

Driver for Avaya PBX Implementation Guide

Novell® Identity Manager

4.0

October 15, 2010

www.novell.com



Legal Notices

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

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

Any products or technical information provided under this Agreement may be subject to U.S. export controls and the trade laws of other countries. You agree to comply with all export control regulations and to obtain any required licenses or classification to export, re-export or import deliverables. You agree not to export or re-export to entities on the current U.S. export exclusion lists or to any embargoed or terrorist countries as specified in the U.S. export laws. You agree to not use deliverables for prohibited nuclear, missile, or chemical biological weaponry end uses. See the [Novell International Trade Services Web page \(http://www.novell.com/info/exports/\)](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 © 2000-2010 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.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.
www.novell.com

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

About This Guide	9
1 Overview	11
1.1 The Role of the Avaya PBX Driver	11
1.2 What's Different about the Avaya PBX Driver?	12
1.3 Benefits of the Avaya PBX Driver	13
1.4 Basic Functionality of the Avaya PBX Driver	14
1.4.1 New Objects Used by the Driver	14
1.4.2 Overview of Driver Functionality	17
1.4.3 Assumptions about the PBX	18
1.4.4 What the Driver Does in the PBX	20
1.4.5 What Is Emulation Mode and Why Should I Use it?	21
1.4.6 Support for Manual Changes in the PBX	21
1.5 Support for Standard Driver Features	22
1.5.1 Local Platforms	22
1.5.2 Remote Platforms	22
1.5.3 Entitlements	23
2 Planning	25
2.1 Supported Avaya PBX Systems	25
2.2 Planning Issues for All Configurations	25
2.3 Planning for User Provisioning (Workforce Tree) Implementations	27
2.4 Planning Driver and Replica Placement on Your Servers	29
3 Installing the Driver Files	31
4 Creating a New Driver	33
4.1 Prerequisites	33
4.1.1 Selecting a Configuration File	33
4.1.2 Creating Containers	33
4.2 Creating the Driver in Designer	34
4.2.1 Importing the Driver Configuration File	34
4.2.2 Configuring the Driver	35
4.2.3 Using Emulation to Test the Driver	35
4.2.4 Deploying the Driver	35
4.2.5 Starting the Driver	36
4.3 Creating the Driver in iManager	36
4.3.1 Importing the Driver Configuration File	37
4.3.2 Configuring the Driver	39
4.3.3 Using Emulation to Test the Driver	39
4.3.4 Starting the Driver	40
4.4 Activating the Driver	40
5 Upgrading an Existing Driver	41
5.1 Supported Upgrade Paths	41
5.2 What's New in Version 4.0	41

5.3	Upgrade Procedure	41
6	Base Configuration	43
6.1	How to Use the Base Configuration	43
6.2	How the Base Configuration Works	43
6.2.1	How the Subscriber Channel Is Configured	46
6.2.2	How the Publisher Channel Is Configured	47
7	Workforce Tree Configuration	49
7.1	How to Use the Workforce Tree Configuration	49
7.2	How the Workforce Tree Configuration Works	49
7.2.1	How the Driver Uses the ObjectID to Identify and Update Users	52
7.2.2	How the Subscriber Channel Is Configured	52
7.2.3	How the Publisher Channel Is Configured	54
7.2.4	User Events That Can Trigger a Work Order	55
7.3	Using Log/Reporting Functionality to Report Warnings	56
8	Work Order Database Configuration	57
8.1	How to Use the Work Order Database Configuration	57
8.2	Creating a Work Order Database Configuration	57
8.2.1	How To Configure the Subscriber Channel	60
8.2.2	How To Configure the Publisher Channel	61
8.3	About Work Order Systems	61
8.4	Using the Log/Reporting Functionality to Report Warnings	61
9	Managing the Driver	63
9.1	Changing How Often the Driver Performs Work Orders	63
9.2	Changing the Location of PBX Site, Work Orders, User, and Extension Objects	63
9.3	Managing Existing Users	64
9.4	Common Management Tasks	65
10	Troubleshooting	67
10.1	Troubleshooting Driver Processes	67
10.2	Tree information in the WorkOrder DN of the Avaya PBX Driver throws an Exception for PBX WorkOrders	67
A	Driver Properties	69
A.1	Driver Configuration	69
A.1.1	Driver Module	69
A.1.2	Driver Object Password (iManager Only)	70
A.1.3	Authentication	70
A.1.4	Startup Option	71
A.1.5	Driver Parameters	72
A.1.6	ECMAScript	73
A.1.7	Global Configuration	73
A.2	Global Configuration Values	73

B	Schema for PBX Management	75
B.1	pbxSite Object	75
B.2	DirXML-nwoWorkOrder Object.	78
B.3	DirXML-pbxExtension Object.	83
B.4	DirXML-pbxAudixSubscriber Object.	85
B.5	User Objects and Their Identity Manager Associations	86

About This Guide

The Identity Manager driver for Avaya PBX lets you use Identity Manager to centrally manage PBX extensions and work orders and keep user phone numbers up-to-date. Additional driver features allow you to also configure voice mail through Audix. This guide explains what the Avaya driver does and how the driver can be customized through three sample configurations that are provided for instructional purposes.

This guide contains the following sections:

- ♦ [Chapter 1, “Overview,” on page 11](#)
- ♦ [Chapter 2, “Planning,” on page 25](#)
- ♦ [Chapter 3, “Installing the Driver Files,” on page 31](#)
- ♦ [Chapter 4, “Creating a New Driver,” on page 33](#)
- ♦ [Chapter 5, “Upgrading an Existing Driver,” on page 41](#)
- ♦ [Chapter 6, “Base Configuration,” on page 43](#)
- ♦ [Chapter 7, “Workforce Tree Configuration,” on page 49](#)
- ♦ [Chapter 8, “Work Order Database Configuration,” on page 57](#)
- ♦ [Chapter 9, “Managing the Driver,” on page 63](#)
- ♦ [Chapter 10, “Troubleshooting,” on page 67](#)
- ♦ [Appendix A, “Driver Properties,” on page 69](#)
- ♦ [Appendix B, “Schema for PBX Management,” on page 75](#)

Audience

This manual is for Novell Identity Manager administrators, Avaya PBX communications system administrators, and others who manage user accounts in eDirectory, PBX phone extensions, and PBX work orders.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with Novell Identity Manager. Please use the User Comments feature at the bottom of each page of the online documentation, or go to <http://www.novell.com/documentation/feedback.html> and enter your comments there.

Documentation Updates

For the most recent version of this document, see the [Identity Manager Identity Manager 4.0 Driver for Avaya PBX Implementation Guide Drivers Documentation Web site](http://www.novell.com/documentation/idm40drivers/index.html) (<http://www.novell.com/documentation/idm40drivers/index.html>).

Additional Documentation

For documentation on using Identity Manager, see the [Identity Manager Identity Manager 4.0 Documentation Web site](http://www.novell.com/documentation/idm40) (<http://www.novell.com/documentation/idm40>).

The Identity Manager driver for Avaya PBX (referred to as the Avaya PBX driver) lets you use the Identity Vault to centrally manage PBX extensions and work orders and keep user phone numbers up-to-date.

This configurable solution gives you the ability to automate work order and PBX processes, such as provisioning a phone extension for new users; moving, modifying or disabling existing extensions; disconnecting extensions, updating phone number data for User objects in the Identity Vault, and provisioning voice mail/Audix for phone extensions.

In this section:

- ♦ [Section 1.1, “The Role of the Avaya PBX Driver,” on page 11](#)
- ♦ [Section 1.2, “What’s Different about the Avaya PBX Driver?,” on page 12](#)
- ♦ [Section 1.3, “Benefits of the Avaya PBX Driver,” on page 13](#)
- ♦ [Section 1.4, “Basic Functionality of the Avaya PBX Driver,” on page 14](#)
- ♦ [Section 1.5, “Support for Standard Driver Features,” on page 22](#)

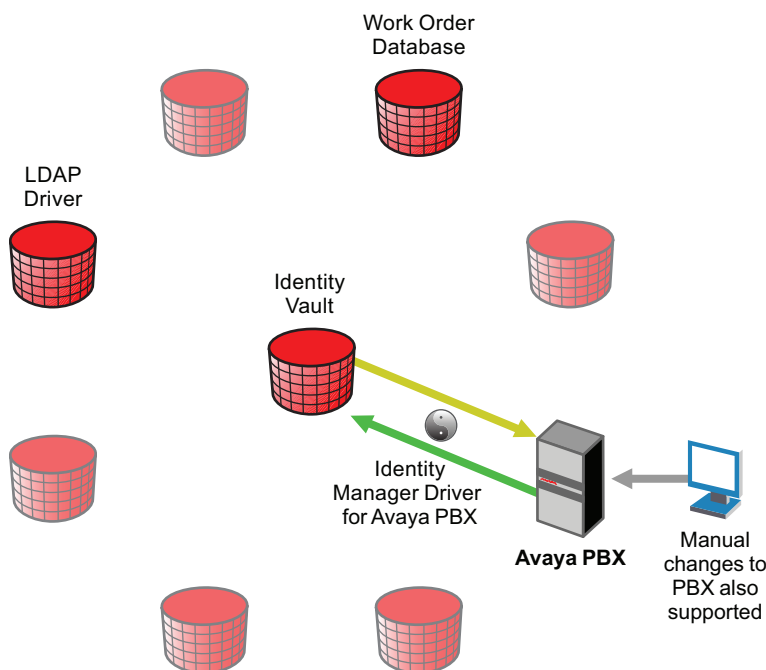
1.1 The Role of the Avaya PBX Driver

The Avaya PBX driver communicates with the PBX and carries out work orders by logging in just as a PBX administrator would. The driver uses information from the Identity Vault as well as customizable settings and policies to configure the extension correctly. After configuration is completed, the driver publishes the new phone number to the Identity Vault, and the phone can be delivered and physically installed.

The driver can be used in conjunction with other systems such as a human resources application or work order database to provide additional functionality.

The following diagram shows some of the systems you might have in your environment, and shows that you can connect the PBX to the Identity Vault by using the Avaya PBX driver.

Figure 1-1 The Avaya PBX Driver Connecting the Identity Vault To One or More PBX Sites



The terminal shown in the diagram demonstrates that you can still access the PBX and make changes manually, if necessary. However, with the driver in place, you shouldn't need to make changes directly in the PBX very often. The purpose of the driver is to allow you to use work orders to control changes in the PBX, so you don't need to make them manually. Work orders can be triggered by events elsewhere in your environment, depending on what other systems you have connected by using Identity Manager.

1.2 What's Different about the Avaya PBX Driver?

The Avaya PBX driver is different from some other drivers in the following ways:

- ♦ Through policies, the driver can map users to phone extensions, rather than users to users.

Some drivers are used for user account administration, mapping user accounts in one system to user accounts in another. The Avaya PBX driver can help you maintain correct phone information for user objects, but the PBX understands only extensions and does not contain the concept of a user account, so the user objects in Identity Vault must be connected to extensions in the PBX by using the Object ID in the work order. See [Section 9.3, "Managing Existing Users,"](#) on page 64.

- ♦ The ObjectID in the work order (such as user object entry ID, or employee ID) is the piece of information that allows a user object to be updated with new phone information after a task is performed in the PBX. This ID should be unique to the user, and a policy must be customized to match the attribute that you use.

The driver places the Object ID in the *Cable* field of the extension in the PBX. This field has a character limit of 5 characters.

Even if the user has an Identity Manager association for this driver through a policy, in most cases the user cannot be updated unless the ObjectID is entered correctly. In fact, a user object can only be updated if the ObjectID is correct, even if the user does not yet have an Identity Manager association for this driver. See [Section 9.3, “Managing Existing Users,” on page 64](#).

- ♦ The Publisher performs tasks in the Avaya PBX, rather than the Subscriber.

For many drivers, the Subscriber performs changes in the third-party application in response to events in the Identity Vault. However for this driver, the Publisher is the agent that performs work orders in the PBX. The Subscriber merely picks up events such as Add User events, creates work orders if configured to do so, and sends work orders to the Publisher if they are marked Send to Publisher or DoItNow.

There are several reasons for this design; for example, the Publisher has the ability to run at a certain time of day, which is desirable for allowing you to specify that work orders be performed after business hours.

- ♦ Creating a robust test environment can be a challenge (often, the only PBX available in an environment is the production PBX). If this is the case, you can use emulation mode for testing.

Another option is to set aside a certain range of extensions on the production PBX to use for testing, and use extra caution because it's not an entirely separate test environment.

- ♦ You can manage existing users without using the `Migrate into NDS` command. See [Section 9.3, “Managing Existing Users,” on page 64](#).

1.3 Benefits of the Avaya PBX Driver

Benefits of using the driver include the following:

- ♦ Eliminates redundant data entry.

If you use a work order system without the driver, you must enter the data twice, once in the work order and again in the PBX when performing the work order. With the driver, you enter the data once in the work order, and then the driver performs it for you, reducing the possibility of human error.

In fact, in a provisioning implementation you can avoid human intervention entirely for actions such as new hires, moves, and deletions; it can all be based on changes in a human resources application.

- ♦ Allows automated provisioning of extensions for new users.
- ♦ Allows you to use the Identity Vault as your work order database for PBX tasks, if you don't already have one.

An iManager interface lets you create and track PBX work orders.

- ♦ Helps reduce costs and data entry errors by letting you enforce business policies for phone usage.

For example, you could ensure that only users in the Localization, Sales, and Human Resources departments can make international calls. Another example could be making sure that extensions for call center employees are always non-DID extensions.

- ♦ Allows updates to be sent to the Identity Vault when an extension is changed directly in the PBX.

This means that after deploying the driver you can still make changes in the PBX manually when necessary.

When the driver prepares to perform work orders at the specified time, it queries the PBX to see if changes have been made. If a manual change has been made to a PBX extension, the driver can detect the change. If the PBX administrator who made the manual change also enters the user ID in the PBX to identify the user of the extension, then the driver can update the user object in the Identity Vault.

- ♦ Audix support to the Avaya driver. Through this support, the driver can add voice mail to an extension, remove voice mail from an extension, or modify the attributes of the voice mail of an extension.

For Audix support, the driver uses the `add`, `change` and `remove` the work order commands.


1.4 Basic Functionality of the Avaya PBX Driver

- ♦ [Section 1.4.1, “New Objects Used by the Driver,” on page 14](#)
- ♦ [Section 1.4.2, “Overview of Driver Functionality,” on page 17](#)
- ♦ [Section 1.4.3, “Assumptions about the PBX,” on page 18](#)
- ♦ [Section 1.4.4, “What the Driver Does in the PBX,” on page 20](#)
- ♦ [Section 1.4.5, “What Is Emulation Mode and Why Should I Use it?,” on page 21](#)
- ♦ [Section 1.4.6, “Support for Manual Changes in the PBX,” on page 21](#)

1.4.1 New Objects Used by the Driver

Using four new object classes in eDirectory, the Avaya PBX driver performs work orders, records the results, and provides extension information. The Avaya driver also supports Audix account provisioning. For a description of the schema for these objects, see [Appendix B, “Schema for PBX Management,” on page 75](#).

DirXML-pbxSite

 DirXML-pbxSite represents the PBX.

The driver depends on the information in PBX Site objects to know which sites to manage, and how to connect to them.

An iManager plug-in is provided to help you set up these sites. In iManager, select *PBX Utilities > Site Management*. The following figure shows an example of the interface for setting up a PBX site.

Figure 1-2 *The PBX Site Page*

The screenshot shows a web browser window titled "Novell iManager - Microsoft Internet Explorer". The address bar shows "File Edit View Favorites Tools Help". The main content area is titled "PBX Site: LabPBX". Below this, there is a tab labeled "PBX Site". The instructions state: "Configure this PBX site object to enable the driver to connect to the PBX." A red asterisk indicates required fields. The "Access Type" is set to "Emulate" and "Are jacks hot?" is set to "Yes". A section titled "General Information" contains the following fields:

PBX name: *	pbx name
Login name: *	login name
Password: *	*****
Extension length: *	5
Beginning number of DID extension range: *	12345
Ending number of DID extension range: *	23456
Beginning number of non-DID extension range:	23457
Ending number of non-DID extension range:	3000

At the bottom, there are buttons for "OK", "Cancel", and "Apply".

nwoWorkOrder

 DirXML-nwoWorkOrder represents work orders.

This object lets you indicate what actions you want the driver to perform in the PBX, and specify other details such as when the work order should be performed, the object ID and display name of the person who owns the extension, and whether a specific extension number is requested. After the driver performs the work order, it updates the work order with information such as the status (Configured, Error, etc.).

An iManager plug-in is provided to help you create and maintain work orders. In iManager, select *PBX Utilities > Work Order Management*. The following figure shows an example of the interface for creating a work order.

