

# Novell Identity Manager Driver for PeopleSoft\* 5.2

3.5.1

[www.novell.com](http://www.novell.com)

---

IMPLEMENTATION GUIDE

September 28, 2007



**Novell**<sup>®</sup>

## Legal Notices

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

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

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

Copyright © 2007 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 on the [Novell Legal Patents Web page \(http://www.novell.com/company/legal/patents/\)](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](http://www.novell.com)

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

## **Novell Trademarks**

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

## **Third-Party Materials**

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



# Contents

<b>About This Guide</b>	<b>9</b>
<b>1 Introducing the Identity Manager Driver 5.2 for PeopleSoft</b>	<b>11</b>
1.1 Changes in Terminology	11
1.2 Driver Components	11
1.2.1 How the Driver Works	12
1.2.2 Configuring Your PeopleSoft Environment	14
1.2.3 Configuring Your Identity Manager System	15
1.3 Prerequisites	15
1.4 Key Driver Features	15
1.4.1 Local Platforms	15
1.4.2 Remote Platforms	16
1.4.3 Entitlements	16
1.5 New Features	16
1.5.1 Driver Features	16
1.5.2 Identity Manager Features	16
<b>2 Configuring Your PeopleSoft Environment</b>	<b>17</b>
2.1 Importing the PeopleSoft Driver	17
2.1.1 Importing the Driver Configuration File in Designer	17
2.1.2 Importing the Driver Configuration in iManager	17
2.2 Using the PeopleSoft Service Agent	18
2.2.1 The Component Interface Infrastructure for Identity Manager	19
2.2.2 The Sample Application	19
2.3 Installing the PSA Sample Project	20
2.3.1 Installing the PSA Files	20
2.3.2 Importing the PSA Project into the PeopleSoft Database	20
2.3.3 Building Project Record Definitions	21
2.3.4 Applying Security to the PSA	21
2.3.5 Understanding the Architecture of the PSA Sample Project	22
2.3.6 Testing Sample PeopleSoft Applications	24
2.4 Component Interfaces	27
2.4.1 Accessing Transactions and Data through Component Interfaces	27
2.4.2 Configuring the Transaction Record SQL Date/Time Format	30
2.4.3 Configuring PeopleCode to Trigger Transactions	31
2.4.4 Testing Component Interfaces	33
<b>3 Installing and Configuring the Driver</b>	<b>39</b>
3.1 Installation Instructions	39
3.1.1 Pre-installation Instructions	39
3.2 Installing the Driver	40
3.3 Importing the Driver Configuration in iManager	40
3.4 Installing the Peoplesoft Driver through Designer	44
3.5 Activating the Driver	47

<b>4</b>	<b>Customizing the Driver</b>	<b>49</b>
4.1	Customizing the PSA by Triggering Transactions	49
4.2	Changing the Driver by Changing the Data Schema Component Interface	51
4.2.1	Building the PeopleSoft Java Component Interface API	51
4.2.2	Compiling the Java CI API	53
4.2.3	Building the CI API JAR File	53
4.3	Customizing the Driver by Modifying Driver Policies	54
4.3.1	Modifying the Driver Mapping Policy	55
4.3.2	Using the Schema Query to Refresh the PeopleSoft Schema CI	55
4.3.3	Publisher Channel Objects	55
4.3.4	Understanding the Publisher Filter	56
4.3.5	Publisher Filter Attributes	56
4.3.6	Securing the Data	57
4.3.7	Publisher Object Policies	57
4.3.8	Input Transformation Policy	57
4.3.9	Matching Policy	58
4.3.10	Create Policy	58
4.3.11	Placement Policy	58
4.3.12	Command Transformation Policy	59
4.3.13	Subscriber Channel Objects	61
4.3.14	Understanding the Subscriber Filter	62
4.3.15	Securing the Data	63
4.3.16	Modifying the Filter	63
4.3.17	Subscriber Object Policies	63
<b>5</b>	<b>Upgrading the Driver</b>	<b>67</b>
5.1	Changes in Policy Architecture	67
5.2	Upgrading the Driver in Designer	67
5.3	Upgrading the Driver in iManager	69
<b>6</b>	<b>Activating the Driver</b>	<b>71</b>
<b>7</b>	<b>Managing the Driver</b>	<b>73</b>
7.1	Starting, Stopping, or Restarting the Driver	73
7.1.1	Starting the Driver in Designer	73
7.1.2	Starting the Driver in iManager	73
7.1.3	Stopping the Driver in Designer	73
7.1.4	Stopping the Driver in iManager	73
7.1.5	Restarting the Driver in Designer	74
7.1.6	Restarting the Driver in iManager	74
7.2	Migrating and Resynchronizing Data	74
7.3	Using the DirXML Command Line Utility	74
7.4	Viewing Driver Versioning Information	75
7.4.1	Viewing a Hierarchical Display of Versioning Information	75
7.4.2	Viewing the Versioning Information As a Text File	76
7.4.3	Saving Versioning Information	78
7.5	Reassociating a Driver Set Object with a Server Object	79
7.6	Changing the Driver Configuration	79
7.7	Storing Driver Passwords Securely with Named Passwords	80
7.7.1	Using Designer to Configure Named Passwords	80
7.7.2	Using iManager to Configure Named Passwords	81
7.7.3	Using Named Passwords in Driver Policies	82
7.7.4	Using the DirXML Command Line Utility to Configure Named Passwords	83

7.8	Adding a Driver Heartbeat . . . . .	86
<b>8</b>	<b>Synchronizing Objects</b>	<b>89</b>
8.1	What Is Synchronization? . . . . .	89
8.2	When Is Synchronization Done? . . . . .	89
8.3	How Does the Metadirectory Engine Decide Which Object to Synchronize? . . . . .	90
8.4	How Does Synchronization Work? . . . . .	91
8.4.1	Scenario One . . . . .	91
8.4.2	Scenario Two . . . . .	93
8.4.3	Scenario Three . . . . .	94
<b>9</b>	<b>Troubleshooting the Driver</b>	<b>95</b>
9.1	The Driver is Not Processing Available Transactions or is Processing Them Out of Order . . . . .	95
9.2	Error Trying to Obtain Data Record . . . . .	95
9.3	Error: joltServiceException: Invalid Session . . . . .	96
9.4	The Driver Does Not Start . . . . .	96
9.5	Attributes Are Not Refreshed on the Data Schema Object . . . . .	96
9.6	Data Does Not Show Up in Identity Vault on the Publisher Channel. . . . .	96
9.7	Error: Check Application Server IP Address and Jolt Port Number. . . . .	96
9.8	Data Does Not Update in PeopleSoft on the Subscriber Channel. . . . .	97
9.9	No Transactions Are Coming across the Publisher Channel. . . . .	97
9.10	Transactions Are Not Placed in the PeopleSoft Queue. . . . .	97
9.11	Transactions Are Left in the "Process" State and Not Processed . . . . .	97
9.12	Errors on the Publisher Channel When Processing a Transaction . . . . .	97
9.13	Component Interface Relationships Not Functioning. . . . .	98
9.14	SQL Error During Save of "Sample Person" Records . . . . .	98
9.15	Troubleshooting Driver Processes. . . . .	99
9.15.1	Viewing Driver Processes . . . . .	99
<b>10</b>	<b>Backing Up the Driver</b>	<b>107</b>
10.1	Exporting the Driver in Designer . . . . .	107
10.2	Exporting the Driver in iManager . . . . .	107
<b>11</b>	<b>Security: Best Practices</b>	<b>109</b>
<b>A</b>	<b>DirXML Command Line Utility</b>	<b>111</b>
A.1	Interactive Mode . . . . .	111
A.2	Command Line Mode . . . . .	120
<b>B</b>	<b>Properties of the Driver</b>	<b>125</b>
B.1	Driver Configuration . . . . .	125
B.1.1	Driver Module . . . . .	126
B.1.2	Driver Object Password. . . . .	126
B.1.3	Authentication . . . . .	127
B.1.4	Startup Option . . . . .	128
B.1.5	Driver Parameters . . . . .	129
B.2	Global Configuration Values . . . . .	130
B.3	Named Passwords. . . . .	132

B.4	Engine Control Values .....	132
B.5	Log Level .....	134
B.6	Driver Image .....	135
B.7	Security Equals .....	135
B.8	Filter .....	136
B.9	Edit Filter XML .....	136
B.10	Misc .....	137
B.11	Excluded Users .....	137
B.12	Driver Manifest .....	138
B.13	Driver Inspector .....	138
B.14	Driver Cache Inspector .....	139
B.15	Inspector .....	139
B.16	Server Variables .....	139



# About This Guide

## Introduction

The Identity Manager Driver 5.2 for PeopleSoft provides a solution for synchronizing data between Novell® Identity Vault and PeopleSoft.

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

- ♦ [Chapter 1, “Introducing the Identity Manager Driver 5.2 for PeopleSoft,” on page 11](#)
- ♦ [Chapter 2, “Configuring Your PeopleSoft Environment,” on page 17](#)
- ♦ [Chapter 3, “Installing and Configuring the Driver,” on page 39](#)
- ♦ [Chapter 4, “Customizing the Driver,” on page 49](#)
- ♦ [Chapter 5, “Upgrading the Driver,” on page 67](#)
- ♦ [Chapter 6, “Activating the Driver,” on page 71](#)
- ♦ [Chapter 7, “Managing the Driver,” on page 73](#)
- ♦ [Chapter 8, “Synchronizing Objects,” on page 89](#)
- ♦ [Chapter 9, “Troubleshooting the Driver,” on page 95](#)
- ♦ [Appendix A, “DirXML Command Line Utility,” on page 111](#)
- ♦ [Appendix B, “Properties of the Driver,” on page 125](#)

## Audience

This guide is intended for consultants, administrators, and IS personnel who need to install, configure, and maintain the Identity Manager Driver 5.2 for PeopleSoft.

## Feedback

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

## Documentation Updates

For the most recent version of this document, see the [Drivers Web site \(http://www.novell.com/documentation/idm35drivers/index.html\)](http://www.novell.com/documentation/idm35drivers/index.html).

## Additional Documentation

For documentation on using Novell Identity Manager and the other drivers, see the [Identity Manager Documentation Web site \(http://www.novell.com/documentation/idm35/\)](http://www.novell.com/documentation/idm35/).

## Documentation Conventions

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

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

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

# Introducing the Identity Manager Driver 5.2 for PeopleSoft

PeopleSoft applications are some of the most popular Enterprise Resource Planning (ERP) systems available. The Identity Manager Driver 5.2 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 Identity Vault. Using Novell Identity Manager technology, you can share and synchronize authoritative PeopleSoft data with other enterprise applications, databases, or directories. As new records are added, modified, disabled, or deactivated in PeopleSoft, tasks associated with these events can be processed automatically using Identity Manager.

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

This section contains the following topics:

- ♦ [Section 1.1, “Changes in Terminology,” on page 11](#)
- ♦ [Section 1.2, “Driver Components,” on page 11](#)
- ♦ [Section 1.3, “Prerequisites,” on page 15](#)
- ♦ [Section 1.4, “Key Driver Features,” on page 15](#)
- ♦ [Section 1.5, “New Features,” on page 16](#)

## 1.1 Changes in Terminology

The following terms have changed from earlier releases:

**Table 1-1** *Changes in Terminology*

Earlier Terms	New Terms
DirXML®	Identity Manager
DirXML Server	Metadirectory server
DirXML engine	Metadirectory engine
eDirectory	Identity Vault (except when referring to eDirectory attributes or classes)

## 1.2 Driver Components

The driver includes the following components:

- ♦ Driver shim

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

- ◆ **Driver Configuration**

`PeopleSoft50-IDM3_5_0-V1.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 39](#) and [Section 4.3, “Customizing the Driver by Modifying Driver Policies,” on page 54](#).

- ◆ **PeopleSoft Service Agent**

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

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

- ◆ Implementation of an intermediate staging table. The synchronization between the Novell-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.

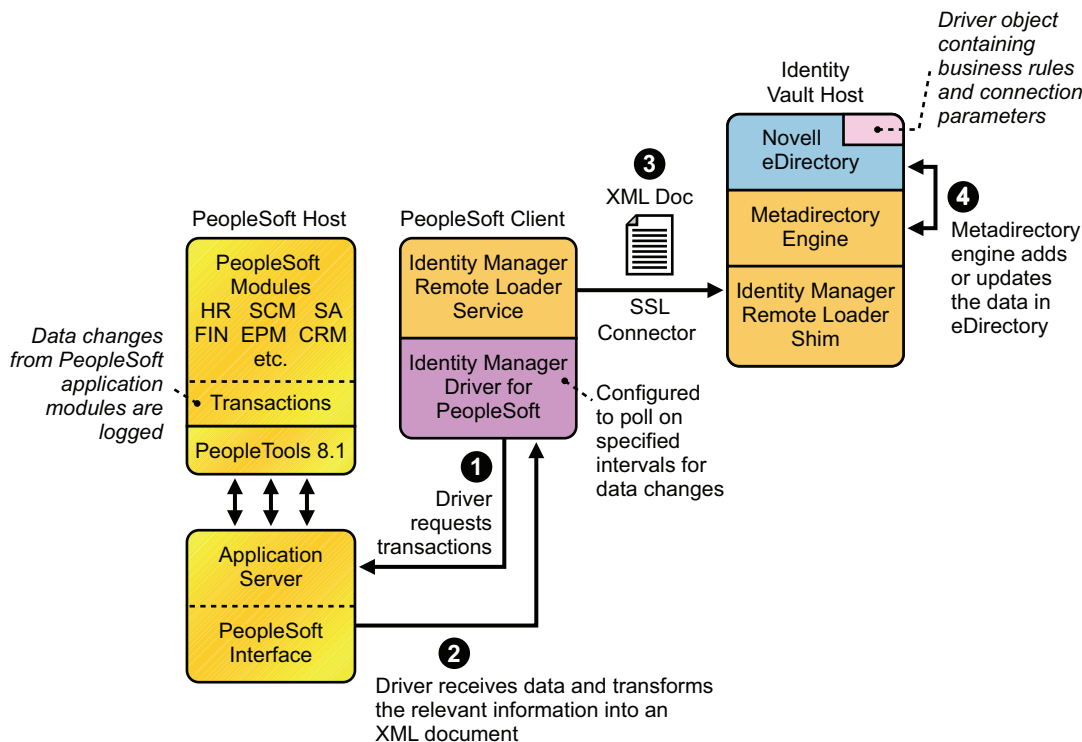
As with the driver configuration, the PSA is not intended for use in a production environment. It acts as a template that contains most of the common synchronization tasks needed in typical integration scenarios. You should configure your policies based on your own business processes and integration points. For more information, refer to [Section 2.2, “Using the PeopleSoft Service Agent,” on page 18](#).

## 1.2.1 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 39](#).

## The Publisher Channel

Figure 1-1 The Publisher Channel



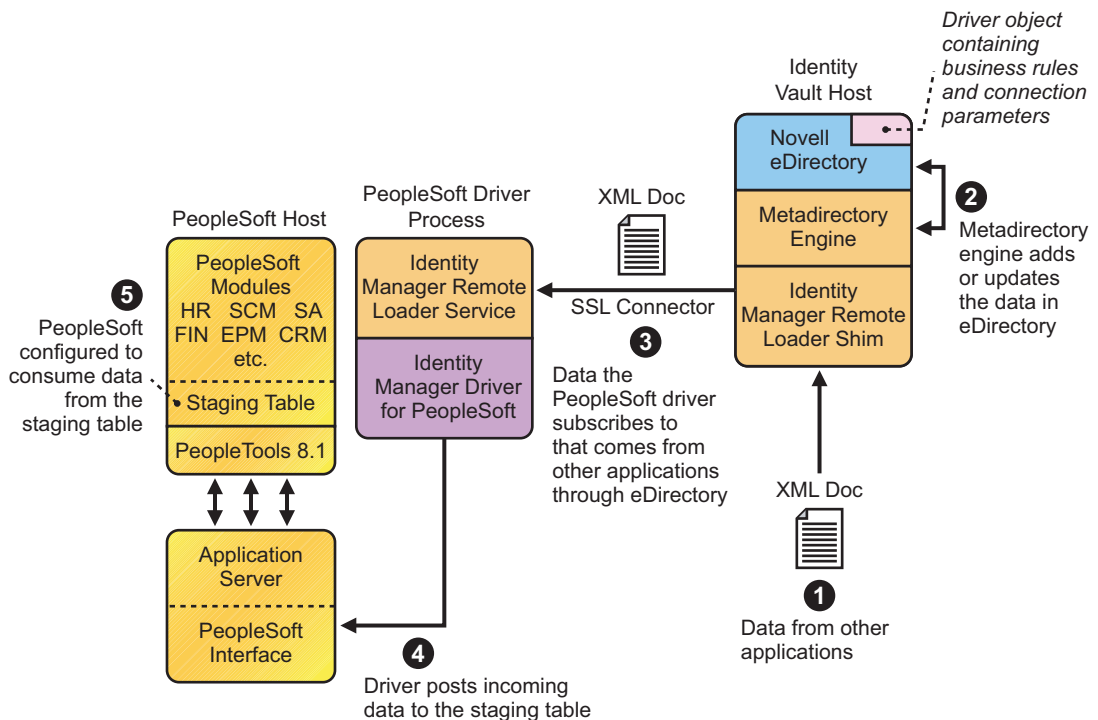
The Publisher channel synchronizes data from PeopleSoft to Identity Vault. As events occur within PeopleSoft, transactions are placed into a transaction table. These transactions are usually written to the table through PeopleCode (you can use other methods such as Batch SQL, COBOL, SQR, and so forth.)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

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

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

## The Subscriber Channel

Figure 1-2 The Subscriber Channel



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

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

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

### 1.2.2 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 17.

## 1.2.3 Configuring Your Identity Manager System

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

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

For more information on configuring your Identity Manager system, refer to [Chapter 4, “Customizing the Driver,”](#) on page 49.

## 1.3 Prerequisites

- ❑ Novell Identity Manager 2.0.1 or later.
- ❑ PeopleSoft Application Server with PeopleTools version 8.17 or later, 8.20 or later, or 8.41 or later.
- ❑ The appropriate version of the PeopleTools psjoa.jar client to match the PeopleTools version of the target PeopleSoft Application Server.
- ❑ A .jar file containing the compiled Java\* Component Interface APIs for the desired integration component. For the default PSA components, the file containing the interfaces is named dirxmlcomps.jar. For more information on creating Component Interface APIs, refer to [Chapter 4, “Customizing the Driver,”](#) on page 49.

## 1.4 Key Driver Features

The sections below contains a list of the key driver features.

- ◆ [Section 1.4.1, “Local Platforms,”](#) on page 15
- ◆ [Section 1.4.2, “Remote Platforms,”](#) on page 16
- ◆ [Section 1.4.3, “Entitlements,”](#) on page 16

### 1.4.1 Local Platforms

The PeopleSoft driver can be installed locally on the following platforms:

- ◆ Windows\* 2000, or 2003 with the latest Service Patch
- ◆ Linux Red Hat\* 3.0 and 4.0 for AMD64/EM64T

- ♦ SUSE® LINUX Enterprise Server 9 and 10 (with latest Support Pack)
- ♦ Solaris\* 9 or 10 (Sparc only)
- ♦ AIX\* 5.2L, v5.2, and v5.3

## 1.4.2 Remote Platforms

The PeopleSoft driver can use the Remote Loader service. The Remote Loader service for the PeopleSoft driver can be installed on the following platforms:

- ♦ Windows NT, 2000, or 2003 with the latest Service Patch
- ♦ Linux Red Hat\* 3.0 and 4.0 for AMD64/EM64T
- ♦ SUSE® LINUX Enterprise Server 9 and 10 (with latest Support Pack)
- ♦ Solaris\* 9 or 10
- ♦ AIX\* 5.2L, v5.2 and v5.3

---

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

---

For more information about installing the Remote Loader services, see “[Installing the Remote Loader](#)” in the *Novell Identity Manager 3.5.1 Administration Guide*.

## 1.4.3 Entitlements

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

## 1.5 New Features

- ♦ [Section 1.5.1, “Driver Features,” on page 16](#)
- ♦ [Section 1.5.2, “Identity Manager Features,” on page 16](#)

### 1.5.1 Driver Features

The PeopleSoft driver now supports two options for Publisher Transaction record polling: interval polling and crontab format polling. If you select the interval polling method, you need to specify the number of seconds between checks for available transactions to process. The default is 5.

If you select the crontab format polling method, you need to specify five required crontab field parameters that are separated by blank characters. The default value of \* \* \* \* \* generates a poll every minute.

### 1.5.2 Identity Manager Features

Identity Manager includes new features. For more information, refer to the “[What's New in Identity Manager 3.5.1?](#)” in the *Identity Manager 3.5.1 Installation Guide*.



# Configuring Your PeopleSoft Environment

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

- ♦ [Section 2.1, “Importing the PeopleSoft Driver,” on page 17](#)
- ♦ [Section 2.2, “Using the PeopleSoft Service Agent,” on page 18](#)
- ♦ [Section 2.3, “Installing the PSA Sample Project,” on page 20](#)
- ♦ [Section 2.4, “Component Interfaces,” on page 27](#)

## 2.1 Importing the PeopleSoft Driver

This section talks briefly about importing the PeopleSoft driver through Designer and iManager utilities.

### 2.1.1 Importing the Driver Configuration File in Designer

Designer allows you to import the basic driver configuration file for PeopleSoft. 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.

There are many different ways of importing the driver configuration file. This procedure only documents one way.

- 1 Open a project in Designer and in the modeler, right-click on the Driver Set object and select *Add Connected Application*.
- 2 From the drop-down list, select *PeopleSoft50-IDM3\_5\_0-V1.xml*, then click *Run*.
- 3 Click *Yes*, in the Perform Prompt Validation window. It has you fill in all of the fields to correctly configure the PeopleSoft driver.
- 4 Configure the driver by filling in the fields. Specify information specific to your environment.
- 5 After specifying parameters, click *OK* to import the driver.
- 6 After the driver is imported, customize and test the driver.
- 7 Once the driver is fully tested, deploy the driver into the Identity Vault. See [“Deploying a Driver to an Identity Vault”](#) in the *Designer 2.1 for Identity Manager 3.5.1*.

### 2.1.2 Importing the Driver Configuration in iManager

The Create Driver Wizard in iManager helps you import the basic driver configuration file for PeopleSoft. 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 *Identity Manager Utilities > Import Drivers*.

- 2 Select a driver set, then click *Next*.

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

- 3 Select *PeopleSoft--Release 5.0.1 for PeopleTools 8*, and then click *Next*.

- 4 Configure the driver by filling in the fields. Specify information specific to your environment.

- 5 After specifying parameters, 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 write to. You can do this by granting Security Equivalence to the driver object. Normally, the driver should be given security equal to Admin.

- 6 Identify all objects that represent administrative roles and exclude them from replication.

Exclude the security-equivalence object (for example, DriversUser) that you specified in Step 5. If you delete the security-equivalence object, you have removed the rights from the driver. Therefore, the driver can't make changes to Identity Manager.

- 7 Review the driver objects in the Summary page, then click *Finish*.

Keep in mind that installing the driver software lets you get the driver up and running, but it does not install the product license. Without the license and activation, the driver will not run after 90 days. For more information, refer to “[Activating Novell Identity Manager Products](#)” in the *Identity Manager 3.5.1 Installation Guide*.

## 2.2 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® Identity Vault. 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 find installation and configuration information for the two main PSA components: “[The Component Interface Infrastructure for Identity Manager](#)” on page 19 and “[The Sample Application](#)” on page 19.

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.

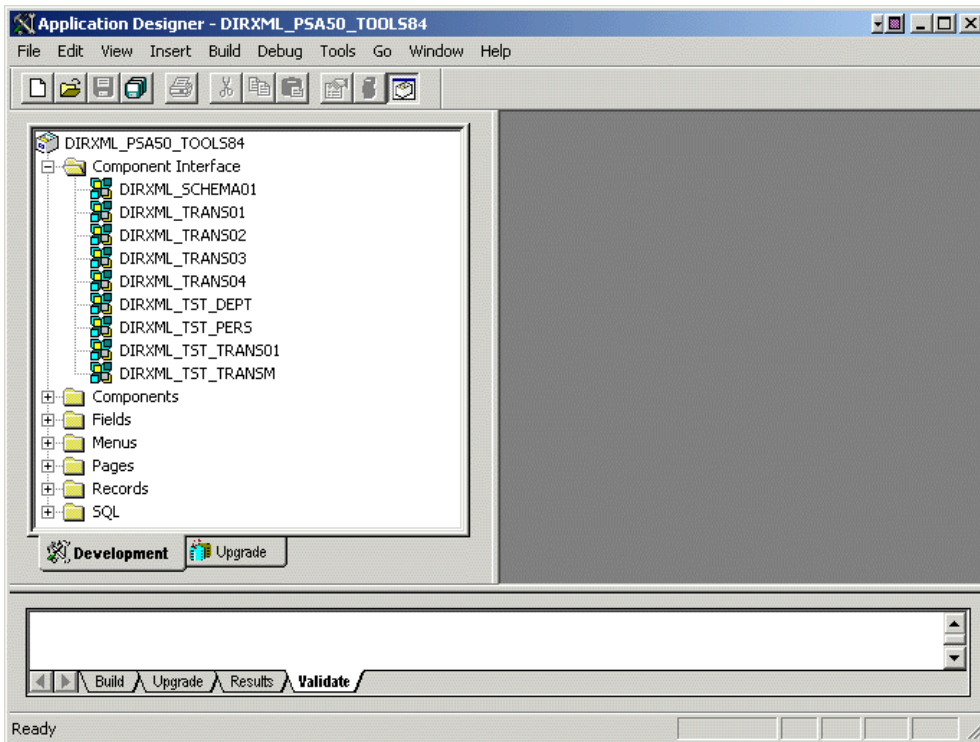
---

**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 9, “Troubleshooting the Driver,”](#) on page 95 for specific issues and consult with your DBMS/PeopleSoft Database administrator for additional assistance.

