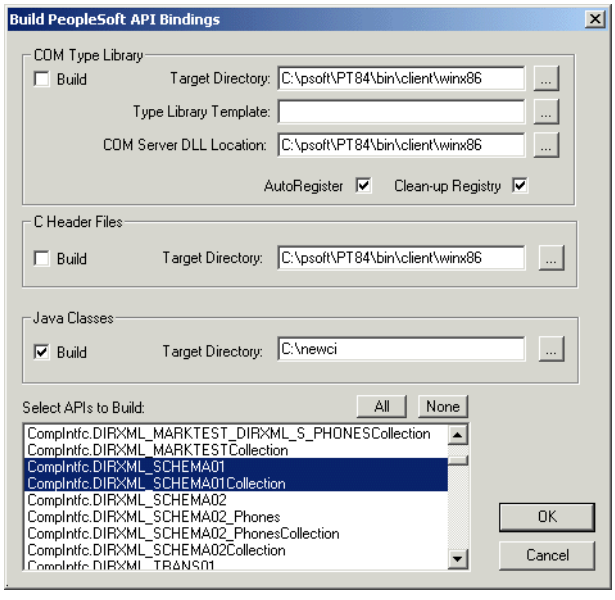
4 Select a target directory for the Java CI APIs (For our example, C:\newci).

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

6 Using the scroll areas, 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.

Compiling the Java CI API

Now that the APIs have been generated, they must be compiled. In the C:\newci target directory there is a path generated to the Java API files. 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

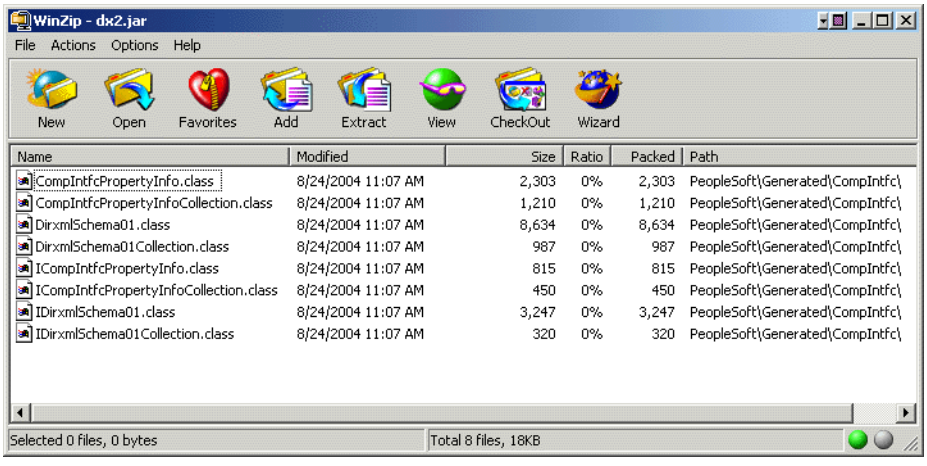
After a successful compile, there is a .class file for each .java file in the directory.

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 jar can be verified using WinZIP or other appropriate tool.



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 will also be 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.

Customizing the Driver by Modifying Driver Policies

This section contains information to help you understand how the driver's policies are implemented, as well as information about how you can modify these objects. Topics include the following:

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 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. 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.

Using the Schema Query to Refresh the PeopleSoft Schema CI

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

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 DirXML 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 eDirectory.

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 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 DirXML® 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 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.

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:

Attributes	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

Attributes	Attributes
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.

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 it is not be synchronized to eDirectory via the driver.

Publisher Object Policies

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

- Input Transformation Policy (page 48)
- Matching Policy (page 49)
- Create Policy (page 49)
- Placement Policy (page 49)
- Command Transformation Policy (page 50)

Policies contain templates that perform specific operations that can manipulate data, query either eDirectory 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 describes the operations of each policy or template. Because XML, DirXMLScript 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 not addressed in this section. For more information, refer to “Modifying the Driver Mapping Policy” on page 46.

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 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 eDirectory isManager attribute.

Matching Policy

The Matching policy is used by the DirXML engine to apply criteria to determine if a matching data object already exists in eDirectory. 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 DirXML 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 a 0 or 1 match. More than one policy can exist, and the DirXML 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 DirXMLScript policy that attempts to match eDirectory User objects containing the same value in the workforceID attribute (mapped from the DIRXML_SCHEMA01 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 and default password.

