Novell DirXML® Driver for PeopleSoft*

4.0.2

www.novell.com

IMPLEMENTATION GUIDE

January 15, 2004

Novell®

Legal Notices

Novell, Inc.
1800 South Novell Place
Provo, UT 84606
U.S.A.

www.novell.com

DirXML Driver for PeopleSoft Implementation Guide
January 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

ConsoleOne is a registered trademark of Novell, Inc., in the United States and other countries.
DirXML is a registered trademark of Novell, Inc., in the United States and other countries.
eDirectory is a trademark of Novell, Inc.
NetWare is a registered trademark of Novell, Inc., in the United States and other countries.
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 7

1 Introducing the DirXML Driver 4.0 for PeopleSoft 9

Driver Components . . . 9
- How the Driver Works . . . 10
- Configuring Your PeopleSoft Environment . . . 11
- Configuring Your DirXML System . . . 11

Prerequisites . . . 12
New Features . . . 12
- Driver Features . . . 12
- DirXML 2.0 Features . . . 13

2 Configuring Your PeopleSoft Environment 15

Using the PeopleSoft Service Agent . . . 15
- The Component Interface Infrastructure for DirXML . . . 16
- The Sample Application . . . 16
- Installing the PeopleSoft External API . . . 16

Installing the PSA Sample Project . . . 17
- Extracting the PSA Files . . . 17
- Importing the PSA Project into the PeopleSoft Database . . . 17
- Building Project Record Definitions . . . 18
- Applying Security to the PSA . . . 18
- Testing Sample PeopleSoft Applications . . . 19

Component Interfaces . . . 21
- Accessing Transactions and Data through Component Interfaces . . . 21

Configuring PeopleCode to Trigger Transactions . . . 23
- Transaction Component . . . 24
- Schema Component . . . 25
- Testing Component Interfaces . . . 25

3 Installing and Configuring the Driver 27

Configuration Information . . . 27
Importing the Driver Configuration . . . 29
Activating the Driver . . . 29

4 Customizing the Driver 31

Customizing the PSA by Triggering Transactions . . . 31
- Triggering Transactions . . . 31

Customizing the Driver by Modifying Driver Policies . . . 33
- Modifying the Driver Mapping Policy . . . 33
- Using the Schema Query to Refresh PeopleSoft Schema CI . . . 34

Publisher Channel Objects . . . 34
- Understanding the Publisher Filter . . . 34
- Publisher Filter Attributes . . . 34
- Securing the Data . . . 35

Publisher Object Policies . . . 35

Input Transformation Policy . . . 36
Matching Policy . . . 37
Create Policy . . . 37
Placement Policy . . . 37
Command Transformation Policy . . . 37
Subscriber Channel Objects . . . 41
Understanding the Subscriber Filter . . . 41
Securing the Data . . . 42
Modifying the Filter . . . 42
Subscriber Object Policies . . . 42
Matching Policy . . . 43
Command Transformation Policy . . . 43
Output Transformation Policy . . . 44

5 Troubleshooting the Driver 45

Resolving Errors . . . 45
The Driver Will Not Start . . . 45
Attributes Do Not Get Refreshed on the Data Map Object . . . 45
Data Does Not Show Up in eDirectory on the Publisher Channel . . . 45
Error: Check Application Server IP Address and Jolt Port Number . . . 45
Data Does Not Update in PeopleSoft on the Subscriber Channel . . . 45
No Transactions Are Coming across the Publisher Channel . . . 46
Transactions Are Not Placed in the PeopleSoft Queue . . . 46
No Data Is Returned When Running the Component Interface Test Program . . . 46
Transactions Are Left in "Process" State and Not Processed . . . 46
Receiving Errors on the Publisher Channel When Processing a Transaction . . . 46
Component Interface Relationships Not Functioning . . . 47

About This Guide

The DirXML® Driver 4.0 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:

- Chapter 1, "Introducing the DirXML Driver 4.0 for PeopleSoft," on page 9
- Chapter 2, "Configuring Your PeopleSoft Environment," on page 15
- Chapter 4, "Customizing the Driver," on page 31
- Chapter 5, "Troubleshooting the Driver," on page 45

Additional Documentation

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

Documentation Updates

For the most recent version of this document, see the DirXML 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 DirXML. To contact us, send e-mail to proddoc@novell.com.

1 Introducing the DirXML Driver 4.0 for PeopleSoft

PeopleSoft* applications are some of the most popular Enterprise Resource Planning (ERP) systems available. The DirXML® Driver 4.0 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 DirXML 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 DirXML.

Because DirXML 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.

Driver Components

The driver includes the following components:

- Driver shim

  The driver shim (NPS8SHIM.DLL) enables communication between PeopleSoft and eDirectory. It reports object change events it receives from PeopleSoft to the DirXML engine and vice-versa.

- PeopleSoft Service Agent

  The PeopleSoft Service Agent (PSA) is a collection of objects that can be deployed into a PeopleSoft database application. The PSA defines how and what data will be available from the PeopleSoft database for synchronization with eDirectory. The PSA also defines how PeopleSoft data is accessed.

  The PSA contains a collection of PeopleTools objects and must be installed on the PeopleSoft server database. You can use this collection of objects to:

  - Configure how you want to trigger transactions when events occur in PeopleSoft.
  - Determine how the driver accesses PeopleSoft data.

  This PSA project is not intended for use in a production environment, but as a template to get you started in your DirXML implementation. Based on your business processes, you should configure the PSA project accordingly. For more information, refer to "Using the PeopleSoft Service Agent" on page 15.