---

## 2.2.1 The Component Interface Infrastructure for Identity Manager

Figure 2-1 The DirXML\* Component Interface



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

For more information, refer to [“Configuring PeopleCode to Trigger Transactions” on page 31](#).

## 2.2.2 The Sample Application

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

Depending on your business requirements, you should configure internal processes to trigger events into transaction tables, synchronize with other PeopleSoft tables, etc. either by replicating the provided PeopleCode or by merging the components within your PeopleSoft environment. When synchronizing data internally between application tables, you should always try to use the tools that provide the highest degree of data integrity checking. (For example, the Staging CI to Application CI synchronization in the PSA utilizes the PeopleCode CI interface to ensure proper syntax, translate table usage, related fields, etc. as it has been defined on the Application record.) For more information, refer to [“Understanding the Architecture of the PSA Sample Project” on page 22](#).

## 2.3 Installing the PSA Sample Project

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

- ◆ [Section 2.3.1, “Installing the PSA Files,” on page 20](#)
- ◆ [Section 2.3.2, “Importing the PSA Project into the PeopleSoft Database,” on page 20](#)
- ◆ [Section 2.3.3, “Building Project Record Definitions,” on page 21](#)
- ◆ [Section 2.3.4, “Applying Security to the PSA,” on page 21](#)
- ◆ [Section 2.3.5, “Understanding the Architecture of the PSA Sample Project,” on page 22](#)
- ◆ [Section 2.3.6, “Testing Sample PeopleSoft Applications,” on page 24](#)

### 2.3.1 Installing the PSA Files

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

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

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

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

### 2.3.2 Importing the PSA Project into the PeopleSoft Database

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

#### For PeopleSoft 8.1x and 8.2x

- 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 8.4x

- 1 Connect to the PeopleSoft database as administrator in two tier mode.
- 2 In the Application Designer, select *Tools > Copy Files > 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.

### 2.3.3 Building Project Record Definitions

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

- 1 Log in to PeopleSoft Application Designer using an administrator username that has administrative and development rights.
- 2 In the Application Designer, select *Build > Project*.
- 3 From Build Options, click *Create Tables and Execute SQL Now*. After project tables are created, click *Close* to close the Build Progress window.
- 4 Click *Build* to create sample project tables.  
You must create project tables before creating the views. Views are created 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.

### 2.3.4 Applying Security to the PSA

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

#### For PeopleSoft 8.1x and 8.2x

- 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<sup>®</sup> permissions list, then click *Save*.
- 8 Assign *DirXML Administrator* role to your administrative user by clicking *PeopleTools > Security > User Profiles > User Profiles*.
- 9 Click *Search* to locate the User Profile that you want to add the DirXML Administrator role to, then click the *User ID*.
- 10 Click the *Roles* tab, then click one of the “+” buttons to add a new role.
- 11 Search and select the role, click *DirXML Administrator* to add it, then click *Save*.

## 2.3.5 Understanding the Architecture of the PSA Sample Project

The PSA Sample project is intended to provide recommended PeopleSoft integration scenarios through the use of very comprehensive 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 is recorded with an NPSDriver1P identifier that shows the data is being published by a PeopleSoft administrator. Changes made via the Component Interface API are recorded with an NPSDriver1S identifier that shows the data was subscribed into the application tables programmatically.

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

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

## PSA Best Practices

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

---

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

---

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

- ◆ Whenever possible, always provide a Component Interface 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. Most importantly, 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, the generation process ensures that Translate table values and related field values generated by the changes are properly synchronized.
- ◆ If you are using SQLExec() statements to update tables or create records, use great care to ensure that you are not violating the logic and rules of the applications overlying the tables. SQL is the easiest and most powerful, and therefore most destructive, method for updating data.
- ◆ Do not generate Transaction records until after you have successfully updated the application tables.
- ◆ If desired, it is possible to completely bypass the staging table interface in your synchronization scenario. The driver can be directed to interact with any 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 39](#).
- ◆ Test everything thoroughly.

### 2.3.6 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 8.1x and 8.2x

- 1 In the Application Designer, select *Go > DirXML Administrator*.
- 2 In the DirXML Administrator menu, select *Use > DirXML Sample Department*.
- 3 Click an empty Department field row to add sample department and description values.
- 4 Click *Save* to add the Department.
- 5 From the DirXML Administrator menu, select *Use > DirXML Sample People > Add*.
- 6 Type data into the various fields for the new person, then click *Save*.  
Asterisks represent required fields.
- 7 Verify that an *ADD* transaction was created by selecting *Use > DirXML Transaction 01*.
- 8 Click the *Search* button.
- 9 Verify that the transaction was created and select the transaction.

DirXML Administrator - Use - DirXML Transaction 01

File Edit View Go Favorites Use Help

Dirxml Transaction 01

SubType: NPSDriver1P Field Name:  
Instance: 35 Event: ADD Field Key:  
Schema: DIRXML\_SCHEMA01  
Assoc ID: P000004

Transaction Status:  
 Available  Warning  
 In Process  Error  
 Success  Cancelled

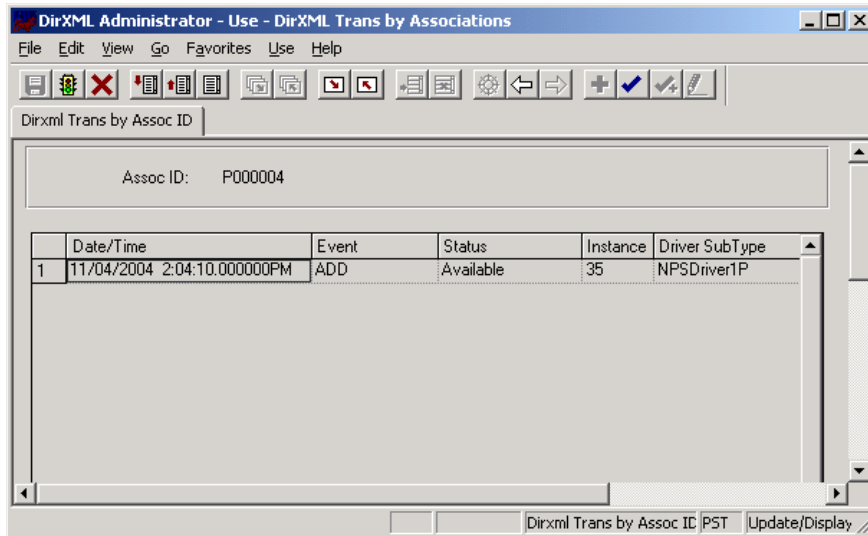
Action Date/Time: 2004-11-04-14.04.10.000000  
Status Date/Time: 11/04/2004 2:04:10.000000PM  
Reset Date/Time: 11/04/2004 2:04:10.000000PM

Comment:  
Transaction Value:  
P000003|Smith

Dirxml Transaction 01 PST Update/Display

- 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 8.4x

- 1 Log in to the PeopleSoft portal.
- 2 Click *DirXML Administrator* from the left menu.

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

- 3 Click *DirXML Sample Department*.

The screenshot shows the PeopleSoft DirXML Administrator interface. On the left is a 'Menu' with a search box and a tree view. The tree view is expanded to 'DirXML Sample Department'. The main window is titled 'Dirxml Sample Department' and contains a table with the following data:

Department	Description	DirXML DN		
1 00001	HR	HR	+	-
2 00002	OPS	OPS	+	-
3 00003	MAINTENANCE	MAINTENANCE	+	-
4 00004	PILOTS	PILOTS	+	-
5 00005	OFFICE OF CEO	OOC	+	-
6 00006	SALES	SALES	+	-

Below the table are two buttons: 'Save' and 'Notify'.

- 4 Specify a sample department, then click *Save*.
- 5 Click *DirXML Sample People*.
- 6 Enter values for a sample user, then click *Save* and verify that the user's data appears in the Transaction01 table. You do this by searching in the DirXML Transaction01 application.
- 7 Verify that other delivered applications work by selecting them from the *DirXML Administrator* menu.

## 2.4 Component Interfaces

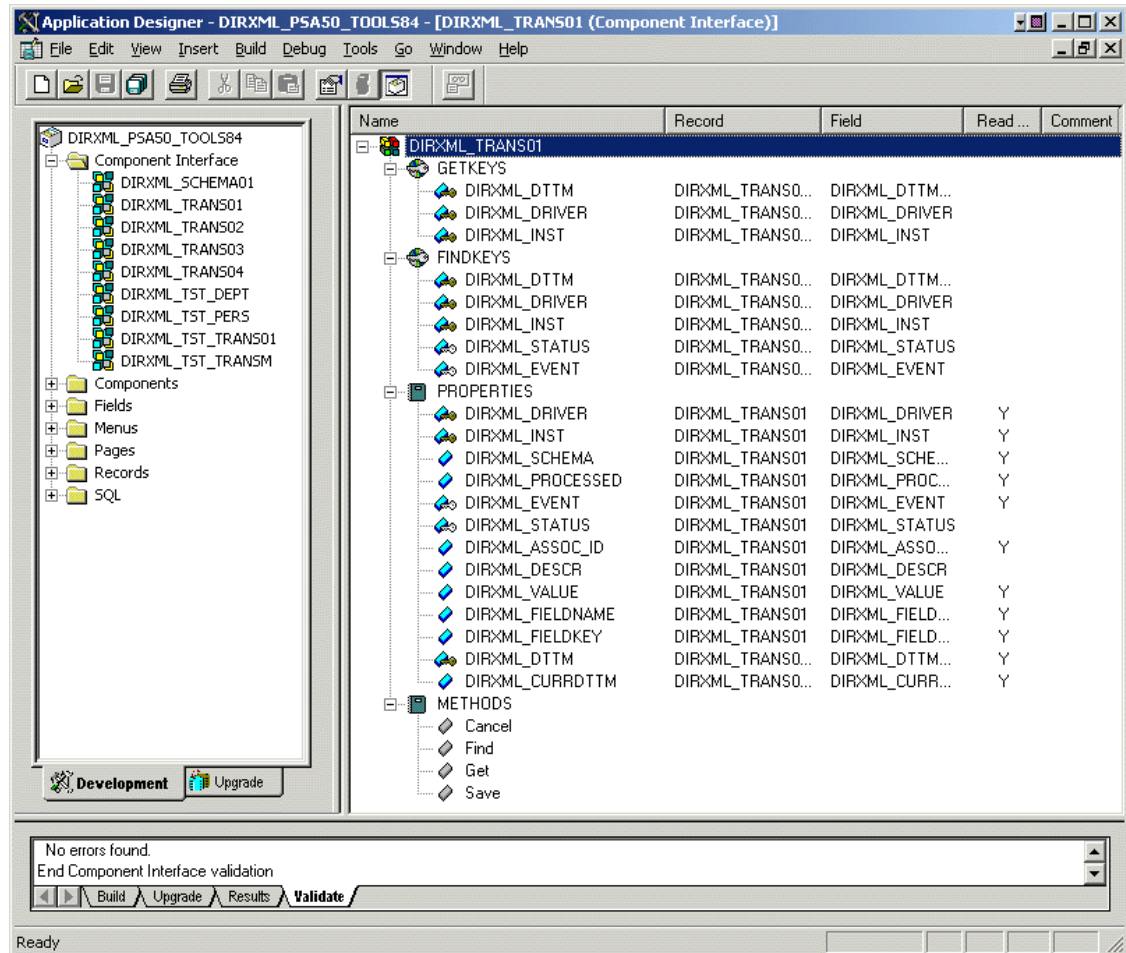
- Section 2.4.1, “Accessing Transactions and Data through Component Interfaces,” on page 27
- Section 2.4.2, “Configuring the Transaction Record SQL Date/Time Format,” on page 30
- Section 2.4.3, “Configuring PeopleCode to Trigger Transactions,” on page 31
- Section 2.4.4, “Testing Component Interfaces,” on page 33

### 2.4.1 Accessing Transactions and Data through Component Interfaces

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

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

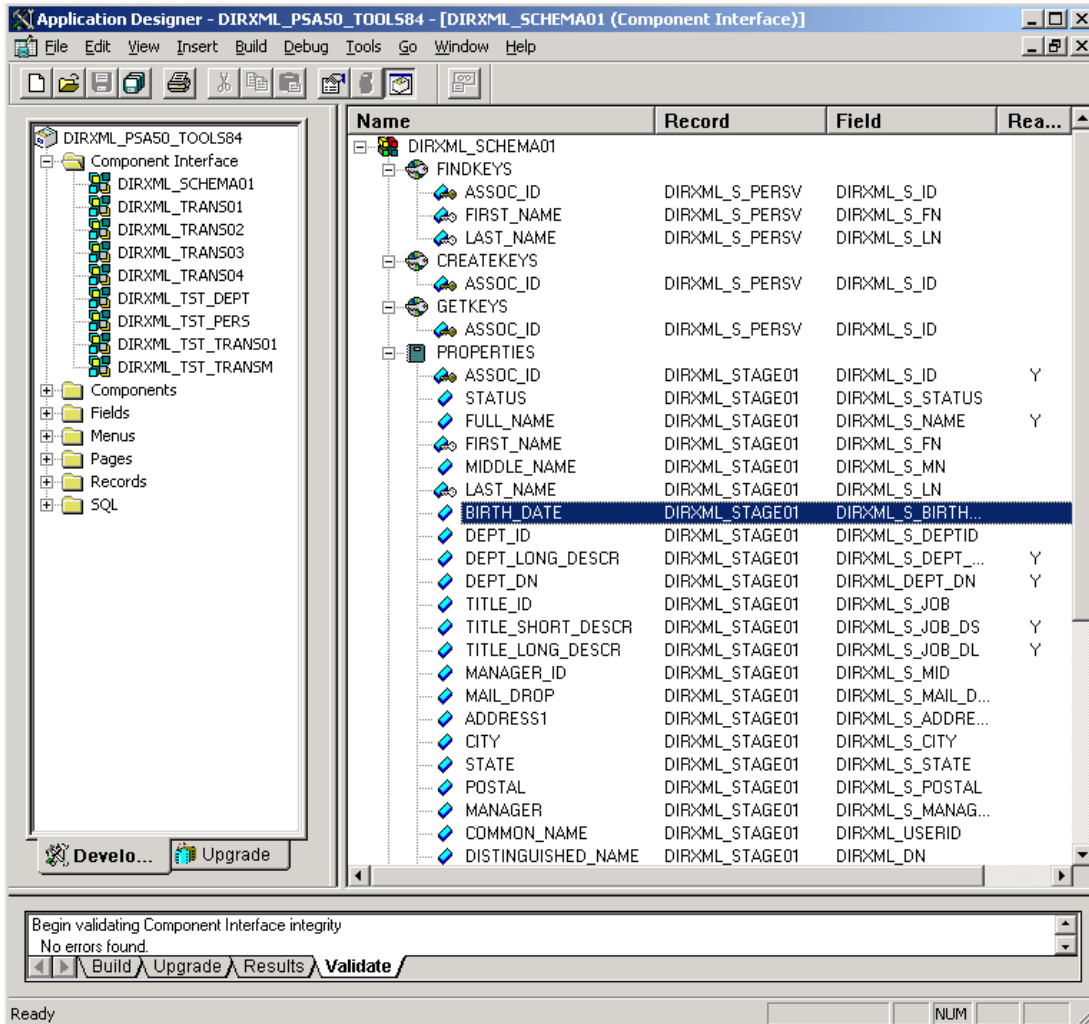
**Figure 2-2** Transaction Table CI



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

The following figure represents the CI for a Schema Component:

Figure 2-3 Schema Component



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

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

### Changes to Field Names in PeopleSoft 8.41

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

- Spaces are no longer allowed in 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

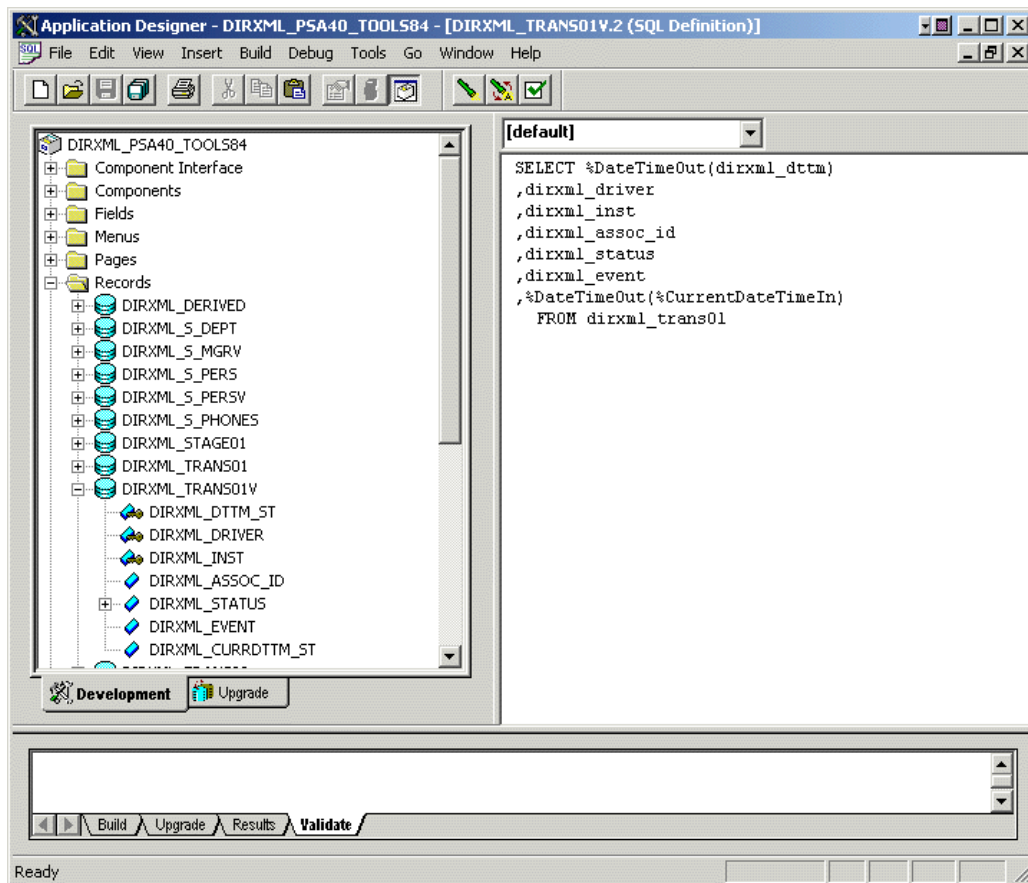
default schema of the Component Interfaces used by the driver now uses naming conventions that eliminate this unusual sorting error. This issue is particularly important for any field name modifications or additions customized for a prior implementation of the driver.

- ♦ Use the standard ALL-CAPS format to avoid any field name issues.

## 2.4.2 Configuring the Transaction Record SQL Date/Time Format

The proper functioning of the driver depends on the Date/Time strings in the Transaction View record to determine processing availability and relative event order of Transaction data rows. The Date/Time fields in the Transaction data rows are converted to formatted strings in the Transaction Views using the PeopleCode Meta-SQL%DateTimeOut() function. The following image shows the default SQL View code for the DIRXML\_TRANS01V record:

**Figure 2-4** SQL View of Code for the DirXML\_TRANS01V Record



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

- ♦ The date should be presented first in YYYY-MM-DD format.
- ♦ The time should be presented in 24-hour form with HH:MM:SS format (Additional information concatenated to this string, such as <milliseconds> is acceptable)

- ◆ These two strings should be placed together in “date-time” format.

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

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

or

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

The fields used by the driver are DIRXML\_DTTM\_ST and DIRXML\_CURRDTTM\_ST. These fields represent the Date/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:** Because the configuration of the Date and Time format varies depending on the DBMS being utilized, changing this format should be done by the DBMS/PeopleSoft Database Administrator or other qualified personnel.

---

## 2.4.3 Configuring PeopleCode to Trigger Transactions

The PSA contains a number of PeopleTools objects that enable PeopleSoft to trap events into a transaction table. The driver then accesses the transaction tables through 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 Metadirectory engine for processing. It updates the transaction status and any applicable messages on the transaction table inside of PeopleSoft after processing is completed by the Metadirectory engine. When events occur within eDirectory, the driver connects to the appropriate CI and updates the PeopleSoft staging table as appropriate.

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

The driver requires a Transaction CI and a Schema CI to process transactions. For more information on calling the PeopleSoft function that creates transaction records, please refer to [“Customizing the PSA by Triggering Transactions” on page 49](#).

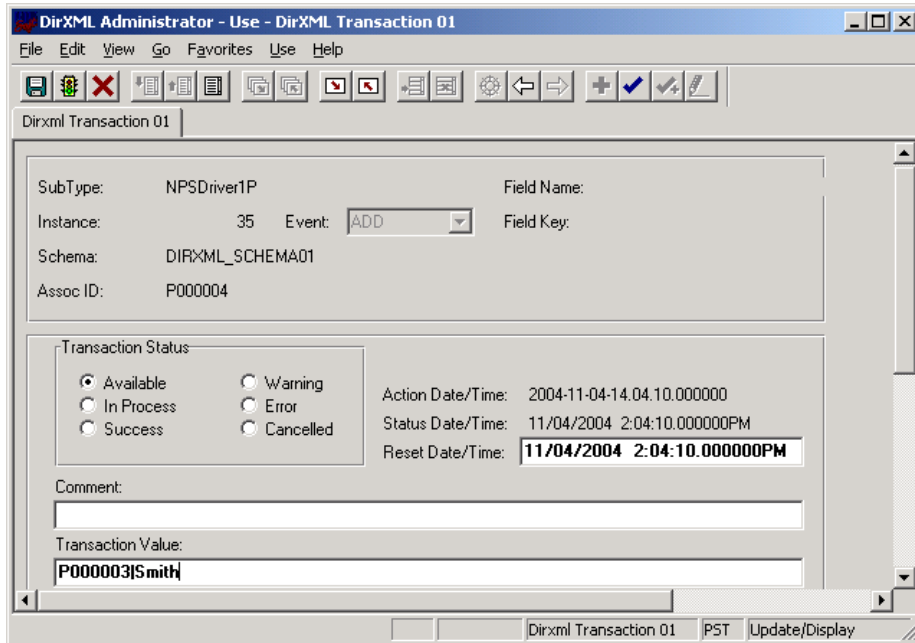
### Transaction Component

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

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

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

**Figure 2-5** *DirXML Transaction01*



## Schema Component

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

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

For example, assume you want to process transactions for an employee with the DIRXML\_ASSOC\_ID field (key) value = P000001. The driver accesses the Schema CI with a key value of P000001. It retrieves all of the configured component elements that have been defined for that employee. The driver then converts the data collection into XML documents to be consumed by the Metadirectory engine. If there is a write-back command processed, or when data is written on the Subscriber channel, the driver uses this CI to update the staging table with the appropriate information into PeopleSoft for this particular employee.



## 2.4.4 Testing Component Interfaces

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

The `CITester` completes the following checks during four phases:

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

\*If you are not using the default Schema CI, it is necessary to build the APIs for the desired CI. See [Section 4.2, “Changing the Driver by Changing the Data Schema Component Interface,” on page 51](#) for information on building custom CI API JAR files.

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

### Important Considerations

When 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 `dirxmlcomps.jar` files and any required custom CI API JAR files.

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

### How Do I Run the Test?

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

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

### Phase I: Creating a PeopleSoft Client Session

If the test program establishes a session with the PeopleSoft client, you 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.

**Table 2-1** *Phase I Errors and Solutions*

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

## Phase II: Authenticating to the PeopleSoft Client

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

If the test program verifies the connection and authentication parameters, you 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.

