Novell Identity Manager Driver for Avaya* PBX

1.0

www.novell.com

IMPLEMENTATION GUIDE

March 30, 2005

Novell®

Legal Notices

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

www.novell.com

Implementation Guide: Identity Manager Driver for Avaya PBX
March 30, 2005

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

Novell Trademarks

Novell is a registered trademark of Novell, Inc. in the United States and other countries.
SUSE is a registered trademark of SUSE AG, a Novell business.

Third-Party Materials

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

Contents

About This Guide 7

1 Overview 9

The Role of the Identity Manager Driver for Avaya PBX . . . 9
Benefits of the Driver . . . 10
Basic Functionality of the Driver . . . 11
New Objects Used by the Driver . . . 11
Overview of Driver Functionality . . . 13
What is Emulation Mode and Why Should I Use it? . . . 15
What the Driver Does in the PBX . . . 16
Explanation of DoItNow and SendToPublisher Flags . . . 17
Assumptions about the PBX . . . 18
Support for Manual Changes in the PBX . . . 19
What's Different about This Driver? . . . 20

2 Planning 21

Planning Issues for All Configurations . . . 21
Planning for User Provisioning (Workforce Tree) Implementations . . . 23
Planning for Work Order System Implementations . . . 24
Planning Driver and Replica Placement on Your Servers . . . 24

3 Installation 25

Before You Install . . . 25
Identify or Create Containers for New Objects . . . 25
Create PBX Site Objects . . . 25
Use Emulation . . . 25
Prerequisites . . . 26
Installation . . . 26
Creating a Driver Object . . . 27
Setting the Driver Parameters for Emulation . . . 28

4 Base Configuration 31

How the Base Configuration Works . . . 31
How the Subscriber Channel is Configured . . . 34
How the Publisher Channel is Configured . . . 35
Planning for the Base Configuration . . . 36
Setting Up the Base Configuration . . . 36

5 Workforce Tree Configuration 37

How the Workforce Tree Configuration Works . . . 37
How the Driver Uses ObjectID to Identify and Update Users . . . 40
How the Subscriber Channel is Configured . . . 41
How the Publisher Channel is Configured . . . 42
User Events That Can Trigger a Work Order . . . 43
Planning for the Workforce Tree Configuration . . . 43
Setting Up the Workforce Tree Configuration . . . 43

Using Log/Reporting Functionality to Report Warnings . . . 43

6 Work Order Database Configuration 45

How You Could Configure a Work Order Database Configuration . . . 46
How To Configure the Subscriber Channel . . . 48
How To Configure the Publisher Channel . . . 49
About Work Order Systems . . . 49
Planning for the Work Order Database Configuration . . . 49
Setting Up the Work Order Database Configuration . . . 49
Using Log/Reporting Functionality to Report Warnings . . . 49

7 Managing the Identity Manager Driver for Avaya PBX 51

Changing How Often the Driver Performs Work Orders . . . 51
Changing the Location of PBX Site, Work Orders, User, and Extension Objects . . . 52

8 Managing Existing Users 53

A Schema for PBX Management 55

pbxSite Object . . . 55
DirXML-nwoWorkOrder Object . . . 57
DirXML-pbxExtension object . . . 60
User Objects and their DirXML Associations . . . 61

About This Guide

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

The Identity Manager Driver 1.0 for Avaya PBX lets you use eDirectory to centrally manage PBX extensions and work orders and keep user phone numbers up-to-date.

The guide contains the following sections:

- Chapter 1, “Overview,” on page 9
- Chapter 2, “Planning,” on page 21
- Chapter 3, “Installation,” on page 25
- Chapter 4, “Base Configuration,” on page 31
- Chapter 5, “Workforce Tree Configuration,” on page 37
- Chapter 6, “Work Order Database Configuration,” on page 45
- Appendix A, “Schema for PBX Management,” on page 55

This book explains what the driver does and what the driver can be customized to do by describing three sample configurations that are provided with the driver for instructional purposes.

Additional Documentation

For documentation on using Nsure™ Identity Manager and the other drivers, see the Identity Manager Documentation Web site (http://www.novell.com/documentation/lg/dirxml20).

Documentation Updates

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

Documentation Conventions

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

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

User Comments

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

1 Overview

The Identity Manager Driver 1.0 for Avaya PBX lets you use eDirectory 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, and updating phone number data for User objects in eDirectory.

This product can be an important part of a provisioning solution.

In this section:

- "The Role of the Identity Manager Driver for Avaya PBX" on page 9
- "Benefits of the Driver" on page 10
- "Basic Functionality of the Driver" on page 11
- "What's Different about This Driver?" on page 20

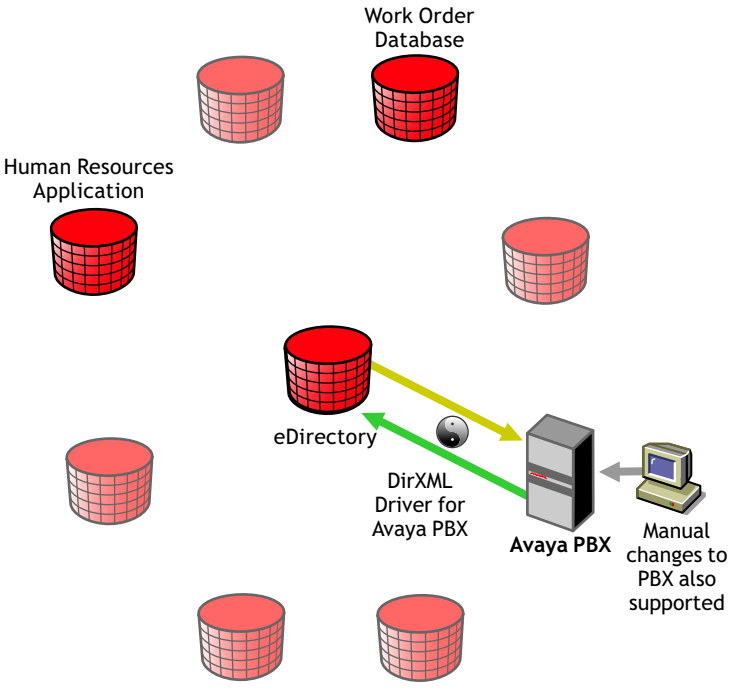
The Role of the Identity Manager Driver for Avaya PBX

The Identity Manager Driver for Avaya PBX communicates with the PBX and carries out work orders by logging in just as a PBX administrator would. The driver uses information from eDirectory as well as customizable settings and policies to configure the extension correctly. After configuration is completed, the driver publishes the new phone number to eDirectory, 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 eDirectory using the Identity Manager Driver for Avaya PBX.

The terminal shown in the diagram represents the fact 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 using Identity Manager.



Benefits of the Driver

Benefits of using the driver include the following:

- Eliminates redundant data entry.

  Without the driver, if you use a work order system 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 eDirectory 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 eDirectory 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 change has been made to a PBX extension manually, 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 eDirectory.

Basic Functionality of the Driver

In this section:

- "New Objects Used by the Driver" on page 11
- "Overview of Driver Functionality" on page 13
- "What is Emulation Mode and Why Should I Use it?" on page 15
- "What the Driver Does in the PBX" on page 16
- "Explanation of DoItNow and SendToPublisher Flags" on page 17
- "Assumptions about the PBX" on page 18
- "Support for Manual Changes in the PBX" on page 19

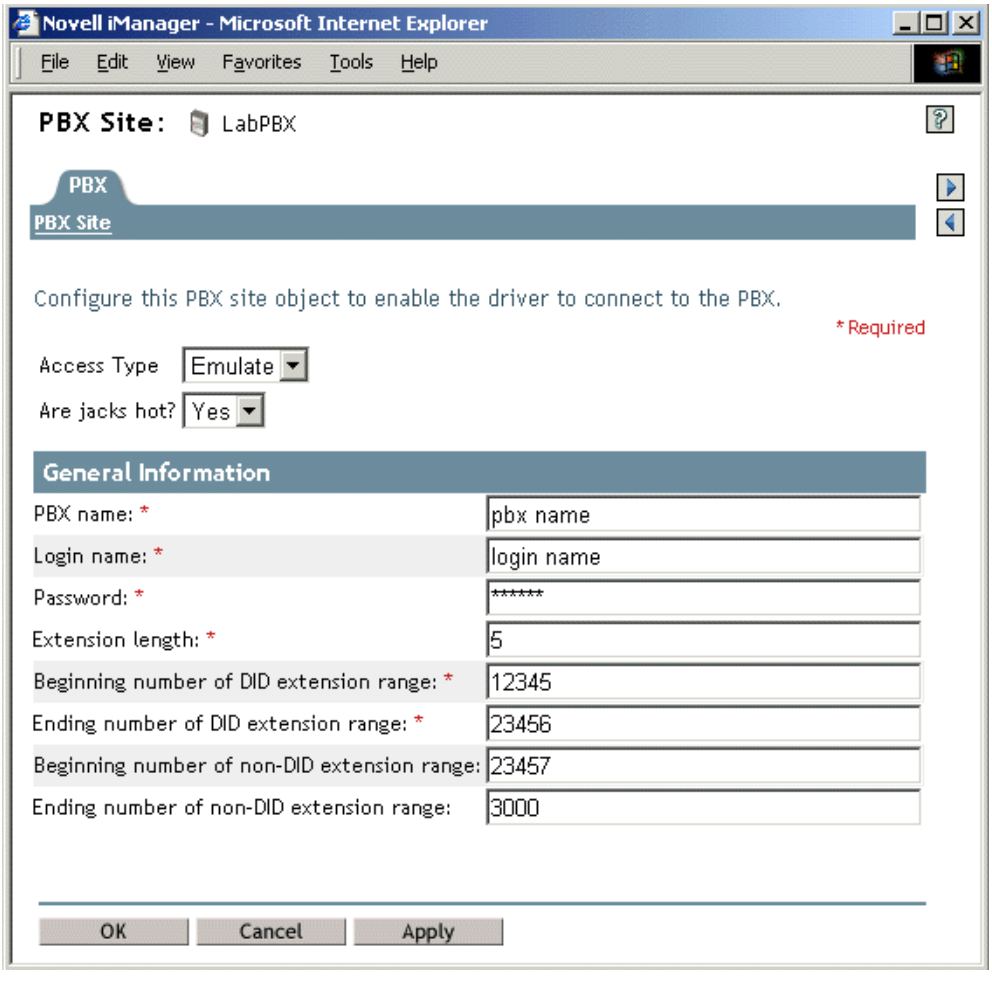
New Objects Used by the Driver

Using three new object classes in eDirectory, the Identity Manager Driver for Avaya PBX performs work orders, records the results, and provides extension information. The new objects are

- DirXML-pbxSite to represent the PBX.