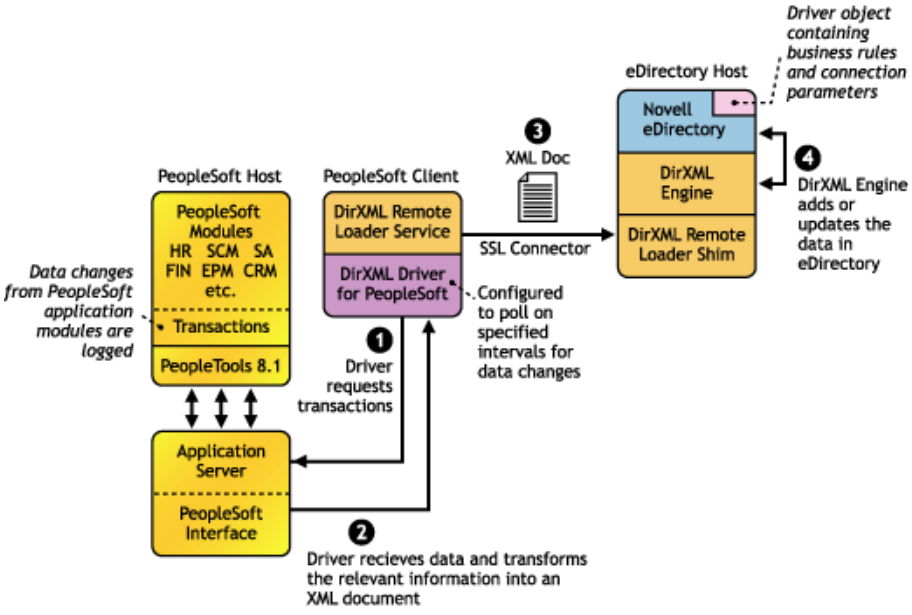
- Driver Configuration File

  PEOPLESOFT40.XML is a driver configuration you import through Novell iManager. It enables DirXML to handle data object creation, matching, and synchronization between PeopleSoft and eDirectory.

This file is not intended for use in a production environment, but as a template to get you started in your DirXML implementation. Based on your business processes, you should configure the file accordingly. For more information, refer to "Installing and Configuring the Driver" on page 27 and "Customizing the Driver by Modifying Driver Policies" on page 33.

How the Driver Works

The Publisher Channel

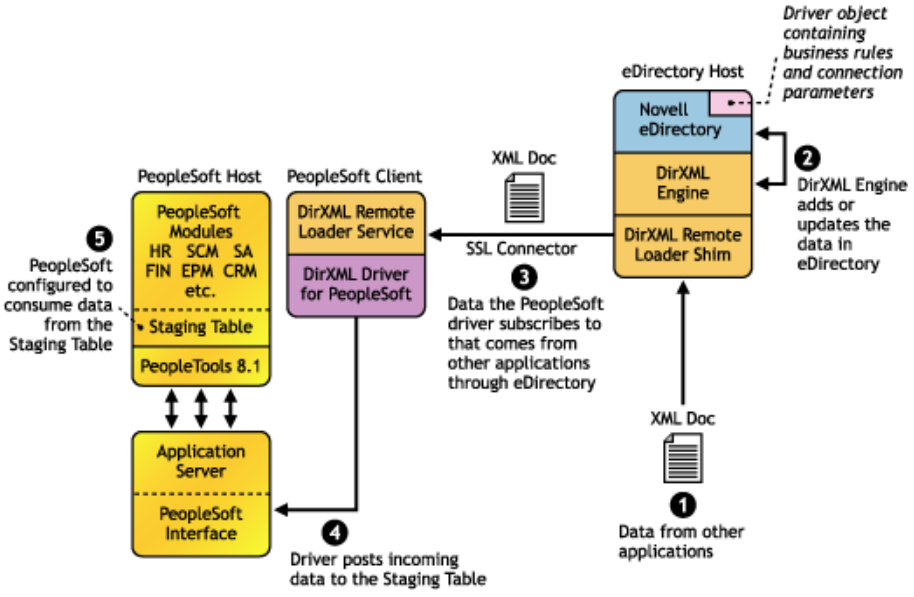


The Publisher channel synchronizes data from PeopleSoft to eDirectory. As events occur within PeopleSoft, transactions are placed into a transaction table. These transactions are usually written to the table through PeopleCode (you can use other methods such as Batch SQL, COBOL, SQR, and so forth.) Component 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 need to be processed based on their available status and processes only those transactions that have 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. Through 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 DirXML System

The driver interacts with PeopleSoft at the PeopleTools level. By using object definitions within the PeopleSoft modules, along with a collection of preconfigured objects, you enable PeopleSoft so that you can do the following:

- Create directory objects and initial passwords as new data is entered into PeopleSoft.
- Synchronize data bidirectionally between a PeopleSoft application and eDirectory.
- Delete or move eDirectory objects based on some event that occurs within PeopleSoft. eDirectory can manage changes through using record associations and unique identifiers.

- Maintain PeopleSoft authority over data against the object and its relevancy within the enterprise.
- Establish Group, Roles, or Other relationships with other objects in eDirectory based on relationships defined within the PeopleSoft application.
- Provide notification based on various events occurring or requirement approvals.
- Adhere to enterprise business processes and policies.
- Pass data to other systems that are integrated into an Enterprise Provisioning solution.
- Expose trapped events to eDirectory.
- Collect and process the exposed data.
- Expose the data through a PeopleTools interface.

After the driver is configured, you can add new data and the objects will be created in eDirectory. Similarly, when a record is deactivated within the PeopleSoft application, the associated directory object can be deleted or disabled, and it can also be moved. You can synchronize data (people data, department records, financial accounts, and so forth) to the directory based on the PeopleSoft configuration and the driver's configuration. For more information on configuring your DirXML system, refer to Chapter 4, "Customizing the Driver," on page 31.

Prerequisites

- Novell Nsure™ Identity Manager 2
- Windows* NT* 4.0 with service pack 6a or later, or Windows 2000 Professional or Server with the latest service pack.
- The appropriate version of the PeopleTools client and Application Server

PeopleSoft Platform	PeopleTools Client Version	PeopleSoft Application Server
PeopleSoft 8.1	PeopleTools 8.17 or later	PeopleTools 8.17 or later
PeopleSoft 8.4	PeopleTools 8.4 or later	PeopleTools 8.4 or later

- Installation of the PeopleSoft External API (PSEXTAPI) software.

New Features

Driver Features

Architecture Changes

To accommodate changes to the PeopleTools interface, this release of driver contains major architecture changes. The driver is less intrusive to the PeopleSoft application than previous versions. The architectural changes include:

- An enhanced PeopleSoft Service Agent (PSA).
- The Event Server is no longer used.
- PeopleSoft Workflow configuration is no longer needed.

Express Implementation

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 your 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.

Parallel Testing

This driver can coexist with prior releases, 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.

DirXML 2.0 Features

DirXML 2.0 includes the following new features. For more information, refer to the *Nsure Identity Manager 2 Administration Guide* (http://www.novell.com/documentation/lg/dirxml20/admin/data/alxnk27.html).