**Table 2-2** *Phase II Errors and Solutions*

Error Message	Solution
ERROR: Failed Connection to the PeopleSoft Application Server. Please make sure you entered your authentication information correctly.  PeopleSoft Error/Warning Messages Pending. Number of Messages: 1 Message 1: Connect Failed: No additional information available (90, 01)	The target Application Server generates error and warning messages. This error indicates that you entered the wrong Application Server Name or Application Server Port Number.  Ensure that the Server Name or address you entered contains a leading double slash (//) and that the address and name data is correct. Also, verify that you entered the Jolt port configured on the Application Server.

Error Message	Solution
<p>ERROR: Failed Connection to the PeopleSoft Application Server. Please make sure you entered your authentication information correctly.</p>	<p>This message indicates an invalid Application Server Name or Port Number. In some instances, if an invalid port number is specified, the CITester program hangs and requires a manual interrupt.</p>
<p>PeopleSoft Error/Warning Messages Pending. Number of Messages: 1 Message 1: DOWNbea.jolt.ServiceException: Invalid Session</p>	
<p>PeopleSoft Error/Warning Messages Pending Number of Messages: 2  Message 1: PeopleTools release (8.&lt;num&gt;) from web server '' is not the same as Application Server PeopleTools release (8.&lt;num&gt;) Access denied.</p>	<p>This message indicates that the PeopleTools version of the specified <code>psj0a.jar</code> does not match the version of the target PeopleSoft Application Server. PeopleTools requires a version match of the client and server.</p>
<p>PeopleSoft Error/Warning Messages Pending. Number of Messages: 3  Message 1: &lt;UserID&gt;@&lt;Client computer&gt; 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 &lt;User ID&gt;</p>	<p>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.</p>

### 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 see the following message:

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

- Property 'DIRXML\_ASSOC\_ID' is present.
- Property 'DIRXML\_CURRDTM' 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.

**Table 2-3** Phase III Errors and Solutions

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 primary 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 see the following message:

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

- Property 'ASSOC_ID' is present and validated as key field.
- Property 'STATUS' is present.
- Property 'FULL_NAME' is present.
- Property 'FIRST_NAME' is present.
- Property 'MIDDLE_NAME' is present and validated as key field.
- Property 'LAST_NAME' is present.
- Property 'BIRTH_DATE' is present.
- Property 'DEPT_ID' is present.
- Property 'DEPT_LONG_DESCR' is present.
- Property 'DEPT_DN' is present.
- Property 'TITLE_ID' is present.
- Property 'TITLE_SHORT_DESCR' is present.
- Property 'TITLE_LONG_DESCR' is present.
- Property 'MANAGER_ID' is present.
- Property 'MAIL_DROP' is present.
- Property 'ADDRESS1' is present.
- Property 'CITY' is present.
- Property 'STATE' is present.
- Property 'POSTAL' is present.
- Property 'MANAGER' is present.
- Property 'COMMON_NAME' is present.
- Property 'DISTINGUISHED_NAME' is present.
- Property 'DESCRIPTION' is present.
- Property 'EMAIL_ID' is present.
- Property 'PHONE_BUSN' is present.
- Property 'PHONE_CELL' is present.
- Property 'PHONE_HOME' is present.
- Property 'PHONE_PGR' is present.
** Schema Component Interface element validation succeeded.
** All expected platform support is verified correct.
```

## Phase IV Errors

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

**Table 2-4** Phase IV Errors and Solutions

Error Message	Solution
<p>PeopleSoft Error/Warning Messages Pending.            Number of Messages: 4</p> <p>Message 1: Cannot find Component Interface {&lt;Schema CI Name&gt;} (91,2)            Message 2: Initialization Failed (90,7)            Message 3: Not Authorized (90,6)            Message 4: Failed to execute PSSession request</p>	<p>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.</p>
<p>ERROR: Retrieval of Component Interface &lt;Schema CI Name&gt; failed.</p>	<p>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.</p>
<p>ERROR: Specified Schema Component Interface Data Record ID Field '&lt;Data Record ID Field Name&gt;' not found.</p>	<p>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.</p>
<p>ERROR: Property '&lt;Data Record ID Field Name&gt;' is not defined as key field.</p>	

## Summary

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

Component Interface Test Summary

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

```

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

# Installing and Configuring the Driver

- ♦ [Section 3.1, “Installation Instructions,” on page 39](#)
- ♦ [Section 3.2, “Installing the Driver,” on page 40](#)
- ♦ [Section 3.3, “Importing the Driver Configuration in iManager,” on page 40](#)
- ♦ [Section 3.4, “Installing the Peoplesoft Driver through Designer,” on page 44](#)
- ♦ [Section 3.5, “Activating the Driver,” on page 47](#)

## 3.1 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 the Remote Loader exists, the PSA where the PeopleSoft Application Server exists, and the driver policies to your iManager server.

For PeopleSoft v5.2, there is no standalone installation method. When you install Identity Manager, you can at that time install all of the drivers that come with Identity Manager, or you can select the drivers you want to install, as well as the driver utilities. This is covered in the [“Installing Identity Manager”](#) in the *Identity Manager 3.5.1 Installation Guide*.

After you install the PeopleSoft driver, you must configure the driver. You do this through iManager or through Designer. To configure the PeopleSoft driver through iManager, see [Section 3.3, “Importing the Driver Configuration in iManager,” on page 40](#). To configure the PeopleSoft driver through Designer, see [Section 3.4, “Installing the Peoplesoft Driver through Designer,” on page 44](#).

### 3.1.1 Pre-installation Instructions

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 in the `\lib` subdirectory with the DirXML<sup>®</sup> Java library files. By default, this is `Novell\NDS\lib` for a local driver installation or `Novell\RemoteLoader\lib` for a remote installation.

- ♦ **psftshim.jar**: This is the driver shim. It is installed when you run the Identity Manager installation program.

- ♦ **dirxmlcomps.jar:** : This library contains the compiled APIs for the DIRXML\_SCHEMA01, DIRXML\_TST\_PERS, and DIRXML\_TRANS*nm* Component interfaces delivered in the PSA. This library is installed when you run the Identity Manager installation program.

The test utility `citester.class` file is also installed when you run the Identity Manager installation program and select to install the driver utilities.

## 3.2 Installing the Driver

You install the driver as part of the Novell Identity Manager 3.5.1 installation program. For installation instructions, refer to the “[Installing Identity Manager](#)” in the *Identity Manager 3.5.1 Installation Guide*. If you are upgrading the driver, see [Chapter 5, “Upgrading the Driver,”](#) on [page 67](#).

Importing the driver configuration creates the driver object. After you have imported the configuration, you can use iManager to configure and manage the driver. See [Section 3.3, “Importing the Driver Configuration in iManager,”](#) on [page 40](#) or [Section 3.4, “Installing the Peoplesoft Driver through Designer,”](#) on [page 44](#) for instructions on how to configure the driver.

## 3.3 Importing the Driver Configuration in iManager

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 *Identity Manager Utilities > New 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 (XML file)*, then select *PeopleSoft50-IDM3\_5\_0-V1.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 driver’s parameters. See [Table 3-1 on page 41](#) for a list of parameters you can set.

- 5 Click *Import*.

When the import is finished, you should 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, resources, and distribution lists. Normally, the driver should be given security equal to Admin.

- 6 Review the driver objects in the *Summary* page, then click *Finish*.



**Table 3-1** Specify Values for These Parameters

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 from PeopleSoft are placed. You can modify this parameter through a global configuration variable (GCV) after installation.
Inactive Users Container	The name of the Organizational Unit where Inactive users from PeopleSoft are placed. You can modify this parameter through a GCV after installation.
Active Employees Group	The name of the Group Object to which Active Employee users from PeopleSoft are added. You can modify this parameter through a GCV after installation.
Active Managers Group	The name of the Group Object to which Active Manager users from PeopleSoft are added. You can modify this parameter through a GCV after installation.
PeopleSoft Connection String	<p>The hostname or IP address and port number for connecting to the appropriate PeopleSoft Application server. This is typically referred to as the PeopleSoft application server connection string. The default port is 9000.</p> <p>The connection string uses the following format:            &lt;hostname or IP address&gt;:&lt;Jolt Port Number&gt;            Example: //PSServer:9000</p> <p>To enable failover and loadbalancing, you can supply multiple server connection strings separated by a comma. Example: //PSServer:9000,//111.222.3.4:9000</p>
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.
Password Failure Notification User	Password synchronization policies are configured to send e-mail notifications to the associated user when password updates fail. You have the option of sending a copy of the notification e-mail to another user, such as a security administrator. If you want to send a copy, specify the DN of that user now. Otherwise, leave this field blank.
Driver is Local/Remote	Configure the driver for use with the Remote Loader service by selecting the <i>Remote</i> option, or select <i>Local</i> to configure the driver for local use. (If you are using PeopleTools 8.4x, you must select a Remote installation. <i>Local</i> implementations are not supported.) If <i>Local</i> is selected, you can skip the remaining parameters.
Remote Host Name and Port	Specify the hostname 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.
Allow 'add' events	Subscriber Add events are implemented by invoking the Component Interface Create method (if present). If you want the driver to allow Subscriber channel add events, select <i>Allow Subscriber add</i> .
Allow 'delete' events	Subscriber Delete events are implemented by invoking the Component Interface Delete method (if present). If you want the driver to allow Subscriber channel Delete events, select <i>Allow Subscriber delete</i> .

Parameter	Description
Driver Password	The driver object password is used by the Remote Loader to authenticate itself to the Metadirectory server. It must be the same password that is specified as the driver object password on the 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 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).

**Table 3-2** *Additional Driver Parameters*

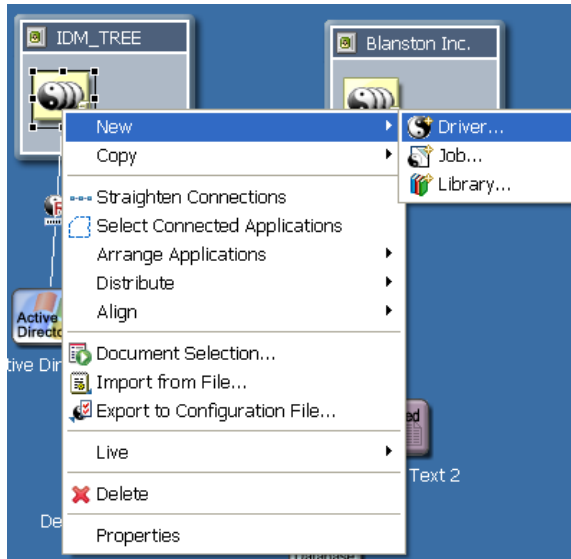
Parameter	Description	Default Value
Schema CI Name	List of the names of the PeopleSoft CI objects that define the set of data to be synchronized by the driver.	DIRXML_SCHEMA01
Data Record ID Field	The name of the field in the Data Schema CI that uniquely identifies a PeopleSoft object. The value in this field is used as the DirXML object association identifier.	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 if Subscriber channel Add events are allowed.	NEW
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

Parameter	Description	Default Value
Driver Subset Identifier	<p>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.</p> <p>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.</p> <p>This allows multiple drivers to utilize the same transaction CI, which in turn can be populated by multiple PeopleSoft applications or processes.</p>	NPSDriver1
Publisher Polling Option	The PeopleSoft driver supports two options for Publisher Transaction record polling. To choose an interval of seconds between polls, select <i>Utilize Interval Polling</i> . To use a crontab format, select <i>Utilize crontab Format Polling</i> .	Utilize Interval Polling
Queue Poll Interval (seconds) or Enter Queue Poll crontab Format String	<p>If you select <i>Utilize Interval Polling</i>, this entry displays <i>Queue Poll Interval</i>. Specify the number of seconds between checks for available transactions to process. The default is 5.</p> <p>If you select <i>Utilize crontab Format Polling</i>, this entry displays <i>Enter Queue Poll crontab Format String</i>. Specify five required crontab field parameters that are separated by blank characters. The default value of * * * * * generates a poll every minute.</p>	<p>5 seconds</p> <p>or</p> <p>* * * * *</p>

## 3.4 Installing the Peoplesoft Driver through Designer

Designer has a driver configuration wizard to help you import and configure the PeopleSoft driver configuration file. This file creates and configures the objects and policies needed to make the driver work properly.

- 1 In Designer, right-click the driver set where you plan to install and configure the PeopleSoft driver, then select *New > Driver*.



- 2 From the Connecting to Application window, select the `PeopleSoft50-IDM3_5_0-V1.xml` file. Also select *Perform required prompt checking*, then click *Run*.
- 3 Specify values for the following parameters:

**Table 3-3** Driver Configuration Parameters

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 from PeopleSoft are placed. You can modify this parameter through a global configuration variable (GCV) after installation.
Inactive Users Container	The name of the Organizational Unit where Inactive users from PeopleSoft are placed. You can modify this parameter through a GCV after installation.
Active Employees Group	The name of the Group Object to which Active Employee users from PeopleSoft are added. You can modify this parameter through a GCV after installation.
Active Managers Group	The name of the Group Object to which Active Manager users from PeopleSoft are added. You can modify this parameter through a GCV after installation.

Parameter	Description
PeopleSoft Connection String	<p>The hostname or IP address and port number for connecting to the appropriate PeopleSoft Application server. This is typically referred to as the PeopleSoft application server connection string. The default port is 9000.</p> <p>The connection string uses the following format:            &lt;hostname or IP address&gt;:&lt;Jolt Port Number&gt;            Example: //PSServer:9000</p> <p>To enable failover and loadbalancing, you can supply multiple server connection strings separated by a comma. Example: //PSServer:9000,//111.222.3.4:9000</p>
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. Re-enter the password.
Password Failure Notification User	Password synchronization policies are configured to send e-mail notifications to the associated user when password updates fail. This option allows you to send a copy of the notification e-mail to another user, such as a security administrator. If you want to send a copy, specify the DN of that user. Otherwise, leave the field blank.
Allow 'add' events	Subscriber Add events are implemented by invoking the Component Interface 'Create' method (if present). If you want the driver to allow Subscriber channel add events, select <i>Allow Subscriber add</i> .
Allow 'delete' events	Subscriber Delete events are implemented by invoking the Component Interface Delete method (if present). If you want the driver to allow Subscriber channel delete events, select <i>Allow Subscriber delete</i> .
Driver is Local/ Remote	Configure the driver for use with the Remote Loader service by selecting the <i>Remote</i> option, or select <i>Local</i> 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 <i>Local</i> is selected, you can skip the remaining parameters.
Remote Host Name and Port	Specify the hostname 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 Metadirectory 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).

**Table 3-4** *Additional Driver Parameters*

<b>Parameter</b>	<b>Description</b>	<b>Default Value</b>
Schema CI Name	List of the names of the PeopleSoft CI objects that define 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.	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.	Disallow Subscriber add
Data Record ID Field Default Value	Allows an administrator to specify the default value for the Schema CI key field. Only used if Subscriber channel Add events are allowed.	NEW
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.	Disallow Subscriber delete
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.	NPSDriver1
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

## 3.5 Activating the Driver

Novell Identity Manager, Integration Modules, and the Provisioning Module must be activated within 90 days of installation, or they shut down. At any time during the 90 days, or afterward, you can choose to activate Identity Manager products.

For more information, refer to “[Activating Novell Identity Manager Products](#)” in the *Identity Manager 3.5.1 Installation Guide*.





# Customizing the Driver

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

- ♦ [Section 4.1, “Customizing the PSA by Triggering Transactions,” on page 49](#)
- ♦ [Section 4.2, “Changing the Driver by Changing the Data Schema Component Interface,” on page 51](#)
- ♦ [Section 4.3, “Customizing the Driver by Modifying Driver Policies,” on page 54](#)

## 4.1 Customizing the PSA by Triggering Transactions

Transaction record creation is triggered by PeopleCode associated with modifications and additions to data on the Schema Data record (DIRXML\_S\_PERS) and row Delete events on the Staging record (DIRXML\_SCHEMA01). If desired, the PeopleSoft administrator or consultant can use these examples to trigger transactions based on other events within the PeopleSoft application.

---

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

---

The default Transaction creation function is defined on the DIRXML\_DRIVER field of the DIRXML\_DERIVED Record definition:

A PeopleCode function call would be as follows:

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

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

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

"A") ;

**Table 4-1** *Function Call Parameter Definitions*

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

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

## 4.2 Changing the Driver by Changing the Data Schema Component Interface

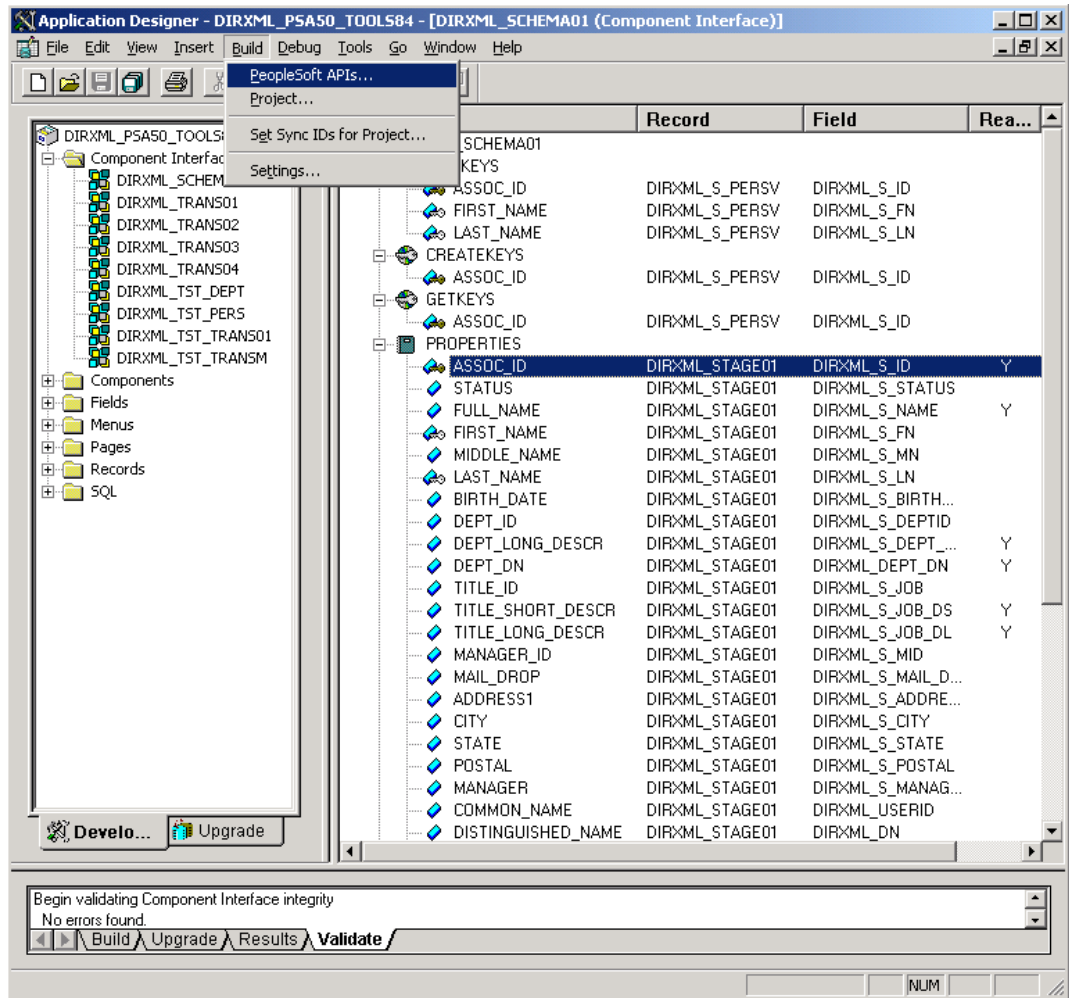
The driver is preconfigured to use the Component Interfaces defined in the PSA. The APIs for these CIs have been compiled and combined into the `dirxmlcomps.jar` file. At run time, the driver imports the interfaces required to interact with the appropriate CI.

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.

### 4.2.1 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.



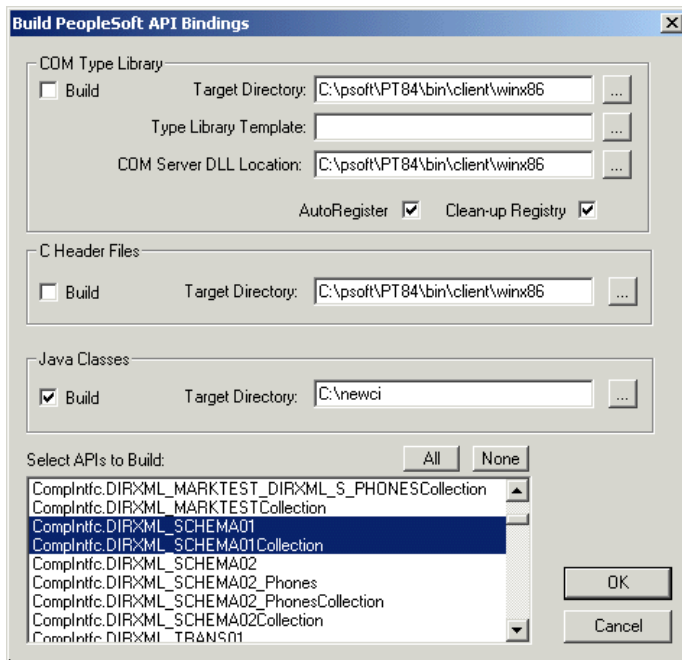
- 3 From the Build PeopleSoft API Bindings dialog box, select the *build* option for the Java classes.
- 4 Select a target directory for the Java CI APIs (For our example, *C:\newci*).
- 5 For the *Select APIs to Build* prompt, click *None* to deselect all CIs.
- 6 Using the scroll area, select the CIs for which you wish to generate an API. Make sure you select all CIs that begin with the name of the desired interface. In this example we selected *CompIntfc.DIRXML\_SCHEMA01* and *CompIntfc.DIRXML\_SCHEMA01Collection*.

---

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

---

- 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.

## 4.2.2 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.

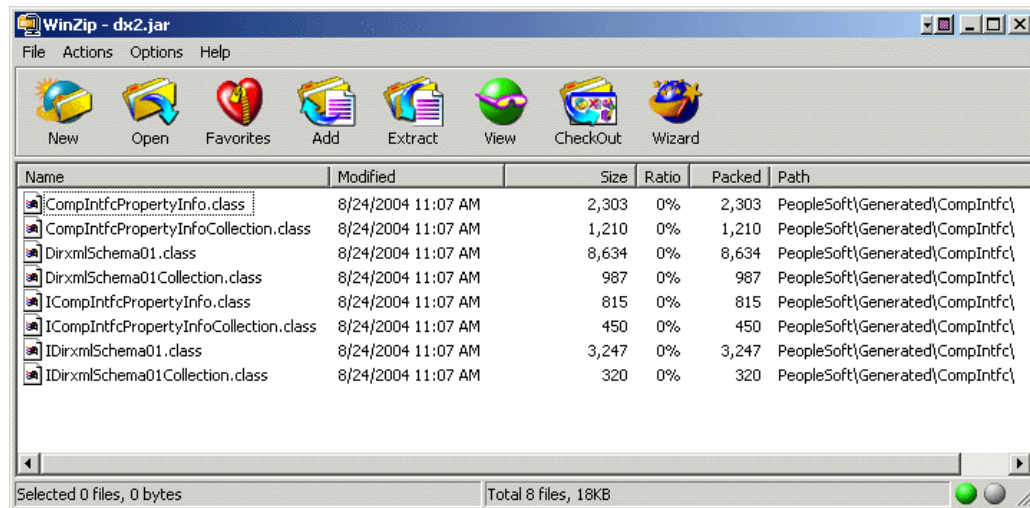
## 4.2.3 Building the CI API JAR File

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

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

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

**Figure 4-1** Building a new `.jar` file



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

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

## 4.3 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:

- ◆ [Section 4.3.1, “Modifying the Driver Mapping Policy,” on page 55](#)
- ◆ [Section 4.3.2, “Using the Schema Query to Refresh the PeopleSoft Schema CI,” on page 55](#)
- ◆ [Section 4.3.3, “Publisher Channel Objects,” on page 55](#)
- ◆ [Section 4.3.4, “Understanding the Publisher Filter,” on page 56](#)
- ◆ [Section 4.3.5, “Publisher Filter Attributes,” on page 56](#)
- ◆ [Section 4.3.6, “Securing the Data,” on page 57](#)
- ◆ [Section 4.3.7, “Publisher Object Policies,” on page 57](#)
- ◆ [Section 4.3.8, “Input Transformation Policy,” on page 57](#)
- ◆ [Section 4.3.9, “Matching Policy,” on page 58](#)
- ◆ [Section 4.3.10, “Create Policy,” on page 58](#)
- ◆ [Section 4.3.11, “Placement Policy,” on page 58](#)
- ◆ [Section 4.3.12, “Command Transformation Policy,” on page 59](#)

- ◆ [Section 4.3.13, “Subscriber Channel Objects,” on page 61](#)
- ◆ [Section 4.3.14, “Understanding the Subscriber Filter,” on page 62](#)
- ◆ [Section 4.3.15, “Securing the Data,” on page 63](#)
- ◆ [Section 4.3.16, “Modifying the Filter,” on page 63](#)
- ◆ [Section 4.3.17, “Subscriber Object Policies,” on page 63](#)

### 4.3.1 Modifying the Driver Mapping Policy

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

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

The default Data Schema CI is called DIRXML\_SCHEMA01 and represents a set of employee data. The data fields in this CI are mapped to similar attributes of the eDirectory User object.