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

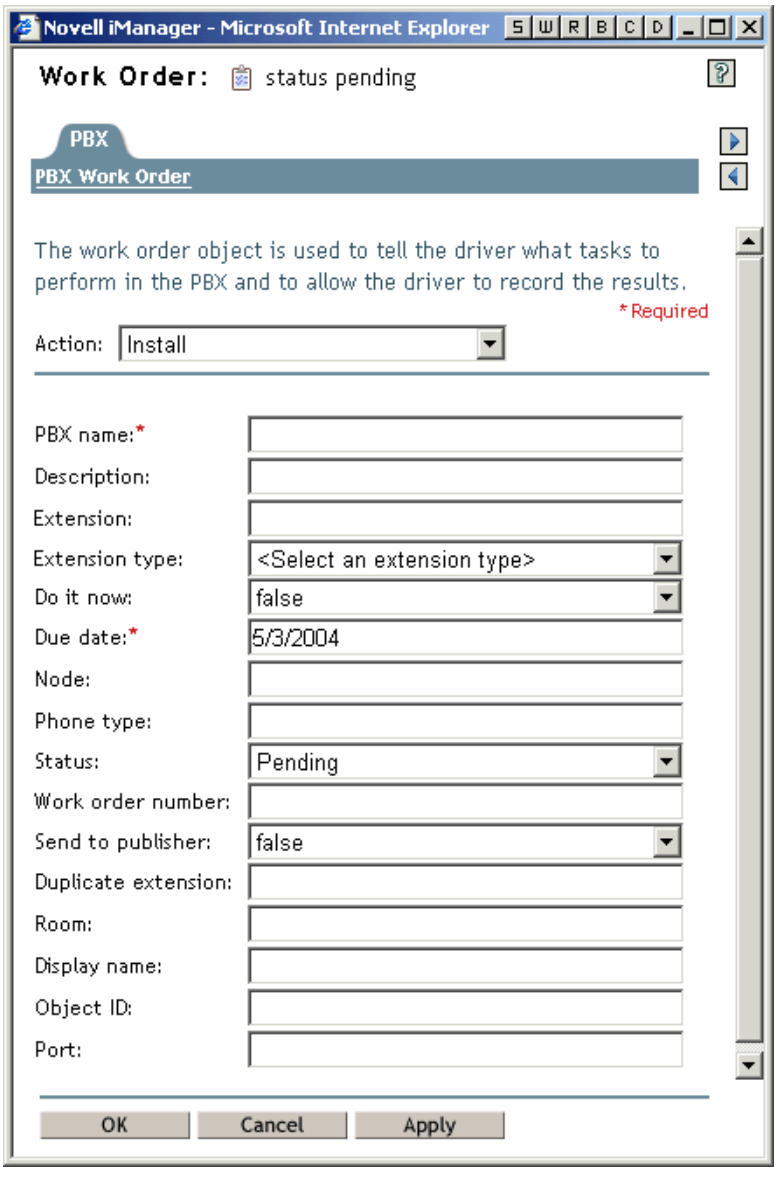
  An iManager plug-in is provided to help you set up these sites, PBX Utilities > Site Management. The following figure shows an example of the interface for setting up a PBX site.



- DirXML-nwoWorkOrder to represent 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, PBX Utilities > Work Order Management. The following figure shows an example of the interface for creating a work order.



- DirXML-pbxExtension to represent 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.)

For a description of the schema for these objects, see Appendix A, “Schema for PBX Management,” on page 55.

Overview of Driver Functionality

When configuring the driver, you have the option to run the driver in emulation mode. While 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 15.

The driver shim itself functions in the following basic way:

1. When the driver starts up, 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 eDirectory representing the change.

   See “Support for Manual Changes in the PBX” on page 19.

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.

   NOTE: For many drivers, the Subscriber performs changes in the third-party application in response to events in eDirectory. However for this driver, the Publisher is the agent which 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 “DoIt Now.” 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.

   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.

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 install work orders, the Publisher also sends a DirXML-pbxExtension object to eDirectory 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 eDirectory with the new extension in the DirXML-nwoExtension attribute.

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 sent to the Subscriber channel, which sends them to the Publisher channel to be performed.

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:

- Cause a work order to be created when a new user is added to eDirectory, to automate the assignment of an extension to a new employee. This could be further automated by using an

additional Identity Manager driver to automatically create users in eDirectory 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 eDirectory. 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 orders from another application to be synchronized to eDirectory and performed by the Avaya PBX driver.
- Transform a new DirXML-pbxExtension object coming from the driver into a modification of a User object, so you can update the User object with the new extension when a work order is completed.
- 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 4, “Base Configuration,” on page 31
- Chapter 5, “Workforce Tree Configuration,” on page 37
- Chapter 6, “Work Order Database Configuration,” on page 45

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 testing environment. When you are comfortable with the test environment, you can move into production. Most companies, however, don’t have PBX systems available for 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. But instead of connecting to the PBX, the driver is able to add, modify, and delete extension 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.

How Does Emulation Work?

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 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 it was created.

How Do I Configure Emulation Mode?

When configuring the driver's parameters, there are five parameters (in the Publisher settings) that you need to specify in order configure the driver for emulation. They are:

- The IP address of the LDAP host you want to use for emulation. This can be eDirectory or any other LDAP directory.

- The LDAP port that the host uses for LDAP.
- The DN of the login user on the LDAP host you are using.
- The password of that user.
- The DN for the container that will hold the emulated extensions.

For more information on these settings, see "Setting the Driver Parameters for Emulation" on page 28.

What the Driver Does in the PBX

At the most basic level, these are the actions that the driver can perform in the PBX: install, disable, enable, move, modify, setcor, 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 will assign 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 will assign 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 an extension in one of the following ways:
  - Deactivating it in one location and activating it in another location within the same PBX system.
  - With the use of custom style sheets, deactivating it in one PBX system and activating 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: Delete an extension from the PBX.
- Set values such as the following, based on data you provide in the work order:
  - The due date to complete the desired action.

    If the DoItNow flag is set, the driver will perform the action immediately.

    The driver is configured so that at a specified time once a day it will perform 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).
- Perform the work order either immediately (using the DoItNow flag) or on a specified due date.

  You can set the driver to check for work orders at a certain time each day or at a specified polling interval.

- 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.

  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 if necessary, and the driver will be 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 employee ID is inserted the definition 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 employee ID into the definition field, the driver will discover that change and automatically update the user object with the new extension.

  See "Support for Manual Changes in the PBX" on page 19.

Explanation of DoItNow and SendToPublisher Flags

DoItNow Flag

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 eDirectory by an administrator (as in Chapter 4, "Base Configuration," on page 31) or an application (like the solution described in Chapter 6, "Work Order Database Configuration," on page 45).

The scenario where this flag is useful is one in which the work order is triggered by another eDirectory event through the use of a policy or style sheet, and the work order does not yet exist in eDirectory as an object. This kind of scenario is described in Chapter 5, "Workforce Tree Configuration," on page 37.

Assumptions about the PBX

The PBX is the Public Branch Avaya PBX or corporate phone system. The PBX may 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 will be physically cross-connected to a PBX digital port.

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

The Identity Manager Driver for Avaya PBX 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 18, below.

In this section:

- “Voice Jacks” on page 18
- “Querying the PBX” on page 19

Voice Jacks

The behavior of the driver will depend 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

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 will be followed. (See “Cold Jacks” on page 18.)

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 will assign the first available number in numeric order. When the phone is delivered and plugged in, punching in a code can activate the extension on that phone by automatically assigning the port.

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 will assign 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.

Querying the PBX

Because of the limitations of the PBX system, the driver queries the PBX only for extensions.

The PBX doesn't support queries on display name or the field containing a user identifier. To find those pieces of data when querying the PBX, you would have to create a custom style sheet 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.

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 supported as well.

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 eDirectory to reflect changes that have been made manually, as much as possible.

Here are 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.

Change made manually in the PBX	Can the driver detect the change?	Can the driver update eDirectory in response to the change?	How the driver can update eDirectory
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.)
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. It's preferable to use two work orders, one to disconnect the extension for the first user, and another to install the extension for a different user.
Setcor	No	No update to eDirectory is necessary.	
Disable an extension	Yes	Yes	The driver
Disconnect an extension	Yes The driver detects that this extension is not connected and is available for reuse.	No	In the base configuration, the driver can delete the nwoExtension object that represented the extension. It can do this by querying for the nwoExtension that has the corresponding extension number in the Extension attribute. ???(In the workforce tree configuration, the driver updates all users who have this extension in their phone list, 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 eDirectory for User objects that have that extension.)

What's Different about This Driver?

The Identity Manager Driver for Avaya PBX is different from some other drivers in the following ways:

- The driver maps 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 eDirectory must be connected to extensions in the PBX using the Object ID in the work order. See Chapter 8, "Managing Existing Users," on page 53.

- The ObjectID in the work order (which might be user object entry ID, employee ID, etc.) 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 needs to be a unique way to identify the user, and a style sheet 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 a DirXML association for this driver, in most cases the user is not be updated unless the ObjectID has also been entered correctly. In fact, a user object can be updated if the ObjectID is correct, even if the user does not yet have a DirXML association for this driver. See Chapter 8, "Managing Existing Users," on page 53.

