Configuration and Administration Guide

iPlanet™ Meta-Directory

Version 5.0

August 2001

Contents

About This Guide

The iPlanet Meta-Directory Configuration and Administration Guide describes how to set up and manage iPlanet Meta-Directory. This includes installing and configuring the join engine and any connectors needed to integrate your choice of external data sources as well as information on how manage the system. This preface contains the following sections:

- What You Are Expected to Know
- The iPlanet Meta-Directory Documentation Set
- Organization of This Guide
- Documentation Conventions
- Where to Find Additional Information

What You Are Expected to Know

This *Configuration and Administration Guide* is intended for use by system administrators. Before reading this guide, you should be familiar with Lightweight Directory Access Protocol (LDAP) as well as the Windows NT or the Solaris™ operating environment on the servers on which Meta-Directory is installed.

Before beginning to configure iPlanet Meta-Directory, it is also assumed that you read the *iPlanet Meta-Directory Deployment Guide* and install Meta-Directory software according to the instructions in the *iPlanet Meta-Directory Installation Guide*.

The iPlanet Meta-Directory Documentation Set

The iPlanet Meta-Directory 5.0 documentation set includes the following materials:

- *iPlanet Meta-Directory Deployment Guide* describes the iPlanet Meta-Directory system and how to plan and implement a Meta-Directory solution.

- *iPlanet Meta-Directory Installation Guide* gives instructions on how to install the iPlanet Meta-Directory software on both Solaris and Windows NT systems.
- *iPlanet Configuration and Administration Guide* (this guide) documents how to set up and manage iPlanet Meta-Directory. Configuring the join engine and components as well as managing the system from the console and the command-line are addressed.
- *Release Notes* include information about what is new in this release, last minute installation tips, known software limitations, and how to report problems.

NOTE Be sure to check the Meta-Directory documentation web site for updates to the release notes and for revisions of guides. Updated documents will be marked with the revision date. `http://docs.iplanet.com/docs/manuals/metadir.html`

Organization of This Guide

The Configuration and Administration Guide (this guide) has five parts:

- Part 1, “Getting Started,” gives an overview of Meta-Directory as well as the essential steps needed to configure the software. It also includes a sample configuration and a comparison of the graphical user interface for Meta-Directory and related iPlanet products.
- Part 2, “The Join Engine,” explains the join engine and how to configure it to flow data. This includes a discussion of the join process, the join process rules and the data views necessary to implement them. There is also a chapter on procedures related directly to the Directory Server to which Meta-Directory is connected.
- Part 3, “Meta-Directory Connectors,” explains how to configure the data connectors provided with Meta-Directory. The concept and configuration of both indirect and direct connectors are discussed.
- Part 4, “System Management,” explains how to manage your Meta-Directory system. Logging and statistics, command-line administration, utilities, data servers and access permissions are included.
- Part 5, “Appendices,” contains reference information on the join process operators, the administration protocol, error messages and troubleshooting.

The table below lists and briefly describes the content of the *Configuration and Administration Guide.*

Table 1 *Configuration and Administration Guide* Chapters

Chapter	Description
About This Guide	An outline of the Configuration and Administration Guide and a description of the iPlanet Meta-Directory documentation set
Part 1, "Getting Started"	
Chapter 1, "Meta-Directory Overview"	A brief explanation of iPlanet Meta-Directory concepts
Chapter 2, "Meta-Directory Principles at Work"	An example of a typical way to configure Meta-Directory
Chapter 3, "Quick Start"	An easy reference on the steps needed to set up Meta-Directory
Chapter 4, "The Console Interface"	An overview of the different iPlanet product consoles used within Meta-Directory
Part 2, "The Join Engine"	
Chapter 5, "Configuring The Join Engine"	An explanation of the join engine and how to configure it using the join process rules
Chapter 6, "Views in Meta-Directory"	An explanation of connector, participating and meta views and how to enable them
Part 3, "Meta-Directory Connectors"	
Chapter 7, "Connectors and Connector Rules"	A explanation of the rules for transferring data between the external data source and the connector view
Chapter 8, "Configuring The Universal Connector"	A explanation and configuration of the Universal Text Connector
Chapter 9, "Configuring the Universal Text Parser"	A explanation and configuration of the Universal Text Parser
Chapter 10, "Configuring the NT Domain Connector"	A explanation and configuration of the NT Domain connector
Chapter 11, "Configuring the Active Directory Connector"	A explanation and configuration of the Active Directory connector.
Chapter 12, "Configuring the Database Connector"	A explanation and configuration of the Database Connector
Part 4, "System Management"	
Chapter 13, "Starting and Stopping Components"	A discussion of the processes for starting and stopping Meta-Directory components

Table 1 *Configuration and Administration Guide* Chapters *(Continued)*

Chapter	Description
Chapter 14, “Monitoring Meta-Directory Components”	An explanation of the logging process
Chapter 15, “Administration Tools”	An explanation of the tools to query and fix entries
Chapter 16, “Command-Line Administration”	A description of managing the system via the command-line
Chapter 17, “Managing Servers and Permissions”	A description of how to configure data servers and access permissions
Part 5, “Appendices”	
Appendix A, “The Join Process Operators”	A description of the join process protocol
Appendix B, “The Meta-Admin Protocol”	A description of the command-line protocol
Appendix C, “Troubleshooting Meta-Directory”	A description of problem-solving techniques and common configuration problems
Appendix D, “Error Messages”	A listing of error messages along with possible causes and solutions

Documentation Conventions

In all iPlanet Meta-Directory documentation, certain typographic conventions and terminology are used to simplify discussion and to help better understand the material.

Typographic Conventions

This book uses the following typographic conventions:

- *Italic type* is used within text for book titles, new terminology, emphasis, and words used in the literal sense.
- `Monospace font` is used for sample code and code listings, API and language elements (such as function names and class names), filenames, pathnames, directory names, HTML tags, and any text that must be typed on the screen.

- *Italic serif font* is used within code and code fragments to indicate variable placeholders. For example, the following command uses *filename* as a vairable placeholder for an argument to the gunzip command:

  ```
  gunzip -d filename.tar.gz
  ```

Terminology

Below is a list of the general terms that are used in the iPlanet Meta-Directory documentation set:

- *Meta-Directory* refers to iPlanet Meta-Directory and any installed instances of the iPlanet Meta-Directory software.
- *Meta-Directory components* refers to the collective set of iPlanet Meta-Directory components and software you have installed and running on your system, including the join engine and any external data source connectors.
- E*xternal data source* refers to any user data that originates outside of the core Meta-Directory components, whether the data is coming from another database, directory server, data file, or other source of data.
- *Directory Server* refers to an installed instance of iPlanet Directory Server or Netscape Directory Server.

  iPlanet Meta-Directory can synchronize data using any LDAPv2 and LDAPv3-compliant directory server, as long as the LDAP server supports a change log mechanism similar to the one implemented in Netscape™ Directory Server 4.1*x*. The term *Directory Server* refers to the instances of iPlanet Directory Server and Netscape Directory Server that you have installed to work with iPlanet Meta-Directory.
- Similarly the term *Administration Server* refers to an installed instance of Netscape Administration Server or iPlanet Administration Server, whether it be used with the Meta-Directory components or another Netscape or iPlanet server.
- *NETSITE_ROOT* is a variable placeholder for the home directory where you have installed iPlanet Meta-Directory and any other iPlanet servers installed into the same server group.
- The term *flow* is used rather loosely to refer to the process of synchronizing data between an external data source and the meta view. You “flow” data through a connector to the connector view and then “flow” it to the meta view. The contrary is also true, you “flow” data from the meta view back to the connector views and out to the external data sources.

Where to Find Additional Information

In addition to iPlanet Meta-Directory documentation set, you should be familiar with the documentation for products that are used in conjunction with it. Of particular interest are the iPlanet Console and iPlanet Directory Server documentation sets. This section lists additional sources of information you may find helpful as you use iPlanet Meta-Directory.

iPlanet Console Documentation

You can find the iPlanet Console documentation at the following site:

`http://docs.iplanet.com/docs/manuals/console.html`

iPlanet Directory Server Documentation

You can find the iPlanet Directory Server documentation at the following site:

`http://docs.iplanet.com/docs/manuals/directory.html`

Directory Server Developer Information

In addition to the Directory Server documentation, you can find information on Meta-Directory, LDAP, the iPlanet Directory Server, and associated technologies at the following iPlanet developer sites:

`http://developer.iplanet.com/tech/directory/`

`http://www.iplanet.com/downloads/developer/`

Other iPlanet Product Documentation

Documentation for all iPlanet and Netscape servers and technologies can be found at the following web site:

`http://docs.iplanet.com/docs/manuals/`

iPlanet Technical Support

you can contact iPlanet Technical Support through the following location:

`http://www.iplanet.com/support/`

Part 1

Getting Started

Chapter 1, “Meta-Directory Overview”

Chapter 2, “Meta-Directory Principles at Work”

Chapter 3, “Quick Start”

Chapter 4, “The Console Interface”

Chapter 1

Meta-Directory Overview

This chapter provides an overview of iPlanet Meta-Directory including how data is integrated and how components interact. It contains the following sections:

- What is Meta-Directory?
- Meta-Directory Components
- Meta-Directory Views
- How Meta-Directory Works

What is Meta-Directory?

iPlanet Meta-Directory is a set of software components that synchronize data from one or more external data sources into a single repository. This repository then becomes the authoritative source for the combined data, acting as a starting point for searches and for modifications that flow back to the original data sources. These data sources can include LDAP directories, SQL (Oracle) databases, and other proprietary formats.

Meta-Directory Components

The join engine and direct and indirect connectors are the components which flow data into and out of Meta-Directory and integrate it into one definitive source. The connectors transfer information to and from a number of data sources to the join engine. The join engine links these entries together to form one repository.

Connectors

A *connector* transfers information between an external data source and its corresponding *connector view*. When configuring a connector to transfer the information, an LDAP sub-tree is created on a Directory Server. This sub-tree is called a connector view. The connector view is populated with LDAP-structured copies of the entries that reside in the external data source. From this connector view, Meta-Directory is able to integrate the new LDAP-structured data into an LDAP meta view.

There are two types of connectors: direct and indirect. The connector view for an LDAP Directory Server or a SQL (Oracle) database uses a *direct connector*; it communicates directly with the join engine. The connector view for other sources of data uses an *indirect connector* which translates the data into LDAP for inclusion into a connector view before passing it to the join engine.

Direct Connectors

iPlanet Directory Server 4.1x and 5.0 and an Oracle database accessible using Oracle Call Interface have direct connections to the join engine. Because the join engine understands LDAP, it can directly read or write any entry stored on an LDAP-based Directory Server. On the other hand, in order to read and write to an entry stored within Oracle's SQL database, the join engine uses the *Database connector* plug-in to provide direct, two-way access. (Because the Database Connector is a join engine plug-in as opposed to a connector outside the join engine, it is considered a direct connector.)

Indirect Connectors

Indirect connectors are used to transport entries stored in external data sources that use protocol not directly accessible by the join engine. The indirect connectors manage the process, transforming the data using rules and filters. (The indirect connector rules include Attribute Flow Rules, Default Attribute Value Rules and Filter Rules.) Meta-Directory supports the following indirect connectors:

Universal Connector and Universal Text Parser

The Universal connector (also known as the Universal Text Connector or UTC) is an indirect connector that enables the transfer of data between data sources and a connector view. The Universal Text Parser (UTP) is a set of text file parsers and generators that are used with the UTC to make certain text files [currently Comma-Separated Values (CSV) files, LDAP Data Interchange Format (LDIF) files and Name-Value Pair (NVP) files] compatible with the connector view.

NT Domain Connector

The NT Domain connector is a Universal connector with Windows NT-specific Perl scripts and binaries that provides two-way synchronization of user and group data between a Windows NT SAM database and its connector view.

Active Directory Connector

The Active Directory connector is a Universal connector with Active Directory-specific Perl scripts and binaries that provides two-way synchronization of user and group data between an Active Directory database and its connector view.

Join Engine

The core service of Meta-Directory is the join engine, responsible for linking entries and controlling the flow of information from a connector view into or out of the *meta view*. The meta view is an LDAP sub-tree where entries from one or more connector views are linked, stored and accessed. The join engine synchronizes the data by flowing it through a sequence of rules and filters which *map* attributes from the connector view with corresponding attributes in the meta view. (*Mapping* is the assignment of an attribute or entry in one source to link with a particular attribute or entry in another source.) The movement of this data through the sequence of rules is referred to as the *join process*. The join engine also monitors the connector views for changes and incorporates them into the meta view; conversely, it monitors the meta view for changes and incorporates them into the connector view and back to the related external data source.

Join Process

The join process is the movement of data through a sequence of rules and filters that are used by the join engine to determine how connector view entries will be represented in the meta view. To successfully join entries, the join engine must match data in the connector view with a congruous entry in the meta view. To do this, rules containing values and attributes are issued against the connector view. These rules find a matching entry, and create a link to it, in the meta view. The search strings include Join Rules, Attribute Construction Definitions, Attribute Flow Rules, Filters, and DN Mapping Rules.

Meta-Directory Views

In connecting data from several data sources, Meta-Directory allows the information to be accessed in two ways:

- The meta view, a representation of the culmination of the join process — a view of the combined data from all connector views.
- The connector view, a representation of the data from its initial source, the external data source.

The following sections describe these views in more detail.

Meta View

The *meta view* is a unified view of LDAP entries from one or more connector views, representing the result of the join engine's join process. After the join engine receives information from a connector view, it synchronizes the information in the meta view.

From the meta view, you can view the linked entries as well as modify them and send the modifications back to the original connector views. Meta-Directory supports only one meta view per join engine.

Connector Views

A *connector view* is an LDAP representation of an entry (or entries) that resides in an external data source. This representation is merged with other connector views to form the meta view. In order for this merger to take effect, you must specify that a connector view is a *participating view*. Adding a participating view allows the connector view to 'participate' in the join process; enabling it allows the data to flow from the connector view into the meta view. (It is through the participating view that the rules of the join process are applied.)

How Meta-Directory Works

Information from data sources is channeled to one or more connector views via a direct or indirect connector. The join engine combines the data from the connector views into one meta view. All the information in the meta view is then accessible via a directory server and other web-based applications. This is how the process plays itself out:

1. The join engine works with direct and indirect connectors that channel data from external sources into a Directory Server connector view.

   a. The direct connectors read entries from either an LDAP directory or an Oracle database and map the data into LDAP-configured connector views.

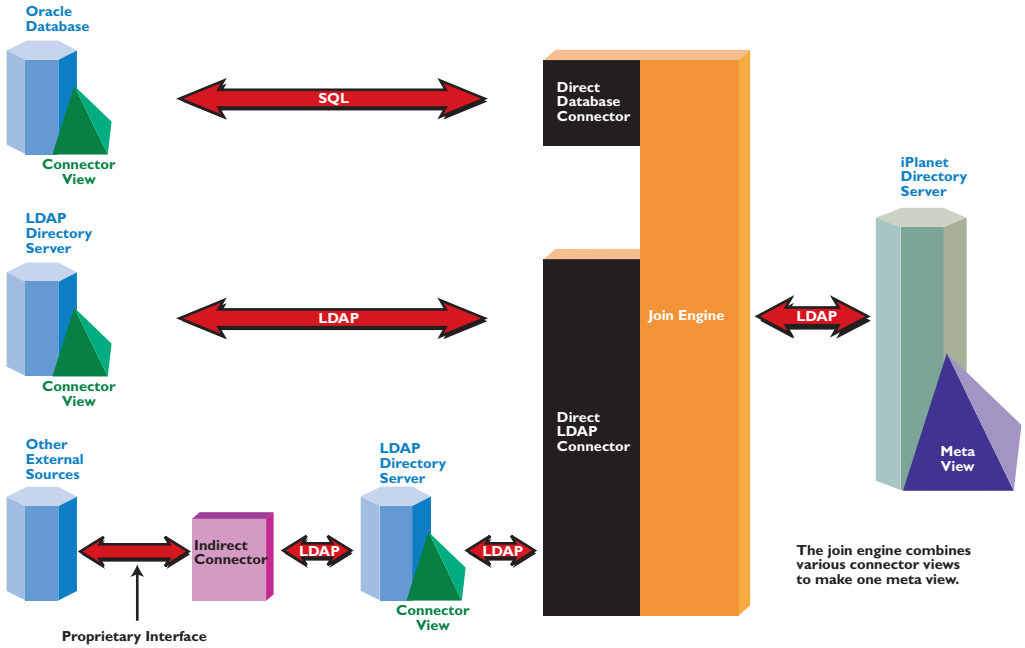
   b. For other sources of data, an indirect connector transforms information from a proprietary format into LDAP and flows this data into an LDAP-configured connector view.

   The data in these LDAP-configured connector views can then be flowed into, and stored on, a meta view (another directory subtree in a Directory Information Tree) via the direct LDAP connector.

2. The meta view becomes the replicated source for the combined data, allowing searches and changes to be made and flowed back to the original source.

Figure 1-1 is a graphical representation of Meta-Directory's process.

Figure 1-1 Data Integration Into Meta-Directory



Chapter 2

Meta-Directory Principles at Work

This chapter uses an example to show how to configure Meta-Directory to integrate three different external data sources into one unified meta view. In order to make this example compliant with the Windows NT or Solaris operating system, the sample uses data sources that work with both. These include an LDIF data file imported into a Directory Server, a CSV data file using the Universal Text Parser (UTP), and an Oracle database using the Database Connector. These data samples are small but the principles of working with them using Meta-Directory are the same as they would be for large bodies of data. This chapter includes the following sections:

- The LDIF Data File
- The CSV Data File
- The Oracle Database
- Background Information
- Configuration of Connectors and Join Process Rules
- Integration of the Data With Join Rules

The LDIF Data File

An LDIF file is a text version of LDAP data. The following LDIF data is what will be imported into Directory Server. Once it is imported, the LDAP data sub-tree is directly accessible by Meta-Directory and would be considered a direct connection.

Code Example 2-1 The LDIF Data

```
dn: uid=aquinn,o=firstgroup
objectclass: top
objectclass: person
objectclass: organizationalperson
objectclass: inetorgperson
cn: Al Quinn
sn: Quinn
mail: alanquinn@siroe.com
title: Writer
uid: aquinn
givenname:Alan

dn: uid=bmoore,o=firstgroup
objectclass: top
objectclass: person
objectclass: organizationalperson
objectclass: inetorgperson
cn: Betty Moore
sn: Moore
mail: bettymoore@siroe.com
title: Designer
uid: bmoore
givenname: Betty

dn: uid=cwayne,o=firstgroup
objectclass: top
objectclass: person
objectclass: organizationalperson
objectclass: inetorgperson
cn: Cal Wayne
sn: Wayne
mail: calvinwayne@siroe.com
title: Accountant
uid: cwayne
givenname: Calvin

dn: uid=dgordon,o=firstgroup
objectclass: top
objectclass: person
objectclass: organizationalperson
objectclass: inetorgperson
cn: Debby Gordon
sn: Gordon
mail: deborahgordon@siroe.com
title: Writer
uid: dgordon
givenname: Deborah

dn: uid=ecrane,o=firstgroup
objectclass: top
objectclass: person
objectclass: organizationalperson
objectclass: inetorgperson
```

Code Example 2-1 The LDIF Data *(Continued)*

```
cn: Eric Crane
sn: Crane
mail: ericcrane@siroe.com
title: Designer
uid: ecrane
givenname: Eric
```

The CSV Data File

A CSV (comma-separated values) file is a text file format that can be output by a standard spreadsheet program. (Its name is derived from the fact that each record is stored in a single line with the individual items of data separated by commas.) The following CSV data will be imported into Meta-Directory via the UTP. (The UTP is considered an indirect connector as it is necessary to map the CSV file data to the Directory Server's LDAP schema before it can be read by Meta-Directory.)

Code Example 2-2 The CSV Data

```
REALNAME,LASTNAME,EMAIL,PHONE,BUILDING,JOBTITLE,ALTERNATIVENAME,
NICKNAME

Frankie Shaden,Shaden,frankshaden@siroe.com,408 555
1234,15,Accountant,fshaden,Frank

Gary Parker,Parker,garyparker@siroe.com,408 555
5678,7,Writer,gparker,Gary

Hal Kent,Kent,haroldkent@siroe.com,408 555
9012,18,Designer,hkent,Harold

Indy Connor,Connor,indranilconnor@siroe.com,408 555
3456,7,Accountant,iconnor,Indranil

Joe Hayes,Hayes,josephhayes@siroe.com,408 555
7890,15,Writer,jhayes,Joseph
```

The Oracle Database

Data stored in an Oracle database can be viewed in Meta-Directory with a Database Connector. The following data will be accessed from an Oracle 8 database via Structured Query Language (SQL). The Database Connector is considered a direct connector because it plugs directly into the join engine.

Table 2-1 The Oracle Data

Employee ID	First Name	Last Name	Salary	Department ID
aquinn	Al	Quinn	75,000	10
bmoore	Betty	Moore	85,000	43
cwayne	Cal	Wayne	80,000	24
dgordon	Debby	Gordon	95,000	10
ecrane	Eric	Crane	85,000	43
fshaden	Frankie	Shaden	80,000	24
gparker	Gary	Parker	95,000	10
hkent	Hal	Kent	85,000	43
iconnor	Indy	Connor	80,000	24
jhayes	Joe	Hayes	95,000	10

Background Information

Siroe, Inc. is a small company that outsources services. It currently has ten employees on its payroll. Personal and employment information on their employees is divided among three data sources. Siroe's Directory Server contains names and email addresses of five employees. Similar information for five consultants is in an Excel spreadsheet program. An Oracle database contains the salary information, employee numbers, and departments of all ten employees. Siroe needs to integrate all three of these sources and identify the employees by job function: four writers, three designers, and three accountants.

The example assumes that iPlanet Directory Server and Meta-Directory have been installed and a join engine has been created.

Configuration of Connectors and Join Process Rules

Siroe, Inc. has hired a systems administrator, Tracy, to help with the integration of their data. Tracy decides to flow the data on Siroe's Directory Server first. The next steps are integrating the CSV file exported from Excel and then syncing the Oracle tables into Siroe's unified employee database.

Flowing LDIF Data to the Connector View

Siroe has two directory servers: one is an iPlanet Directory Server and the other is Netscape Directory Server 1.0. The Directory Server 1.0 is not completely compatible with Meta-Directory so Tracy exports its LDAP data to an LDIF file in order to import it into the newer iPlanet Directory Server. (The Netscape Directory Server will no longer be used; its place is taken by the iPlanet Directory Server.) Tracy creates an instance of the Universal Connector (UTC) from iPlanet Console which also adds a connector view for the data on the newer iPlanet Directory Server by:

- Naming the instance: `firstgroup`
- Naming the view ID: `fg`
- Naming the base dn: `o=firstgroup`

  The LDIF file is already configured as `uid=`*variable*`, o=firstgroup` therefore, the base DN of our connector is input as `o=firstgroup`.
- Entering the LDAP URL of the Directory Server on which the data will be stored
- Entering an authenticating Distinguished Name and password

When Tracy refreshes the Directory Server and Meta-Directory consoles, the *o=firstgroup* LDAP sub-tree is visible in the navigation tree of the Directory Server and the `firstgroup` connector and connector view are visible in the navigation tree of Meta-Directory.

Once the connector view is created, Tracy imports the LDIF file into the connector view sub-tree via the iPlanet Directory Server by using the `Import Database` command from the Directory Server console.

Flowing LDAP Data to the Meta View

The LDIF data is now written to a LDAP Directory Server that is directly connected to the join engine. Therefore, Tracy can begin configuring the join process to flow the LDAP data through the join engine and into the meta view.by creating the rules to define how Siroe would like the data to flow.

Constructing an Attribute

destinationIndicator is an allowed attribute of the `organizationalPerson` object class so Siroe will use this attribute to keep track of an employee's off-site locations. Tracy, therefore, needs to put a value to this attribute. One way to do this is to take a value that is already present and break it into sub-tokens, enabling the use of these sub-tokens for other attributes. As their personnel's email address changes with each job, Tracy constructs an attribute with a value by:

- Naming the constructed attribute: `constructeddestinationIndicator`
- Entering token assignments: `{mail}=%mailto%@%company%.%com%`
- Entering attribute construction: `%company%`

With this rule, Tracy deconstructs the email address field, *variable@company.com* into three tokens: `variable`, `company` and `com`. From these fields, `%company%` is chosen as the value for `constructeddestinationIndicator` that will sync with the LDAP *destinationIndicator* attribute indicating the employee's off-site location.

Mapping Attribute Flow Rules

After constructing the attribute, Tracy configures flow tables to match up the attributes from the LDIF/LDAP data to the attributes in Meta-Directory's schema. Attribute flow rules can be configured for two directions: from the connector view to the meta view and from the meta view back to the connector view. The LDIF/LDAP data contains the following attributes: `objectclass`, `cn`, `sn`, `mail`, `title`, `uid`, and `givenname`; in addition, Tracy needs to flow the constructed attribute `constructeddestinationIndicator`. Tracy creates attribute flow rules by:

- Naming the rule: `fg2mv`
- Choosing the direction: `To Meta View`
- Entering selection criteria: `(%objectclass%==person)`

  This specification means that any entry with an `%objectclass%==person` attribute/value pair will be selected and allowed to map its attributes.

- Adding attribute mappings

  `objectclass`, `cn`, `sn`, `mail`, `title`, `uid`, `givenname` and `constructeddestinationIndicator` map respectively to `objectclass`, `cn`, `sn`, `mail`, `title`, `uid`, `givenname` and `destinationIndicator`

All attributes from the LDIF / LDAP data will flow to the same-titled attribute in the meta view (although Tracy could flow them differently if Siroe wanted different mappings). The constructed attributes are visible when Constructed Attributes is chosen from the `source objectclass` drop-down list. In this case, Tracy maps the `constructeddestinationIndicator` constructed attribute to the schema attribute `destinationIndicator`.

When the attribute mappings are complete, Tracy needs to create a rule set entitled `affg2mvset` and add the `fg2mv` rule to it. Since attribute flow rule sets can not contain rules configured for data that flows in opposite directions, Tracy must also make a separate rule and rule set to map the attributes from the meta view back to the connector view. He names them `fg2cv` and `affg2cvset`, respectively. (The difference between the two rules is that, in the latter case, Tracy does not want the constructed attribute to be flowed back to the connector view from the meta view and therefore does not map it.)

Creating the Entry's Distinguished Name

Tracy needs to configure the distinguished name (DN) for each entry. DN mapping rules create a place for each entry in the destination view. The first entry in the LDIF / LDAP data has as its source DN `uid=aquinn,o=firstgroup`. Tracy wants the meta view DN of this entry to be `uid=aquinn,ou=Writers,o=firstgroup`. The DN mapping rule is created by:

- Naming the rule: `fgwriter2mv`
- Entering the selection criteria: `((%objectclass%==person) AND (%title%==Writer))`

  This specification means that any entry with `%objectclass%==person` and `%title%==Writer` attribute / value pairs will be selected and assigned the following DN.

- Constructing the DN: `uid=%uid%,ou=Writers`

  Tracy wants the unique identifier (in this case, `aquinn`) to be the unique name of each partial DN relative to the base DN. This DN construction creates the organization unit *Writers* in which all entries with `title=Writer` and `objectclass=person` will be placed.

Tracy must now configure a rule for Accountants and Designers. The three completed rules, `fgaccountant2mv`, `fgdesigner2mv` and `fgwriter2mv`, are then added to one rule set that Tracy calls `DNrulesfg2mv` because they are all for flowing data to the meta view.

Tracy then makes three more rules which flow the information back to the connector view, using as selection criteria `cn` in `ou=Writers`, `ou=Accountants`, and `ou=Designers` and constructing a DN of `uid=%uid%, o=firstgroup`. Tracy puts them in a rule set entitled `DNrulesfg2cv`. The two rule sets `DNrulesfg2mv` and `DNrulesfg2cv` will flow the LDAP data into and out of the meta view. Next, Tracy flows the CSV data into the meta view.

Mapping CSV Data to LDAP Using the UTP

The CSV data must be flowed into the connector view using the Universal Text Parser (UTP), so Tracy configures an instance of the Universal Connector (UTC) from iPlanet Console by:

- Naming the instance: `secondgroup`
- Naming the view ID: `sg`
- Naming the base dn: `o=secondgroup`
- Entering the LDAP URL of the Directory Server on which the data will be stored
- Entering an authenticating Distinguished Name and password
- Entering the path name to the `template.pl` script included with Meta-Directory: `NETSITE_ROOT/md50/utc-secondgroup/utp/template.pl`

  The `template.pl` file is part of the UTP. It is a Perl script that tells the UTC what to do. It must be moved to the configured UTC's directory (entered in the path name above) from its default location at `NETSITE_ROOT/bin/utc50/install/templates/universalparser/template.pl`.

Tracy refreshes both consoles so that the *o=secondgroup* sub-tree is visible in the navigation tree of the Directory Server and the `secondgroup` connector view is visible in the navigation tree of Meta-Directory. Now, Tracy needs to convert the data in the CSV file into LDAP data using the Universal Text Parser's `csv.cfg` file.

NOTE When using the Universal Text Parser, attributes are mapped twice. First, CSV data is mapped to LDAP attributes within the `csv.cfg` file and second, the CSV/LDAP attributes are mapped to the Meta-Directory schema by configuring indirect connector rules.

Mapping CSV Data to LDAP Attributes

The `csv.cfg` text file is used to map comma-separated values to LDAP attributes. Tracy finds the file and the `task.cfg` file in `NETSITE_ROOT/bin/utc50/install/templates/universalparser`. Tracy first copies the `csv.cfg` file to the `task.cfg` file. (`task.cfg` is called by `template.pl` but it needs to be configured specifically for CSV conversion into LDIF using the `csv.cfg` file.) Tracy edits the new version of `task.cfg` by editing the `LineFormat` and `ImportLineFormat` sections to map the CSV data attributes `REALNAME, LASTNAME, EMAIL, PHONE, BUILDING, JOBTITLE, ALTERNATIVENAME` and `NICKNAME` to the LDAP attributes `cn`, `sn`, `mail`, `telephoneNumber`, `l`, `title`, `uid` and `givenName`, respectively. Tracy also changes the `InputFile` section of `csv.cfg` to `%ScriptBase%csv.data`. Tracy then places the `csv.data` file and the modified `task.cfg` file in the directory with `template.pl` located at `NETSITE_ROOT/md50/utc-secondgroup/utp/`.

Mapping LDAP Attributes to the Connector View

Once the CSV data has been mapped to LDAP-recognized attributes within the `csv.cfg` file, these LDAP attributes need to be mapped to the LDAP attributes in the connector view using indirect connector rules. First, Tracy specifies the attributes defined in `csv.cfg` as attributes that are available to the connector. For this, each attribute name is entered on the connector's External Available Attributes panel. Once defined, the attributes are accessible for mapping data to and from the connector view attributes. Tracy can now create attribute flow rules for the indirect connector by:

- Naming the rule: `secondgroupflow`
- Choosing the mapping type: Mappings for Locally Owned Objects

  This choice identifies the rule as applying to data flow from the external source to the connector view.
- Match the external attributes `cn`, `sn`, `mail`, `telephoneNumber`, `l`, `title`, `uid` and `givenName` To Connector View attributes `cn`, `sn`, `mail`, `telephoneNumber`, `l`, `title`, `uid` and `givenName`

- Match the external attributes cn, sn, mail, telephoneNumber, l, title, uid and givenName From Connector View attributes cn, sn, mail, telephoneNumber, l, title, uid and givenName
- Choosing the mapping type: Mappings for Connector View Owned Objects

  This choice identifies the rule as applying to data flow from the connector view to the external data source.
- Match the external attributes cn, sn, mail, telephoneNumber, l, title, uid and givenName To Connector View attributes cn, sn, mail, telephoneNumber, l, title, uid and givenName
- Match the external attributes cn, sn, mail, telephoneNumber, l, title, uid and givenName From Connector View attributes cn, sn, mail, telephoneNumber, l, title, uid and givenName

NOTE Again, in order to flow data in both directions, the attributes need to be mapped in both directions AND under each mapping type.

Tracy can now apply the indirect connector rule by selecting the utc-secondgroup indirect connector and its corresponding secondgroupflow rule.

Flowing LDAP-configured CSV Data to the Meta View

From this point, Tracy follows the same procedure established for the LDIF / LDAP data. First, DN mapping rules are configured, selecting the entries and linking them to the DN as laid out in the table below.

Select entries that contain	Link entries to DN constructed as
objectclass=person AND l=15	uid=%uid%, ou=Accountants
objectclass=person AND l=7	uid=%uid%, ou=Writers
objectclass=person AND l=18	uid=%uid%, ou=Designers

Tracy names the rules `sgwriters2mv`, `sgdesigners2mv` and `sgaccountants2mv` and puts them in a rule set called `DNrulessg2mvset`. Whereas the LDIF file had employees identified by job title, the CSV data has them identified by building location with each building occupied by accountants, writers, or designers exclusively. These DN mapping rules place people in the correct sub-trees based on their building location.

Mapping LDAP-configured CSV Data to the Meta View

Attribute flow rules are needed to map the CSV / LDAP attributes in the connector view to the LDAP attributes in the join engine's meta view. Tracy creates attribute flow rules by:

- Naming the rule: `sg2mv`
- Choosing the direction: `To Meta View`
- Entering the selection criteria: `(%objectclass%==person)`

  This specification means that any entry with an `%objectclass%==person` attribute / value pair will be selected and assigned the following mappings.
- Adding attribute mappings

  `uid`, `objectclass`, `telephoneNumber`, `sn`, `mail`, `title`, `givenName` and `l` map respectively to `uid`, `objectclass`, `telephonenumber`, `sn`, `mail`, `title`, `givenname` and `l`

Tracy also creates a separate rule to map the attributes from the meta view back to the connector view; this is called `sg2cv`. Rule sets are created (`afsg2mvset` and `afsg2cvset`)to hold the two configured rules. With that done, all the rules needed to manage the flow of data from the CSV file are complete.

Flowing Oracle Data to the Meta View

Tracy must now flow the Oracle data to the meta view. This data includes salary information and employee identification numbers. To make the flow possible, Tracy needs to configure the Oracle database as a Data Server that can be recognized by the Meta-Directory's join engine.

NOTE In order to bring Oracle data into Meta-Directory, the OCI libraries, available from Oracle, must be installed on the same machine in which the join engine is installed.

Configuring the Oracle Database as a Recognizable Data Server

In order to configure the Oracle database as a recognizable Data Server, Tracy accesses the Data Server panel from Meta-Directory console and completes the general information by:

- Entering a name for the data server
- Entering an alias for the data server

  This is the Oracle connect string.
- Entering a user name that does not already exist within the Oracle database

  This name will be used by Meta-Directory's Database Connector as the privileged user.
- Entering the path name to where the OCI libraries are installed
- Using the default NLS Language Environment Variable

Next, Tracy selects the tables and columns that will participate in the data flow by:

- Choosing the Oracle columns for syncing

  The Uninstrumented Columns are the full list of Oracle columns that Tracy will choose from. The Instrumented Columns are the ones Tracy has selected: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY and DEPARTMENT_ID.
- Choosing (or nominating) a primary key for Meta-Directory's use

  The primary key is unique and not null among all the entries. Tracy chooses EMPLOYEE_ID.

The name and location for the change log tables and triggers are also chosen. (Tuning and DCNS schedule use the default option.) When these selections are saved, a script is generated which, when applied to the database, enables change detection on the selected tables and columns. Tracy then runs the script on the command line via SQLplus (part of Oracle client software) and creates both the connector view and participating view from Meta-Directory console.

NOTE The Oracle table data can never be seen through the Meta-Directory console. Only when the data is synced in the meta view can it be viewed.