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

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

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

### 4.3.2 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, the CI is renamed, or additional Data Schema CIs are added to the driver configuration, it is necessary to refresh the PeopleSoft application schema definition and re-map affected attributes.

### 4.3.3 Publisher Channel Objects

The Publisher object contains a filter and a set of policies. Policies are necessary for converting data from the PeopleSoft CI into XDS format. The driver then submits the data to the Metadirectory

engine. The engine applies the Publisher filter to the data and applies the business logic defined by the Publisher policies prior to submitting the data to Identity Vault.

### 4.3.4 Understanding the Publisher Filter

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

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

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

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

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

### 4.3.5 Publisher Filter Attributes

By default, PeopleSoft applications are considered to be highly authoritative. Therefore, most of the field names in the default DIRXML\_SCHEMA01 Component Interface are passed through the Publisher filter. As noted earlier, the names of attributes in the filter are eDirectory attribute names that have been mapped via the Mapping policies.

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

**Table 4-2** *The Publisher Channel Attributes*

departmentNumber	OU
employeeStatus	pager
Full Name	Physical Delivery Office Name
Given Name	Postal Code
homePhone	S
Initials	SA
isManager	Surname



---

jobCode	Telephone Number
mailstop	Title
managerWorkforceID	workforceID
mobile	

---

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

### 4.3.6 Securing the Data

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

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

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

### 4.3.7 Publisher Object Policies

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

- ◆ [Section 4.3.8, “Input Transformation Policy,” on page 57](#)
- ◆ [Section 4.3.9, “Matching Policy,” on page 58](#)
- ◆ [Section 4.3.10, “Create Policy,” on page 58](#)
- ◆ [Section 4.3.11, “Placement Policy,” on page 58](#)
- ◆ [Section 4.3.12, “Command Transformation Policy,” on page 59](#)

Policies contain rules or templates that perform specific operations that can manipulate data, query either Identity Vault or PeopleSoft for additional data required for processing, create new attributes based on values of other attributes, or even discard entire data events. The following section explains each policy and describes the operations of each policy or template. Because XML, DirXML Script, and XSLT allow for great flexibility, all policies can be modified to meet the individual needs of your organization. The Mapping policy has been previously described and is addressed in this section under [“Modifying the Driver Mapping Policy” on page 55](#).

### 4.3.8 Input Transformation Policy

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

transformation contained in this policy is the transformation of character attributes in PeopleSoft to Boolean attributes in eDirectory.

### **Manager Flag Data Transformation Template**

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

### **4.3.9 Matching Policy**

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

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

The default Publisher Matching policy is a DirXML Script policy that attempts to match eDirectory User objects containing the same value in the workforceID attribute (mapped from the DIRXML\_SCHEMA01 attribute). A secondary policy attempts to match using the Surname and Given Name attribute.

### **4.3.10 Create Policy**

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

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

### **4.3.11 Placement Policy**

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

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

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

### 4.3.12 Command Transformation Policy

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

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

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

As the final transformation policy, the Command Transformation policy provides an excellent location to define operations that must be applied without the risk of further event transformation, thus allowing complicated policy processing to be programmed in one location. 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 <operation-data> element to facilitate the addition of the CN and DN attributes in PeopleSoft.

### **match <modify> element**

This template does the following:

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

### **get-empl-status**

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

### **get-empl-isManager**

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

### **get-empl-CN**

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

### **get-empl-managerWorkforceID**

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

### **get-empl-ID**

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

### **set-manager-on-user**

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

### **set-manager-on-direct-reports**

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

### **clear-manager-on-direct-reports**

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

### **set-directReports-on-manager**

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

### **clear-directReports-on-manager**

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

### **set-directReports-on-user**

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

## **4.3.13 Subscriber Channel Objects**

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

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

### 4.3.14 Understanding the Subscriber Filter

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

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

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

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

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

#### Subscriber Filter Attributes

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

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

**Table 4-3** *The Subscriber Channel Attributes*

CN	managerWorkforceID
departmentNumber	mobile
Description	pager
employeeStatus	Physical Delivery Office Name

---

Given Name	Postal Code
homePhone	S
Initials	SA
Internet EMail Address	Surname
jobCode	Telephone Number
mailstop	workforceID

---

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

### 4.3.15 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.

### 4.3.16 Modifying the Filter

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

### 4.3.17 Subscriber Object Policies

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

- ◆ [“Event Transformation Policy” on page 64](#)
- ◆ [“Matching Policy” on page 64](#)
- ◆ [“Create Policy” on page 64](#)
- ◆ [“Output Transformation Policy” on page 65](#)

Policies contain templates that perform specific operations that can manipulate data, query either Identity Vault or PeopleSoft for additional data required for processing, create new attributes based on values of other attributes, or even discard entire data events. The following section explains each policy and provides a description of the operations each policy performs. Because XML, DirXML Script, and XSLT allow for great flexibility, all policies can be modified to meet the individual needs of your organization. The Schema Mapping policy has been previously described and is not be addressed in this section. For information on the Schema Mapping policy, refer to [“Modifying the Driver Mapping Policy” on page 55](#).

## 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 CN and DISTINGUISHED\_NAME fields in the CI.

### Match <move> Element

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

### Match <delete> Element

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

## Matching Policy

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

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

The default Subscriber Matching policy is an XML policy that attempts to find a PeopleSoft DIRXML\_SCHEMA01 object that contains an ASSOC\_ID attribute that matches the eDirectory User object's workforceID attribute. If that policy fails, the driver uses another policy to locate a PeopleSoft DIRXML\_SCHEMA01 secondary object with a matching Given Name and Surname. The policy is defined in eDirectory class and attribute names because schema mapping has not yet been applied.

## Create Policy

The Create policy specifies the criteria for creating a new object after the Matching policy has failed to find a match. This policy performs various tests and transformations based on the requirements for object creation in the DIRXML\_SCHEMA01 CI in PeopleSoft and the business logic being applied.



The default Subscriber Create policy is a DirXML Script policy that asserts that the Add document is for a User object and that it must contain a value for Given Name, Surname, employeeStatus, departmentNumber, and jobCode. These fields are required because the default PSA has defined these fields as required in the DIRXML\_SCHEMA01 CI. (Additional required fields, such as BirthDate and Manager, have defined default values in the CI and are thus not required here.) By making sure all required fields are present before submitting the Add event to the driver, synchronization performance and integrity is enhanced.

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

## **Output Transformation Policy**

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

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

## **Write-back**

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

## **From-merge Write-back**

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

## **Manager Flag Data Transformation**

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

## **Add DN Value to Subscribe Adds**

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



# Upgrading the Driver

- ♦ [Section 5.1, “Changes in Policy Architecture,” on page 67](#)
- ♦ [Section 5.2, “Upgrading the Driver in Designer,” on page 67](#)
- ♦ [Section 5.3, “Upgrading the Driver in iManager,” on page 69](#)

## 5.1 Changes in Policy Architecture

Identity Manager 3.5 and 3.5.1 contain a new policy architecture, which affects how drivers reference policies. While the 3.5.1 driver architecture offers increased functionality in the Identity Manager 3.5.1 environment, the 3.0.x Metadirectory engine cannot run 3.5.1 driver configurations.

However, Identity Manager 3.5 and 3.5.1 can run 3.0.x driver configurations. If you have 3.0.x driver configurations associated with both 3.0.x and 3.5.1 Metadirectory engines, do not upgrade the 3.0.x drivers. The 3.0.x driver configurations work in a 3.5.1 environment, but they don't have the increased functionality that Identity Manager 3.5 and above affords. When 3.0.x driver configurations are only associated with a 3.5 or later Metadirectory engine, you should then upgrade the 3.0.x drivers to 3.5.1.

For more information on policy architecture and upgrading drivers to 3.5.1, see [“Upgrading Identity Manager Policies”](#) in *Understanding Policies for Identity Manager 3.5.1*.

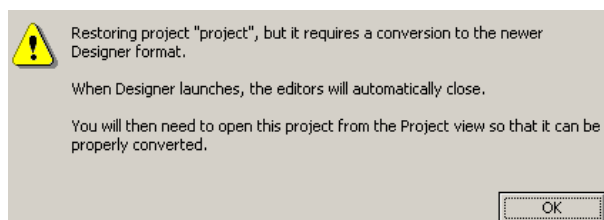
If you are upgrading from Identity Manager 3.5.0 to Identity Manager 3.5.1, the following information does not apply.

## 5.2 Upgrading the Driver in Designer

- 1 Make sure you have updated your driver with all the patches for the version you are currently running.

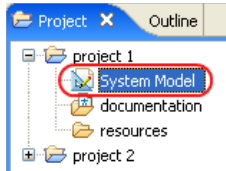
We recommend this step for all drivers, to help minimize upgrade issues.

- 2 Back up the driver. See [Chapter 10, “Backing Up the Driver,” on page 107](#) for instruction on how to back up the driver.
- 3 Install Designer version 2.0 or above, then launch Designer.  
If you had a project open in Designer when you upgraded Designer, proceed to [Step 4](#). If you didn't have a project open in Designer when you upgraded Designer, skip to [Step 5](#).
- 4 If you had a project open when upgrading Designer, the following warning message is displayed. Read the warning message, then click *OK*.

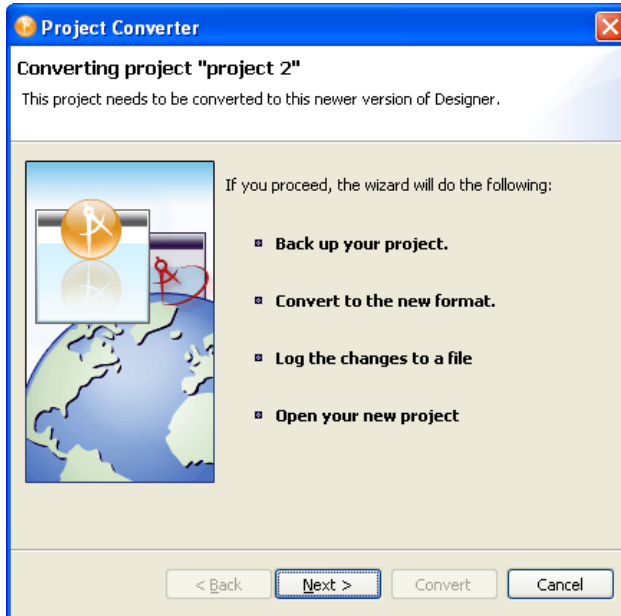


Designer closes the project to preform the upgrade.

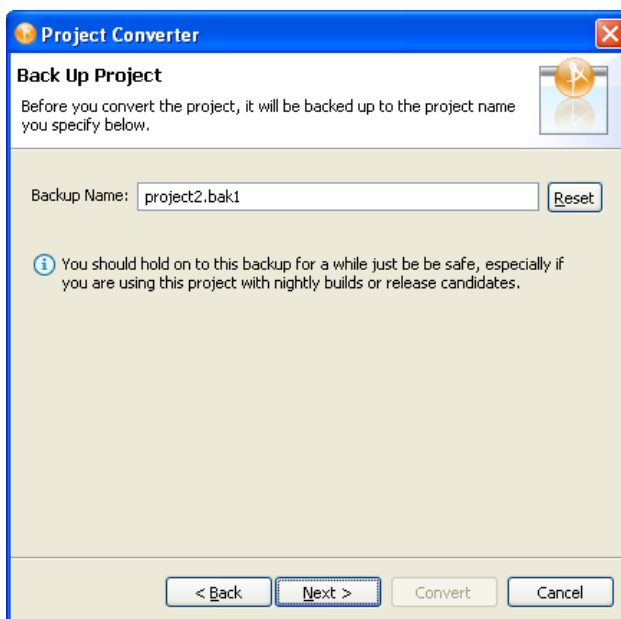
- 5 In the Project view, double-click *System Model* to open and convert the project.



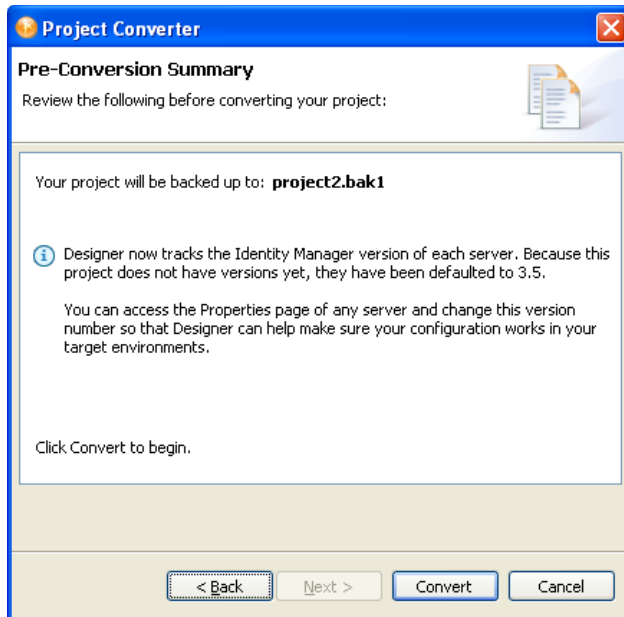
- 6 Read the Project Converter message explaining that the project is backed up, converted to the new format, changes logged to a file, and the new project is opened, then click *Next*.



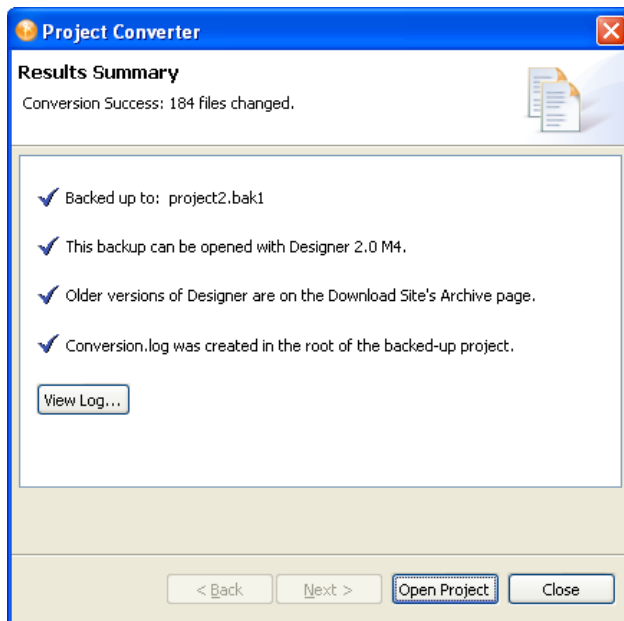
- 7 Specify the name of the backup project name, then click *Next*.



8 Read the project conversion summary, then click *Convert*.



9 Read the project conversion result summary, then click *Open Project*.



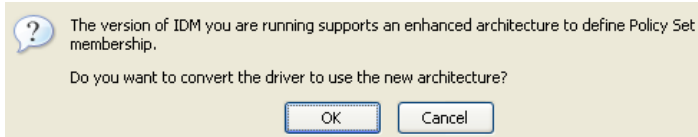
If you want to view the log file that is generated, click *View Log*.

## 5.3 Upgrading the Driver in iManager

1 Make sure you have updated your driver with all the patches for the version you are currently running.

We recommend this step for all drivers, to help minimize upgrade issues.

- 2 Back up the driver. See [Chapter 10, “Backing Up the Driver,”](#) on page 107 for instruction on how to back up the driver.
- 3 Verify that Identity Manager 3.5.1 has been installed and you have the current plug-ins installed, then launch iManager.
- 4 Click *Identity Manager > Identity Manager Overview*.
- 5 Click *Search* to find the Driver Set object, then click the driver you want to upgrade.
- 6 Read the message that is displayed, then click *OK*.



- 7 If there is more than one driver to upgrade, repeat [Step 2](#) through [Step 6](#).

# Activating the Driver

# 6

Novell® Identity Manager, Integration Modules, and the Provisioning Module must be activated within 90 days of installation, or they shut down. At any time during the 90 days, or afterward, you can choose to activate Identity Manager products.

To activate the driver, see “[Activating Novell Identity Manager Products](#)” in the *Identity Manager 3.5.1 Installation Guide*.





# Managing the Driver

The driver can be managed through Designer, iManager, or the DirXML<sup>®</sup> Command Line utility.

- ♦ [Section 7.1, “Starting, Stopping, or Restarting the Driver,” on page 73](#)
- ♦ [Section 7.2, “Migrating and Resynchronizing Data,” on page 74](#)
- ♦ [Section 7.3, “Using the DirXML Command Line Utility,” on page 74](#)
- ♦ [Section 7.4, “Viewing Driver Versioning Information,” on page 75](#)
- ♦ [Section 7.5, “Reassociating a Driver Set Object with a Server Object,” on page 79](#)
- ♦ [Section 7.6, “Changing the Driver Configuration,” on page 79](#)
- ♦ [Section 7.7, “Storing Driver Passwords Securely with Named Passwords,” on page 80](#)
- ♦ [Section 7.8, “Adding a Driver Heartbeat,” on page 86](#)

## 7.1 Starting, Stopping, or Restarting the Driver

- ♦ [Section 7.1.1, “Starting the Driver in Designer,” on page 73](#)
- ♦ [Section 7.1.2, “Starting the Driver in iManager,” on page 73](#)
- ♦ [Section 7.1.3, “Stopping the Driver in Designer,” on page 73](#)
- ♦ [Section 7.1.4, “Stopping the Driver in iManager,” on page 73](#)
- ♦ [Section 7.1.5, “Restarting the Driver in Designer,” on page 74](#)
- ♦ [Section 7.1.6, “Restarting the Driver in iManager,” on page 74](#)

### 7.1.1 Starting the Driver in Designer

- 1 Open a project in the Modeler, then right-click the driver line.
- 2 Select *Live > Start Driver*.

### 7.1.2 Starting the Driver in iManager

- 1 In iManager, click *Identity Manager > Identity Manager Overview*.
- 2 Browse to the driver set where the driver exists, then click *Search*.
- 3 Click the upper right corner of the driver icon, then click *Start driver*.

### 7.1.3 Stopping the Driver in Designer

- 1 Open a project in the Modeler, then right-click the driver line.
- 2 Select *Live > Stop Driver*.

### 7.1.4 Stopping the Driver in iManager

- 1 In iManager, click *Identity Manager > Identity Manager Overview*.

- 2 Browse to the driver set where the driver exists, then click *Search*.
- 3 Click the upper right corner of the driver icon, then click *Stop driver*.

### 7.1.5 Restarting the Driver in Designer

- 1 Open a project in the Modeler, then right-click the driver line.
- 2 Select *Live > Restart Driver*.

### 7.1.6 Restarting the Driver in iManager

- 1 In iManager, click *Identity Manager > Identity Manager Overview*.
- 2 Browse to the driver set where the driver exists, then click *Search*.
- 3 Click the upper right corner of the driver icon, then click *Restart driver*.

## 7.2 Migrating and Resynchronizing Data

Identity Manager synchronizes data when the data changes. If you want to synchronize all data immediately, you can choose from the following options:

- ♦ **Migrate Data from Identity Vault:** Allows you to select containers or objects you want to migrate from the Identity Vault to an application. When you migrate an object, the Identity Manager engine applies all of the Matching, Placement, and Create policies, as well as the Subscriber filter, to the object.
- ♦ **Migrate Data into Identity Vault:** Assumes that the remote application (usually a Web Service) can be queried for entries that match the criteria in the publisher filter. However, because of the general nature of the PeopleSoft driver the method for querying the Web Service (if there is one) is not known to the driver shim. Therefore, this feature does not usually work with the PeopleSoft driver.
- ♦ **Synchronize:** The Identity Manager engine looks in the Subscriber class filter and processes all objects for those classes. Associated objects are merged. Unassociated objects are processed as Add events.

To use one of the options explained above:

- 1 In iManager, click *Identity Manager > Identity Manager Overview*.
- 2 Browse to and select the driver set where the driver exists, then click *Search*.
- 3 Click the driver icon.
- 4 Click the appropriate migration button.

For more information, see [Chapter 8, “Synchronizing Objects,” on page 89](#).

## 7.3 Using the DirXML Command Line Utility

The DirXML Command Line utility provides command line access to manage the driver. This utility is not a replacement for iManager or Designer. The primary use of this utility is to allow you to create platform-specific scripts to manage the driver.

For example, you could create a shell script on Linux to check the status of the driver. See [Appendix A, “DirXML Command Line Utility,”](#) on page 111 for detailed information about the DirXML Command Line utility. For daily tasks, use iManager or Designer.

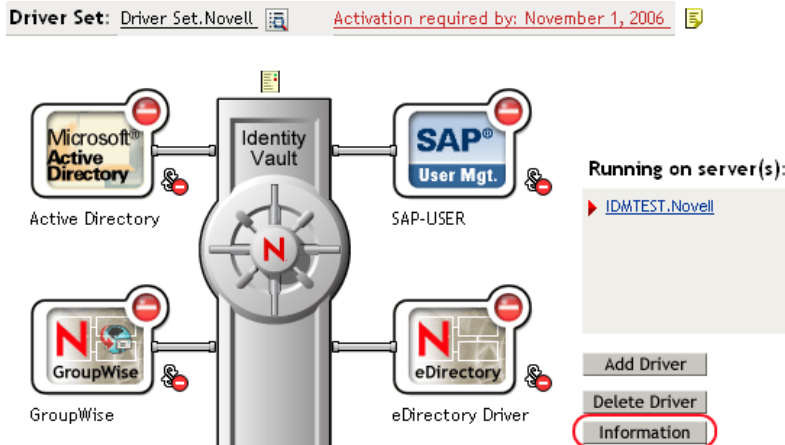
## 7.4 Viewing Driver Versioning Information

The Versioning Discovery tool only exists in iManager.

- ♦ [Section 7.4.1, “Viewing a Hierarchical Display of Versioning Information,”](#) on page 75
- ♦ [Section 7.4.2, “Viewing the Versioning Information As a Text File,”](#) on page 76
- ♦ [Section 7.4.3, “Saving Versioning Information,”](#) on page 78

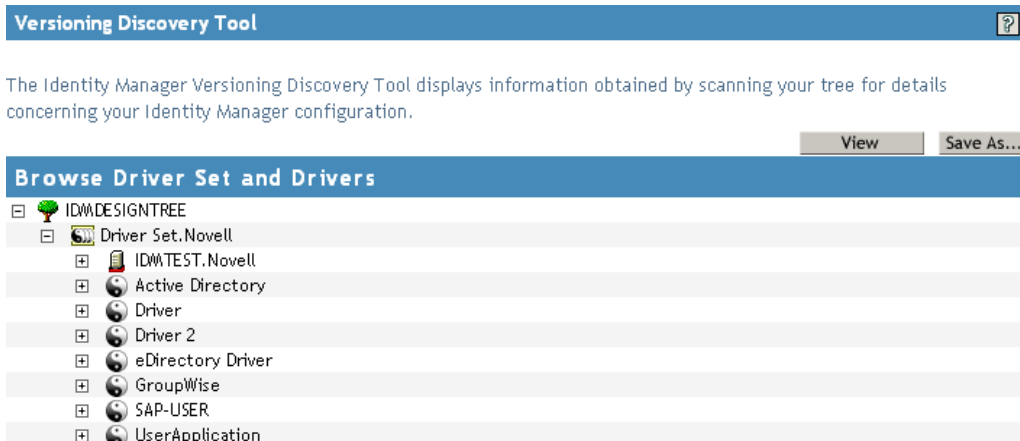
### 7.4.1 Viewing a Hierarchical Display of Versioning Information

- 1 To find your Driver Set object in iManager, click *Identity Manager > Identity Manager Overview*, then click *Search*.
- 2 In the Identity Manager Overview, click *Information*.



You can also select *Identity Manager Utilities > Versions Discovery*, browse to and select the Driver Set object, then click *OK*.

- 3 View a top-level or unexpanded display of versioning information.



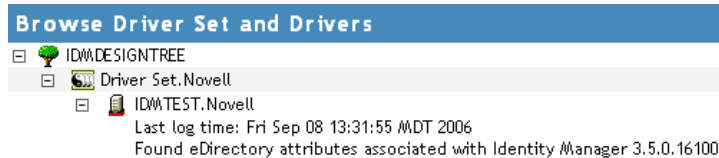
The unexpanded hierarchical view displays the following:

- ◆ The eDirectory™ tree that you are authenticated to
- ◆ The Driver Set object that you selected
- ◆ Servers that are associated with the Driver Set object

If the Driver Set object is associated with two or more servers, you can view Identity Manager information on each server.