Password Management

The new password management framework includes the following benefits:

- New Password Policies let you create rules for passwords and assign them to users, containers, or the whole eDirectory tree. You can enable Universal Password, which lets you enforce detailed criteria for passwords and allows for special characters.
- Password Synchronization 2.0 is now cross-platform, and it lets you enforce your Password Policies across connected systems. New notification templates let you automatically send messages to users about their password synchronization status.
- Using Password Policies, you can also provide Forgotten Password Self-Service and Reset Password Self-Service to your users. These new features can help you reduce help desk calls. Notification templates are also included for automatically sending forgotten password and password hint messages to users.

Policy Builder Interface and DirXML Script for Creating Policies

For the most common tasks, you can now use the new Policy Builder interface to create policies for your driver without writing XSLT code. The Policy Builder helps you set up policies using the new DirXML Script.

Role-Based Entitlements

For many drivers, Role-Based Entitlements is an option in the sample driver configuration that you can choose when importing the driver.

Role-Based Entitlements let you grant entitlements on connected systems to a group of Novell® eDirectory™ users. Using Entitlement Policies, you can streamline management of business policies and reduce the need to configure your DirXML drivers.

Novell Nsure Audit

Novell Nsure™ Audit is a centralized, cross-platform auditing service. It collects event data from multiple applications across multiple platforms and writes the data to a single, non-repudiable data store. Nsure Audit is also capable of creating filtered data stores. Based on criteria you define, Nsure Audit captures specific types of events and writes those events to secondary data stores.

Global Configuration Values

Global configuration values (GCVs) are new settings that are similar to driver parameters. Global configuration values can be specified for a driver set as well as an individual driver. If a driver does not have a value for a particular GCV, the driver inherits the value for that GCV from the driver set.

GCVs allow you to specify settings for new DirXML features such as Password Synchronization, as well as settings that are specific to the function of an individual driver configuration. Some GCVs are provided with the drivers, but you can also add your own. You can refer to these values in a policy to help you customize your driver configuration.

Driver Heartbeat

The DirXML engine now accepts driver heartbeat documents from drivers, and drivers can be configured to send them.

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

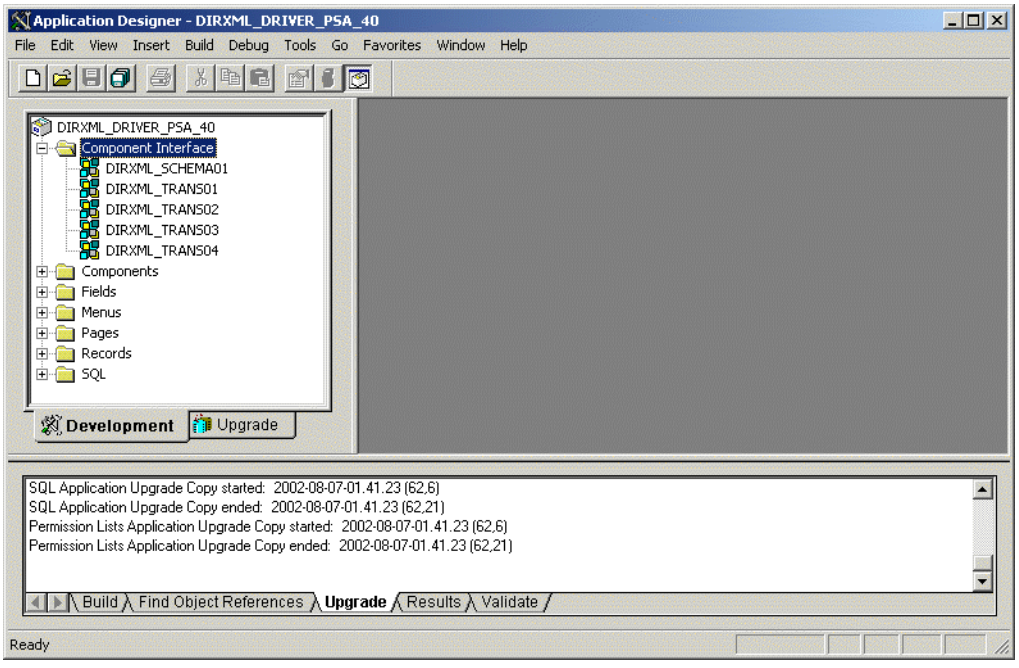
The PeopleSoft Service Agent (PSA) is a set of PeopleTools objects that enables you to capture transactions from PeopleSoft and expose data to the driver. Through the PSA, you can quickly implement a DirXML® solution without being intrusive within the PeopleSoft system. The delivered PSA does not interrupt or effect existing PeopleSoft objects or processes.

In this section, you will find installation and configuration information for the two main PSA components: "The Component Interface Infrastructure for DirXML" 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.

You should also ensure that the PeopleSoft External API is installed. For more information, refer to "Installing the PeopleSoft External API" on page 16.

The Component Interface Infrastructure for DirXML



You use the Component Interface infrastructure and PeopleCode function calls to specify how transactions are processed (for example, new hire, termination, department transfer, and so forth.) Component Interfaces replace Message Agent Definitions. For more information, refer to "Configuring PeopleCode to Trigger Transactions" on page 23.

The Sample Application

This self-contained project has a sample application that you can install on your PeopleSoft system for configuration and testing purposes.

Depending on your business requirements, you should configure internal processes to trigger events into transaction tables, either by replicating provided PeopleCode or by merging the components within your PeopleSoft environment. For more information, refer to "Installing the PSA Sample Project" on page 17

Installing the PeopleSoft External API

Before configuring the driver, you need to install the PeopleSoft External Application Interface software (PSEXTAPI) provided by PeopleSoft if you are using PeopleSoft 8.1*x* systems. If using PeopleSoft 8.4*x*, you do not need to install the PSEXTAPI.

The PSEXTAPI requires a Java Virtual Machine because the calls to the application server are made through Jolt. The JVM* should be installed on the Operating System where the driver shim (NPS8Shim.dll) runs. If you don't have the JVM, you can download it from the Microsoft* Download Center (http://www.microsoft.com/downloads) (msjavx86.exe); or you can use the JVM supplied with Sun's JDK* 1.2.2. For additional information, please refer to your PeopleSoft documentation.