Constructing an Attribute for the Oracle Database

When flowing data from an Oracle database, the `objectclass` attribute must always be constructed and flowed with the data as Oracle does not use object classes. Tracy creates an `objectclass` attribute by:

- Naming the constructed attribute: `dbobjectclass`
- No selection criteria is necessary
- Entering attribute construction: `top;person;organizationalperson;inetorgperson`

Tracy also constructs a `commonname` attribute by:

- Naming the constructed attribute: `dbcommonname`
- Entering selection criteria: `(%cv.FIRST_NAME%=@)`

  This specification means that any entry with a connector view value of FIRST_NAME will be selected.
- Entering attribute construction: `%cv.FIRST_NAME% %cv.LAST_NAME%`

  The `dbcommonname` will be made from the connector view's FIRST_NAME and LAST_NAME attributes.

To complete the constructed attributes, Tracy puts each rule in a separate constructed attribute rule set entitled `dbobjectclass` and `dbcommonname`, respectively.

Creating Attribute Flow Rules for the Oracle Database

Tracy can now create the attribute flow rules to map the Oracle columns to their counterpart LDAP attributes. This is done by:

- Naming the rule: `oracle2mv`
- Choosing the direction: `To Meta View`
- Entering the selection criteria: `(%cv.EMPLOYEE_ID%=@)`

  This specification means that any entry with a connector view value of EMPLOYEE_ID will be selected for these mappings.
- Adding attribute mappings

  `EMPLOYEE_ID`, `FIRST_NAME`, `LAST_NAME`, `SALARY`, `DEPARTMENT_ID`, `dbobjectclass` and `dbcommonname` map respectively to `uid`, `givenName`, `sn`, `description`, `departmentnumber`, `objectclass` and `cn`.

This attribute flow rule is then put in an attribute flow rule set entitled `oracle2mvset`. An attribute flow rule going back to the Oracle column is also created (using the selection criteria `%uid%=@`) and put in a rule set entitled `oracle2cvset`.

Creating DN Mapping Rules for the Oracle Database

Tracy can now configure rules that will create DNs for the Oracle data. Three rules are needed to flow the data into the meta view's three organizational units. A DN rule for Writers is created by:

- Naming the rule: `oraclewriter2mv`
- Entering the selection criteria: `((%cv.EMPLOYEE_ID%=@) AND (%cv.DEPARTMENT_ID%==10))`

  This specification means that any entry with a connector view value of EMPLOYEE_ID and a department ID of 10 will be selected for this DN.
- Constructing an DN: `%cv.EMPLOYEE_ID%, ou=Writers`

This DN rule creates the organizational unit *Writers* in which all entries with an employee ID and a department ID of 10 will be placed. Tracy must now configure a different rule for accountants and designers. The three completed rules that apply to data flowing into the meta view are put into one rule set entitled `DNrulesoracle2mvset`.

Tracy then makes three more rules which flow the information back to the connector view, using as selection criteria `%mv.EMPLOYEE_ID%=@` and constructing a DN of `%mv.EMPLOYEE_ID%`. Tracy puts them in a rule set entitled `DNrulesoracle2cv`. The two rule sets `DNrulesoracle2mv` and `DNrulesoracle2cv` will flow the LDAP data into and out of the meta view.

Integration of the Data With Join Rules

With the bulk of the rules configured for the three data sources, Tracy is now able to flow the data into the meta view. Join rules need to be configured to synchronize the data correctly as each data source has different information flowing to the same meta view entry. Tracy needs to configure join rules that look at all source entries and link only those that have passed certain criteria. Tracy creates a join rule for the Oracle database by:

- Naming the rule: `oraclecv2mv`
- Entering the selection criteria: `(%cv.EMPLOYEE_ID%=@)`

  This specification means that any entry with a connector view value of EMPLOYEE_ID will be selected for joining.
- Constructing the join filter: `cn=%cv.FIRST_NAME% %cv.LAST_NAME%`

Tracy puts the Oracle join rule into a join rule set called `oraclejrcv2mvset`. For the LDIF data source the join rule is configured by:

- Naming the rule: `ldifcv2mv`
- Entering the selection criteria: `(%objectclass%==person)`

  This specification means that any entry with an object class equal to `person` will be selected for joining.
- Constructing the join filter: `cn=%cn%`

Tracy creates a join rule set called `ldifjrcv2mvset` and the LDIF join rule is put into it. Tracy then realizes that the join rule created for LDIF data will also work for the CSV data source because they both have been converted into LDAP. Now Tracy is ready to apply the join process rules.

Applying Rules to the Participating View

Tracy applies the join process rules to the correct participating views. (The participating view allows the connector view to 'participate' in the join process.) From Meta-Directory console, a participating view for the `firstgroup`, `secondgroup` and `oracle` connector views is added. It is here that the join process rules are applied as follows:

Participating View	Join Process Rule	Rule Set Name
firstgroup	Attribute Flow To Meta View	affg2mvset
	Attribute Flow To Connector View	`affg2cvset`
	DN Mapping To Meta View	`DNrulesfg2mv`
	DN Mapping To Connector View	`DNrulesfg2cv`
	Filters	none configured
	Join Rules To Meta View	`ldifjrcv2mvset`
secondgroup	Attribute Flow To Meta View	`afsg2mvset`
	Attribute Flow To Connector View	`afsg2cvset`
	DN Mapping	`DNrulessg2mvset`
	Filters	none configured
	Join Rules To Meta View	`ldifjrcv2mvset`
oracle	Attribute Flow To Meta View	`oracle2mvset`
	Attribute Flow To Connector View	`oracle2cvset`
	DN Mapping	`DNrulesoracle2mvset`
	Filters	none configured
	Join Rules To Meta View	`oraclejrcv2mvset`

Once the join engine is started, the connectors are enabled and the views are refreshed, Siroe's LDIF data, CSV data and Oracle data will flow to the meta view forming one complete source of employee information.

At this point, Tracy checks the Meta-Directory error logs to see if any of the data has failed to flow or link. He does this by issuing the following `grep` command in the log directory:

```
grep -i fail *.log
```

Had any errors been found, Tracy could use the Fix-It Tool to manually link the orphaned entries. Fortunately for him no errors were found so Tracy steps outside for lunch.

Chapter 3

Quick Start

Follow these general steps to get Meta-Directory up and running.

1. Install Meta-Directory and its components.

   Meta-Directory is used to establish an interdependent relationship between two or more sources of data, allowing the data to be integrated, viewed, searched and/or modified from one authoritative source. The components used to join the data include the join engineand connectors (both direct and indirect). During installation, you will be asked to choose a Directory Server and a path to a directory on the Directory Server where the Meta-Directory configuration will be stored. For more information on installation, see the *iPlanet Meta-Directory Installation Guide.*

2. Start iPlanet Console, your server group's Administration Server and your external data source.

   iPlanet Administration Services, which includes iPlanet Administration Server and iPlanet Console, was installed along with Meta-Directory. iPlanet Administration Server (accessed through Administration Server console) is the common front-end for all iPlanet Servers. iPlanet Console manages all installed iPlanet server groups. For instance, you might install Directory Server in one directory tree and Meta-Directory in another. Each server group will have its own Administration Server but, iPlanet Console allows the management of both server groups from a single graphical user interface. Similarly, you access Meta-Directory from iPlanet Console although the initial Meta-Directory configuration (step 3 and step 4) is achieved through iPlanet Console. For more information on iPlanet Console, see Chapter 4, "The Console Interface" or the iPlanet Console and Administration Server documentation set.

3. Create an instance of the join engine.

   The join engine manages the flow and integration of data from different sources into one meta view. An instance of the join engine needs to be configured to create the meta view. The URL of the Directory Server where the meta view will be located as well as an authenticating DN and password are needed for instance creation. Once the instance is created, a meta view is created automatically in the navigation tree of Meta-Directory console. The instance is created from iPlanet Console and Meta-Directory console can be opened from the instance in iPlanet Console. For more information on the join engine, see Chapter 5, "Configuring The Join Engine."

4. Connect one or more sources of external data to Meta-Directory by creating one or more instances of a server.

   An indirect or direct connector is used to flow data from the external data source to the connector view. Connectors allow information to flow in two directions: original or modified data can move from the data source to the connector view **and** modified data can flow from the connector view back to the data source. You associate external data with a connector view in one of two ways:

   a. Create an instance of an indirect connector, configure and apply the indirect connector rules and start the instance.

      Any external data source that goes through the process of having its data transformed into LDAP *before* flowing through the join engine is considered to have an indirect connection to the join engine. Active Directory, NT Domain and Universal Text Parser are indirect connectors which translate proprietary data into LDAP. When an instance of one of these indirect connectors is created (from iPlanet Console, not Meta-Directory console), it links to an external data source and creates a connector view on a Directory Server with which to view the data. Starting the indirect connector instance begins the flow of data from the external data source to the connector view. (It is important to remember that creation of the instance does not flow data; it only opens a data path.)

      Meta-Directory provides rules to manage the flow of data from the external data source through the indirect connector and to the connector view. These *indirect connector rules* can alter the flow in three ways: by filtering entries from the process, by setting entry ownership, and by matching attributes in the data source to attributes in the connector view. Other indirect connector configuration choices include setting up log options, scheduling synchronization, choosing from available attributes,

and, in the case of the Universal connector only, adapting the connector with a Perl script to help it to synchronize a proprietary data source. Indirect connector rules are configured at the connector level on the navigation tree but are applied to specific connectors separately.

For information on creating and applying indirect connector rules, see Chapter 7, “Connectors and Connector Rules.” For information on creating an indirect connector instance specific to your data source, see the corresponding connector’s chapter in Part 3, “Meta-Directory Connectors.” For information on starting and stopping components, see Chapter 13, “Starting and Stopping Components.”

b. Create a direct connector view and start it.

The join engine allows access to LDAP Directory Servers and SQL (Oracle) databases within its architecture; these data sources are considered to have a direct connection. A direct connector is not needed; only a connector view need be created. (As there is no connector, the indirect connector rules do not apply either.)

To create a direct connector view, a Directory Server or an Oracle database needs to be recognized by the join engine. You configure a Directory Server by inputting address information (host name, port, DN name and password), scheduling operation and data modification times and setting up binary attributes. An Oracle database is also configured with database address information with the addition of changelog scripts, operation and data modification times and column selections. Once the Data Servers are connected to Meta-Directory, the direct connector view can be instantiated by pointing it to the server.

For more information on configuring for connection to these types of data sources, see “Configuring Data Servers,” on page 297 of Chapter 17, “Managing Servers and Permissions.” For more information on creating an LDAP connector view, see “Connector Views,” on page 103 of Chapter 6, “Views in Meta-Directory.” For information specific to connecting to a database, see Chapter 12, “Configuring the Database Connector.”

5. Define the join process rules.

The key to managing the manipulation of data from the connector view to the meta view is the join process. This process is the flow of data through the join engine and the data’s synchronization within the meta view. Administrators write join process rules to take full advantage of Meta-Directory’s synchronization tools. There are five types of rules: join rules, attribute flow rules, DN mapping rules, filters and attribute construction rules. (Don’t confuse these rules with the similarly-named indirect connector rules which

manage the flow of data between the external data source and the connector view.) Generally, these rules decide which entries (or attributes) from the connector view will exist in the meta view. Specifically, the administrator can, among other things, prevent specific entries from flowing back and forth and specify which attribute values in the connector view map to which attribute values in the meta view.

The join process rules written at this step will not actually manage the data until they are specifically applied to a participating view in Step 6. For information on creating rules for the join process, see "Creating the Join Engine Instance," on page 64 of Chapter 5, "Configuring The Join Engine."

6. Configure a participating view and apply the join process rules.

   A participating view must be added before data will flow from the connector view into the meta view. Once added and enabled, the participating view is telling the join engine that the corresponding connector view is 'participating' in the data flow. Applying the join process rules to the participating view will manage data flow between the connector view and meta view.

   The previously-configured join process rules are applied to the participating view after it has been added to the navigation tree. If no rules have been configured, the participating view uses *Atomic* rules. (The Atomic rules are a set of default DN Mapping rules and Attribute Flow rules that allow data to flow in the absence of previously-configured join process rules.) In addition to the join process rules, the direction of data flow, a data refresh schedule and data refresh filters can be applied to a participating view.

   For more information on participating views, see Chapter 6, "Views in Meta-Directory" where you can find a section called "Participating Views," on page 109 of Chapter 6, "Views in Meta-Directory." For information on applying the join process rules, see "Configuring a Participating View," on page 111 of the same chapter.

7. Start the join engine.

   The join engine is started from either iPlanet Console or Meta-Directory console. Even when the join engine is started, data will not flow until the participating view and the connector view are enabled. For information on starting the join engine, see Chapter 13, "Starting and Stopping Components."

8. Enable the participating view.

   The participating view is enabled from the join engine's Operations window in Meta-Directory console. When you enable the participating view, its connector view will flow data into the meta view, using the join process rules applied to it. For information on enabling the participating view, see "To Enable a Participating View," on page 119 of Chapter 6, "Views in Meta-Directory." For information on the join engine's Status options, see "Join Engine Operations," on page 251 of Chapter 14, "Monitoring Meta-Directory Components."

9. Ensure that the connector view is enabled.

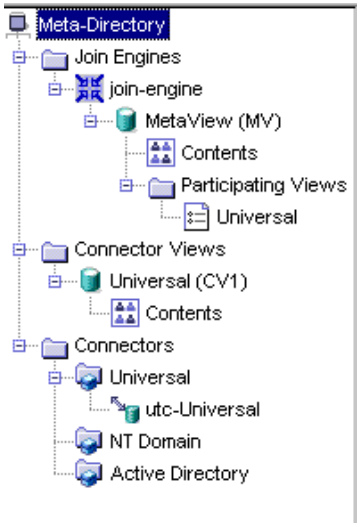
   By default, the connector view is always enabled so data flow from the external source to the meta view is always possible, providing the prior eight steps have been executed correctly. The connector view's status can be verified from the connector's Operations window in Meta-Directory console. For information on enabled connector views, see "An Enabled Connector View," on page 107 of Chapter 6, "Views in Meta-Directory." For information on the connector's Status options, see "Connector Operations," on page 254 of Chapter 14, "Monitoring Meta-Directory Components."

10. Refresh the data.

    Refreshing the data ensures that the most recent data is flowing. This step is most important when changes and modifications have been made to data. For information on how to refresh connector and meta views, see "Refreshing the Views," on page 120 of Chapter 6, "Views in Meta-Directory."

Figure 3-1 shows how the Meta-Directory navigation tree appears if you were to create a Universal connector and its corresponding connector view, then enabled it as a participating view.

Figure 3-1 Meta-Directory Navigation Tree With a Configured Connector View



Chapter 4

The Console Interface

Most iPlanet products use the console graphical user interface for navigating and configuring iPlanet software. The console always takes the same form and layout. The product/console name can be found in the top left hand corner. The *navigation tree* can be found in the left panel of the console and the product features interface in the right panel. This panel will change depending on what is clicked in the navigation tree. This chapter describes the different consoles and how to recognize and customize them. This chapter contains the following sections:

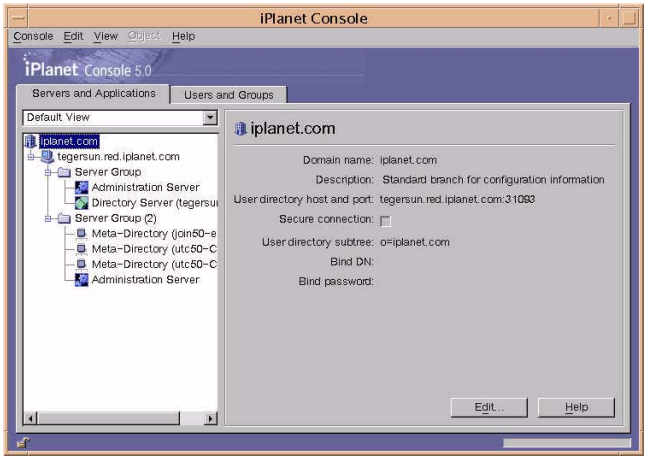
- iPlanet Console
- iPlanet Administration Server Console
- iPlanet Directory Server Console
- iPlanet Meta-Directory Console

NOTE The consoles for all iPlanet products are the iPlanet Console application with integrated, product-specific JAR files.

iPlanet Console

iPlanet Console is the front-end Java™ application for the management of all iPlanet software in your enterprise. It finds all iPlanet servers and applications registered in your configuration directory, displays them in a navigation tree, and allows you to manage and configure them. When you log in to iPlanet Console, it connects to a server group's Administration Server using the Hypertext Transfer Protocol (HTTP) although the two function independently. For more information, see the documentation for iPlanet Console and Administration Server 5.0, Managing Servers With iPlanet Console. Figure 4-1 shows iPlanet Console's interface.

Figure 4-1 iPlanet Console



To Access A Console

1. Double-click the icon representing the Administration Server, Directory Server, or Meta-Directory from the navigation tree on the left of iPlanet Console.

   The interface for that product will appear in the right panel.

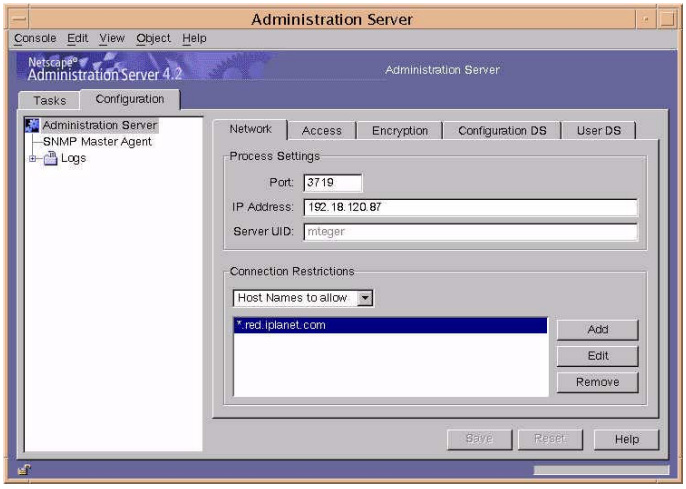
2. Press Open in the upper right corner of the component information panel.

   This will open up the console for that component in a new window.

iPlanet Administration Server Console

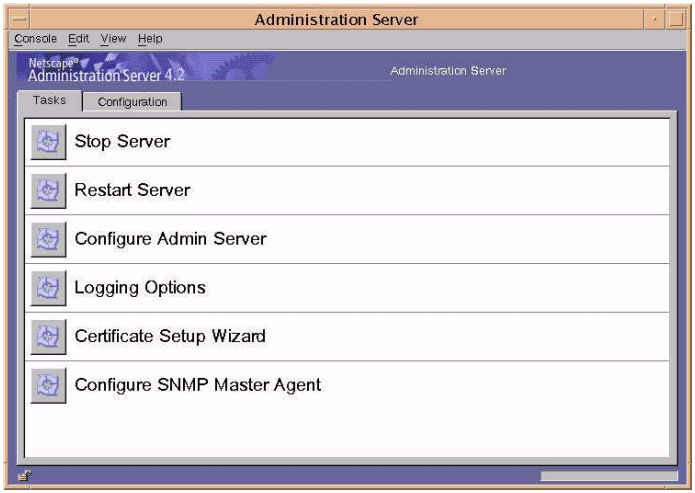
iPlanet Administration Server processes requests for servers that are installed in a server group (a single root folder), and then invokes the programs required to fulfill them. When you install an iPlanet product in a new folder, Administration Server is also installed for you. If, later, you install additional products in the same folder, they use the instance of Administration Server that is already there. If a product includes a newer version of Administration Server, the installer updates with the latest versions. The iPlanet Administration Server console offers navigation and feature options for the Administration Server software. For more information, see the documentation for iPlanet Console and Administration Server. Figure 4-2 shows the iPlanet Administration Server's interface.

Figure 4-2 iPlanet Administration Server's Interface



The Administration Server console also includes a convenient interface for starting and stopping the server as well as other tasks. This interface can be found by clicking the Tasks tab at the upper left corner of the window.

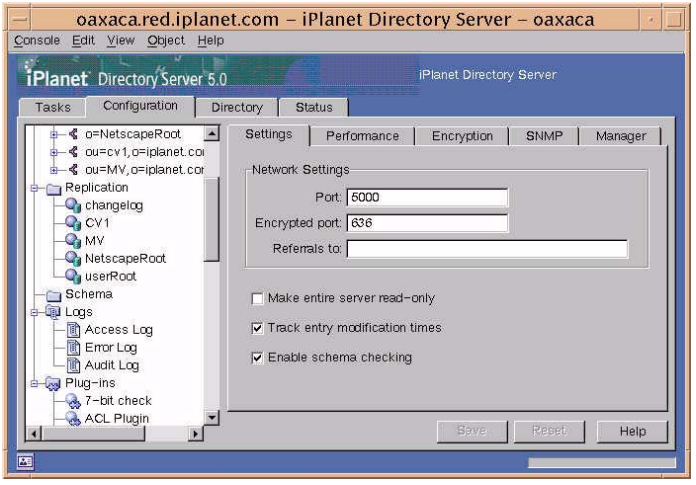
Figure 4-3 iPlanet Administration Server Task Tab



iPlanet Directory Server Console

You can perform most Directory Server administrative tasks from iPlanet Directory Server console. The iPlanet Directory Server console (pictured in Figure 4-4) offers navigation options as well as the task interface pictured in Figure 4-3. For more information, see iPlanet Directory Server documentation.

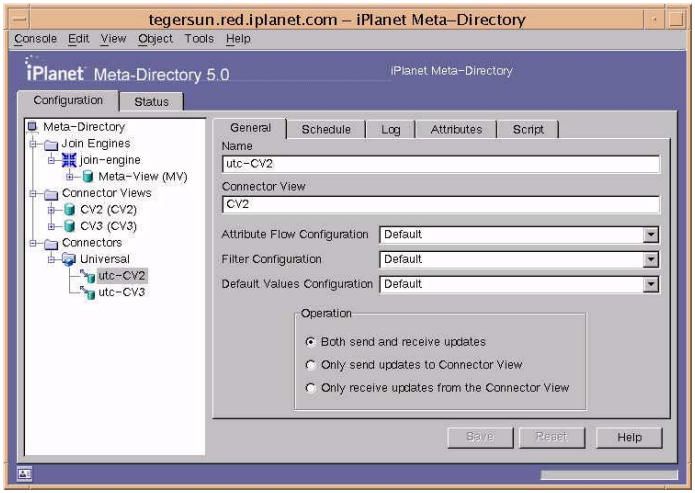
Figure 4-4 iPlanet Directory Server Console



iPlanet Meta-Directory Console

The iPlanet Meta-Directory console offers management, configuration, navigation and feature options for iPlanet Meta-Directory software. iPlanet Meta-Directory console is pictured in Figure 4-5.

Figure 4-5 iPlanet Meta-Directory Console



Menus

Meta-Directory console has a number of features that are accessible from within the graphical user interface as well as from the menus at the top of the console.

Console

From the Console menu, you have the following choices:

- Logon As A New User...

  When provided with a new DN and password, your current session will end and the console will re-open with another user as administrator.

- Close

  This option closes the Meta-Directory window.

- Exit

  This option ends your session with Meta-Directory.

Edit

From the Edit menu, you have the following choices:

- Preferences

  From the Console Preferences window, you can specify the directory where your settings will be stored as well as your choice of font for the console window.

- Cut

  The standard command for use with Meta-Directory property editing tools.

- Copy

  The standard command for use with Meta-Directory property editing tools.

- Paste

  The standard command for use with Meta-Directory property editing tools.

- Delete

  The standard command for use with Meta-Directory property editing tools.

- Copy DN

  Choosing an entry from one of the Contents sub-trees in the navigation tree, selecting an entry in the right window and choosing Copy DN will copy the Distinguished Name of that entry to the Clipboard for pasting it into another field.

View

From the View menu, you have the following choices:

- Banner Bar

  The Banner Bar runs across the top of the console and contains the Meta-Directory logo. Choose this option to show or hide the Banner Bar.

- Status Bar

  The Status Bar runs along the bottom of the console and allows you to view the status of an operation. Choose this option to show or hide the Status Bar.

- Show Main Console

  Choose Show Main Console to bring the main iPlanet Console interface forward.

- Tree

  Choose this option to show or hide the navigation tree on the left side of the console.

- Refresh

  Choose this option to refresh the information in the navigation tree. When a new view is added, the console should be refreshed to access the latest information. For additional information, see “Refreshing the Views,” on page 120 of Chapter 6, “Views in Meta-Directory.”

NOTE The Refresh command always refreshes the entire navigation tree, even when you access the command by right-clicking on the components in the navigation tree.

Object

From the Object menu, you have the following choices:

- Open

  When the contents folder of a connector view is selected in the navigation tree, the Open command opens up the Property Editor. For more information, see “The Property Editor,” on page 286.

- Search

  When the contents folder of a connector view is selected in the navigation tree, choose this option to search users and groups. For more information, see “The Query Tool,” on page 267.

- New

  When the contents folder of a meta view is selected in the navigation tree, choose New to create a new User, Group or Organization.

CAUTION The option of creating new users within the connector view is available for testing purposes only; for instance, testing the flow between the external data source and its connector view. Once the join engine and meta view have been configured, new entries should only be added from within the meta view or the external data source, not from within the connector view.

- Start Server

  To start a Meta-Directory component, select it in the navigation tree and choose Start Server.

- Stop Server

  To stop a Meta-Directory component, select it in the navigation tree and choose Stop Server.

- Attribute Flow Summary

  Select meta view in the navigation tree to see a summary of the attribute flow rules used by a specific view as described in “Attribute Flow Summary,” on page 91 of Chapter 5, “Configuring The Join Engine.”

Tools

From the Tools menu, you have the following option:

- Query

  The Query Tool is used to generate searches of Meta-Directory entries as discussed in “The Query Tool,” on page 267 of Chapter 15, “Administration Tools.”

Help

From the Help menu, you have the following options:

- Contents

  This option links you to the contents page for the help available within the Meta-Directory.

- Index

  This option brings up a Search Index for the help files available within the Meta-Directory.

- Documentation Home

  This option brings you to the home page for Meta-Directory 5.0 Documentation.

- About

  This option brings up copyright and trademark information related to the Meta-Directory product.

Part 2

The Join Engine

Chapter 5

Configuring The Join Engine

This chapter provides information on the join engine, the core component of Meta-Directory. The primary purpose of the join engine is to link data and manage the flow of information from the connector view, into the meta view and back to the connector view. This chapter includes the following sections:

- What is the Join Engine?
- Creating the Join Engine Instance
- What is the Join Process?
- Configuring the Join Process
  - Join Rules
  - Constructed Attributes
  - Attribute Flow Rules
  - Filters
  - Distinguished Name Mapping Rules
  - Data Servers

What is the Join Engine?

The join engine is responsible for directing the flow and synchronization of data between one or more connector views and the meta view. It allows data to flow in either direction, monitoring the connector views for changes and incorporating these changes into the meta view as well as monitoring the meta view for changes and incorporating these changes back into the connector views. For example, if information from a human resources database that contains a change in an employee's address enters the join engine, the join engine relays the change to the meta view. The change will be made in the address attribute of the user's meta view entry as well as flow back to reflect the change in other connector views the entry might also be linked with.

Creating the Join Engine Instance

You create one instance of the join engine within a server group. (A server group consists of one or more servers that share a root directory and are managed by a common instance of Administration Server.) The instance is created from iPlanet Console. General server information, such as the base DN of the meta view and Directory Server URL, is needed to create the instance. It also requires enabling the Directory Server change log as well as loading the schema into the Directory Server. (The schema should be loaded into any new Directory Server introduced into the system.) Once the join engine instance is created, Directory Server should be restarted and Meta-Directory console can be accessed.

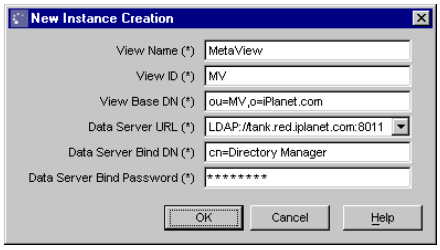
To Create a Join Engine Instance

1. In iPlanet Console, right-click the Server Group.

   A context menu appears.

2. Select Create Instance Of, then select Meta-Directory Join Engine.

   The New Instance Creation dialog box appears.



3. Enter values for all of the following fields:

Fields	Values
View Name	Enter a name for the meta view associated with this join engine.
View ID	Enter no more than five characters to represent a view ID that is unique within the system. The default is MV.
View Base DN	Enter the base DN under which the view's information will be stored. If this suffix does not exist, the default created is `o=MV`. Creating the root suffix on the Directory Server before instantiating the join engine is recommended.
Data Server URL	Select a Directory Server on which the meta view will be stored from the drop-down list, or type in a new one.
Data Server Bind DN	Enter a DN with which the join engine will authenticate to the Directory Server.
Data Server Bind Password	Enter the password associated with the authenticating DN.

4. Click OK.

   A pop-up window appears with change log options. A change log is a file that summarizes changes made to a server. Directory Server writes the change logs in a subtree (cn=changelog).

5. Select one of the following options concerning the change log:

   - Enable Changelog NOW.

     The Enable Changelog dialog box will appear. Enter the directory path where you want to store the change log and accept the default Changelog Suffix or provide your own. Click OK. (You will be reminded to restart the Directory Server. For information on how to do this, see the Deployment Guide.)

   - Enable Changelog LATER.

     It is recommended that you enable the change log now. The process for doing this in Directory Servers 4.1x and 5.0 is different. Please see iPlanet Directory Server documentation for enabling the change log after the instance has been created.

   - Abort Instance Creation.

     This option will stop the instance creation process.

   NOTE In the Solaris operating environment, Directory Server is normally installed as root. Because of this, the permissions of the directory need to be changed to allow console to create the change log directory. Apply a command of `chmod -R 777` against the directory where you wish to create the change log.

6. Click Yes to load the schema into the Directory Server.

   Loading the schema requires several minutes. After it is loaded, the message "Instance Creation Succeeded" appears.

   NOTE Meta-Directory cannot detect if the schema has been loaded into Directory Server so it asks whether to load the schema every time an instance is created. Every Directory Server should have the schema loaded at least once. Loading it more than once does not harm the configuration.

To Remove a Join Engine Instance

1. In iPlanet Console, right-click the join engine instance.

   A context menu appears.

2. Select Remove Server and click Yes in the message box.

   The join engine instance is deleted.

What is the Join Process?

The join process is the movement of data through a sequence of rules and filters, configured by the administrator and administered by the join engine, to determine how connector view entries will be linked in the meta view. To successfully join connector view entries, the join engine must match each connector view value with a corresponding value in the meta view. To do this, rules containing values and attributes are used as criteria for finding and creating links. The rules include join rules, attribute construction definitions, attribute flow rules, filters, and DN mapping rules.

Join Process Rules and Rule Sets

Before beginning configuration, you should be familiar with the concept of join process rules and rule sets. Join process rules contain values and attributes that are used as criteria for moving data through the join engine and linking the data in the meta view. For the join process, rules are configured and then placed in rule sets. These sets are applied when setting up the participating view. After they are applied, the rules are used by the join engine to steer the join process.

Rule Sets

A rule set contains an ordered group of configured rules which are sequentially tested until either one of two things occur:

- An individual rule identifies an entry
- All rules fail and the entry is not passed

It is important to remember that each rule set should be made up of rules that search through entries flowing in one direction. In other words, do not add a rule that searches through entries flowing to the meta view to a rule set that already contains a rule that searches through entries flowing to a connector view.

Join Process Rules

A configured rule consists of one or more of these three operator strings:

- Optional Token Assignments (Format Operator)
- Selection Criteria (Requirements Operator)
- Distinguished Name Construction / Join Filter / Attribute Construction (Substitution Operator)

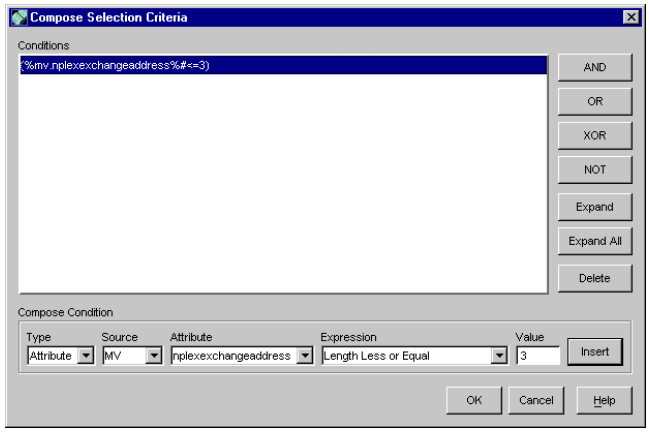
The operators are executed in the order stated. The Format Operator is recognized first so that sub-tokens can be created. The entries are then checked by the Requirements Operator. The entries that contain the proper selection criteria are then searched by the Substitution Operator for those that match the query. For additional information regarding these operators, see Appendix A, “The Join Process Operators.”

Optional Token Assignments (Format Operator)

The format operator string is an optional field that specifies the format for attributes which consist of multiple values. It is is used to break up an attribute’s value into separate, more specific values. Values can be divided into sub-tokens for use in the configuration of other rules. For example, the value of the attribute `{cn}=%last,first%` can be separated into two values by entering `{cn}=%last%,%first%` in the optional token assignment field. These sub-tokens (`%last%` or `%first%`) are then used as attributes for placement in other rules. (When this field is left empty, all pairs default to their entire source value.)

Selection Criteria (Requirements Operator)

The requirements operator string specifies conditions that are applied to a source entry; the conditions must evaluate to TRUE for the entry to pass to the next operator. Join rules, DN mapping rules, constructed attributes and attribute flow rules include the Selection Criteria option. The Compose Selection Criteria dialog box configures the specifications which are used to determine if the particular rule for which they are configured should be applied against the entry being flowed. If the entry does not meet the required condition (for instance, `objectclass==person`), the entry will not synchronize.



Selecting the option to make a new rule from within the join rules, DN mapping rules, constructed attributes or attribute flow rules screens and clicking the (. . .) button will open the Compose Selection Criteria dialog box. This dialog box enables you to set conditions against entries or attributes:

- Entry Conditions

  The Entry type enables you to compose criteria that would select or not select entire entries.

- Attribute Conditions

  The Attribute type allows you to compose conditions that would define your selection criteria as the value of an attribute.

- Language Tagged Attribute Conditions

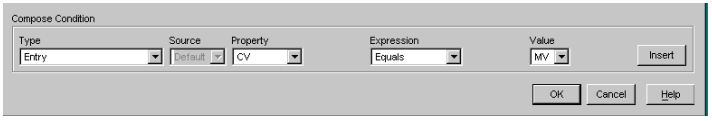
  The Language Tagged Attribute type enables language subtype support. If you specify a language subtype for an attribute, the entry allows searches based on a secondary, specified language.

The following sections explain how to configure each type of condition.

To Compose Entry Conditions

1. In the Compose Selection Criteria dialog box, select Entry as the Type.

   This figure shows the available fields for the Entry conditions.



2. Provide values for the following fields:

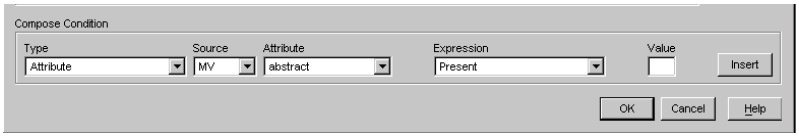
Property	Select Operation and the condition selects entries based on whether or not the join, add, update, or delete action is true or false. Select CV or MV and the condition selects entries based on where the entry will flow from or to. The entry will synchronize if the value ID is equal or not equal to the property ID.
Expression	Select an expression to describe the relationship between the Property and the Value. The Expression drop-down list contains two choices: Equal and Does Not Equal.
Value	Provide a value to complete the expression.