Figure 1-3 *The Work Order Page*

Novell iManager - Microsoft Internet Explorer

Work Order: status pending

PBX

PBX Work Order

The work order object is used to tell the driver what tasks to perform in the PBX and to allow the driver to record the results.

* Required

Action: Install

PBX name: *

Description:

Extension:

Extension type: <Select an extension type>

Do it now: false

Due date: * 5/3/2004

Node:

Phone type:

Status: Pending

Work order number:

Send to publisher: false

Duplicate extension:

Room:

Display name:

Object ID:

Port:


OK Cancel Apply

pbxExtension

 DirXML-pbxExtension represents extensions.

After performing a work order, the driver shim sends one of these objects to represent the work that was done. For example, if a new extension was configured, a new extension object is sent with the correct extension number and display name. The driver updates the work order with information such as the status (Actions taken if successful, Configured, Error, etc.) If you create a new AudixSubscriber object, a new DirXML-pbxAudixSubscriber object is sent to the engine.

pbxAudixSubscriber

 DirXML-pbxAudixSubscriber represents Audix support.

With Audix support for the Avaya driver, the driver can add voice mail to an extension, remove voice mail from an extension, or modify the attributes of the voice mail of an extension. However, this driver support does not replace general administration of the Avaya Audix system.

Adding Audix support to the current Avaya driver requires the driver to access and manipulate PBX subscriber objects. This is performed by extending the eDirectory schema to include a DirXML-pbxAudixSubscriber object and by adding a new site object that defines the Audix server.

1.4.2 Overview of Driver Functionality

When configuring the driver, you have the option to run the driver in emulation mode. Although this mode of driver operation is optional, we highly recommend its use. Emulation mode enables you to fully test the driver and its policies before connecting to a live PBX system. When you run in emulation mode, the driver emulates the PBX specified by the pbxSite object. All driver actions are directed to the LDAP site that you specify in the Configuration Parameters. To run in emulation mode, locate the pbxSite object and set the DirXML-AccessType attribute to “emulate.” This causes the driver to emulate the PBX. For more information, see [“What Is Emulation Mode and Why Should I Use it?” on page 21.](#)

The driver shim itself functions in the following basic way:

1. When the driver starts, it reads the information for the DirXML-pbxSite objects from the container you specified in the driver parameters.

These objects tell the driver which PBXs to perform work orders on, and they provide other details such as which number ranges can be used for extensions.
2. At the polling time or interval you set, the driver shim “wakes up,” and the Publisher queries the PBX for all the extensions. If it detects that a change has been made manually since the last polling interval, it sends a DirXML-pbxExtension object to the Identity Vault representing the change.

See [“Support for Manual Changes in the PBX” on page 21.](#)

3. The Publisher then polls for DirXML-nwoWorkOrder objects in eDirectory in the container you specified in the driver parameters. It checks the DirXML-nwoDueDate and DirXML-nwoStatus attributes on the work orders to see which of them need to be performed.
4. The Publisher performs the work orders that are due, completing the appropriate action based on attributes of the DirXML-nwoWorkOrder objects. If a value was present in the work order for the DirXML-nwoObjectID attribute, the Object ID is placed in the Cable field for that extension in the PBX.
5. The Publisher updates the DirXML-nwoWorkOrder with the results.

For example, if a work order to install a new extension is successful, the DirXML-nwoStatus attribute is updated with the value “Configured.” If no extension number was requested in the work order, or if the requested extension was not available, the Publisher specifies which extension was chosen.
6. For Installation work orders, the Publisher also sends a DirXML-pbxExtension object to the Identity Vault for each work order, representing the results.

For example, if a work order to install a new extension is successful, a DirXML-pbxExtension object is sent to the Identity Vault with the new extension in the DirXML-nwoExtension attribute.
7. For Add Audix Subscriber work orders, the Publisher also sends a new DirXML-pbxAudixSubscriber object to the Identity Vault.

The driver can also perform a work order immediately instead of waiting for the next polling interval, if you indicate Do It Now in the work order. Work orders with this attribute are immediately processed by the Publisher channel.

You can customize the driver to enhance what it does. The sample configurations demonstrate some of these customizations.

For example, you can use driver policies to do the following:

- ♦ Create a work order when a new user object is added to the Identity Vault in order to automate an extension assignment to a new employee. This could be further automated by using an additional Identity Manager driver to automatically create user objects in the Identity Vault when they are created in a human resources application.
- ♦ Assign certain phone restrictions based on the location or job title indicated for the user object in the Identity Vault. For example, you could configure the driver to use non-DID extensions for employees in the call center.
- ♦ Using an additional Identity Manager driver, cause work order objects from another application to be synchronized to the Identity Vault and performed by the Avaya PBX driver.
- ♦ Transform an add DirXML-pbxExtension object coming from the driver into an add and a modification of a User object, so you can update the User object with the new extension when a work order completes.
- ♦ Add other customizations to fit your business processes.

To demonstrate some of the things you can do with the Avaya PBX driver, two sample driver configurations are provided. A third kind of configuration is also described in this guide, but the sample scripting provided is not a complete driver configuration.

These three kinds of configurations are discussed in the following chapters:

- ♦ [Chapter 6, “Base Configuration,” on page 43](#)
- ♦ [Chapter 7, “Workforce Tree Configuration,” on page 49](#)
- ♦ [Chapter 8, “Work Order Database Configuration,” on page 57](#)

1.4.3 Assumptions about the PBX

The PBX is the Public Branch Avaya PBX or corporate phone system. The PBX can be centralized or distributed across buildings or sites. A PBX system can be composed of several PBX cabinets, including remote cabinets, controlled by a master PBX cabinet. Typically, each phone is physically cross-connected to a PBX digital port.

Because the driver is designed to be generic, the PBX can be used in any environment, from an enterprise call center system to a small office key system.

The Avaya PBX driver relies on the following assumptions:

1. Responsibility for PBX administration lies with the user rather than a third party.
 2. The PBX administrator has sufficient training and certification, and appropriate access rights to the PBX.
 3. The PBX supports administration by telnet.
 4. Voice jacks are either “hot jacks” or “cold jacks.” The driver supports both kinds of environments. See [“Voice Jacks” on page 19](#).
- ♦ [“Voice Jacks” on page 19](#)
 - ♦ [“Querying the PBX” on page 19](#)

Voice Jacks

The behavior of the driver depends on whether or not the PBX environment has hot jacks (jacks that are permanently cross-connected to live PBX ports) or cold jacks (jacks that are cross-connected at the time of phone installation).

- ♦ [“Hot Jacks” on page 19](#)
- ♦ [“Cold Jacks” on page 19](#)

Hot Jacks

Hot jacks are simpler from a configuration standpoint, if the PBX supports port selection from the phone set. If the PBX does not support port selection from the phone set, the cold jack process is followed. (See [“Cold Jacks” on page 19](#).) The driver assigns the port specified in the work order. If no port is specified, the driver will “X out” the port.

When a work order requests a new phone, the driver creates the extension in the PBX, but leaves the port unassigned. If the work order did not request a specific extension number, the driver assigns the first available number in numeric order. When the phone is delivered and plugged in, punching in a code activates the extension on that phone by automatically assigning the port (if the port was “Xed out.”)

When moving an existing extension, the driver simply unassigns the port, meaning it is Xed out. This deactivates the extension. When the phone is delivered and plugged in at the new location, punching in the activation code assigns the new port to the extension.

Cold Jacks

If the PBX environment has cold jacks, the installation process requires you to specify which node the user is in as part of the work order. If the extension is not given, the next available number is assigned. The driver also scans for available ports that work with the phone type and are in the correct node. The new port is assigned to the extension and is noted in the work order.

When the phone is delivered, a technician must cross-connect the port to the new jack.

If you have cold jacks, but you prefer that the driver X out the port instead of automatically choosing one, you can specify the hot jacks option. When the phone is physically installed, the driver sees the change after its next polling cycle and changes the status of the work order.

Querying the PBX

Because of the limitations of the PBX system, the driver queries the PBX only for DirXML-pbxExtension and DirXML-pbxAudixSubscriber extensions.

The PBX doesn’t support queries on the display name or the field containing a user identifier. To find those pieces of data when querying the PBX, you need to create a custom policy configuration that queries the PBX for all the extensions and their associated data, and then searches through the entire list to find the value you want.

1.4.4 What the Driver Does in the PBX

At the most basic level, the driver can perform several actions in the PBX: install, disable, enable, move, modify, setcor, and disconnect. The following list describes the main tasks you can perform using work orders:

- ♦ **Install:** Assign an extension in the PBX at a given location, and activate the extension.

You can request a specific extension number, and the driver assigns that number if it's available in the PBX. If it's not available, or if no extension is specified in the work order, the driver assigns the next available extension number in numeric order within the range of extensions specified in the PBX site object.
- ♦ **Disable:** Stop allowing an extension to be used, but leave it installed in case you want to enable it again in the future.
- ♦ **Enable:** Allow an extension to be used again after it has been disabled.
- ♦ **Move:** Move an extension in one of the following ways:
 - ♦ Deactivate it in one location and activate it in another location within the same PBX system.
 - ♦ With the use of custom policies, deactivate it in one PBX system and activate it in another PBX system.
- ♦ **Setcor:** Use the `setcor` command to change restrictions for the extension, such as whether long distance or international calls are allowed.
- ♦ **Disconnect:** Remove an extension from the PBX.
- ♦ Set values such as the following, based on data you provide in the work order:
 - ♦ The date to complete the desired action.
If the `DoItNow` flag is set, the driver performs the action immediately.
The driver is configured so that at a specified time once a day it performs work orders that are due that day. You can also set a polling interval for performing work orders.
 - ♦ The type of phone (DID or non-DID).
 - ♦ Restrictions for use of the phone extension, such as whether long distance or international calls are allowed.
 - ♦ The port number for the phone connection (necessary if you use cold jacks).
- ♦ Query the PBX to find out what manual changes have been made since the last time the driver polled for work orders. This feature supports manual changes to the PBX extensions. This feature is available in Avaya PBX but not in Audix.

At the specified time of day or polling interval, the Publisher wakes up to perform work orders. Before polling for and performing work orders, the Publisher first queries the PBX to find out whether anything has changed since the last time the Publisher performed work orders. This means that you can make changes manually in the PBX, and the driver is made aware of those changes when it next communicates with the PBX.

This feature makes the driver implementation more flexible, because you are not limited to making changes only through the driver. In addition, if you make sure the object ID is inserted the label field in the PBX whenever you make a manual change, you can use this feature to automate updates to user objects to reflect changes in the PBX made manually, as well as

changes made by the driver. For example, in the workforce tree sample configuration, if you install an extension and insert the object ID into the label field, the driver discovers that change and automatically updates the user object with the new extension.

See [“Support for Manual Changes in the PBX” on page 21](#).

1.4.5 What Is Emulation Mode and Why Should I Use it?

For all new Identity Manager products, we recommend that you configure and use them first in a test environment. When you are comfortable with the test environment, you can move into production. Most companies, however, don't have PBX systems dedicated to testing. To aid you in testing and debugging, the driver has a built-in emulation mode. In emulation mode, you configure the driver as if you were connecting to your PBX system. Instead of connecting to the PBX, the driver is able to add, modify, and delete extension and Audix Subscriber objects in an LDAP container, which is simulating the PBX. You can fully test policies without affecting the live PBX. It enables you to make PBX changes more easily because you can simulate and watch work order processing before moving to a production environment.

When the driver loads and detects the emulation settings, it uses the emulation parameters set during configuration. The driver binds to the LDAP directory and looks for a PBX site container that holds extensions. If this container does not exist, the driver creates a container with the same name as the PBX name in the site object. During emulation, this is the container the driver looks for when making changes to extension objects. All read/write activity occurs in the PBX site container, and the driver treats the container like it is a PBX system. If a work order is created to install a new extension, you should look in this container to verify that a DirXML-Extension object was created.

1.4.6 Support for Manual Changes in the PBX

The preferred method of performing PBX tasks is to create a work order and let the driver configure the PBX. However, manual changes made in the PBX are also supported.

When the driver prepares to perform work orders, at the polling interval or time of day you specify, it first queries the PBX for the extensions to see if anything has changed since the last time the driver communicated with the PBX. This functionality allows the driver to update the Identity Vault to reflect changes that have been made manually, as much as possible.

The following table indicates the changes the driver can detect in the PBX if the changes were made manually, and what the driver can do in response to them.

Table 1-1 *Changes the Driver Can Detect in the PBX*

Change made manually in the PBX	Can the driver detect the change?	Can the driver update the Identity Vault in response to the change?	How the driver can update the Identity Vault
Install a new extension	Yes	Yes, if the ObjectID is entered correctly in the PBX.	The driver creates a new nwoExtension object.
Modify the display name	Yes	Yes, if the ObjectID is entered correctly in the PBX.	The driver modifies the DisplayName attribute of the nwoExtension object. (In the workforce tree configuration, no change would be necessary.)

Change made manually in the PBX	Can the driver detect the change?	Can the driver update the Identity Vault in response to the change?	How the driver can update the Identity Vault
Modify the ObjectID	Yes	Yes, if the ObjectID is entered correctly in the PBX.	The driver can update the nwoExtension object with a new ObjectID.
Setcor	No	No update to Identity Vault is necessary.	
Disable an extension	Yes	Yes	The driver updates the nwoExtension object in the Identity Vault.
Disconnect an extension	Yes	No	<p>In the base configuration, the driver can delete the nwoExtension object that represented the extension.</p> <p>You can write a policy to update all users who have this extension in their phone lists, by removing the extension from the list. Instead of relying on the ObjectID in this case, the driver determines which users to update by querying Identity Vault for User objects that have that extension.</p>

1.5 Support for Standard Driver Features

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

- ♦ [Section 1.5.1, “Local Platforms,” on page 22](#)
- ♦ [Section 1.5.2, “Remote Platforms,” on page 22](#)
- ♦ [Section 1.5.3, “Entitlements,” on page 23](#)

1.5.1 Local Platforms

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

For information about the operating systems supported for the Metadirectory server, see “[System Requirements](#)” in the *Identity Manager 4.0 Integrated Installation Guide*.

1.5.2 Remote Platforms

The Avaya PBX driver can use the Remote Loader service to run on a server other than the Metadirectory server.

For information about the operating systems supported for the Remote Loader, see “[System Requirements](#)” in the *Identity Manager 4.0 Integrated Installation Guides*.

1.5.3 Entitlements

The Avaya driver does not come with preconfigured entitlements. However, the driver supports entitlements. For information about entitlements, see the [Identity Manager 4.0 Entitlements Guide](#).

Planning issues can vary significantly depending on your environment and goals; this planning section provides a starting point for developing your custom implementation.

- ♦ [Section 2.1, “Supported Avaya PBX Systems,” on page 25](#)
- ♦ [Section 2.2, “Planning Issues for All Configurations,” on page 25](#)
- ♦ [Section 2.3, “Planning for User Provisioning \(Workforce Tree\) Implementations,” on page 27](#)
- ♦ [Section 2.4, “Planning Driver and Replica Placement on Your Servers,” on page 29](#)

2.1 Supported Avaya PBX Systems

The Avaya PBX driver works with Avaya PBX software versions 9, 10, 11, 12, 13, 14 and 15.

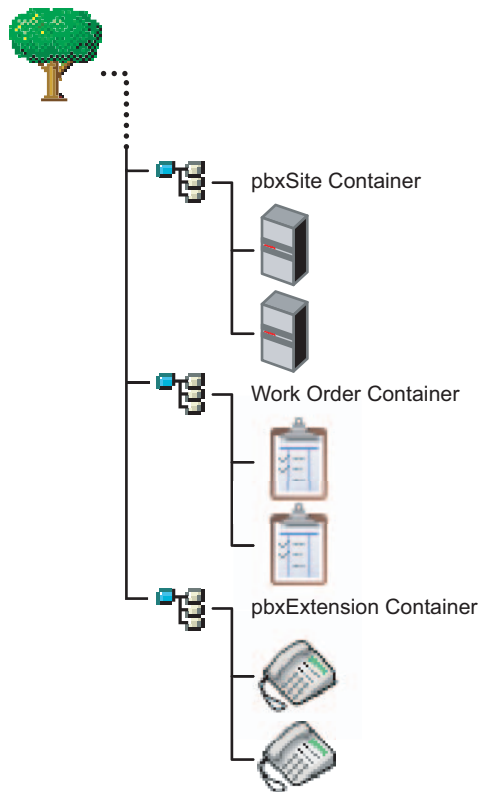
NOTE: Avaya PBX 15 is supported with Communication Manager 5.0.x having the terminal type W2KTT.

2.2 Planning Issues for All Configurations

The items in this section should be relevant regardless of how you want to implement the driver.

- ♦ Identify or create containers to hold the new kinds of objects used by the driver, as shown in the following figure and described in the list of questions after the figure.

Figure 2-1 Example of Containers for New Objects



- ◆ Which container do you want to use for pbxSite objects?