- ◆ Drivers

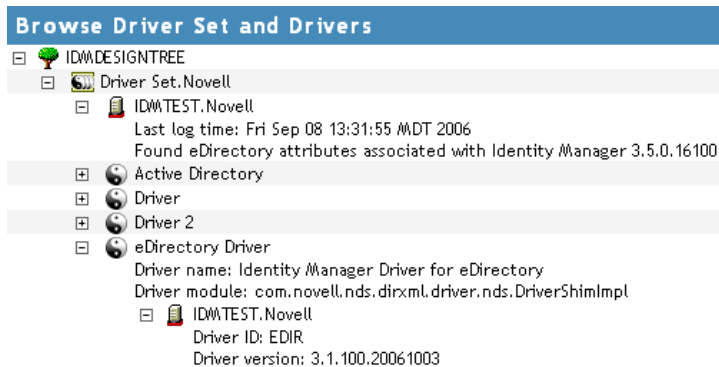
#### 4 View versioning information related to servers by expanding the server icon.



The expanded view of a top-level server icon displays the following:

- ◆ Last log time
- ◆ Version of Identity Manager that is running on the server

#### 5 View versioning information related to drivers by expanding the driver icon.



The expanded view of a top-level driver icon displays the following:

- ◆ The driver name
- ◆ The driver module (for example, com.novell.nds.dirxml.driver.delimitedtext.DelimitedTextDriver)

The expanded view of a server under a driver icon displays the following:

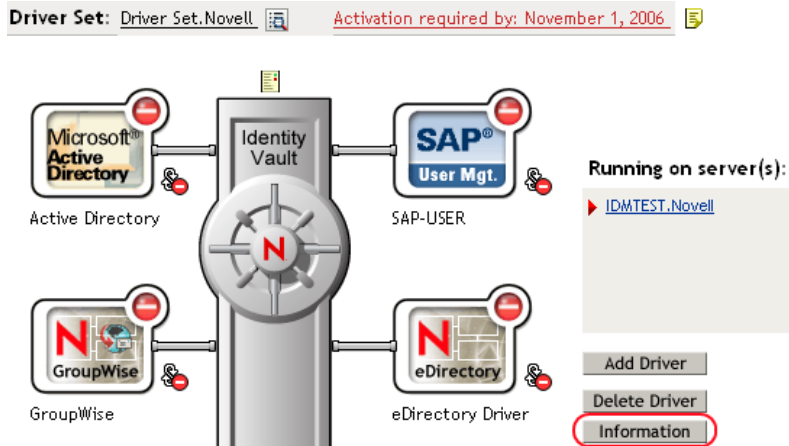
- ◆ The driver ID
- ◆ The version of the instance of the driver running on that server

## 7.4.2 Viewing the Versioning Information As a Text File

Identity Manager publishes versioning information to a file. You can view this information in text format. The textual representation is the same information contained in the hierarchical view.

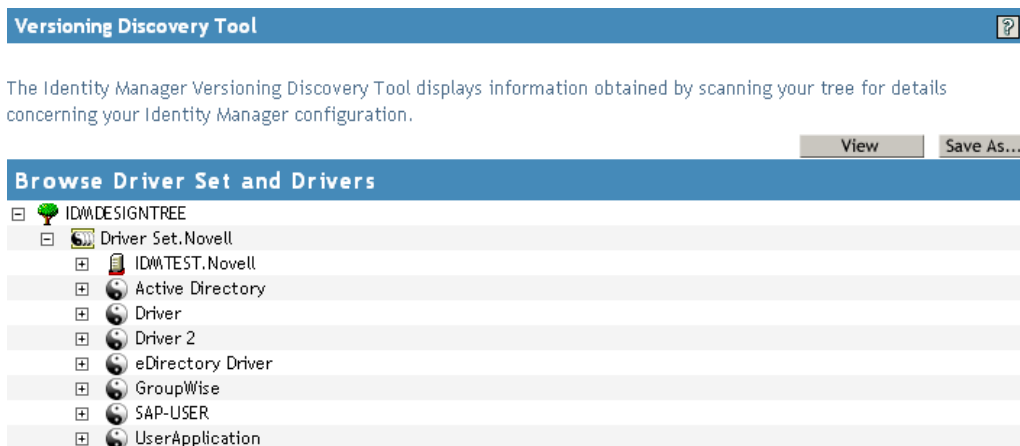
- 1 To find your Driver Set object in iManager, click *Identity Manager > Identity Manager Overview*, then click *Search*.

2 In the Identity Manager Overview, click *Information*.



You can also select *Identity Manager Utilities > Versioning Discovery*, browse to and select the Driver Set object, then click *Information*.

3 In the Versioning Discovery Tool dialog box, click *View*.



The information is displayed as a text file in the Report Viewer window.

## Versioning Discovery Tool - Report Viewer

```
Identity Manager Version Discovery Tool v2.0
Novell, Inc. Copyright 2003, 2004

Version Query started Saturday, January 20, 2007 11:02:52 AM MST

Parameter Summary:
  Default server's DN:  IDMTEST.Novell
  Default server's IP address:  137.65.151.208
  Logged in as admin, context Novell
  Tree name:  IDMDESIGNTREE
  Found 7 Identity Manager Drivers

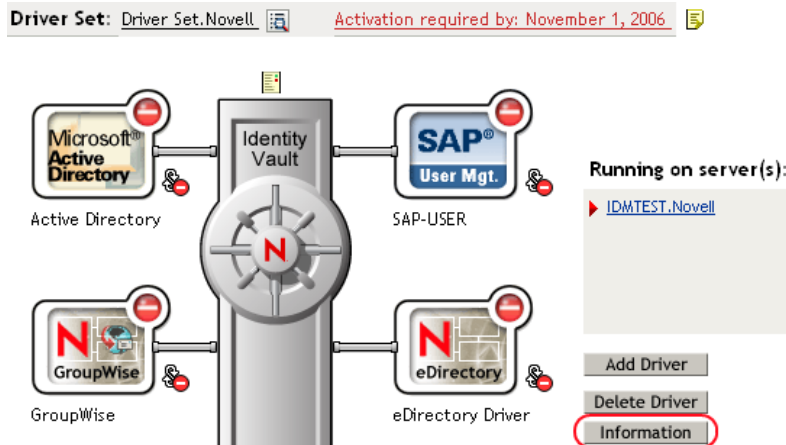
Driver Set:  Driver Set.Novell
  Driver Set running on Identity Vault:  IDMTEST.Novell
    Last log time:  Fri Sep 08 13:31:55 MDT 2006
    Found eDirectory attributes associated with Identity Manager 3.5.0.1
  Driver:  Active Directory.Driver Set.Novell
    Driver name:  Identity Manager Driver for Active Directory and Excha
    Driver module:  addriver.dll
    Driver Set running on Identity Vault:  IDMTEST.Novell
      Didn't find any DirXML-DriverVersion attributes associated w
      This may mean the Metadirectory engine is older than
      It does not indicate anything about the version of t
  Driver:  Driver.Driver Set.Novell
    Driver name:  Identity Manager Driver for Peoplesoft
    Driver module:  NPSSHim.dll
    Driver Set running on Identity Vault:  IDMTEST.Novell
```

OK

### 7.4.3 Saving Versioning Information

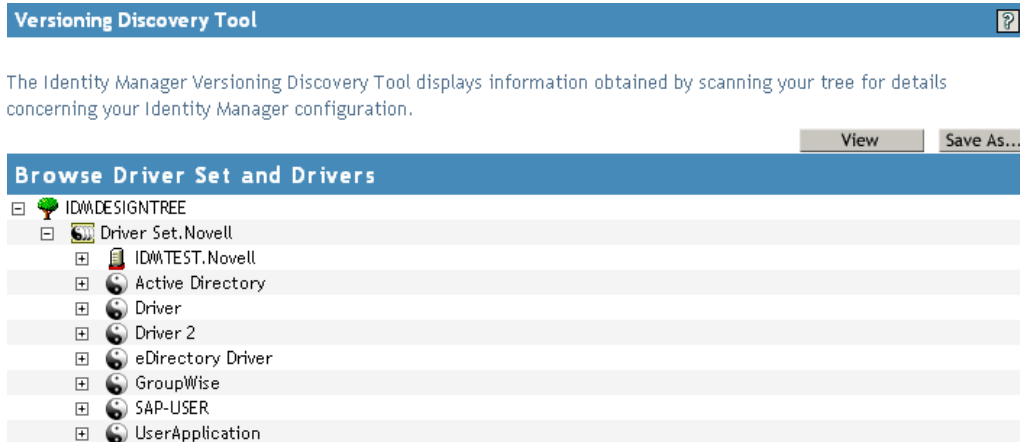
You can save versioning information to a text file on your local or network drive.

- 1 To find the Driver Set object in iManager, click *Identity Manager > Identity Manager Overview*, then click *Search*.
- 2 In the Identity Manager Overview, click *Information*.



You can also select *Identity Manager Utilities > Versioning Discovery*, browse to and select the Driver Set object, then click *Information*.

- 3 In the Versioning Discovery Tool dialog box, click *Save As*.



- 4 In the File Download dialog box, click *Save*.
- 5 Navigate to the desired directory, type a filename, then click *Save*.

Identity Manager saves the data to a text file.

## 7.5 Reassociating a Driver Set Object with a Server Object

The driver set object should always be associated with a server object. If the driver set is not associated with a server object, none of the drivers in the driver set can start.

If the link between the driver set object and the server object becomes invalid, you see one of the following conditions:

- ♦ When upgrading eDirectory your Identity Manager server, you get the error UniqueSPIException error -783.
- ♦ No server is listed next to the driver set in the Identity Manager Overview window.
- ♦ A server is listed next to the driver set in the Identity Manager Overview window, but the name is garbled text.

To resolve this issue, disassociate the driver set object and the server object, then reassociate them.

- 1 In iManager click *Identity Manager > Identity Manager Overview*, then click *Search* to find the driver set object that the driver should be associated with.
- 2 Click the *Remove server* icon, then click *OK*.
- 3 Click the *Add server* icon, then browse to and select the server object.
- 4 Click *OK*.

## 7.6 Changing the Driver Configuration

If you need to change the driver configuration, Identity Manager allows you to make the change through iManager or Designer.

To change the driver configuration in iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties*.

To change the driver configuration in Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties*.

For a listing of all of the configuration fields, see [Appendix B, “Properties of the Driver,” on page 125](#).

## 7.7 Storing Driver Passwords Securely with Named Passwords

Identity Manager allows you to store multiple passwords securely for a particular driver. This functionality is referred to as Named Passwords. Each different password is accessed by a key, or name.

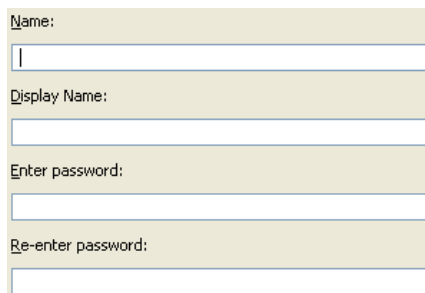
You can also use the Named Passwords feature to store other pieces of information securely, such as a user name.

To use a Named Password in a driver policy, you refer to it by the name of the password, instead of using the actual password, and the Metadirectory engine sends the password to the driver. The method described in this section for storing and retrieving Named Passwords can be used with any driver without making changes to the driver shim.

- ♦ [Section 7.7.1, “Using Designer to Configure Named Passwords,” on page 80](#)
- ♦ [Section 7.7.2, “Using iManager to Configure Named Passwords,” on page 81](#)
- ♦ [Section 7.7.3, “Using Named Passwords in Driver Policies,” on page 82](#)
- ♦ [Section 7.7.4, “Using the DirXML Command Line Utility to Configure Named Passwords,” on page 83](#)

### 7.7.1 Using Designer to Configure Named Passwords

- 1 Right-click the driver object, then select *Properties*.
- 2 Select *Named Password*, then click *New*.



Name:

Display Name:

Enter password:

Re-enter password:

- 3 Specify the *Name* of the Named Password.



- 4 Specify the *Display name* of the Named Password.
- 5 Specify the Named Password, then re-enter the password.
- 6 Click *OK* twice.

## 7.7.2 Using iManager to Configure Named Passwords

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 In the Identity Manager Overview, click the upper right corner of the driver icon, then click *Edit properties*.
- 3 On the Modify Object page on the Identity Manager tab, click *Named Passwords*.

The Named Passwords page appears, listing the current Named Passwords for this driver. If you have not set up any Named Passwords, the list is empty.



- 4 To add a Named Password, click *Add*, complete the fields, then click *OK*.

#### **Named Password**

Named Passwords lets you securely store multiple passwords for a driver. Instead of including a password in clear text in a driver policy, you can configure the policy to request a Named Password.

Name:

Display name:

Enter password:

Reenter password:

OK

Cancel

- 5 Specify a name, display name and a password, then click *OK* twice.  
You can use this feature to store other kinds of information securely, such as a username.
- 6 Click *OK* to restart the driver and have the changes take effect.
- 7 To remove a Named Password, select the password name, then click *Remove*.  
The password is removed without prompting you to confirm the action.

### 7.7.3 Using Named Passwords in Driver Policies

- ♦ [“Using the Policy Builder” on page 82](#)
- ♦ [“Using XSLT” on page 83](#)

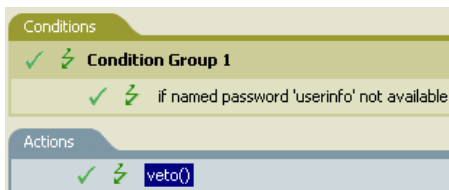
#### Using the Policy Builder

Policy Builder allows you to make a call to a Named Password. Create a new rule and select Named Password as the condition, then set an action depending upon if the Named Password is available or not available.

- 1 In Designer, launch Policy Builder, right-click, then click *New > Rule*.
- 2 Specify the name of the rule, then click *Next*.
- 3 Select the condition structure, then click *Next*.
- 4 Select *Named Password* for the *Condition*.
- 5 Browse to and select the Named Password that is stored on the driver.  
In this example, it is *userinfo*.
- 6 Select whether the Operator is available or not available.
- 7 Select an action for the *Do* field.  
In this example, the action is *veto*.

The example indicates that if the *userinfo* Named Password is not available, then the event is vetoed.

Figure 7-1 A Policy Using Named Passwords



## Using XSLT

The following example shows how a named password can be referenced in a driver policy on the Subscriber channel in XSLT:

```
<xsl:value-of
select="query:getNamedPassword($srcQueryProcessor, 'mynamedpassword') "
xmlns:query="http://www.novell.com/java/
com.novell.nds.dirxml.driver.XdsQueryProcessor/>
```

## 7.7.4 Using the DirXML Command Line Utility to Configure Named Passwords

- [“Creating a Named Password in the DirXML Command Line Utility” on page 83](#)
- [“Using the DirXML Command Line Utility to Remove a Named Password” on page 84](#)

### Creating a Named Password in the DirXML Command Line Utility

- 1 Run the DirXML Command Line utility.

For information, see [Appendix A, “DirXML Command Line Utility,” on page 111](#).

- 2 Enter your username and password.

The following list of options appears.

```
DirXML commands
1: Start driver
2: Stop driver
3: Driver operations...
4: Driver set operations...
5: Log events operations...
6: Get DirXML version
7: Job operations...
99: Quit
Enter choice:
```

- 3 Enter 3 for driver operations.

A numbered list of drivers appears.

- 4 Enter the number for the driver you want to add a Named Password to.

The following list of options appears.

```
Select a driver operation for:
driver_name
1: Start driver
2: Stop driver
```

```
3: Get driver state
4: Get driver start option
5: Set driver start option
6: Resync driver
7: Migrate from application into DirXML
8: Submit XDS command document to driver
9: Submit XDS event document to driver
10: Queue event for driver
11: Check object password
12: Initialize new driver object
13: Passwords operations
14: Cache operations
99: Exit
Enter choice:
```

**5** Enter 13 for password operations.

The following list of options appears.

Select a password operation

```
1: Set shim password
2: Reset shim password
3: Set Remote Loader password
4: Clear Remote Loader password
5: Set named password
6: Clear named password(s)
7: List named passwords
8: Get passwords state
99: Exit
```

Enter choice:

**6** Enter 5 to set a new Named Password.

The following prompt appears:

Enter password name:

**7** Enter the name by which you want to refer to the Named Password.

**8** Enter the actual password that you want to secure at the following prompt:

Enter password:

The characters you type for the password are not displayed.

**9** Confirm the password by entering it again at the following prompt:

Confirm password:

**10** After you enter and confirm the password, you are returned to the password operations menu.

**11** After completing this procedure, you can use the 99 option twice to exit the menu and quit the DirXML Command Line Utility.

### Using the DirXML Command Line Utility to Remove a Named Password

This option is useful if you no longer need named passwords that you previously created.

**1** Run the DirXML Command Line utility.

For information, see [Appendix A, “DirXML Command Line Utility,”](#) on page 111.

**2** Enter your username and password.

The following list of options appears.

DirXML commands

```
1: Start driver
2: Stop driver
3: Driver operations...
4: Driver set operations...
5: Log events operations...
6: Get DirXML version
7: Job operations
99: Quit
Enter choice:
```

**3** Enter 3 for driver operations.

A numbered list of drivers appears.

**4** Enter the number for the driver you want to remove Named Passwords from.

The following list of options appears.

Select a driver operation for:

*driver\_name*

```
1: Start driver
2: Stop driver
3: Get driver state
4: Get driver start option
5: Set driver start option
6: Resync driver
7: Migrate from application into DirXML
8: Submit XDS command document to driver
9: Submit XDS event document to driver
10: Queue event for driver
11: Check object password
12: Initialize new driver object
13: Passwords operations
14: Cache operations
99: Exit
Enter choice:
```

**5** Enter 13 for password operations.

The following list of options appears.

Select a password operation

```
1: Set shim password
2: Reset shim password
3: Set Remote Loader password
4: Clear Remote Loader password
5: Set named password
6: Clear named password(s)
7: List named passwords
8: Get passwords state
99: Exit
Enter choice:
```

**6** (Optional) Enter 7 to see the list of existing Named Passwords.

The list of existing Named Passwords is displayed.

This step can help you make sure you are removing the correct password.

**7** Enter 6 to remove one or more Named Passwords.

**8** Enter No to remove a single Named Password at the following prompt:

```
Do you want to clear all named passwords? (yes/no):
```

**9** Enter the name of the Named Password you want to remove at the following prompt:

```
Enter password name:
```

After you enter the name of the Named Password you want to remove, you are returned to the password operations menu:

```
Select a password operation
```

```
1: Set shim password
2: Reset shim password
3: Set Remote Loader password
4: Clear Remote Loader password
5: Set named password
6: Clear named password(s)
7: List named passwords
8: Get passwords state
99: Exit
```

```
Enter choice:
```

**10** (Optional) Enter 7 to see the list of existing Named Passwords.

This step lets you verify that you have removed the correct password.

**11** After completing this procedure, you can use the 99 option twice to exit the menu and quit the DirXML Command Line utility.

## 7.8 Adding a Driver Heartbeat

The driver heartbeat is a feature of the Identity Manager drivers that ship with Identity Manager 2 and later. Its use is optional. The driver heartbeat is configured by using a driver parameter with a time interval specified. If a heartbeat parameter exists and has an interval value other than 0, the driver sends a heartbeat document to the Metadirectory engine if there is no communication on the Publisher channel for the specified interval of time.

The intent of the driver heartbeat is to give you a trigger to allow you to initiate an action at regular intervals, if the driver does not communicate on the Publisher channel as often as you want the action to occur. To take advantage of the heartbeat, you must customize your driver configuration or other tools. The Metadirectory engine accepts the heartbeat document but does not take any action because of it.

For most drivers, a driver parameter for heartbeat is not used in the sample configurations, but you can add it.

A custom driver that is not provided with Identity Manager can also provide a heartbeat document, if the driver developer has written the driver to support it.

To configure the heartbeat:

- 1** In iManager, click *Identity Manager > Identity Manager Overview*.
- 2** Browse to and select your driver set object, then click *Search*.

**3** In the Identity Manager Overview, click the upper right corner of the driver icon, then click *Edit properties*.

**4** On the Identity Manager tab, click *Driver Configuration*, scroll to *Driver Parameters*, then look for Heart Beat or a similar display name.

If a driver parameter already exists for heartbeat, you can change the interval and save the changes, and configuration is then complete.

The value of the interval cannot be less than 1. A value of 0 means the feature is turned off.

The unit of time is usually minutes; however, some drivers might choose to implement it differently, such as using seconds.

**5** If a driver parameter does not exist for heartbeat, click *Edit XML*.

**6** Add a driver parameter entry like the following example, as a child of <publisher-options>. (For an AD driver, make it a child of <driver-options>.)

```
<pub-heartbeat-interval display-name="Heart Beat">10</pub-heartbeat-interval>
```

---

**TIP:** If the driver does not produce a heartbeat document after being restarted, check the placement of the driver parameter in the XML.

---

**7** Save the changes, and make sure the driver is stopped and restarted.

After you have added the driver parameter, you can edit the time interval by using the graphical view. Another option is to create a reference to a global configuration value (GCV) for the time interval. Like other global configuration values, the driver heartbeat can be set at the driver set level instead of on each individual driver object. If a driver does not have a particular global configuration value, and the driver set object does have it, the driver inherits the value from the driver set object.





# Synchronizing Objects

# 8

This section explains driver and object synchronization in DirXML<sup>®</sup> 1.1a, Identity Manager 2.0, and Identity Manager 3.x. Driver synchronization was not available for DirXML 1.0 and DirXML 1.1.

After the driver is created, instead of waiting for objects to be modified or created, the data between the two connected systems can be sent through the synchronization process.

- ♦ [Section 8.1, “What Is Synchronization?” on page 89](#)
- ♦ [Section 8.2, “When Is Synchronization Done?” on page 89](#)
- ♦ [Section 8.3, “How Does the Metadirectory Engine Decide Which Object to Synchronize?” on page 90](#)
- ♦ [Section 8.4, “How Does Synchronization Work?” on page 91](#)

## 8.1 What Is Synchronization?

The actions commonly referred to as “synchronization” in Identity Manager refer to several different but related actions:

- ♦ Synchronization (or merging) of attribute values of an object in the Identity Vault with the corresponding attribute values of an associated object in a connected system.
- ♦ Migration of all Identity Vault objects and classes that are included in the filter on the Subscriber channel.
- ♦ Generation of the list of objects to submit to the driver’s Subscriber channel for synchronization or migration in response to a user request (a manual synchronization).
- ♦ Generation of the list of objects to submit to the driver’s Subscriber channel for synchronization or migration in response to enabling a formerly disabled driver, or in response to a cache error.

## 8.2 When Is Synchronization Done?

The Metadirectory engine performs object synchronization or merging in the following circumstances:

- ♦ A `<sync>` event element is submitted on the Subscriber or Publisher channel.
- ♦ A `<sync>` event element is submitted on the Subscriber channel in the following circumstances:
  - ♦ The state of the object’s association value is set to “manual” or “migrate.” (This causes an eDirectory™ event, which in turn causes the Identity Manager caching system to queue an object synchronization command in the affected driver’s cache.)
  - ♦ An object synchronization command is read from the driver’s cache.
- ♦ A `<sync>` event element is submitted on the Publisher channel in the following circumstances:
  - ♦ A driver submits a `<sync>` event element. No known driver currently does this.

- ◆ The Metadirectory engine submits a <sync> event element for each object found as the result of a migrate-into-NDS query. These <sync> events are submitted using the Subscriber thread, but are processed using the Publisher channel filter and policies.
- ◆ An <add> event (real or synthetic) is submitted on a channel and the channel Matching policy finds a matching object in the target system.
- ◆ An <add> event with an association is submitted on the Subscriber channel. This normally occurs only in exceptional cases, such as the bulk load of objects into eDirectory with DirXML-Associations attribute values.
- ◆ An <add> event is submitted on the Publisher channel and an object is found in eDirectory that already has the association value reported with the <add> event.

The Metadirectory engine generates synchronization requests for zero or more objects in the following cases:

- ◆ The user issues a manual driver synchronization request. This corresponds to the *Resync* button in the Driver Set property page in ConsoleOne®, or to the *Synchronize* button on the iManager Identity Manager Driver Overview page.
- ◆ The Metadirectory engine encounters an error with the driver's cache and cannot recover from the cache error. The driver's cache is deleted and the engine generates object synchronization commands as detailed in [Section 8.3, "How Does the Metadirectory Engine Decide Which Object to Synchronize?," on page 90.](#)

## 8.3 How Does the Metadirectory Engine Decide Which Object to Synchronize?

The Metadirectory engine processes both manually initiated and automatically initiated synchronization requests in the same manner. The only difference in the processing of manually initiated versus automatically initiated driver synchronization requests is the starting filter time used to filter objects being considered for synchronization.

The starting filter time is used to filter objects that have modification or creation times that are older than the starting time specified in the synchronization request.