3. Click Insert. Your composition appears in the Conditions list window above.
4. Optional: Add other conditions as necessary.
5. Optional: Modify the conditions using the buttons at the right as described in “Combining Conditions,” on page 74.
6. Click OK to save the conditions and return to the previous dialog box.

To Compose Attribute Conditions

1. Select Attribute as the Type.

   This figure shows the available fields for the Attribute conditions.



2. Provide values for the following fields:

Source — Choose the source for the attribute from the drop-down list. The list includes all configured connector and meta views plus Default. Choose Default if you want all sources.

Attribute — Choose an attribute used in the source schema.

Expression — Select an expression that describes the relationship between the attribute and the value. The expressions in the drop-down list are self-explanatory except for these two:

- Constructed From is for use with constructed attributes. As an example, the following constructed attribute is defined
  `description = %title%,%telephonenumber%`
  To specify an attribute `telephonenumber`, an Expression of `Constructed From`, and a value of `description` would be used. When the join engine applies the selection criteria, it searches for `description` to specify telephonenumber.
- Free Format appends to anything you type in the Value field. Do not use Free Format to type `description @ AND cn @` in the Value field. These values should be selected using the drop-down lists. (When using this option, be aware of your use of the space bar.)

Value — Provide a value to complete the expression, if applicable.

3. Click Insert. Your composition appears in the Conditions list window above.
4. Optional: Add other conditions as necessary.

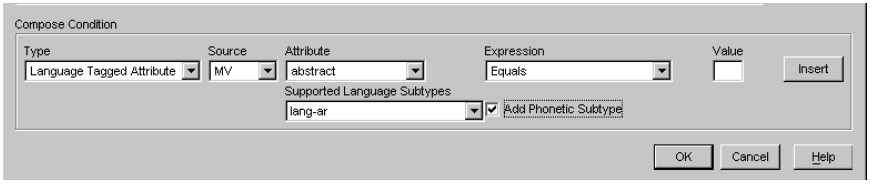
5. Optional: Modify the conditions using the buttons at the right as described in “Combining Conditions,” on page 74.

6. Click OK to save the conditions and return to the previous dialog box.

To Compose Language Tagged Attribute Conditions

1. Select Language Tagged Attribute as the Type.

   This figure shows the available fields for the Language Tagged Attribute conditions.



2. Provide values for the following fields:

Source	Choose the source for the attribute from the drop-down list. The list includes all configured connector and meta views plus Default. Choose Default if you want all sources.
Attribute	Choose an attribute used in the source schema.

Expression	Select an expression that describes the relationship between the attribute and the value. The expressions in the drop-down list are self-explanatory except for these two: • Constructed From is for use with constructed attributes. As an example, the following constructed attribute is defined `description = %title%,%telephonenumber%` To specify an attribute `telephonenumber`, an Expression of `Constructed From`, and a value of `description` would be used. When the join engine applies the selection criteria, it searches for `description` to specify telephonenumber. • Free Format appends to anything you type in the Value field. Do not use Free Format to type `description @ AND cn @` in the Value field. These values should be selected using the drop-down lists. (When using this option, be aware of your use of the space bar.)
Value	Provide a value to complete the expression, if applicable.
Supported Language Subtypes	Select a language subtype. For more information, see “Language Subtype Tagged Attributes,” on page 92.
Add Phonetic Subtype	Click this checkbox to indicate that the attribute value is a phonetic representation. For example, `givenname;lang-ja;phonetic` indicates that the attribute value is the phonetic version of the entry's Japanese name.

3. Click Insert.

   Your composition appears in the Conditions list window above.

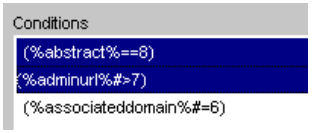
4. Optional: Add other conditions as necessary.

5. Optional: Modify the conditions using the buttons at the right as described in “Combining Conditions,” on page 74.

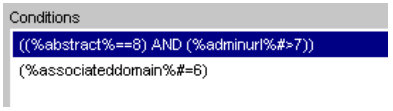
6. Click OK to save the conditions and return to the previous dialog box.

Combining Conditions

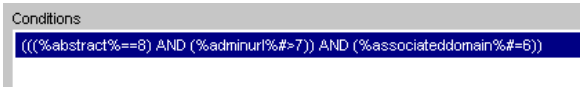
You can combine configured conditions at any time using one or more of the operator buttons on the right side of the window. For instance, to combine the first two of the following conditions with an AND operator, you would select them and click AND.



The system would respond as shown:



To add the third condition with an AND operator and nest the first two, you would select the already combined conditions and the third condition and click AND. The system would respond as shown:



For definitions of the combining operator buttons, see “Combining Requirements Operators,” on page 311.

NOTE If single requirements are inserted without the use of the logical operator buttons, the requirements appear in the Selection Criteria box delineated by a semi-colon. The semi-colon is treated as a logical AND, i.e.: all requirements must evaluate to TRUE for the entry.

Distinguished Name Construction/Join Filter/Attribute Construction (Substitution Operator)

The final operator string is the substitution operator. The substitution operator contains the conditions that the join engine applies to the target view in order to find one entry with which to link the source entry. This is the required entry when defining rules. The join engine will apply the substitution operator criteria to the data in the destination source and join the entry flowing to that source with the one entry in the destination source that matches the criteria. For example, an entry flowing from the connector view has a `uid=agreen`. A substitution operator of `uid=%uid%` will join this entry with the one entry in the destination source that also has a `uid=agreen`. The substitution operator will find only one unique entry to match or else the rule will fail. The rule will also fail if this field is left empty.

NOTE See Appendix A, “The Join Process Operators” to learn more about this syntax.

Configuring the Join Process

The configuration of the join rules, attribute construction definitions, attribute flow rules, filters, and DN mapping rules is the basis of the join process. After these configurations are finished, the rules are placed in rule sets and applied to the data. Based on the rule configurations, the join engine can allow one or more of the following:

- Entry creation in the meta view or in the external data sources
- Entry deletion from the meta view, and subsequently, from the external data sources in which the entry originated or to which it is connected
- Entry modification in the meta view and the corresponding entry in any number of external data sources
- Incremental updates in the external data sources as modifications occur

Once the join process rules are configured they need to be applied to the participating view in order to modulate the process. Information and procedures on how to apply these rules can be found in “Configuring a Participating View,” on page 111 of Chapter 6, “Views in Meta-Directory.”

Join Rules

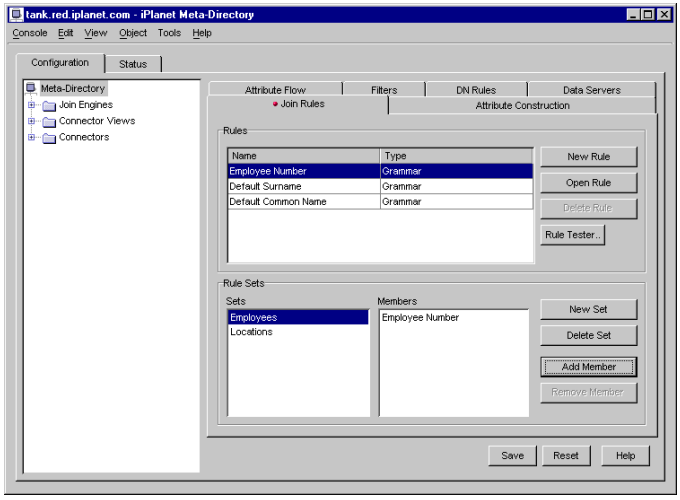
A *join rule* is one of the join process rules. To successfully join entries from different connector views into one meta view entry, join rules are specified. Join rules identify which values and attributes in the source entry will be used to search the destination view for one entry with which to join it. When a matching value is found in the destination view, a join between the two entries is created.

After join rules are configured, they are placed in rule sets which define the order in which the rules are applied. The join engine applies the rules sequentially within the rule set until either a rule identifies a single entry or all rules fail. If a match is made, the join engine moves on to the next rule set in the join process. If the entire search fails, the join engine may still add the source entry to the destination view, depending on the configuration of the other rules. Alternatively, the join engine can leave the source entry unlinked and the entry can be manually joined using The Fix-It Tool as described in Chapter 15, “Administration Tools.”

Working With Join Rules

1. From the Meta-Directory console, choose the Configuration tab.
2. Select Meta-Directory in the navigation tree.
3. Click the Join Rules tab.

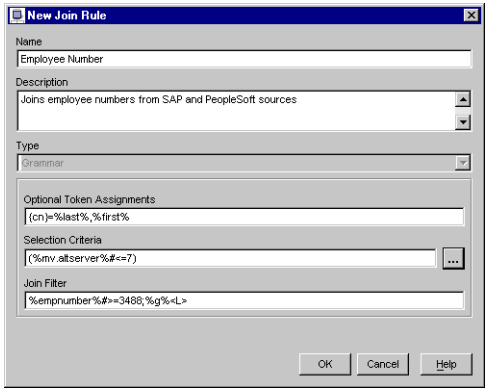
   The join rules window appears.



To Create New Rules

1. Click New Rule.

   The New Join Rule dialog box appears.



2. Provide values for the following fields:

Name	Enter a name for the join rule. Restrict the name to the following characters: `A-Z a-z 0-9 _ - <space>`
Description	Optional. Enter a description.
Type	The default rule type is Grammar. (The join engine provides condition and substitution statements that are based on attribute names and values.) Additional script-based rule options may be available through iPlanet Professional Services.
Optional Token Assignments	This field is used to break up an attribute's value into separate, more specific values. These sub-tokens can then be used as attributes. For example, the attribute `{cn}=%last,first%` can be sliced into two attributes by entering `{cn}=%last%,%first%`. To compose token assignments, see "Optional Token Assignments (Format Operator)," on page 68 for instructions.

Selection Criteria	Optional. This field determines the conditions that must be met by an entry before a rule is applied. For instance, you can choose to flow only those entries that contain (`%objectclass%=person`); using this field parses the entries using that standard. To compose selection criteria, see "Selection Criteria (Requirements Operator)," on page 69 for instructions.
Join Filter	This required field is the search filter used to join entries. The join engine will apply the rule to the destination data and join the source entry with the one destination entry that matches the criteria. For example, a join filter of `uid=%uid%` will join a connector view entry and a meta view entry whose unique ID match. To compose join filters, see "Distinguished Name Construction / Join Filter / Attribute Construction (Substitution Operator)," on page 75 for instructions.

3. Click OK.

   The new rule and its type appear in the Rules list box.

4. Repeat the steps above to add other join rules.

5. Click Save from the Join Rules window when you are finished.

Example:

In order to join entries that have the same values for the attribute `uid`, you would tell the join engine to select all entries with the attribute uid [`(%uid%=@)` in the Selection Criteria field] and join those that have an identical user ID [`(uid=%uid%)` in Join Filter field].

To Edit Rules

1. Select the rule you want to edit from the Rules list box and click Open Rule.

2. Alter the fields in the Edit Join Rule dialog box as described in "To Create New Rules," on page 77.

3. Click `OK` to save your changes and return to the Join Rules window.

4. Edit other rules following steps 1 - 3.

5. Click Save from the Join Rules window when you are finished editing rules.

To Delete Rules

1. Select the rule you want to delete.
2. Click Delete Rule.

   The rule disappears from the Rules list box.
3. Click Save.

To Create A New Rule Set Name

1. Click New Set.

   The New Set dialog box appears.
2. Enter the new rule set name and click OK.

   The new name appears in the Rule Sets list box.
3. Click Save.

To Assign Members to a Rule Set

1. Select the rule set name from the Rule Sets list box.
2. Select the rule you want to add from the Rules list box.
3. Click Add Member.

   The rule you add appears in the Members list box.
4. Repeat steps 1 - 3 to add more rules to the set
5. Click Save when you are finished.

To Delete Members from a Rule Set

1. Select the rule set name from the Rule Sets list box.
2. Select the member you want to delete from the Members list box and click Remove Member.
3. Repeat step 2 to delete other members.
4. Click Save when you are finished removing members.

To Delete a Rule Set

1. Select the set you want to delete.
2. Click Delete Set.

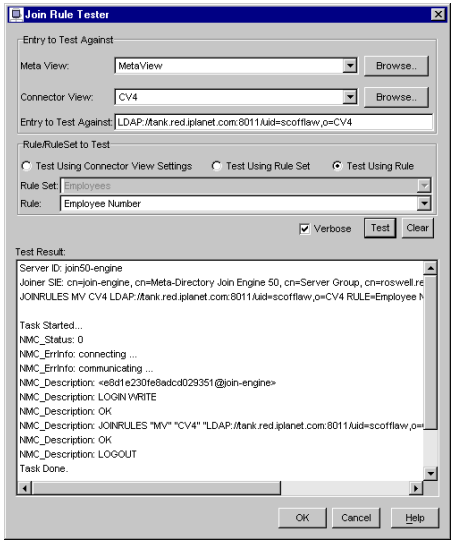
   The set disappears from the Rule Sets list box.
3. Click Save when you are finished.

To Test a Rule

Before a rule can be tested, a connector view with at least one entry must be configured and added as a participating view. Also check that the join engine has been started.

1. Click Rule Tester.

   The Join Rule Tester dialog box appears.



2. Choose the Entry To Test Against:

   - To test the rule on a meta view entry, click the `Browse` button next to the Meta View drop-down list. The Select An Entry dialog box appears from which you can select the entry within the meta view you want to test on and click `OK`. The entry's location appears in the Entry to Test Against field.

- To test the rule on a connector view entry, select the desired connector view from the Connector View list and click the adjacent Browse button. The Select An Entry dialog box appears. Select an entry within the connector view you want to test on and click OK. The entry's location appears in the Entry to Test Against field.

3. Select the Rule / Rule Set To Test:

   - **Test Using Connector View Settings** uses the join rule set employed by the connector view selected from the drop-down list. This option is available whether you are testing an entry from a connector view or from a meta view. Select the rule set name in the Rule Set field.
   - **Test Using Rule Set** uses a join rule set. Select the rule set from the Rule Set field.
   - **Test Using Rule** uses a join rule. Select the rule from the Rule field.

4. Select Verbose to receive detailed test results, including which rule or rule set (along with the rules within the rule set) were tested. Deselect Verbose to receive final results only.
5. Click Test to execute the test.

Constructed Attributes

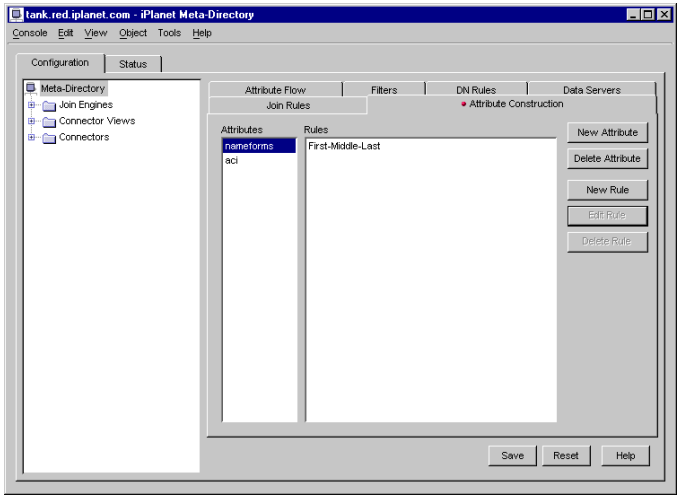
A *constructed attribute* is an attribute name and its value created from information in a source entry. Once created, the constructed attribute will show up as an attribute choice during the creation of attribute flow rules. For example, if a data source's entries have a Comments field that contains three pieces of information (department, job title, and phone extension), this information can be broken into three constructed attributes. Conversely, there may be three separate attributes that list a car type, a car color and a transmission type that can be consolidated into a single constructed attribute. The final value of the attribute can be static (a telephone number which remains the same) or dynamic (a value based on an entry's attributes or culled from sub-token assignments).

NOTE One common use of constructed attributes is defining objectclasses for data flowing from Oracle tables into the meta view.

Working With Constructed Attributes

1. From the Meta-Directory console, choose the Configuration tab.
2. Select Meta-Directory in the navigation tree.
3. Click the Attribute Construction tab.

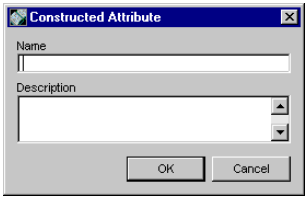
   The Attribute Construction panel appears.



To Name a New Constructed Attribute

1. Click New Attribute.

   The Constructed Attribute dialog box appears.



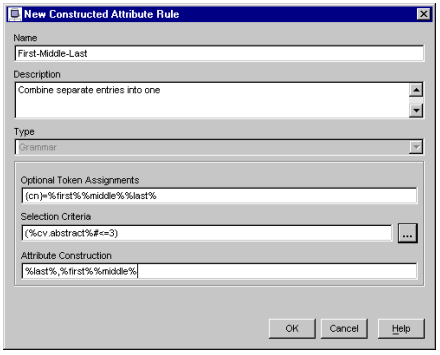
2. Provide a name and description for the new attribute.
3. Click OK.

   The new attribute appears in the Attributes list.

To Create a Rule for a New Constructed Attribute

1. Select the attribute, then click New Rule.

   The New Constructed Attribute Rule dialog box appears.



2. Provide values for the following fields:

Name	Enter a name for the constructed attribute. Restrict the name to the following characters: `A-Z a-z 0-9 _ -` Spaces are not allowed in constructed attribute names.
Description	Optional. Enter a description.
Type	The default rule type is Grammar (join engine provides condition and substitution statements that are based on attribute names and values). Additional script-based rule options may be available through iPlanet Professional Services.
Optional Token Assignments	This field is used to break up an attribute's value into separate, more specific values. These sub-tokens can then be used as attributes. For example, the attribute `{cn}=%last,first%` can be sliced into two attributes by entering `{cn}=%last%,%first%.` To compose token assignments, see "Optional Token Assignments (Format Operator)," on page 68 for instructions.

Selection Criteria	Optional. This field determines the conditions that must be met by an entry before a rule is applied. For instance, you can choose to flow only those entries that contain (`%objectclass%=person`); using this field parses the entries using that standard. To compose selection criteria, see "Selection Criteria (Requirements Operator)," on page 69 for instructions.
Attribute Construction	This required field carries the value of the constructed attribute. A hard-coded value, the entry's attribute, token assignments or a combination are acceptable. For example, `%givenname%.%sn%@netscape.com` is a combination of attributes and hard-coded values. For more information, see "Distinguished Name Construction / Join Filter / Attribute Construction (Substitution Operator)," on page 75.

3. Click OK.

   The new rule for this attribute appears in the Rules list box.

4. Repeat the steps above to add more rules.

5. Click Save when you are finished.

Example:

In order to construct an objectclass attribute for use with an Oracle database, you might name it `dbobjectclass` and add the objectclasses to be defined in the Attribute Construction field. Depending on the attributes in the database, `top;person;organizationalPerson;inetOrgPerson` is recommended.

To Edit a Rule

1. Select the rule you want to edit.

2. Click Edit Rule.

   The Edit Constructed Attribute Rule dialog box appears.

3. Alter the fields as necessary, and click OK.

   The Attribute Construction window reappears.

4. Click Save when you are finished editing rules.

To Delete a Rule

1. Select the rule to be deleted.
2. Click Delete Rule.

   The rule disappears from the Rules column.
3. Click Save when you are finished deleting rules.

To Delete an Attribute

1. Select the attribute you want to delete.
2. Click Delete Attribute.

   The attribute disappears from the Attributes list.
3. Click Save when you are finished deleting attributes.

Attribute Flow Rules

An *attribute flow rule* allows you to map attributes between a connector view and a meta view. Attribute flow tables define matching attributes; they consist of a single source attribute and one destination attribute. When a new entry appears in a connector view, attribute flow rules determine which attributes will be flowed and how to map the attribute names between the connector and meta view.

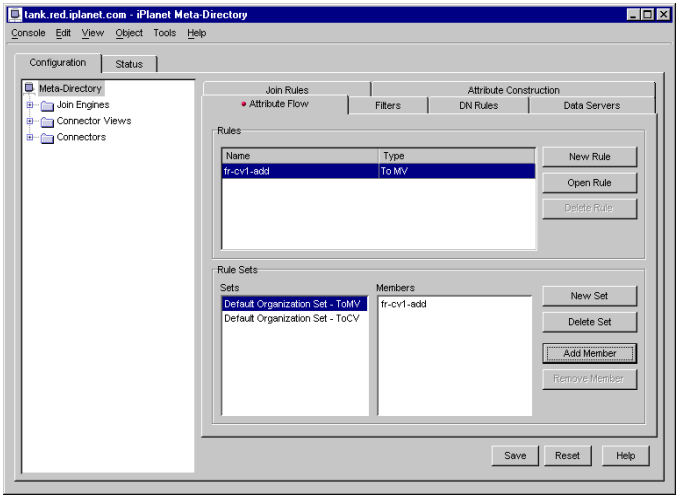
NOTE Default attribute flow rules, referred to as *Atomic*, can also be used. When the join engine applies atomic attribute flow, attributes in both views flow one-to-one while attributes in the destination entry that are not present in the source entry are deleted.

Working With Attribute Flow

1. From the Meta-Directory console, choose the Configuration tab.
2. Select Meta-Directory in the navigation tree.

3. Click the Attribute Flow tab.

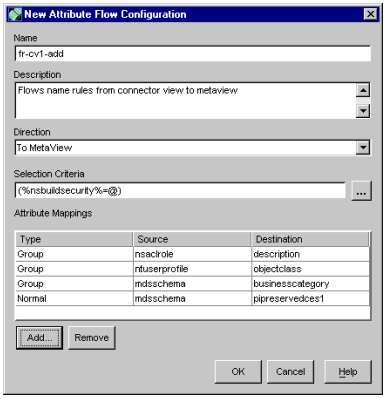
The Attribute Flow panel appears.



To Add a New Rule

1. Click New Rule.

The New Attribute Flow Configuration dialog box appears.



2. Provide values for the following fields:

Name	Enter a name for the attribute flow rule. Restrict the name to the following characters: `A-Z a-z 0-9 _ - <space>`
Description	Optional. Enter a description of the rule.
Direction	Specify whether you want the new rule to map entries that flow to the meta view or to the connector view.
Selection Criteria	Optional. This field determines the conditions that must be met by an entry before a rule is applied. For instance, you can choose to flow only those entries that contain `(%objectclass%=person)`; using this field parses the entries using that standard. To compose selection criteria, see “Selection Criteria (Requirements Operator),” on page 69 for instructions.
Attribute Mappings	See To Add Attribute Mappings for instructions. (The meta view and at least one connector view must be configured before moving on to this step. For those procedures, see Chapter 6, “Views in Meta-Directory.”)

3. Click `OK` from the New Attribute Flow Configuration dialog box.

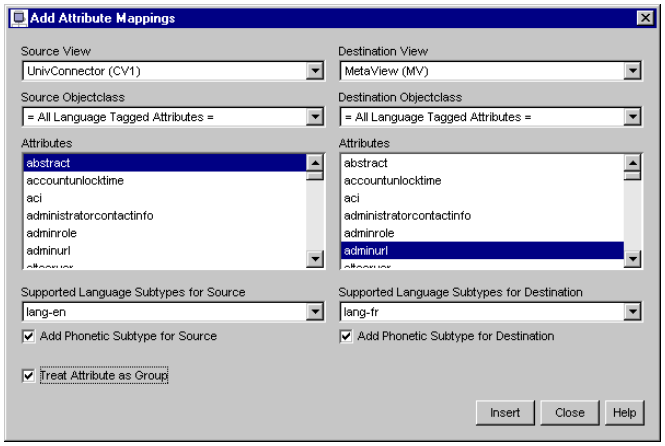
   The new rule appears in the Rules section of the Attribute Flow window.

4. Click Save when you are finished adding new rules.

To Add Attribute Mappings

1. From the New Attribute Flow Configuration dialog box, click Add to configure attribute mappings for this rule.

   The Add Attribute Mappings dialog box appears.



2. Select your choices from the following fields:

Source View and Destination View	The object class choices shown are based on the view chosen.
Source Objectclass and Destination Objectclass	Choose one of the following to define the Attribute field below: • All Attributes to list attributes in the chosen view's schema. • All Language Tagged Attributes to list attributes with an associated language tag. • Constructed Attributes to list attributes created as constructed attributes.
Supported Language Subtypes for Source and Supported Language Subtypes for Destination	To define an attribute with a language subtype, choose All Language Tagged Attributes and select the language subtype for the attribute desired. A language subtype allows searches in other languages. For more information, see "Language Subtype Tagged Attributes," on page 92.

Add Phonetic Subtype checkbox	Select to indicate that the attribute value is a phonetic representation. For example, `givenname;lang-ja;phonetic` indicates that the attribute value is the phonetic version of the entry's Japanese name.
Treat Attribute as Group	Select if the selected attribute applies to a number of entries. Your choice is reflected in the Type column of the New Attribute Flow Configuration dialog box after you click Insert.

3. Select an attribute from the source view and one from the destination view and click Insert.

   The mapping appears in the lower half of the New Attribute Flow Configuration dialog box under the Type, Source, and Destination columns.

4. Repeat Step 2 and Step 3 to add additional mappings to the configuration.

5. Click Close when you are finished.

To Edit a Rule

1. Select the rule you want to edit.

2. Click Open Rule.

   The Edit Attribute Flow Configuration dialog box appears.

3. Change information in the fields as necessary. You can add or remove attribute mappings by using the buttons at the bottom of the dialog box.

   - To add a mapping, see "To Add Attribute Mappings" on page 88.
   - To delete a mapping, select the mapping you want to delete and click Remove.

4. Click OK to return to the Attribute Flow window.

5. Click Save when you are finished editing rules.

To Delete a Rule

1. Select the rule you want to delete.
2. Click Delete Rule.

   The rule disappears from the Rules list.
3. Click Save when you are finished.

To Create a New Rule Set

1. Click New Set.

   The New Set dialog box appears.
2. Provide a name for the new set and click OK.

   The new name appears in the Sets list box.
3. Click Save when you are finished.

To Assign a Rule to the New Rule Set

1. Select the name of the new rule set from the Rule Sets list box.
2. Select a rule in the Rules list.
3. Click Add Member.

   The rule appears in the Members list box.
4. Repeat steps 2 and 3 to add additional rules to the set as desired.
5. Click Save when you are finished.

CAUTION When defining attribute flow rule sets, each rule set must contain rules defined for the same direction only. Rule members flowing to the meta view or flowing to the connector view should be set up in rule sets flowing to the meta view or flowing to the connector view, respectively. If the same rule set is used for rules flowing in both directions, an objectclass violation will be encountered and entries will not flow.

To Delete a Rule From a Rule Set

1. Select the rule member you want to delete.
2. Click Remove Member.

   The member disappears from the Members list box.
3. Click Save when you are finished.

To Delete a Rule Set

1. Select the rule set you want to delete.
2. Click Delete Set.

   The rule set disappears from the Sets list box.
3. Click Save when you are finished.

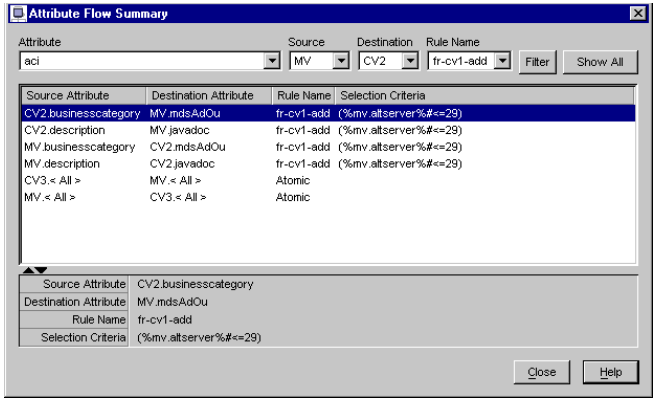
Attribute Flow Summary

After you have configured attribute flow rules and rule sets, and specified attribute flow for each connector view or meta view relationship, you can view a summary of attribute flow rules. The Attribute Flow Summary dialog box displays all mappings for a particular attribute, all mappings for a particular meta or connector view, or limits the display by using a filter.

To View an Attribute Flow Summary

1. In Meta-Directory console, right-click on the join engine.
2. Select Attribute Flow Summary from the context menu.

   The Attribute Flow Summary dialog box appears.



To Configure an Attribute Flow Summary

1. Provide values for the following fields:

Attribute	Select an attribute to limit the display to the mappings that contain the attribute. This can be a connector view attribute or a meta view attribute.
Source/Destination	Select a meta view or connector view ID to limit the display to mappings that include attributes from the corresponding connector view or meta view.
Rule Name	Select an attribute flow rule name to limit the display to mappings that use a particular attribute flow rule.

2. Click Filter to display attribute mappings based on your choices.

   Alternatively, you can display all attribute flow mappings by clicking Show All.

3. The summary is divided into four columns of information:

Source Attribute	The attribute that is native to the source chosen.The source attribute coupled with the destination attribute is the attribute flow rule.
Destination Attribute	The attribute that is *not* native to the source chosen. The source attribute flows to the destination attribute.
Rule Name	The name of the attribute flow rule that includes the attribute mapping.
Selection Criteria	The selection criteria configuration used by the attribute flow rule that contains this attribute mapping.

Language Subtype Tagged Attributes

A language subtype can be added to any attribute using the attribute flow rules. By choosing a language other than the default (English), users are offered the option of searching in that language. For example, Noriko prefers that her name be represented by Japanese characters as well as English. Selecting Japanese as a language subtype for the `givenname` attribute allows other users to search for her Japanese name.

If a language subtype is specified for an attribute, it takes the form *attribute*;lang-*subtype* where *subtype* is the two character abbreviation for the specified language. You can assign only one language subtype per attribute instance in an entry. To assign multiple language subtypes, add another attribute instance to the entry and then assign the second language subtype. For example, the following is invalid: cn;lang-ja;lang-en-GB:Smith. Instead, use:

cn;lang-ja:*ja_value* together with cn;lang-en-GB:*en-GB_value* .

The following table contains the list of supported language subtypes.

Table 5-1 Supported Language Subtypes

Language Tag	Language
af	Afrikaans
be	Byelorussian
bg	Bulgarian
ca	Catalan
cs	Czechoslovakian
da	Danish
de	German
el	Greek
en	English
es	Spanish
eu	Basque
fi	Finnish
fo	Faroese
fr	French
ga	Irish
gl	Galician
hr	Croatian
hu	Hungarian
id	Indonesian
is	Icelandic
it	Italian

Table 5-1 Supported Language Subtypes *(Continued)*

Language Tag	Language
ja	Japanese
ko	Korean
nl	Dutch
no	Norwegian
pl	Polish
pt	Portuguese
ro	Romanian
ru	Russian
sk	Slovakian
sl	Slovenian
sq	Albanian
sr	Serbian
sv	Swedish
tr	Turkish
uk	Ukraianian
zh	Chinese

Filters

By default, the join engine includes all entries in the join process. However, filters can be configured to prevent particular subtrees or entries or both from participating. These filters are based on the DNs of the entries and not the attributes. They do not support random inclusions and exclusions.

Example

To filter out all organizations except one, as well as exclude two users from the organization during processing, the filter might appear as follows:

“NoSubtreesExcept” `o=siroe,c=us`

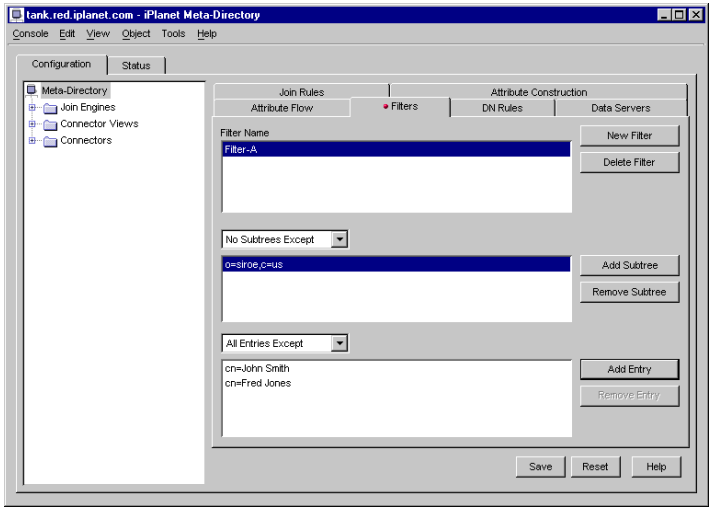
“AllEntriesExcept” `cn=John Smith cn=Fred Jones`

This means that the only subtree to be processed is `o=siroe, c=us`. Within this subtree, the connector is interested in all entries except `cn=John Smith` and `cn=Fred Jones`. All entries beneath the subtree will pass the filter except the ones with the names John Smith and Fred Jones.

Working With Filters

1. From the Meta-Directory console, choose the Configuration tab.
2. Select Meta-Directory in the navigation tree.
3. Click the Filters tab.

   The Filters panel appears.



To Create a New Filter

1. In the Filters window, click New Filter.

   The Filter Name dialog box appears.
2. Enter a name and click OK.

   The new name appears in the Filter Name list box.

To Add a Subtree Filter

1. In the Filters window, select a filter name, then click Add Subtree.

   The Subtree DN dialog box appears.

2. Enter a DN for the subtree and click OK.

   The DN appears in the list box.

3. Select All Subtrees Except or No Subtrees Except from the drop-down list.
4. Repeat steps 1-3 to add other subtree filters.
5. Click Save when you are finished.

To Add an Entry Filter to the Subtree

1. Select a filter name and then select the subtree DN to which you want to add an entry filter.
2. Click Add Entry.

   The Entry RDN dialog box appears.

3. Enter a relative distinguished name and click OK.

   The RDN appears in the list box.

4. Select either All Entries Except, or No Entries Except from the drop-down list.
5. Repeat steps 1-4 to add other entry filters.
6. Click Save when you are finished.

To Remove a Subtree Filter

1. Select the DN you want to remove.
2. Click Remove Subtree.
3. Click Save when you are finished.

NOTE This action removes all entry filters associated with this subtree.

To Remove an Entry Filter

1. Select the entry you want to remove.
2. Click Remove Entry.
3. Click Save when you are finished.

To Delete a Filter

1. Select the filter name you want to delete from the Filter Name list box.
2. Click Delete Filter.

   The filter and its associated subtrees and entries disappear from the window.
3. Click Save when you are finished.

Distinguished Name Mapping Rules

If join rules cannot identify a suitable entry in the target view to link to the source entry, the join engine will create such an entry. *Distinguished Name (DN) mapping rules* are used as guidelines to compose a DN within the target view for the replicated entry. You can create your own DN mapping rules or use the *Atomic* (default) rules provided with the join engine.

NOTE When the join engine applies an atomic DN mapping rule, the RDN of the source entry is appended to the destination view's base DN even though the partial DN (relative to the source's base DN) may be different. For instance, an RDN of `user1` in an assigned connector view of `cn=user1,ou=cv1,o=iplanet.com` would remain the same when atomic rules are applied to the meta view as `cn=user1,o=mv`.