To install the PSEXTAPI:

1. Run setup.exe from {pshome}\bin\client\winx86_api\.

2. Change the environment path to include the directory {pshome}\bin\client\winx86_api\.
3. Reboot your workstation.

Verifying PeopleSoft's Connection to the JVM: You should verify that PeopleSoft can connect successfully to the JVM. Run jnitest.exe from the PeopleSoft External API directory {pshome}\bin\client\winx86_api\.

Installing the PSA Sample Project

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

1. "Extracting 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. "Testing Sample PeopleSoft Applications" on page 19

Extracting the PSA Files

If you didn't install the PSA during the initial driver installation, locate the product CD or download, go to the PeopleSoft application server, and run install.exe. After the PSA files are on the PeopleSoft application server, you should:

1. Locate the PSA folder and run DIRXML DRIVER 4_0 psa.exe to extract the PSA files.
2. Click Next, select the destination directory, then click Next to begin extracting the PSA files.
3. Click Finish to close the extraction process.

Importing the PSA Project into the PeopleSoft Database

The PSA project comes as a self-extracting project. With PeopleSoft version 8, projects are now delivered in a cache directory structure, which is similar to the cache structure found within PeopleSoft. With previous versions of PeopleSoft you needed to use the Data Mover script and compare, but this process is no longer necessary.

NOTE: Any references to specific paths represent the defaults indicated during the installation procedures. Apply the necessary changes if applicable.

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 and select the PSA project directory: c:\psa\psa-psa8\.
4. Click Copy.
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\psa-psa8\.
4. Click Copy.
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 then build project views.

For PeopleSoft 8.1 and 8.4

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.

   NOTE: 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 4`** and click OK.
4. In the Description field, type **`DirXML Administration 4`**.
5. Click the Permission Lists tab, then click the drop-down arrow.
6. For the Permission List value, type **`DIRXML`** and 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 down 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 permission 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 for the role, click DirXML Administrator to add it, then click Save.

Testing Sample PeopleSoft Applications

You can test to ensure transactions are created by entering a new person using the PeopleSoft DirXML sample application. This example uses Departments, so you will 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 > 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 Click the ADD button and a valid ID will be entered by default.

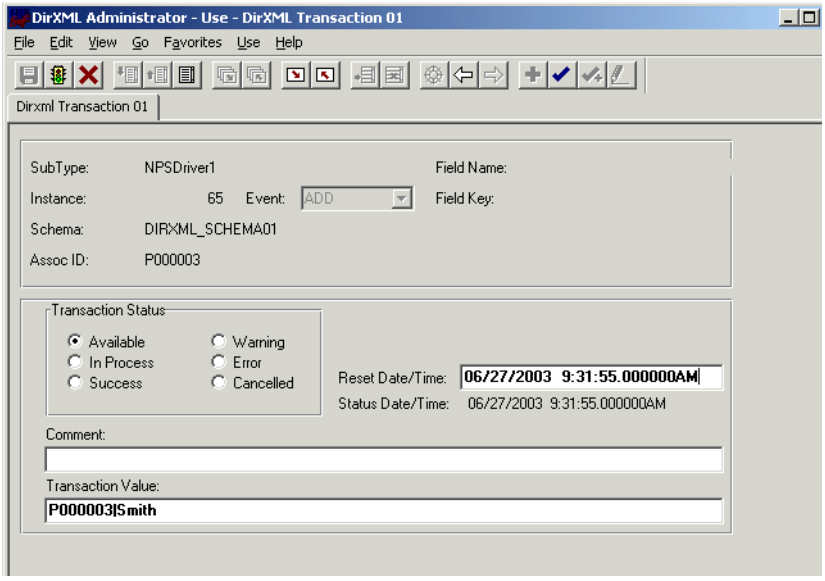
7 Type data into the various fields for this ID and click Save.

Asterisks represent required fields.

8 Validate that an ADD transaction was created by selecting Use > DirXML TRANSACTION01.

9 Click the Search button.

10 Verify that the transaction was created and select the transaction.

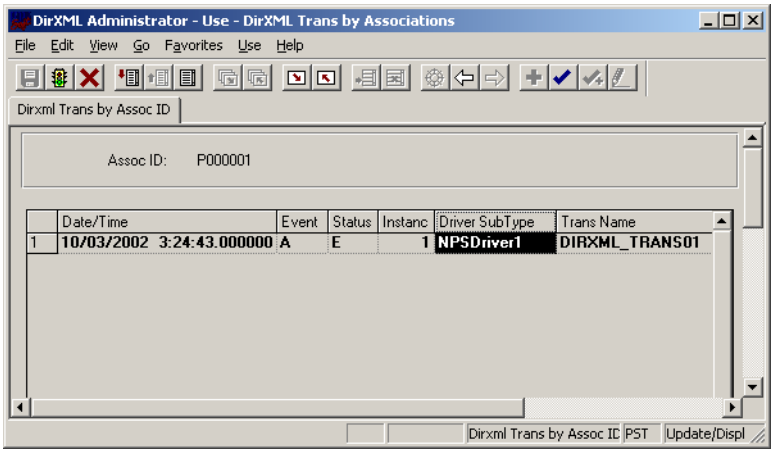


11 Click Use > DirXML Schema 01 > DirXML Schema.

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

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

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



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

16 Verify that other Transaction table applications work by clicking Use > DirXML Transaction 02 (03, 04, etc.)

NOTE: You can create additional transaction tables (Transaction 02, Transaction 03, and so forth.) However, the delivered sample application is not configured to use these additional transaction tables.