- 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 eDirectory. However for this driver, the Publisher is the agent which 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 "DoIt Now."

  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.

  One 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 Migrate into NDS.

  See Chapter 8, "Managing Existing Users," on page 53.

2 Planning

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

In this section:

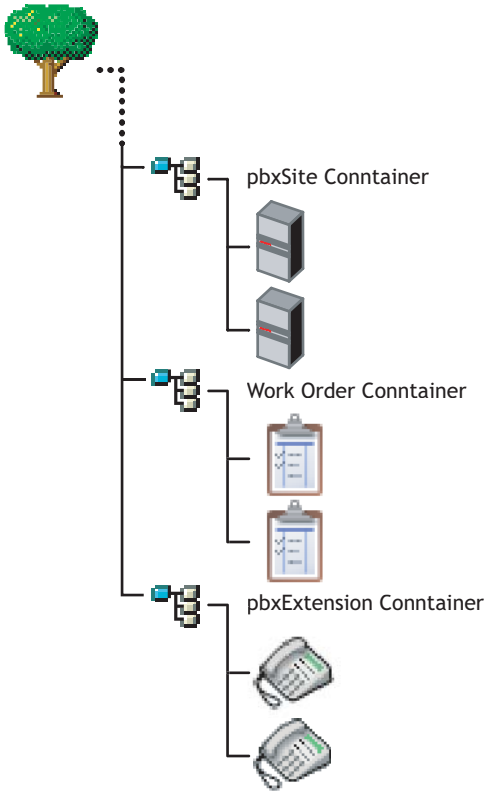
- "Planning Issues for All Configurations" on page 21
- "Planning for User Provisioning (Workforce Tree) Implementations" on page 23
- "Planning for Work Order System Implementations" on page 24
- "Planning Driver and Replica Placement on Your Servers" on page 24

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 1 Example of Containers for New Objects



- Which container do you want to use to hold 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 "pbxSite Object" on page 55.

- Which container do you want to use to hold nwoWorkOrder objects?

  These are the objects used to tell the driver which tasks to perform in the PBX. For more information, see "DirXML-nwoWorkOrder Object" on page 57.

- Are you going to use nwoExtension objects, and if so, which container do you want to use to hold 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 "DirXML-pbxExtension object" on page 60.

  Depending on your implementation, you might not need to use nwoExtension objects. Instead, you can configure your style sheets 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 5, "Workforce Tree Configuration," on page 37).

- Gather all the information about your PBX sites that you'll 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'll need to provide in the pbxSite objects, see "pbxSite Object" on page 55.

- 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 "DirXML-nwoWorkOrder Object" on page 57.

  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.

- What are the duplicate extensions for each phone type? You'll need to reproduce that mapping in the style sheets.

  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 style sheets for the sample configurations demonstrate how to map the phone type to the dupe extension.

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

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

  Of course, if the work order is marked DoItNow, the driver will perform it immediately and won'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 "What is Emulation Mode and Why Should I Use it?" on page 15 for more information.

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 "Workforce Tree Configuration" on page 37.)

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

  This number is entered in the work order as the ObjectID attribute, and you can use a custom style sheet 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, such as Social Security number. 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 work order for any work orders entered manually or in a work order database.

- How will you specify the user's location so that the style sheets can determine where in the PBX to configure the extension?

  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. which can so that a single PBX site might be comprised of cabinets at more than one campus.

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

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

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

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

  The phone type must be assigned by the style sheets automatically, 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 could 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 could 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.

- Using the driver with style sheets that update user phone information, you can make sure phone information is correct going forward. Do you also want to make an effort to clean up existing phone information for existing user objects?

  This kind of an effort would be largely manual, but you might be able to do some automation by creating custom tools to compare existing user information with information in the PBX. You can get a list of what exists in the PBX using the Migrate into NDS feature. See Chapter 8, “Managing Existing Users,” on page 53.

- Do you want the driver to be able to manage all existing users, or just provision new users? If you want the driver to be able to manage existing users, some manual work will be necessary.

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

  See Chapter 8, “Managing Existing Users,” on page 53.

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

  Determine what information from a user object you want the style sheets 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 would be to create eDirectory objects for those rooms or other entities, so that they can be identified by an ObjectID. Then you could customize the driver to update them the same way it can update user objects.

Planning for Work Order System Implementations

This section explains the additional planning issues you need to consider when creating an implementation of the driver for use with a work order database. (See “Work Order Database Configuration” on page 45.)

- 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.

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.

3 Installation

In this section:

- "Before You Install" on page 25
- "Prerequisites" on page 26
- "Installation" on page 26
- "Creating a Driver Object" on page 27

Before You Install

Identify or Create Containers for New Objects

You will need to specify 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 sample base configuration uses these objects, but usually in a production environment you would transform events for DirXML-pbxExtension objects into events for User objects.

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 also described in "Planning Issues for All Configurations" on page 21 and Appendix A, "Schema for PBX Management," on page 55.

Create PBX Site Objects

- Install Identity Manager and the driver for Avaya PBX, as explained in Chapter 3, "Installation," on page 25
- Create PBX Site objects, one for each PBX
- Create an instance of the driver using one of the sample configurations

Use Emulation

This driver is different from other Identity Manager drivers because it provides Emulation capabilities. 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 "What is Emulation Mode and Why Should I Use it?" on page 15 for more information before installing the driver.

Prerequisites

- ❑ Nsure™ Identity Manager 2.0.1 with the latest patches and product updates
- ❑ Software required by Identity Manager 2.0.1

  For example, the correct version of iManager.
- ❑ Avaya PBX software versions 9, 10 or 11
- ❑ eDirectory administrator username and password, so you can log in during the installation to allow schema extension. The schema extensions are described in Appendix A, "Schema for PBX Management," on page 55.
- ❑ An eDirectory server with a master or read/write replica of all the objects that will be affected by PBX changes.
- ❑ The items in "Before You Install" on page 25.

Installation

You need to install the following:

- The driver shim (Avayashim.jar), on the server where Identity Manager is installed or on a server where you will use Remote Loader to run the driver.
- The driver policies for the driver (AvayaPBXShip.XML and AvayaUser.xml) and the iManager plug-ins for using work orders, on the iManager server.

If Identity Manager and iManager are on the same server, you only need to run the install program once.

1. Download the driver for Avaya PBX.
2. Unzip the download package in a location of your choice on the server.
3. Run the installation program for your platform:

   On Windows: Double-click avayainstall.exe to run the installation program

   On NetWare: Enter java -jar sys:/*path* avayainstall.jar

   On Linux: Run avayainstall_linux.bin

   On Solaris: Run avayainstall_solaris.bin

   On AIX: Run avayainstall_aix.bin
4. Review the Introduction page, which contains a reminder that iManager 2.*x* or later is a prerequisite. Click Next.
5. On the License Agreement page, review the agreement. If you accept, select I Accept the Terms of the License Agreement and click Next.
6. On the Install Components page, select the items you want to install.
   - Nsure Identity Manager Driver for Avaya PBX

     Select this item to install the driver on the same server as Identity Manager.
   - Avaya Driver Policies

     Select this item to install the driver policy files and the plug-ins for using work orders, on an iManager server.

7 If installing to the Identity Manager server, you will see the Schema Extension page. On the Schema Extension page, specify the administrator username, in LDAP format (for example: cn=admin,o=novell). Specify the administrator password, then click Next.

This login information is necessary so that the installation can update the schema for your eDirectory tree. The schema updates that are made are described in Appendix A, "Schema for PBX Management," on page 55.

After this step, there might be a delay as the installation is prepared, and a Please Wait message is displayed.

8 On the Installation Summary page, review the information. Click Install.

A message is displayed indicating that the installation is being completed.

9 On the Installation Complete page, review the summary and click Done.

10 Review the information about possible configurations in the following chapters:

- Chapter 4, "Base Configuration," on page 31
- Chapter 5, "Workforce Tree Configuration," on page 37
- Chapter 6, "Work Order Database Configuration," on page 45

11 Continue with "Creating a Driver Object" on page 27.

Creating a Driver Object

To use the driver, you must create a driver object in eDirectory and configure the driver policies. Two driver configuration files are provided as samples:

- AvayaPBXShip.XML

  Use this sample configuration to set up a basic configuration of the driver, described in Chapter 4, "Base Configuration," on page 31.

- AvayaUser.xml

  Use this sample configuration to set up a configuration of the driver that updates users, described in Chapter 5, "Workforce Tree Configuration," on page 37.

Specify the following information when importing one of the sample configurations.

Import Prompt	Description
Driver name	The name of the driver contained in the driver configuration file is AvayaPBX. Specify the actual name you want to use for the driver.
Site Container	Enter, or browse for, the container to hold PBX site related objects. For example: PbxSite.MyOrganization
Users Container	(For workforce tree configuration only.) Enter, or browse for, the container to hold user related objects. For example: Users.MyOrganization
Work Orders Container	Enter, or browse for, the container to hold work order related objects. For example: WorkOrders.MyOrganization

Import Prompt	Description
Extensions Container	Enter, or browse for, the container to hold extension related objects, for example: Extensions.MyOrganization
Polling Method	Select the method the driver should use to poll for changes. By Interval means the driver will poll every so often for a given interval specified in minutes. By Time means the driver will poll only at a specified time. Or, you may choose to do both.
Driver is Local/Remote	Do you want this driver to run locally, or remotely with the Remote Loader service? Configure the driver for use with the Remote Loader service by selecting Remote, or select Local to configure the driver for local use.
Poll Interval (Minutes)	If you choose to use a polling interval, when you click Next this prompt is displayed. Enter a poll interval for the driver to use. For example, 90.
Poll Time	If you choose to use a polling time, when you click Next this prompt is displayed. Select a poll time for the driver to use. For example, 12:00 AM.
Remote Host Name and Port	For remote driver configuration only. If you specify Remote in the Driver is Local/Remote prompt, when you click Next this prompt is displayed. Enter the Host Name or IP Address and Port Number where the Remote Loader Service has been installed and is running for this driver. The Default Port is 8090.
Driver Password	For remote driver configuration only. If you specify Remote in the Driver is Local/Remote prompt, when you click Next this prompt is displayed. The Driver Object Password is used by the Remote Loader to authenticate itself to the Identity Manager server. It must be the same password that is specified as the Driver Object Password on the IDM Remote Loader.
Remote Password	For remote driver configuration only. If you specify Remote in the Driver is Local/Remote prompt, when you click Next this prompt is displayed. The Remote Loader password is used to control access to the Remote Loader instance. It must be the same password that is specified as the Remote Loader password on the IDM Remote Loader.