You must create a pbxSite object to represent each of your PBXs. The driver queries for these objects at startup so it knows how to communicate with each PBX. For more information, see [Section B.1, “pbxSite Object,” on page 75](#).

- ◆ Which container do you want to use for nwoWorkOrder objects?

These are the objects used to tell the driver which tasks to perform in the PBX. For more information, see [Section B.2, “DirXML-nwoWorkOrder Object,” on page 78](#).

- ◆ Are you going to use nwoExtension objects, and if so, which container do you want to use for them?

In the base configuration, nwoExtension objects are used to represent extensions, so you can see the results of the tasks the driver performs in the PBX. For more information, see [Section B.3, “DirXML-pbxExtension Object,” on page 83](#).

Depending on your implementation, you might not need to use nwoExtension objects. Instead, you can configure your policies to transform events for nwoExtension objects from the driver into events that update phone information for User objects, as demonstrated in the Workforce Tree configuration (see [Chapter 7, “Workforce Tree Configuration,” on page 49](#)).

- ◆ Gather all the information about your PBX sites that you need to enter into each pbxSite object.

For example, you need to know the answers to questions like the following:

- ◆ How many digits are the extensions?
- ◆ How many nodes do you have?

For a list of all the requirements you need to meet and the information you need to provide in the pbxSite objects, see [Section B.1, “pbxSite Object,” on page 75](#).

- ♦ Do you have hot jacks, or cold jacks?

This aspect of your environment affects what information you must supply when creating a work order. The driver needs more information to perform tasks for cold jacks than it does for hot jacks, so for a cold jack environment more attributes are required on a nwoWorkOrder object. See the list of attributes required for hot jacks and cold jacks in [Section B.2, “DirXML-nwoWorkOrder Object,” on page 78](#).

If you have cold jacks, but you prefer that the driver “X out” a port instead of automatically choosing one, you can specify the hot jacks option.

- ♦ What are the duplicate extensions for each phone type?

You need to reproduce the mapping in the policies when a work order asks the driver to install an extension. The driver refers to the duplicate extension policy on the PBX to create the new extension.

Duplicate (dupe) extensions are set up by the PBX administrator in the PBX, using extensions that are outside the range of extensions that are available for use as DID and non-DID phone numbers. They are used when installing an extension. These dupe extensions are used only as a template to set up each phone type (such as phone type 8410) correctly.

In each work order the duplicate extension for the phone type needs to be specified, so the PBX has the right template to follow. The policies for the sample configurations demonstrate how to map the phone type to the duplicate extension.

- ♦ When do you want the driver to perform work orders?

You can control the timing by using polling interval, time of day, or both.

Of course, if the work order is marked DoItNow, the driver performs it immediately and doesn’t wait for a polling interval or time of day.

- ♦ How will you create a test environment for testing the driver with the PBX?

Creating a robust test environment can be a challenge, because often the only PBX available in an environment is the production PBX. To solve this problem, we recommend using the driver’s emulation mode. See [Section 1.4.5, “What Is Emulation Mode and Why Should I Use it?,” on page 21](#) for more information.

2.3 Planning for User Provisioning (Workforce Tree) Implementations

This section explains the additional planning issues you need to consider when creating an implementation of the driver that is like the sample workforce tree configuration provided. (See [Chapter 7, “Workforce Tree Configuration,” on page 49](#).)

- ♦ What ID will you use to identify the user for whom the work order is being performed?

This ID number is entered in the work order as the ObjectID attribute, and you can use a custom policy to cause Identity Manager to match the work order with the corresponding user.

The sample workforce tree configuration provided uses the eDirectory Workforce ID for the User object, but you could use the employee ID, or some other number that is used in your organization. This number must be stored as an attribute of the user object so the driver can find it.

Because this number is stored in the PBX in the *Cable* field, it can be only 5 digits long.

Keep in mind that this ID must also be entered in the PBX for any extensions entered manually.

- ♦ How will you specify the user's location so that the policies can determine where in the PBX to configure the extensions port?

When you are planning how to determine which PBX to configure the user's extension in, it's important to understand how the PBX sites are configured. A PBX cabinet can be configured as a remote cabinet, meaning that it is controlled by a master PBX but is at a different office or even in a different city. This can mean that a user location in Human Resources might not have a simple mapping to a PBX site and location within the PBX system.

- ♦ How will you determine whether a user should receive a DID or a non-DID extension?

You need to determine what business rules to follow. For example, you could customize the policies to assign non-DID extensions to users who work in a call center, based on job title.

It is not necessary to have both a DID and a non-DID range. If you only need one range, use the DID range. You can use two ranges if desired.

- ♦ How will you determine the correct phone type for a user?

The phone type must be automatically assigned by the policies, based on business rules about user information such as job title. For example, you could specify that users with the job title of Administrative Assistant receive the phone type 83424.

- ♦ Do you want to set phone use restrictions on extensions for certain users?

You can set restrictions on time of day usage, long distance calls, and international calls based on business rules and attributes of the user. For example, you can use criteria such as job title or the user object's placement in the tree.

- ♦ What user events do you want to automate with the driver, and what do you want the automated response to be?

The sample configuration demonstrates automated responses for certain user events, such as updating the display name in the PBX when the name of a user changes.

- ♦ Do you want the driver to manage all existing users, or just provision new users? If you want the driver to manage existing users, some manual setup is necessary.

The Avaya PBX driver can manage new users. It can also manage existing users.

See [Section 9.3, "Managing Existing Users," on page 64](#).

- ♦ What do you want to use as the display name for an extension?

Determine what information from a user object you want the policies to use when creating the display name.

- ♦ How do you want to handle extensions that are not assigned to a particular user, such as conference rooms or lab extensions?

The driver does not require entities that have phone numbers to be represented in eDirectory. The driver can perform work orders in the PBX regardless of whether there is an object in eDirectory that should receive updates to phone information. But if you want to use the driver to update phone information in eDirectory for entities other than a user, one option is to create eDirectory objects for those rooms or other entities, so that they can be identified by an ObjectID. Then you can customize the driver to update them the same way it can update user objects.

- ♦ What should you consider when implementing the driver with a work order database?

Keep in mind that you could use more than one container for work orders, such as one for work orders created in eDirectory manually, and one for work orders that come from the work order database.

2.4 Planning Driver and Replica Placement on Your Servers

For each server where you install Identity Manager and run the driver, you need to have a master or read/write replica of all the users you want to manage.

Installing the Driver Files

3

The Avaya PBX driver must be located on a server that can access the Avaya PBX via telnet. This can be either a Metadirectory server (Metadirectory engine, Identity Vault, and Avaya PBX driver) or a non-Metadirectory server (Remote Loader service and Avaya PBX driver. If the driver is not already installed on the server you want to use the integrated installed to install the driver files. For more information, see the [Identity Manager 4.0 Integrated Installation Guide](#).

Creating a New Driver

4

After the Avaya PBX driver files are installed on the server where you want to run the driver (see [Chapter 3, “Installing the Driver Files,” on page 31](#)), you can create the driver in the Identity Vault. You do so by importing the basic driver configuration file and then modifying the driver configuration to suit your environment.

The following sections provide instructions:

- ♦ [Section 4.1, “Prerequisites,” on page 33](#)
- ♦ [Section 4.2, “Creating the Driver in Designer,” on page 34](#)
- ♦ [Section 4.3, “Creating the Driver in iManager,” on page 36](#)
- ♦ [Section 4.4, “Activating the Driver,” on page 40](#)

4.1 Prerequisites

Before you create a driver, you need to satisfy the following prerequisites:

- ♦ [Section 4.1.1, “Selecting a Configuration File,” on page 33](#)
- ♦ [Section 4.1.2, “Creating Containers,” on page 33](#)

4.1.1 Selecting a Configuration File

There are two basic configuration files: Avaya PBX (`AvayaPBXShip-IDM3_6_0-V2.xml`) and Avaya User (`AvayaUser-IDM3_6_0-V2.xml`). The file you use to create the driver depends on the type of configuration you need to implement.

The Avaya PBX configuration file provides the base configuration described in [Chapter 6, “Base Configuration,” on page 43](#).

The Avaya User configuration file contains additional policies that illustrate the workforce tree and workorder configurations described in [Chapter 7, “Workforce Tree Configuration,” on page 49](#) and [Chapter 8, “Work Order Database Configuration,” on page 57](#).

4.1.2 Creating Containers

You need to specify or create some containers when importing the driver configuration:

- ♦ Container for holding the `DirXML-pbxSite` objects
- ♦ Container for holding the `DirXML-nwoWorkOrder` objects
- ♦ Container for holding the `DirXML-pbxExtension` objects, if you are using them
The basic configuration uses these objects, but in a production environment you would usually transform events for `DirXML-pbxExtension` objects into events for User objects.
- ♦ Container for holding the `DirXML-pbxAudixSubscriber` objects, if you are using them

You should restrict rights to these containers so that only authorized administrators can change these containers or the objects they hold.

These containers and objects are described in [Section 2.2, “Planning Issues for All Configurations,” on page 25](#) and [Appendix B, “Schema for PBX Management,” on page 75](#).

4.2 Creating the Driver in Designer

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

- ♦ [Section 4.2.1, “Importing the Driver Configuration File,” on page 34](#)
- ♦ [Section 4.2.2, “Configuring the Driver,” on page 35](#)
- ♦ [Section 4.2.3, “Using Emulation to Test the Driver,” on page 35](#)
- ♦ [Section 4.2.4, “Deploying the Driver,” on page 35](#)
- ♦ [Section 4.2.5, “Starting the Driver,” on page 36](#)

4.2.1 Importing the Driver Configuration File

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

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

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

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

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

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

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

- 6 Click *Next* to import the driver configuration.

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

- 7 To review or modify the default configuration settings, click *Configure*, then continue with the next section, [Configuring the Driver](#).

or

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

4.2.2 Configuring the Driver

There are many settings that can help you customize and optimize the driver. The settings are divided into categories such as Driver Configuration, Engine Control Values, and Global Configuration Values (GCVs). Although it is important for you to understand all of the settings, your first priority should be to review the [Driver Parameters](#) located on the Driver Configuration page and the [Global Configuration Values](#). These settings must be configured properly for the driver to start and function correctly.

If you do not have the Driver Properties page displayed in Designer:

- 1 Open your project.
- 2 In the Modeler, right-click the driver icon  or the driver line, then select *Properties*.

In addition to the driver settings, you need to modify the default policies provided by the basic driver configuration. Review the information in the following sections to help with the policies:

- ♦ [Chapter 6, “Base Configuration,” on page 43](#)
- ♦ [Chapter 7, “Workforce Tree Configuration,” on page 49](#)
- ♦ [Chapter 8, “Work Order Database Configuration,” on page 57](#)

4.2.3 Using Emulation to Test the Driver

Creating a robust test environment can be a challenge because the only PBX available in an environment might be the production PBX. To solve this problem, the Avaya PBX driver provides emulation capabilities. For an explanation of these capabilities, see [“What Is Emulation Mode and Why Should I Use it?” on page 21](#).


To set up emulation:

- 1 Configure the Emulation options on the driver parameters. For instructions, see [Section A.1.5, “Driver Parameters,” on page 72](#).
- 2 Modify the PBX Site object to change the Access Type to *Emulate* or *AudixEmulate*. You must do this in iManager:
 - 2a In iManager, click the *View Objects* icon to display the Identity Vault tree.
 - 2b Open the PBX Site container.
 - 2c Open the PBX Site object.
 - 2d In the *Access Type* list, select *Emulate* or *AudixEmulate*.
 - 2e Click *OK* to save your changes.

After you configure the emulation settings, you must deploy the driver to the Identity Vault before you can test it. Continue with the next section, [Deploying the Driver](#).

4.2.4 Deploying the Driver

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

- 1 In Designer, open your project.
- 2 In the Modeler, right-click the driver icon  or the driver line, then select *Live > Deploy*.

3 If you are authenticated to the Identity Vault, skip to [Step 5](#); otherwise, specify the following information:

- ♦ **Host:** Specify the IP address or DNS name of the server hosting the Identity Vault.
- ♦ **Username:** Specify the DN of the user object used to authenticate to the Identity Vault.
- ♦ **Password:** Specify the user's password.

4 Click *OK*.

5 Read the deployment summary, then click *Deploy*.

6 Read the message, then click *OK*.

7 Click *Define Security Equivalence* to assign rights to the driver.

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

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

7b Click *OK* twice.

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

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

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

8b Click *OK*.

8c Repeat [Step 8a](#) and [Step 8b](#) for each object you want to exclude.

8d Click *OK*.

9 Click *OK*.

4.2.5 Starting the Driver

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

To start the driver:

1 In Designer, open your project.

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

For information about management tasks with the driver, see [Chapter 9, "Managing the Driver,"](#) on [page 63](#).


4.3 Creating the Driver in iManager

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

- ♦ [Section 4.3.1, "Importing the Driver Configuration File,"](#) on [page 37](#)

- ♦ [Section 4.3.2, “Configuring the Driver,” on page 39](#)
- ♦ [Section 4.3.3, “Using Emulation to Test the Driver,” on page 39](#)
- ♦ [Section 4.3.4, “Starting the Driver,” on page 40](#)

4.3.1 Importing the Driver Configuration File

- 1 In iManager, click  to display the Identity Manager Administration page.
- 2 In the Administration list, click *Import Configuration* to launch the Import Configuration wizard.
- 3 Follow the wizard prompts, filling in the requested information (described below) until you reach the Summary page.

Prompt	Description
Where do you want to place the new driver?	You can add the driver to an existing driver set, or you can create a new driver set and add the driver to the new set. If you choose to create a new driver set, you are prompted to specify the name, context, and server for the driver set.
Import a configuration into this driver set	Use the default option, <i>Import a configuration from the server (.XML file)</i> . In the <i>Show</i> field, select <i>Identity Manager 3.6.1 configurations</i> . In the <i>Configurations</i> field, select the AvayaPBXShip file.
Driver name	Type a name for the driver. The name must be unique within the driver set.
Avaya PBX version	Select the version of the Avaya PBX system from the drop-down list.
Driver is Local/Remote	Select <i>Local</i> if this driver will run on the Metadirectory server without using the Remote Loader service. Select <i>Remote</i> if you want the driver to use the Remote Loader service, either locally on the Metadirectory server or remotely on another server.
Remote Host Name and Port	This applies only if the driver is running remotely. Specify the host name or IP address of the server where the driver's Remote Loader service is running.
Driver Password	This applies only if the driver is running remotely. Specify the driver object password that is defined in the Remote Loader service. The Remote Loader requires this password to authenticate to the Metadirectory server.
Remote Password	This applies only if the driver is running remotely. Specify the Remote Loader's password (as defined on the Remote Loader service). The Metadirectory engine (or Remote Loader shim) requires this password to authenticate to the Remote Loader

Prompt	Description
Define Security Equivalences	The driver requires rights to objects within the Identity Vault. The Admin user object is most often used to supply these rights. However, you might want to create a DriversUser (for example) and assign security equivalence to that user. Whatever rights that the driver needs to have on the server, the DriversUser object must have the same security rights.
Exclude Administrative Roles	You should exclude any administrative User objects (for example, Admin and DriversUser) from synchronization.

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

Import Configuration

Summary - Current Configuration

Warning: Drivers May Require Configuration

Drivers imported from a configuration file may require additional configuration settings to be fully functional. Select the driver's link to edit its configuration settings.

The following summarizes the state of the driver as it currently exists.

fabio19 (NCP Server)

DS (Driver Set)

Avaya PBX (Drivers May Require Configuration) (Driver)

mapping rule (Schema Mapping Policy)

none (Input Transformation Policy)

none (Output Transformation Policy)

Publisher (Publisher)

none (Command Transformation Policy)

none (Event Transformation Policy)

none (Matching Policy)

none (Creation Policy)

pub-pp (Placement Policy)

Subscriber (Subscriber)

none (Command Transformation Policy)

none (Event Transformation Policy)

none (Matching Policy)

sub-cp (Creation Policy)

sub-pp (Placement Policy)

<< Back

Next >>

Cancel

Finish

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

- To modify the default configuration settings, click the linked driver name, then continue with the next section, [Configuring the Driver](#).


or

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

4.3.2 Configuring the Driver

There are many settings that can help you customize and optimize the driver. The settings are divided into categories such as Driver Configuration, Engine Control Values, and Global Configuration Values (GCVs). Although it is important for you to understand all of the settings, your first priority should be to review the [Driver Parameters](#) located on the Driver Configuration page and the [Global Configuration Values](#). These settings must be configured properly for the driver to start and function correctly.