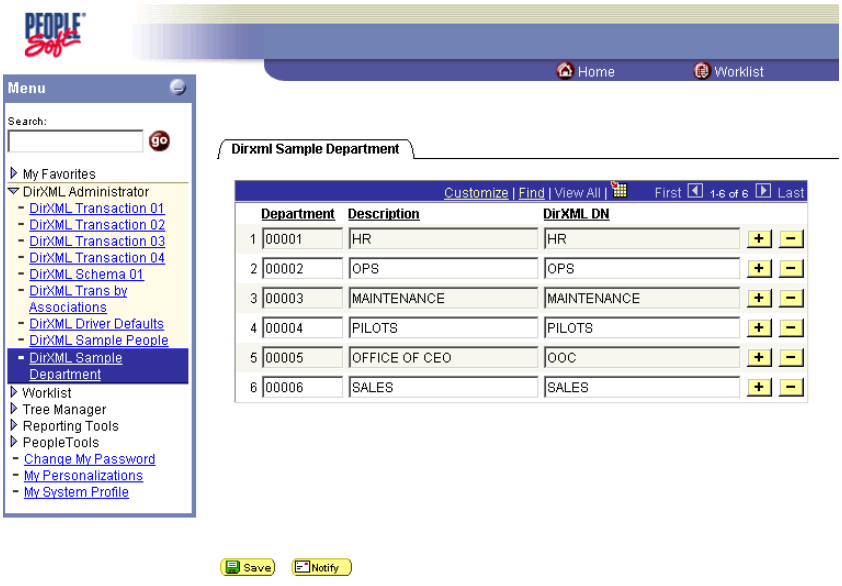
For PeopleSoft 8.4

1 Log in to the PeopleSoft portal.

2 Click Structure & Content > DirXML_Administrator.

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 Specify a sample user, then click Save and ensure that the user's data appears in all of the Transaction tables.

Component Interfaces

The following section contains information regarding:

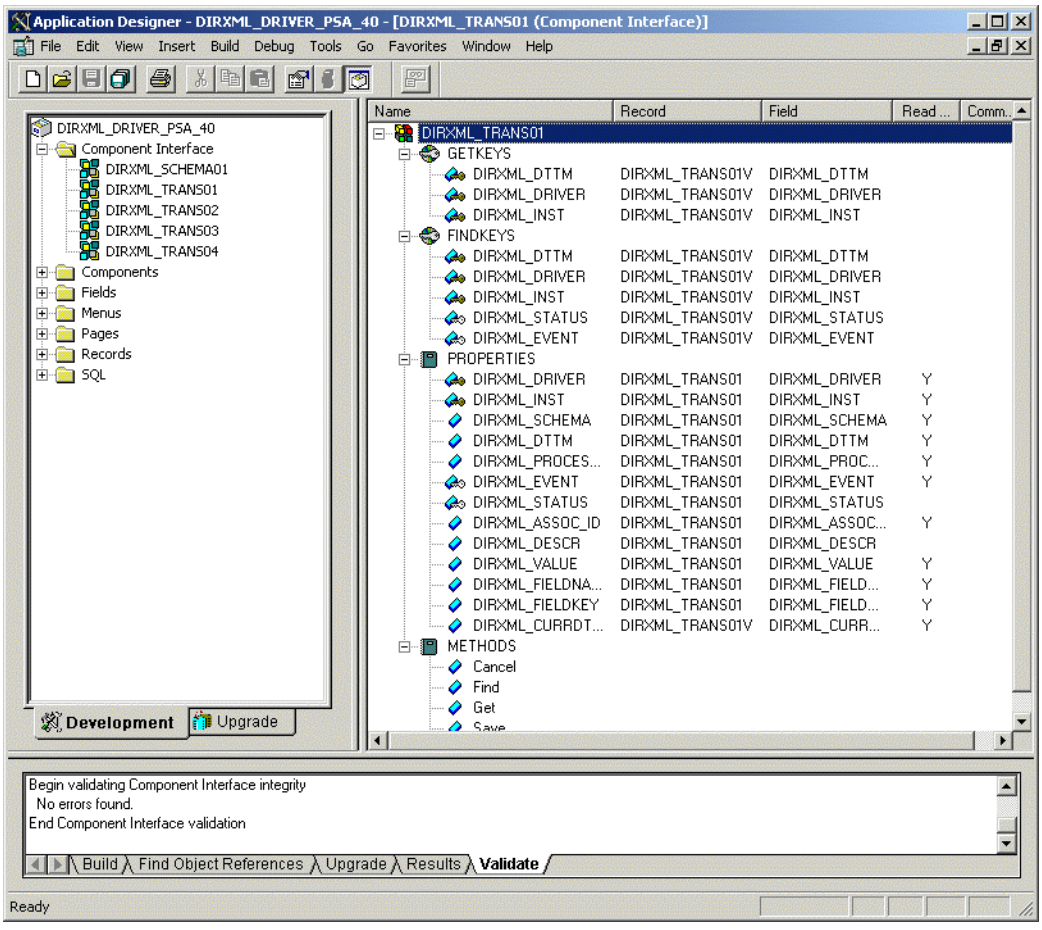
- "Accessing Transactions and Data through Component Interfaces" on page 21
- "Configuring PeopleCode to Trigger Transactions" on page 23

Accessing Transactions and Data through Component Interfaces

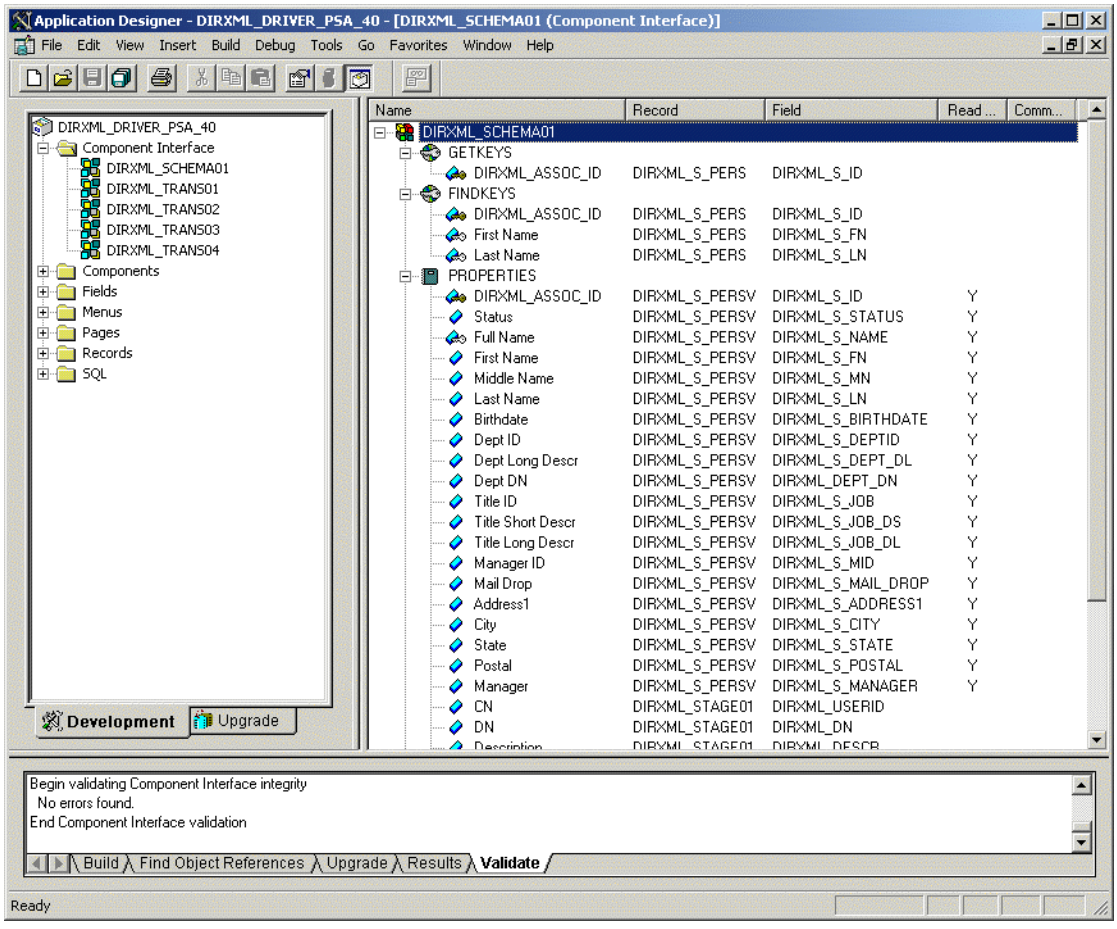
The driver accesses transactions waiting to be processed from the transaction table via the Component Interface object that is defined within PeopleTools. Each Component Interface maps to a particular component. Components are built in order to access transaction tables and schema data. Schema objects 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 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 will then process all transactions available through the CI. If the parameter exists, the driver limits processing to the transaction type specified in the subtype string.