Setting the Driver Parameters for Emulation

If you choose to use the Emulation mode, you should specify the following parameters when importing the driver.

Import Prompt	Description
IP Address for LDAP Host for Emulation	For emulation, the driver needs to access an LDAP host (eDirectory or any other host). Enter the IP address of the LDAP server.
Port of LDAP Host	Enter the port number of the LDAP server used for emulation.
DN of the Login User for Emulation	For emulation, the driver needs to know which username has access to the LDAP server. This user must have rights to read/write to the extension containers on the server. Enter the name of the user for the LDAP server.

Import Prompt	Description
User Password for Emulation	Enter the password of the user who accesses the LDAP server.
Extension Container DN for Emulation	Enter the DN of the container where the driver will add or modify extension objects (this container must already exist on the LDAP server.)

4 Base Configuration

The base configuration (AvayaPBXShip.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.

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 style sheets 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 5, “Workforce Tree Configuration,” on page 37.

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 will want to 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 IDM Driver for JDBC. This kind of implementation is described in Chapter 6, “Work Order Database Configuration,” on page 45.

In this section:

- “How the Base Configuration Works” on page 31
- “Planning for the Base Configuration” on page 36
- “Setting Up the Base Configuration” on page 36

How the Base Configuration Works

Figure 2 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 with the task “Install.” In this example, the flags are set on the work order for SendToPublisher and DoItNow.
2. The Subscriber channel is notified by the DirXML Engine that a new nwoWorkOrder object has been created in eDirectory.

3. The Subscriber creates the DirXML association for the object, and sends the work order to the Publisher.

   (If the SendToPublisher flag were not set, the Subscriber would not "wake up" Publisher channel to send the work order to the Publisher. Instead, the Publisher would read 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 were not set, the Publisher wouldn'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 2 shows an illustration of generally how the base configuration works. Figure 3 is a flowchart of how the configuration works for an Install work order with the DoItNow flag set to true, and Figure 4 is a flowchart showing how the configuration works for an Install work order with the DoItNow flag not set.

Figure 2 Graphical Representation of Base Configuration

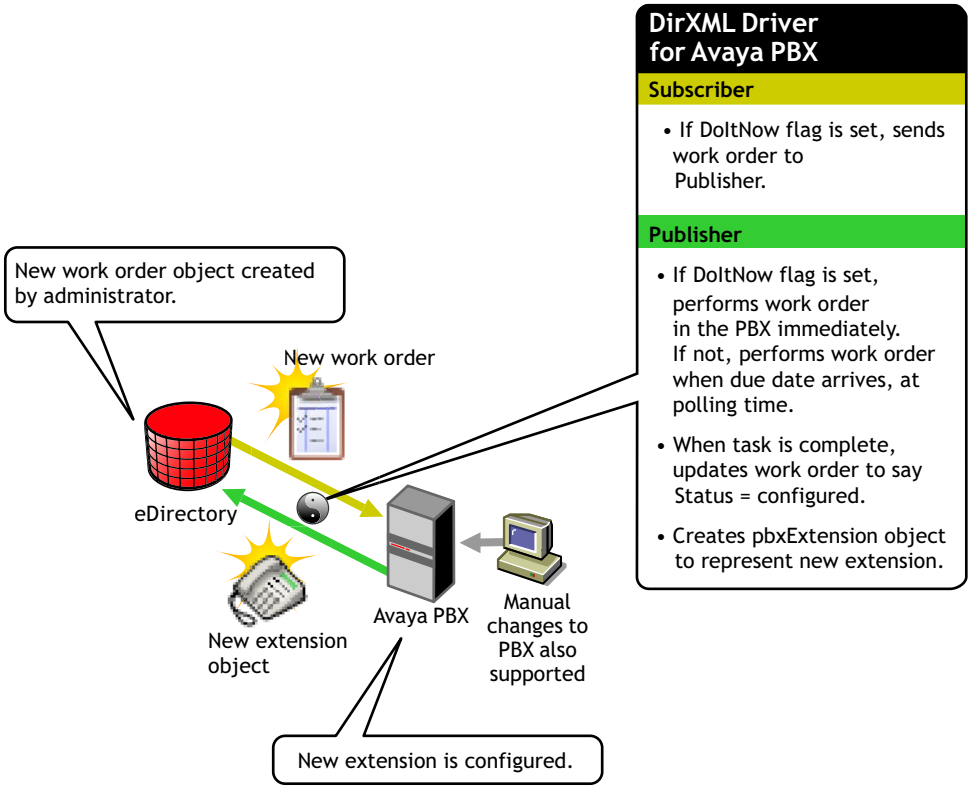


Figure 3 Flowchart of Base Configuration for Work Order with DoItNow Flag Set to True

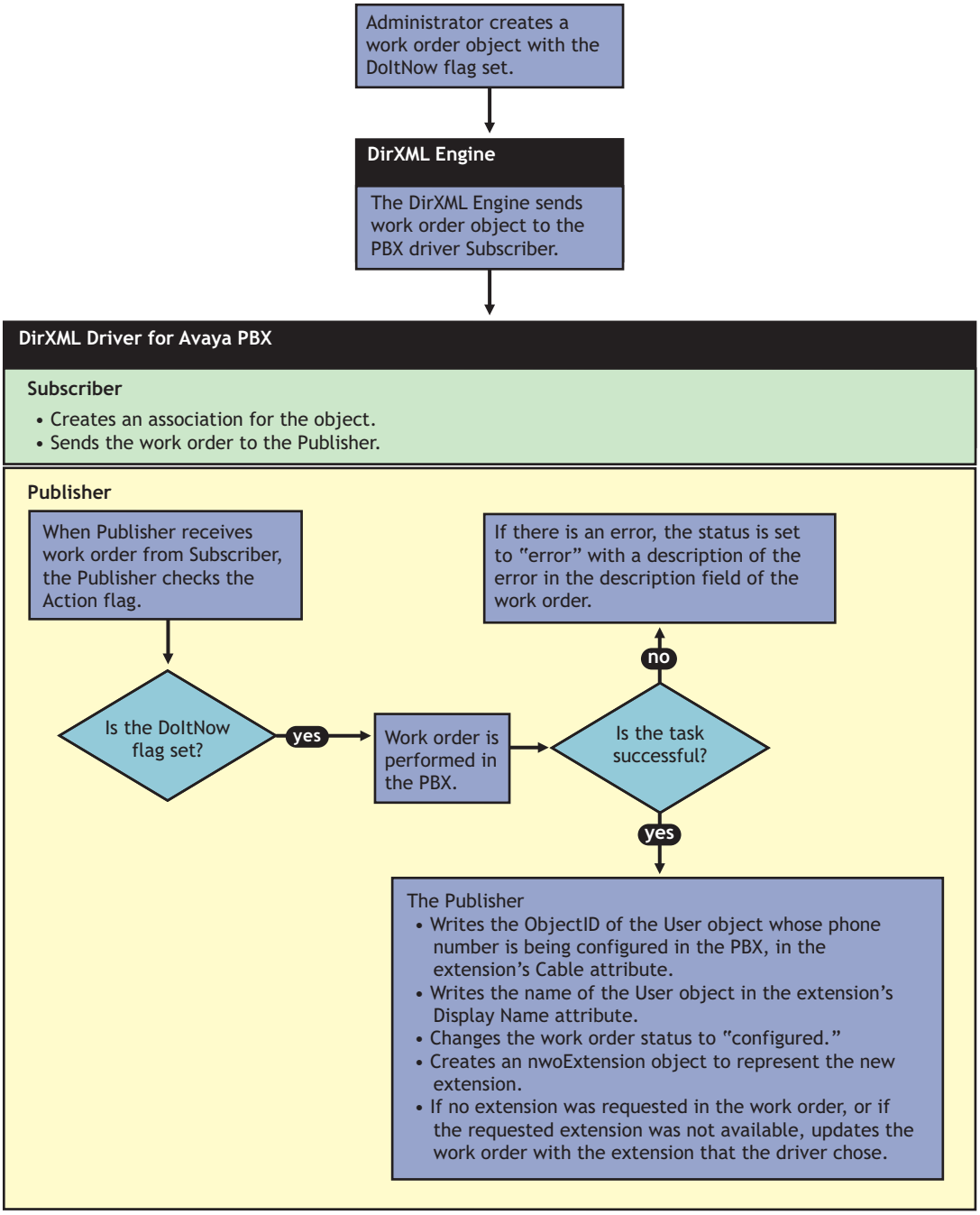
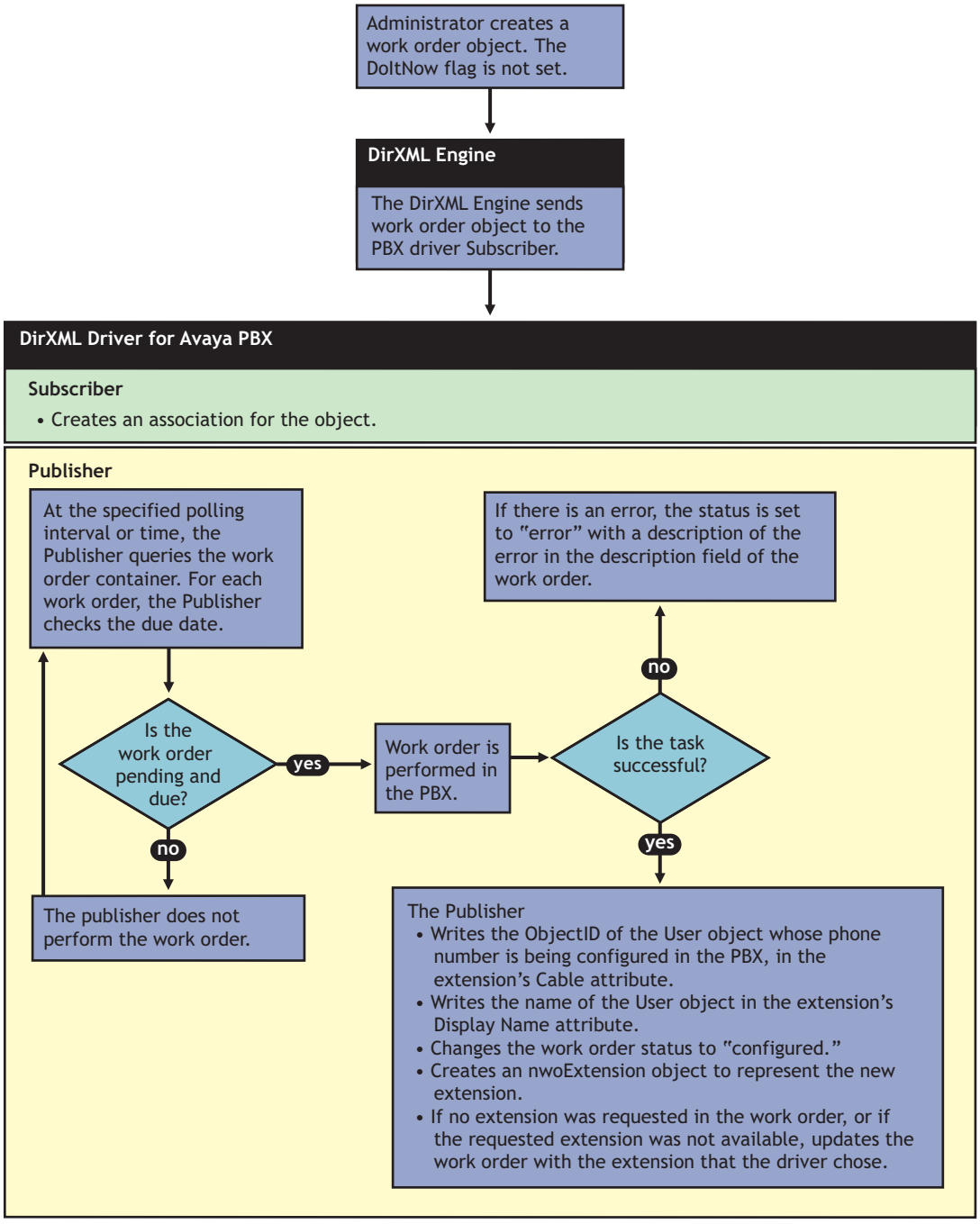