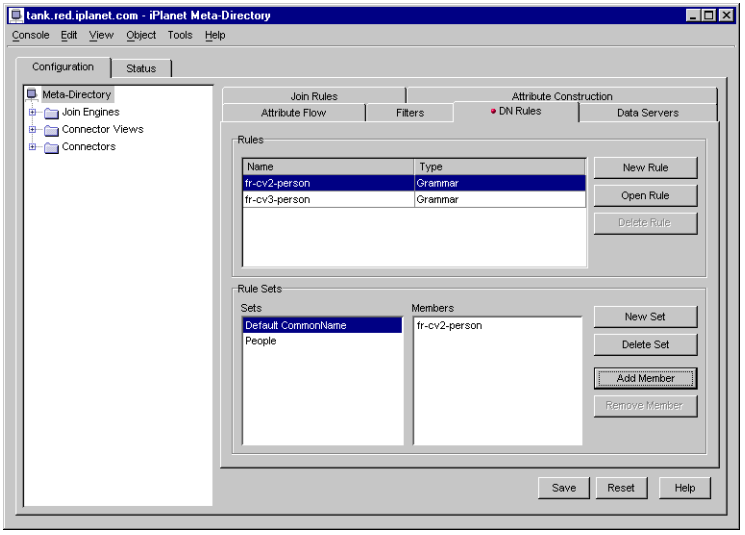
You can group multiple DN mapping rules into a DN mapping rule set to allow for ordered testing. You can also define rules differently for entries originating from different connector views.

NOTE The DN Mapping only specifies the partial DN, relative to the view's base DN.

Working With DN Mapping Rules

1. From the Meta-Directory console, choose the Configuration tab.
2. Select Meta-Directory in the navigation tree.
3. Click the DN Rules tab.

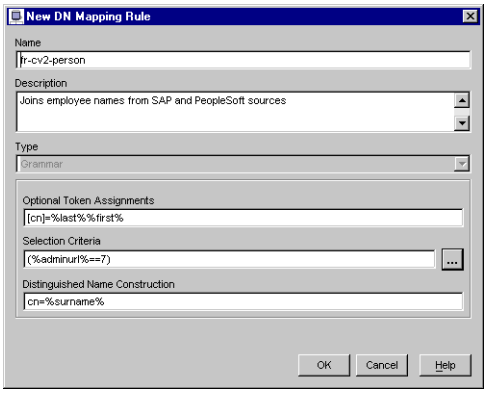
   The DN Rules panel appears.



To Add a New Rule

1. Click New Rule.

The New DN Mapping Rule dialog box appears.



2. Provide values for the following fields:

Name	Enter a name for the DN mapping rule. Restrict the name to the following characters: `A-Z a-z 0-9 _ - <space>`
Description	Enter an optional description.
Type	The default rule type is Grammar (join engine provides condition and substitution statements that are based on attribute names and values). Additional script-based rule options may be available through iPlanet Professional Services.
Optional Token Assignments	This field is used to break up an attribute's value into separate, more specific values. These sub-tokens can then be used as attributes. For example, the attribute `{cn}=%last,first%` can be sliced into two attributes by entering `{cn}=%last%,%first%`. To compose token assignments, see "Optional Token Assignments (Format Operator)," on page 68 for instructions.

Selection Criteria	Optional. This field determines the conditions that must be met by an entry before a rule is applied. For instance, you can choose to flow only those entries that contain (`%objectclass%=person`); using this field parses the entries using that standard. To compose selection criteria, see "Selection Criteria (Requirements Operator)," on page 69 for instructions.
Distinguished Name Construction	This required field carries the value of the entry's RDN. Be certain that the attributes used have corresponding attributes in the Directory Server schema. For more information, see "Distinguished Name Construction / Join Filter / Attribute Construction (Substitution Operator)," on page 75.

3. Click OK.

   The new rule for this DN mapping appears in the Rules list box.

4. Repeat steps 1-3 to add more DN mapping rules.

5. Click Save when you are finished.

Example

A connector with user information at `ou=users,o=MD` flows to a meta view with user information at `ou=users,ou=internal,o=MV`. The DN Construction might be: `uid=%uid%,ou=users,ou=internal,o=MV`

To Edit a Rule

1. Select the rule you want to edit.

2. Click Open Rule.

   The Edit DN Mapping Rule dialog box appears.

3. Alter the fields as necessary, then click OK.

   The DN Rules window reappears.

4. Click Save when you are finished.

To Delete a Rule

1. Highlight the rule you want to delete.
2. Click Delete Rule.

   The rule disappears from the Rules list.
3. Click Save when you are finished.

To Create a New Rule Set

1. Click New Set.

   The New Set dialog box appears.
2. Enter the new rule set name and click OK.

   The new name appears in the Sets list.

To Assign a Rule to a Rule Set

1. Select the rule set name from the Sets list.
2. From the Rules list, select a rule you want to add to the rule set, and click Add Member.

   The rule you add appears in the Members list.
3. Repeat steps 1-2 to add more rules to the rule set.
4. Click Save when you are finished.

To Edit a Rule Set

1. Select the set you want to edit.
2. Add or remove a member by doing one of the following:
   - To add a member, select the name of the member you want to add from the Rules menu and click Add Member. The added member appears in the Members list. Click Save when you are finished adding members.
   - To remove a member, select the member you want to remove from the Members list and click Remove Member. Repeat or click Save when you are finished.

To Delete a Rule Set

1. Select the set you want to delete.
2. Click Delete Set.

   The set disappears from the Sets list.
3. Click Save when you are finished.

Data Servers

Meta Directory has the ability to connect to a number of data servers for a number of functions. For instance, these servers can be added and used as external data sources or as space to hold connector views or meta views. Currently, Meta-Directory can connect to two types of data servers: iPlanet Directory Server and Oracle. For information on how to connect to and manage these data servers, see Chapter 17, “Managing Servers and Permissions.”

Chapter 6

Views in Meta-Directory

In joining data, Meta-Directory lets you view information in two ways, in connector views and in meta views. A connector view displays data from an external data source. A meta view displays integrated data from a number of connector views. This chapter contains the following sections:

- Connector Views
- Meta View
- Participating Views
- Refreshing the Views

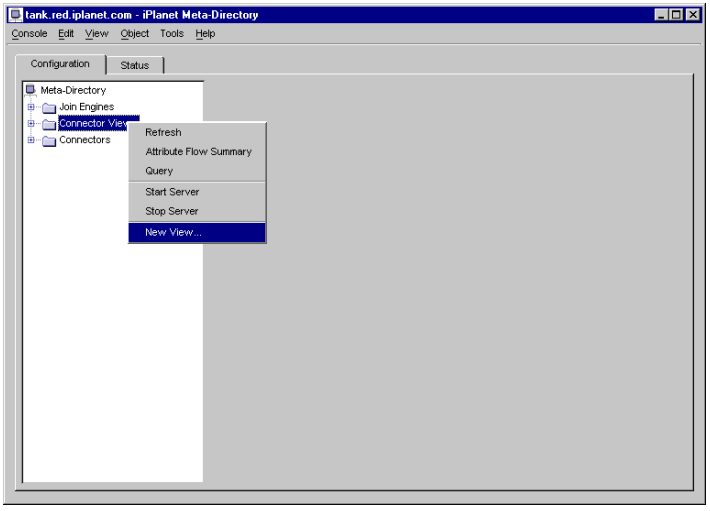
Connector Views

A *connector view* is an LDAP representation of data that resides in an external source. Connector views provide LDAP-ready information to the join engine, which uses this information to build the meta view. In order for data from the connector view to become part of the meta view, the connector view must be enabled as a *participating view*, in effect, allowing connector view data to flow. Only connector views added to the participating views list will synchronize entries to and from the meta view. (Data can flow in both directions between the meta view and the participating connector view.) The following sections explain how to add and remove connector view instances.

To Add a Connector View Instance

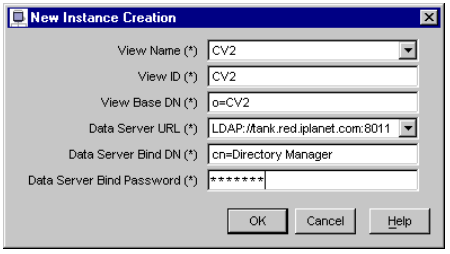
1. Select Connector Views from the Meta-Directory console navigation tree, and right-click.

   A context menu appears.



2. Select New View.

   The New Instance Creation dialog appears.

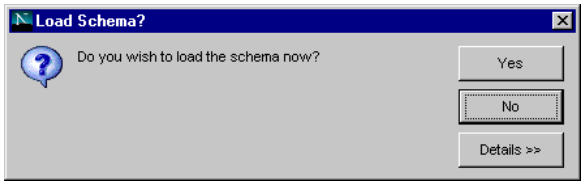


3. Provide information for the following fields:

View Name	Enter a name for the connector view. The default is CV#, where # is an integer.
View ID	Enter no more than five characters to represent a unique ID for the connector. The default is CV#, where # is an integer.
View Base DN	Enter the base DN under which the view's information will be stored. The default is `o=`CV#, where # is an integer. (If you are creating a connector view instance for the Universal connector, do not create the location under the root for all Netscape products, `o=NetscapeRoot`.)
Data Server URL	Select (or enter) a Directory Server or Oracle database URL where the data will be stored.
Data Server Bind DN	Enter a DN with which the join engine will authenticate to the Directory Server.
Data Server Bind Password	Enter the password associated with the bind DN.

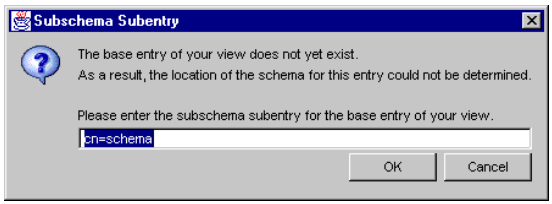
4. Click OK.

The Load Schema dialog box appears:



5. Click Yes to load the schema.

   If the base entry of your schema does not exist, the Subschema Subentry dialog box appears:



   This dialog box appears because Meta-Directory has proprietary attributes that are not contained within the Directory Server.

6. Accept the default location of `cn=schema`, or provide another subentry, to store these attributes and click `OK`.

7. The system will create the new instance, which may require several minutes.

   The message “Instance Creation Succeeded” appears after the instance has been created.

NOTE While loading the Meta-Directory schema to a Directory Server instance which does not contain the join engine and meta view configuration information, “cannot delete” error messages might result from the `ldapmodify` utility trying to delete an entry that does not exist. These messages are not serious. For more information, see Appendix C, “Troubleshooting Meta-Directory.”

To Remove a Connector View Instance

1. In the Meta-Directory console, highlight the instance in the navigation tree that you want to remove, and right-click.

   A context menu appears.

2. Choose Delete View.

   The instance is deleted.

Providing a Connector View Description

Descriptive information can be added to an existing or new connector view instance using the following procedure.

1. In the navigation tree of Meta-Directory console, select the connector view for which you want to add a description.

   The General tab displays the values for the View ID, Name, and DN fields that were provided in the New Instance Creation dialog box. These values cannot be changed.

2. Enter descriptive text in the Description field.

3. Click Save.

An Enabled Connector View

The connector view has the same monitoring options as the join engine. These options can be accessed through the Status tab of Meta-Directory console. See “Connector Operations,” on page 254 of Chapter 14, “Monitoring Meta-Directory Components” for more information.

By clicking the Status tab and then selecting the Operations tab, you can check that your connector view is enabled. By default, the connector view is always enabled unless the data is being refreshed, in which case Enabled, in the View window, is replaced by either Refreshing CV.

Meta View

The *meta view* is a unified view of entries from one or more connector views; it represents the result of the join process. After the join engine processes the information received from a connector view, it transfers the information to the meta view. (Like the connector view, the meta view is a sub-tree on a Directory Server.) From the meta view, you can view linked entries as well as modify them and send the changes back to the original entries via the connector views. When you create an instance of the join engine, the meta view is created at the same time. You can see the meta view icon in the Meta-Directory console navigation tree under the join engine instance you created. It is an empty meta view until it is joined with at least one connector view. Meta-Directory supports only one meta view per join engine instance.

For information on creating instances of the join engine and creating a meta view, see “Creating the Join Engine Instance,” on page 64 of Chapter 5, “Configuring The Join Engine.”

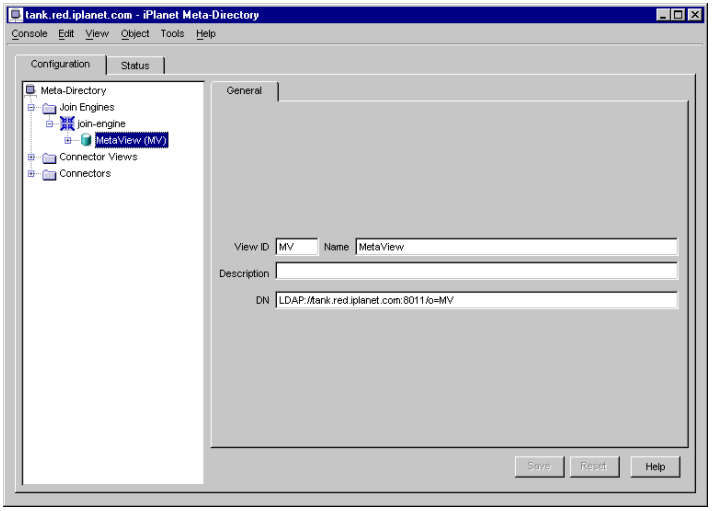
Providing a Meta View Description

Descriptive information can be added to a meta view using the following procedure.

To Provide Descriptive Information for a Meta View

1. Select Meta View from the Meta-Directory console navigation tree.

   The General tab displays the values for the View ID, Name, and DN fields that were provided in the New Instance Creation dialog. These values cannot be changed.



2. Enter descriptive text in the Description field.
3. Click Save.

Participating Views

In order for a connector view to be accessed by the meta view it must be added as a participating view and configured to participate in the join process. Once a connector view becomes a participating view and is enabled, data can flow bi-directionally between that view and the meta view.

Once a participating view is added, it is configured by applying join process rules to it. Each participating connector view is configured separately. Enabling the participating view is the final step in allowing the connector view to participate in the join process.

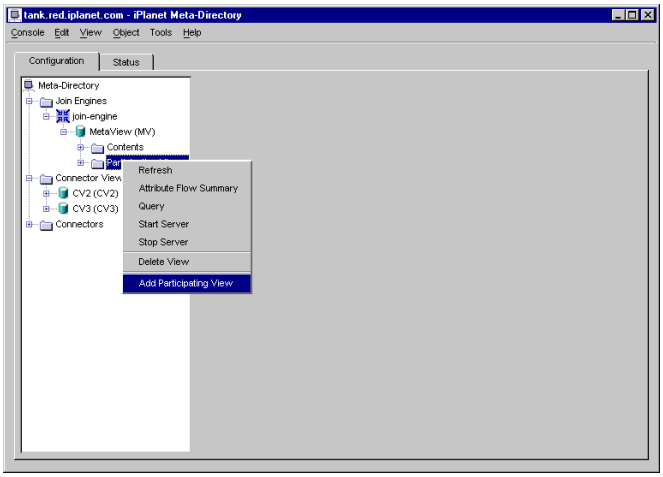
Atomic Rules

When choosing join process rules for a participating view, Attribute Flow rules and DN Mapping rules contain a selection called *Atomic*. Atomic refers to Meta-Directory default rules that flow, map, and join LDAP attributes that are clearly the same. For example, when the join engine applies an atomic attribute flow rule, attributes in the source entry that have no match in the destination entry will not be flowed. When the join engine applies an atomic DN mapping rule, the RDN of the source entry is added to the base DN of the destination view to form a full DN. For instance, an RDN of `user1` in a connector view located in `cn=user1,ou=cv1,o=iplanet.com` would remain the same when applied atomically to the meta view as `cn=user1,o=mv`.

To Add a Participating View

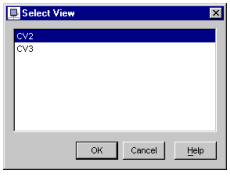
1. In the navigation tree of the Meta-Directory console, right-click the Participating Views object.

   A context menu appears.



2. Choose Add Participating View.

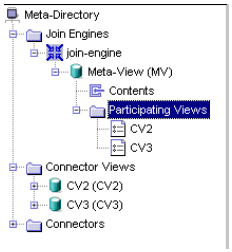
   The Select View dialog box appears.



3. Select the connector view or views you want to participate in a synchronization to the meta view.

4. Click OK.

   Selected views are added to the navigation tree as shown below.



To Remove a Participating View

1. Highlight the view you want to remove, and right-click.

   A context menu appears.

2. Select Delete View.

   The view is deleted.

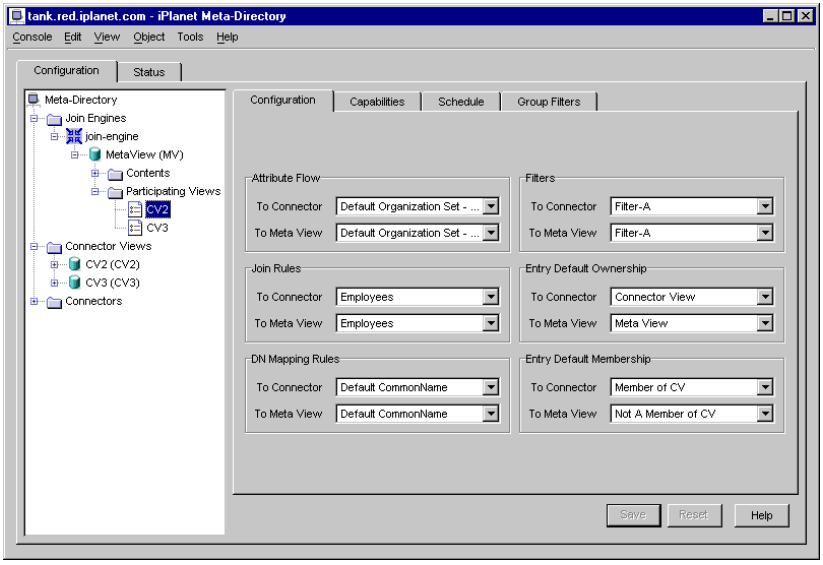
Configuring a Participating View

Before enabling a participating view, you need to configure it so that data flow between the connector view and the meta view can be managed. Join process rules are applied to the participating view which the join engine will then apply to the connector view entries. In addition, you can specify capability settings, refresh schedules and group filters for each participating view.

To Configure a Participating View

1. Select the participating connector view you want to configure and choose the Configuration tab from the right panel.

   The Configuration tab for a participating connector view contains a set of drop-down lists that allow you to apply the rules for the join process.



2. Select the appropriate combinations of rules from the drop-down list boxes, and click Save.

   a. Select rule sets for Attribute Flow, Join Rules, DN Mapping Rules, and Filters. The choices are derived from the rules that you set up in “Creating the Join Engine Instance,” on page 64 of Chapter 5, “Configuring The Join Engine.”

   b. For Entry Default Ownership, select Connector or Meta View for each of these two drop-down lists:

   - The selection made in To Connector specifies the view that owns the entries replicated from the meta view to the connector view.

- The selection made in To Meta View specifies the view that owns the entries replicated from the connector view to the meta view.

When an entry is owned by either the meta view or the connector view, it can only be deleted through that view. By default, an entry is owned by the view from which it originates; the default ownership can be changed with this option.

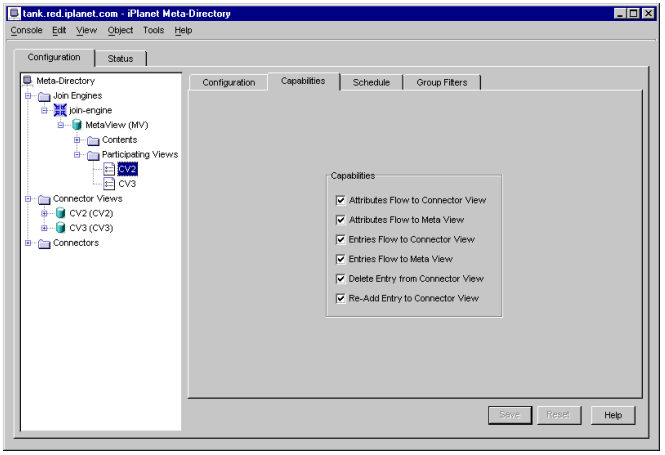
NOTE Ownership here is not the same as granularity and ownership discussed in Chapter 7, "Connectors and Connector Rules." The values discussed here refer to ownership of entries shared between the connector view and the meta view. Chapter 7 refers to ownership of entries shared between the data source and a connector view.

c. For Entry Default Membership, select Member of CV or Not A Member of CV for each of these two drop-down lists:

- The selection made in To Connector specifies whether new entries will or will not be members of the connector view as the data flows from the meta view to the connector view.
- The selection made in To Meta View specifies whether new entries will or will not be members of the connector view as the data flows from the connector view to the meta view.

Membership identifies an entry within a connector view that is native to the data source represented by the connector view. Rules can then be configured and applied based on the attributes that are already present in the data source.

3. Select the Capabilities tab.

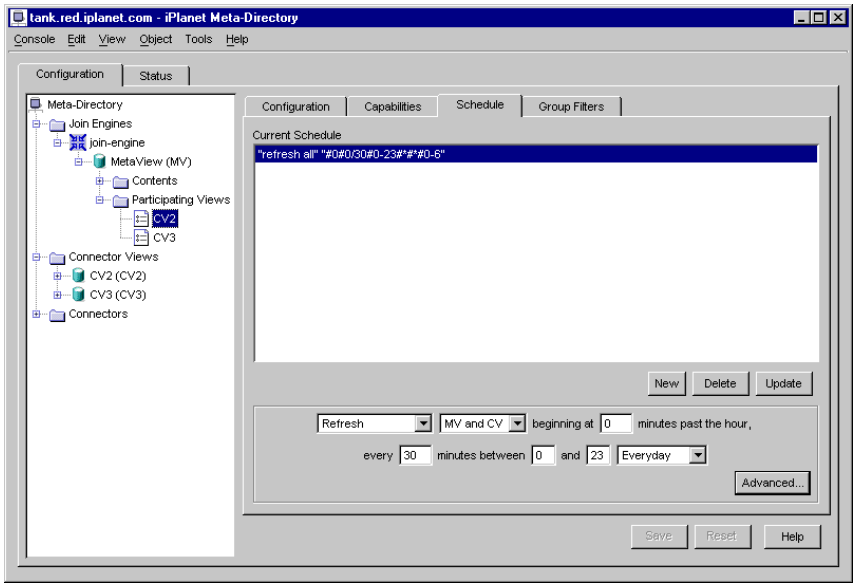


The Capabilities tab contains a set of options that control the behavior of data flow back and forth between the meta view and connector view. Select the Capabilities you want:

Attributes Flow to Connector View	Additions and changes to attributes in a meta view entry are sent to the connector view if there are suitable attribute flow rules.
Attributes Flow to Meta View	Additions and changes to attributes in a connector view entry are sent to the meta view if there are suitable attribute flow rules.
Entries Flow to Connector View	Entries added to a meta view are sent to the connector view if there are suitable DN mapping and attribute flow rules.
Entries Flow to Meta View	Entries added to a connector view are sent to the meta view if there are suitable DN mapping and attribute flow rules.
Delete Entry from Connector View	Deletes the entry in connector view if either: • The corresponding meta view (or different connector view) entry has been deleted, or • The connector view you are configuring does not own the entity.

Re-Add Entry to Connector View	Re-adds a deleted connector view entry if either: • An agent other than the join engine has deleted the connector view entry, or • The connector view you are configuring does not own the entity.

4. Select the Schedule tab.



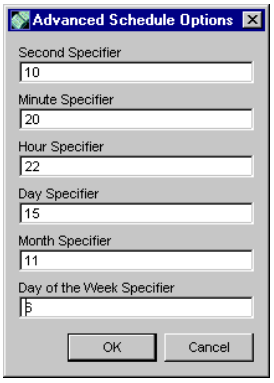
In the Schedule tab, you can configure a refresh schedule for the participating connector view. (If no schedule is configured, the view will only be refreshed manually.)

a. Click New to add a new schedule entry.

The scheduler can operate as many times as once every second; therefore, the finest granularity occurs every second.

b. Change the default values in the list boxes and field entries at the bottom of the window to schedule the desired task.

Alternatively, you can provide settings in a tabular format by clicking Advanced . . . The Advanced Schedule Options dialog box appears.



Numerals can be used in the Advanced Schedule Options fields:

Field Name	Value
Second Specifier	Enter a value from 0 to 59.
Minute Specifier	Enter a value from 0 to 59.
Hour Specifier	Enter a value from 0 to 23.
Day Specifier	Enter a value from 1 to 31.
Month Specifier	Enter a value from 1 to 12.
Day of the Week Specifier	Enter a value from 0 to 6, where 0 is Sunday and 6 is Saturday.

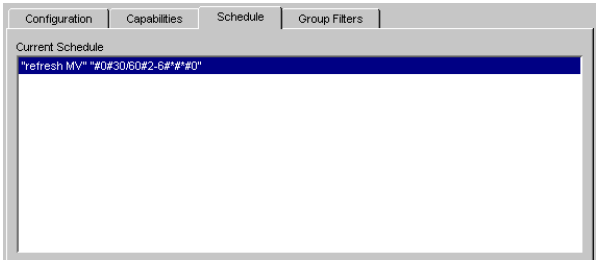
When configuring a schedule with numerals, both single values and ranges (such as 1,2,5-7,10/5) can be used.

Expressions can also be used in the Advanced Schedule Options fields:

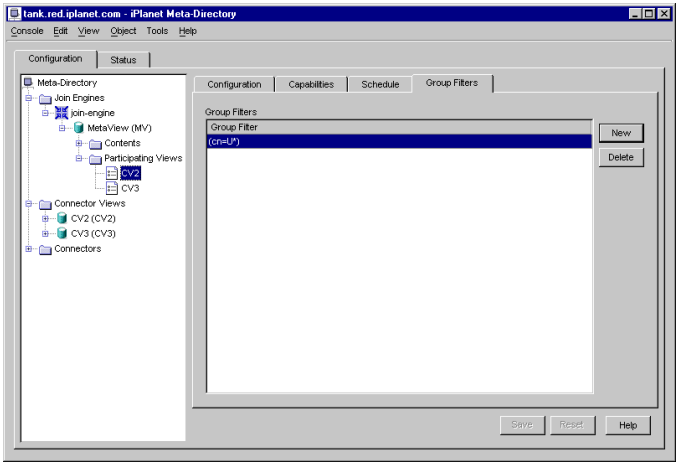
Expression	Definition
*	Matches any value.
*/step	Matches any value in increments. For example, */2 matches 0,2,4,6... up to the maximum allowed value for values that start with zero, or it matches 1,3,5,7... up to the maximum value allowed.
x-y	Specifies a range where: • Both x and y are greater than or equal to the minimum allowed value. • y is less than or equal to the maximum allowed value. • x is less than y. The expression matches any value in the range.
x-y/step	Specifies a range as above, but with a step value that is not necessarily 1.
x	Specifies a single number within the allowed range.
x/step	Matches any value starting at x and then at x + step, x + 2*step, and so forth.
x,y,z...	Specifies a comma-separated list of values.
x-y,z-q	Specifies a comma-separated list of ranges.

c. Click Update.

Your specifications appear in the Current Schedule. For instance, if you wanted to refresh only the meta view beginning at 30 minutes past the hour, every 60 minutes between 2 and 6 on Sunday, the Current Schedule would appear as shown:



5. Select the Group Filters tab.



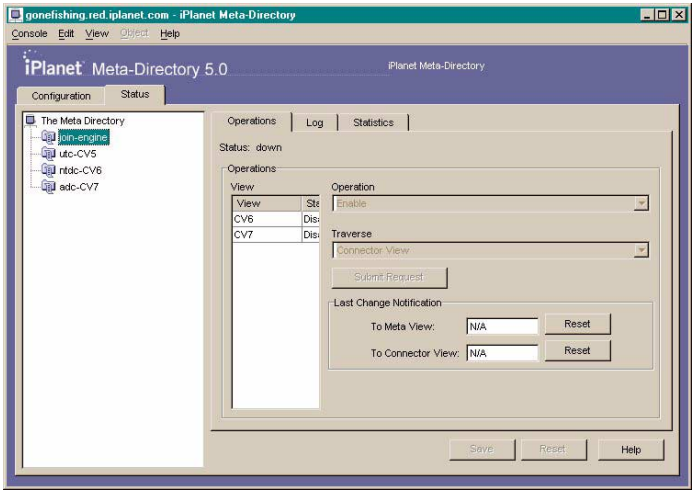
The Group Filters tab enables you to create one or more filters for LDAP data sources. You can use the group filters to refresh entries of a connector view only. The format of the filter is (*attribute=value*). Note that the parentheses are part of the syntax.

When you refresh groups, the join engine refreshes only the entries that match the group filter or filters you have specified. For information about refreshing groups, see “Join Engine Operations,” on page 251.

6. Click Save when you finished configuring the participating connector view.

To Enable a Participating View

Enabling the participating view is what allows data to flow. You enable the participating view through the Operations tab when the join engine is selected. Before enabling the participating view, make sure it has been added to participating views and has been configured. Then enable it as described next:



1. Click the Status tab from above the navigation tree window in Meta-Directory console.
2. Select the join engine from the navigation tree and click the Operations tab.
3. Select a participating view listed in the View list box that is disabled.

   The View list box has two columns: View and Status. (The size of the View column can be reduced by dragging the column divider to the left; this should make the Status column visible. Both columns can be increased in size by enlarging the console window.) All participating views that have been configured are listed in these columns along with their status: Enabled or Disabled.
4. Choose Enable from the Operation drop-down list.
5. Click Start.

   The status of the view changes from Disabled to Enabled allowing data to flow to the meta view.

Checking Entry Links

There are several reasons why an entry in a connector view might not link up to an entry in the meta view. One reason is that the join engine found more than one entry to link to. Another possible reason is that the external data contains errors. Because of these possibilities, you should check, as a standard procedure, for errors and omissions by doing one or both of the following:

- Review the Directory Server error log for reports of failures. This can be done by using a Perl script or using the command `grep -i fail *.log` in the log directory.
- As discussed in Chapter 15, "Administration Tools", use the Query Tool to check for entries which were not linked.

If you find errors, you can use join commands in the Fix-It Tool to fix the problems as described in Chapter 15, "Administration Tools."

Refreshing the Views

To incorporate new or modified data into the meta view or connector view, you use the join engine's Refresh option in the Operations tab within the Status tab of Meta-Directory console. When views are refreshed, the join rules are re-applied (or traversed) to each entry in the targeted view and the data is reconstructed. The other rules are then applied accordingly:

- If the entry is found, the attribute flow rule is applied.
- If the entry is not found, the DN mapping rule is applied.
- If the entry exists, the attribute flow rule is applied.
- If the entry doesn't exist, an entry is created.

NOTE If you change any of your rules, you need to refresh the join engine to get the data to flow according tho the new rules.

To Refresh the Connector View or Meta View

1. Select the join engine in the Meta-Directory console's navigation tree.
2. Select the Status tab; then the Operations tab.

   All participating views are listed in the View field.

3. Select the participating view whose data needs to be refreshed.
4. Choose Refresh, Refresh Unlinked or Refresh Groups from the Operation drop-down menu.
   - If you choose Refresh, the entire view will be scanned for new entries and changes to existing entries.
   - If you choose Refresh Unlinked, the view will be scanned for only entries that are not currently linked.
   - If you choose Refresh Groups, the entire view will be scanned after the application of the Group Filter. (The group filter is an LDAP filter used to select certain entries prior to refresh. Information on configuring group filters can be found on page 118 of this chapter.)
5. Choose Meta View or Connector View from the Traverse menu.

   Choosing Meta View will re-apply join rules to all entries in the meta view and, similarly, choosing Connector View will re-apply join rules to all entries in the connector view.
6. Press Submit Request to begin the process.

NOTE The Refresh command in the View menu at the top of the console window does not refresh Meta-Directory the join engine or connector. It, and the Refresh command seen when a component name in the navigation tree is right-clicked, are commands that refresh the data displayed in the console navigation tree only.

Part 3

Meta-Directory Connectors

Connectors and Connector Rules

In order to transfer data to and from an external database, Meta-Directory uses a *connector*. The connector takes the external data and creates a sub-tree on a Directory Server which displays the data in LDAP. This sub-tree is called a *connector view*.

There are two types of connectors: direct and indirect. The connector view for an LDAP directory or a SQL database uses a *direct connector*; in other words, it communicates directly with the join engine. The connector view for other sources of data uses an *indirect connector*, which translates data into LDAP so that the join engine can work with it. Indirect connectors use indirect connector rules, in much the same way as the join engine, to manage the transfer of entries between an external data source and the connector view. This chapter contains the following sections:

- Direct Connectors
- Indirect Connectors
- Indirect Connector Rules
- Attribute Flow Rules
- Default Attribute Value Rules
- Filter Rules

Direct Connectors

A directory accessible by LDAP (such as the directory provided by iPlanet Directory Server) and a database accessible by SQL (such as an Oracle database) are considered to have direct connectors. Because the join engine understands LDAP, it can directly read or write any entry stored on an LDAP-based directory.

On the other hand, in order to read and write to an entry stored within a SQL database, the join engine uses the *Database connector* to provide direct, two-way SQL access. (Because the Database Connector is a join engine plug-in as opposed to software outside the join engine, it is considered a direct connector.)

Indirect Connectors

Indirect connectors transport entries stored in external data sources that use protocol not directly accessible by the join engine. Meta-Directory supports the following indirect connectors:

The Universal Connector and Universal Text Parser

The Universal connector (also known as the Universal Text Connector or UTC) is an indirect connector that enables the transfer of data between data sources and a connector view. The Universal Text Parser (UTP) is a set of text file parsers and generators that are used with the UTC to make certain text files [currently Comma-Separated Values (CSV) files, LDAP Data Interchange Format (LDIF) files and Name-Value Pair (NVP) files] compatible with the connector view.

The Windows NT Domain Connector

The Windows NT Domain connector is a Universal connector with NT-specific Perl scripts and binaries that provides two-way synchronization of user and group data between a Windows NT SAM database and its connector view.

Active Directory Connector

The Active Directory connector is a Universal connector with Active Directory-specific Perl scripts and binaries that provides two-way synchronization of user and group data between an Active Directory database and its connector view.

Indirect Connector Rules

When an indirect connector is synchronizing entries from the external data source to the connector view, it directs the process and transforms the data using rules similar to those used during the join process. The *indirect connector rules* include Attribute Flow Rules, Default Attribute Value Rules and Filter Rules. (The Attribute Flow, Default Attribute and Filter rules used by the connectors are different from, and should not be confused with, the Attribute Flow, Default Attribute and Filter rules used by the join engine.) Indirect connector rules are defined at the connector node and applied to the connector instance specifically.

NOTE Indirect connector rules can be applied at anytime but the connector instance should be restarted after changes in the configuration are made in order to re-flow data using the new rules.

Attribute Flow Rules

Attribute flow rules are established to specify which external data source attributes are mapped to which connector view attributes and vice versa. (The assignment of an attribute in one source to a particular attribute in another source is called *mapping*.) When you establish attribute flow rules, you also specify which source owns the entry which, by default, is the external data source.

Default Attribute Value Rules

If no values exist for a particular attribute in an entry, either because the attribute isn't part of the entry or the attribute exists with no value, the connector applies pre-configured attribute rules for creating appropriate default values. You can change these default attribute rules as needed.

Filter Rules

An indirect connector uses filtering rules to selectively exclude entries from the synchronization process.

Attribute Flow Rules

Attribute flow rules specify which attributes in the external data source will be mapped to which LDAP attributes in the connector view. In the definition and application of these rules there are two concepts that, although not specifically referred to in the GUI, are important to remember. G*ranularity* refers to the complexity of the application of the rules, i.e.: whether the entry flows as a whole piece or whether the entry is divided into its base attributes which then flow separately. *Ownership* refers to where the entry originates (in the external data source or in the connector view), i.e.: whichever source the entry originates from is considered the owner of the entry.