For automatically initiated driver synchronization, the starting filter time is obtained from the time stamps of cached eDirectory events. In particular, the starting filter time is the earliest time for the cached events that haven't yet been successfully processed by the driver's Subscriber channel.

For manually initiated driver synchronization, the default starting filter time is the earliest time in the eDirectory database. In Identity Manager 2 and Identity Manager 3, an explicit starting filter time can also be set. In DirXML 1.1a there is no facility to set the starting filter time value for synchronization when manually initiating driver synchronization.

The Metadirectory engine creates a list of objects to be synchronized on the Subscriber channel in the following manner:

1. It finds all objects that:
  - ◆ Have an entry modification time stamp greater than or equal to the starting filter time and
  - ◆ Exist in the filter on the Subscriber channel.

2. It finds all objects that have an entry creation time stamp greater than or equal to the starting filter time.
3. It adds a `synchronize object` command to the driver cache for each unique object found that has an entry modification time stamp greater than or equal to the starting filter time and all objects and classes that are in the Subscriber filter channel in the driver being synchronized.

## 8.4 How Does Synchronization Work?

After the Metadirectory engine determines that an object is to be synchronized, the following processes occur:

1. Each system (the Identity Vault and the connected system) is queried for all attribute values in the appropriate filters.
  - ♦ eDirectory is queried for all values in the Subscriber filter, and for values that are marked for synchronization in Identity Manager 2.x and Identity Manager 3.x.
  - ♦ The connected system is queried for all values in the Publisher filter, and for values that are marked for synchronization in Identity Manager 2.x and Identity Manager 3.x.
2. The returned attribute values are compared and modification lists are prepared for the Identity Vault and the connected system according to [Table 8-1 on page 92](#), [Table 8-2 on page 93](#), and [Table 8-3 on page 94](#).

In the tables the following pseudo-equations are used:

- ♦ “Left = Right” indicates that the left side receives all values from the right side.
- ♦ “Left = Right[1]” indicates that the left side receives one value from the right side. If there is more than one value, it is indeterminate.
- ♦ “Left += Right” indicates that the left side adds the right side values to the left side’s existing values.
- ♦ “Left = Left + Right” indicates that the left sides receives the union of the values of the left and right sides.

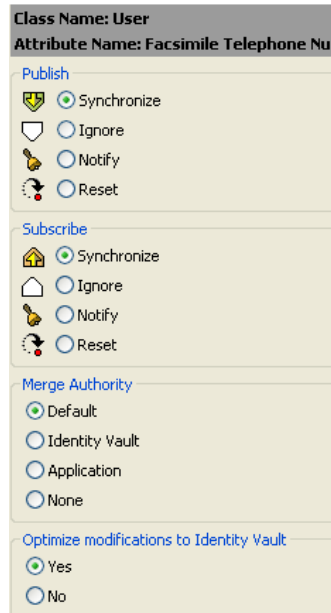
There are three different combinations of selected items in the filter, and each one creates a different output.

- ♦ [Section 8.4.1, “Scenario One,” on page 91](#)
- ♦ [Section 8.4.2, “Scenario Two,” on page 93](#)
- ♦ [Section 8.4.3, “Scenario Three,” on page 94](#)

### 8.4.1 Scenario One

The attribute is set to *Synchronize* on the Publisher and Subscriber channels, and the merge authority is set to *Default*.

**Figure 8-1** Scenario One



The following table contains the values that the Metadirectory engine synchronizes when the attribute is sent through a filter that is set to the configuration for Scenario One. The table shows different outputs depending upon whether the attribute comes from the Identity Vault or the Application, if the attribute is single-valued or multi-valued, and if the attribute is empty or non-empty.

**Table 8-1** Output of Scenario One

	Identity Vault single-valued empty	Identity Vault single-valued non-empty	Identity Vault multi-valued empty	Identity Vault multi-valued non-empty
<b>Application single-valued empty</b>	No change	App = Identity Vault	No change	App = Identity Vault[1]
<b>Application single-valued non-empty</b>	Identity Vault = App	App = Identity Vault	Identity Vault = App	Identity Vault + = App
<b>Application multi-valued empty</b>	No change	App = Identity Vault	No change	App = Identity Vault
<b>Application multi-valued non-empty</b>	Identity Vault = App[1]	App + = Identity Vault	Identity Vault = App	App = App + Identity Vault  Identity Vault = App + Identity Vault

## 8.4.2 Scenario Two

The attribute is set to *Synchronize* only on the Subscriber channel, or it is set to *Synchronize* on both the Subscriber and Publisher channels. The merge authority is set to *Identity Vault*.

**Figure 8-2** Scenario Two

**Class Name: User**

**Attribute Name: Description**

**Publish**

Synchronize

Ignore

Notify

Reset

**Subscribe**

Synchronize

Ignore

Notify

Reset

**Merge Authority**

Default

Identity Vault

Application

None

**Optimize modifications to Identity Vault**

Yes

No

The following table contains the values that the Metadirectory engine synchronizes when the attribute is sent through a filter that is set to the configuration for Scenario Two. The table shows different outputs depending upon whether the attribute comes from the Identity Vault or the Application, if the attribute is single-valued or multi-valued, and if the attribute is empty or non-empty.

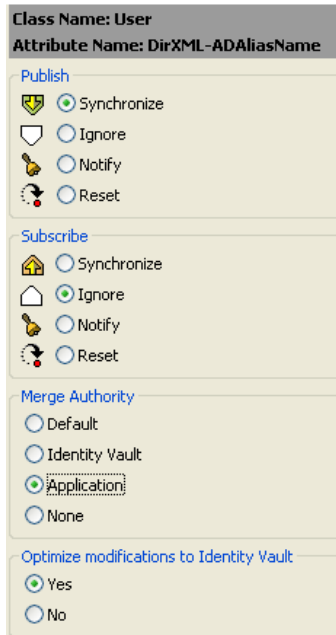
**Table 8-2** Output of Scenario Two

	Identity Vault single-valued empty	Identity Vault single-valued non-empty	Identity Vault multi-valued empty	Identity Vault multi-valued non-empty
<b>Application single-valued empty</b>	No change	App = Identity Vault	No change	App = Identity Vault[1]
<b>Application single-valued empty</b>	App = empty	App = Identity Vault	Identity Vault = App	App = Identity Vault[1]
<b>Application multi-valued empty</b>	No change	App = Identity Vault	No change	App = Identity Vault
<b>Application multi-valued non-empty</b>	App = empty	App = Identity Vault	App = empty	App = Identity Vault

### 8.4.3 Scenario Three

The attribute is set to *Synchronize* on the Publisher channel or the merge authority is set to *Application*.

**Figure 8-3** Scenario Three



The following table contains the values that the Metadirectory engine synchronizes when the attribute is sent through a filter that is set to the configuration for Scenario Three. The table shows different outputs depending upon whether the attribute comes from the Identity Vault or the Application, if the attribute is single-valued or multi-valued, and if the attribute is empty or non-empty.

**Table 8-3** Output of Scenario Three

	Identity Vault single-valued empty	Identity Vault single-valued non-empty	Identity Vault multi-valued empty	Identity Vault multi-valued non-empty
<b>Application single-valued empty</b>	No change	Identity Vault = empty	No change	Identity Vault = empty
<b>Application single-valued non-empty</b>	Identity Vault = App	Identity Vault = App	Identity Vault = App	Identity Vault = App
<b>Application multi-valued empty</b>	No change	Identity Vault = empty	No change	Identity Vault = empty
<b>Application multi-valued non- empty</b>	Identity Vault = App[1]	Identity Vault = App[1]	Identity Vault = App	Identity Vault = App

# Troubleshooting the Driver

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

- ◆ [Section 9.1, “The Driver is Not Processing Available Transactions or is Processing Them Out of Order,” on page 95](#)
- ◆ [Section 9.2, “Error Trying to Obtain Data Record,” on page 95](#)
- ◆ [Section 9.3, “Error: joltServiceException: Invalid Session,” on page 96](#)
- ◆ [Section 9.4, “The Driver Does Not Start,” on page 96](#)
- ◆ [Section 9.5, “Attributes Are Not Refreshed on the Data Schema Object,” on page 96](#)
- ◆ [Section 9.6, “Data Does Not Show Up in Identity Vault on the Publisher Channel,” on page 96](#)
- ◆ [Section 9.7, “Error: Check Application Server IP Address and Jolt Port Number,” on page 96](#)
- ◆ [Section 9.8, “Data Does Not Update in PeopleSoft on the Subscriber Channel,” on page 97](#)
- ◆ [Section 9.9, “No Transactions Are Coming across the Publisher Channel,” on page 97](#)
- ◆ [Section 9.10, “Transactions Are Not Placed in the PeopleSoft Queue,” on page 97](#)
- ◆ [Section 9.11, “Transactions Are Left in the “Process” State and Not Processed,” on page 97](#)
- ◆ [Section 9.12, “Errors on the Publisher Channel When Processing a Transaction,” on page 97](#)
- ◆ [Section 9.13, “Component Interface Relationships Not Functioning,” on page 98](#)
- ◆ [Section 9.14, “SQL Error During Save of “Sample Person” Records,” on page 98](#)
- ◆ [Section 9.15, “Troubleshooting Driver Processes,” on page 99](#)

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

- ◆ Set the driver trace level to 5 and verify that the DIRXML\_DTTM and DIRXML\_CURRDTM values of the Transaction records being processed are in proper lexicographic format.
- ◆ If the records are not in the correct format, refer to [Chapter 2, “Configuring Your PeopleSoft Environment,” on page 17](#).
- ◆ If the records are in the correct format, verify that Transaction date and time field values are correct and correspond to the system date and time.

## 9.2 Error Trying to Obtain Data Record

The following are typical reasons for this error:

- ◆ The Data record identified in a Transaction record was deleted from the PeopleSoft server before the Transaction was processed.
- ◆ The Data record identified in a query or Subscriber channel operation has been deleted from the PeopleSoft server.

- ♦ Through a database error or bad configuration, multiple Data records with the same primary key value exist in the PeopleSoft database.  
Verify the reason for the problem by using either an SQL tool, the PSA DirXML Schema 01 sample application, or the PeopleSoft Application Designer's Test Component Interface tool (see [Chapter 2, "Configuring Your PeopleSoft Environment," on page 17.](#)) Correct any errors that might exist.

## 9.3 Error: joltServiceException: Invalid Session

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

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

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

## 9.4 The Driver Does Not Start

- ♦ Verify that `pssoftshim.jar`, `dirxmlcomps.jar`, `psjoa.jar`, and the required CI API jar files are present in the DirXML<sup>®</sup> lib subdirectory.
- ♦ Verify that the connection parameters are set correctly.
- ♦ Ensure that the configured CI names are valid.

## 9.5 Attributes Are Not Refreshed on the Data Schema Object

Verify that the Component Interfaces are working correctly by using PeopleSoft Application Designer tool. Refer to [Chapter 2, "Configuring Your PeopleSoft Environment," on page 17.](#)

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

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

## 9.7 Error: Check Application Server IP Address and Jolt Port Number

Run the CITester utility to verify that proper connection and authentication parameters are set. Refer to [Chapter 2, "Configuring Your PeopleSoft Environment," on page 17.](#)



## 9.8 Data Does Not Update in PeopleSoft on the Subscriber Channel

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

## 9.9 No Transactions Are Coming across the Publisher Channel

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

## 9.10 Transactions Are Not Placed in the PeopleSoft Queue

Verify that PeopleCode is working properly.

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

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

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

---

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

---

## 9.12 Errors on the Publisher Channel When Processing a Transaction

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

- ♦ Operation vetoed by the Create policy

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

- ◆ generateKeyPair: -216 DSERR\_PASSWORD\_TOO\_SHORT  
The attribute used for the initial password does not comply 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.

## 9.13 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.

## 9.14 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.

## 9.15 Troubleshooting Driver Processes

Viewing driver processes is necessary to analyze unexpected behavior. To view the driver processing events, use DSTRACE. You should only use it during testing and troubleshooting the driver. Running DSTRACE while the drivers are in production increases the utilization on the Identity Manager server and can cause events to process very slowly.

### 9.15.1 Viewing Driver Processes

In order to see the driver processes in DSTRACE, values are added to the driver set and the driver objects. You can do this in Designer and iManager.

- ♦ [“Adding Trace Levels in Designer” on page 99](#)
- ♦ [“Adding Trace Levels in iManager” on page 101](#)
- ♦ [“Capturing Driver Processes to a File” on page 102](#)

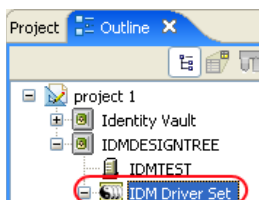
#### Adding Trace Levels in Designer

You can add trace levels to the driver set object or to each driver object.

- ♦ [“Driver Set” on page 99](#)
- ♦ [“Driver” on page 100](#)

#### Driver Set

- 1 In an open project in Designer, select the driver set object in the *Outline* view.



- 2 Right-click and select *Properties*, then click *5. Trace*.
- 3 Set the parameters for tracing, then click *OK*.

Parameter	Description
Driver trace level	<p>As the driver object trace level increases, the amount of information displayed in DSTRACE increases.</p> <p>Trace level 1 shows errors, but not the cause of the errors. If you want to see password synchronization information, set the trace level to 5.</p>
XSL trace level	DSTRACE displays XSL events. Set this trace level only when troubleshooting XSL style sheets. If you do not want to see XSL information, set the level to zero.
Java debug port	Allows developers to attach a Java* debugger.
Java trace file	<p>When a value is set in this field, all Java information for the driver set object is written to a file. The value for this field is the path for that file.</p> <p>As long as the file is specified, Java information is written to this file. If you do not need to debug Java, leave this field blank.</p>
Trace file size limit	<p>Allows you to set a limit for the Java trace file. If you set the file size to <i>Unlimited</i>, the file grows in size until there is no disk space left.</p> <hr/> <p><b>NOTE:</b> The trace file is created in multiple files. Identity Manager automatically divides the maximum file size by ten and creates ten separate files. The combined size of these files equals the maximum trace file size.</p>

If you set the trace level on the driver set object, all drivers appear in the DSTRACE logs.

## Driver

- 1 In an open project in Designer, select the driver object in the *Outline* view.
- 2 Right-click and select *Properties*, then click *8. Trace*.
- 3 Set the parameters for tracing, then click *OK*.

Parameter	Description
Trace level	<p>As the driver object trace level increases, the amount of information displayed in DSTRACE increases.</p> <p>Trace level 1 shows errors, but not the cause of the errors. If you want to see password synchronization information, set the trace level to 5.</p> <p>if you select <i>Use setting from Driver Set</i>, the value is taken from the driver set object.</p>
Trace file	<p>Specify a filename and location for where the Identity Manager information is written for the selected driver.</p> <p>if you select <i>Use setting from Driver Set</i>, the value is taken from the driver set object.</p>

Parameter	Description
Trace file size limit	<p>Allows you to set a limit for the Java trace file. If you set the file size to <i>Unlimited</i>, the file grows in size until there is no disk space left.</p> <p>If you select <i>Use setting from Driver Set</i>, the value is taken from the driver set object.</p> <hr/> <p><b>NOTE:</b> The trace file is created in multiple files. Identity Manager automatically divides the maximum file size by ten and creates ten separate files. The combined size of these files equals the maximum trace file size.</p> <hr/>
Trace name	The driver trace messages are prepended with the value entered instead of the driver name. Use this option if the driver name is very long.

If you set the parameters only on the driver object, only information for that driver appears in the DSTRACE log.

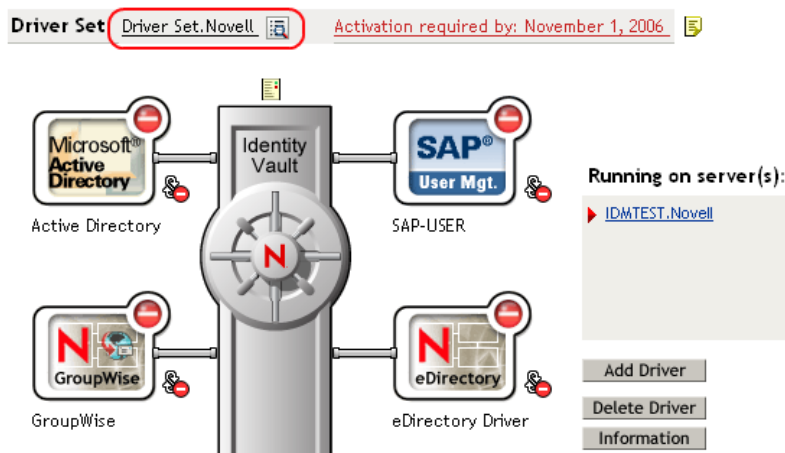
## Adding Trace Levels in iManager

You can add trace levels to the driver set object or to each driver object.

- ♦ “Driver Set” on page 101
- ♦ “Driver” on page 102

### Driver Set

- 1 In iManager, select *Identity Manager > Identity Manager Overview*.
- 2 Browse to the driver set object, then click *Search*.
- 3 Click the driver set name.



- 4 Select the *Misc* tab for the driver set object.
  - 5 Set the parameters for tracing, then click *OK*.
- See “Misc” on page 137 for the parameters.

## Driver

- 1 In iManager, select *Identity Manager > Identity Manager Overview*.
- 2 Browse to the driver set object where the driver object resides, then click *Search*.
- 3 Click the upper right corner of the driver object, then click *Edit properties*.
- 4 Select the *Misc* tab for the driver object.
- 5 Set the parameters for tracing, then click *OK*.  
See “Misc” on page 137 for the parameters.

---

**NOTE:** The option *Use setting from Driver Set* does not exist in iManager.

---

## Capturing Driver Processes to a File

You can save driver processes to a file by using the parameter on the driver object or by using DSTRACE. The parameter on the driver object is the *Trace file* parameter, under the *MISC* tab.

The driver processes that are captured through DSTRACE are the processes that occur on the Identity Manager engine. If you use the Remote Loader, you need to capture a trace on the Remote Loader at the same time as you are capturing the trace on the Identity Manager engine.

The following methods help you capture and save Identity Manager processes through DSTRACE on different platforms.

- ♦ “NetWare” on page 102
- ♦ “Windows” on page 103
- ♦ “UNIX” on page 103
- ♦ “iMonitor” on page 103
- ♦ “Remote Loader” on page 104

## NetWare

Use `dstrace.nlm` to display trace messages on the system console or trace messages to a file (`sys:\system\dstrace.log`). Use `dstrace.nlm` to display the trace messages to a screen labeled DSTrace Console.

- 1 Enter `dstrace.nlm` at the server console to load `dstrace.nlm` into memory.
- 2 Enter `dstrace screen on` at the server console to allow trace messages to appear on the DSTrace Console screen.
- 3 Enter `dstrace file on` at the server console to capture trace messages sent to the DSTrace Console to the `dstrace.log` file.
- 4 (Optional) Enter `dstrace -all` at the server console to make it easier to read the trace log.
- 5 Enter `dstrace +dxml dstrace +dvrs` at the server console to display Identity Manager events.
- 6 Enter `dstrace +tags dstrace +time` at the server console to display message tags and time stamps.
- 7 Toggle to the DSTrace Console screen and watch for the event to pass.
- 8 Toggle back to the server console.

- 9 Enter `dstrace file off` at the server console.

This stops capturing trace messages to the log file. It also stops logging information into the file.

- 10 Open the `dstrace.log` in a text editor and search for the event or the object you modified.

## Windows

- 1 Open the *Control Panel* > *NDS Services* > `dstrace.dlm`, then click *Start* to display the NDS Server Trace utility window.
- 2 Click *Edit* > *Options*, then click *Clear All* to clear all of the default flags.
- 3 Select *DirXML* and *DirXML Drivers*.
- 4 Click OK.
- 5 Click *File* > *New*.
- 6 Specify the filename and location where you want the DSTRACE information saved, then click *Open*.
- 7 Wait for the event to occur.
- 8 Click *File* > *Close*.  
This stops the information from being written to the log file.
- 9 Open the file in a text editor and search for the event or the object you modified.

## UNIX

- 1 Enter `ndstrace` to start the `ndstrace` utility.
- 2 Enter `set ndstrace=nodebug` to turn off all trace flags currently set.
- 3 Enter `set ndstrace on` to display trace messages to the console.
- 4 Enter `set ndstrace file on` to capture trace messages to the `ndstrace.log` file in the directory where eDirectory is installed. By default it is `/var/nds`.
- 5 Enter `set ndstrace+=dxml` to display the Identity Manager events.
- 6 Enter `set ndstrace+=dvrs` to display the Identity Manager driver events.
- 7 Wait for the event to occur.
- 8 Enter `set ndstrace file off` to stop logging information to the file.
- 9 Enter `exit` to quite the `ndstrace` utility.
- 10 Open the file in a text editor. Search for the event or the object that was modified.

## iMonitor

iMonitor allows you to get DSTRACE information from a Web browser. It does not matter where Identity Manager is running. The following files run iMonitor:

- ♦ `ndsmon.nlm` runs on NetWare®.
  - ♦ `ndsmon.dlm` runs on Windows.
  - ♦ `ndsmonitor` runs on UNIX\*.
- 1 Access iMonitor from `http://server_ip:8008/nds`.

Port 8008 is the default.

- 2 Specify a username and password with administrative rights, then click *Login*.
- 3 Select *Trace Configuration* on the left side.
- 4 Click *Clear All*.
- 5 Select *DirXML* and *DirXML Drivers*.
- 6 Click *Trace On*.
- 7 Select *Trace History* on the left side.
- 8 Click the document with the *Modification Time* of *Current* to see a live trace.
- 9 Change the *Refresh Interval* if you want to see information more often.
- 10 Select *Trace Configuration* on the left side, then click *Trace Off* to turn the tracing off.
- 11 Select *Trace History* to view the trace history.

The files are distinguished by their time stamp.

If you need a copy of the HTML file, the default location is:

- ◆ NetWare: `sys:\system\nds\simon\dstrace*.htm`
- ◆ Windows: `Drive_letter:\novell\nds\nds\simon\dstrace\*.htm`
- ◆ UNIX: `/var/nds/dstrace/*.htm`

## Remote Loader

You can capture the events that occur on the machine running the Remote Loader service.

- 1 Launch the Remote Loader Console by clicking the icon.
- 2 Select the driver instance, then click *Edit*.
- 3 Set the *Trace Level* to 3 or above.
- 4 Specify a location and file for the trace file.
- 5 Specify the amount of disk space that the file is allowed.
- 6 Click *OK*, twice to save the changes.

You can also enable tracing from the command line by using the following switches. For more information, see “[Configuring the Remote Loader](#)” in the *Novell Identity Manager 3.5.1 Administration Guide*.

**Table 9-1** *Command Line Tracing Switches*

Option	Short Name	Parameter	Description
-trace	-t	integer	Specifies the trace level. This is only used when hosting an application shim. Trace levels correspond to those used on the Identity Manager server.  Example: <code>-trace 3</code> or <code>-t3</code>



Option	Short Name	Parameter	Description
-tracefile	-tf	filename	<p>Specify a file to write trace messages to. Trace messages are written to the file if the trace level is greater than zero. Trace messages are written to the file even if the trace window is not open.</p> <p>Example: <code>-tracefile c:\temp\trace.txt</code> or <code>-tf c:\temp\trace.txt</code></p>
-tracefilemax	-tfm	size	<p>Specifies the approximate maximum size that trace file data can occupy on disk. If you specify this option, there is a trace file with the name specified using the tracefile option and up to 9 additional “roll-over” files. The roll-over files are named using the base of the main trace filename plus “_n”, where n is 1 through 9.</p> <p>The size parameter is the number of bytes. Specify the size by using the suffixes K, M, or G for kilobytes, megabytes, or gigabytes.</p> <p>If the trace file data is larger than the specified maximum when the Remote Loader is started, the trace file data remains larger than the specified maximum until roll-over is completed through all 10 files.</p> <p>Example: <code>-tracefilemax 1000M</code> or <code>-tfm 1000M</code></p>



# Backing Up the Driver

You can use Designer or iManager to create an XML file of the driver. The file contains all of the information entered into the driver during configuration. If the driver becomes corrupted, the exported file can be imported to restore the configuration information.

---

**IMPORTANT:** If the driver has been deleted, all of the associations on the objects are purged. When the XML file is imported again, new associations are created through the migration process.

---

Not all server-specific information stored on the driver is contained in the XML file. Make sure this information is documented through the Document Generation process in Designer. See “[Generating a Document](#)” in the *Designer 2.1 for Identity Manager 3.5.1*.

- ♦ [Section 10.1, “Exporting the Driver in Designer,” on page 107](#)
- ♦ [Section 10.2, “Exporting the Driver in iManager,” on page 107](#)

## 10.1 Exporting the Driver in Designer

- 1 Open a project in Designer, then right-click the driver object.
- 2 Select *Export to Configuration File*.
- 3 Specify a unique name for the configuration file, browse to location where it should be saved, then click *Save*.
- 4 Click *OK* in the Export Configuration Results window.