Figure 4 Flowchart of Base Configuration for Work Order with DoItNow Flag Not Set



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 eDirectory, 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 eDirectory. However for this driver, the Publisher is the agent which performs work orders in the PBX.

Rule or Style Sheet	What it does
Subscriber Filter	Allows only events for nwoWorkOrder objects to be processed.
Event Transformation	Not used in sample configuration.
Matching Rule	Not used in sample configuration.
Create Rule	Contains rules only for work order objects. Requires values for the following attributes on a work order object. ◆ PBXName ◆ nwoStatus ◆ nwoSendtoPublisher ◆ nwoDoItNow ◆ nwoAction ◆ nwoDisplayName If they are not present, the work order is not sent to the Publisher, and therefore is not performed in the PBX. For a description of these attributes, see "DirXML-nwoWorkOrder Object" on page 57.
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 DoItNow flag is set to True.
Command Transformation	Not used in sample configuration.
Schema Mapper	Maps eDirectory namespace to PBX namespace. Handles mapping of work orders, extensions, and PBX site objects.
Output Transformation	Not used in sample configuration.

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 is the one that performs tasks in the Avaya PBX, rather than the Subscriber. For a general overview of what the Publisher does, see "Overview of Driver Functionality" on page 13.

Rule or Style Sheet	What it does
Input Transformation	Not used in sample configuration.
Schema Mapper	Maps PBX namespace to eDirectory namespace. Handles mapping of work orders, extensions, and PBX site objects.
Event Transformation	Not used in sample configuration.

Rule or Style Sheet	What it does
Publisher Filter	Allows only events for nwoWorkOrder and pbxExtension objects to be processed.
Matching Rule	Not used in 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 sample configuration.

Planning for the Base Configuration

See "Planning" on page 21.

Setting Up the Base Configuration

Follow the instructions in "Installation" on page 25, and when you are creating a driver object, use the sample configuration file named AvayaPBXShip.XML.

5 Workforce Tree Configuration

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.

Like the base configuration, the workforce tree configuration is meant to be instructional, used 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, in the workforce tree configuration, rules are in place to maintain the relationships between users in the workforce tree and extensions in the PBX. For example, changes to users will 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 will want to 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 intended to be used is explained in Chapter 6, "Work Order Database Configuration," on page 45.

In this section:

- "How the Workforce Tree Configuration Works" on page 37
- "Planning for the Workforce Tree Configuration" on page 43
- "Using Log/Reporting Functionality to Report Warnings" on page 43

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 5 **Graphical Representation of Workforce Tree Configuration**

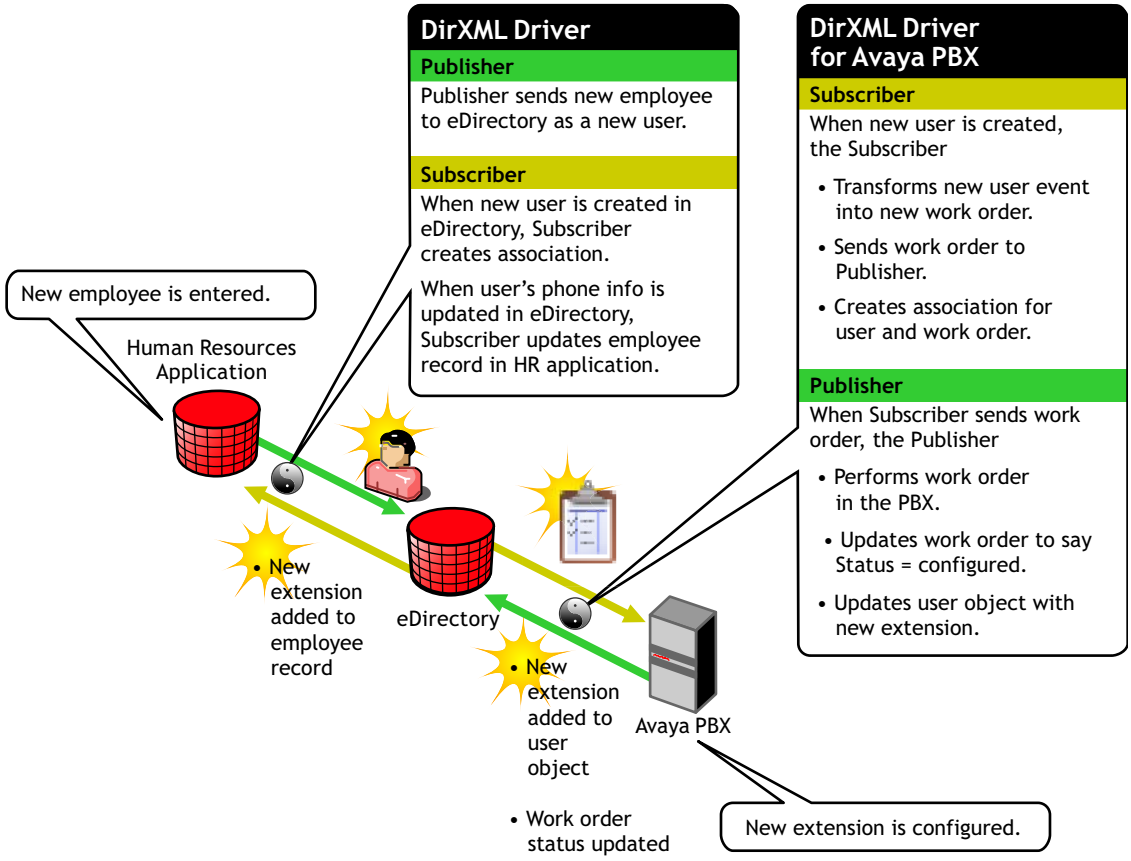
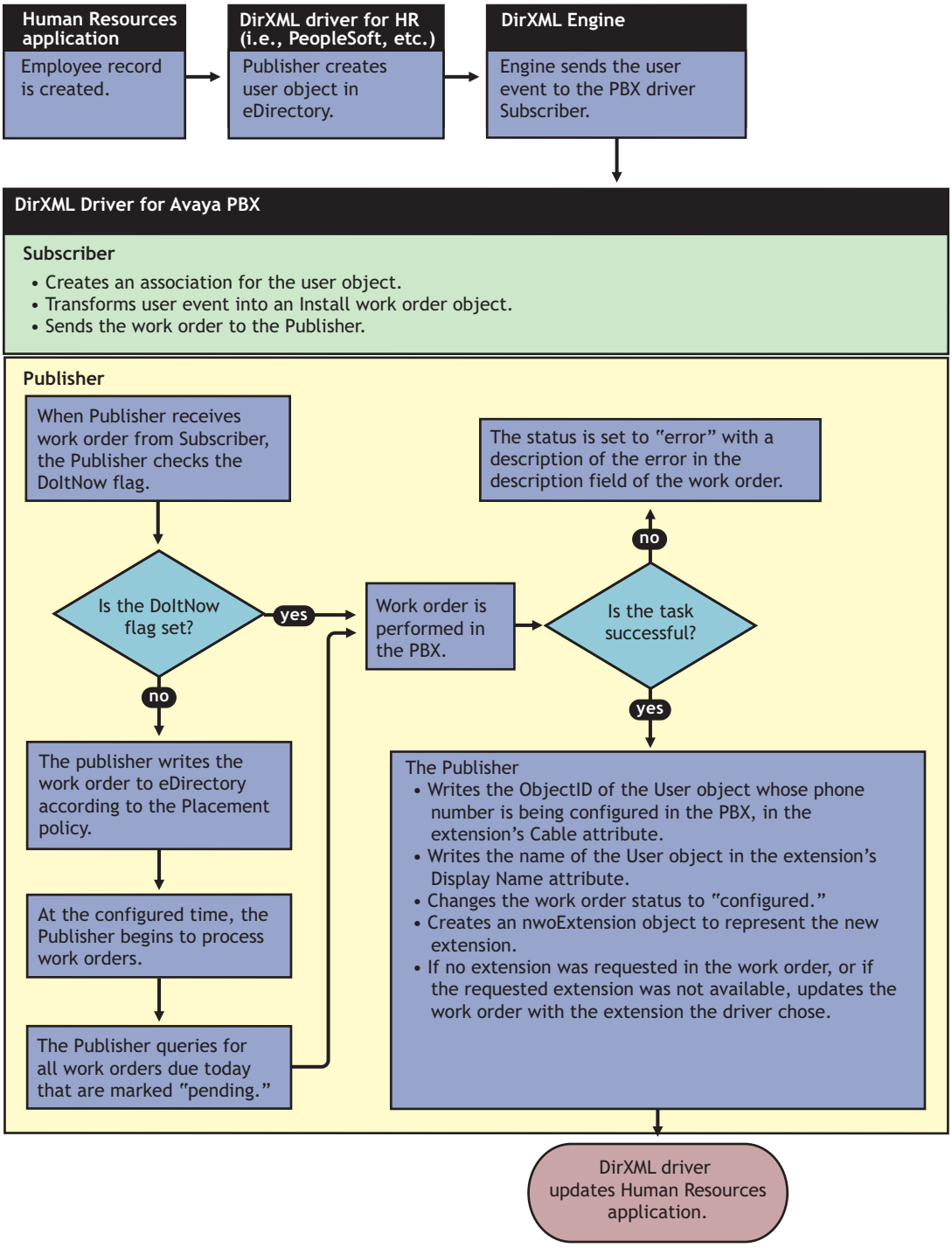