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

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

1. Spaces are no longer allowed in Component Interface field names.
2. There are now case sensitivity issues in the CI API. Field names and field name values no longer align due to case-sensitive sorting. For example, a field named "CN" will be 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 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 subtype. It will process only those transactions that have a transaction date/time less than or

equal to the current date/time value with a status of available. 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, please refer to PeopleSoft for more information.

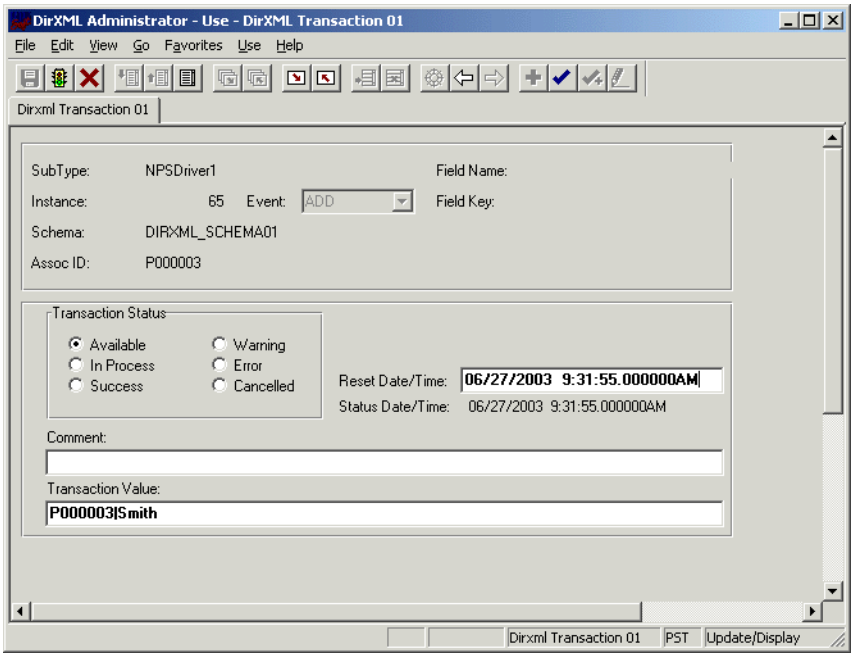
The driver requires the Transaction and Schema Component to process transactions. For more information on calling the PeopleSoft function that creates transaction records, please refer to “Triggering Transactions” on page 31.

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 will use to create either 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, you want to process transactions for employees with the Associated ID=EMPLID and key value=0001.

The driver accesses the Schema Component with a key value of 0001. It retrieves all of the component elements that have been defined for that employee on that specific component. The driver then converts the data collection one transaction at a time 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 object acts as its own class of objects. 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, included as part of the download, ensures that your PeopleTools client software is installed and configured properly on the computer hosting the driver.

For more information, refer to the Component Tester folder and its accompanying documentation.

3 Installing and Configuring the Driver

This section contains information to help you:

- Review "Configuration Information" on page 27
- Complete procedures for "Importing the Driver Configuration" on page 29.
- Understand the process of "Activating the Driver" on page 29.

Configuration Information

As you import the driver configuration file, you will be prompted for the following information.

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.
Active Managers Group	The name of the Group Object to which Active "Manager" users are added.
Event Server Host Name and Port	The host name or IP address and port number of the computer where the PeopleSoft Event Server.
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 this driver uses for authentication to PeopleSoft.
PeopleTools Client Library Path	The driver uses the PeopleTools client libraries to establish databases connectivity. This parameter must specify the file system path to the location of the appropriate psapiadapter.dll client library.

Parameter Name	Parameter Description
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 8.4x, you must select a Remote Installation. Local implementations are not supported.) If Local is selected, you can skip the remaining parameters.
Driver Shim Path	(Local Installations Only) Enter the full path of the NPS8Shim.dll driver shim to load.
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 DirXML 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 DirXML 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
Transaction CI Name	This parameter 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	This parameter identifies which transactions in the transaction CI will 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. That is to say, 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 may 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

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.

The following instructions explain how to create the driver and import the driver's configuration.

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 PeopleSoft40.xml.

   The driver configuration files are installed on the Web server when you install DirXML. During the import, you will be prompted for the driver's parameters and other information. Refer to "Configuration Information" on page 27 for more information.
4. Specify the driver's parameters, then click OK to import the driver.

   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.
5. Review the driver objects in the Summary screen, and then click Finish.