## 10.2 Exporting the Driver in iManager

- 1 In iManager, select *Identity Manager > Identity Manager Overview*.
- 2 Browse to and select the driver set object, then click *Search*.
- 3 Click the driver icon.
- 4 Select *Export* in the Identity Manager Driver Overview window.
- 5 Browse to and select the driver object you want to export, then click *Next*.
- 6 Select *Export all policies, linked to the configuration or not* or select *Only export policies that are linked to the configuration*, depending upon the information you want to have stored in the XML file.
- 7 Click *Next*.
- 8 Click *Save As*, then click *Save*.
- 9 Browse and select a location to save the XML file, then click *Save*.
- 10 Click *Finish*.



# Security: Best Practices

# 11

In order to secure the driver and the information it is synchronizing, see “[Security: Best Practices](#)” in the *Novell Identity Manager 3.5.1 Administration Guide*.



# DirXML Command Line Utility

# A

The DirXML<sup>®</sup> Command Line utility allows you to use a command line interface to manage the driver. You can create scripts to manage the driver with the commands.

The utility and scripts are installed on all platforms during the Identity Manager installation. The utility is installed to the following locations:

eDirectory 8.7.x

- ♦ Windows: \Novell\Nds\dxcmd.bat
- ♦ NetWare<sup>®</sup>: sys:\system\dxcmd.ncf
- ♦ UNIX: /usr/bin/dxcmd

eDirectory 8.8.x

- ♦ Windows: \Novell\Nds\dxcmd.bat
- ♦ NetWare: sys:\system\dxcmd.ncf
- ♦ UNIX: /opt/novell/eDirectory/bin/dxcmd

There are two different methods for using the DirXML Command Line utility:

- ♦ [Section A.1, “Interactive Mode,” on page 111](#)
- ♦ [Section A.2, “Command Line Mode,” on page 120](#)

## A.1 Interactive Mode

The interactive mode provides a text interface to control and use the DirXML Command Line utility.

- 1 At the console, enter `dxcmd`.
- 2 Enter the name of a user with sufficient rights to the Identity Manager objects, such as `admin.novell`.
- 3 Enter the user’s password.

```
DirXML commands
1: Start driver
2: Stop driver
3: Driver operations...
4: Driver set operations...
5: Log events operations...
6: Get DirXML version
7: Job operations...
99: Quit
Enter choice:
```

- 4 Enter the number of the command you want to perform.  
[Table A-1 on page 112](#) contains the list of options and what functionality is available.
- 5 Enter 99 to quit the utility.

---

**NOTE:** If you are running eDirectory™ 8.8 on UNIX or Linux\*, you must specify the -host and -port parameters. For example, `dxcmd -host 10.0.0.1 -port 524`. If the parameters are not specified, a jclient error occurs.

`novell.jclient.JCException: connect (to address) 111 UNKNOWN ERROR`

By default, eDirectory 8.8 is not listening to localhost. The DirXML Command Line utility needs to resolve the server IP address or hostname and the port to be able to authenticate.

---

**Table A-1** *Interactive Mode Options*

Option	Description
1: <i>Start Driver</i>	Starts the driver. If there is more than one driver, each driver is listed with a number. Enter the number of the driver to start the driver.
2: <i>Stop Driver</i>	Stops the driver. If there is more than one driver, each driver is listed with a number. Enter the number of the driver to stop the driver.
3: <i>Driver operations</i>	Lists the operations available for the driver. If there is more than one driver, each driver is listed with a number. Enter the number of the driver to see the operations available. See <a href="#">Table A-2 on page 113</a> for a list of operations.
4: <i>Driver set operations</i>	Lists the operations available for the driver set. <ul style="list-style-type: none"><li>◆ 1: Associate driver set with server</li><li>◆ 2: Disassociate driver set from server</li><li>◆ 99: Exit</li></ul>
5: <i>Log events operations</i>	Lists the operations available for logging events through Novell® Audit. See <a href="#">Table A-5 on page 118</a> for a description of these options.
6: <i>Get DirXML version</i>	Lists the version of the Identity Manager installed.
7: <i>Job operations</i>	Manages jobs created for Identity Manager.1
99: <i>Quit</i>	Exits the DirXML Command Line utility

---



**Figure A-1** *Driver Options*

```
Select a driver operation for:
Active Directory.Driver Set.Novell.IDMDESIGNTREE.

1: Start driver
2: Stop driver
3: Get driver state
4: Get driver start option
5: Set driver start option
6: Resync driver
7: Migrate from application into DirXML
8: Submit XDS command document to driver
9: Submit XDS event document to driver
10: Queue event for driver
11: Check object password
12: Initialize new driver object
13: Passwords operations
14: Cache operations
99: Exit

Enter choice:
```

**Table A-2** *Driver Options*

Options	Description
1: <i>Start driver</i>	Starts the driver.
2: <i>Stop driver</i>	Stops the driver.
3: <i>Get driver state</i>	Lists the state of the driver. <ul style="list-style-type: none"><li>◆ 0 - Driver is stopped</li><li>◆ 1 - Driver is starting</li><li>◆ 2 - Driver is running</li><li>◆ 3 - Driver is stopping</li></ul>
4: <i>Get driver start option</i>	Lists the current driver start option. <ul style="list-style-type: none"><li>◆ 1 - Disabled</li><li>◆ 2 - Manual</li><li>◆ 3 - Auto</li></ul>
5: <i>Set driver start option</i>	Changes the start option of the driver. <ul style="list-style-type: none"><li>◆ 1 - Disabled</li><li>◆ 2 - Manual</li><li>◆ 3 - Auto</li><li>◆ 99 - Exit</li></ul>

Options	Description
6: <i>Resync driver</i>	<p>Forces a resynchronization of the driver. It prompts for a time delay: <i>Do you want to specify a minimum time for resync? (yes/no).</i></p> <p>If you enter Yes, specify the date and time you want the resynchronization to occur: <i>Enter a date/time (format 9/27/05 3:27 PM).</i></p> <p>If you enter No, the resynchronization occurs immediately.</p>
7: <i>Migrate from application into DirXML</i>	<p>Processes an XML document that contains a query command: <i>Enter filename of XDS query document:</i></p> <p>Create the XML document that contains a query command by using the <a href="http://developer.novell.com/ndk/doc/dirxml/dirxmlbk/ref/ndsstd/query.html">Novell nds.dtd (http://developer.novell.com/ndk/doc/dirxml/dirxmlbk/ref/ndsstd/query.html)</a>.</p> <p>Examples:</p> <p>NetWare: <code>sys:\files\query.xml</code></p> <p>Windows: <code>c:\files\query.xml</code></p> <p>Linux: <code>/files/query.xml</code></p>
8: <i>Submit XDS command document to driver</i>	<p>Processes an XDS command document:</p> <p><i>Enter filename of XDS command document:</i></p> <p>Examples:</p> <p>NetWare: <code>sys:\files\user.xml</code></p> <p>Windows: <code>c:\files\user.xml</code></p> <p>Linux: <code>/files/user.xml</code></p> <p><i>Enter name of file for response:</i></p> <p>Examples:</p> <p>NetWare: <code>sys:\files\user.log</code></p> <p>Windows: <code>c:\files\user.log</code></p> <p>Linux: <code>/files/user.log</code></p>
9: <i>Submit XDS event document to driver</i>	<p>Processes an XDS event document:</p> <p><i>Enter filename of XDS event document:</i></p> <p>Examples:</p> <p>NetWare: <code>sys:\files\add.xml</code></p> <p>Windows: <code>c:\files\add.xml</code></p> <p>Linux: <code>/files/add.xml</code></p>

Options	Description
10: <i>Queue event for driver</i>	Adds and event to the driver queue  <i>Enter filename of XDS event document:</i>  Examples:  NetWare: <code>sys:\files\add.xml</code>  Windows: <code>c:\files\add.xml</code>  Linux: <code>/files/add.xml</code>
11: <i>Check object password</i>	Validates that an object's password in the connected system is associated with a driver. It matches the object's eDirectory password (Distribution Password, used with Universal Password).  <i>Enter user name:</i>
12: <i>Initialize new driver object</i>	Performs an internal initialization of data on a new Driver object. This is only for testing purposes.
13: <i>Password operations</i>	There are nine Password options. See <a href="#">Table A-3 on page 115</a> for a description of these options.
14: <i>Cache operations</i>	There are five Cache operations. See <a href="#">Table A-4 on page 117</a> for a descriptions of these options.
99: <i>Exit</i>	Exits the driver options.

**Figure A-2** *Password Operations*

```
Select a password operation
1: Set shim password
2: Clear shim password
3: Set Remote Loader password
4: Clear Remote Loader password
5: Set named password
6: Clear named password(s)
7: List named passwords
8: Get passwords state
99: Exit
Enter choice:
```

**Table A-3** *Password Operations*

Operation	Description
1: <i>Set shim password</i>	Sets the application password. This is the password of the user account you are using to authenticate into the connected system with.
2: <i>Clear shim password</i>	Clears the application password.

Operation	Description
3: <i>Set Remote Loader password</i>	<p>The Remote Loader password is used to control access to the Remote Loader instance.</p> <p>Enter the Remote Loader password, then confirm the password by typing it again.</p>
4: <i>Clear Remote Loader password</i>	<p>Clears the Remote Loader password so no Remote Loader password is set on the Driver object.</p>
5: <i>Set named password</i>	<p>Allows you to store a password or other pieces of security information on the driver. See <a href="#">Section 7.7, "Storing Driver Passwords Securely with Named Passwords,"</a> on page 80 for more information.</p> <p>There are four prompts to fill in:</p> <ul style="list-style-type: none"> <li>◆ <i>Enter password name:</i></li> <li>◆ <i>Enter password description:</i></li> <li>◆ <i>Enter password:</i></li> <li>◆ <i>Confirm password:</i></li> </ul>
6: <i>Clear named passwords</i>	<p>Clears a specified named password or all named passwords that are stored on the driver object: <i>Do you want to clear all named passwords? (yes/no).</i></p> <p>If you enter Yes, all Named Passwords are cleared. If you enter No, you are prompted to specify the password name that you want to clear.</p>
7: <i>List named passwords</i>	<p>Lists all named passwords that are stored on the driver object. It lists the password name and the password description.</p>
8: <i>Get password state</i>	<p>Lists if a password is set for:</p> <ul style="list-style-type: none"> <li>◆ Driver Object password</li> <li>◆ Application password</li> <li>◆ Remote loader password</li> </ul> <p>The dxcmnd utility allows you to set the Application password and the Remote Loader password. You cannot set the Driver Object password with this utility. It shows if the password has been set or not.</p>
99: <i>Exit</i>	<p>Exits the current menu and takes you back to the Driver options.</p>

**Figure A-3** Cache Operations

```
Enter choice: 14

Select a cache operation

1: Get driver cache limit
2: Set driver cache limit
3: View cached transactions
4: Delete cached transactions
99: Exit

Enter choice:
```

**Table A-4** Cache Operations

Operation	Description
1: <i>Get driver cache limit</i>	Displays the current cache limit that is set for the driver.
2: <i>Set driver cache limit</i>	Sets the driver cache limit in kilobytes. A value of 0 is unlimited.
3: <i>View cached transactions</i>	A text file is created with the events that are stored in cache. You can select the number of transactions to view. <ul style="list-style-type: none"><li>◆ <i>Enter option token (default=0):</i></li><li>◆ <i>Enter maximum transactions records to return (default=1):</i></li><li>◆ <i>Enter name of file for response:</i></li></ul>
4: <i>Delete cached transactions</i>	Deletes the transactions stored in cache. <ul style="list-style-type: none"><li>◆ <i>Enter position token (default=0):</i></li><li>◆ <i>Enter event-id value of first transaction record to delete (optional):</i></li><li>◆ <i>Enter number of transaction records to delete (default=1):</i></li></ul>
99: <i>Exit</i>	Exits the current menu and takes you back to the Driver options.

**Figure A-4** Log Event Operations

```
Select a log events operation

1: Set driver set log events
2: Reset driver set log events
3: Set driver log events
4: Reset driver log events
99: Exit

Enter choice:
```

**Table A-5** *Log Events Operations*

<b>Operation</b>	<b>Description</b>
1: <i>Set driver set log events</i>	Allows you to log driver set events through Novell Audit. There are 49 items you can select to log. See <a href="#">Table A-6 on page 118</a> for a list of these options.  Type the number of the item you want to log. After the items are selected, enter 99 to accept the selections.
2: <i>Reset driver set log events</i>	Resets all of the log event options.
3: <i>Set driver log events</i>	Allows you to log driver events through Novell Audit. There are 49 items to select to log. See <a href="#">Table A-6 on page 118</a> for a list of these options.  Type the number of the item you want to log. After the items are selected, enter 99 to accept the selections.
4: <i>Reset driver log events</i>	Resets all of the log event options.
99: <i>Exit</i>	Exits the log events operations menu.

**Table A-6** *Driver Set and Driver Log Events*

<b>Options</b>
1: Status success
2: Status retry
3: Status warning
4: Status error
5: Status fatal
6: Status other
7: Query elements
8: Add elements
9: Remove elements
10: Modify elements
11: Rename elements
12: Move elements
13: Add-association elements
14: Remove-association elements
15: Query-schema elements
16: Check-password elements

---

**Options**

---

- 17: Check-object-password elements
  - 18: Modify-password elements
  - 19: Sync elements
  - 20: Pre-transformed XDS document from shim
  - 21: Post input transformation XDS document
  - 22: Post output transformation XDS document
  - 23: Post event transformation XDS document
  - 24: Post placement transformation XDS document
  - 25: Post create transformation XDS document
  - 26: Post mapping transformation <inbound> XDS document
  - 27: Post mapping transformation <outbound> XDS document
  - 28: Post matching transformation XDS document
  - 29: Post command transformation XDS document
  - 30: Post-filtered XDS document <Publisher>
  - 31: User agent XDS command document
  - 32: Driver resync request
  - 33: Driver migrate from application
  - 34: Driver start
  - 35: Driver stop
  - 36: Password sync
  - 37: Password request
  - 38: Engine error
  - 39: Engine warning
  - 40: Add attribute
  - 41: Clear attribute
  - 42: Add value
  - 43: Remove value
  - 44: Merge entire
  - 45: Get named password
  - 46: Reset Attributes
  - 47: Add Value - Add Entry
  - 48: Set SSO Credential
-

---

**Options**

---

49: Clear SSO Credential

50: Set SSO Passphrase

51: User defined IDs

99: Accept checked items

---

**Table A-7** *Enter Table Title Here*

---

Options	Description
1: <i>Get available job definitions</i>	Allows you to select an existing job.  <i>Enter the job number:</i>  <i>Do you want to filter the job definitions by containment? Enter Yes or No</i>  <i>Enter name of the file for response:</i>  Examples:  NetWare: <code>sys:\files\user.log</code>  Windows: <code>c:\files\user.log</code>  Linux: <code>/files/user.log</code>
2: <i>Operations on specific job object</i>	Allows you to perform operations for a specific job.

---

## A.2 Command Line Mode

The command line mode allows you to use script or batch files. [Table A-8 on page 120](#) lists the different options that are available.

To use the command line options, decide which items you want to use and string them together.

Example: `dxcmd -user admin.headquarters -host 10.0.0.1 -password n0vell -start test.driverset.headquarters`

This example command starts the driver.

**Table A-8** *Command Line Options*

---

Option	Description
<b>Configuration</b>	
-user <i>&lt;user name&gt;</i>	Specify the name of a user with administrative rights to the drivers you want to test.
-host <i>&lt;name or IP address&gt;</i>	Specify the IP address of the server where the driver is installed.
-password <i>&lt;user password&gt;</i>	Specify the password of the user specified above.

---



Option	Description
-port <port number>	Specify a port number, if the default port is not used.
-q <quiet mode>	Displays very little information when a command is executed.
-v <verbose mode>	Displays detailed information when a command is executed.
-s <stdout>	Writes the results of the <code>dxcmd</code> command to <code>stdout</code> .
-? <show this message>	Displays the help menu.
-help <show this message>	Displays the help menu.
<b>Actions</b>	
-start <driver dn>	Starts the driver.
-stop <driver dn>	Stops the driver.
-getstate <driver dn>	Shows the state of the driver as running or stopped.
-getstartoption <driver dn>	Shows the startup option of the driver.
-setstartoption <driver dn> <disabled manual auto> <resync noresync>	Sets how the driver starts if the server is rebooted. Sets whether the objects are to be resynchronized when the driver restarts.
-getcachelimit <driver dn>	Lists the cache limit set for the driver.
-setcachelimit <driver dn> <0 or positive integer>	Sets the cache limit for the driver.
-migrateapp <driver dn> <filename>	Processes an XML document that contains a query command.  Create the XML document that contains a query command by using the Novell <code>nds.dtd</code> ( <a href="http://www.novell.com/documentation/idm35/index.html?page=/documentation/idm35/policy_dtd/data/dtdndsoverview.html#dtdndsoverview">http://www.novell.com/documentation/idm35/index.html?page=/documentation/idm35/policy_dtd/data/dtdndsoverview.html#dtdndsoverview</a> ).
-setshimpassword <driver dn> <password>	Sets the application password. This is the password of the user account you are using to authenticate into the connected system with.
-clearshimpassword <driver dn> <password>	Clears the application password.
-setremoteloaderpassword <driver dn> <password>	Sets the Remote Loader password.  The Remote Loader password is used to control access to the Remote Loader instance.
<clearremoteloaderpassword <driver dn>	Clears the Remote Loader password.

Option	Description
-sendcommand <driver dn> <input filename> <output filename>	<p>Processes an XDS command document.</p> <p>Specify the XDS command document as the input file.</p> <p>Examples:</p> <p>NetWare: sys:\files\user.xml</p> <p>Windows: c:\files\user.xml</p> <p>Linux: /files/user.log</p> <p>Specify the output filename to see the results.</p> <p>Examples:</p> <p>NetWare: sys:\files\user.log</p> <p>Windows: c:\files\user.log</p> <p>Linux: /files/user.log</p>
-sendevent <driver dn> <input filename>	<p>Submits a document to the driver's Subscriber channel, bypassing the driver cache. The document is processed ahead of anything that might be in the cache at the time of the submission. It also means that the submission fails if the driver is not running.</p>
-queueevent <driver dn> <input filename>	<p>Submits a document to the driver's Subscriber channel by queuing the document in the driver cache. The document gets processed after anything that might be in the cache at the time of the submission. The submission won't fail if the driver isn't running.</p>
-setlogevents <dn> <integer ...>	<p>Sets Novell Audit log events on the driver. The integer is the option of the item to log. See <a href="#">Table A-6 on page 118</a> for the list of the integers to enter.</p>
-clearlogevents <dn>	<p>Clears all Novell Audit log events that are set on the driver.</p>
-setdriverset <driver set dn>	<p>Associates a driver set with the server.</p>
-cleardriverset	<p>Clears the driver set association from the server.</p>
-getversion	<p>Shows the version of Identity Manager that is installed.</p>
-initdriver object <dn>	<p>Performs an internal initialization of data on a new Driver object. This is only for testing purposes.</p>
-setnamedpassword <driver dn> <name> <password> [description]	<p>Sets named passwords on the driver object. You specify the name, the password, and the description of the named password.</p>
-clearnamedpassword <driver dn> <name>	<p>Clears a specified named password.</p>
-startjob <job dn>	<p>Starts the specified job.</p>

Option	Description
-abortjob <job dn>	Aborts the specified job.
-getjobrunningstate <job dn>	Returns the specified job's running state.
-getjobenabledstate <job dn>	Returns the specified job's enabled state.
-getjobnextruntime <job dn>	Returns the specified job's next run time.
-updatejob <job dn>	Updates the specified job.
-clearallnamedpasswords <driver dn>	Clears all named passwords set on a specific driver.

If a command line is executed successfully, it returns a zero. If the command line returns anything other than zero, it is an error. For example 0 means success, and -641 means invalid operation. -641 is an eDirectory error code. [Table A-9 on page 123](#) contains other values for specific command line options.

**Table A-9** *Command Line Option Values*

Command Line Option	Values
-getstate	0- stopped 1- starting 2- running 3- shutting down 11- get schema Anything else that is returned is an error.
-getstartoption	0- disabled 1- manual 2- auto Anything else that is returned is an error.
-getcachelimit	0- unlimited Anything else that is returned is an error.
-getjobrunningstate	0- stopped 1- running Anything else that is returned is an error.
-getjobenabledstate	0- disabled 1- enabled 2- configuration error Anything else that is returned is an error.


---

Command Line Option	Values
-getjobnextruntime	Return is the next scheduled time for the job in eDirectory time format (number of seconds since 00:00:00 Jan 1, 1970UTC).

---

# Properties of the Driver

There are many different fields and values for the driver. Sometimes the information is displayed differently in iManager than in Designer. This section is a reference for all of the fields on the driver as displayed in iManager and Designer.

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

- ◆ [Section B.1, “Driver Configuration,” on page 125](#)
- ◆ [Section B.2, “Global Configuration Values,” on page 130](#)
- ◆ [Section B.3, “Named Passwords,” on page 132](#)
- ◆ [Section B.4, “Engine Control Values,” on page 132](#)
- ◆ [Section B.5, “Log Level,” on page 134](#)
- ◆ [Section B.6, “Driver Image,” on page 135](#)
- ◆ [Section B.7, “Security Equals,” on page 135](#)
- ◆ [Section B.8, “Filter,” on page 136](#)
- ◆ [Section B.9, “Edit Filter XML,” on page 136](#)
- ◆ [Section B.10, “Misc,” on page 137](#)
- ◆ [Section B.11, “Excluded Users,” on page 137](#)
- ◆ [Section B.12, “Driver Manifest,” on page 138](#)
- ◆ [Section B.13, “Driver Inspector,” on page 138](#)
- ◆ [Section B.14, “Driver Cache Inspector,” on page 139](#)
- ◆ [Section B.15, “Inspector,” on page 139](#)
- ◆ [Section B.16, “Server Variables,” on page 139](#)

## B.1 Driver Configuration

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Driver Configuration*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and click *Properties > Driver Configuration*.

There are different sections under *Driver Configuration*. Each section is listed in a table. The table contains a description of the fields, and the default value or an example of what value should be specified in the field.

## B.1.1 Driver Module



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

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Driver Configuration > Driver Module*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > Driver Configuration*.
- 2 Select the *Driver Module* tab.

Option	Description
<i>Java</i>	Used to specify the name of the Java class that is instantiated for the shim component of the driver. This class can be located in the <code>classes</code> directory as a class file, or in the <code>lib</code> directory as a <code>.jar</code> file. If this option is selected, the driver is running locally.
<i>Native</i>	Used to specify the name of the <code>.dll</code> file that is instantiated for the application shim component of the driver. If this option is selected, the driver is running locally.
<i>Connect to Remote Loader</i>	Used when the driver is connecting remotely to the connected system.
 <i>Remote Loader Client Configuration for Documentation</i>	 Includes the Remote Loader client configuration information in the driver documentation that is generated by Designer.

## B.1.2 Driver Object Password

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Driver Configuration > Driver Object Password > Set Password*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and click *Properties > Driver Configuration*.
- 2 Click *Driver Module > Connect to Remote Loader > Driver Object Password > Set Password*.

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

### B.1.3 Authentication



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









In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Driver Configuration > Authentication*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > Driver Configuration*.
- 2 Click *Authentication*.

Option	Description
<i>Authentication ID</i> or  <i>User ID</i>	Specify a user application ID. This ID is used to pass Identity Vault subscription information to the application.  Example: Administrator
<i>Authentication Context</i> or  <i>Connection Information</i>	Specify the IP address or name of the server the application shim should communicate with.  The connection string uses the following format: <hostname or IP address>:<Port Number> Example: //PSServer:9000  To enable failover and loadbalancing, you can supply multiple server connection strings separated by a comma. Example: //PSServer:9000,//111.222.3.4:9000

Option	Description
<i>Remote Loader Connection Parameters</i> or  <i>Host name</i>  <i>Port</i>  <i>KMO</i>  <i>Other parameters</i>	Used only if the driver is connecting to the application through the remote loader. The parameter to enter is hostname=xxx.xxx.xxx.xxx port=xxxx kmo=certificatename, when the host name is the IP address of the application server running the Remote Loader server and the port is the port the remote loader is listening on. The default port for the Remote Loader is 8090.  The kmo entry is optional. It is only used when there is an SSL connection between the Remote Loader and the Metadirectory engine.  <b>Example:</b> hostname=10.0.0.1 port=8090 kmo=IDMCertificate
<i>Driver Cache Limit (kilobytes)</i> or  <i>Cache limit (KB)</i>	Specify the maximum event cache file size (in KB). If it is set to zero, the file size is unlimited.   Click <i>Unlimited</i> to set the file size to unlimited in Designer.
<i>Application Password</i> or  <i>Set Password</i>	Specify the password for the user object listed in the <i>Authentication ID</i> field.
<i>Remote Loader Password</i> or  <i>Set Password</i>	Used only if the driver is connecting to the application through the Remote Loader. The password is used to control access to the Remote Loader instance. It must be the same password specified during the configuration of the Remote Loader on the connected system.