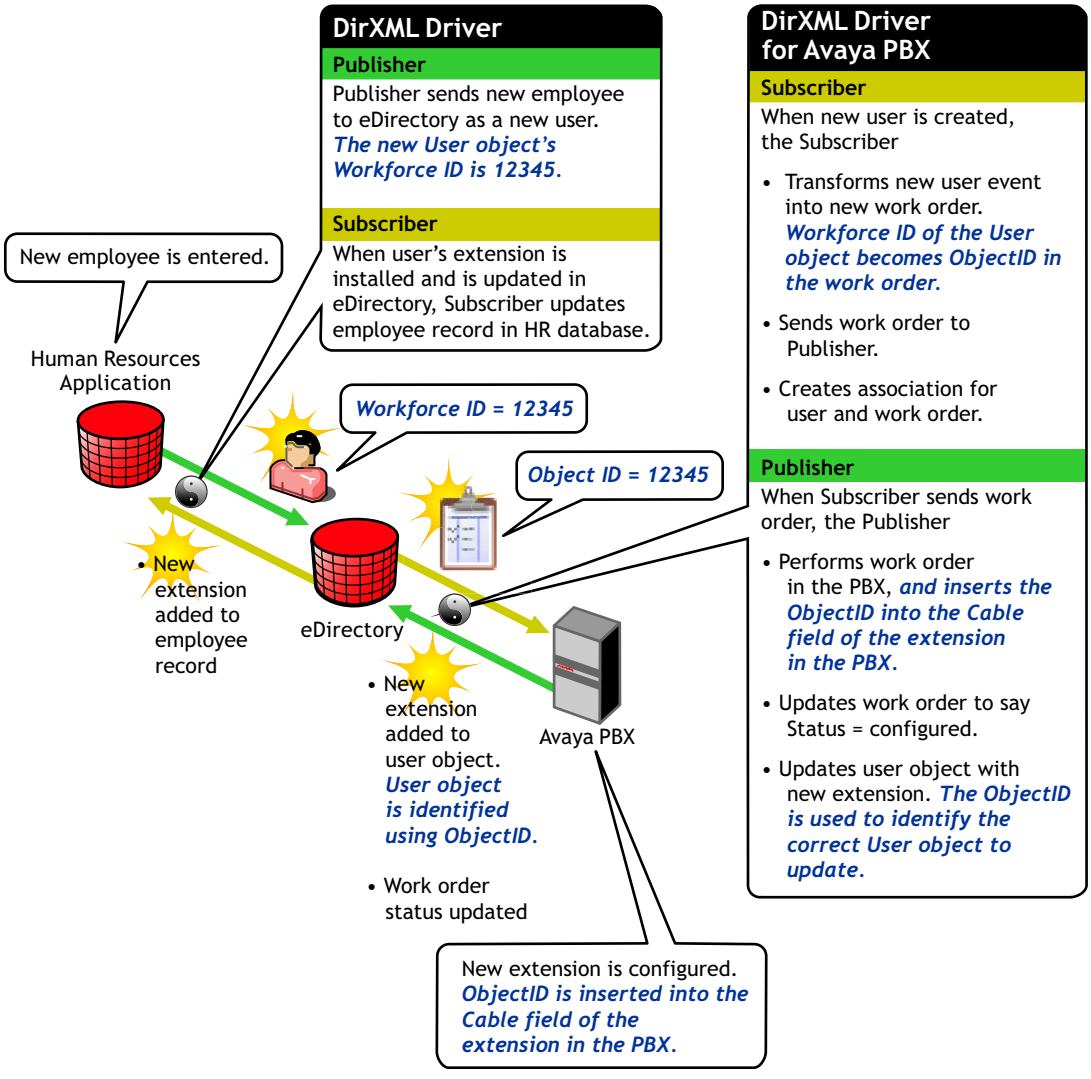
Figure 6 Flowchart of Workforce Tree Configuration



How the Driver Uses 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 How the Object ID Is Used in the Process



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.

Rule or Style Sheet	What it does
Subscriber Filter	Allows only events for nwoWorkOrder and User objects to be processed.
Event Transformation	Not used in sample configuration.
Matching Rule	Not used in sample configuration.
Create Rule	Contains rules only for nwoWorkOrder and User objects. For an nwoWorkOrder object, requires values for the following attributes. ◆ PBXName ◆ nwoStatus ◆ nwoSendtoPublisher ◆ nwoDoItNow ◆ nwoAction ◆ nwoDisplayName For a User object, requires values for the following attributes. ◆ Surname ◆ L 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 style sheet to use L to determine which PBX site to use, or what setcor restrictions to use.) ◆ Workforce ID 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. 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. For a description of these attributes, see “DirXML-nwoWorkOrder Object” on page 57 and “User Objects and their DirXML Associations” on page 61.
Placement Rule	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 DoItNow flag is set to True. 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.

Rule or Style Sheet	What it does
Command Transformation	Takes user events and changes them to PBX actions. ◆ 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	Maps eDirectory namespace to PBX namespace. Handles mapping of work orders, extensions, and PBX site objects.
Output Transformation	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. (PBX "template" for how to configure the phone type).???

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.

Rule or Style Sheet	What it does
Input Transformation	Not used in sample configuration.
Schema Mapper	Maps PBX namespace to eDirectory namespace. Handles mapping of work orders, extensions, and PBX site objects.
Event Transformation	Transforms pbxExtension add, or modify, or delete events into events that modify user objects. ◆ Transforms nwoExtension Add events into user modify events. Uses 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 modify events to modify the preferred name or object ID. ◆ Transforms delete events for extensions into modify events to remove the extension from all that users that 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 sample configuration.
Create Rule	Not used in sample configuration.
Placement Rule	◆ Places work order objects in the correct container if SendToPublisher = True. ◆ Puts the extension objects in the correct container.
Command Transformation	Not used in sample configuration.

User Events That Can Trigger a Work Order

User event from the Human Resources application	Work Order that is created by the style sheets	Ideas for what you could do when customizing the policies
New employee is entered in HR application	◆ Install Work Order created to install new extension ◆ Work order is marked DoItNow, so it is performed immediately. ◆ The user is updated with the new phone extension in eDirectory.	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.	You could customize the policies to do the following: ◆ Move work order is created ◆ If the move requires a new extension, the employee is updated with the new extension ◆ The old extension is removed from the user object.
Employee's name changes	◆ Modify work order is created and is performed immediately ◆ The display name in the PBX is modified, so the users' 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.

Planning for the Workforce Tree Configuration

See "Planning" on page 21.

Setting Up the Workforce Tree Configuration

Follow the instructions in "Installation" on page 25, and when you are creating a driver object, use the sample configuration file named AvayaUser.xml.

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 could keep track of this by using the log file or Nsure Audit to capture the warning messages that are generated.

6 Work Order Database Configuration

As described in the Base Configuration section, you can use eDirectory 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 Identity Manager Driver for Avaya PBX can fit into that environment as well. The driver can perform work orders that are created in another application and synchronized to eDirectory.

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 samples provided for this kind of configuration (in a file named OracleWorkOrderNew.xml) are some individual policies that contain scripting that could be used in a driver configuration for the JDBC driver. A complete driver configuration is not provided, but this section explains the concept of this kind of solution.