To configure the settings:

- 1 Make sure the Modify Object page for the Identity Manager driver for Avaya PBX is displayed in iManager. If it is not:
 - 1a In iManager, click  to display the Identity Manager Administration page.
 - 1b Click *Identity Manager Overview*.
 - 1c Browse to and select the driver set object that contains the new driver.
 - 1d Click the driver set name to access the Driver Set Overview page.
 - 1e Click the upper right corner of the driver, then click *Edit properties*.
- 2 Review the settings on the various pages and modify them as needed for your environment. The configuration settings are explained in [Appendix A, “Driver Properties,” on page 69](#).
- 3 After modifying the settings, click *OK* to save the settings and close the Modify Object page.
- 4 (Conditional) If the Identity Manager driver for Avaya PBX’s Summary page for the Import Configuration wizard is still displayed, click *Finish*.

WARNING: Do not click *Cancel* on the Summary page. This removes the driver from the Identity Vault and results in the loss of your work.

In addition to the driver settings, you need to modify the default policies and rules provided by the basic driver configuration. Review the information in the following sections to help with the policies:

- ♦ [Chapter 6, “Base Configuration,” on page 43](#)
- ♦ [Chapter 7, “Workforce Tree Configuration,” on page 49](#)
- ♦ [Chapter 8, “Work Order Database Configuration,” on page 57](#)

4.3.3 Using Emulation to Test the Driver

Creating a robust test environment can be a challenge because the only PBX available in an environment might be the production PBX. To solve this problem, the Avaya PBX driver provides emulation capabilities. For an explanation of these capabilities, see [“What Is Emulation Mode and Why Should I Use it?” on page 21](#).


To set up emulation:

- 1 Configure the Emulation options on the driver parameters. For instructions, see [Section A.1.5, “Driver Parameters,” on page 72](#).
- 2 Modify the PBX Site object to change the Access Type to *Emulate* or *AudixEmulate*:
 - 2a In iManager, click the *View Objects* icon to display the Identity Vault tree.
 - 2b Open the PBX Site container.
 - 2c Open the PBX Site object.
 - 2d In the *Access Type* list, select *Emulate* or *AudixEmulate*.
 - 2e Click *OK* to save your changes.

4.3.4 Starting the Driver

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

To start the driver:

- 1 In iManager, click  to display the Identity Manager Administration page.
- 2 Click *Identity Manager Overview*.
- 3 Browse to and select the driver set object that contains the driver you want to start.
- 4 Click the driver set name to access the Driver Set Overview page.
- 5 Click the upper right corner of the driver, then click *Start driver*.

For information about management tasks with the driver, see [Chapter 9, “Managing the Driver,” on page 63](#).

4.4 Activating the Driver

If you create the Avaya PBX driver in a driver set where you've already activated a driver that comes with the Integration Module for PBX, the driver inherits the activation. If you created the Avaya PBX driver in a driver set that has not been activated, you must activate the driver, with the Integration Module for PBX activation, within 90 days. Otherwise, the driver stops working.

For information on activation, refer to “[Activating Novell Identity Manager Products](#)” in the *Identity Manager 4.0 Integrated Installation Guide*.

Upgrading an Existing Driver

5

The following sections provide information to help you upgrade an existing driver to version 4.0:

- ♦ [Section 5.1, “Supported Upgrade Paths,” on page 41](#)
- ♦ [Section 5.2, “What’s New in Version 4.0,” on page 41](#)
- ♦ [Section 5.3, “Upgrade Procedure,” on page 41](#)

5.1 Supported Upgrade Paths

You can upgrade from any 3.x version of the Avaya PBX driver. Upgrading a pre-3.x version of the driver directly to version 4.0 is not supported.

5.2 What’s New in Version 4.0

Version 4.0 of the driver does not include any new features.

5.3 Upgrade Procedure

The process for upgrading the Avaya PBX driver is the same as for other Identity Manager drivers. For detailed instructions, see “[Upgrading Drivers to Packages](#)” in the *Identity Manager 4.0 Framework Installation Guide*.

Base Configuration

6

The base configuration (`AvayaPBXShip-IDM3_6_0-V2.xml`) is a sample configuration to demonstrate the most basic functionality of the driver. It shows how the driver can manage PBX extensions using work order objects in eDirectory.

- ♦ [Section 6.1, “How to Use the Base Configuration,” on page 43](#)
- ♦ [Section 6.2, “How the Base Configuration Works,” on page 43](#)

6.1 How to Use the Base Configuration

This configuration would be appropriate to import and configure in a test environment for instructional purposes, to help you learn how to configure the solution you need. It is not meant to be an out-of-the-box solution.

The rules and policies in the base configuration are set up so that you can create work order objects (using a new object class, `nwoWorkOrder`) in a Work Order container, and the driver will perform the work orders in the PBX and show the results by creating a `pbxExtension` object and updating the `nwoWorkOrder` object in eDirectory.

Most real-life implementations would want User objects to be updated when an extension is assigned or changed, but to keep the configuration sample as simple as possible, the base configuration does not include rules to do this. Instead, `pbxExtension` objects are used to represent extensions, and User objects are not affected by the changes.

To see how user objects can be involved in the process, review the workforce tree configuration, explained in [Chapter 7, “Workforce Tree Configuration,” on page 49](#).

Similarly, the base configuration does not demonstrate a connection between eDirectory and a work order database. In implementations that have an existing work order database, you should connect to it so that the Identity Manager Driver for Avaya PBX can send and receive work order data. This connection could be made using another driver such as the Identity Manager Driver for JDBC*. This kind of implementation is described in [Chapter 8, “Work Order Database Configuration,” on page 57](#).

6.2 How the Base Configuration Works

[Figure 6-1](#) shows what happens in the base configuration if you create an `nwoWorkOrder` object for installing a new extension.

1. You create an `nwoWorkOrder` object in eDirectory by using the Install task. In this example, the flags are set on the work order for `SendToPublisher` and `DoItNow`.
2. The Subscriber channel is notified by the Metadirectory engine that a new `nwoWorkOrder` object has been created in eDirectory.
3. The Subscriber creates the Identity Manager association for the object, and sends the work order to the Publisher.

(If the `SendToPublisher` flag is not set, the Subscriber does not “wake up” Publisher channel to send the work order to the Publisher. Instead, the Publisher reads the work order the next time it polled for work orders.)

4. The Publisher performs the work order immediately, because the DoItNow flag is set.
(If the DoItNow flag is not set, the Publisher doesn't perform the action until the date specified in the DueDate attribute.)
5. After successfully configuring the extension in the PBX, the Publisher writes "configured" in the Status attribute of the work order.
6. The Publisher creates a pbxExtension object in eDirectory to represent the new extension that has been configured.

After the driver completes the work order in the PBX, the PBX admin can complete any manual tasks required, such as plugging in the phone and punching in the activation code for hot jacks, or cross-connecting the wiring, etc., for cold jacks.

The following figures describe the base configuration. [Figure 6-1](#) shows an illustration of how the base configuration generally works. [Figure 6-2](#) is a flowchart of how the configuration works for an Install work order with the DoItNow flag set to true, and [Figure 6-3](#) is a flowchart showing how the configuration works for an Install work order with the DoItNow flag not set.

Figure 6-1 Graphical Representation of Base Configuration

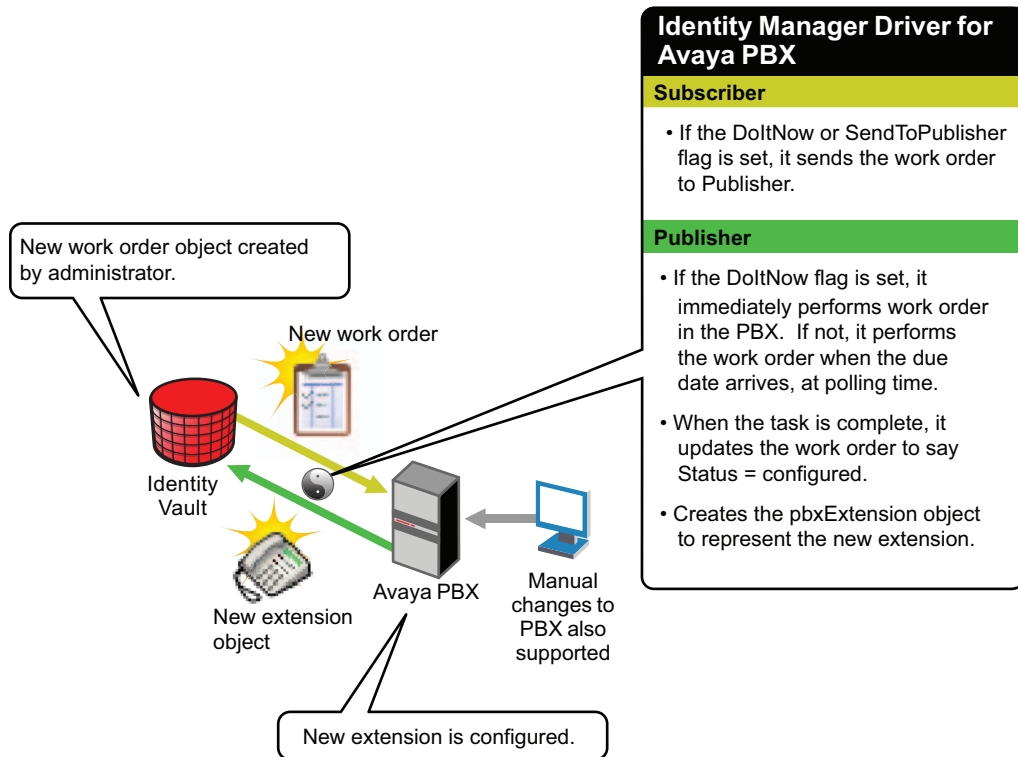


Figure 6-2 Flowchart of Base Configuration for Work Order with DoItNow Flag Set to True

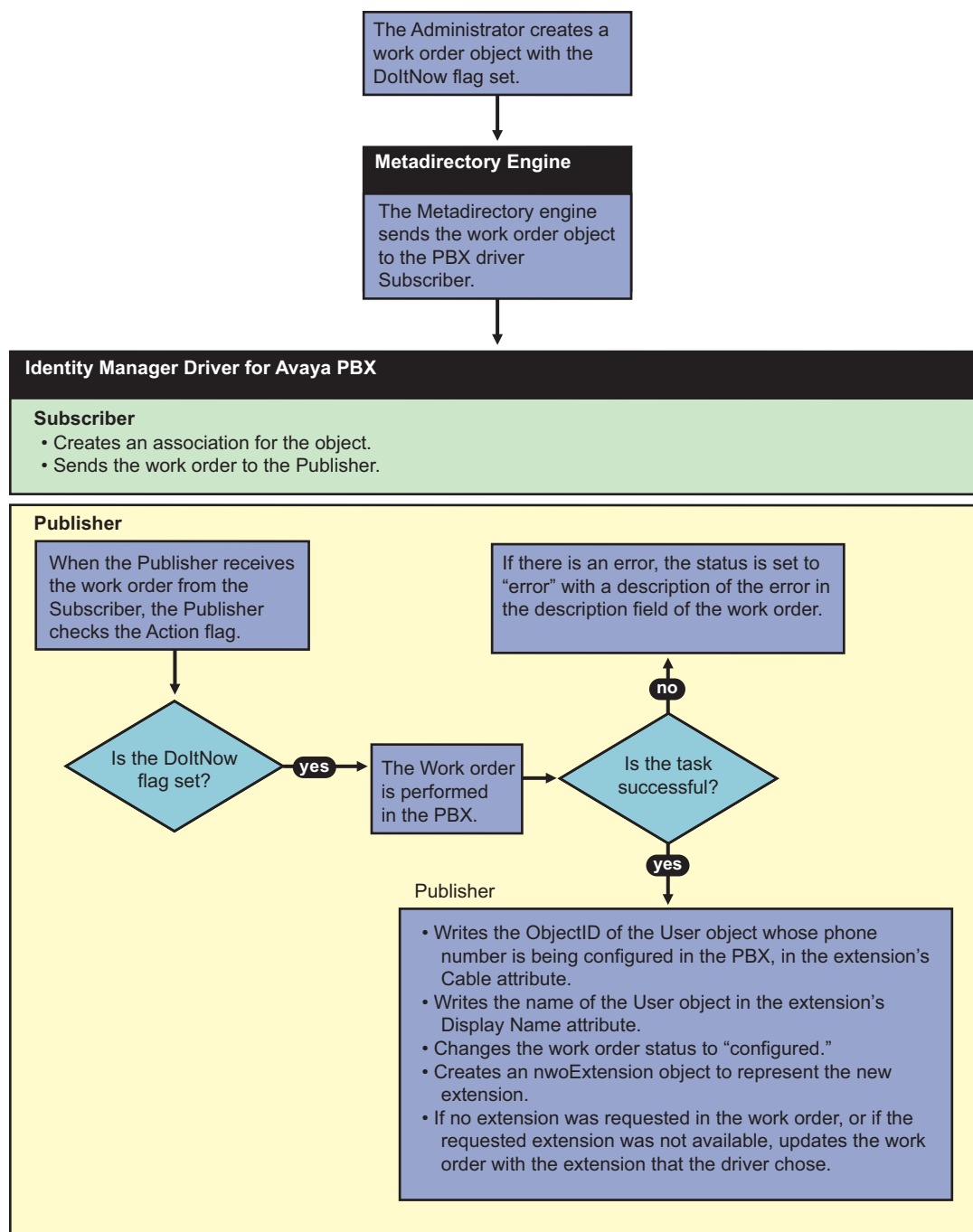
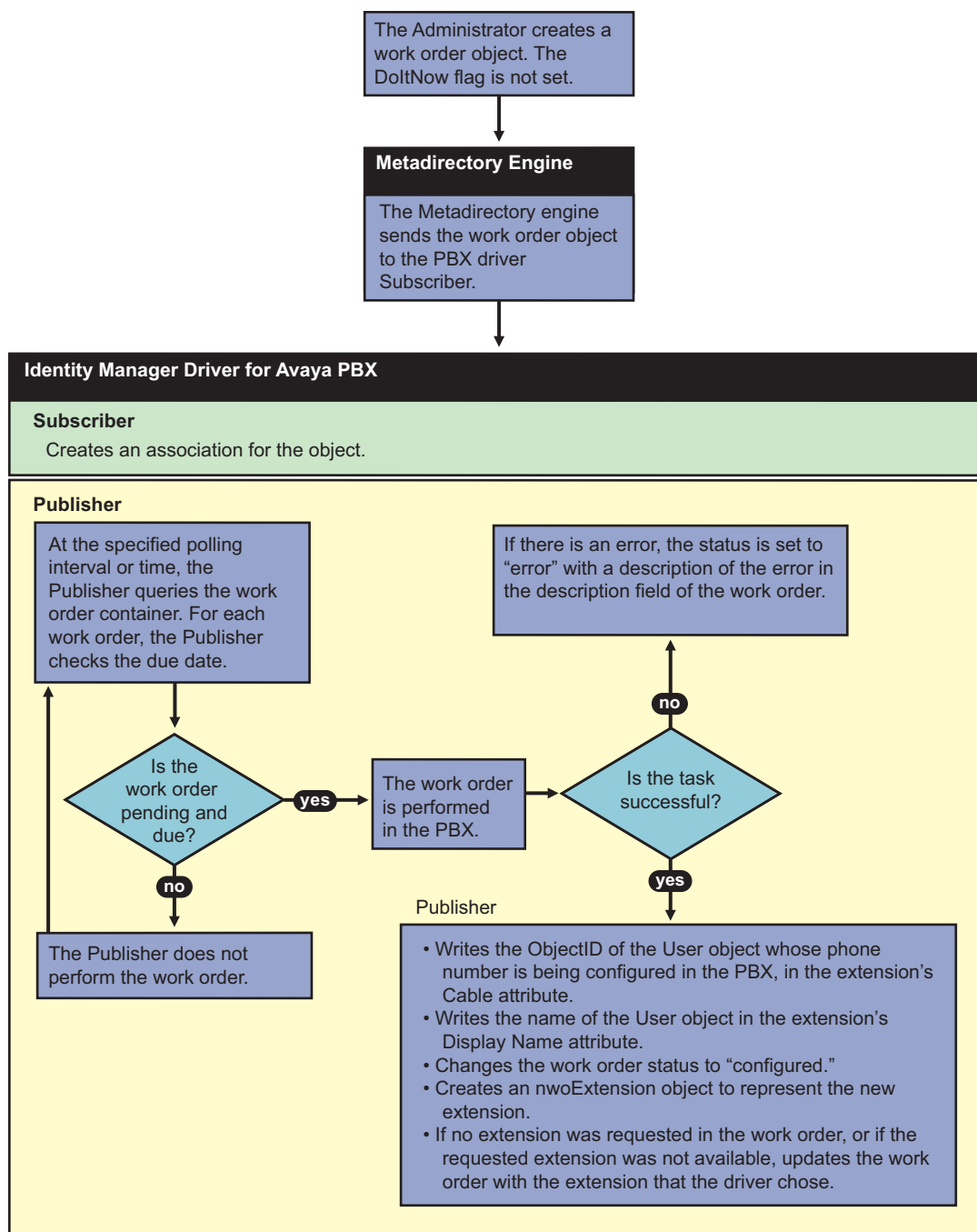


Figure 6-3 Flowchart of Base Configuration for Work Order with DoItNow Flag Not Set



6.2.1 How the Subscriber Channel Is Configured

In the base configuration, the Subscriber channel processes only events that pertain to work orders.