Activating the Driver

DirXML and DirXML drivers must be activated within 90 days of installation, or they will not run. At any time during the 90 days, or afterward, you can choose to activate DirXML products to a fully licensed state.

To activate your driver, you should:

- Purchase DirXML licenses
- Generate a Product Activation Request
- Submit the Product Activation Request
- Install the Product Activation Credential received from Novell

For more information about completing these tasks, refer to Activating Your DirXML Product (http://www.novell.com/documentation/lg/dirxml20/index.html).

4 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

Triggering Transactions

Transactions are triggered using PeopleCode within PeopleSoft and are written to transaction tables. If you want to allow processing by multiple drivers, you can categorize the processing of transactions by using a different driver subtype value.

The PeopleSoft administrator or consultant has the ability to trigger transactions under any circumstance associated with a pre-defined PeopleSoft process, or he or she can create transactions via a batch mechanism.

You can use the following information to create your own transaction triggers, also known as PeopleCode function calls:

A PeopleCode function call would be as follows:

```
DirXML_Trans(Transaction Table,
             Transaction Sub Type,
             Transaction Schema,
             Transaction Event,
             Transaction Association ID,
             Transaction Date Time,
             Transaction Value,
             Row Delete Field Name,
             Row Delete Field Key);
```

Using the above format and sample data, a real function call would look like this:

```
DirXML_Trans("DIRXML_TRANS01",
             "NPSDriver1",
             DIRXML_SCHEMA01,
             "A",
             DIRXML_PERS_VW.EMPLID,
             %Datetime,
             "",
             "",
             "");
```

Function Call Parameter Definitions

Parameter	Description	Default Value
Transaction Table	The name of the table where transactions are written to. This table is built within PeopleTools and the field elements should be consistent with the delivered DIRXML_TRANS01 table.	DIRXML_TRANS01
Transaction Sub Type	The name used to identify the type of transaction. The driver uses this parameter to process the types of transactions you specify.	NPSDriver1
Transaction Schema	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
Transaction Event	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
Transaction Association ID	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.	DIRXML_ASSOC_ID
Transaction Date Time	The date/time element used to determine when the transaction is processed.	%Datetime
Transaction Value	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.	DIRXML_S_ID | "|" | DIRXML_S_LN
Row Delete Field Name	The field name of the scroll level attribute.	PHONES
Row Delete Field Key	The data type of the attribute.	BUSN

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:

- "Publisher Channel Objects" on page 34
- "Publisher Object Policies" on page 35
- "Subscriber Channel Objects" on page 41
- "Subscriber Object Policies" on page 42

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 default, preconfigured 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 message definition. If you use the DirXML® 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 PeopleSoft Schema CI

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 will be necessary to refresh the PeopleSoft application schema definition.

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 a DirXML 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 specifies the object classes and accompanying attributes that will be 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 will allow 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 will remove this data because the 'Telephone Number' attribute is not in the filter. It must be noted that other attribute values may 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 wish to synchronize.
- Attributes on those objects you wish 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.

The Publisher filter specifies a set of data that will be considered as Authoritative from the PeopleSoft database. Based on business logic, there may be multiple authoritative sources of specific data (such as another DirXML driver or eDirectory itself). The configuration of the filters in your environment will determine 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 will be 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 DIRXML_SCHEMA01 CI also defines a level-1 scroll element named PHONES. The PHONES scroll contains multiple elements, each of which contains two fields that identify telephone numbers and the type of the number. Since these elements cannot be directly mapped to eDirectory attributes they cannot be in the Mapping policy. However, with DirXML it is possible to create a policy that can convert the PHONES elements into single-valued eDirectory attributes.

The eDirectory attributes can be listed in the filter to allow them to be passed to eDirectory. The Input Transformation policy of the Publisher channel performs this operation.

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

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

The homePhone, mobile, and pager attributes are also in the Subscriber filter. This implies that these attributes have dual authority or a peer-to-peer relationship between eDirectory and PeopleSoft.

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 will not leave the PeopleSoft application. The second method guarantees that it will not be synchronized to eDirectory via the DirXML driver.

Publisher Object Policies

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

- Input Transformation Policy (page 36)
- Matching Policy (page 37)
- Create Policy (page 37)
- Placement Policy (page 37)
- Command Transformation Policy (page 37)

Policies contain templates that perform specific operations that may 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 each policy or template. Because XML 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 will not be addressed in this section. For more information, refer to "Modifying the Driver Mapping Policy" on page 33.

Input Transformation Policy

The Input Transformation policy is implemented as a policy by default 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 may 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 structured, multi-valued attributes in PeopleSoft being converted to string attributes in eDirectory.

The following templates exist in the default Input Transformation policy. In addition to the listed templates, all DirXML policies contain identity-transform templates that allow copying of XML attributes and elements that will be passed through unmodified. Multiple instances of each listed template exist for each type of document that may be received by the policy (query response, add, and modify):

Manager Flag Data transformation Template

These templates convert 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.

Phone Number Data Transaction Templates

These templates convert the structured format values of the PeopleSoft PHONES scroll data elements into the appropriate attributes and string formats in eDirectory. Based on the Phone Type component of PHONES, the Phone Number component will be written to an appropriate string formatted eDirectory attribute.

The following mappings are provided:

Phone Type	eDirectory Attribute
BUSN	Telephone Number
HOME	homePhone
CELL	mobile
PGR	pager

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 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 will apply 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 match on eDirectory User objects containing the same value in the workforceID attribute. (Mapped from the DIRXML_SCHEMA01 DIRXML_ASSOC_ID 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.

Placement Policy

The Placement policy defines where an object will be placed in the eDirectory tree when the object is created. This placement may be determined based on the presence (or absence) of attributes, particular values of attributes, etc. Placement may 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 set of XML policies that do placement of User objects based on the employeeStatus attribute. In 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. This policy is new with DirXML 1.1 and provides a great deal of new functionality. To demonstrate this functionality, the default PeopleSoft Driver configuration implements an unusual implementation of business logic that demonstrates the flexibility and power of DirXML.

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 DirXML 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 policies 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 DirXML policies contain identity-transform templates that allow the copying of XML attributes and elements that will be 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 found the user's manager is found and the manager's employeeStatus attribute is set to A, the template will set the manager and managerWorkforceID attribute on the User.
- Sets the Login Disabled attribute based on employeeStatus. If status is A, Login Disabled is set to False. If 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 who 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 User's 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 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 Login Disabled attribute to true.
- Removes the Group Membership attribute from the User object. It will also remove 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 that 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 DirXML association attribute from a specified User object in eDirectory. The association value represents the unique ID of the object in the PeopleSoft application.