Granularity and Ownership

If you don't configure your own indirect connector rules, the indirect connector uses default attribute flow rules and the process is considered to have *entry-level granularity*. Entry-level granularity is characterized by all of the following:

- Entries can be added in, and therefore flow from, either the data source or the meta view and the entry's ownership is based on this
- Only the owner of an entry can modify or delete that entry
- Entries flow back and forth as complete entries with no specific attribute mapping or filtering allowed.

If, on the other hand, an attribute flow rule is written and applied, the flow is considered to have *attribute-level granularity*. Attribute-level granularity is characterized as follows:

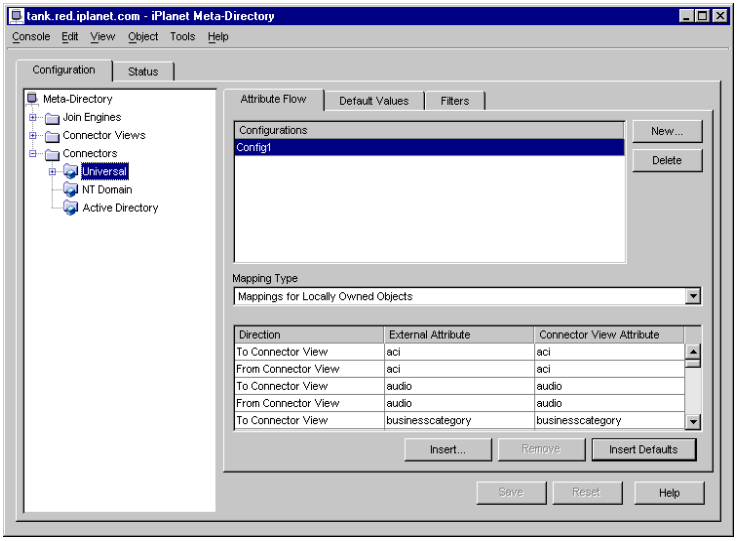
- Entries can be added in, and therefore flow from, either the data source or the meta view and the entry's ownership is based on this
- Only the owner of an entry can delete that entry
- Because specific attributes flow independently of complete entries, modifications can be made from either the data source or the meta view

These concepts explain certain flow behaviors and should be kept in mind when configuring and applying attribute flow rules for the indirect connectors.

To Configure an Attribute Flow Rule

To achieve attribute-level granularity, an attribute flow rule is written and applied, as described in the following procedure.

1. Select the connector you want to configure from the Meta-Directory console navigation tree and click The Attribute Flow tab.



2. Click New.

   The New Flow Configuration Name dialog box appears. Reset can be clicked at any time to delete all new configurations and return to the last saved state.

3. Type a name for the new attribute flow configuration and click OK.

   The name appears in the Configurations list box.

4. In the Mapping Type drop-down list, select Mappings for Connector View Owned Objects, or Mappings for Locally Owned Objects.

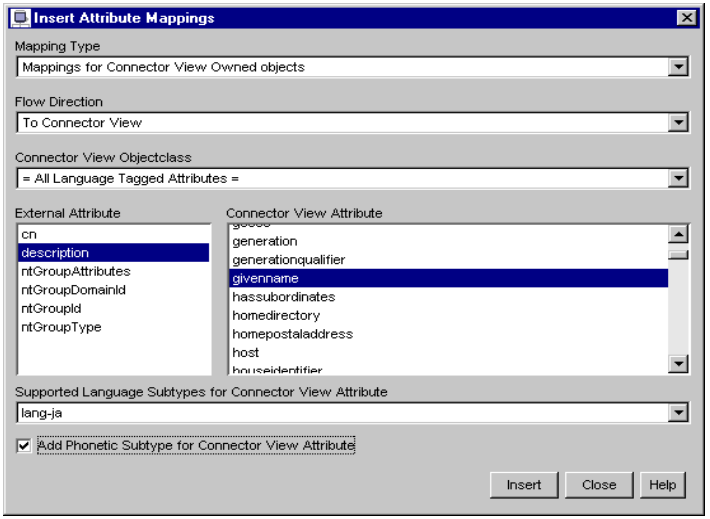
   Mappings for Connector View Owned Objects is selected for entries created within the connector view and Mappings for Locally Owned Objects is selected for entries created within the external data source.

NOTE When creating attribute flow rules, all attributes must be mapped in both directions: "From Connector View" and "To Connector View". Mappings are configured this way for both locally owned objects and connector view owned objects in order to propagate all changes.

5. Click Insert.

The Insert Attribute Mappings dialog box appears. This displays a list of all attributes configured as external attributes for the specific connector. (Adding external attributes is described in “To add external attributes for connectors” on page 148.)

Alternately, you can click Insert Defaults and Meta-Directory populates the list box at the bottom of the window with default mappings, in which the external data source attributes match the connector view attributes. These default mappings are the same as those chosen at the connector node in the General configuration window.



a. The mapping type, selected in Step 4, can be changed from within this dialog box.

b. Specify the flow direction, either mappings of attributes from external data source to the connector view or from the connector view to the external data source.

c. Specify either All Attributes or All Language Tagged Attributes from the Connector View Objectclass drop-down list.

If you specify All Language Tagged Attributes as the connector view objectclass, choose a supported language subtype. Check Add Phonetic Type box to indicate if the attribute value is a phonetic representation. For more information on these fields, see “To Compose Language Tagged Attribute Conditions,” on page 72 of Chapter 7, “Connectors and Connector Rules.”

d. Select an external attribute and the connector view attribute you wish to map it to.

If you select an external attribute for which there is a matching connector view attribute, the connector view attribute is automatically selected. However, any connector view attribute can be selected for any given external attribute. You can also use a keyword search by typing the first letter of the external attribute or connector view attribute you want to find. For instance, if you wanted to find `uid`, you would only have to type `u`.

e. Click Insert.

The mapping for your configuration appears at the bottom of the Attribute Flow window.

f. Select additional pairs, clicking Insert after each pair is selected. Click Close when finished.

6. Click Save in the Attribute Flow tab to save the attribute flow rules.

To Change a Configuration

1. In the Attribute Flow tab, select the configuration you want to change.

2. Either add or remove a mapping.

You can add or remove mappings, but you cannot edit them.

- To add a mapping, see Step 5 above.
- To remove a mapping, select the mapping you want to remove and click Remove.

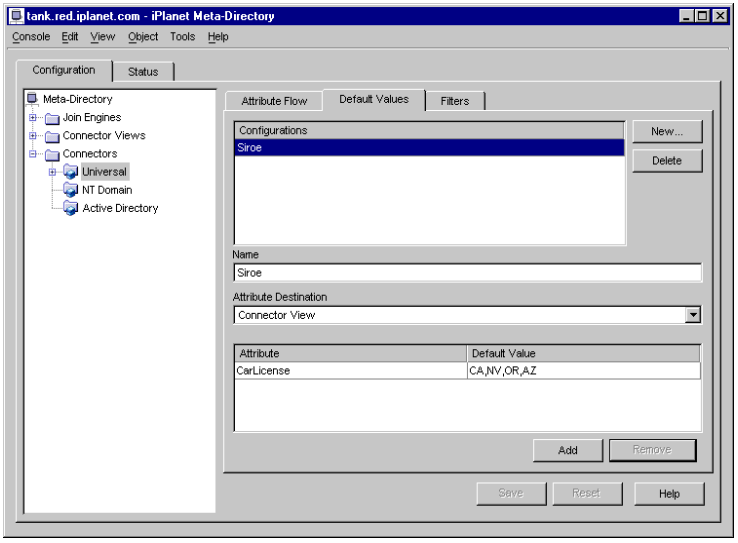
3. Click Save when you are finished changing the configuration.

Default Attribute Value Rules

In the absence of any attribute values, a connector applies default attribute value rules to ensure that specified attributes contain a value. If an attribute does not exist in the external data source or connector view or if the attribute does exist but has a NULL value, a default attribute value rule is called upon to allow the transfer of data. Default attribute value rules don't affect the connectors' behavior.

To Configure Default Attribute Values

1. In the navigation tree of the Configuration tab, select the indirect connector whose default attribute values you want to specify.
2. Select the Default Values tab.



3. Click New.
4. In the Name field, specify a name for the default attribute configuration you are creating.

   The name appears in the Configurations list box.

5. Select either Connector View or External Directory from the Attribute Destination drop-down list.
6. Click Add.

   Blank fields appear below the Attribute and Default Value fields.
7. Click within the blank Attribute field.

   A drop-down list appears.
8. Either select an attribute from the list or type in an attribute.

   To create attributes for this list, follow the procedure headed “To add external attributes for connectors” on page 148.
9. Double-click within the blank Default Value field and type in a value.
10. Repeat steps 4 - 9 to set up additional default attributes for the selected configuration.
11. Click Save when you are finished.

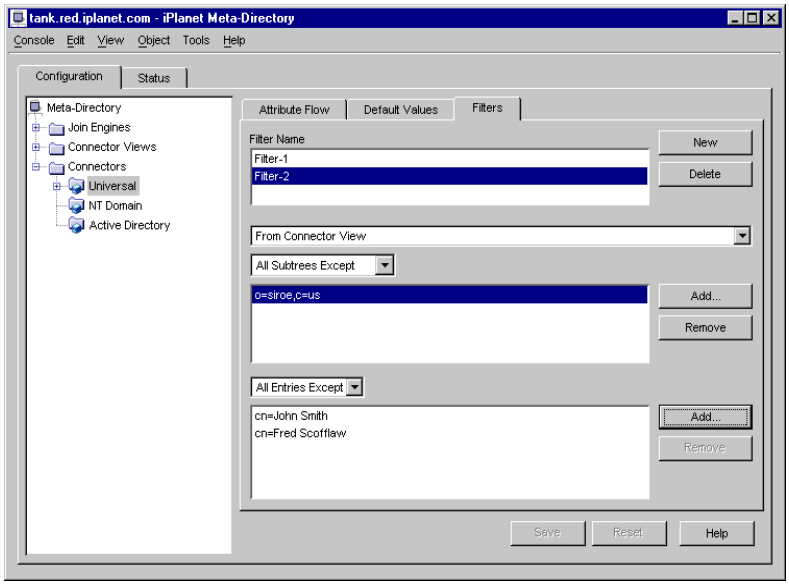
Filter Rules

The indirect connector uses filter rules to exclude source data from the synchronization process. Filters can be configured to exclude entire sub-trees while individual entries from the sub-trees can be re-included using entry filters.

NOTE DNs used in Filter Rules should be the DN from the connector view’s entry regardless of the ownership of the entry.

To Add a New Filter

1. In the navigation tree of the Configuration tab, select the indirect connector and click the Filters tab.



2. Click New.

   The Filter Name dialog box appears.

3. Enter a name and click OK.

   The new name appears in the Filter Name list box.

4. Select either To Connector View or From Connector View.

5. Provide a list of subtrees to exclude or include by selecting All Subtrees Except or No Subtrees Except, then click Add.

   The Sub-tree DN dialog box appears.

6. Specify the full DN of a connector view sub-tree to exclude or include, then click OK.

   The subtree appears in the list box. Repeat steps 5 - 6 for additional sub-trees.

7. Provide a list of entries to exclude or include by selecting a sub-tree, then selecting All Entries Except or No Entries Except. Click Add.

   The Entry RDN dialog box appears.

   - To filter entries back in from the excluded sub-trees using All Entries Except:

     I. Specify a connector view RDN entry to include, then click OK.

        The entry appears in the list box. The entry must be a child entry of the sub-tree specified.

     II. Repeat this procedure to include additional entries.

   - To specify singular excluded entries using No Entries Except:

     I. Specify a connector view RDN entry to exclude, then click OK.

        The entry appears in the list box. Again, the entry must be a child entry of the sub-tree specified.

     II. Repeat this procedure to exclude additional entries.

   - To filter entries back out from included sub-trees using All Entries Except:

     I. Specify a connector view RDN entry to exclude, then click OK.

        The entry appears in the list box. Again, the entry must be a child entry of the sub-tree specified.

     II. Repeat this procedure to exclude additional entries.

   - To specify singular included entries using No Entries Except:

     I. Specify a connector view RDN entry to include, then click OK.

        The entry appears in the list box. Again, the entry must be a child entry of the sub-tree specified.

     II. Repeat this procedure to exclude additional entries.

   NOTE With this filter, entries in all subtrees that are not specifically included are excluded, no matter how the associated entry-level filters are set.

8. Click Save when finished.

To Remove an RDN for an Entry

1. Select the RDN you want to remove.
2. Click Remove next to the Entry RDN list box.

To Remove a Subtree DN

1. Select the DN you want to remove.
2. Click Remove next to the Subtree DN list box.

NOTE This action also removes all RDNs associated with this DN.

To Delete a Filter

1. Select the filter you want to delete from the Filter Name list box.
2. Click Delete.

   The filter and its associated DNs and RDNs disappear from the window.

Chapter 8

Configuring The Universal Connector

The Universal connector is included with Meta-Directory to allow administrators to configure connectors for external data sources not directly supported by Meta-Directory. The following sections are included in this chapter:

- Function of the Universal Connector
- Configuring a Universal Connector Instance
- Restarting a Connector Instance
- Implementing the Configuration

Function of the Universal Connector

Meta-Directory includes a generic connector named the Universal Connector (also known as the Universal Text Connector or UTC). When configured, this connector enables data transfer to and from its connector view. It is termed an indirect connector because it creates a view of the external data source in LDAP (connector view) that enables the transfer of data to and from the proprietary data sources.

iPlanet provides the Universal Text Parser (UTP), a suite of scripts, to help the Meta-Directory administrator configure connectors for manipulating data from external data sources whose data can be exported or dumped into a text file. This enables the administrator to synchronize data to a meta view using a data source not specifically configured for Meta-Directory. iPlanet supports data or text files in the following text file formats:

- CSV (comma-separated values)
- NVP (name-value pairs)
- LDIF

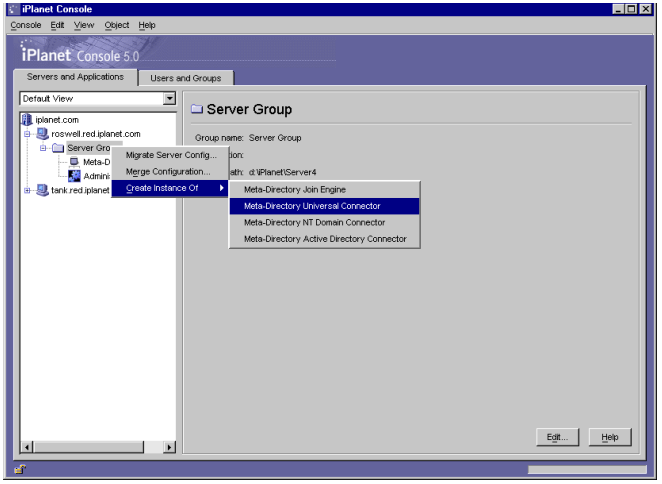
The NT Domain and Active Directory connectors are special cases of the UTC provided by iPlanet for a specific external data source, NT Domain and Active Directory respectively. When you create an instance of either of these connectors, the UTC base is automatically installed. The interface for these connectors is very similar to that of the UTC.

Besides their dependence on the Universal Connector, the UTP, NT Domain Connector, and Active Directory Connector each require their own special configuration setup. To configure the UTP, see Chapter 9, “Configuring the Universal Text Parser.” To configure the NT Domain Connector, see Chapter 10, “Configuring the NT Domain Connector.” To configure the Active Directory Connector, see Chapter 11, “Configuring the Active Directory Connector.”

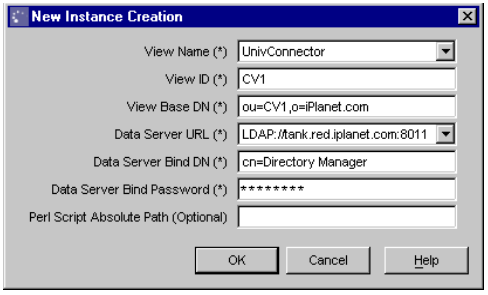
Before configuring one or more of these connectors, it is recommended that you first configure the Universal Connector, as documented in the following sections.

Creating a Universal Connector Instance

1. From the iPlanet Console window, right-click Server Group. A context menu appears.



2. Select Create Instance Of, and select the desired connector. The New Instance Creation dialog box appears.



Provide input for the following data fields:

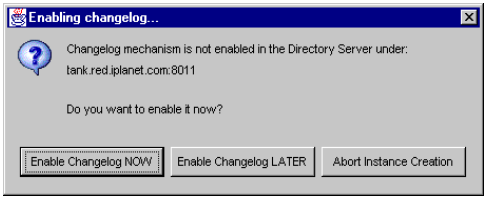
View Name — Enter a name of any length that more fully describes the View ID. The default is the View ID.

View ID — Enter up to five characters to represent the view ID. The default is CVn.

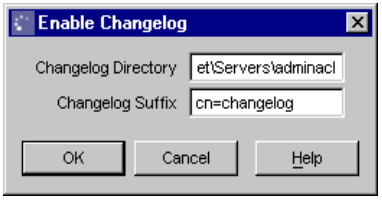
View Base DN — Enter the subtree DN where this view is located. The default is o=CV*x*, where *x* is the next successive integer following the last instance created.

Data Server URL	From the drop-down list, select the data server from which the new instance should be created. You can also type in a data server URL.
Data Server Bind DN	Enter a DN to be bound to the data server URL for access rights to the subtree. The default is `cn=Directory Manager`.
Data Server Bind Password	Enter the password associated with the data server bind DN.
Perl Script Absolute Path	This field only appears if you are creating an instance of the Universal Text Connector. Enter a path and file name if you want Meta-Directory to extract the Perl script and then parse a third-party database. Instead of entering the information here, you can alternatively supply the path and file name in the Script tab window. For information on this procedure, see “To include scripts for the Universal Text Connector,” on page 149.

3. Click OK in the New Instance Creation dialog box. If the changelog is not enabled, the following message appears:



If you click Enable Changelog NOW, the Enable Changelog dialog box appears.



a. Enter a directory where you want to store the changelog. For Solaris systems, make sure you change the directory permission mode of the file to allow the console to create the changelog directory. It is recommended that you issue the following command against the directory where you want to create the changelog directory:

```
chmod -R 777
```

b. Either accept the changelog suffix default or provide your own.

c. Click OK. A message appears reminding you to restart the Directory Server.

4. The Load Schema appears whenever you add a new view if the base entry does not exist. If the schema already exists, you do not need to click Yes. If you click Yes, a dialog box appears that informs you of progress while the installer is loading the schema. The message “Instance Creation Succeeded” appears after the instance has been created.

5. Restart the Directory Server:

a. From the iPlanet Console, select the Directory Server object, then right-click.

b. From the context menu, select Stop Server. Click Yes in the Stop Server message box. A message appears stating that the Directory Server has been stopped.

c. Select the Directory Server object again, then right-click.

d. From the context menu, select Start Server. A message appears stating that the Directory Server has been started.

To remove a connector instance

1. From the iPlanet Console, highlight the instance you want to delete, then right-click. A context menu appears.

2. Select Remove Server and click yes in the message box. Another message box appears.

- If you click Yes, the connector view is removed. If it is a participating view of the join engine, it will also be removed. This does not remove connector view data from the LDAP DIT, however.
- If you click No, the connector instance is removed, but the connector view will remain.

Configuring a Universal Connector Instance

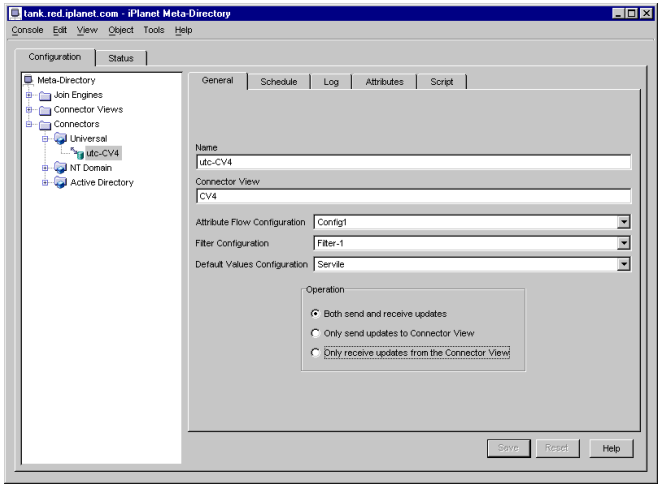
You can use the windows associated with the instance to perform the following tasks:

- Specify how updates are handled
- Configure the schedule from and to connector views
- To configure attributes for log files
- Add external attributes for connectors
- Include scripts

The examples for the procedures below apply to the Universal Text Connector. This connector has an extra window, called Script, that is not present when you configure the NT Domain Connector or Active Directory Connector.

To specify how updates are handled

1. Click on the instance you want to configure. The General tab window appears. The Name and Connector View Location fields are read-only. You specified this information when you created the instance.

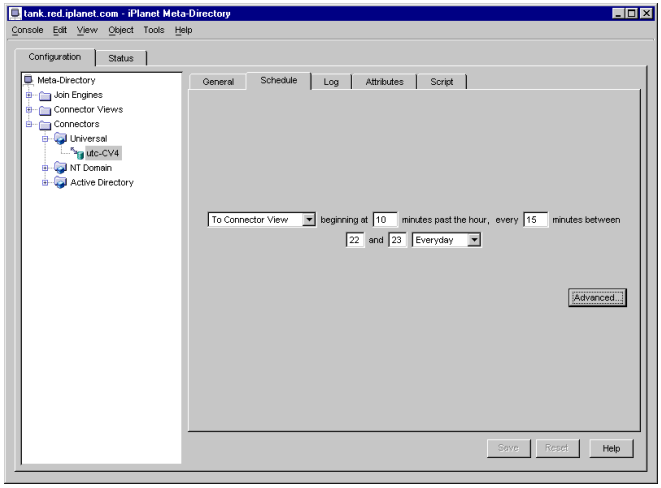


2. Select the desired attribute flow, filter, and default values from the drop-down lists. You configured these values in Chapter 7, “Connectors and Connector Rules.”

3. Indicate how you want to send and receive updates by selecting the appropriate radio button, then clicking Save before continuing. The updates can be modifications, deletions, and additions to the external directory and/or to the connector view.

To configure the schedule from and to connector views

1. Click the Schedule tab. The Schedule window appears.

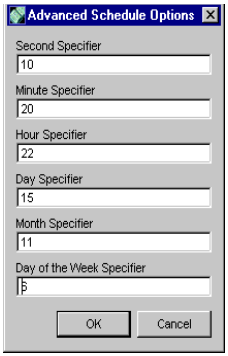


2. Select either To Connector View or From Connector View, and indicate in the text boxes when updates should be sent.

   For every scheduled synch cycle, the Universal Connector launches the accessor to create a dump file of all the external directory entries. If the accessor did not succeed in creating the dump file and it is empty, the Universal Connector thinks that all the entries from the external directory are deleted.

   Consequently, the Universal Connector then attempts to delete the external directory-owned entries from the connector view. However, after restarting the server and restarting the Universal UTC, the accessor launches and a new dump file is created. All the external directory-owned entries are then resynchronized back to the connector view.

3. Optional: Alternatively, you can manually provide settings in a tabular format. To do so, click Advanced. The Advanced Schedule Options dialog box appears.



Provide values for the following fields:

Second Specifier	Enter a value from 0 to 59.
Minute Specifier	Enter a value from 0 to 59.
Hour Specifier	Enter a value from 0 to 23.
Day Specifier	Enter a value from 1 to 31.
Month Specifier	Enter a value from 1 to 12.
Day of the Week Specifier	Enter a value from 0 to 6, where 0 is Sunday and 6 is Saturday.

You can use either a single number or an expression as described below.

Expression	Definition
*	Matches any value.
`*/step`	Matches any value in steps. For example, */2 matches 0,2,4,6 ... up to the maximum allowed value for values that start with zero, or it matches 1,3,5,7 ... up to the maximum value allowed.

Expression	Definition
`x-y`	Specifies a range where: Both x and y are greater than or equal to the minimum allowed value. y is less than or equal to the maximum allowed value. x is less than y. The expression matches any value in the range.
`x-y/step`	Specifies a range as above, but with a step value that is not necessarily 1.
`x`	Specifies a single number within the allowed range.
`x/step`	Matches any value starting at x and then at x + step, x + 2*step, and so forth.
`x,y,z...`	Specifies a comma-separated list of values.
`x-y,z-q`	Specifies a comma-separated list of ranges.

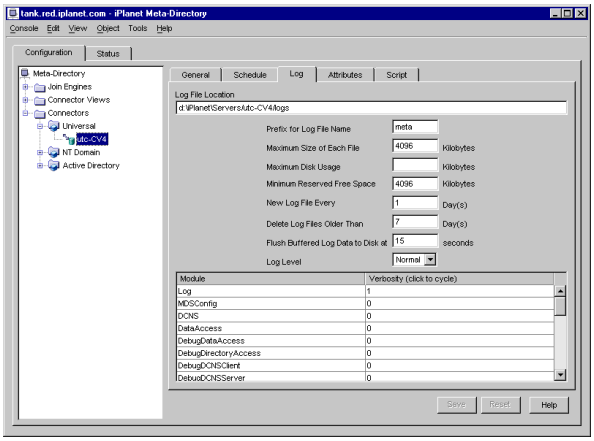
Lists can also contain both single values and ranges such as 1,2,5-7,10/5.

The scheduler operates once every second, so the finest granularity occurs every second.

4. Click OK to return to the Schedule window, then click Save before continuing.

To configure attributes for log files

1. Click the Log tab. The Log window appears.



2. Provide information for the following fields:

Log File Location

Specifies the directory in which the log files reside. To specify a directory other than the default, enter the full path name of the directory from the system where the connector is installed.

Prefix for Log File Name

Specifies the name you want to precede the log file of the form:

name-yyyymmdd-nn`.log`

For example, if you chose `delta` as the prefix, it would appear as shown in the following example:

```
delta-20010521-01.log
```

Maximum Size of Each File

Specifies the maximum size of the log file. After the file reaches this size, the service creates a new log file. The default is 4096 KB.

Maximum Disk Usage

Specifies the maximum disk usage set aside for logging. When the maximum disk usage is reached, the oldest log file is deleted. By default, the maximum disk usage is 100 KB.

The maximum available disk space must be at least twice the size of the maximum log file size, because the service always stores one backup log file. If the maximum available disk space is less than twice the maximum log file size, the new file cannot reach its maximum size.

Minimum Reserved Free Space	The service reserves disk space to use for log file storage. By default, the service reserves a minimum of 5000 KB of space.
New Log File Every	Determines the length of time that a log file should record the service's activities before creating a new log file. If the maximum size of the log file is reached first, a new log file is created. This value must be less than the value in the Delete Log Files Older Than field.
Delete Log Files Older Than	Determines when the service should automatically delete old log files. This value must be greater than the value in the New Log File Every field.
Flush Buffered Log Data to Disk at	Determines when the log data buffered in memory is to be flushed to the disk. If you enter 0 in this field, every message is flushed. Although this is helpful for debugging, a performance degradation will occur.
Log Level	The choices are Off, Normal, and Debug. • Off suppresses logging. • Normal logs minimal information. Maximum disk space may be small, and new files are created infrequently. • Debug logs maximum information. Maximum disk space is large, and new files may be created frequently. You can switch modes without restarting the connector.
Verbosity	Specifies how much information is logged for a given module. The modules are functional areas of the Join Engine and connector, and are created during installation. Choose a verbosity level of 0 - 3 by selecting a module, then clicking on its corresponding level until you have selected the desired number. The higher the number, the greater the amount of information displayed.

3. Click Save.

To add external attributes for connectors

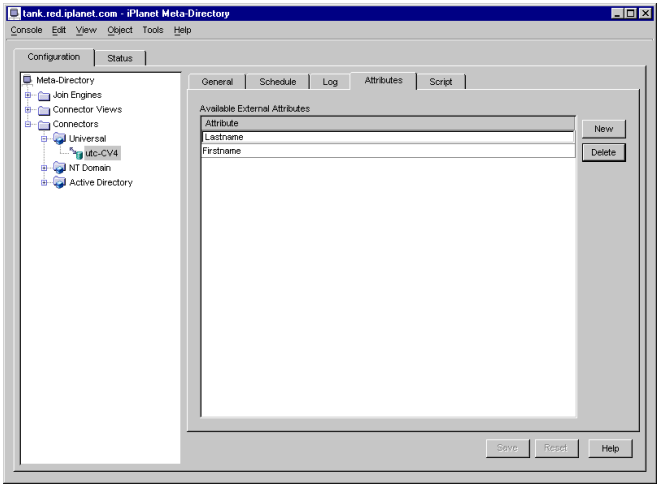
You can create a list of attributes that you want to flow from the external data source for UTC-based connectors. For example, in a `task.cfg` file for the Universal Text Parser, you might have the following line format:

```
LineFormat=ALTEREGO:uid,REALNAME:cn,LASTNAME:sn
```

Since `ALTEREGO`, `REALNAME`, and `LASTNAME` are not LDAP attribute names, you would want to declare `uid`, `cn`, and `sn` as attribute names, which will correspond to the schema defined in the directory DIT.

You can store the external attributes as described in the following procedure.

1. Click the Attributes tab. The Attributes window appears.

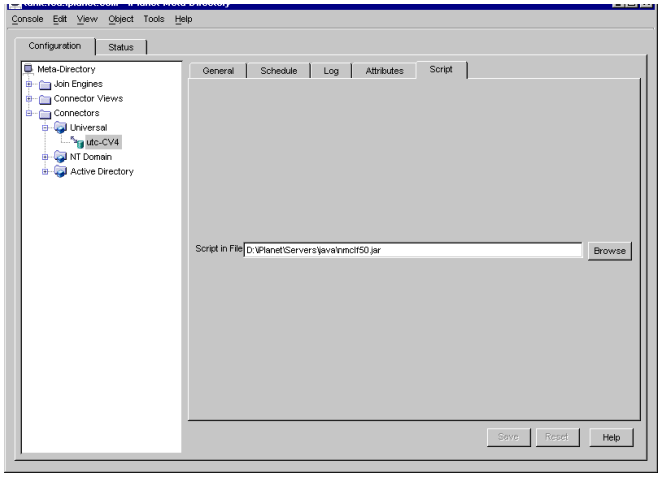


2. Click New. A blank field appears below the Attribute label.
3. Click within the blank field, then type the name of an external attribute you want to map to an internal attribute.
4. Repeat the steps above to add other attributes, then click Save. The order of the attributes you create is unimportant. After you click Save, they are sorted when you refresh the console.
5. See “To Configure an Attribute Flow Rule,” on page 129 to map the external attributes with connector view attributes.

To include scripts for the Universal Text Connector

You can provide special settings for the Universal Text Connector specific to the source directory on which it operates. This is handled by a plug-in tab called Script.

1. Click the Script tab. The Script window appears.



2. Specify a path and file name in the field, or click Browse to select the path. Meta-Directory will extract the Perl script and parse a third-party database. Do not specify a package name in the script.

   The Browse button is enabled and error checking implemented only if the Universal Connector is installed on the same system as the console. If the Browse button is disabled, you need to type in the absolute path for the script where the connection is installed.

3. Click Save.

Restarting a Connector Instance

You must restart the connector instance to activate your configuration. Both instance-specific and shared configurations will not become effective for a given instance until you have restarted the instance.

1. Stop the connector by right-clicking on the connector instance. A context menu appears.
2. Select Stop Server. A message appears stating that the stop command has been issued to the component.
3. Start the connector by right-clicking on the connector instance. A context menu appears.
4. Select Start Server. A message appears stating that the start command has been issued to the component.

Implementing the Configuration

After you start the join engine and enable the connector view, your data can flow to the meta view. The following sections provide procedures for doing these tasks.

Starting the Join Engine

Before you start the join engine, ensure that you have already enabled the changelog in the Directory Server configuration.

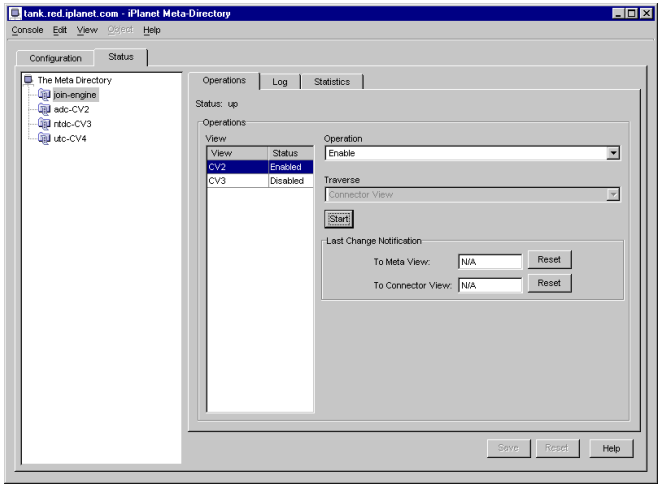
To start the join engine

1. Select the join-engine object from the navigation tree and right-click. A context menu appears.
2. Select Start Server. A message appears stating that the server has been started.

You can also start the server from the iPlanet Console. To do this, select the Join Engine object and right-click. Select Start Server from the context menu.

Enabling the Connector View

1. From the iPlanet Meta-Directory window, click on the Status tab.
2. Click on the Join Engine object. The Operations tab window appears.



3. Select the participating view you want to enable.
4. Select Enable from the Operation list menu, then click Submit Request.

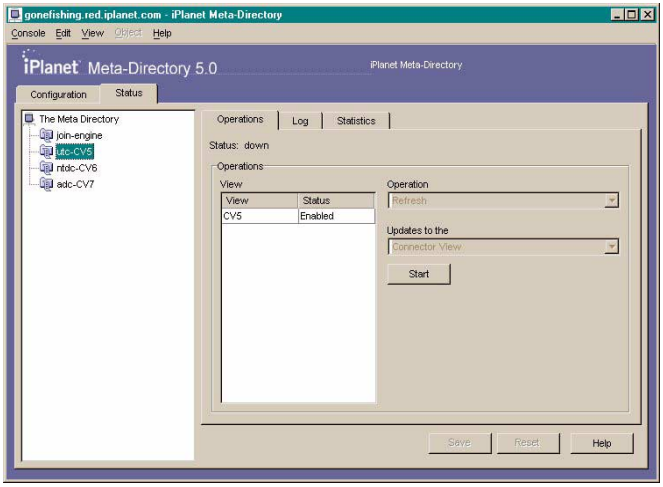
   This option disables the Traverse drop-down menu. You can only enable the participating view if the configuration for setting up the view is valid. Any error in the configuration automatically changes the view to a disable status.
5. Select Refresh from the Operation List Window, then select either Meta View or Connector View from the Traverse menu list.
6. Click Start.

Refreshing the View

You can optionally refresh the view if you want to observe updates immediately and bypass the regularly scheduled refresh synchronization. Note that after any type of refresh, you might see a “None” group in the meta view Contents or connector view Contents, particularly with non Primary Domain Controller systems. “None” is a valid group in Windows NT.

1. From the iPlanet Meta-Directory window, click on the Status tab.

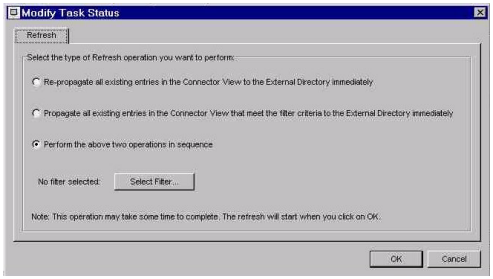
2. Click on the NT Domain connector instance object. The Operations tab window appears. The only operation available is Refresh.