You must create pbxSite objects for each site in Identity Vault, so the driver knows how to contact each PBX, but the driver queries for this information only at startup. The Subscriber does not listen for events pertaining to pbxSite objects, so if changes are made to these objects you must stop and restart the driver for the changes to be recognized by the driver.

For many drivers, the Subscriber performs changes in the third-party application in response to events in Identity Vault. However for this driver, the Publisher is the agent that performs work orders in the PBX.

Table 6-1 *Configuring the Subscriber Channel*

Rule or Policy	What it does
Subscriber Filter	Allows only events for nwoWorkOrder objects to be processed.
Event Transformation	Not used in the sample configuration.
Matching Rule	Not used in the sample configuration.
Create Rule	<p>Contains rules only for work order objects.</p> <p>Requires values for the following attributes on a work order object.</p> <ul style="list-style-type: none"> ♦ PBXName ♦ nwoStatus ♦ nwoSendtoPublisher ♦ nwoDoltNow ♦ nwoAction ♦ nwoDisplayName <p>If the values are not present, the work order is not sent to the Publisher, and therefore is not performed in the PBX.</p> <p>For a description of these attributes, see Section B.2, “DirXML-nwoWorkOrder Object,” on page 78.</p>
Placement Rule	Maps work orders from the work order container you specified to the PBX driver. This mapping is necessary so that the Subscriber can check the work orders to see whether the DoltNow flag is set to True.
Command Transformation	Not used in the sample configuration.
Schema Mapper	<p>Maps the eDirectory namespace to the PBX namespace.</p> <p>Handles mapping of work orders, extensions, and PBX site objects.</p>
Output Transformation	Not used in the sample configuration.

6.2.2 How the Publisher Channel Is Configured

Through the Publisher channel, the Identity Manager Driver for Avaya PBX queries the PBX for information about extensions. The Publisher performs tasks in the Avaya PBX, rather than the Subscriber. For a general overview of what the Publisher does, see [Section 1.4.2, “Overview of Driver Functionality,” on page 17](#).

Table 6-2 *Configuring the Publisher Channel*

Rule or Policy	What it does
Input Transformation	Not used in the sample configuration.

Rule or Policy	What it does
Schema Mapper	Maps the PBX namespace to the eDirectory namespace. Handles mapping of work orders, extensions, and PBX site objects.
Event Transformation	Not used in the sample configuration.
Publisher Filter	Allows only events for nwoWorkOrder and pbxExtension objects to be processed.
Matching Rule	Not used in the sample configuration
Placement Rule	Places workOrder objects in the correct container as defined in the driver's configuration parameters. Places nwoExtension objects in the correct container.
Command Transformation	Not used in the sample configuration.

Workforce Tree Configuration

7

The workforce tree configuration demonstrates how the driver can be configured to provision users in the PBX system by doing the following:

- ♦ Assigning an extension to new users by creating a work order when a new user is added
- ♦ Enforcing business policies about phone use restrictions, based on user attributes
- ♦ Performing work orders to install, modify, move, disable, or disconnect existing extensions, based on user events such as change in location or job status.
- ♦ Updating user objects with new phone extension information to reflect changes made in the PBX.

The following sections provide additional information about the workforce tree configuration:

- ♦ [Section 7.1, “How to Use the Workforce Tree Configuration,” on page 49](#)
- ♦ [Section 7.2, “How the Workforce Tree Configuration Works,” on page 49](#)
- ♦ [Section 7.3, “Using Log/Reporting Functionality to Report Warnings,” on page 56](#)

7.1 How to Use the Workforce Tree Configuration

Like the base configuration, the workforce tree configuration is meant to be instructional, and to demonstrate what the driver can do. It is not meant to be an out-of-the-box solution.

In contrast with the base configuration, rules are in place in the workforce tree configuration to maintain the relationships between users in the workforce tree and extensions in the PBX. For example, changes to users cause appropriate work orders to be created. And, when an extension is assigned in the PBX, the phone number is added to the user object’s attributes.

In some implementations, you should connect to an existing work order database so that the Identity Manager Driver for Avaya PBX can perform work orders from a system that is already in place. The workforce tree configuration does not demonstrate this functionality. This aspect of how the Avaya PBX driver is explained in [Chapter 8, “Work Order Database Configuration,” on page 57](#).

7.2 How the Workforce Tree Configuration Works

The Subscriber adds to the work order the ID of the user object for which the phone number is being provisioned. This can be an ID of your choice, for example, Employee ID or Social Security number. The sample configuration uses Workforce ID. When the Publisher has configured the extension, it updates the user object with the extension number. The driver puts the Object ID in the Cable field of the extension in the PBX. This field has a character limit of 5 characters.

Figure 7-1 Workforce Tree Configuration

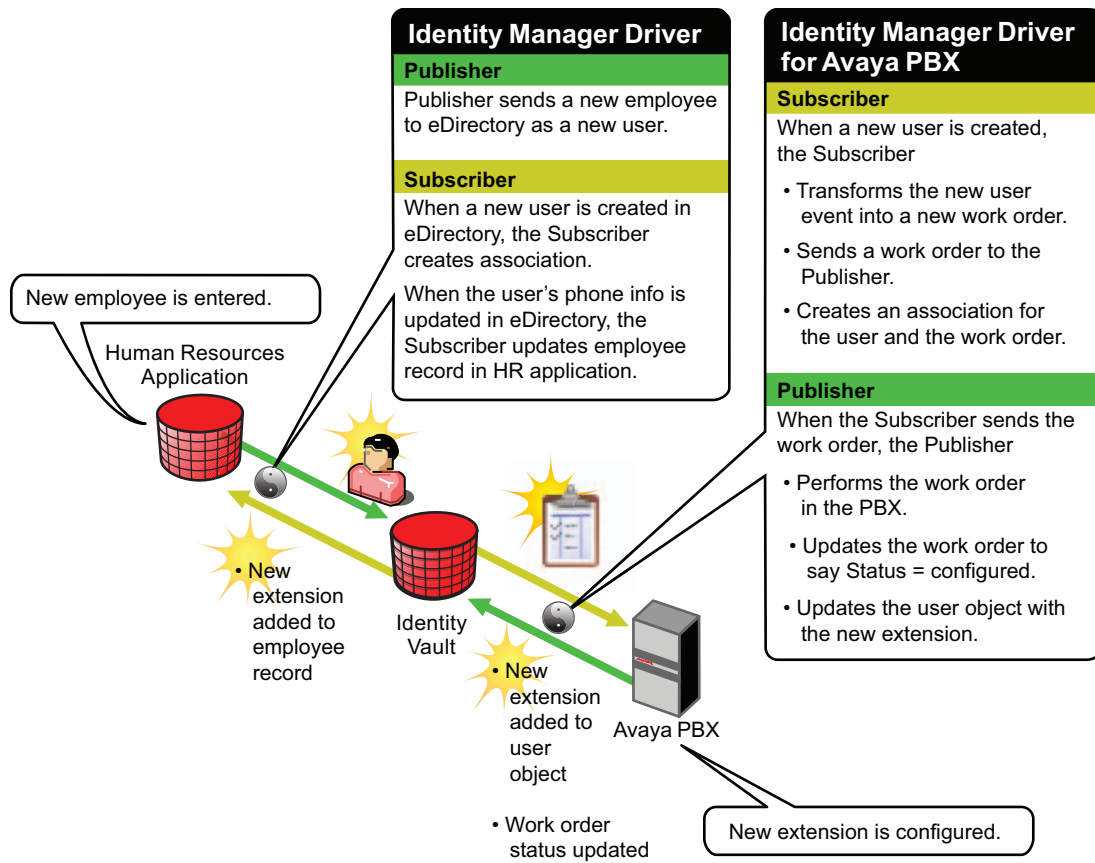
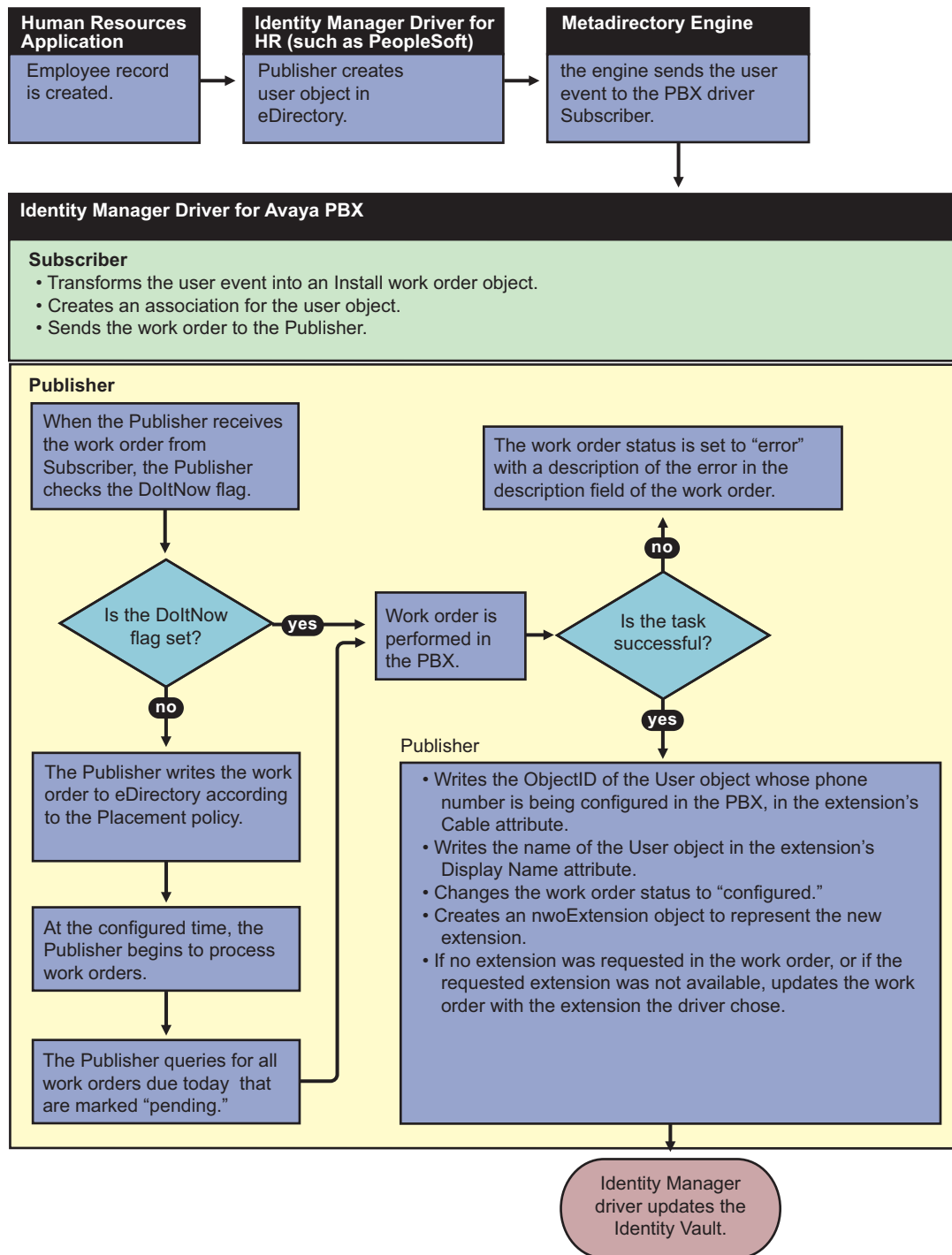


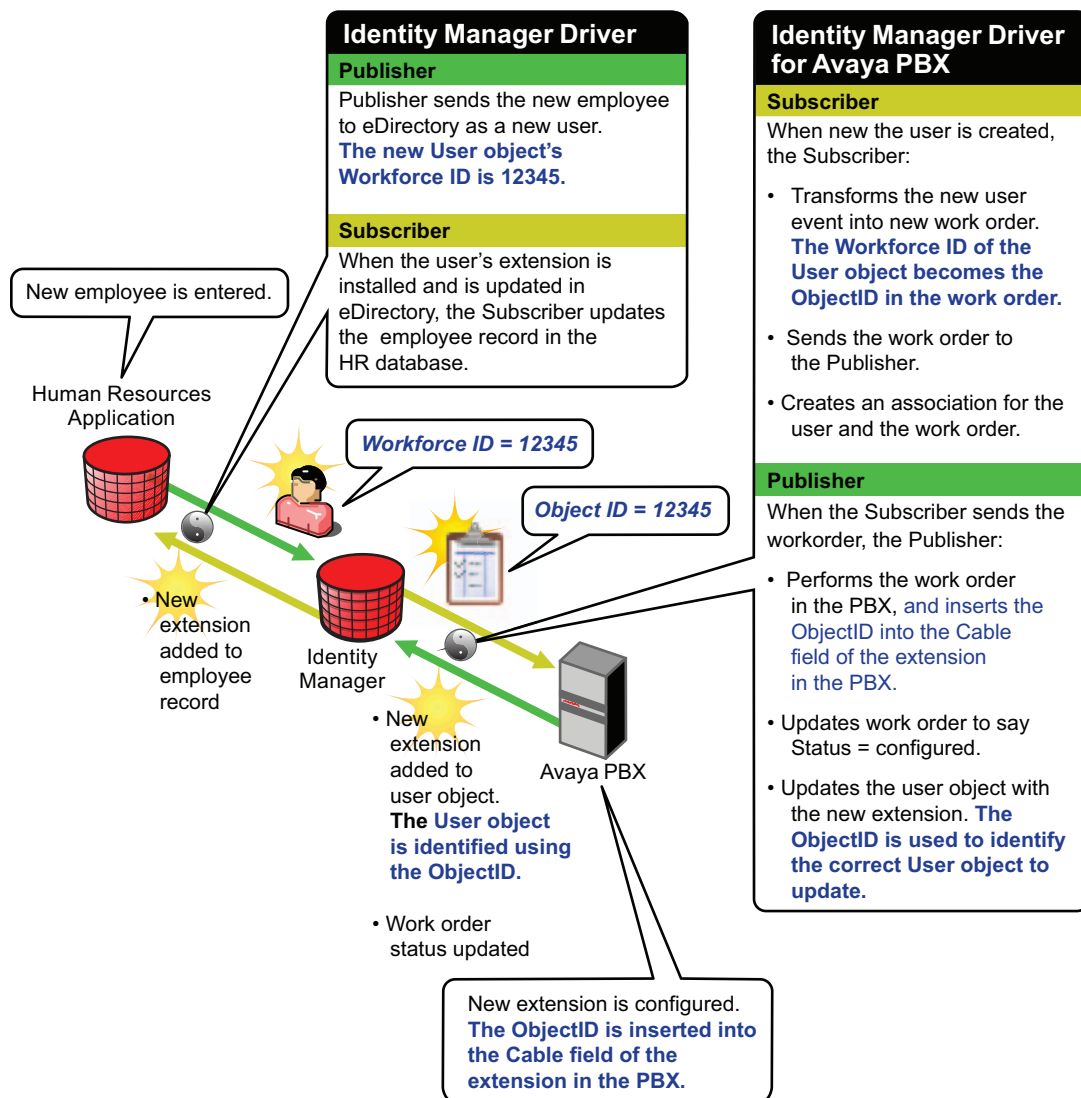
Figure 7-2 Flowchart of the Workforce Tree Configuration



7.2.1 How the Driver Uses the ObjectID to Identify and Update Users

The following figure shows where the ObjectID comes from and how it is used by the driver to update users. In this diagram you can trace the role of the ObjectID; the user's ID becomes the ObjectID of the work order, and is entered into the PBX as an attribute of the extension. After the task is complete, the ObjectID is used to identify the user object that needs to be updated.

Figure 7-3 How the Object ID Is Used in the Process



7.2.2 How the Subscriber Channel Is Configured

In the workforce tree configuration, the Subscriber channel listens for events that pertain to work orders and users. (PBX site objects must also exist in eDirectory, but they are used only for the driver to query for information about how to contact each PBX. Changes to PBX objects are not processed by the Subscriber; instead you must restart the driver for changes to be recognized.)

In the workforce tree configuration, a work order can be created or a user can be created, which then causes Identity Manager to create a work order.