## B.1.4 Startup Option

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

In iManager:


- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Driver Configuration > Startup Option*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > Driver Configuration*.
- 2 Click *Startup Option*.

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



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

## B.1.5 Driver Parameters

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Driver Configuration > Driver Parameters*.

In Designer:

- 1 Open a project in the modeler, then right-click the driver line and select *Properties > Driver Configuration*.
- 2 Click *Driver Parameters*.

Parameter	Description
<b>Driver Settings</b>	
Driver parameters for server: Server-name	Specifies the name of the server on which the driver resides.
<i>Edit XML</i>	Allows you to edit the PeopleSoft driver parameters through an XML editor.
<i>Schema CI Name</i>	Allows you to list the Data Schema Component Interfaces that the driver synchronizes. Use the <i>Plus</i> icon to add an item, the <i>Minus</i> icon to delete an item, and the <i>Pen</i> icon to edit an item. The default is <i>DIRXML_SCHEMA01</i> .
<i>Data Record ID Field</i>	Specifies the name of the data record definition that uniquely identifies a PeopleSoft object. The default is <i>Assoc_ID</i> .
<i>Use Case-Sensitive Search</i>	Controls if the driver uses case-sensitive matching criteria while evaluating search attribute matches. The default is <i>No Case-Sensitive Matching</i> .
<b>Subscriber Setting</b>	
<i>Allow "add" events</i>	Subscriber Add events are implemented by invoking the Component Interface Create method (if present). If you want the driver to allow Subscriber channel Add events, select <i>Allow Subscriber add</i> . The default is <i>Disallow Subscriber add</i> .

Parameter	Description
Data Record ID Field Default Value	Specifies a default value for the Schema CI key field. This parameter is used only for Subscriber channel Add events. The default is <i>New</i> .
<i>Allow "delete" events</i>	Subscriber Delete events are implemented by invoking the Component Interface Delete method (if present). If you want the driver to allow Subscriber channel Delete events, select <i>Allow Subscriber delete</i> . The default is <i>Disallow Subscriber delete</i> .
<b>Publisher Settings</b>	
<i>Transaction CI Name</i>	Contains the name of the PeopleSoft CI object that defines the set of fields required for the driver Transaction interface. The set of fields in the specified transaction CI must contain the same fields and keys that are identified in the default transaction CI in order for the driver to work. The default is <i>DIRXML_TRANS01</i> .
<i>Driver Subset Identifier</i>	<p>The name of the field in the Data Schema Component Interface that uniquely identifies a PeopleSoft object. This name is used for All Data Schema CIs that are specified in the <i>Schema CI Name</i> parameter.</p> <p>The field identifies which transactions in the transaction CI are to be processed by the driver. When the driver reads a transaction CI record, it compares the values of the DIRXML_DRIVER field with this parameter value and only processes transactions that match.</p> <p>A match is determined by matching characters for the length of this parameter value. For instance, if this parameter is <i>NPSDriver</i> and the DIRXML_DRIVER field in a transaction is <i>NPSDriver1</i>, a match is made.</p> <p>This allows multiple drivers to utilize the same transaction CI, which in turn can be populated by multiple PeopleSoft applications or processes.</p>
<i>Publisher Polling Option</i>	The PeopleSoft driver supports two options for Publisher Transaction record polling. To choose an interval of seconds between polls, select <i>Utilize Interval Polling</i> . To use a crontab format, select <i>Utilize crontab Format Polling</i> .
<i>Queue Poll Interval (seconds) or Enter Queue Poll crontab Format String</i>	<p>If you select <i>Utilize Interval Polling</i>, this entry displays <i>Queue Poll Interval</i>. Specify the number of seconds between checks for available transactions to process. The default is 5.</p> <p>If you select <i>Utilize crontab Format Polling</i>, this entry displays <i>Enter Queue Poll crontab Format String</i>. Specify five required crontab field parameters that are separated by blank characters. The default value of <i>* * * * *</i> generates a poll every minute.</p>

## B.2 Global Configuration Values

Global configuration values (GCVs) allow you to specify settings for the Identity Manager features such as Password Synchronization and driver heartbeat, 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.

---

**IMPORTANT:** Password Synchronization settings are GCVs, but it's best to edit them in the graphical interface provided on the Server Variables page for the driver, instead of the GCV page. The Server Variables page that shows Password Synchronization settings is accessible as a tab like other driver parameters, or by clicking *Password Management > Password Synchronization*, searching for the driver, and clicking the driver name. The page contains online help for each Password Synchronization setting.

---

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Global Config Values*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > Global Config Values*.

For *Password Configuration*, you should only edit the first two settings listed here. The others are GCVs regarding Password Synchronization that are common to all drivers. They should be edited using iManager in *Passwords > Password Synchronization*, not here. Some of them have dependencies on each other that are represented only in the iManager interface. They are explained in “[Password Synchronization across Connected Systems](#)” in the *Novell Identity Manager 3.5.1 Administration Guide*.

**Table B-1** *Global Configuration Values > Password Configuration*

Option	Description
<i>Identity Manager accepts passwords from application</i>	If <i>True</i> , allows passwords to flow from the connected system to Identity Manager.
<i>Publish passwords to NDS password</i>	Use the password from the connected system to set the non-reversible NDS® password in eDirectory.
<i>Publish passwords to Distribution Password</i>	Use the password from the connected system to set the NMAS™ Distribution Password used for Identity Manager password synchronization.
<i>Require password policy validation before publishing passwords</i>	If <i>True</i> , applies NMAS password policies during publish password operations. The password is not written to the data store if it does not comply.
<i>Notify the user of password synchronization failure via e-mail</i>	If <i>True</i> , notify the user by e-mail of any Password Synchronization failures.
<i>Connected System or Driver Name</i>	The name of the connected system, application, or Identity Manager driver. This value is used by the e-mail notification templates.
<i>Password Failure Notification User</i>	Password Synchronization policies are configured to send e-mail notifications to the associated user when password updates fail. To send a notification copy, specify the DN of that user. Otherwise, leave this field blank.

---

Option	Description
<i>Active Users Container</i>	The name of the Organizational Unit object where Active users will be placed. Specify the DN of the OU object. Otherwise, leave this field blank.
<i>Inactive Users Container</i>	The name of the Organizational Unit object where the inactive users will be placed. Specify the DN of the OU object. Otherwise, leave this field blank.
<i>Active Employees Group</i>	The name of the Group object where Active Employee users will be added. Specify the DN of the object. Otherwise, leave this field blank.
<i>Active Managers Group</i>	The name of the Group object where Active Manager users will be added. Specify the DN of the object. Otherwise, leave this field blank.

## B.3 Named Passwords

Identity Manager allows you to store multiple passwords securely for a particular driver. This functionality is referred to as Named Passwords. Each different password is accessed by a key, or name.

You can also use the Named Passwords feature to store other pieces of information securely, such as a user name. To configured Named Passwords, see [Section 7.7, “Storing Driver Passwords Securely with Named Passwords,” on page 80](#).

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Named Passwords*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > Named Passwords*.

## B.4 Engine Control Values

The engine control values are a means through which certain default behaviors of the Metadirectory engine can be changed. The values can only be accessed if a server is associated with the Driver Set object.

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Engine Control Values*.

This option does not exist in Designer at this time.

**Table B-2** Engine Control Values

Option	Description
<i>Subscriber channel retry interval in seconds</i>	The Subscriber channel retry interval controls how frequently the Metadirectory engine retries the processing of a cached transaction after the application shim's Subscriber object returns a retry status.
<i>Qualified form for DN-syntax attribute values</i>	The qualified specification for DN-syntax attribute values controls whether values for DN-syntax attribute values are presented in unqualified slash form or qualified slash form. A True setting means the values are presented in qualified form.
<i>Qualified form from rename events</i>	The qualified form for rename events controls whether the new-name portion of rename events coming from the Identity Vault are presented to the Subscriber channel with type qualifiers. For example, CN=. A True setting means the names are presented in qualified form.
<i>Maximum eDirectory replication wait time in seconds</i>	The maximum eDirectory™ replication wait time controls the maximum time that the Metadirectory engine waits for a particular change to replicate between the local replica and a remote replica. This only affects operations where the Metadirectory engine is required to contact a remote eDirectory server in the same tree to perform an operation and might need to wait until some change has replicated to or from the remote server before the operation can be completed (for example, object moves when the Identity Manager server does not hold the master replica of the moved object; file system rights operations for Users created from a template.)
<i>Use non-compliant backwards-compatible mode for XSLT</i>	<p>This control sets the XSLT processor used by the Metadirectory engine to a backwards-compatible mode. The backwards-compatible mode causes the XSLT processor to use one or more behaviors that are not XPath 1.0 and XSLT 1.0 standards-compliant. This is done in the interest of backwards-compatibility with existing DirXML® style sheets that depend on the non-standard behaviors.</p> <p>For example, the behavior of the XPath “!=” operator when one operand is a node-set and the other operand is other than a node-set is incorrect in DirXML releases up to and including Identity Manager 2.0. This behavior has been corrected; however, the corrected behavior is disabled by default through this control in favor of backwards-compatibility with existing DirXML style sheets.</p>
<i>Maximum application objects to migrate at once</i>	<p>This control is used to limit the number of application objects that the Metadirectory engine requests from an application during a single query that is performed as part of a Migrate Objects from Application operation.</p> <p>If java.lang.OutOfMemoryError errors are encountered during a Migrate from Application operation, this number should be set lower than the default. The default is 50.</p> <hr/> <p><b>NOTE:</b> This control does not limit the number of application objects that can be migrated; it merely limits the batch size.</p> <hr/>

Option	Description
<i>Set creatorsName on objects created in Identity Vault</i>	<p>This control is used by the Identity Manager engine to determine if the creatorsName attribute should be set to the DN of this driver on all objects created in the Identity Vault by this driver.</p> <p>Setting the creatorsName attribute allows for easily identifying objects created by this driver, but also carries a performance penalty. If not set, the creatorsName attribute defaults to the DN of the NCP™ Server object that is hosting the driver.</p>
<i>Write pending associations</i>	<p>This control determines whether the Identity Manager engine writes a pending association on an object during Subscriber channel processing.</p> <p>Writing a pending association confers little or no benefit but does incur a performance penalty. Nevertheless, the option exists to turn it on for backward compatibility.</p>
<i>Use password event values</i>	<p>This control determines the source of the value reported for the nspmDistributionPassword attribute for Subscriber channel Add and Modify events.</p> <p>Setting the control to False means that the current value of the nspmDistributionPassword is obtained and reported as the value of the attribute event. This means that only the current password value is available. This is the default behavior.</p> <p>Setting the control to True means that the value recorded with the eDirectory event is decrypted and is reported as the value of the attribute event. This means that both the old password value (if it exists) and the replacement password value at the time of the event are available. This is useful for synchronizing passwords to certain applications that require the old password to enable setting a new password.</p>
<i>Enable password synchronization status reporting</i>	<p>This control determines whether the Identity Manager engine reports the status of Subscriber channel password change events.</p> <p>Reporting the status of Subscriber channel password change events allows applications such as the Identity Manager User Application to monitor the synchronization progress of a password change that should be synchronized to the connected application.</p>

## B.5 Log Level

Every driver set and driver has a log level field where you can define the level of errors that should be tracked. The level you indicate here determines which messages are available to the logs. By default, the log level is set to track error messages (this also includes fatal messages). Change the log level if you want to track additional message types.

Novell® recommends that you use Novell Audit instead of setting the log levels. See “[Integrating Identity Manager with Novell Audit](#)” in the *Identity Manager 3.5.1 Logging and Reporting*.


In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Log Level*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > Driver Log Level*.

---

Option	Description
<i>Use log settings from the DriverSet</i>	If this is selected, the driver logs events as the options are set on the Driver Set object.
<i>Log errors</i>	Logs just errors
<i>Log errors and warnings</i>	Logs errors and warnings
<i>Log specific events</i>	Logs the events that are selected. Click the  icon to see a list of the events.
<i>Only update the last log time</i>	Updates the last log time.
<i>Logging off</i>	Turns logging off for the driver.
<i>Turn off logging to DriverSet, Subscriber and Publisher logs</i>	If selected, turns all logging off for this driver on the Driver Set object, Subscriber channel, and the Publisher channel.
<i>Maximum number of entries in the log (50-500)</i>	Number of entries in the log. The default value is 50.

---

## B.6 Driver Image

Allows you to change the image associated with the driver. You can browse and select a different image from the default image.

The image associated with a driver is used by the Identity Manager Overview plug-in when showing the graphical representation of your Identity Manager configuration. Although storing an image is optional, it makes the overview display more intuitive.

---

**NOTE:** The driver image is maintained when a driver configuration is exported.

---

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Driver Image*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > iManager Icon*.

## B.7 Security Equals

Use the Security page to view or change the list of objects that the driver is explicitly security equivalent to. This object effectively has all rights of the listed objects.

If you add or delete an object in the list, the system automatically adds or deletes this object in that object's "Security Equal to Me" property. You don't need to add the [Public] trustee or the parent containers of this object to the list, because this object is already implicitly security equivalent to them.

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Security Equals*.

Designer does not list the users the driver is security equals to.

## B.8 Filter

Launches the Filter editor. You can edit the Filter from this tab.

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Filter*.

The filter editor is accessed through the outline view in Designer.

- 1 In an open project, click the *Outline* tab.
- 2 Select the driver you want to manage the filter for, then click the plus sign to the left.
- 3 Double-click the *Filter* icon to launch the Filter Editor.

## B.9 Edit Filter XML

Allows you to edit the filter directly in XML instead of using the Filter Editor.

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Filter*.

You can edit the Filter in XML through the Filter Editor.

- 1 In an open project, click the *Outline* tab.
- 2 Select the driver you want to manage the filter for, then click the plus sign to the left.
- 3 Double-click the *Filter* icon and to launch the Filter Editor, then click *XML Source* at the bottom of the Filter Editor.



## B.10 Misc


Allows you to add a trace level to your driver. With the trace level set, DSTRACE displays the Identity Manager events as the Metadirectory engine processes the events. The trace level only affects the driver it is set for. Use the trace level for troubleshooting issues with the driver when the driver is deployed. DSTRACE displays the output of the specified trace level.

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Misc*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > Trace*.

Option	Description
<i>Trace level</i>	Increases the amount of information displayed in DSTRACE. Trace level 1 shows errors, but not the cause of the errors. If you want to see password synchronization information, set the trace level to 5.
<i>Trace file</i>	When a value is set in this field, all Java information for the driver is written to the file. The value for this field is the path for that file.  As long as the file is specified, Java information is written to this file. If you do not need to debug Java, leave this field blank.
<i>Trace file size limit</i>	Allows you to set a limit for the Java trace file. If you set the file size to Unlimited, the file grows in size until there is no disk space left.  <b>NOTE:</b> The trace file is created in multiple files. Identity Manager automatically divides the maximum file size by ten and creates ten separate files. The combined size of these files equals the maximum trace file size.
<i>Trace name</i>	Driver trace messages are prepended with the value entered in this field.
 <i>Use setting from Driver Set</i>	This option is only available in Designer. It allows the driver to use the same setting that is set on the Driver Set object.

## B.11 Excluded Users

Use this page to create a list of users or resources that are not replicated to the application. Novell recommends that you add all objects that represent an administrative role to this list (for example, the Admin object).

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.

- 3 Click *Edit Properties > Excluded Users*.

Designer does not list the excluded users.

## B.12 Driver Manifest

The driver manifest is like a resumé for the driver. It states what the driver supports, and includes a few configuration settings. The driver manifest is created by default when the Driver object is imported. A network administrator usually does not need to edit the driver manifest.

In iManager:

- 1 Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
- 2 Browse to the driver, then click the upper right corner of the driver icon.
- 3 Click *Edit Properties > Driver Manifest*.

In Designer:

- 1 Open a project in the Modeler, then right-click the driver line and select *Properties > Driver Manifest*.

## B.13 Driver Inspector

The Driver Inspector page displays information about all of the objects associated with the driver.

- ♦ **Driver:** A link to run the *Driver Overview* on the driver that is being inspected.
- ♦ **Driver Set:** A link to run the *Driver Set Overview* of the driver set that holds the driver.
- ♦ **Delete:** Deletes the associations of the selected objects.
- ♦ **Refresh:** Select this option to re-read all of the objects associated with the driver and refresh the displayed information.
- ♦ **Actions:** Allows you to perform actions on the objects associated with the driver. Click *Actions* to expand the menu, which includes:
  - ♦ **Show All Associations:** Displays all objects associated with the driver.
  - ♦ **Filter for Disabled Associations:** Displays all objects associated with the driver that have a Disabled state.
  - ♦ **Filter for Manual Associations:** Displays all objects associated with the driver that have a Manual state.
  - ♦ **Filter for Migrate Associations:** Displays all objects associated with the driver that have a Migrate state.
  - ♦ **Filter for Pending Associations:** Displays all objects associated with the driver that have a Pending state.
  - ♦ **Filter for Processed Associations:** Displays all objects associated with the driver that have a Processed state.
  - ♦ **Filter for Undefined Associations:** Displays all objects associated with the driver that have an Undefined state.
  - ♦ **Association Summary:** Displays the state of all objects associated with the driver.

- ◆ **Object DN:** Displays the DN of the associated objects.
- ◆ **State:** Displays the association state of the object.
- ◆ **Object ID:** Displays the value of the association.

## B.14 Driver Cache Inspector

The Driver Cache Inspector page uses a table format to display information about the cache file that stores events while the driver is stopped.

- ◆ **Driver:** A link to run the *Driver Overview* on the driver that is associated with this cache file.
- ◆ **Driver Set:** A link to run the *Driver Set Overview* on the driver set that holds the driver.
- ◆ **Driver's cache on:** Lists the server object that contains this instance of the cache file.
- ◆ **Start/Stop Driver icons:** Displays the current state of the driver and allows you to start or stop the driver.
- ◆ **Edit icon:** Allows you to edit the properties of the currently selected Server object.
- ◆ **Delete:** Deletes the selected items from the cache file.
- ◆ **Refresh:** Select this option to re-read the cache file and refresh the displayed information.
- ◆ **Show:** Limits the number of items to be displayed. The options are:
  - ◆ 25 per page
  - ◆ 50 per page
  - ◆ 100 per page
  - ◆ Other: Allows you to specify a desired number.
- ◆ **Actions:** Allows you to perform actions on the entries in the cache file. Click *Actions* to expand the menu, which includes:
  - ◆ **Expand All:** Expands all of the entries displayed in the cache file.
  - ◆ **Collapse All:** Collapses all of the entries displayed in the cache file.
  - ◆ **Go To:** Allows you to access a specified entry in the cache file. Specify the entry number, then click *OK*.
  - ◆ **Cache Summary:** Summarizes all of the events stored in the cache file.

## B.15 Inspector

The Inspector displays information about the connected system without directly accessing the system. Designer does not have this option.

## B.16 Server Variables

This page lets you enable and disable Password Synchronization and the associated options for the selected driver.

When setting up Password Synchronization, consider both the settings on this page for an individual driver and the Universal Password Configuration options in your password policies.

This page lets you control which password Identity Manager updates directly, either the Universal Password for an Identity Vault, or the Distribution Password used for password synchronization by Identity Manager.

However, Novell Modular Authentication Service (NMAS) controls whether the various passwords inside the Identity Vault are synchronized with each other. Password Policies are enforced by NMAS, and they include settings for synchronizing Universal Password, NDS Password, Distribution Password, and Simple Password.

To change these settings in iManager:

- 1 In iManager, select *Passwords > Password Policies*.
- 2 Select a password policy, then click *Edit*.
- 3 Select *Universal Password*.

This option is available from a drop-down list or a tab, depending on your version of iManager and your browser.

- 4 Select *Configuration Options*, make changes, then click *OK*.

---

**NOTE:** Enabling or disabling options on this page corresponds to values of True or False for certain global configuration values (GCVs) used for password synchronization in the driver parameters. Novell recommends that you edit them here in the graphical interface, instead of on the GCVs page. This interface helps ensure that you don't set conflicting values for the password synchronization GCVs.

---

---

Option	Description
<i>Identity Manager accepts password (Publisher Channel)</i>	<p>If this option is enabled, Identity Manager allows passwords to flow from the connected system driver into the Identity Vault data store.</p> <p>Disabling this option means that no <i>&lt;password&gt;</i> elements are allowed to flow to Identity Manager. They are stripped out of the XML by a password synchronization policy on the Publisher channel.</p> <p>If this option is enabled, and the option below it for Distribution Password is disabled, a <i>&lt;password&gt;</i> value coming from the connected system is written directly to the Universal Password in the Identity Vault if it is enabled for the user. If the user's password policy does not enable Universal Password, the password is written to the NDS Password.</p>

---

Option	Description
<i>Use Distribution Password for password synchronization</i>	<p>To use this setting, you must have a version of eDirectory that supports Universal Password, regardless of whether you have enabled Universal Password in your password policies.</p> <p>If this option is enabled, a password value coming from the connected system is written to the Distribution Password. The Distribution Password is reversible, which means that it can be retrieved from the Identity Vault data store for password synchronization. It is used by Identity Manager for bidirectional password synchronization with connected systems. For Identity Manager to distribute passwords to connected systems, this option must be enabled.</p> <p>NMAS and Password policies control whether the Distribution Password is synchronized with other passwords in the Identity Vault. By default, the Distribution Password is the same as the Universal Password in the Identity Vault.</p> <p>If the password in the Identity Vault is to be independent of Password Synchronization, so that Identity Manager is a conduit only for synchronizing passwords among connected systems, change this default setting. In the Universal Password Configuration Options in a Password policy, disable <i>Synchronize Universal Password with Distribution Password</i>. This use of Identity Manager Password Synchronization is also referred to as “tunneling.”</p>
<i>Accept password only if it complies with user's Password Policy</i>	<p>To use this setting, users must have a Password policy assigned that has Universal Password enabled, and Advanced Password Rules enabled and configured.</p> <p>If this option is chosen, Identity Manager does not write a password from this connected system to the Distribution Password in the Identity Manager data store or publish it to connected systems unless the password complies with the user's Password policy.</p> <p>By using the notification option that is also on this page, you can inform users when a password is not set because it is not compliant.</p>

Option	Description
<p><i>If password does not comply, ignore Password Policy on the connected system by resetting user's password to the Distribution Password</i></p>	<p>This option lets you enforce Password policies on the connected system by replacing a password that does not comply. If you select this option, and a user's password on the connected system does not comply with the user's Password policy, Identity Manager resets the password on the connected system by using the Distribution Password from the Identity Vault data store.</p> <p>Keep in mind that if you do not select this option, user passwords can become out-of-sync on connected systems.</p> <p>By using the notification option that is also on this page, you can inform users when a password is not set or reset. Notification is especially helpful for this option. If the user changes to a password that is allowed by the connected system but rejected by Identity Manager because of the Password policy, the user won't know that the password has been reset until the user receives a notification or tries to log in to the connected system with the old password.</p> <hr/> <p><b>NOTE:</b> Consider the connected system's password policies when deciding whether to use this option. Some connected systems might not allow the reset because they don't allow you to repeat passwords.</p>
<p><i>Always accept password; ignore Password Policies</i></p>	<p>If you select this option, Identity Manager does not enforce the user's Password policy for this connected system. Identity Manager writes the password from this connected system to the Distribution Password in the Identity Vault data store, and distributes it to other connected systems, even if the password does not comply with the user's Password policy.</p>
<p><i>Application accepts passwords (Subscriber Channel)</i></p>	<p>If you select this option, the driver sends passwords from the Identity Vault data store to this connected system. This also means that if a user changes the password on a different connected system that is publishing passwords to the Distribution Password in the Identity Vault data store, the password is changed on this connected system.</p> <p>By default, the Distribution Password is the same as the Universal Password in the Identity Vault, so changes to the Universal Password made in the Identity Vault are also sent to the connected system.</p> <p>If you want the password in the Identity Vault to be independent of Password Synchronization, so that Identity Manager is a conduit only for synchronizing passwords among connected systems, you can change this default setting. In the Universal Password Configuration Options in a password policy, disable <i>Synchronize Universal Password with Distribution Password</i>. This use of Password Synchronization is also referred to as "tunneling."</p>
<p><i>Notify the user of password synchronization failure via-email</i></p>	<p>If you select this option, e-mail is sent to the user if a password is not synchronized, set, or reset. The e-mail that is sent to the user is based on an e-mail template. This template is provided by the Password Synchronization application. However, for the template to work, you must customize it and specify an e-mail server to send the notification messages.</p> <hr/> <p><b>NOTE:</b> To set up e-mail notification, select <i>Passwords &gt; Edit EMail Templates</i>.</p>