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 DirXMLScript 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 *Org Name*\Users\Active organizational unit. If the employeeStatus is I, the employee is placed in the *Org Name*\Users\InActive organizational unit. If the employeeStatus attribute is not present, the employee is placed in the *Org Name*\Users\InActive organizational unit.

Command Transformation Policy

The Command Transformation policy is the final transformation policy to be processed prior to submission of a Publisher document to eDirectory. 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 at all, 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 the event-id attribute of the document causing the add or move of the object. After Identity Manager applies the data to eDirectory, 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 the embedded CN and DN data 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. As will be seen, 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

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 event-id attribute 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 if 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, then Login Disabled is set to False. If the status is I, then Login Disabled is set to True.
- Adds or removes the Group Membership attribute based on employeeStatus. All active employees with the isManager attribute set to False are placed into a default Employee Group. All active employees with the isManager attribute and associated links on the group objects are cleared if the employeeStatus is set to I. If the isManager attribute is modified in the modify document, then the User's name is cleared from the Member attribute of his or her previous group.
- Adds the manager attribute to any other active User objects in the directory whose managerWorkforceID attribute specifies this new User.
- Adds the directReports attribute to any active User object in the directory 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.

match <delete> element

This template does the following:

- Transforms the <delete> into a <modify> document.
- Sets the Login Disabled attribute to True.
- Removes the Group Membership attribute from the User object. It also removes the User's name from the Member attribute of the current group.
- Removes the User's manager attribute.
- Removes the User's directReports attribute.
- Removes the manager attribute from any Users who were in the directReports list.
- Removes the User's name from the directReports attribute of his or her manager.
- 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 Active organizational unit to the InActive organizational unit.

buildAddEventID and buildDeleteEventID

These templates are part of the CN and DN write-back implementation. They are responsible for embedding the User object CN and DN attributes into the event-id attribute of the document.

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 eDirectory 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 eDirectory 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 eDirectory 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 eDirectory 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 eDirectory 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 eDirectory 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.

Subscriber Channel Objects

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

eDirectory sends filtered data modification events to PeopleSoft through the DirXML 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.

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 DirXML 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 eDirectory or any other authoritative application that might have written the data to eDirectory. 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 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 is configured to function in either a predominant Subscriber or Publisher mode.

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

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

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.

Modifying the Filter

A properly configured Subscriber filter promotes a secure environment and secures data sharing from eDirectory to PeopleSoft. Follow the steps in “Modifying the Filter” on page 54 to configured the filter as desired. 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.

Subscriber Object Policies

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

- “Event Transformation Policy” on page 55
- “Matching Policy” on page 55
- “Create Policy” on page 56
- “Output Transformation Policy” on page 56

Policies contain templates that perform specific operations that can manipulate data, query either eDirectory 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, DirXMLScript, 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 46.

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

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 CommonName and DistinguishedName 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 DistinguishedName 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 (CommonName, DistinguishedName, Internet Email Address, and Description).

Matching Policy

The Matching policy is used by the DirXML 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 eDirectory 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 eDirectory, and queries eDirectory 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 DirXML 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 a AssocID 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 DirXMLScript 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 DirXMLScript and 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 eDirectory, 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

As documented in the Publisher Channel Command Transformation policy, this template monitors all status document responses from eDirectory. If a status document with a Success value is received and it contains an event-id attribute with an embedded CN and DN value, 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.

manager flag data transformation

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

6 Troubleshooting the Driver

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

The Driver is Not Processing Available Transactions or is Processing Them Out of Order

- Set 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 "Configuring the Transaction Record SQL Date/Time Format" on page 26.
- If the records are in correct format, verify that Transaction date and time field values are correct and correspond to system date and time.

Error Trying to Obtain Data Record

The following are typical reasons for this error:

- The Data record identified in a Transaction record has been 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 "Testing Component Interfaces" on page 28.) Correct any errors that may exist.

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.

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.

Attributes Are Not Refreshed on the Data Schema Object

- Verify that the Component Interfaces are working correctly by using PeopleSoft Application Designer tool (see "Testing Component Interfaces" on page 28.)

Data Does Not Show Up in eDirectory 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.

Error: Check Application Server IP Address and Jolt Port Number

- Run the CITester utility to verify proper connection and authentication parameters are set. Refer to "Testing Component Interfaces" on page 28.

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.

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).

Transactions Are Not Placed in the PeopleSoft Queue

- Verify that PeopleCode is working properly.

Transactions Are Left in "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.

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 to 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.

Component Interface Relationships 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. Test all of the processes manually 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.

SQL Error During Save of "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.