Table 7-1 *Configuring the Subscriber Channel*

Rule or Policy	What it does
Subscriber Filter	Allows only events for nwoWorkOrder and User objects to be processed.
Event Transformation	Not used in the sample configuration.
Matching Rule	Not used in the sample configuration.
Create Rule	<p>Contains rules only for nwoWorkOrder and User objects.</p> <p>For an nwoWorkOrder object, requires values for the following attributes.</p> <ul style="list-style-type: none"> ◆ PBXName ◆ nwoStatus ◆ nwoSendtoPublisher ◆ nwoDoltNow ◆ nwoAction ◆ nwoDisplayName <p>For a User object, requires values for the following attributes.</p> <ul style="list-style-type: none"> ◆ Surname ◆ L <p>As an example in the sample configuration, L (location) is used to provision the extension differently depending on where the user is physically located. (You could customize the policy to use L to determine which PBX site to use, or what setcor restrictions to use.)</p> <ul style="list-style-type: none"> ◆ Workforce ID <p>The Workforce ID is important in the sample configuration because it is used to populate the ObjectID so the user can be identified with a work order.</p> <p>If values are not present for the required attributes, the work order is not sent to the Publisher, and therefore is not performed in the PBX.</p> <p>For a description of these attributes, see Section B.2, "DirXML-nwoWorkOrder Object," on page 78 and Section B.5, "User Objects and Their Identity Manager Associations," on page 86.</p>
Placement Rule	<p>Maps work orders from the work order container you specified to the Identity Manager Driver for Avaya PBX. This mapping is necessary so that the Subscriber can check the work orders to see whether the DoltNow flag is set to True.</p> <p>Maps user object containers to the Identity Manager Driver for Avaya PBX. This makes sure that users, when they are created, are sent to the driver so that work orders can be created to provision their phone extensions.</p>

Rule or Policy	What it does
Command Transformation	<p>Takes user events and changes them to PBX actions.</p> <ul style="list-style-type: none"> ♦ Converts a user Add event into a work order object. ♦ Determines the PBX and node to use based on the Location attribute of the user object. (The configuration includes sample code for how to do this. You could use another user attribute instead, such as job title.) ♦ Determines the phone type based on the Location attribute of the user object. Then, determines the dupe extension to use based on the phone type. The sample configuration shows an example of mapping phone type to dupe extension.
Schema Mapper	<p>Maps the eDirectory namespace to the PBX namespace.</p> <p>Handles mapping of work orders, extensions, and PBX site objects.</p>
Output Transformation	<p>For work orders you create in the work order container, maps the phone type to the duplicate extension if the dupe extension is not indicated in the work order. (Use the PBX “template” for how to configure the phone type).</p>

7.2.3 How the Publisher Channel Is Configured

Through the Publisher channel, the Identity Manager Driver for Avaya PBX queries the PBX for information about extensions, places work order objects in the correct container, performs work orders, and sends updates to user objects.

Table 7-2 *Configuring the Publisher Channel*

Rule or Policy	What it does
Input Transformation	Not used in the sample configuration.
Schema Mapper	<p>Maps the PBX namespace to the eDirectory namespace.</p> <p>Handles mapping of work orders, extensions, and PBX site objects.</p>
Event Transformation	<p>Transforms pbxExtension Add, Modify, or Delete events into events that modify user objects.</p> <ul style="list-style-type: none"> ♦ Transforms nwoExtension Add events into user Modify events. Uses the ObjectID from the work order to identify the user so it can add the new extension to the User object's telephone number. ♦ Transforms Modify events into events to modify the preferred name or object ID. ♦ Transforms Delete events for extensions into Modify events to remove the extension from all users who have it in their phone list. This is done by a query for users, rather than by the Object ID.
Publisher Filter	Allows only events for nwoWorkOrder, User, and extension objects to be processed.
Matching Rule	Not used in the sample configuration.
Create Rule	Not used in the sample configuration.

Rule or Policy	What it does
Placement Rule	<ul style="list-style-type: none"> ◆ Places work order objects in the correct container. ◆ Places extension objects in the correct container.
Command Transformation	Not used in the sample configuration.

7.2.4 User Events That Can Trigger a Work Order

Table 7-3 *Configuring User Events*

User event from the Human Resources application	Work Order that is created by the policies	Ideas for what you could do when customizing the policies
New employee is entered in HR application	<ul style="list-style-type: none"> ◆ An Install work order is created to install new extension ◆ The work order is marked DoItNow, so it is performed immediately ◆ The user is updated with the new phone extension in the Identity Vault 	You could configure the HR driver to update the user's information in the HR application.
Employee moves to a different location	No action is taken in the sample configuration.	<p>You could customize the policies to do the following:</p> <ul style="list-style-type: none"> ◆ Create a Move work order ◆ If the move requires a new extension update, give the employee a new extension ◆ Remove the old extension from the user object.
Employee's name changes	<ul style="list-style-type: none"> ◆ A Modify work order is created and is performed immediately ◆ The display name in the PBX is modified, so the user's phone shows the correct name on outgoing calls. 	
Employee goes on extended leave	No action is taken in the sample configuration.	You could customize the policies to create a Disable work order.
Employee leaves the company	No action is taken in the sample configuration.	You could customize the policies to disconnect the extension.

7.3 Using Log/Reporting Functionality to Report Warnings

You might want to record the warnings that occur if a user object that is created does not match the Create policy criteria. This would mean that someone is not following the business processes.

You can keep track of this by using the log file, the Event Auditing Service, or Novell Sentinel to capture the warning messages that are generated.

Work Order Database Configuration

8

As described in [Chapter 7, “Workforce Tree Configuration,” on page 49](#), you can use the Identity Vault as your work order database, using work order objects. However, if you have another work order database that you are already using to enter PBX work orders, the Avaya PBX driver can fit into that environment as well. The driver can perform work orders that are created in another application and synchronized to the Identity Vault.

For example, if personnel in your organization are accustomed to using a JDBC database product to enter work orders for phone extensions, you could preserve the existing process for entering work orders while using the Avaya PBX driver to automate how the work orders are performed. This kind of solution would use two Identity Manager drivers, one for Avaya PBX, and one for the JDBC database.

The following sections provide additional information about the work order database configuration

- ♦ [Section 8.1, “How to Use the Work Order Database Configuration,” on page 57](#)
- ♦ [Section 8.2, “Creating a Work Order Database Configuration,” on page 57](#)
- ♦ [Section 8.3, “About Work Order Systems,” on page 61](#)
- ♦ [Section 8.4, “Using the Log/Reporting Functionality to Report Warnings,” on page 61](#)

8.1 How to Use the Work Order Database Configuration

Like the base configuration, the work order database configuration is meant to be instructional, and to demonstrate what the driver can do. It is not meant to be an out-of-the-box solution.

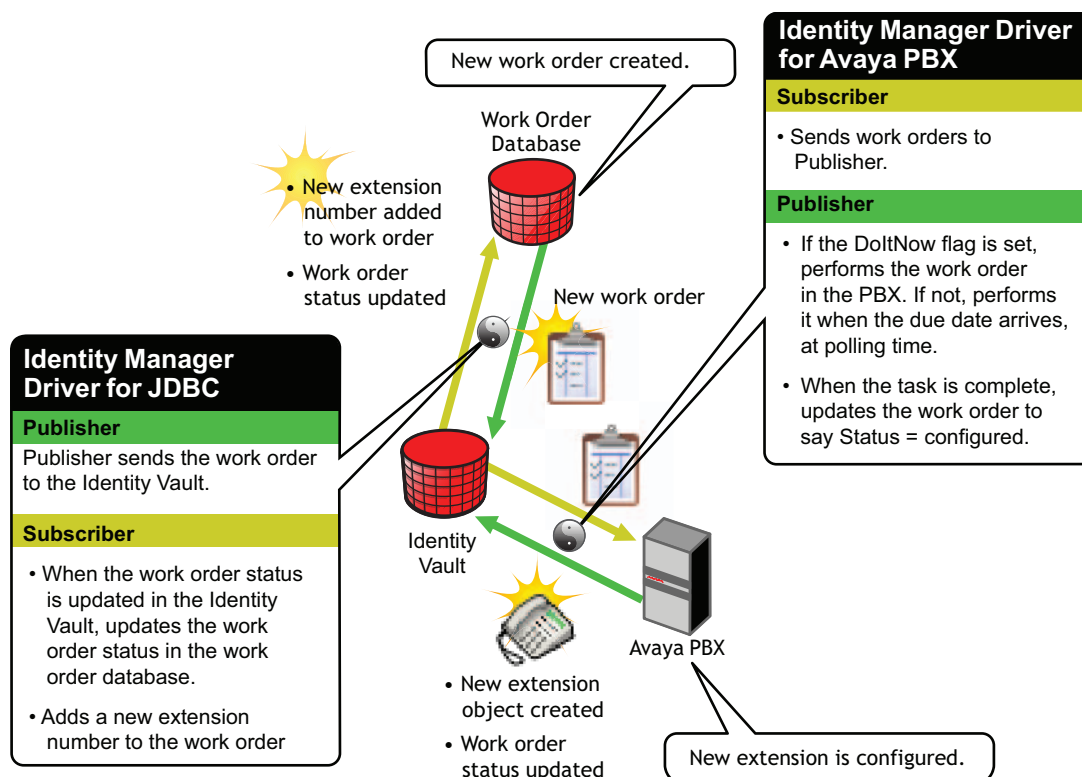
This configuration could be combined with the workforce tree configuration so that work orders can have two sources:

- ♦ Automated requests triggered by user events in the Identity Vault, such as new extensions assigned for new employees.
- ♦ Manual requests for existing employees entered in a work order database by administrative assistants or IT personnel, such as modifying a display name or moving offices.

8.2 Creating a Work Order Database Configuration

The following diagram shows how a work order database configuration could be set up, using the example of a work order for installing a new extension.

Figure 8-1 Work Order Database Configuration



The following flowcharts show the process that could be followed for a work order database configuration. [Figure 8-2](#) is a flowchart of how the configuration could work for an Install work order with the DoItNow flag set to true, and [Figure 8-3](#) is a flowchart showing how the configuration could work for an Install work order with the DoItNow flag not set.

Figure 8-2 Flowchart of the Work Order Database Configuration with the DoItNow flag Set to True

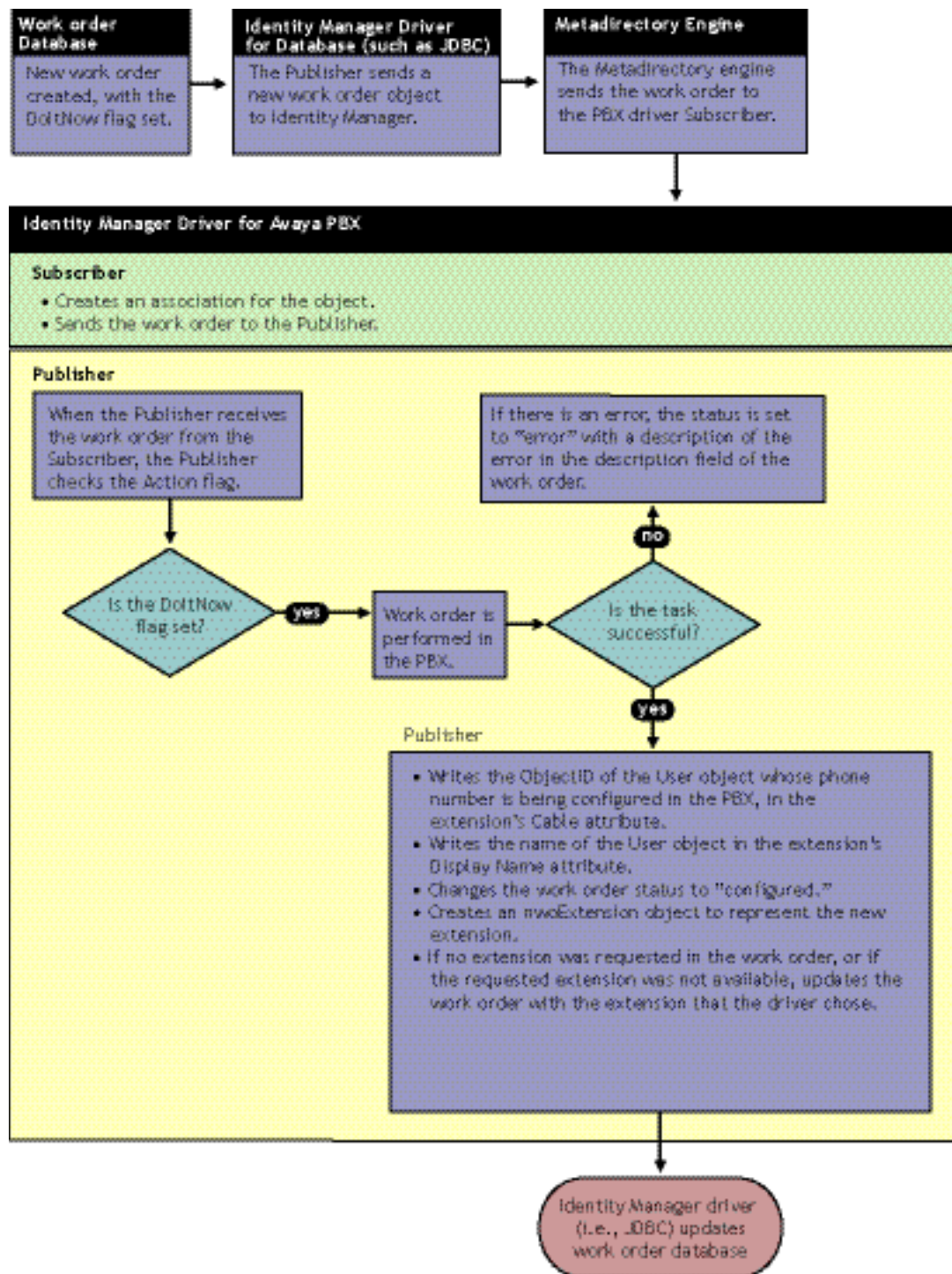
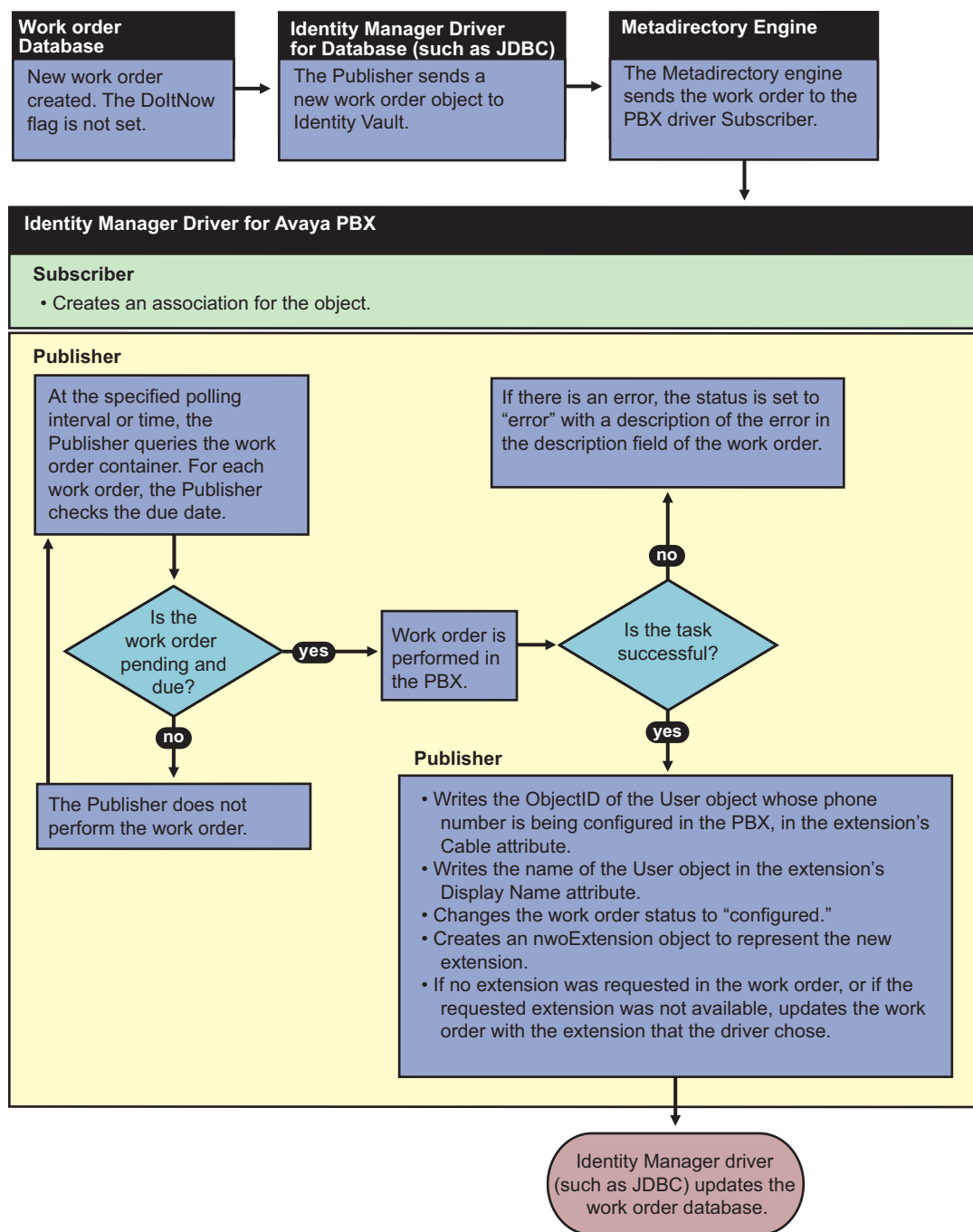


Figure 8-3 Flowchart of the Work Order Database Configuration with the DoItNow Flag Not Set



8.2.1 How To Configure the Subscriber Channel

In a work order database configuration, the Subscriber channel listens for work order events.