3. In the “Updates to the” drop-down list, select either External Directory or Connector.

4. Click Start. The Modify Task Status dialog box appears. If you are refreshing the connector view, the following version of the box appears:

If you are refreshing the external directory, the following version of the box appears:



You must select a filter for the second and third options. Only filters configured for the "NoSubtreesExcept" option are displayed when you click Select Filter, not filters configured for the "AllSubtreesExcept" option.

Configuring the Universal Text Parser

The Universal Text Parser enables you to link an external data source with the Meta-Directory join engine. With the Universal Text Parser, you can synchronize a wide variety of text-based data with your Meta-Directory views.

This chapter has the following sections:

- An Overview of the Universal Text Parser
- The task.cfg Configuration File
- Setting Up the Universal Text Parser
- Creating a task.cfg File for Comma-Separated Value Data Files
- Creating a task.cfg File for Name-Value Pair Data Files
- Creating a task.cfg File for LDIF Data Files

An Overview of the Universal Text Parser

The *Universal Text Parser* (UTP) is a generic text file parser and generator that you can use to build connectors for sources that supply data in a text-based format. You customize the Universal Text Parser by describing the text-based data that is input to the Universal connector. You can also specify the output of the Universal connector, enabling you to fully synchronize data with an external data source.

The Universal Text Parser uses Perl scripts to provide the linkage between your external data source and the Universal connector. To use the Universal Text Parser, you *export* your database to ASCII text which the Universal Text Parser reads and synchronizes with the connector view. The Universal Text Parser generates data to a file that your external database can *import*. (The external database exports and imports data.)

Meta-Directory supplies three pre-configured configuration setups for the most common text-based data representations:

- Comma-separated values
- Name-value pairs
- LDIF

You can configure the Universal Text Parser to support data in just about any text format. iPlanet Professional Services offers assistance if you need to create a custom Universal Text Parser module to adapt to different data formats.

Universal Text Parser Modules

The Universal Text Parser consists of the following program modules:

Table 9-1 Universal Text Parser modules

`task.cfg`	Text configuration file used to describe data that you want to flow into and out of the connector view. You must customize this file for each external data source you want to synchronize with your Meta-Directory views.
`template.pl`	Perl library that implements the interface between the script and the Universal Text Connector.
`universal.pm`	Perl module that contains the main engine for interpreting user settings in the configuration file.
`textparser.pm`	Perl module that contains a generic set of routines for parsing files.
`connectorutils.pm`	Perl module that contains various generic routines that a Perl connector may use.

The program modules are located in the following directory of your installed Meta-Directory product:

NETSITE_ROOT`/bin/utc50/install/templates/universalparser`

In the path above, *NETSITE_ROOT* is the root directory of your iPlanet Meta-Directory installation.

CAUTION Do not modify any of the Universal Text Parser modules other than the `task.cfg` configuration file.

The task.cfg Configuration File

To synchronize information between a text-based data source and your Meta-Directory views, you must properly configure an instance of the Universal Text Parser for each separate text-based data source you need to synchronize.

You configure the linkage between your external data source and the Universal Text Parser by creating a custom configuration file, named `task.cfg`. The configuration file describes the data format and file specifics of the text-based data.

The `task.cfg` configuration file describes the text-based data that your external data source exports to the Universal Text Parser. The `task.cfg` configuration also describes the text-based output of the Universal Text Parser, which you can import into your external database.

Pre-configured Configuration Files

The `task.cfg` configuration file specifies the details of your text-based data. The iPlanet Meta-Directory package supplies three pre-configured files that you can use as the boilerplate for your `task.cfg` file. Each pre-configured file supports a specific format of text-based information, as shown in Table 9-2.

Table 9-2 Pre-configured templates for the task.cfg configuration file

Format Supported	Boilerplate File Name	Description
Comma-Separated Value	`csv.cfg`	Supports the synchronization of data formatted as a comma-separated value (CSV) list.
Name-Value Pair	`nvp.cfg`	Supports the synchronization of data formatted as a name-value pair (NVP) list.
LDIF	`ldif.cfg`	Supports the synchronization of data formatted in LDIF.

Each of the pre-configured boilerplate files has an example input file that matches the settings in the boilerplate. Accordingly, these files are named `sampledata.csv`, `sampledata.nvp`, and `sampledata.ldif`, and can be found in the directory containing the rest of the Universal Text Parser files.

To customize a configuration file for your needs, modify the appropriate boilerplate file to fit the entry attributes and mappings of your data stream. Details on how to customize each file are located within the file as comments.

Non-Conforming Formats

If you have text-based data that does not conform to the format of one of the pre-configured configuration files, you can modify the `task.cfg` file to suit your needs.

The modification of the supplied `task.cfg` file is supported through the services of the iPlanet Professional Services department. If your needs require a more detailed modification of the Universal Text Parser modules than what is described in this chapter, please contact your iPlanet Professional Services representative.

Setting Up the Universal Text Parser

To set up the files in the Universal Text Parser, do the following:

1. Copy the following Universal Text Parser modules to the directory containing the text-based data you want to process:
   - `template.pl`
   - `connectorutils.pm`
   - `textparser.pm`
   - `universal.pm`

   These modules are described on page 156.
2. Copy the appropriate boilerplate file to the directory used in Step 1 and rename the file to `task.cfg`.

   The boilerplate file is either `csv.cfg`, `nvp.cfg` or `ldif.cfg`, as described in the section “Pre-configured Configuration Files,” on page 157.
3. Modify the boilerplate file (now named `task.cfg`) according to the instructions in one of the following sections:
   - Creating a task.cfg File for Comma-Separated Value Data Files
   - Creating a task.cfg File for Name-Value Pair Data Files
   - Creating a task.cfg File for LDIF Data Files

When modifying the `task.cfg` file, remember that regular expressions are parsed using the Perl regular expression syntax, and not by the regular expression syntax used by UNIX® systems.

4. If you have not already done so, configure the Universal Connector as described in Chapter 8, "Configuring The Universal Connector."
   - When you create the connector instance, ask the system to load the schema, as described on page 141.
   - Select the connector instance, then go to the Script tab.
   - Enter the path name and file name of the script `.pl` file.
5. Restart the connector instance, as described on page 150.

Creating a task.cfg File for Comma-Separated Value Data Files

Use the `csv.cfg` file as a boilerplate for your `task.cfg` configuration file if your data is formatted in a comma-separated value (CSV) ASCII text file.

Before You Begin

Consider the following advisory information when synchronizing a comma-separated value data file with the Universal Text Parser:

- All attribute names in the `LineFormat` and `ImportLineFormat` statements must map to the attribute names in the declared LDAP schema before the Universal Text Parser can process entries. By default, the name-value pair boilerplate file uses `inetOrgPerson` as the default LDAP schema.
- When modifying the `task.cfg` text file, be sure to use a text editor that does not delete trailing whitespace from the end of lines. Removing the whitespace at the end of certain lines might lead to errors when the Universal Text Parser interprets these lines.

Creating the File

After you have set up the Universal Text Parser, make the following modifications to the `task.cfg` file to tailor the configuration file to your specific data needs:

1. Modify the `LineFormat` statement so that it contains the attribute names of the data you want to synchronize. Separate each attribute name with a comma.

   a. If needed, map the attribute names in the data file you are inputting to the names defined in the specified LDAP schema.

      All attribute names must be contained in the LDAP schema that is specified in the `task.cfg` file. By default, this is `inetOrgPerson`. If the attribute names in your input file differ from those contained in the declared schema, you must map the attribute names in your input file to ones defined in `inetOrgPerson`.

      Map the attribute names by first specifying the external database attribute name, a colon, then the associated LDAP attribute name. Separate attributes with a comma, as shown in the following example:

      ```
      LineFormat=ID:uid,NTDOMAINACCOUNT:cn,SURNAME:sn,
      GIVENNAME:givenName,INITIALS:initials
      ```

      The order in which you specify the attributes is important; it must match exactly the order of the data supplied in your comma-separated value data file.

      Normally, you can find the order of the attributes listed in the header of the data file. For an example of this, review the comma-separated value data file that is supplied with the product, `sampledata.csv`.

   b. Specify a “format” if your data file uses a separator other than a comma.

      The format can be either a character delimiter, a regular expression, or a field size (which is indicated by a digit). To specify a format, follow the external database attribute name with a pound sign (#) and list the necessary format, as shown below in the example for `ImportLineFormat`.

2. Specify the `ImportLineFormat` if you are going to import into your external database any modifications that are generated through the Universal Text Connector.

   a. In the `ImportLineFormat` statement, supply the names of attributes generated by the Universal Text Connector in the order that you need them to be written to the output file. This is the order that your external database will import the entry attributes.

b. Specify the LDAP attribute name, a colon, then the external database attribute name. Follow the external database attribute name with a pound sign (#) and the delimiter, which in the following example is a comma. Separate each attribute listed with a comma, as shown in the following example:

```
ImportLineFormat=operation#,,uid:ID#,,
cn:NTDOMAINACCOUNT#,,   sn:SURNAME#,,
givenName:GIVENNAME#,,initials:INITIALS#,,
```

As shown in the example above, the first data value is an operation, the value of which can be either `add`, `modify`, or `delete`. The operation indicates the database action needed to process the respective entry.

`ImportLineFormat` is like the `LineFormat` statement in that you must map the external database attributes to attribute names defined in the specified LDAP schema, which is `inetOrgPerson` by default.

If needed, you can specify a different LDAP object class using the `AdditionalAttributes` statement. Note, however, that you can use only a single schema in your comma-separated value file.

3. Modify the `IndexAttribute=` statement if you need to provide an attribute to index other than `cn`. By default, the line reads:

```
IndexAttribute=cn
```

To index on a different attribute value, specify the required attribute name in this statement. When specifying an attribute to index, use an attribute that contains unique values among each of the database entries. It is important, however, that you not use the distinguished name (`dn`) as the `IndexAttribute` value.

4. Specify the name of the file containing your comma-separated value data on the line that reads:

```
InputFile=%ScriptBase%sampledata.csv
```

On this line, replace the text `sampledata.csv` with the name of your data file.

`InputFile` is generated by `DumpCommand`, and `OutputFile` is used by `ImportCommand`. By default, the `InputFile` statement declares that your data file is located in the same directory as the Universal Text Parser modules. Because of this, you must copy your data file to this directory so the Universal Text Parser can read the file. If needed, you can place your data file in a different directory; be sure to supply the full path and file name to your data file in this statement.

5. Configure the `ImportCommand` statement if you are going to import data into your external database. To do so:
   a. Uncomment the `ImportCommand` statement by removing the pound sign (#) listed at the beginning of the statement.
   b. Specify the import command your external database uses.
6. If it is supported, supply the command that exports the data from your external database using the `DumpCommand` statement:
   a. Uncomment the `DumpCommand` statement by removing the pound sign (#) listed at the beginning of the statement.
   b. Specify the export command your external database uses.

Creating a task.cfg File for Name-Value Pair Data Files

Use the `nvp.cfg` file as a boilerplate for your `task.cfg` configuration file if your data is formatted in a name-value pair (NVP) ASCII text file.

Before You Begin

Consider the following advisory information when synchronizing a name-value pair data file with the Universal Text Parser:

- All attribute names listed in the data file must map to the attribute names contained in the declared LDAP schema before the Universal Text Parser can process the entries. By default, the name-value pair boilerplate file uses `inetOrgPerson` as the default LDAP schema.
- When modifying the `task.cfg` text file, be sure to use a text editor that does not delete trailing whitespace from the end of lines. Removing the whitespace at the end of certain lines might lead to errors when the Universal Text Parser interprets these lines.

- The name-value pair for each entry should include the object class of the attributes for that record. For example, the following line shows a valid example entry in an name-value pair text file:

```
uid=nvp1_uid
ObjectClass=top
ObjectClass=person
ObjectClass=organizationalperson
ObjectClass=inetOrgPerson
cn=nvp1_cn
sn=nvp1
mail=nvp1@iplanet.com
title=Title for nvp1
description=This is the description for nvp1
```

- If you are importing data into your external database that the Universal Text Parser has generated, the external database must not assume the order of the attributes generated for any given entry.

Creating the File

After you have set up the Universal Text Parser, make the following modifications to the `task.cfg` file to tailor the configuration file to your specific data needs:

1. Modify the `IndexAttribute=` statement if you need to provide an attribute to index other than `cn`. By default, the line reads:

   ```
   IndexAttribute=cn
   ```

   To index on a different attribute value, specify the required attribute name in this statement. When specifying an attribute to index, use an attribute that contains unique values among each of the database entries. It is important, however, that you not use the distinguished name (`dn`) as the `IndexAttribute` value.

2. Specify the name of the file containing your name-value pair data on the line that reads:

   ```
   InputFile=%ScriptBase%sampledata.nvp
   ```

   On this line, replace the text `sampledata.nvp` with the name of your data file.

   By default, the `InputFile` statement declares that your data file is located in the same directory as the Universal Text Parser modules. Because of this, you must copy your external data file to this directory so the Universal Text Parser can read the file. If needed, you can place your data file in a different directory; be sure to supply the full path and file name to your data file in this statement.

3. Configure the `ImportCommand` statement if you are going to import data into your external database. To do so:

   a. Uncomment the `ImportCommand` statement by removing the pound sign (#) listed at the beginning of the statement.

   b. Specify the import command that your external database uses.

4. If it is supported, supply the command that exports the data from your external database using the `DumpCommand` statement:

   a. Uncomment the `DumpCommand` statement by removing the pound sign (#) listed at the beginning of the statement.

   b. Specify the export command that your external database uses.

Creating a task.cfg File for LDIF Data Files

Use the `ldif.cfg` file as a boilerplate for your `task.cfg` configuration file if your data is formatted in LDIF.

Before You Begin

Consider the following advisory information when synchronizing an LDIF data file with the Universal Text Parser:

- All attribute names in the `ImportLineFormat` statement must map to the attribute names in the declared LDAP schema before the Universal Text Parser can process the entries. By default, the name-value pair boilerplate file uses `inetOrgPerson` as the default LDAP schema.

- When modifying the `task.cfg` text file, be sure to use a text editor that does not delete trailing whitespace from the end of lines. Removing the whitespace at the end of certain lines might lead to errors when the Universal Text Parser interprets these lines.

  In particular, note that the two lines `ImportAttributeNamesSeparator` and `AttributeNamesSeparator` must both end with a trailing whitespace.

- If you specify an attribute flow rule, you turn on attribute data flow and you must provide the mappings. If you do not specify an attribute flow rule (if you use only entry level data flow), then all the mappings are configured in the `LineFormat` statement in your `task.cfg` file.

Creating the File

After you have set up the Universal Text Parser, make the following modifications to the `task.cfg` file to tailor the configuration file to your specific data needs:

1. Specify the `ImportLineFormat` if you are going to import into your external database any modifications that are generated through the Universal Text Connector.

   a. In the `ImportLineFormat` statement, supply the names of attributes generated by the Universal Text Connector in the order that you need them to be written to the output file, with each attribute separated by a comma. For example:

   ```
   ImportLineFormat=distinguishedname:dn,
   operation:changetype,objectclass,uid,cn,sn,mail,
   title,description
   ```

   The Universal Text Parser will not generate any attributes excluded from this line.

   b. Before listing the attributes you want to generate, you must specify two special values to support data output in LDIF. You specify these values as follows:

   ```
   distinguishedname:dn,operation:changetype,
   ```

2. Modify the `IndexAttribute=` statement if you need to provide an attribute to index other than `cn`. By default, the line reads:

   ```
   IndexAttribute=cn
   ```

   To index on a different attribute value, specify the required attribute name in this statement. When specifying an attribute to index, use an attribute that contains unique values among each of the database entries. It is important, however, that you not use the distinguished name (`dn`) as the `IndexAttribute` value.

3. Specify the name of the file containing your LDIF data on the line that reads:

   ```
   InputFile=%ScriptBase%sampledata.ldif
   ```

   On this line, replace the text `sampledata.ldif` with the name of your data file.

   By default, the `InputFile` statement declares that your data file is located in the same directory as the Universal Text Parser modules. Because of this, you must copy your data file to this directory so the Universal Text Parser can read the file. If needed, you can place your data file in a different directory; be sure to supply the full path and file name to your data file in this statement.

4. Configure the `ImportCommand` statement if you are going to import data into your external database. To do so:

   a. Uncomment the `ImportCommand` statement by removing the pound sign (#) listed at the beginning of the statement.

   b. Specify the import command that your external database uses. For example, you could specify the following `ImportCommand` statement:

      ```
      ImportCommand=myDBUtil -l administrator -p password -f
      "d:\\iplanet\\servers\\utc-cv3\\logs\\test.out"
      ```

5. If it is supported, supply the command that exports the data from your external database using the `DumpCommand` statement:

   a. Uncomment the `DumpCommand` statement by removing the pound sign (#) listed at the beginning of the statement.

   b. Specify the export command that your external database uses.

Chapter 10

Configuring the NT Domain Connector

This chapter discusses configuration factors specific to the NT Domain Connector, which provides bi-directional synchronization of NT user and group data into its connector view.

The topics in this chapter are:

- Installing the Connector
- Configuring a Participating connector view
- Creating Users
- Configuring Connector Rules
- Configuring a Connector Instance
- Activating the Configuration
- Implementing the Configuration
- Monitoring the Connector
- Data Flow for User and Group Entries
- Running the Connector from a Non-PDC Host
- Configuration Example

Installing the Connector

The following components must be installed before you install the connector:

- iPlanet Directory Server 4.*x* or 5.0, as described in the *Deployment Guide*. Restart the server after enabling the changelogs.
- iPlanet Meta-Directory 5.0, as described in the *Deployment Guide*. Make sure to select NT Domain Connector in the Components screen when you install Meta-Directory.

To create an NT Domain Connector instance

You can set connector parameters during instance creation or from the configuration file. The configuration file contains an extra parameter for log file size.

To set connector parameters during instance creation

1. From the iPlanet Console 5.0 window, right-click on Server Group. A context menu appears.
2. Select Create Instance Of, then select Meta-Directory NT Domain Connector. The New Instance Creation dialog box appears.
3. Provide input for the data fields. The dialog box for the NT Domain Connector contains the following additional fields:

NT Domain Name — Enter the name of the NT domain to synchronize.

NT Domain Host Read — Enter the host where NT user and group information is read from. Values can be the name of the primary domain controller (PDC) or backup domain controller (BDC). No value specifies the local host.

NT Domain Host Write — Enter the host where NT user and group information is written. If you are synchronizing a domain, the value must be the name of the PDC. No value specifies a local host.

NT Domain Log Level — Enter the log level for the task script and NT accessor utility. Values are as follows:

0 - None
1 - Error
2 - Warning
3 - Debug

To set connector parameters from the configuration file

1. Locate the `ntdc.conf` configuration file in the following directory:

   $Netsite_Root`/ntdc-`*ViewName*`/config/ntdc.conf`

   $Netsite_Root is the installed path for Meta-Directory. The default is `c:\iPlanet\Servers`. The *ViewName* is the name you provided in the New Instance Creation dialog box.

2. Provide values for the file parameters. The file appears as shown in the following example:

```
[NT Domain Connector Task]
NT Domain Name=MyDomain
NT Domain Host Read=MyDomainBDC
NT Domain Host Write=MyDomainPDC
NT Domain Connector Loglevel=1
NT Domain Connector Logfilesize=4096
Record Size Limit=50
```

   Most of the parameters correspond to those found in the New Instance Creation dialog box. However, the following parameters are specific to this file:

Logfilesize	Specifies the maximum size, in Kilobytes (Kb) for the `ntdc-ntacc` (NT accessor) and `ntdc-Perl` logs.
Record Size Limit	Specifies the maximum size, in kilobytes (Kb), for a block of data sent to the Universal Text Connector. You may consider increasing the limit beyond the default value of 50 for groups with more than 1000 users if you are running on a server with a large memory size, such as 515 Mb and 1 Gb of RAM. If you increase the limit beyond the default value, note that the performance degradation will be proportional to the size of the value.

To add the instance as a participating view

1. Right-click the Participating Views object. A context menu appears.
2. Select Add Participating View. The Select View dialog box appears.
3. Select the connector view you want to add or participate in a join/synchronization with the meta view.

4. Click OK. The view is added to the iPlanet Meta-Directory configuration tree.

To provide authorization

Provide authorization of created users for data server access. See “Setting Access Permissions,” on page 302 for the procedure.

Configuring a Participating connector view

If you have installed the join engine, you can configure a participating view for the NT Domain connector. Refer to the procedures in Chapter 6, “Views in Meta-Directory.”

Creating Users

The following procedures apply only to the meta view. If you have installed the join engine and want to create new entries, you should create them from the meta view. The connector view only reflects the contents of the external data source or meta view.

To create an NT Domain user in the meta view

1. Click on the Contents of the NT Domain meta view.

   From the menu bar, select Object > New > User. The Create New User dialog box appears.

2. Provide input in the required fields. A default user ID is generated when you enter the first and last names. Make sure that the User ID field is alphanumeric. See “User Entries,” on page 179 for attribute conventions and restrictions.

3. Click OK. The user name appears in the right pane of the Meta-Directory console.

You can also create NT Domain users in the meta view by using an LDIF file format within any LDAP client. The LDIF format should be similar to the structures of user entries and group entries, discussed on page 179 and page 184.

To modify an NT Domain user in the meta view

1. Click on the Contents of the NT Domain meta view.

2. Double-click on the NT Domain user you want to modify. The Edit Entry dialog box appears.
3. Alter the fields as needed, then click OK.

Configuring Connector Rules

You can configure the following types of rules for the NT Domain connector:

- Attribute Flow

  The connector uses attribute flow rules to specify which external data source attributes are mapped to which connector view attributes and vice versa. NT Domain provides the following preset configurations:

  - Minimal attribute set (`ntdc_minimal`)
  - Complete attribute set (`ntdc_all`)

  If you select one of the configurations, remove a few attributes, then save the configuration, you cannot revert to the original list of attributes by clicking Insert Defaults. Clicking this button populates the list box at the bottom of the window with default mappings that you can delete or change. If you do not select either configuration, the connector uses the default attribute flow.

- Default Attribute

  The connector applies preconfigured attribute rules to an entry in the external data source if no value is assigned to the same attribute in its corresponding entry in the connector view, or vice versa. A default attribute rule may also be configured.

- Filter

  The connector uses filtering rules to selectively exclude entries from the synchronization process.

To configure connector rules, see “Attribute Flow Rules,” on page 128, “Default Attribute Value Rules,” on page 132, and “Filter Rules,” on page 133.

Configuring a Connector Instance

Consider the following procedure an extension of the comprehensive configuration procedures in “Function of the Universal Connector,” on page 138 and “Configuring a Universal Connector Instance,” on page 142. You need to perform the following product-specific procedure for every NT Domain Connector.

1. To automatically configure attribute flow, proceed to Step a below. To manually configure, go to Step 2.

   a. Select the connector instance for which you want to provide attributes. The General window appears, as shown in .

   b. From the drop-down lists, select the desired attribute flow, filter, and default value configurations. The values that appear are derived from the rules you configured for the connector in the section “Configure Connector Rules,” on page 187.

      You can remove attributes from the complete set, if desired, before saving the configuration. The minimum configuration consists of the following attributes:

Application	Attributes
Users	cn ntUserDomainId objectclass sn uid
Local and Global Groups	cn ntGroupDomainId ntGroupType objectclass

      See Table 10-1 and Table 10-2 for the complete list of external attributes.

   c. Click Save, then go to step 3.

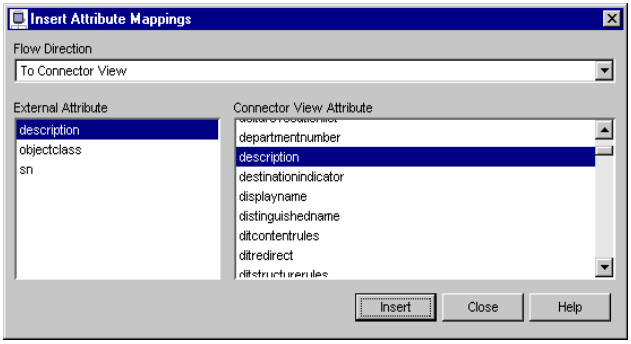
2. Optional: Manually configure the attribute flow by doing the following:

   a. Select the NT Domain Connector, then select the Attribute Flow tab, as shown in .

   b. Click New and enter a new attribute flow configuration name, then click OK.

c. Click Insert. The Insert Attribute Mappings dialog box appears. For both mapping types (locally owned objects and connector view-owned objects), map each attribute to itself for both flow directions (to connector view and from connector view).

For example, the following figure shows the `description` attribute being mapped to itself for a flow direction to the connector view. This would also have to be repeated for a flow direction from the connector view.



d. Click Save, select View from the menu bar, then select Refresh.

e. Select the desired NT Domain Connector instance. The General window appears, as shown in .

f. From the Attribute Flow Configuration drop-down list, select the attribute flow configuration name you created (Step b) and click Save. The name becomes available in the list after refreshing (Step d).

g. Select the desired filters and default values from the drop-down lists.

h. Select the operation you want to perform and click Save.

3. Configure the remaining windows for the connector instance. Begin with “To configure the schedule from and to connector views,” on page 143.

Activating the Configuration

You must restart the connector instance to activate your configuration. Both instance-specific and shared configurations will not become effective for a given instance until you have restarted the instance. If the entries you are saving preexist in an NT Domain connector view, see page 178 for advisory information.

1. Stop the connector by right-clicking on the connector instance. A context menu appears.
2. Click Yes to the prompt. A message appears stating that the stop command has been issued to the component.
3. Start the connector by right-clicking on the connector instance. A context menu appears.
4. Select Start Server. A message appears stating that the start command has been issued to the component.

NOTE To start the connector, you must be a member of the Administrators group on the primary domain controller.

Implementing the Configuration

After you start the join engine and enable the connector view, your data can flow to the meta view. The following sections provide procedures for doing these tasks.

Starting the Join Engine

Before you start the join engine, ensure that you have already enabled the changelog in the Directory Server configuration.

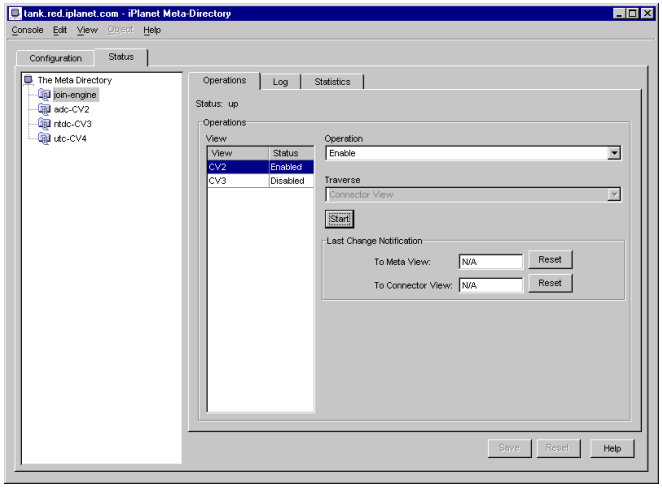
To start the join engine

1. Select the join-engine object from the navigation tree and right-click. A context menu appears.
2. Select Start Server. A message appears stating that the server has been started.

You can also start the server from the iPlanet Console. To do this, select the Join Engine object and right-click. Select Start Server from the context menu.

Enabling the Connector View

1. From the iPlanet Meta-Directory window, click on the Status tab.
2. Click on the Join Engine object. The Operations tab window appears.



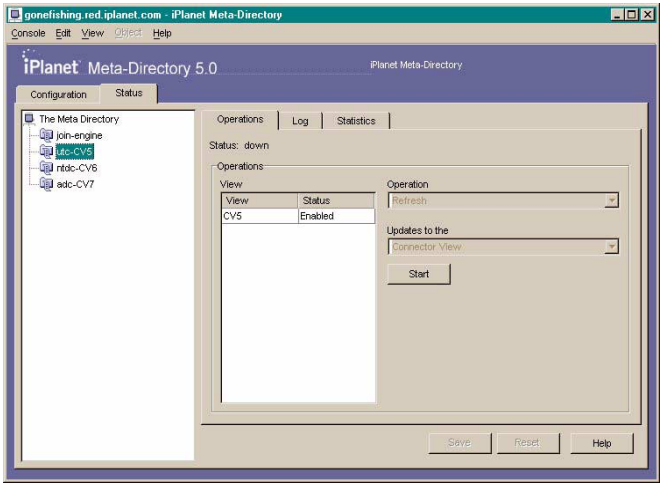
3. Select the participating view you want to enable.
4. Select Enable from the Operation list menu, then click Submit Request.

   This option disables the Traverse drop-down menu. You can only enable the participating view if the configuration for setting up the view is valid. Any error in the configuration automatically changes the view to a disable status.
5. Select Refresh from the Operation List Window, then select either Meta View or Connector View from the Traverse menu list.
6. Click Start.

Refreshing the View

You can optionally refresh the view if you want to observe updates immediately and bypass the regularly scheduled refresh synchronization. Note that after any type of refresh, you might see a “None” group in the meta view Contents or connector view Contents, particularly with non Primary Domain Controller systems. “None” is a valid group in Windows NT.

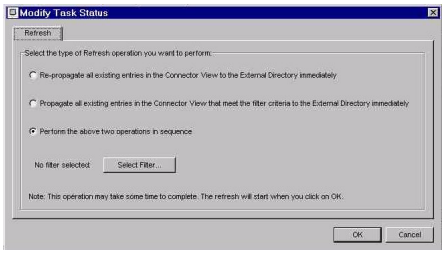
1. From the iPlanet Meta-Directory window, click on the Status tab.
2. Click on the NT Domain connector instance object. The Operations tab window appears. The only operation available is Refresh.



3. In the “Updates to the” drop-down list, select either External Directory or Connector.

4. Click Start. The Modify Task Status dialog box appears. If you are refreshing the connector view, the following version of the box appears:

If you are refreshing the external directory, the following version of the box appears:



You must select a filter for the second and third options. Only filters configured for the "NoSubtreesExcept" option are displayed when you click Select Filter, not filters configured for the "AllSubtreesExcept" option.

Monitoring the Connector

The NT Domain Connector provides logs at the following locations that enable you to monitor connector status.

General Connector	*InstallDir*/ntdc-*ViewName*/logs/meta-*date-index*.log
Accessor Utility	*InstallDir*/ntdc-*ViewName*/logs/ntdc-ntacc-*date-index*.log
Task Script	*InstallDir*/ntdc-*ViewName*/logs/ntdc-perl-*date-index*.log

For example, a general connector log entry might appear as follows:

```
meta-20010405-01.log
```

Data Flow for User and Group Entries

Entries in the NT Domain connector view must adhere to certain conditions to flow from the connector view into NT SAM. Note the following restrictions and advisory information:

- To prevent duplicate user IDs from occurring in the same connector view, the NT Domain connector views must be separate entities. A connector view should not be nested as a subtree of another connector view. That is, the connector view should be a flat tree that does not contain any subentries.
- Entries that preexist in an NT Domain connector view will not flow to the NT SAM database after the connector starts. To flow these entries, the NT Domain connector view must be an enabled participating connector view in the Join Engine. Refreshing the Meta View operation from the Join Engine will trigger the preexisting entries from the NT Domain connector view to flow to the NT SAM database.
- The Windows NT 4.0 registry has a limit of 40,000 users per primary domain controller (PDC). While this is not a hard-coded limit, surpassing this number of users could result in negative consequences for your Windows NT setup. If you do overload the Windows NT registry, the registry will become "full" and you will not be able to modify its contents; you will not even be able to delete the offending users to return the registry to a normal size.

  In this situation, the only choice is to reinstall the operating system since you will not be able to add or delete users, applications, and so forth. While Windows NT provides a registry editing tool, the tool is unable to delete records in the registry if it becomes overloaded. In addition, the Regedit tool is unsupported by Microsoft.

When setting up the Join Engine, you need to ensure that user and group entries meet the required criteria for NT Domain Connector views. The following sections discuss the requirements and list the available external attributes read from NT SAM for both user and group entries.

User Entries

You can create NT users in the connector view with any LDAP client by adhering to the attribute conventions shown in the following structure:

```
dn: uid=userid, cvroot_dn

uid: userid

cn: user_full_name

ntUserDomainId: domainname:uid

objectclass: top

objectclass: person

objectclass: organizationalPerson

objectclass: inetOrgPerson

objectclass: ntUser

sn: user_last_name
```

The following restrictions apply to user names:

- NT user names cannot contain the following characters:

  ```
  - ~ ! @ # $ % ^ & * ( ) _ + | \ : " , . < > / ?
  ```

- The username length of users added to NT SAM using the NT Domain Connector cannot exceed 20. The NTDC accessor does not check for username size when adding from the connector view to NT. You cannot delete invalid entries from the Administrator Tool, but you can delete them from the connector view and have the NT Domain Connector delete the invalid entries.
- The user name cannot consist solely of periods or spaces.

Table 10-1 shows the available external attributes for user entries.

Table 10-1 Attributes for User Entries

Attribute Name	Purpose
`cn`	Specifies the full name of the user. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
`description`	Provides comments associated with the user account.
`ntUserAcctExpires`	Specifies when the account expires. The format is: YYYYMMDDHHMMSS For instance, November 25, 2000 at 11 p.m. would be: 20001125230000
`ntUserAuthFlags`	Consists of a set of bit flags that define the user's operator privileges. The read-only values are in decimal. Possible values are: • `0x01`: Print operator privilege • `0x02`: Communications operator privilege • `0x04`: Server operator privilege • `0x08`: Accounts operator privilege This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
`ntUserBadPwCount`	Indicates the number of times a user attempted to log on to the account with an incorrect password. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
`ntUserCodePage`	Indicates the code page for the user's choice of language.
`ntUserComment`	Provides an additional comment field that is not exposed by the NT User Manager application.
`ntUserCountryCode`	Indicates the country / region code for the user's choice of language.
`ntUserDomainId`	Specifies the NT User ID, which must be of the form *domainname:username*.

Table 10-1 Attributes for User Entries *(Continued)*

Attribute Name	Purpose
ntUserFlags	Provides flags for several purposes.The read-only values are in decimal. Possible values are: • 0x0002: Account disabled • 0x0010: Account currently locked • 0x0020: Password not required • 0x0040: User cannot change password • 0x10000: Password should never expire The following values are not changeable by the connector: • 0x0100: Account to access this domain, but not any other domain it trusts • 0x0200: Default account type for the user • 0x0800: 'Permit to trust' account for a domain that trusts other domains • 0x1000: Computer account for an NT workstation or server that is a member of this domain • 0x2000: Computer account for the BDC that is a member of this domain
ntUserHomeDir	Specifies the user's home directory.
ntUserHomeDirDrive	Specifies the drive letter assigned to the user's home directory.
ntUserLastLogon	Specifies the last user logon. The format is: YYYYMMDDHHMMSS For instance, November 25, 2000 at 11 p.m. would be: 20001125230000 This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.

Table 10-1 Attributes for User Entries *(Continued)*

Attribute Name	Purpose
ntUserLastLogoff	Specifies the last user logoff. The format is: YYYYMMDDHHMMSS For instance, November 25, 2000 at 11 p.m. would be: 20001125230000 This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
ntUserLogonHours	Points to a 21-byte bit-string (168 bits) that specifies the times when the user can log on. Each bit represents a unique hour in the week. The first bit (bit 0, word 0) is Sunday, 0:00 to 0:59; the second bit (bit 1, word 0) is Sunday, 1:00 to 1:59; and so forth.
ntUserLogonServer	Specifies the name of the logon server to which logon requests are sent. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
ntUserMaxStorage	Indicates the maximum amount of disk space. No value means no limitation.
ntUserNumLogons	Indicates the number of times a user attempted to successfully log on to the account. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
ntUserParms	Provides a string for private data used by applications.
ntUserPasswordExpired	Contains password expiration information. The NT Domain Connector does not currently support password synchronization. If you create a user account on NT, the ntUserPassword attribute is not synched to the connector view. If you create a user entry in the connector view, specify a non-zero value to inform users that they have to change their password for the next logon. Turn this off by specifying zero. Note that you cannot specify zero to negate an expiration that has already occurred.