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 that 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 that have the specified manager-User object ID in the managerWorkforceI' 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 that 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 that 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 that 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 DirXML 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 specifies the object classes and accompanying attributes that will be 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 will allow changes to only these attributes to be passed to the DirXML engine. If a Telephone Number attribute is modified, the Subscriber filter will remove 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.

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

Subscriber Filter Attributes

By default, PeopleSoft applications are considered to be highly authoritative. Therefore most of the field names in the Subscriber filter will be data elements that are generally not assigned by a PeopleSoft application. As noted earlier, the names of attributes in the filter are eDirectory attribute names that have been mapped via the Mapping policy. As noted in the Publisher Filter documentation, the default PeopleSoft application, DIRXML_SCHEMA01, also defines a level-1 scroll element named PHONES. The PHONES scroll contains multiple elements, each of which contains two fields that identify telephone numbers and the type of the number. Because these elements cannot be directly mapped to eDirectory attributes they cannot be in the Mapping policy. However, with DirXML it is possible to create a policy that can convert the single-valued eDirectory attributes into unique instances of the PHONES scroll element. The eDirectory attributes can be listed in the filter to allow them to be passed to the DirXML engine. The Output Transformation policy of the Subscriber channel performs the appropriate data transformations.

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

Attributes	Attributes
CN	mobile
Description	pager
homePhone	workforceID
Internet EMail Address	

The CN, Description, and Internet EMail Address attributes are unique to the Subscriber filter. The homePhone, mobile, and pager attributes are also in the Publisher filter. This implies that these attributes have dual authority or a peer-to-peer relationship between eDirectory and PeopleSoft. The workforceID attribute is in the filter for the purpose of policies processing and is not synchronized back to PeopleSoft.

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 the Modifying a DirXML Publication or Subscription Filter operation 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:

- Schema Mapping policy
- Matching policy
- Command Transformation policy
- Output Transformation policy

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 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 will not be addressed in this section. For information on the Schema Mapping policy, refer to "Modifying the Driver Mapping Policy" on page 33.

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 will query PeopleSoft for all attributes in the Publisher filter and apply them to eDirectory, and query eDirectory for all attributes in the Subscriber filter and apply them to the PeopleSoft application. The process is finalized when an association value is written on the eDirectory User object. Because there is no Create policy on the Subscriber channel, if no match is found the Add event is discarded.

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 will apply 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 DIRXML_ASSOC_ID attribute that matches the eDirectory User object's workforceID attribute. The policy is defined in eDirectory class and attribute names since schema mapping has not yet been applied.

Command 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 policies processing to be programmed in one location. As will be seen, the bulk of the business logic transformations in the Subscriber channel are implemented in this policy.

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

match <modify> element

This template checks for an association value on all User Modify elements. Modify documents for unassociated User objects are discarded.

match <rename> element

This template converts eDirectory Rename events on associated User objects into Modify documents containing the new User CN and DN attributes.

match <move> element

This template converts eDirectory move events on associated User objects into Modify documents containing the new DN attribute.

match <delete> element

This template converts eDirectory Delete events on associated User objects into Modify documents that will remove the values of the CN and DN. The values for all attributes identified as unique to the Subscriber filter, Description, and Internet EMail Address, are also removed.

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

Output Transformation Policy

The Output Transformation policy is implemented as a 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-valued eDirectory attributes into structured, multi-valued scroll elements in PeopleSoft.

The following templates exist in the default Output Transformation policy. In addition to the listed templates, all DirXML policies contain identity-transform templates that allow the copying of XML attributes and elements that will be 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 will hold the status document and issue 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.

Phone number data transformation

These templates convert the single-valued telephone number attributes into structured format values of the PeopleSoft PHONES scroll data elements. Based on the eDirectory attribute, a different Phone Type component of PHONES element and the Phone Number component will be written to the PeopleSoft application. The following mappings are provided:

Phone Type	eDirectory Attribute
BUSN	Telephone Number
HOME	homePhone
CELL	mobile
PGR	pager

Phone number query transformation templates

These templates convert a query for any eDirectory telephone number attribute value into a query for all PHONES scroll elements. The DirXML engine filters out any data that was not requested.

5 Troubleshooting the Driver

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

Resolving Errors

The Driver Will Not Start

- Check that the nps8shim.dll is accessible.
- Check that the connection parameters are set correctly.

Attributes Do Not Get Refreshed on the Data Map Object

- Verify that the Component Interfaces are working correctly.

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

- You should verify that PeopleSoft can connect successfully to the JVM. Run jnitest.exe from PeopleSoft's External API directory: {pshome}\bin\client\winx86_api\.

  Before configuring the driver, you should install the PeopleSoft External Application Interface software (PSEXTAPI). The PSEXTAPI requires a Java Virtual Machine because the calls to the application server are done through Jolt. The JVM should be installed on the Operating System where the driver shim (nps8shim.dll) runs.

  If you don't have the JVM, you can download it from the Microsoft Download Center (http://www.microsoft.com/downloads) (msjavx86.exe); or you can use the JVM supplied with Sun's JDK 1.2.2. For additional information, please refer to your PeopleSoft documentation.

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.

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 will not process if they are in the PROD database, but 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.

No Data Is Returned When Running the Component Interface Test Program

No data is returned, particularly when you are running the Component Interface Test Program.

There are two typical reasons for this error:

- The Key Input elements are not pointing to the Search Record
- The field elements point to an invalid reference

Make sure the Key Input elements are associated with the Search Record entry. Also, make sure that the field elements are mapped to an appropriate field record definition that exists in the Component buffer. You should also ensure that the data elements exist on a page within the component and that the links are configured correctly.

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), Working (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 state of selected with a status of 1.

  NOTE: If notification processing is required, Novell recommends using the DirXML Notification Service instead of using SMTP processing as configured in PeopleSoft.

Receiving Errors on the Publisher Channel When Processing a Transaction

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

- Operation vetoed by Create policy

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

- generateKeyPair: -216 DSERR_PASSWORD_TOO_SHORT

  The attribute used for the initial password does not comply to policy; however, the user object will still be 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.