Work orders are entered in the work order database, and are mirrored in Identity Vault as `nwoWorkOrder` objects because the JDBC driver sends them to Identity Vault. You might need to perform some data manipulation to ensure that the database work order maps correctly to the PBX driver work order.

The Identity Manager Driver for Avaya PBX performs the work orders and the results are returned to Identity Vault. The JDBC driver then updates the work order database with the results. This allows the person who entered the work order to check on the status in the work order database.

8.2.2 How To Configure the Publisher Channel

In this kind of configuration, work orders should always have the Send to Publisher Flag set to False, because all work orders are already created in the work order container. The Publisher should only update status for work orders. In contrast to the Workforce Tree configuration, the Publisher should not need to create any new work order objects.

8.3 About Work Order Systems

Many environments currently use a work order system to keep track of changes to PBX extensions. A work order database design for the driver can work well with existing work order systems. Most work order systems are forms-based front ends to a set of database tables, capable of enforcing the user's business rules and approval process. To avoid customizing the driver for each work order system, a JDBC driver or other custom driver can be responsible for synchronizing work order data between the corporate work order system and the generic work order container in the Identity Vault used by the driver.

8.4 Using the Log/Reporting Functionality to Report Warnings

You might want to record the warnings that occur if a work order that is created does not match the Create policy criteria. This would mean that someone is not following the business processes.

You can keep track of this by using the log file or Novell Sentinel to capture the warning messages that are generated.

As you work with the Avaya PBX driver, there are a variety of management tasks you might need to perform:

- ♦ [Section 9.1, “Changing How Often the Driver Performs Work Orders,” on page 63](#)
- ♦ [Section 9.2, “Changing the Location of PBX Site, Work Orders, User, and Extension Objects,” on page 63](#)
- ♦ [Section 9.3, “Managing Existing Users,” on page 64](#)
- ♦ [Section 9.4, “Common Management Tasks,” on page 65](#)

The first three sections describe tasks that are specific to the Avaya PBX driver. The last section lists tasks that are common for all drivers.

9.1 Changing How Often the Driver Performs Work Orders

The driver provides two ways to specify when you want work orders to be performed: polling interval and time of day. You can use either one, or both together, to achieve the frequency and timing that’s appropriate for your environment.

- 1 In iManager, go to the properties for the driver object:
 - 1a Click *DirXML Management > Overview*. Choose the driver set, and in the diagram that appears, click the *Avaya PBX driver* icon. Click it again in the next page to see the driver parameters.
- 2 To enter or change the polling interval, change the number of minutes shown.
 - ♦ If you want the driver to perform work orders at a certain polling interval, specify a number of minutes.
 - ♦ If you want to turn off the polling interval so that the driver performs work orders only at a time of day you specify, specify 0 in the field.

This choice is useful if you want to make sure that work orders are not being performed during daytime work hours.
- 3 To specify or change a certain time of day, change the time using the correct format.
 - ♦ If you want the driver to perform work orders at a certain time of day, specify a time.
 - ♦ If you want to turn off the time of day polling so that the driver performs work orders only at a polling interval, leave the field blank.

9.2 Changing the Location of PBX Site, Work Orders, User, and Extension Objects

When you import the driver from a configuration file, you are prompted to enter the location (DN) for the kinds of objects the driver needs to read:

- ♦ pbxSite

- ♦ nwoWorkOrder
- ♦ User objects
- ♦ pbxExtension objects

If you want to change the location, change the DN under the driver configuration parameters found in the driver's Properties page.

Example DN for the container holding pbxSite objects:

```
\t=CHD1_TREE\o=n\L=DirXML\O=Test\OU=PbxSite
```

Example DN for the container holding nwoWorkOrder objects:

```
\t=CHD1_TREE\o=n\L=DirXML\O=Test\OU=PbxWorkOrders
```

9.3 Managing Existing Users

After performing a work order, the driver can update a user's information with the results. For example, the driver can add a new extension to the user's list of phone numbers, or remove an extension that has been deleted, as described in [Chapter 7, "Workforce Tree Configuration," on page 49](#).

This functionality works for new and existing users, if the following conditions are met:

- ♦ The user exists in the Identity Vault.
- ♦ The policies support updating a User object after performing a work order.
- ♦ The correct ID for the user object is entered as the ObjectID in the pbxWorkOrder object.

The ObjectID in the work order is what allows the driver to update the user object when the work order is complete.

This means that your ability to manage existing users (perform work orders for their extensions and update their phone information after a work order is performed) is dependent on the accuracy with which the ObjectID is specified in the work order. This is a contrast to some other Identity Manager drivers, which require the user object to have an Identity Manager association with the driver before you can manage existing users.

Unlike some Identity Manager drivers, it is not necessary to use the `Migrate into NDS` command before being able to manage existing users. In fact, for this driver, the `Migrate into NDS` command does not affect user objects (unless you create custom policies or tools to do so).

The `Migrate into NDS` command allows you to import data from an application into Identity Vault. For the Identity Manager Driver for Avaya PBX, the `Migrate into NDS` command causes all extensions that are configured in the PBX to be created as `pbxExtension` objects in eDirectory.

This action does not create associations between existing User objects in eDirectory and existing extensions. However, this list of existing extensions could be used for the following purposes as part of a manual effort or with custom tools you create:

- ♦ In the PBX, inserting the ID for the User object who uses each extension

When the driver configures an extension, it automatically enters this information in the PBX, using the ObjectID you specify in the work order. However, for existing extensions, the data might not have been filled in correctly.

This is useful if the PBX admin will continue to make changes manually to extensions, as well as using the driver to perform work orders. However, keep in mind that any manual change must include adding the ObjectID in the PBX, or else the driver won't know which user object to update.

You can write a policy for migrating User IDs to the PBX based on their extension numbers. You can also migrate all User objects from eDirectory. To do this, set up a migration policy that creates work orders that modify ObjectID's based on extension numbers. When the migration policy has finished, change the policy to something you want for normal operations.

- ♦ Checking the accuracy of existing phone information listed for users in Identity Vault

Although the driver can add, change, or remove phone extensions listed for an existing user object after it performs a work order, it does not verify that the extensions already listed for the user are correct. So, if an incorrect extension is entered manually, that data entry error remains even after you start using the driver.

If you want to make sure that existing phone information listed for users is correct, you can compare the pbxExtension objects with the user objects by looking at the Display Name in the pbxExtension object. If display names are created with some consistency, you might be able to use a tool to match up many of the extension objects with user objects. However, some manual work is probably still required to match all of the extensions, because of duplicate employee names (such as two people who are both named John Smith) or inconsistent format of display names that were entered manually.

- ♦ Check the PBX for unassigned extensions.

You can use the nwoExtension object to find out whether you have extensions that need to be disconnected but the physical task was never completed.

- ♦ Check the display name of extensions.

You can check to make sure display names are filled in and are following any applicable corporate standards.

- ♦ Populating eDirectory with User objects

This might be an option for you if you are fairly confident in the information that is in the PBX, and you are creating a new eDirectory tree and want to populate it with users based on the users represented in the PBX. You can do this through a policy set up as you migrate User objects to eDirectory.

9.4 Common Management Tasks

The following list includes management tasks that are common to all Identity Manager drivers. For details about how to perform these tasks, see the [Identity Manager 4.0 Common Driver Administration Guide](#).

- ♦ Starting, stopping, and restarting the driver
- ♦ Viewing driver version information
- ♦ Using Named Passwords to securely store passwords associated with the driver
- ♦ Monitoring the driver's health
- ♦ Backing up the driver
- ♦ Inspecting the driver's cache files
- ♦ Viewing the driver's statistics

- ♦ Using the DirXML Command Line utility to perform management tasks through scripts
- ♦ Securing the driver and its information
- ♦ Synchronizing objects
- ♦ Migrating and resynchronizing data

10.1 Troubleshooting Driver Processes

Viewing driver processes is necessary to analyze unexpected behavior. To view the driver processing events, use DSTrace. You should only use it during testing and troubleshooting the driver. Running DSTrace while the drivers are in production increases the utilization on the Identity Manager server and can cause events to process very slowly. For more information, see “[Viewing Identity Manager Processes](#)” in the *Identity Manager 4.0 Common Driver Administration Guide*.

10.2 Tree information in the WorkOrder DN of the Avaya PBX Driver throws an Exception for PBX WorkOrders

If you select the Avaya PBX driver for processing PBX WorkOrder with tree information in the WorkOrder DN, it throws a `StringIndexOutOfBoundsException` exception.


To work around this issue,

- 1 Go to *Edit Properties* option of the AvayaPBX driver for which you want to process the WorkOrder.
- 2 Click the *Driver Configuration* and scroll down to the *Driver Parameters* option.
- 3 In the PBX WorkOrder objects DN text box, remove the tree information from the PBX WorkOrder objects DN. For example, change
`\t=IDM4_FW\o=n\L=DirXML\O=Test\OU=pbxWorkOrders` to
`\o=n\L=DirXML\O=Test\OU=pbxWorkOrders`
- 4 Select *OK*, then restart the AvayaPBX driver.
- 5 Evaluate the WorkOrders.

Driver Properties

A


This section provides information about the Driver Configuration and Global Configuration Values properties for the Avaya PBX driver. These are the only unique properties for the Avaya PBX driver. All other driver properties (Named Password, Engine Control Values, Log Level, and so forth) are common to all drivers. Refer to “[Driver Properties](#)” in the *Identity Manager 4.0 Common Driver Administration Guide* for information about the common properties.

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

- ♦ [Section A.1, “Driver Configuration,” on page 69](#)
- ♦ [Section A.2, “Global Configuration Values,” on page 73](#)

A.1 Driver Configuration

In iManager:

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

In Designer:

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



The Driver Configuration options are divided into the following sections:

- ♦ [Section A.1.1, “Driver Module,” on page 69](#)
- ♦ [Section A.1.2, “Driver Object Password \(iManager Only\),” on page 70](#)
- ♦ [Section A.1.3, “Authentication,” on page 70](#)
- ♦ [Section A.1.4, “Startup Option,” on page 71](#)
- ♦ [Section A.1.5, “Driver Parameters,” on page 72](#)
- ♦ [Section A.1.6, “ECMAScript,” on page 73](#)
- ♦ [Section A.1.7, “Global Configuration,” on page 73](#)

A.1.1 Driver Module

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

Table A-1 *Driver Module*

Option	Description
<i>Java</i>	<p>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.</p> <p>The name of the driver's Java class is:</p> <pre>com.novell.nds.dirxml.driver.nds.DriverShimImpl</pre>
<i>Native</i>	This option is not used with the Avaya PBX driver.
<i>Connect to Remote Loader</i>	<p>Used when the driver is connecting remotely to the connected system. Designer includes two suboptions:</p> <ul style="list-style-type: none">◆  <i>Driver Object Password</i>: Specifies a password for the Driver object. If you are using the Remote Loader, you must enter a password on this page. Otherwise, the remote driver does not run. The Remote Loader uses this password to authenticate itself to the remote driver shim.◆  <i>Remote Loader Client Configuration for Documentation</i>: Includes information on the Remote Loader client configuration when Designer generates documentation for the Delimited Text driver.

A.1.2 Driver Object Password (iManager Only)


Table A-2 *Driver Object 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.

A.1.3 Authentication

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

Table A-3 *Authentication*

Option	Description
<i>Authentication ID</i> or  <i>User ID</i>	<p>Specify a user application ID. This ID is used to pass Identity Vault subscription information to the application.</p> <p>Example: Administrator</p>


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

A.1.4 Startup Option

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

Table A-4 Startup Option

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

Option	Description
 <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.

A.1.5 Driver Parameters

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

The Avaya PBX driver does not have any Driver Options or Subscriber Options. The following table describes the Publisher Options.

Table A-5 *Publisher Options*

Parameter	Description
General Options	
<i>PBX Name</i>	Specify the name of the PBX site object. This is the same name you use for the creation of the PBX site object in the directory. For example: [ProvoPBX]
<i>PBX Site Objects DN</i>	Select the container where the PBX site object resides.
<i>Work Orders Container</i>	Select the container where the PBX work orders object resides.
<i>Extension Container</i>	Select the container where the PBX extension objects will be created. This is required only if your configuration uses extension objects. If your configuration writes PBX information directly to user objects (like the workforce tree configuration does), you do not need to specify an extension container.
<i>Audix Subscriber Container</i>	Select the container where the Audix subscriber objects will be placed in the directory. If your configuration does not include Audix subscribers, you do not need to specify a subscriber container.
<i>Poll Interval (minutes)</i>	Specifies the number of minutes between checks for available transactions to process. The default is 1. Use 0 to disable the polling interval, in which case you need to set a poll time.
<i>Poll Time</i>	Select a poll time for the driver to use. For example, 12:00 AM. If no poll time is selected, the driver uses the poll interval only.
Emulation Options	Use these options to set up emulation for testing purposes. For information about additional tasks required to set up emulation, see Section 4.2.3, “Using Emulation to Test the Driver,” on page 35.
<i>Show emulation options</i>	Select <i>show</i> to display the emulation options. Select <i>hide</i> if you don't need to see the options.
<i>Emulation LDAP host IP address</i>	For emulation, the driver needs to access an LDAP host (eDirectory or any other host). Specify the IP address of the LDAP server.
<i>Emulation LDAP Port</i>	Specify the port number of the LDAP server used for emulation. The default is 389.

Parameter	Description
<i>DN of login user for emulation</i>	For emulation, the driver requires a username that has access to the LDAP server. This user must have rights to read/write to the extension containers on the server. Specify the name of the user for the LDAP server.
<i>User password for emulation</i>	Specify the password of the user who accesses the LDAP server. Confirm the password. You can also clear the password by selecting <i>Remove existing password</i> (iManager) or <i>Clear password</i> (Designer) and then clicking <i>Apply</i> .
<i>Extension Container DN for Emulation</i>	Specify the DN of the container where the driver will add or modify extension objects (this container must already exist on the LDAP server.) For example, <code>ou=PbxExtensions, ou=AvayaPBX, o=Novell</code>

A.1.6 ECMAScript

Displays an ordered list of ECMAScript resource files. The files contain extension functions for the driver that Identity Manager loads when the driver starts. You can add additional files, remove existing files, or change the order the files are executed.

A.1.7 Global Configuration

Displays an ordered list of Global Configuration objects. The objects contain extension GCV definitions for the driver that Identity Manager loads when the driver is started. You can add or remove the Global Configuration objects, and you can change the order in which the objects are executed.


A.2 Global Configuration Values

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

The Avaya PBX driver includes several GCVs that are created from driver parameters. When you modify the driver parameters, the GCVs are updated; likewise, when you modify the GCVs, the driver parameters are updated. These GCVs are created so that the driver parameter information can be more easily used in the driver's policies.

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


To access the driver's GCVs in iManager:

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

or

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

To access the driver's GCVs in Designer:

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

or


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





Table A-6 *Global Configuration Values*

Option	Description
<i>PBX Name</i>	Specify the name of the PBX site object. This is the same name you use for the creation of the PBX site object in the directory. For example: [ProvoPBX]
<i>PBX Site Objects DN</i>	Select the container where the PBX site object resides.
<i>Work Orders Container</i>	Select the container where the PBX work orders object resides.
<i>Extension Container</i>	Select the container where the PBX extension objects will be created. This is required only if your configuration uses extension objects. If your configuration writes PBX information directly to user objects (like the workforce tree configuration does), you do not need to specify an extension container.
<i>Audix Subscriber Container</i>	Select the container where the Audix subscriber objects will be placed in the directory. If your configuration does not include Audix subscribers, you do not need to specify a subscriber container.

Schema for PBX Management

B

As part of the installation for the Identity Manager Driver for Avaya PBX, the eDirectory schema is extended to include four new object classes:

- ♦  [DirXML-pbxSite](#)
- ♦  [DirXML-nwoWorkOrder](#)
- ♦  [DirXML-pbxExtension](#)
- ♦  [DirXML-pbxAudixSubscriber](#)

These objects allow the driver to connect to the PBX correctly, perform work orders, and create pbxExtension objects to represent new extension.

Installing Identity Manager 3 provides iManager plug-ins to help you create or view these objects in the Avaya PBX driver.