Table 10-1 Attributes for User Entries *(Continued)*

Attribute Name	Purpose
ntUserPrimaryGroupID	Specifies the relative ID of the user's primary global group.
ntUserPriv	Specifies the privilege level assigned to the user, which is read-only for the connector. Possible values are: • 0: Guest • 1: User • 2: Administrator This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
ntUserProfile	Specifies the path to the user's profile.
ntUserScriptPath	Indicates the path for the user's logon script.
ntUserUniqueId	Specifies the user's unique numeric ID. SAM defines this when the user is created. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
ntUserUnitsPerWeek	Indicates the time units for the user.
ntUserWorkstations	Specifies the names of the workstations where the user can log on. Commas (maximum of 8) must separate the names. If not present, no restrictions are applied.
objectclass	Specifies the object classes the connector assigns to a new user entry in the connector view. The values are inetOrgPerson and ntUser.
sn	Specifies the last name of the user. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
uid	Specifies the NT user ID. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.

Group Entries

The group entries in the connector view contain the list of member DNs. The connector view applies static group membership.

You can create NT groups in the connector view with any LDAP client by adhering to the attribute conventions shown in the following structure:

```
dn: cn=groupname, cvroot_dn

objectclass: top

objectclass: groupOfUniqueNames

objectclass: ntGroup

ntDomainGroupId: domainname:groupname

ntGroupType: grouptype (grouptype := "local" | "global")
```

The following restriction applies to group entries:

- When synchronizing local groups that contain members from a trusted domain, none of these entries are propagated to the connector view under the local groups.

Table 10-2 shows the available external attributes for group entries.

Table 10-2 Attributes for Group Entries

Attribute Name	Purpose
`cn`	Specifies the group name. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
`description`	Provides comments associated with the group.
`ntGroupAttributes`	Specifies the attributes of the group. This attribute cannot be modified from the connector view. It is only synchronized to the connector view if a modification occurred from NT.
`ntGroupDomainId`	Specifies the NT group domain ID, which must be of the form *domainname:groupname*.
`ntGroupId`	Specifies the relative identifier of the global group.
`ntGroupType`	Specifies the type of the group. Possible values are: • `local`: Local group • `global`: Global group

Table 10-2 Attributes for Group Entries *(Continued)*

Attribute Name	Purpose
`objectclass`	Specifies the object classes the connector assigns to a new group entry in the connector view. The values are `groupOfUniqueNames` and `ntGroup`.
`uniqueMember`	Specifies the DNs of group members. The user entry must have the same immediate parent as the group entry. This attribute contains the DNs of the group members that are in the same connector view subtree, as shown in the following example: `uniqueMember: uid=sharpie, ou=Employees, o=siroe.com`

Running the Connector from a Non-PDC Host

NT services are run by default in a system account that has admin rights, but only to the local machine. It cannot read NT SAM from another machine. To enable the connector to access the SAM database remotely, you have to set the user account that runs the service to an account that has administrator rights in the domain. It is recommended that you create a new account that has appropriate rights to manage NT services on the local system and access the NT SAM database on the PDC (not local).

The following steps explain the configuration required to synchronize data from a PDC other than your machine, which is not in the PDC domain. Before you begin, when you create an instance of the connector, the domain name should be the PDC's domain name, and the hostread and hostwrite should be the machine name of the PDC.

1. Enable trusted and trusting relationships.

   a. Add a trusted domain to your local machine, then add the trusting domain to the PDC with the same password.

   b. Add a trusted domain to the PDC, then add the trusting domain to your local machine with the same password.

   If you establish the trust relationships correctly, you should see a successful confirmation message. If the trust relationships are not established correctly, data cannot be synchronized.

2. Stop the NT Domain connector.
3. From the desktop of the local machine where the connector is installed, go to Settings > Control Panel > Services.
4. Select iPlanet NT Domain Connector.
5. Click the Startup ... button.
6. Select “This Account” and specify your domain iPlanet Administrator user name and password, then click OK.
7. Start the NT Domain Connector. If you have difficulty starting it from the service panel, start it from the console.

Configuration Example

The following example is intended as a quick reference you can use as a checklist. For complete configuration information, refer back to the earlier portions of this chapter.

Install the Connector

1. Ensure that iPlanet Directory Server 4.*x* or 5.0, and iPlanet Meta-Directory 5.0 are already installed.
2. Create a connector instance.

   During instance creation:

   a. From the iPlanet Console 5.0 window, right-click on Server Group. A context menu appears.

   b. Select Create Instance Of, then select Meta-Directory NT Domain Connector. The New Instance Creation dialog box appears.

   c. Provide input for the data fields. For View Name, use `NT`. For View ID, use `CV1`. For View Base DN, use `o=CV1`. For Schema, use `default`. For the remaining fields, see page 168.

   From the configuration file:

   a. Locate the `ntdc.conf` configuration file in the following directory:

      NetsiteRoot`/ntdc-`*ViewName*`/config/ntdc.conf`

b. Provide values for the file parameters. For details, see Step 2 on page 169.

3. Add the instance as a participating view.

a. Right-click the Participating Views object. A context menu appears.

b. Select Add Participating View. The Select View dialog box appears.

c. Select `NT` and click OK. The view is added to the iPlanet Meta-Directory tree.

4. Provide authorization. See “Setting Access Permissions,” on page 302.

Configure Connector Rules

1. Configure attribute flow.

a. Click on the NT Domain connector. The Attribute Flow tab window appears.

b. Select `ntdc_minimal` from the list of configurations.

c. In the Mapping Type drop-down list, select Mappings for connector view Owned objects.

d. Click Insert. The Insert Attribute Mappings dialog box appears. This displays a list of all available attributes from both the external data source and the connector view.

e. For Mapping Type, select Mapping for connector view Owned objects. For Flow Direction, select From connector view. For connector view Objectclass, select All Attributes.

f. For External Attribute, select `homephone`. For connector view Attribute, select `telephonenumber`.

g. Click Insert. The mapping for your configuration appears at the bottom of the Attribute Flow window.

h. Click Close, and then click Save from the Attribute Flow window.

2. Configure default attribute rules.

a. Click on the Default Values tab. The Default Values window appears.

b. Click New.

c. In the Name field, type in `NTDefault`. The name is echoed in the Configurations list box.

d. In the Attribute Destination drop-down list, select External Directory.

e. Click Add. Blank fields appear below the Attribute and Default Value fields.

f. Click within the blank Attribute field. A drop-down list appears. Select `givenname` from the list.

g. Double-click within the blank Default Value field and type in `surname`.

h. Click Save.

3. Configure filters.

a. Click on the Filters tab. The Filters window appears.

b. Click New. The Filter Name dialog box appears.

c. Type in `NTExclude` and click OK. The new name appears in the Filter Name list box.

d. Select From connector view.

e. Filter excluded data:

I. Provide a list of subtrees to exclude by selecting All Subtrees Except, then clicking Add. The Sub-tree DN dialog box appears.

II. Specify a subtree to exclude, such as `o=siroe,c=us`, then click OK. The subtree appears in the list box.

With this filter, entries in all subtrees that are not specifically excluded are included, no matter how you set the associated entry-level filters.

III. Filter back entries from the excluded subtrees using entry-level filters. Select the subtree you just created, select All Entries Except, then click Add. The Entry RDN dialog box appears.

IV. Specify an entry you want to include, such as `cn=Fred Scofflaw`, then click OK. The included entry appears in the list box.

The entry-level filters you apply affect only the entries found in the list of subtrees to include. The entries you specify here will filter through; all others are excluded.

f. Click Save.

g. From the menubar, select View > Refresh.

Configure a Connector Instance

1. Select the ntdc-NT connector instance. The General window appears.
2. Select the following from the drop-down lists:
   - For Attribute Flow Configuration, select `ntdc_minimal`.
   - For Filter Configuration, select NTExclude.
   - For Default Values Configuration, select NTDefault.
3. For Operation, select “Only receive updates from the connector view.”
4. Click Save. Leave the current values for fields in the Schedule, Log, and Attributes windows.

Restart the Connector Instance

1. Stop the connector by right-clicking on ntdc-NT. A context menu appears.
2. Click Yes to the prompt. A message appears stating that the stop command has been issued to the component.
3. Start the connector by right-clicking on ntdc-NT. A context menu appears.
4. Select Start Server. A message appears stating that the start command has been issued to the component.

Chapter 11

Configuring the Active Directory Connector

This chapter discusses configuration factors specific to the Active Directory Connector, which provides bi-directional synchronization of Active Directory user and group data into its connector view. During instance creation, the connector enables you to select from a default schema or ADSpecific schema. The default schema is based on the objectclasses of iPlanet Directory Server, whereas the ADSpecific schema represents all user and group attributes present in the Microsoft Active Directory.

The topics in this chapter are:

- Installing the Connector
- Configuring a Participating Connector View
- Creating Users
- Configuring Connector Rules
- Configuring a Connector Instance
- Restarting the Connector Instance
- Implementing the Configuration
- Monitoring the Connector
- Data Flow for User and Group Entries
- Configuration Example

Installing the Connector

The following components must be installed before you install the connector:

- iPlanet Directory Server 4.*x* or 5.0, as described in the *Deployment Guide*. Restart the server after enabling the change logs.
- iPlanet Meta-Directory 5.0, as described in the *Deployment Guide*. Make sure to select Active Directory Connector in the Components screen when you install Meta-Directory.

To install the Active Directory Server Interface (ADSI) package

1. Access the following URL:

   `http://www.microsoft.com/ntserver/nts/downloads/other/ADSI25/default.asp`

2. Select Version 2.5 for Intel x86 (English), then download `ads.exe`.
3. Run `ads.exe`.

To add a connector view instance

You can set configuration parameters during instance creation or from the configuration file. The configuration file contains extra parameters for setting the schema and modes.

To set configuration parameters during instance creation

1. From the iPlanet Console 5.0 window, right-click on Server Group. A context menu appears.
2. Select Create Instance Of, then select Meta-Directory Active Directory Connector. The New Instance Creation dialog box appears.
3. Provide input for the data fields. The dialog box for the Active Directory Connector contains additional fields. See Table 11-1 for a description of these fields.

Table 11-1 Dialog Box Parameters

Dialog Box Parameter	Definition
Domain	Specifies the domain where Active Directory exists.
Domain Controller User Name	Specifies the user name for the directory connector where Active Directory exists.
Domain Controller Password	Specifies the password associated with the domain controller user name.

Top Level Synch DN	Specifies the top level DN where Active Directory Connector synchronization occurs.
	
	Be advised that you should enter input in this field accurately. If the top level in Active Directory (from where users / groups are being synchronized) is under the 'Users' node in the Management Console (MMC), the entry should be:
	
	`cn=Users,dc=siroe,dc=com`
	
	If the user / group entries in Active Directory are to be added under a new organizational unit, such as `newou`, the entry should be:
	
	`ou=newou,dc=myhost,dc=com`
	
	All other users and groups under the DN mentioned above will be synchronized.

Host Name	Specifies the host address of the domain controller where the Active Directory exists.
Schema	Enables switching from the default schema to the ADSpecific schema. The ADSpecific schema provides additional attributes and object classes that are not ordinarily available in the default user / group object classes in the Directory Server, shown in Table 11-2 and Table 11-3. Every attribute name or object class of Active Directory for ADSpecific is prepended with `mdsAD` (ex: `mdsADTelephoneNumber`).

Table 11-1 Dialog Box Parameters *(Continued)*

Dialog Box Parameter	Definition
`Log Level`	Specifies the log level for the task script and accessor utility. Values are as follows: 0 - None 1 - Minimum 2 - Verbose 3 - Very verbose After you set the log level from the dialog box, you cannot change it from there. You must use the configuration file to change the log level.

To set configuration parameters from the configuration file

1. Locate the `adc.ini` configuration file in the following directory:

   NetsiteRoot`/adc-`*ViewName*`/config/adc.ini`

   Netsite_Root is the installed path for Meta-Directory. The default is `c:\iPlanet\Servers`. The *ViewName* is the name you provided in the New Instance Creation dialog box.

2. Provide values for the file parameters. The following table provides definitions for the configuration file parameters.

Configuration File Parameter	Definition

NTLMdomain`\`*user*	Specifies the pre-Windows 2000 abbreviated name of the domain to be synchronized. Example:
	`restaurants`
	instead of
	`restaurants.aus06.central.sun.com`

`username`	Specifies the Windows 2000 account name that the directory connector uses to authenticate Active Directory.
password	Specifies the password associated with the domain controller user name.
`adtopleveldn`	Specifies the top level DN where Active Directory Connector synchronization occurs.

Configuration File Parameter	Definition
`utctopleveldn`	Specifies the View Base DN as entered in the New Instance Creation dialog box.
`domain`	This parameter is not currently used.
`dc`	Specifies the host address of the domain controller where the Active Directory exists.
`schema`	Enables switching from the default schema to the ADSpecific schema. The ADSpecific schema provides additional attributes and object classes that are not ordinarily available in the default user/group object classes in the Directory Server, shown in Table 11-2 and Table 11-3. Every attribute name or object class of Active Directory for ADSpecific is prepended with `mdsAD` (ex: `mdsADTelephoneNumber`).
`logginglevel`	Specifies the log level for the task script and accessor utility. Values are as follows: 0 - None 1 - Minimum 2 - Verbose 3 - Very verbose After you set the log level from the dialog box, you cannot change it from there. You must use the configuration file to change the log level.
`finddeletedfreq`	Specifies the frequency of synchronization cycles that search for deleted entries. For instance, a value of 5 would search for deleted entries on the fifth cycle. This parameter is used in conjunction with the Schedule window, described on page 143.
`loggingsize`	Specifies the maximum size of the accessor log file in megabytes (Mb). The default is 4096 Mb.
`perllogfilesize`	Specifies the maximum size of the Perl log file in megabytes (Mb). The default is 4096 Mb.
`searchattrs`	Specifies a list of comma-separated Active Directory attributes. The list determines which attributes Active Directory retrieves. If you do not provide a list (blank), all attributes are selected.

To add the instance as a participating view

1. Right-click the Participating Views object. A context menu appears.
2. Select Add Participating View. The Select View dialog box appears.
3. Select the connector view you want to add or participate in a join/synchronization with the meta view.
4. Click OK. The view is added to the iPlanet Meta-Directory configuration tree.

To provide authorization

Provide authorization of created users for data server access. See “Setting Access Permissions,” on page 302 for the procedure.

Configuring a Participating Connector View

If you have installed the join engine, you can configure a participating view for the Active Directory connector. Refer to the procedures in Chapter 6, “Views in Meta-Directory.”

Creating Users

The following procedures apply only to the meta view. If you have installed the join engine and want to create new entries, you should create them from the meta view. The connector view only reflects the contents of the external data source or meta view.

To create an Active Directory user in the meta view

1. Click on the Contents of the Active Directory meta view.

   From the menu bar, select Object > New > User. The Create New User dialog box appears.

2. Provide input in the required fields. A default user ID is generated when you enter the first and last names. Make sure that the User ID field is alphanumeric and does not contain any of the following characters:

   `- ~ ! @ # $ % ^ & * ( ) _ + | \ : " , . < > / ?`

3. Click OK. The user name appears in the right pane of the Meta-Directory console.

You can also create Active Directory users in the meta view by using an LDIF file format within any LDAP client. The LDIF format should be similar to the structures of user entries and group entries, discussed on page 205 and page 207.

To modify an Active Directory user in the meta view

1. Click on the Contents of the Active Directory meta view.
2. Double-click on the Active Directory user you want to modify. The Edit Entry dialog box appears.
3. Alter the fields as needed, then click OK.

Configuring Connector Rules

You can configure the following types of rules for the Active Directory connector:

- Attribute Flow

  The connector uses attribute flow rules to specify which external data source attributes are mapped to which connector view attributes and vice versa. Active Directory provides the following preset configurations:

  - Minimal Attribute Set for Default Schema, which is the minimum set of attributes necessary to flow data
  - Complete Attribute Set for Default Schema, which represents the complete Directory Server schema

  Attribute flow is functional only for the default schema configurations. If you create a new configuration, you cannot apply it to the connector instance running in ADSpecific mode, as documented in 1b in the next section.

  If you select one of the configurations, remove a few attributes, then save the configuration, you cannot revert to the original list of attributes by clicking Insert Defaults. Clicking this button populates the list box at the bottom of the window with default mappings that you can delete or change.

- Default Values

  The connector applies preconfigured attribute rules to an entry in the external data source if no value is assigned to the same attribute in its corresponding entry in the connector view, or vice versa. A default attribute rule may also be configured.

  Default values are functional only for the default schema configurations. If you create a new configuration, you cannot apply it to the connector instance running in ADSpecific mode as documented in 1b in the next section.

- Filters

  The connector uses filtering rules to selectively exclude entries from the synchronization process.

To configure connector rules, see “Attribute Flow Rules,” on page 128, “Default Attribute Value Rules,” on page 132, and “Filter Rules,” on page 133.

Configuring a Connector Instance

Consider the following procedure an extension of the comprehensive configuration procedures in “Configuring a Universal Connector Instance,” on page 142. You need to perform the following product-specific procedure for every Active Directory Connector.

1. To automatically configure attribute flow, proceed to Step a below. To manually configure, go to Step 2.

   a. Select the connector instance for which you want to provide attributes. The General window appears, as shown in .

b. From the drop-down lists, select the desired attribute flow, filter, and default value configurations. The values that appear are derived from the rules you configured for the connector in the section “Configure Connector Rules,” on page 208.

- You can remove attributes from the complete set, if desired, before saving the configuration. The minimum configuration consists of the following attributes:

Application	Attributes

Users	`cn`
	`objectclass`
	`sn`
	`uid`

Local and Global Groups	`cn` `objectclass`

  See Table 11-2 and Table 11-3 for the complete list of external attributes.

- If you chose the default schema while creating an Active Directory Connector instance, and if you chose the minimal attribute set for attribute flow configuration, the following attributes should synchronize from Active Directory to the connector view:

  ```
  uid
  cn
  sn
  objectclass
  mail
  ```

  All 37 default attributes should synchronize from the connector view to Active Directory.

c. Select the operation you want to perform.

d. Click Save, then go to step 3.

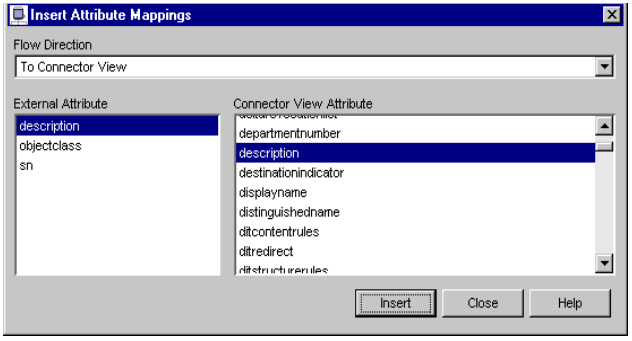
2. Optional: Manually configure the attribute flow by doing the following:

a. Select the Active Directory Connector, then select the Attribute Flow tab, as shown in .

b. Click New and enter a new configuration name, then click OK.

c. Click Insert. The Insert Attribute Mappings dialog box appears. For both mapping types (locally owned objects and connector view-owned objects), map each attribute to itself for both flow directions (to connector view and from connector view).

For example, the figure below shows the `description` attribute being mapped to itself for a flow direction to the connector view. This would also have to be repeated for a flow direction from the connector view.



d. Click Save, select View from the menu bar, then select Refresh.

e. Select the desired Active Directory Connector instance. The General window appears, as shown in .

f. From the Attribute Flow Configuration drop-down list, select the attribute flow configuration name you created (Step b). The name becomes available in the list after refreshing (Step d).

g. Select the desired filters and default values from the drop-down lists.

h. Select the operation you want to perform and click Save.

3. Configure the remaining windows for the connector instance. Begin with “To configure the schedule from and to connector views,” on page 143.

Restarting the Connector Instance

You must restart the connector instance to activate your configuration. Both instance-specific and shared configurations will not become effective for a given instance until you have restarted the instance. If the entries you are saving preexist in an Active Directory connector view, see page 205 for advisory information.

1. Stop the connector by right-clicking on the connector instance. A context menu appears.
2. Click Yes to the prompt. A message appears stating that the stop command has been issued to the component.
3. Start the connector by right-clicking on the connector instance. A context menu appears.
4. Select Start Server. A message appears stating that the start command has been issued to the component.

NOTE To start the connector, you must be a member of the Administrators group on the primary domain controller.

Implementing the Configuration

After you start the join engine and enable the connector view, your data can flow to the meta view. The following sections provide procedures for doing these tasks.

Starting the Join Engine

Before you start the join engine, ensure that you have already enabled the changelog in the Directory Server configuration.

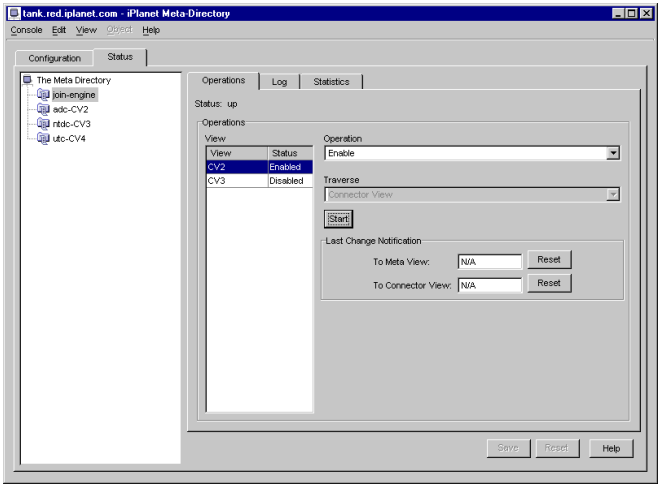
To start the join engine

1. Select the join-engine object from the navigation tree and right-click. A context menu appears.
2. Select Start Server. A message appears stating that the server has been started.

You can also start the server from the iPlanet Console. To do this, select the Join Engine object and right-click. Select Start Server from the context menu.

Enabling the Connector View

1. From the iPlanet Meta-Directory window, click on the Status tab.
2. Click on the Join Engine object. The Operations tab window appears.



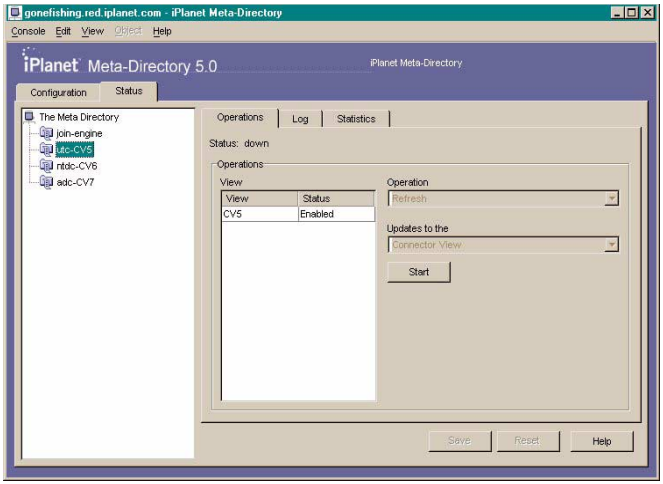
3. Select the participating view you want to enable.
4. Select Enable from the Operation list menu, then click Submit Request.

   This option disables the Traverse drop-down menu. You can only enable the participating view if the configuration for setting up the view is valid. Any error in the configuration automatically changes the view to a disable status.
5. Select Refresh from the Operation List Window, then select either Meta View or Connector View from the Traverse menu list.
6. Click Start.

Refreshing the View

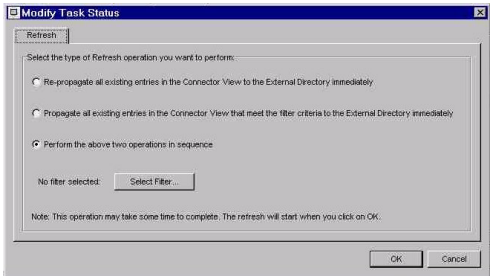
You can optionally refresh the view if you want to observe updates immediately and bypass the regularly scheduled refresh synchronization. Note that after any type of refresh, you might see a “None” group in the meta view Contents or connector view Contents, particularly with non Primary Domain Controller systems. “None” is a valid group in Windows NT.

1. From the iPlanet Meta-Directory window, click on the Status tab.
2. Click on the NT Domain connector instance object. The Operations tab window appears. The only operation available is Refresh.



3. In the “Updates to the” drop-down list, select either External Directory or Connector.
4. Click Start. The Modify Task Status dialog box appears. If you are refreshing the connector view, the following version of the box appears:

If you are refreshing the external directory, the following version of the box appears:



You must select a filter for the second and third options. Only filters configured for the "NoSubtreesExcept" option are displayed when you click Select Filter, not filters configured for the "AllSubtreesExcept" option.

Monitoring the Connector

The Active Directory Connector provides logs at the following locations that enable you to monitor connector status.

UTC Log	*InstallDir*/adc-*ViewName*/logs/meta-*date-index*.log
Accessor Utility Log	*InstallDir*/adc-*ViewName*/logs/acc-*date-index*.log
Perl Script Log	*InstallDir*/adc-*ViewName*/logs/adcpl-*date-index*.log
Task Script	*InstallDir*/adc-*ViewName*/logs/adc-*texttype*.txt

For example, a Perl log file entry might appear as follows:

```
adcpl-20010605-01.log
```

You should ignore the `churchfield.com` domain name if it appears in the UTC log.

Errors you may encounter in the Accessor Utility Log are as follows:

- 8007202A - Invalid domain or user name
- 8007203A - Cannot contact Active Directory

For other errors, refer to the following Microsoft Product Support Services site:

http://support.microsoft.com/support/kb/articles/Q242/0/76.asp

Data Flow for User and Group Entries

Entries in the Active Directory connector view must adhere to certain conditions to flow from the connector view into the Active Directory. Note the following restrictions and advisory information:

- You should use the default in ADSpecific mode.
- To prevent duplicate user IDs from occurring in the same connector view, the meta view connector views must be separate entities. A connector view should not be nested as a subtree of another connector view. That is, the connector view should be a flat tree that does not contain any subentries.
- Entries that preexist in an Active Directory connector view will not flow to the meta view after the connector starts. To flow these entries, the Active Directory connector view must be an enabled participating connector view in the join engine. Refreshing the meta view operation from the join engine will trigger the preexisting entries from the Active Directory connector view to flow to the meta view.
- If you create users or groups from the console with the default objectclasses in ADSpecific mode, the users or groups propagate to Active Directory. If you synch these entries back to a different connector view, the entries are created on the new connector view with mdsADUser or mdsADGroup objectclasses. Sphere icons appear, rather than the standard person or group icons.

When setting up the join engine, you need to ensure that user and group entries meet the required criteria for Active Directory Connector views. The following sections discuss the requirements and list the available external attributes read from Active Directory for both user and group entries.

User Entries

You can create Active Directory users in the connector view with any LDAP client by adhering to the attribute conventions shown in the following structure for the default schema:

```
dn: uid=userid, cvroot_dn
uid: userid
cn: user_full_name
objectclass: top
```

```
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
sn: user_second_name
```

For the ADSpecific schema, the structure would be as follows:

```
dn: uid=userid, cvroot_dn
uid: userid
cn: user_full_name
objectclass: mdsADtop
objectclass: mdsADperson
objectclass: mdsADorganizationalPerson
objectclass: mdsADinetOrgPerson
sn: user_last_name
```

Note the following restrictions:

- The *userid* cannot contain the following characters:

  ```
  " / \ [ ] : ; | = , + - * ? < >
  ```

- The user name cannot consist solely of periods or spaces.

Table 11-2 shows the available external attributes for user entries in default schema mode. Refer to your Active Directory documentation for more information about these attributes.

Table 11-2 Attributes for User Entries

department	homephone
description	telephonenumber
facsimiletelephonenumber	l
homepostaladdress	thumbnailphoto
o	mobile
ou	usercertificate
objectclass	physicaldeliveryofficename
pager	cn
postalcode	mail
postofficebox	street
samaccountname	postaladdress
sn	distinguishedname

Table 11-2 Attributes for User Entries

st	givenname
usermimecertificate	title

Group Entries

The group entries in the connector view contain the list of member DNs. The connector view applies static group membership.

You can create NT groups in the connector view with any LDAP client by adhering to the attribute conventions shown in the following structure for the default schema:

```
dn: cn=groupname, cvroot_dn
objectclass: top
objectclass: groupOfNames
```

For the ADSpecific schema, the structure would be as follows:

```
dn: cn=groupname, cvroot_dn
objectclass: mdsADtop
objectclass: mdsADgroupOfNames
```

Table 11-3 shows the available external attributes for group entries. Refer to your Active Directory documentation for more information about these attributes.

Table 11-3 Attributes for Group Entries

cn	member
description	objectclass
distinguishedname	samaccountname
grouptype	

Configuration Example

The following ADSpecific schema example is intended as a quick reference you can use as a checklist. For complete configuration information, refer back to the earlier portions of this chapter.

Install the Connector

1. Ensure that iPlanet Directory Server 4.*x* or 5.0, and iPlanet Meta-Directory 5.0 are already installed.
2. Install the ADSI package.
3. Create a connector view instance.

   During instance creation:

   a. From the iPlanet Console 5.0 window, right-click on Server Group. A context menu appears.

   b. Select Create Instance Of, then select Meta-Directory Active Directory Connector. The New Instance Creation dialog box appears.

   c. Provide input for the data fields. For View Name, use `Active`. For View ID, use `CV1`. For View Base DN, use `o=CV1`. For Schema, use `default`. For the remaining fields, see Table 11-1 on page 193.

   From the configuration file:

   a. Locate the `adc.ini` configuration file in the following directory:

      NetsiteRoot`/adc-`*ViewName*`/config/adc.ini`

   b. Provide values for the file parameters. Use default parameters and values.
4. Add the instance as a participating view.

   a. Right-click the Participating Views object. A context menu appears.

   b. Select Add Participating View. The Select View dialog box appears.

   c. Select `Active` and click OK. The view is added to the iPlanet Meta-Directory tree.
5. Provide authorization. See “Setting Access Permissions,” on page 302.

Configure Connector Rules

1. Configure attribute flow.

   a. Click on the Active Directory connector. The Attribute Flow tab window appears.

b. Select the Minimal Attribute Set for Default Schema from the list of configurations. Note that the `manager` attribute does not appear in any of the three configuration choices. The Active Directory checks this attribute for referential integrity, and an arbitrary value causes the Active Directory connector to fail.

c. In the Mapping Type drop-down list, select Mappings for Connector View Owned objects.

d. Click Insert. The Insert Attribute Mappings dialog box appears. This displays a list of all available attributes from both the external data source and the Connector View.

e. For Mapping Type, select Mapping for Connector View Owned objects. For Flow Direction, select From Connector View. For Connector View Objectclass, select All Attributes.

f. For External Attribute, select `homephone`. For Connector View Attribute, select `telephonenumber`.

g. Click Insert. The mapping for your configuration appears at the bottom of the Attribute Flow window.

h. Click Close, and then click Save from the Attribute Flow window.

2. Configure default attribute rules.

a. Click on the Default Values tab. The Default Values window appears.

b. Click New.

c. In the Name field, type in `ActiveDefault`. The name is echoed in the Configurations list box.

d. In the Attribute Destination drop-down list, select External Directory.

e. Click Add. Blank fields appear below the Attribute and Default Value fields.

f. Click within the blank Attribute field. A drop-down list appears. Select `givenname` from the list.

g. Double-click within the blank Default Value field and type in `surname`.

h. Click Save.

3. Configure filters.

a. Click on the Filters tab. The Filters window appears.

b. Click New. The Filter Name dialog box appears.

c. Type in `ActiveExclude` and click OK. The new name appears in the Filter Name list box.

d. Select From Connector View.

e. Filter excluded data:

I. Provide a list of subtrees to exclude by selecting All Subtrees Except, then clicking Add. The Sub-tree DN dialog box appears.

II. Specify a subtree to exclude, such as `o=siroe,c=us`, then click OK. The subtree appears in the list box.

With this filter, entries in all subtrees that are not specifically excluded are included, no matter how you set the associated entry-level filters.

III. Filter back entries from the excluded subtrees using entry-level filters. Select the subtree you just created, select All Entries Except, then click Add. The Entry RDN dialog box appears.

IV. Specify an entry you want to include, such as `cn=Fred Scofflaw`, then click OK. The included entry appears in the list box.

The entry-level filters you apply affect only the entries found in the list of subtrees to include. The entries you specify here will filter through; all others are excluded.

f. Click Save.

g. From the menubar, select View > Refresh.

Configure a Connector Instance

1. Select the adc-Active connector instance. The General window appears.
2. Select the following from the drop-down lists:
   - For Configuration, select Minimal Attribute Set for Default Schema.
   - For Filter Configuration, select ActiveExclude.
   - For Default Values Configuration, select ActiveDefault.
3. For Operation, select “Only receive updates from the Connector View.”
4. Click Save. Leave the current values for fields in the Schedule, Log, and Attributes windows.

Restart the Connector Instance

1. Stop the connector by right-clicking on adc-Active. A context menu appears.
2. Click Yes to the prompt. A message appears stating that the stop command has been issued to the component.
3. Start the connector by right-clicking on adc-Active. A context menu appears.
4. Select Start Server. A message appears stating that the start command has been issued to the component.

Chapter 12

Configuring the Database Connector

This chapter discusses configuration factors specific to the Database Connector, which provides bi-directional synchronization between Oracle and the meta view.

The topics in this chapter are:

- Before You Begin
- Creating the Data Server
- Configuring a Connector View
- Configuration Example
- Changing Instrumentation
- Issuing Privileges

Before You Begin

Before continuing, make sure you have done the following:

- Install OCI 8.1.5.
- Create a connect string for the database.
- Create an instance of the join engine.

There are two phases in configuring the database connector:

- Creating the data server
- Configuring the connector view

The following sections provide procedures for these phases, as well as a configuration example. You should perform the procedures in the order presented.

Creating the Data Server

Creating the data server consists of the following tasks:

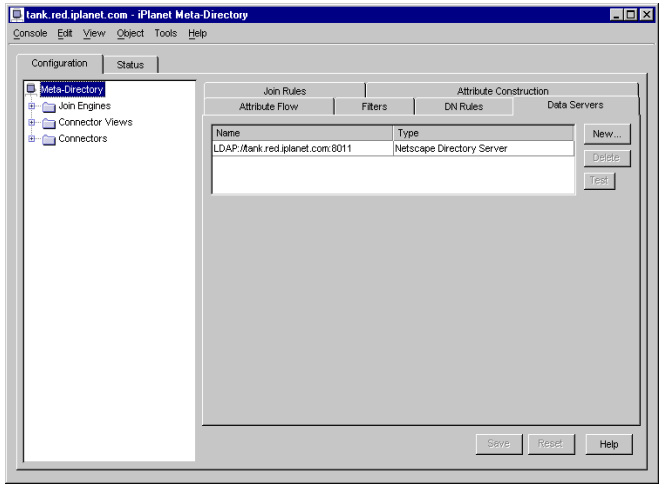
- Configuring the data server
- Running the configuration script
- Adding a connector view instance
- Adding the instance as a participating view

Configuring the Data Server

To invoke the Data Servers window

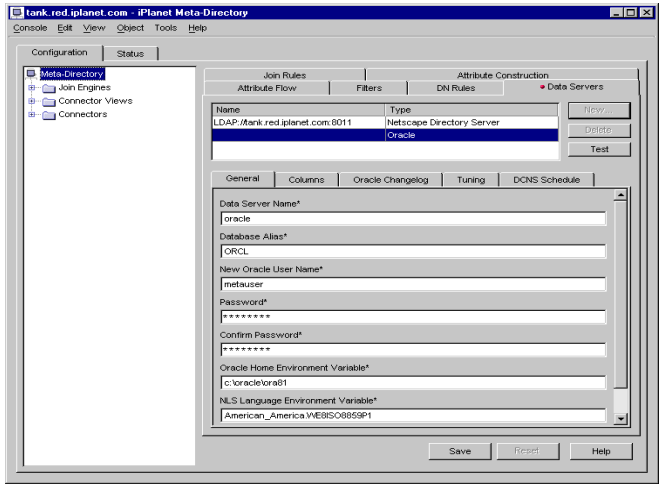
1. From the iPlanet console, double-click Server Group.
2. Double-click the Join Engine object. The iPlanet Meta-Directory window appears.
3. Click the Meta-Directory object. The Join Rules window appears within the Meta-Directory Configuration tab.