This solution could be combined with the Workforce Tree configuration example, so that work orders can have two sources:

- Automated requests triggered by user events in eDirectory, 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.

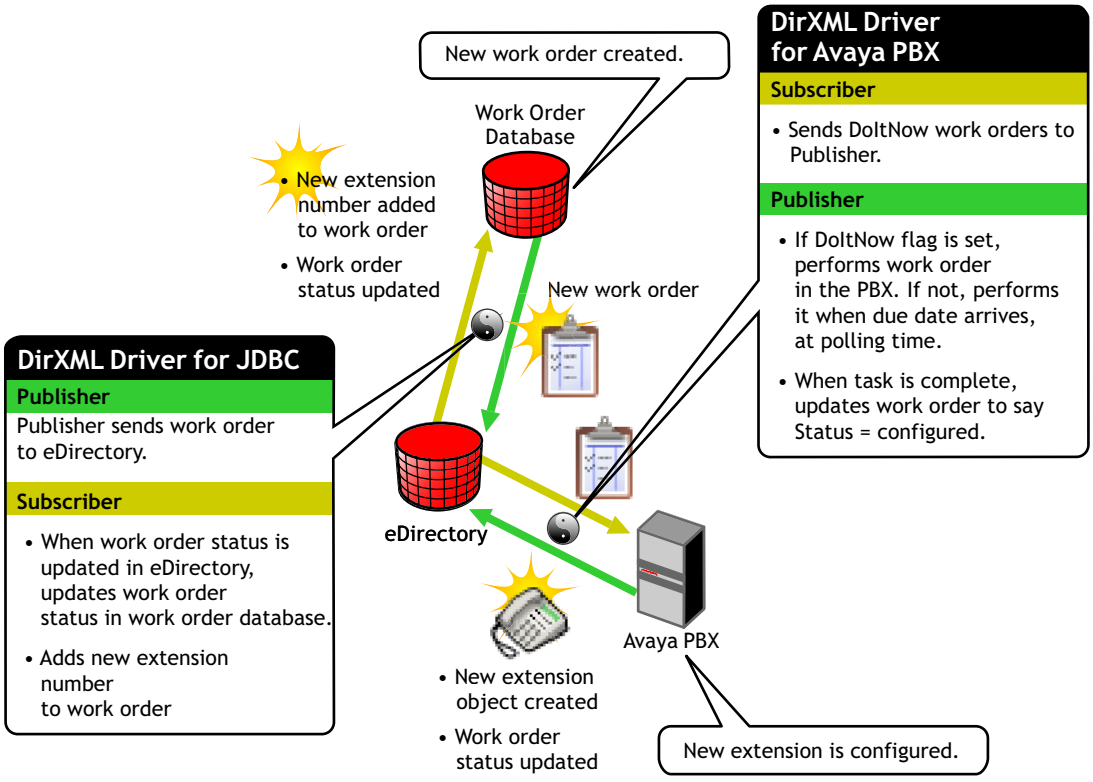
In this section:

- "How You Could Configure a Work Order Database Configuration" on page 46
- "About Work Order Systems" on page 49
- "Planning for the Work Order Database Configuration" on page 49
- "Setting Up the Work Order Database Configuration" on page 49
- "Using Log/Reporting Functionality to Report Warnings" on page 49

How You Could Configure 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 Graphical Representation of Work Order Database Configuration



The following flowcharts show the process that could be followed for a work order database configuration. Figure 9 is a flowchart of how the configuration could work for an Install work order with the DoItNow flag set to true, and Figure 10 is a flowchart showing how the configuration could work for an Install work order with the DoItNow flag not set.

Figure 9 Flowchart of Work Order Database Configuration with DoItNow flag Set to True

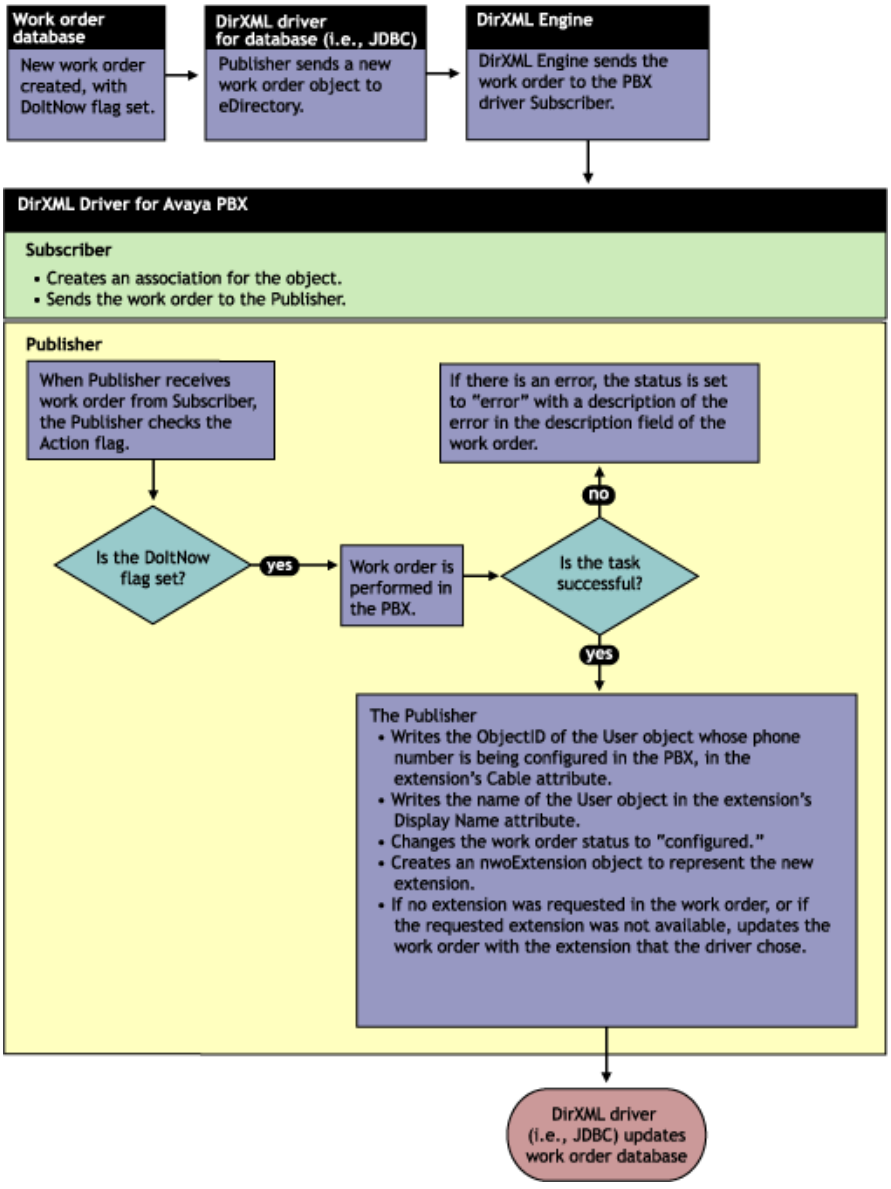
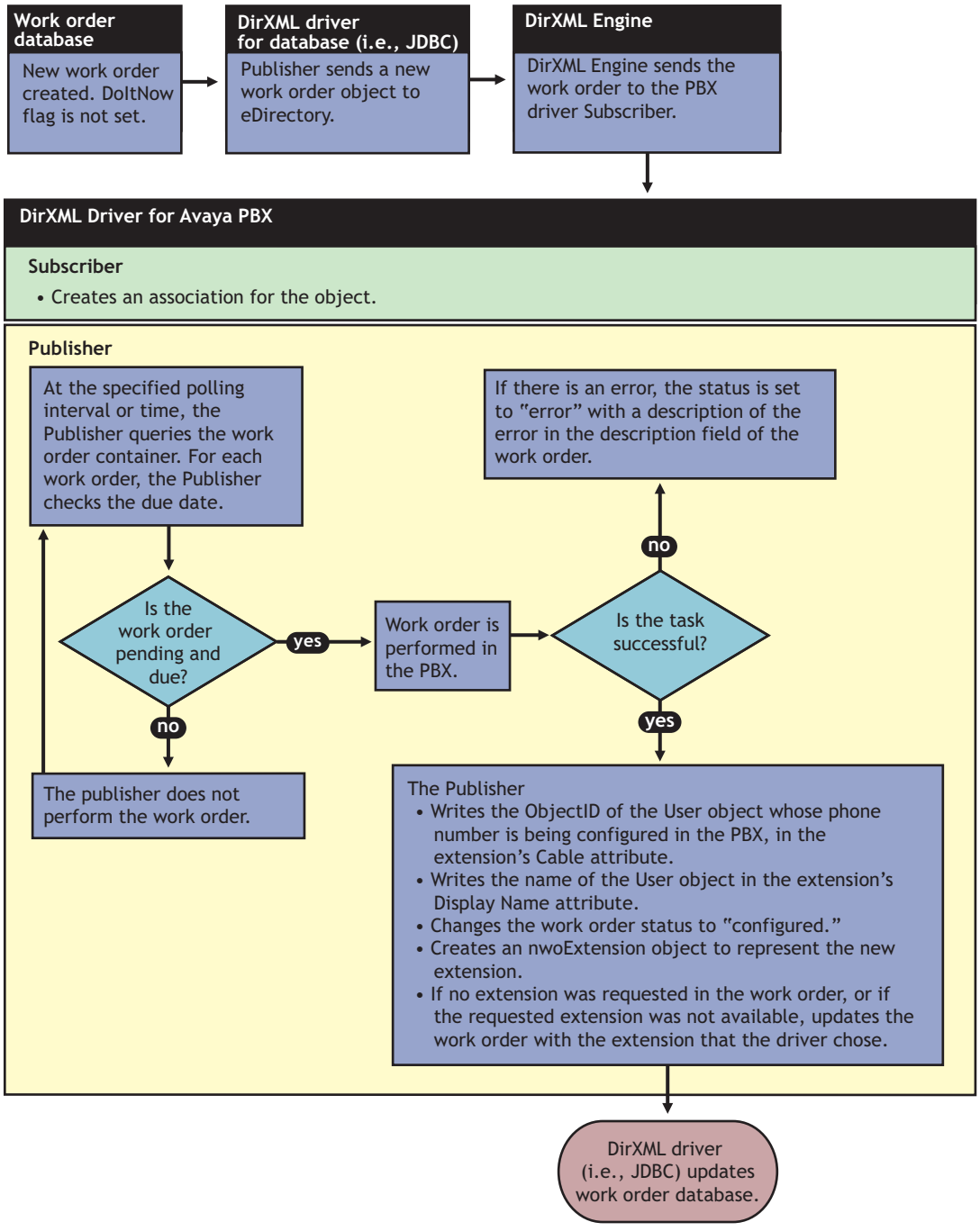


Figure 10 Flowchart of Work Order Database Configuration with DoItNow Flag Not Set



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 eDirectory as nwoWorkOrder objects because the JDBC driver sends them to eDirectory. The Identity Manager Driver for Avaya PBX performs the work orders and the results are returned to eDirectory. 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.