B.1 pbxSite Object

This object is used to represent the PBX site. The driver uses the information about the PBX site to connect to the PBX.

The driver cannot perform work orders unless the following conditions for pbxSite objects are met:

- ♦ A DirXML-pbxSite object is created for each PBX.
- ♦ The object contains correct values for the required attributes.

Not all attributes are required for all environments; some of them depend on factors such as whether you have hot or cold jacks, and whether you have non-DID phone numbers. These are noted in the tables below.
- ♦ In the driver parameters, the correct container is specified for PBX site objects.
- ♦ Each time you create or make changes to a DirXML-pbxSite object, you must stop and restart the driver.

This step is necessary so the driver recognizes the changes. The driver queries for PBX site information only at startup. Because of this, an Identity Manager association is not usually necessary for DirXML-pbxSite objects and they do not need to be included in the Filter.

The following table shows the attributes you need to specify for the PBX site. Use the iManager plug-ins to make this easier.

You must specify values for each attribute unless the table indicates that a value is not required or is conditional.

Table B-1 *pbxSite Object Attributes*

pbxSite Attributes (eDirectory Namespace)	PbxSite Attributes (Driver Namespace)	Description	Format	Sample Value
Common Name	No mapping is necessary for this in the driver namespace	This is the naming attribute for eDirectory. In the sample configurations discussed in this manual, the value comes from the PbxName attribute.	Case ignore string	ProvoPBX
DirXML-pbxName	PbxName	Local name for the PBX.	Single value case ignore string	ProvoPBX
DirXML-pbxLoginName	LoginName	Name used to authenticate to PBX.	Single value case ignore string The Identity Vault and the driver ignore the case, but the PBX itself might be case sensitive.	mylogin
DirXML-pbxPassword	Password	Password used to authenticate to PBX.	Single value case ignore string the Identity Vault and the driver ignore the case, but the PBX itself might be case sensitive.	mypassword
DirXML-pbxBrand	Brand	(Optional) Brand/Model of PBX.	Single value case ignore string	Avaya

pbxSite Attributes (eDirectory Namespace)	PbxSite Attributes (Driver Namespace)	Description	Format	Sample Value
DirXML-pbxHotJacks	HotJacks	Are jacks hot? If your environment has cold jacks, you must also fill in the Nodes attribute.	Single value Boolean	true
DirXML-pbxAccessType	AccessType	How to access PBX: telnet. (All valid values are listed in the Sample column.)	Single value case ignore string	telnet or Emulate Audix or AudixEmulate
DirXML-pbxExtensionLength	ExtenLength	The number of digits used for extensions.	Single value numeric string	5 - 9
DirXML-pbxPhoneBlockBegin	PhoneBlockBegin	Beginning number block of DID extensions assigned to PBX.	Single value Numeric string	51000
DirXML-pbxPhoneBlockEnd	PhoneBlockEnd	Ending number block of DID extensions assigned to PBX.	Single value Numeric string	51999
DirXML-pbxNonDidPhoneBlockBegin	NonDidPhoneBlockBegin	(Conditional: This attribute is required only if you use non-DID phone numbers.) Beginning number block of non-DID extensions assigned to PBX.	Single value Numeric string	600000

pbxSite Attributes (eDirectory Namespace)	PbxSite Attributes (Driver Namespace)	Description	Format	Sample Value
DirXML- pbxNonDidPhoneBlockEnd	NonDidPhoneBlockEnd	(Conditional: This attribute is required only if you use non-DID phone numbers.) Ending number block of non-DID extensions assigned to PBX.	Single value Numeric string	60999
DirXML-pbxIpAddress	IpAddress	Telnet IP address	Single value case ignore string	133.65.121.9 8
DirXML-pbxNodes	Nodes	(Required only if you have cold jacks.) For example, if you had two nodes with three cabinets each, the nodes listed in this attribute might have the names shown in the sample value.	Multi value case ignore string	Building H East, 01, 02, 03 Building H West, 04, 05, 06

B.2 DirXML-nwoWorkOrder Object

The DirXML-nwoWorkOrder object (sometimes referred to as the work order object) is used to tell the driver what tasks to perform in the PBX and to allow the driver to record the results.

If your configuration is intended to update phone information for user objects after PBX tasks are complete, the ObjectID attribute must be filled in correctly on every work order. For example, in the workforce tree configuration explained in [Chapter 7, “Workforce Tree Configuration,” on page 49](#), the ObjectID is used to find and update the User object in the Identity Vault, which can subsequently be used to update phone information for the employee in the human resources application. The ObjectID allows this flow of information to take place. Without it, the driver can perform the work order in the PBX, but no user information is updated afterward.

The Avaya driver has two flags to initiate a work order event.

- ♦ **DoItNowFlag** When this flag is set to True for a work order, the Subscriber wakes up the Publisher by sending the work order to the Publisher. The Publisher performs the task immediately instead of waiting for the next polling time or polling interval.

This flag is useful in a situation where you want the work order to be completed right away. You can set this flag to True when you manually create a work order, or in an automated solution you can use policies to determine whether the flag should be set. For example, you could configure the driver to set the DoItNow flag to True for an incoming work order if a corresponding attribute is set in a work order database. As another example, you could configure the driver to set the DoItNow flag to True if an Install work order is triggered by a new user in a human resources application.

- ♦ **SendToPublisher Flag** When this flag is set to True for a work order, the Subscriber sends the work order to the Publisher, and the Publisher writes the work order object in the correct container according to the work order container specified in the Configuration Parameters.

This flag is not necessary in implementations where the work order object is created in the Identity Vault by an administrator (as in [Chapter 6, “Base Configuration,” on page 43](#)) or an application (like the solution described in [Chapter 8, “Work Order Database Configuration,” on page 57](#)).

This flag is useful when the work order is triggered by another Identity Vault event through the use of a policy or style sheet, and the work order does not yet exist in the Identity Vault as an object. This kind of scenario is described in [Chapter 7, “Workforce Tree Configuration,” on page 49](#).

The following table shows the required attributes for a work order that you do need to specify:

Table B-2 Required Work Order Attributes That Must Be Specified

Required Attributes	Description	Values or examples
DirXML-pbxName	Name of the PBX on which to perform this work order	ProvoPBX
DirXML-nwoStatus	State of the work order so the driver knows what to do with this work order	pending error warning configured
DirXML-nwoDoItNowFlag	When to perform the action on this work order	True or False
DirXML-nwoAction	What action needs to be performed on this work order	install move disable enable disconnect modify setcor

The following table shows all of the attributes for a work order:

Table B-3 *nwoWorkOrder Object Attributes*

nwoWorkOrder Attributes (eDirectory Namespace)	PbxWorkOrder Attributes (Driver Namespace)	Description	Format	Sample Value
Common Name	No mapping is necessary for this in the driver namespace	<p>The naming attribute for eDirectory.</p> <p>In the sample configurations discussed in this guide, the value comes from the WorkOrderNumber attribute.</p>	Case ignore string	WorkOrder1
DirXML- nwoExtension	Extension	<p>The extension for which the work order is being performed.</p> <p>If you are installing a new extension, you can request a particular extension by entering a number here. If the extension is not available, the driver assigns the next available extension in numeric order.</p> <p>If this attribute is blank for an Install work order, the driver assigns the next available extension in numeric order.</p>	Numeric string	12345 or no value
DirXML- nwoDueDate	DueDate	<p>Work order due date.</p> <p>Must be in the format MM/DD/YYYY.</p>	Case ignore string	06/12/2003
displayName	UserName	<p>Name to be displayed on caller ID.</p> <p>You can automate the creation of this attribute based on the name of the User object.</p> <p>You can use a work order to change just the display name for an extension, if desired.</p>	Case ignore string	Doe, John

nwoWorkOrder Attributes (eDirectory Namespace)	PbxWorkOrder Attributes (Driver Namespace)	Description	Format	Sample Value
DirXML-nwoObjectID	ObjectID	(Required by the policies only) The ID used by policies to associate to the user object. You can use an ID of your choice, such as the employee ID.	Case ignore string	0804F
DirXML-pbxName	PbxName	Local PBX name. The name of the pbxSite object you created in the Identity Vault. Use the same name as you put in the Site Object DirXML-pbxPbxName attribute.	Case ignore string	ProvoPBX
DirXML-nwoNode	PbxNode	Node where the new extension should be placed. Used only for cold jacks.	Integer	Building H East Building H West
DirXML-nwoPhoneType	PhoneType	Phone type recognized by the PBX. Used by the policies to set up the extension.	Case ignore string	8410
DirXML-nwoDoltNowFlag	DoltNowFlag	When set, do the action now and ignore the due date. This work order is also sent to the Publisher.	Boolean	false
DirXML-nwoStatus	Status	Status of the work order: pending, configured, resolved, canceled, error.	Case ignore string	pending
DirXML-nwoWorkOrderNumber	WorkOrderNumber	(Optional) Unique work order number assigned by a corporate work order system other than the Identity Vault, such as a work order database.	Integer	00001
DirXML-nwoAction	Action	Work order activity: install, setcor, modify (you can modify display name or objectID), move, remove, disable, disconnect. For Audix: add, change, remove.	Case ignore string	install

nwoWorkOrder Attributes (eDirectory Namespace)	PbxWorkOrder Attributes (Driver Namespace)	Description	Format	Sample Value
Description	Description	(Optional) Description of the work order. If the status is error, this contains a description of the error.	Case ignore string	New extension
DirXML-nwoExtensionType	ExtensionType	Type of extension to configure, such as DID or non-DID.	Case ignore string	DID
DirXML-nwoJack	Jack	The jack number. Use this to help link a user ID in the Identity Vault with an extension on the PBX.	Case ignoring string	12345
DirXML-nwoSendToPublisher	SendToPublisher	If set, this work order is sent to the Publisher for further action.	Case ignore string	false
DirXML-nwoDupExtension	DupExtension	Extension to use to duplicate attributes for new configured extension.	Numeric string	19876
DirXML-nwoPort	Port	Required if you have cold jacks. Port the extension is wired to. If the port is set, the driver uses that port, whether it is set up as hot or cold jacks. If you have hot jacks, this information is not necessary; you can leave the attribute blank.	Case ignore string	10C0611
DirXML-nwoRestrictionClasses	RestrictionClass	(Optional) Class of restriction used to grant specific calling access to the extension. This is used only on setcor. If no class of restriction is specified, the extension receives the default class of restriction as defined in the PBX.	Integer	01
DirXML-nwoRoom	Room	The room number.	Case ignore string	12345

The following table shows the pbxWorkOrder association:

Table B-4 *pbxWorkOrder Association*

Driver	State	Association
AvayaPBXDriver	Associated	/User'sCommonName/Date/Time

B.3 DirXML-pbxExtension Object

In the base configuration, the DirXML-nwoExtension object is specified in the Filter and in the Schema Mapping rule.

However, in some configurations, this object is used only as output from the driver and is immediately transformed into another object by the policies, so this kind of object might never be created in the Identity Vault. For example, nwoExtension objects are not created in the workforce tree configuration explained in [Chapter 7, “Workforce Tree Configuration,” on page 49](#). Instead, nwoExtension object information that comes from the driver shim is transformed into user object information by the policies. In configurations that never create nwoExtension objects in the Identity Vault, it is not necessary to include this kind of object in the Filter and the Schema Mapping rule.

The following table describes the attributes of the DirXML-nwoExtension object.

Table B-5 *pbxExtension Object Attributes*

pbxExtension Attributes (eDirectory Namespace)	pbxExtension Attributes (Driver Namespace)	Description	Format	Sample Value
Common Name	No mapping is necessary for this in the driver namespace	The naming attribute for eDirectory. In the sample configurations discussed in this guide, the value comes from the Extension attribute.	Case ignore string	12345
displayName	Name	The Display name of the extension this object represents.	Case ignore string	Doe, John
DirXML-nwoExtension	Extension	The extension number.	Case ignore string	12345
DirXML-nwoJack	Jack	The jack number. Use this to help link a user ID in the Identity Vault to an extension on the PBX.	Case ignore string	12345
DirXML-nwoRoom	Room	The room name or number. Use this to help link a user ID in the Identity Vault to an extension on the PBX.	Case ignore string	up to 15 characters: Monitorroom

pbxExtension Attributes (eDirectory Namespace)	pbxExtension Attributes (Driver Namespace)	Description	Format	Sample Value
Description	Description	Description of object.	Case ignore string	Extension configured.
DirXML-nwoObjectID	ObjectID	Entry ID of the object (such as a user object) that this phone number belongs to. In the sample configurations this is the entry ID, but it could be an ID of your choice, such as the employee ID.	Case ignore string	0000804F
DirXML-nwoPhoneType	PhoneType	Phone type recognized by the PBX.	Case ignore string	8410
DirXML-nwoRestrictionClass	RestrictionClass	Class of restriction, used to grant specific calling access to the extension. For example, you might define class 01 with a restriction that no international calls can be made.	Integer	01
(Port) DirXML-nwoPort	Port	Required if you have cold jacks. Port the extension is wired to. If the port is set, the driver uses that port, whether it is set up for hot or cold jacks. If you have hot jacks, this information is not necessary; you can leave the attribute blank.	Case ignore string	10C0611
DirXML-pbxName	PbxName	Local name for the PBX.	Single value case ignore string	ProvoPBX

The following table shows the association for the pbxExtension object.

Table B-6 *pbxExtension Object Association*

Driver	State	Association
AvayaPBXDriver	Associated	/Driver Name/Extension For example, /Avaya PBX Driver/12345

B.4 DirXML-pbxAudixSubscriber Object

Adding Audix support to the current Avaya driver requires the driver to access and manipulate PBX subscriber objects. This is performed by extending the eDirectory schema to include a DirXML-pbxAudixSubscriber object. For Audix support, the driver uses `add`, `change` and `remove` work order commands. Other important information includes:

- ♦ The Audix server is separate from the extension server, so you must specify a new site object for the Audix server.
- ♦ The Audix server supports different commands, so the Avaya driver supports an Audix type of access. The new DirXML-AccessType is `audix` and the new actions supported are `add`, `change`, `remove` and `AudixEmulate`.
- ♦ The Avaya driver does not synchronize manual changes made on the Audix server to the Identity Vault.
- ♦ The Avaya driver makes no attempt to see if there is a corresponding extension on an `add` command. The creation of extension and subscriber objects are independent of one another and can be done in any order.
- ♦ Because the Audix server is separate and very different than the extension server, separate work order commands are provided for Audix support. Any successful actions and errors resulting from work orders are reported back through the work order. Work order commands are `add`, `change`, and `remove`.
 - ♦ On an `add` command, the driver checks to see if the Subscriber Extension object is on the the PBX Audix server. If the object already exists, the driver generates an error and reports it back in the work order.

IMPORTANT: While adding a subscriber object for Avaya PBX 15, you must specify a password. IDM Avaya driver assigns `1001` as the default password.

- ♦ On a `change` command, the driver checks to see if the Subscriber object already exists. If it does not, the driver generates an error and reports it back through the work order.
- ♦ On a `delete` command, the driver checks to see if the subscriber object is already there. If it is not, the driver generates an error and reports it in the work order.

The following table describes the attributes of the DirXML-pbxAudixSubscriber object.

Table B-7 *pbxAudixSubscriber Object Attributes*

pbxAudixSubscriber Attributes (eDirectory Namespace)	pbxAudixSubscriber (Driver Namespace)	Description	Format	Sample Value
Class Name	No mapping is necessary for this in the driver namespace	The naming attribute for the Identity Vault. In the sample configurations discussed in this guide, the value comes from the Audix Subscriber attribute.	Case ignore string	12345

pbxAudixSubscriber Attributes (eDirectory Namespace)	pbxAudixSubscriber (Driver Namespace)	Description	Format	Sample Value
displayName	Name	The Display name of the extension this object represents.	Case ignore string	Doe, John
DirXML-nwoExtension	Extension	The extension number.	Case ignore string	12345
DirXML-pbxName	PbxName	Local name for the PBX.	Single value case ignore string	ProvoPBX

B.5 User Objects and Their Identity Manager Associations

The driver can be configured to create work orders based on creation of or changes to a User object, as discussed in [Chapter 7, “Workforce Tree Configuration,” on page 49](#).

Two existing attributes of a User object are used in the workforce tree configuration. Because these attributes already exist in the eDirectory schema, the schema extension that is part of the driver installation does not make any changes to User objects.

User objects receive an Identity Manager association for the driver when they are sent to the Subscriber channel. The table below shows the value of the association for a User object.

Table B-8 *User Object Association*

Driver	State	Association
AvayaPBXDriver	Associated	/PBXName/WorkOrderCommonName/Time

You might expect that the association for a user object for the Avaya PBX Driver is the phone extension. However, this is not the case. The association for the user object is made before a work order is performed. For a work order to install a new extension, the number of the new phone extension is not known until the work order is performed, so the extension can't be used.