How To Configure the Publisher Channel

In this kind of configuration, note that you would want work orders to 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.

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 would 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 would be responsible for synchronizing work order data between the corporate work order system and the generic work order container in eDirectory used by the driver.

Planning for the Work Order Database Configuration

See "Planning" on page 21.

Setting Up the Work Order Database Configuration

Follow the instructions in "Installation" on page 25, and then create a custom configuration. A file named OracleWorkOrderNew.xml is provided that contains sample scripts that could be helpful. However, it does not contain a complete driver configuration.

Using 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 could keep track of this by using the log file or Nsure Audit to capture the warning messages that are generated.

7 Managing the Identity Manager Driver for Avaya PBX

Because the driver is meant to be customized, many aspects of managing the driver are specific to your particular implementation, but this section contains some management procedures that would be applicable to all implementations.

In this section:

- "Changing How Often the Driver Performs Work Orders" on page 51
- "Changing the Location of PBX Site, Work Orders, User, and Extension Objects" on page 52

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 seconds shown.

   - If you want the driver to perform work orders at a certain polling interval, enter a number of seconds.
   - If you want to turn off the polling interval so that the driver performs work orders only at a time of day you specify, enter 0 in the field.

     This choice would be useful if you wanted to make sure that work orders were not being performed during daytime work hours, for example.

3. To enter 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, enter 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.

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 in the Driver Parameters.

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

8 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 5, "Workforce Tree Configuration," on page 37.

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

- The user exists in eDirectory.
- The style sheets 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 a DirXML 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 style sheets or tools to do so).

The Migrate into NDS command allows you to import data from an application into eDirectory. 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 would be 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.

- Checking the accuracy of existing phone information listed for users in eDirectory

  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 were entered manually, that data entry error would remain even after you start using the driver.

If you want to make sure that existing phone information listed for users is correct, you could compare the pbxExtension objects with the user objects by looking at the Display Name in the pbxExtension object. If display names were 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 would probably still be required to match up 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 could 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 could 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 were fairly confident in the information that was in the PBX, and you were creating a new eDirectory tree and wanted to populate it with users based on the users represented in the PBX.

A Schema for PBX Management

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

- DirXML-pbxSite
- DirXML-nwoWorkOrder
- DirXML-pbxExtension

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

iManager plug-ins are provided to help you create or view these objects.

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 stop and restart the driver.

  This step is necessary so the driver will recognize the changes. The driver queries for PBX site information only at startup. Because of this, normally no DirXML association is 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 enter values for each attribute unless the table indicates that a value is not required or is conditional.

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-pbxPbxName	PbxName	Local name for the PBX.	Single value case ignore string	ProvoPBX
Telephone Number	TelephoneNumber			
DirXML-pbxLoginName	LoginName	Name used to authenticate to PBX.	Single value case ignore string eDirectory 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 eDirectory 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
DirXML-pbxHotJacks	HotJacks	Are jacks hot? If your environment has cold jacks, you must fill in the Nodes attribute as well.	Single value boolean	true
DirXML-pbxAccessType	AccessType	How to access PBX: dialup or telnet.	Single value case ignore string	telnet or Emulate
DirXML-pbxExtensionLength	ExtenLength	The number of digits used for extensions.	Single value Numeric string	5
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

pbxSite Attributes (eDirectory namespace)	PbxSite Attributes (Driver namespace)	Description	Format	Sample Value
DirXML-pbxNonDidPhoneBlockBegin	NonDidPhoneBlockBegin	(Conditional: This attribute is required only if you use nonDID phone numbers.) Beginning number block of non DID extensions assigned to PBX.	Single value Numeric string	600000
DirXML-pbxNonDidPhoneBlockEnd	NonDidPhoneBlockEnd	(Conditional: This attribute is required only if you use nonDID 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	133.65.121.98
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

DirXML-nwoWorkOrder Object

The DirXML-nwoWorkOrder object (sometimes referred to as the work order object in this manual) 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 5, “Workforce Tree Configuration,” on page 37, the ObjectID is used to find and update the User object in eDirectory, which can subsequently be used to update phone information for the employee in the human resources application. The ObjectID is what allows this flow of information to take place. Without it, the driver can perform the work order in the PBX, but no user information will be updated afterward.

The following table shows the attributes you need to specify:

nwoWorkOrder Attributes (eDirectory namespace)	PbxWorkOrder 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 WorkOrderNumber attribute.	case ignore string	WorkOrder1???
DirXML-nwoExtension	Extension	The extension for which the work order is being performed. 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 will assign the next available extension in numeric order. If this attribute is blank for an install work order, the driver will assign the next available extension in numeric order.	Numeric string	12345 or no value
DirXML-nwoDueDate	DueDate	Work order due date. Must be in the format MM/DD/YYYY.	case ignore string	06/12/2003
displayName	UserName	Name to be displayed on caller ID. You can automate the creation of this attribute based on the name of the User object. You can use a work order to change just the display name for an extension, if desired.	case ignore string	Doe, John
DirXML-nwoObjectID	ObjectID	(Required by the style sheets only) The ID used by style sheets to associate to the user object. In the base configuration, the value of displayName is used. You can use an ID of your choice, such as the employee ID.	case ignore string	0000804F
DirXML-pbxName	PbxName	Local PBX name. This should be the name of the pbxSite object you created in eDirectory. Use just the common name, not the DN.	case ignore string	ProvoPBX

nwoWorkOrder Attributes (eDirectory namespace)	PbxWorkOrder Attributes (Driver namespace)	Description	Format	Sample Value
DirXML-nwoNode	PbxNode	Node where the new extension should be placed. This is used only for cold jacks.	integer	East West
DirXML-nwoPhoneType	PhoneType	Phone type recognized by the PBX. This is used by the style sheets to set up the extension.	case ignore string	8410
DirXML-nwoDoItNowFlag	DoItNowFlag	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 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 eDirectory, such as a work order database.	integer	00001
DirXML-nwoAction	Action	Work order activity: install, setcor, modify (you can modify display name, objectID), move, remove, disable, disconnect.	case ignore string	install
Description	Description	(Optional) Description of work order. If status is error this will contain a description of the error.	case ignore string	New extension
DirXML-nwoExtensionType	ExtensionType	Type of extension to configure. Example: Did or nonDid.	case ignore string	DID
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-roomNumber	Room	(Optional) Room that the extension is in.	case ignore string	Conf. Room 123-A
DirXML-nwoPort	Port	(Required if you have cold jacks.) Port the extension is wired to. If you have hot jacks, this information is not necessary; you can leave the attribute blank.	case ignore string	10C0611

nwoWorkOrder Attributes (eDirectory namespace)	PbxWorkOrder Attributes (Driver namespace)	Description	Format	Sample Value
DirXML-nwoRestrictionClass	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 will receive the default class of restriction as defined in the PBX.	integer	01
DirXML-nwoPort		Port used for extension.	case ignore string	01A0101
DirXML-nwoPbxName		Name of the PBX where the extension exists.	case ignore string	PBX Name

The following table shows the pbxWorkOrder association.

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

DirXML-pbxExtension object

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

In the base configuration, this object is specified in the Filter and in the Schema Mapping rule.

However, in some configurations, this object will be used only as output from the driver and will be immediately transformed into another object by the style sheets, so this kind of object will never be created in eDirectory. For example, nwoExtension objects are not created in the workforce tree configuration explained in Chapter 5, "Workforce Tree Configuration," on page 37. Instead, nwoExtension object information that comes from the driver shim is transformed into user object information by the style sheets. In configurations that never create nwoExtension objects in eDirectory, it is not necessary to include this kind of object in the Filter and the Schema Mapping rule.

pbxExtension Attributes (eDirectory namespace)	pbxExtension 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 Extension attribute.	case ignore string	12345
displayName	Name	Display name of the extension this object represents.	case ignore string	Doe, John
DirXML-nwoExtension	Extension	The extension number.	case ignore string	12345

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-nwoStatus	Status	Status of extension	case ignore string	configured
DirXML-nwoRestrictionClass	RestrictionClass	Class of restriction, used to grant specific calling access to the extension. For example, you might define class 01with a restriction that no international calls can be made. The names of these classes are determined by the PBX administrator.	integer	01

The following table shows the association for the pbxExtension object.

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

User Objects and their DirXML Associations

The driver can be configured to create work orders based on creation of or changes to a user object, as discussed in Chapter 5, “Workforce Tree Configuration,” on page 37.

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 will receive a DirXML association for the driver when they are sent to the Subscriber channel. The table below shows what the value of the association will be for a user object.

Driver	State	Association
AvayaPBXDriver	Associated	*/PBXName/WorkOrderCommonName/Time*

You might expect that the association for a user object for the Avaya PBX Driver would be the phone extension. However, this is not the case. The association for the user object is made before a work order is performed. In the case of 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.