4. Click on the Data Servers tab. The Data Servers window appears.



To add a new database server

1. Click New. The Select a Data Server Type dialog box appears.
2. Select Oracle and click OK. The server type appears in the list box on the Data Servers window. Several tabs now appear at the bottom of the window that enable you to name and configure the new data server.

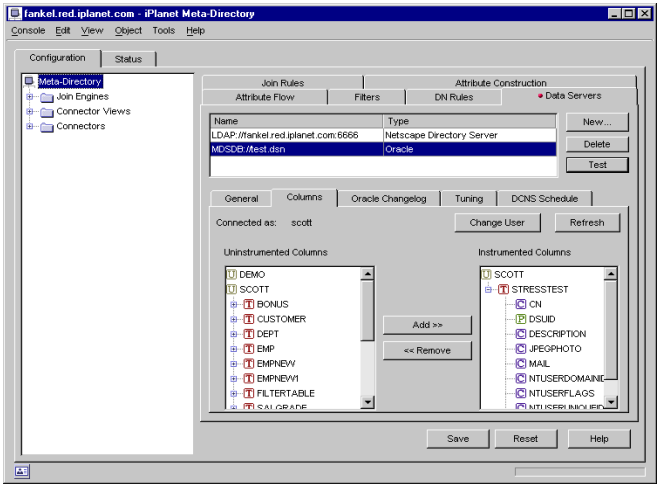


To provide data server information

1. Select the server to be configured.
2. From the General tab, provide information or change the defaults for the following fields:

Data Server Name	Enter any legitimate string without spaces. This is a required field.
Database Alias	Enter a connection string to connect to the database. For Oracle, you typically create the alias by using the Net8 Easy Config utility in Oracle 8. This is a required field.
New Oracle User Name	Enter a user name that does not preexist in the Oracle database. After data server creation, a script is generated, which when executed, adds this user to the Oracle database. The join engine employs this user's credentials to bind to Oracle to retrieve the data. This is a required field.
Password	Enter a password associated with the user name.
Confirm Password	Enter the same password as above.
Oracle Home Environment Variable	Enter the ORACLE_HOME path of this machine. The OCI client libraries must be installed on the machine where you have installed the join engine database connector. This is a required field.
NLS Language Environment Variable	Enter the NLS_LANG environment variable used to indicate the locale, territory, and character set to the Oracle Call Interface (OCI) client library.
Description	Enter a brief description to inform other users about the purpose of this data server. This is an optional field.

3. Click the Columns tab. The Database Credentials dialog box appears. After providing the Oracle user name and password, described in “Script Generation,” on page 237, the Columns window appears. The credentials remain in effect until you leave this particular data server. Note that this is not the same user described in New Oracle User Name above.



You can select columns or tables to participate in the flow to the meta view. This process of selection and setting up triggers is called *instrumentation*. The triggers populate the changelog tables with changes.

4. Instrument by doing the following:

   a. Select a column, table, or user from the Uninstrumented Columns list.

   The icon C precedes column names, T precedes table names, and U precedes user names. An X icon precedes reserved columns that cannot be instrumented.

   Selecting a table instruments all the columns that belong to the table, and selecting a user instruments all the tables that belong to the user.

   b. Click Add, or double click on the node. Your selection now appears in the Instrumented Columns list.

   One of the columns within the table you select for instrumentation must be a primary key, denoted with the letter P. If you have selected a table or column from a table for which there is no primary key, a Nominate Primary Keys dialog box appears.

I. Select one or more columns as the nominated key(s). As a shortcut, you can double click an un-nominated column to move it to the nominated column.

   The set of nominated columns for a table must be unique for the associated row. That is, if you select FirstName and LastName as your nominated keys, there may be more than one person with the same name. A better choice would be AreaCode and PhoneNumber.

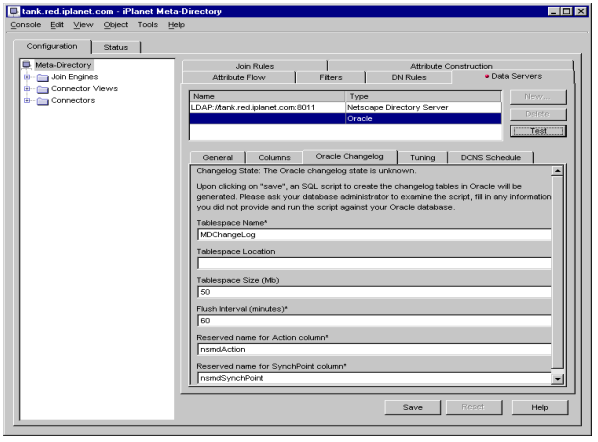
II. Click Add. The column or set of columns now appear in the list of nominated keys.

III. Click OK to close the dialog box and return to the Columns window. Note that an N icon now precedes the column you have nominated.

   You can change any nominated key after clicking OK as long as the associated table isn't currently instrumented, and there are no primary keys in the table. To do so, right click on any node and select "Change nominated keys."

5. Optional: If you want to change the user credentials, click Change User, then provide a new Oracle user name and password. Meta-Directory rereads the database dictionary and rebuilds the schema. You need to change credentials whenever you instrument users, tables, or columns that a different user owns.

6. Click the Oracle Changelog tab to display the Changelog window. You can provide input only for a new data server. For existing data servers, you can only change the Flush Interval field.



Provide information or change the defaults for the following fields:

Tablespace Name

Enter the name of the tablespace used for changelog tables, triggers, and changes. The default is `MDChangeLog`. This is a required field.

See your database administrator for information about the tablespace name.

Tablespace Location

Enter the path and name of the data file, such as:

```
d:/oracle/changelogs/changelog.dat
```

See your database administrator for information about the tablespace location.

Although this is not a required field, if you leave it blank, your database administrator will need to edit the generated script and supply this information.

Tablespace Size

Enter the size of the table space. See your database administrator for information about the tablespace size.

Although this is not a required field, if you leave it blank, your database administrator will need to edit the generated script and supply this information.

Flush Interval

Enter a time interval in minutes when the server is requested to flush (trim) its change log. This is a required field.

Reserved Name for Action Column

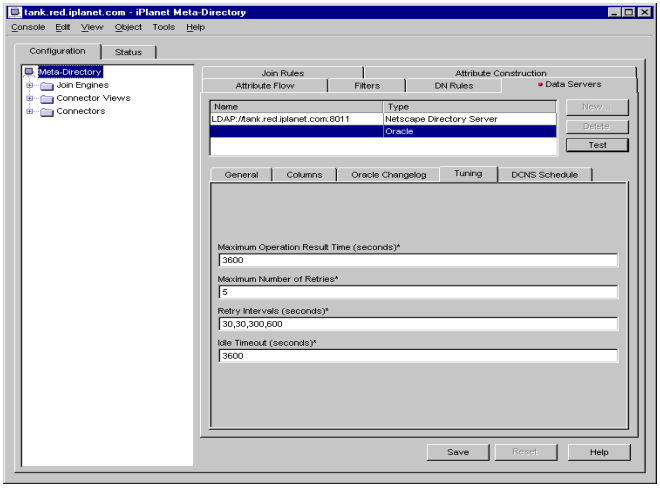
Enter a reserved column name that is not assigned to any column. The name must be unique for all columns and tables you want to instrument. The default is `nsmdAction`. This is a required field.

Note that you cannot change the name after defining it.

Reserved Name for SynchPoint Column

Enter a reserved column name that is not assigned to any other instrumented column. The name must be unique for all columns and tables you want to instrument. The default is `nsmdSynchPoint`. This is a required field.

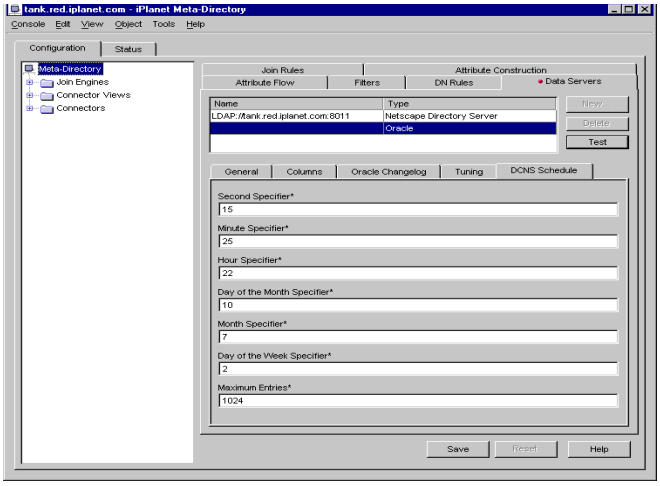
7. Click the Tuning tab to display the Tuning window.



Provide information or change the defaults for the following fields:

Maximum Operation Result Time	Enter the maximum amount of time allowed before time-out for an SQL search to receive its results. The suggested minimum value is 3600.
Maximum Number of Retries	Enter the number of times you want the server to attempt to connect after it initially fails to connect. This field is associated with the Retry Intervals field.
Retry Intervals	Enter the number of seconds to wait before the next retry should begin. There should be one interval for each retry. For example, if you specify 10 retries in the field above, you need 10 retry intervals. The syntax is shown in the window example above.
Idle Timeout	Enter the time, in seconds, that should pass before retries are abandoned if the server is idle.

8. Click the DCNS Schedule tab to display the DCNS Schedule window.



Provide values for the following fields:

Second Specifier	Enter a value from 0 to 59.
Minute Specifier	Enter a value from 0 to 59.
Hour Specifier	Enter a value from 0 to 23.
Day of the Month Specifier	Enter a value from 1 to 31.
Month Specifier	Enter a value from 1 to 12.
Day of the Week Specifier	Enter a value from 0 to 6, where 0 is Sunday and 6 is Saturday.
Maximum Entries	Enter the maximum number of records to be read from the changelog on each poll cycle.

You can use either a single number or an expression as described below.

Expression	Definition
*	Matches any value.
*/step	Matches any value in steps. For example, */2 matches 0,2,4,6 ... up to the maximum allowed value for values that start with zero, or it matches 1,3,5,7 ... up to the maximum value allowed.
x-y	Specifies a range where: • Both x and y are greater than or equal to the minimum allowed value. • y is less than or equal to the maximum allowed value. • x is less than y. The expression matches any value in the range.
x-y/step	Specifies a range as above, but with a step value that is not necessarily 1.
x	Specifies a single number within the allowed range.
x/step	Matches any value starting at x and then at x + step, x + 2*step, and so forth.
x,y,z...	Specifies a comma-separated list of values.
x-y,z-q	Specifies a comma-separated list of ranges.

Lists can also contain both single values and ranges such as 1,2,5-7,10/5.

The scheduler operates once every second, so the finest granularity occurs every second.

9. Click Save to save the configuration. This generates a script that must be applied to the database. Privileges are required for generating and executing scripts. See “Issuing Privileges,” on page 236 for more information.

10. Optional: Test the connection by doing the following:

    a. Execute `inst_orcl.sql`.

    b. Select the new server, then click Test.

    A Test Connect Succeeded message appears if the connection was successful.

To delete a data server

1. Select the data server you want to delete.
2. Click Delete. The data server and its associated configuration disappear from the list box. A message appears that states where the script is stored.
3. Ask your database administrator to execute the script to completely uninstrument the tables. When applied, the script deletes the changelog schema, changelog tablespace, and all triggers from the instrumented tables.

NOTE You should not uninstrument an Oracle data server if a user is connected. If the join engine is running when you try to uninstrument your Oracle data server, the SQL scripts will fail, stating that the user could not be removed.

Running the Configuration Script

1. You or your database administrator need to run the following script:

   `inst_`*`dataservername`*`.sql`

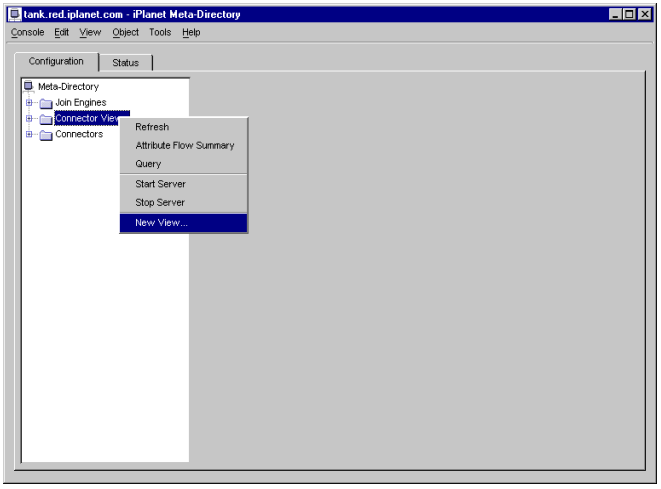
   The script is stored in the `$NETSITE_ROOT` directory.
2. Ask your database administrator to check the script and replace any tokens with information (that is, `$TABLE_SPACE_LOCATION$`). The administrator should then run the script as the system database administrator. After the script has been generated, the administrator should confirm that all fields are present and correct.

   After running the script, the table(s) become instrumented and the changelog tablespace is created.

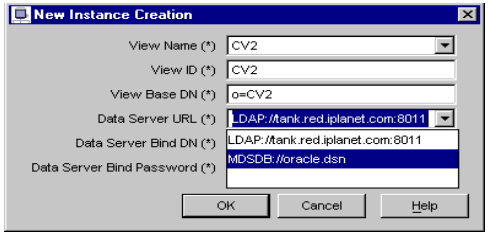
Adding a Connector View Instance

To add the instance

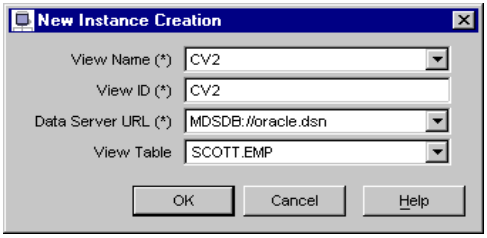
1. Select Connector Views from the Meta-Directory tree, then right-click. A context menu appears.



2. Select New View. The New Instance Creation dialog box appears.



3. Select the newly created data server URL from the drop-down list. An altered New Instance Creation dialog box appears.



Provide information for the following fields:

View Name	Enter a name that more fully describes the View ID. The default is the View ID.
View ID	Enter up to five characters to represent the view ID. The default is CVn, where n is the next available view number.
View Table	Select the table you want to assign to the connector view.

4. Click OK. The message “Instance Creation Succeeded” appears after the instance has been created. The main console navigation tree will redisplay.

To delete the instance

1. Highlight the instance you want to delete, then right-click. A context menu appears.
2. Select Delete View. The instance is deleted.

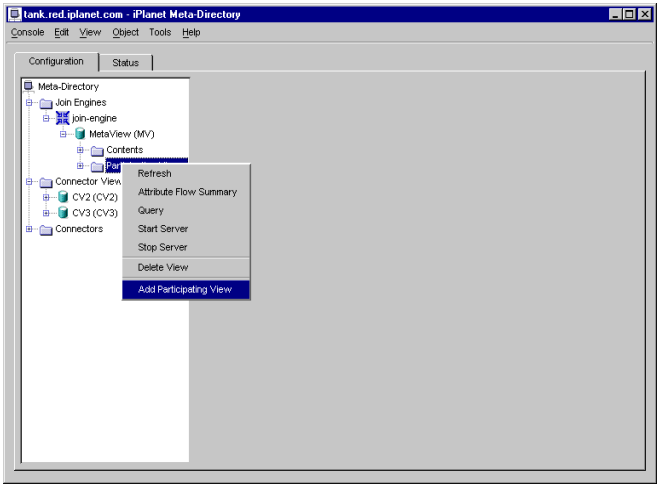
The Oracle connector view is also deleted if you delete the Oracle data server.

Adding the Instance as a Participating View

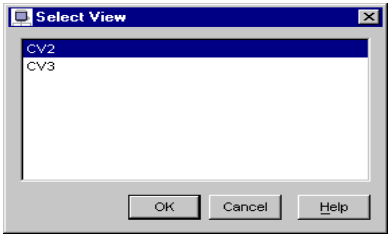
Each participating connector view has its own settings. In addition to these settings (or in place of them), each view also has settings that specify how the join engine handles the view within the meta view.

To add a participating connector view

1. Right-click the Participating Views object. A context menu appears.

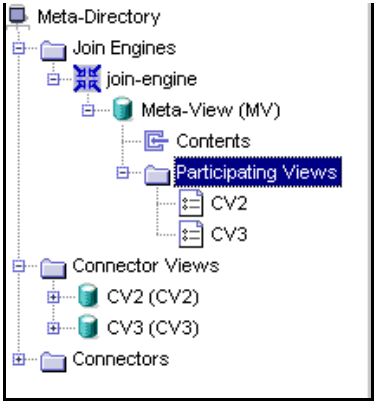


2. Select Add Participating View. The Select View dialog box appears.



3. Select the connector view you want to add or participate in a join/synchronization with the meta view.

4. Click OK. The view is added to the iPlanet Meta-Directory tree as shown below.



To remove a participating connector view

1. Highlight the view you want to remove, then right-click. A context menu appears.
2. Select Delete View. The view is deleted.

Configuring a Connector View

Configuring the connector view consists of the following tasks:

- Creating constructed attribute rules
- Creating attribute flow rules
- Creating DN Mapping rules
- Configuring the participating connector view
- Starting the join engine
- Enabling the connector view

You can create join rules and filters, but you must minimally create the following types of rules:

- Constructed attribute
- Attribute flow

- DN mapping

After you create the rules, you can add them to the connector view and flow data. The data should flow according to these rules from the connector view (Oracle) to the meta view and vice versa.

Creating Constructed Attribute Rules

See “Constructed Attributes,” on page 81 for procedures on creating constructed attribute rules.

Creating Attribute Flow Rules

See “Attribute Flow Rules,” on page 85 for procedures on creating attribute flow rules.

Creating DN Mapping Rules

You need to carefully write the DN mapping rules for flowing data from the meta view to the connector view. You might expect these to look like rules for flowing data from the connector view to the meta view with elements of the rule reversed, but this may not be the case.

The meta view to connector view DN mapping rules typically look like the following example:

```
<PK col name>=%mv.<mapped attribute>
```

Consider an employee table EMP with FIRSTNAME, LASTNAME, EMPNO and DEPT as columns in which EMPNO is the primary key. The connector view to meta view DN mapping rules for an `inetorgperson` entry would typically look like:

```
cn=%cv.FIRSTNAME%
```

However, the meta view to connector view DN mapping rule would need to be:

```
EMPNO=%mv.employeenumber%
```

assuming that you have mapped `employeenumber` to the EMPNO column.

Therefore, you need to map primary key columns to an attribute in the directory in order to correctly map a row in the database to a single entry in the directory and vice versa.

See “Distinguished Name Mapping Rules,” on page 97 for procedures on creating DN mapping rules.

Configuring a Participating Connector View

See “Configuring a Participating View,” on page 111 for procedures on configuring a participating connector view. Note that the Group Filters window is only available for UTC-based connectors or for iPlanet Professional Services.

Starting the Join Engine

Before you start the join engine, ensure that you have already enabled the change log in the Directory Server configuration.

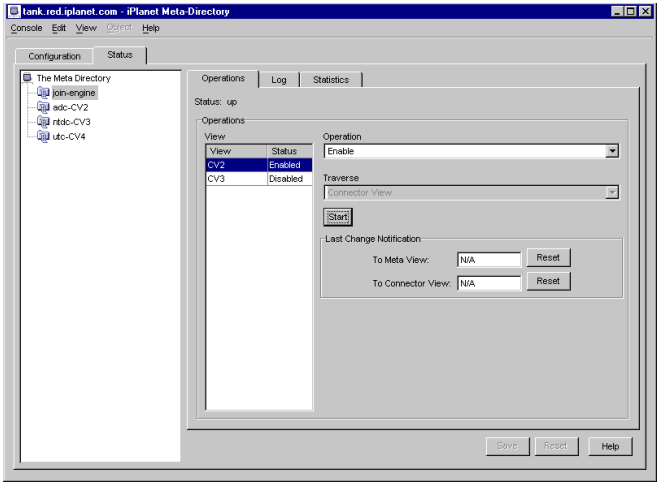
To start the join engine

1. Select the join-engine object from the navigation tree and right-click. A context menu appears.
2. Select Start Server. A message appears stating that the server has been started.

You can also start the server from the iPlanet Console. To do this, select the Join Engine object and right-click. Select Start Server from the context menu.

Enabling the Connector View

1. From the iPlanet Meta-Directory window, click on the Status tab.
2. Click on the Join Engine object. The Operations tab window appears.



3. Select the participating view you want to enable.
4. Select Enable from the Operation list menu, then click Start.

   This option disables the Traverse drop-down menu. You can only enable the participating view if the configuration for setting up the view is valid. Any error in the configuration automatically changes the view to a disable status.
5. Select Refresh from the Operation List Window, then select either Meta View or Connector View from the Traverse menu list.
6. Click Submit Request. The status changes to up.

If you are having trouble enabling the connector view, you should check whether the proper privileges have been issued. See “Issuing Privileges,” on page 236 for more information.

Monitoring the Connector View

You should monitor the connector view status while restarting the database on which you created the connector view. When the Oracle server is restarted while the Oracle connector view is enabled, the join engine requires about one to two minutes for rebinding to the Oracle server. You can adjust the DataAccess logs to level 1 to determine that the rebind has occurred; the logs would show SQLs being executed on the Oracle changelog. Subsequent to the rebind, all refresh / changelog operations would become effective.

It takes about a minute for Oracle to shut down. You can expect the join engine to disable the Oracle connector view if you attempt an operation, such as add, on the database during the database shutdown. In that case, you would need to explicitly enable the connector view after the database is up and running.

Configuration Example

The following example is intended as a quick reference you can use as a checklist. For complete configuration information, refer back to the earlier portions of this chapter.

Create the Data Server

1. Configure the data server.
   a. Click the Meta-Directory object, then click on the Data Servers tab. The Data Servers window appears.
   b. Click New. The Data Server Type dialog box appears.
   c. Select Oracle and click OK.
   d. From the General tab, provide information for the fields.
      - The New Oracle User Name should not currently exist in the Oracle database.
      - The Oracle Home Environment Variable is the OCI root directory.
      - Use the default for the NLS Language Environment Variable.
   e. Click the Columns tab. The Database Credentials dialog box appears. After providing the Oracle user name and password, the Columns window appears.
      - Select the columns you want to synchronize.
      - Select the primary key of the column to be instrumented. If no primary key exists in the column, you need to nominate a primary key.
   f. For the remaining tabs, use the defaults.
   g. Click Save to save the configuration.
2. Run the following configuration script:

`inst_`*`dataservername`*`.sql`

The script is stored in the `$server_root\bin\meta50\bin` directory. This script grants privileges. For more information on privileges granted, see “Issuing Privileges,” on page 236.

3. Add the connector view instance.
   a. Select Connector Views from the Meta-Directory tree, then right-click. A context menu appears.
   b. Select New View. The New Instance Creation dialog box appears.
   c. Select the newly created data server URL from the drop-down list. An altered New Instance Creation dialog box appears.
   d. Provide the view name, view ID, and select the table you want to synchronize.
   e. Click OK. The message “Instance Creation Succeeded” appears after the instance has been created. The main console navigation tree will redisplay.
4. Add the instance as a participating view.
   a. Right-click the Participating Views object. A context menu appears.
   b. Select Add Participating View. The Select View dialog box appears.
   c. Select the newly created connector view and click OK. The view is added to the iPlanet Meta-Directory tree.

Configure the Participating Connector View

1. Create constructed attribute rules.
   a. Select Configuration > Meta-Directory > Attribute Construction.
   b. Click New Attribute, then provide a name, such as `dbobjectclass`, in the New Constructed Attribute dialog box.
   c. Click OK. The new attribute appears in the Attributes list.
   d. Select the attribute, then click New Rule. The New Constructed Attribute Rule dialog box appears.
   e. Configure the rule, using `dbobjectclass` as the name. For the Attribute Construction field, enter the following string:

      `top;person;organizationalperson;inetorgperson`

f. Click OK, then click Save from the Attribute Construction tab.

2. Create attribute flow rules.

a. Select Configuration > Meta-Directory > Attribute Flow.

b. Click New Rule. The New Attribute Flow Configuration dialog box appears.

I. Provide a name of `ora2mvattrflowrule`.

II. For Direction, select To Meta View.

III. For Selection Criteria, click "..." to go to the Compose Condition Criteria dialog box. For Type, select Attribute. For Source, select the Oracle connector view. For Attribute, select the primary key or nominated key. For Expression, select Present. Click Insert, then OK.

c. Click Add. The Add Attribute Mappings dialog box appears.

d. Map attributes by doing the following:

I. For Source View, select the Oracle connector view.

II. For Source Objectclass, select the Oracle table.

III. For Destination View, select Meta View.

IV. For Destination Objectclass, select `inetorgperson`.

V. Select one attribute from the left column, one from the right column, then click Insert. You do not need to map all attributes.

Repeat the process with these variations:

I. For Source Objectclass, select Constructed Attributes.

II. For Destination Objectclass, select `inetorgperson`.

III. Select the constructed attribute (`dbobjectclass`) and map it to `objectclass` in the right column.

IV. Click Insert, then Close.

e. Click OK from the New Attribute Flow Configuration dialog box. The new rule appears in the Rules section of the Attribute Flow window.

f. From the Attribute Flow window, click New Set, and provide a name of `ora2mvattrflowruleset` in the New Set dialog box.

g. Select `ora2mvattrflowrule` and `ora2mvattrflowruleset`, then click Add Member. The rule appears in the Members list box.

h. Click Save, then select View > Refresh from the menubar.

3. Create DN mapping rules.

a. Select Configuration > Meta-Directory > DN Rules.

b. Click New Rule. The New DN Mapping Rule dialog box appears.

I. Provide a name of `ora2mvdnmap`.

II. For Selection Criteria, click "..." to go to the Compose Condition Criteria dialog box. For Type, select Attribute. For Source, select the Oracle connector view. For Attribute, select the column mapped to the RDN. For Expression, select Present. Click Insert, then OK.

III. For Distinguished Name Construction, enter the RDN:

```
cn=%cv.ENAME%
```

where `ENAME` is a column name in the connector view (primary key or unique column). Do not repeat the RDN mapping in the attribute flow rule.

c. Click OK. The new rule for this DN mapping appears in the Rules list box.

d. Click New Set and provide a name of `ora2mvdnmapset` in the New Set dialog box.

e. Select `ora2mvdnmap` and `ora2mvdnmapset`, then click Add Member. The rule appears in the Members list box.

f. Click Save, then select View > Refresh from the menubar.

4. Configure the participating connector view.

a. Select the participating Oracle connector view for the meta view. The Configuration tab appears.

b. For Attribute Flow, select None for To Connector, and select `ora2mvattrflowruleset` for To Meta View.

c. For Join Rules, select None for both drop-down lists.

d. For DN Mapping Rules, select None for To Connector, and select `ora2mvdnmapset` for To Meta View.

e. For the remaining tabs, see "Configuring a Participating View," on page 111.

f. Click Save.

5. Start the join engine.

   a. Select the join-engine object from the navigation tree and right-click. A context menu appears.

   b. Select Start Server. A message appears stating that the server has been started.

6. Enable the connector view.

   a. Select Status > join-engine > Operations.

   b. For View, select the Oracle connector view, for Operation, select Enable, and then click Start.

   c. Repeat the step above, except select Refresh instead of Enable.

Changing Instrumentation

You can add or remove instrumentation any time after you have followed all of the procedures in this chapter to this point in the order presented.

To add instrumentation

1. Click the Columns tab. The Database Credentials dialog box appears. After providing the Oracle user name and password, the Columns window appears. The credentials remain in effect until you leave this particular data server.
2. Select one or more columns, tables, or users from the Uninstrumented Columns list.
3. Click Add. If you selected a column, the column now appears in the Instrumented Columns list. If you selected a table, the table and its associated column(s) now appear in the Instrumented Columns list. If you selected a user, the user and its associated table(s) now appear in the Instrumented Columns list.
4. Click Save.
5. Run the script. The changes are now instrumented.

To remove instrumentation

1. Click the Columns tab. The Database Credentials dialog box appears. After providing the Oracle user name and password, the Columns window appears. The credentials remain in effect until you leave this particular data server.
2. Select one or more columns, tables, or users from the Instrumented Columns list. Note that if you select a primary key column or nominated column, the entire table will be uninstrumented.
3. Click Remove.
    - If you selected a column, the column now appears in the Uninstrumented Columns list.
    - If you selected a table, the table and its associated column(s) now appear in the Uninstrumented Columns list.
    - If you selected a user, the user and its associated table(s) now appear in the Uninstrumented Columns list.
4. Click Save. After running the script, the uninstrumented columns, tables, or users are removed and the remaining columns, tables, and users become instrumented.

Issuing Privileges

Oracle privileges are required for performing the following tasks:

- Script generation
- Script execution
- Synchronization of instrumented tables

Typically, different individuals would perform each of these tasks, but one person could perform all of the tasks if preferred. The following sections explain these tasks and the associated privileges required. Contact your database administrator or refer to your Oracle server documentation set for specific questions about privileges.

Script Generation

The console reads the database dictionary to gather information about the database schema. The user can select tables and columns from this schema for instrumentation.

The user for which the console connects to Oracle must be given sufficient privileges to read the database dictionary tables. The associated user name and password are supplied to the Meta-Directory Console, which the console uses to read the data dictionary. The user name and password are not stored in Meta-Directory configuration.

Script Execution

To instrument the database, a user executes the script and has privileges to do the following:

- Create and delete the changelog user, changelog tables, and tablespaces
- Grant the changelog user privileges to insert, update, and delete rows from instrumented source tables
- Install and uninstall triggers associated with the instrumented tables

This user name and password are supplied to the tool that executes the script; they are not stored in Meta-Directory configuration, nor handled by Meta-Directory software.

Table 12-1 provides reference information for the user roles discussed above.

Table 12-1 Database Privileges

Task	Required Privileges	Appropriate User
Script Generation	CREATE SESSION To instrument tables that other users own, SELECT privileges for these tables should be granted to this user before attempting to instrument them.	Any existing database user with these privileges. (Known as the console user.)

Table 12-1 Database Privileges *(Continued)*

Task	Required Privileges	Appropriate User
Script Execution	CREATE ANY TRIGGER DROP ANY TRIGGER CREATE TABLESPACE DROP TABLESPACE CREATE USER DROP USER CREATE ROLLBACK SEGMENT DROP ROLLBACK SEGMENT CREATE ANY TABLE DROP ANY TABLE INSERT ANY TABLE DELETE ANY TABLE CREATE SESSION CREATE ANY PROCEDURE EXECUTE ANY PROCEDURE	Any user with these privileges.
Synchronization	CREATE SESSION SELECT / INSERT / UPDATE / DELETE Also, privileges are required to synchronize all instrumented source tables. These privileges are granted to the changelog user by the script Meta-Directory generates.	The changelog user the Meta-Directory creates.

Synchronization of Instrumented Tables

The Meta-Directory console generates a script to instrument the tables and columns you select. The set of tables you can select for instrumentation consists of the tables you own, and those for which you have SELECT privileges. The database schema is built by querying the data dictionary, which is accessed through public synonyms (defined in data dictionary views).

For synchronization, each data server has one user (whose user name and password are in its mdsAuthenticationDetails attribute). The Meta-Directory Database Connector authenticates this user to synchronize any associated connector view. The generated script grants privileges to the changelog user to:

- Read the changelog
- Delete entries from the changelog that have already been processed
- Insert, update, and delete rows from instrumented source tables

Part 4

System Management

Starting and Stopping Components

The process of starting, stopping and restarting Meta-Directory components is key to refreshing data entries and keeping records current. This process is handled by a Perl script that creates a file named `start.conf`. This chapter contains the following sections:

- The start.conf File
- Starting Meta-Directory Components
- Stopping Meta-Directory Components
- Checking a Component's Operational Status
- Automated Restarts

The start.conf File

The `start.conf` file is a text file; there is one `start.conf` file for each Meta-Directory component. The files can be found at:

NETSITE_ROOT`/`*component_directory*`/config/start.conf`

where *component_directory* is a variable to be replaced by one of the following component name placeholders: join-engine, adc, ntdc, or utc.

At startup, the objective of a Meta-Directory component is to connect to the data server as well as validate the distinguished name and password (referred to as `authenticationDetails`) needed to open access to it. Initially, the `start.conf` file contains only the URL of the server. At the implementation of `Start Server` for the specified component, the distinguished name and password are retrieved from the Administration Server and written to the `start.conf` file. With this information, the component is able to bind to the data server specified as well as read and confirm the authentication details thus allowing itself to start.

Shortly after startup, the component reopens the `start.conf` file and erases the distinguished name and password so that, once again, it contains only the URL of the data server. (Specifically, the password is overwritten with white space characters and truncated to zero length.) This process maximizes security.

NOTE If the component is unable to erase the authentication details from `start.conf`, a warning message will be logged although the component will continue to function properly.

Starting Meta-Directory Components

On Windows NT systems, you can start Meta-Directory components using the Console, the Meta-Admin command-line tool or the Windows NT Service Control Manager. On Solaris systems, you can start Meta-Directory components using the Console or the Meta-Admin command-line tool. If you are unable to start any of the components, log files recorded to each component's directory can be viewed at *NETSITE_ROOT*`/`*component_directory*`/logs/` to determine the problem.

Starting Components Via the Consoles

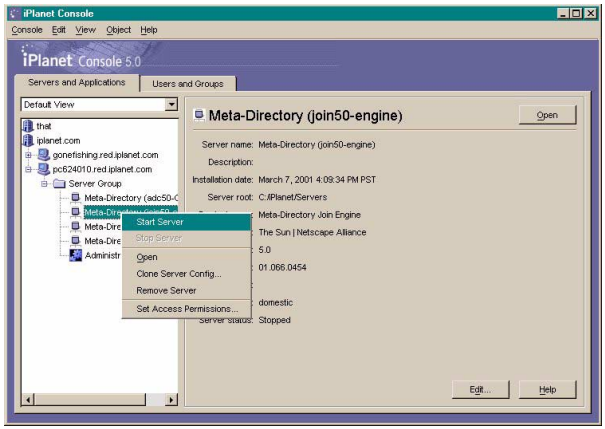
You can start the join engine or connectors on a Windows NT or Solaris system by using either the iPlanet Console or Meta-Directory console.

To Start From the iPlanet Console Navigation Tree

1. Open iPlanet Console by clicking `Start` and choosing Programs > iPlanet Server Products > iPlanet Console.
2. Open the branches of the navigation tree until you find the instance of the Meta-Directory component you would like to start.

3. Select the instance and right-click.

   A context menu appears.



4. Select Start Server.

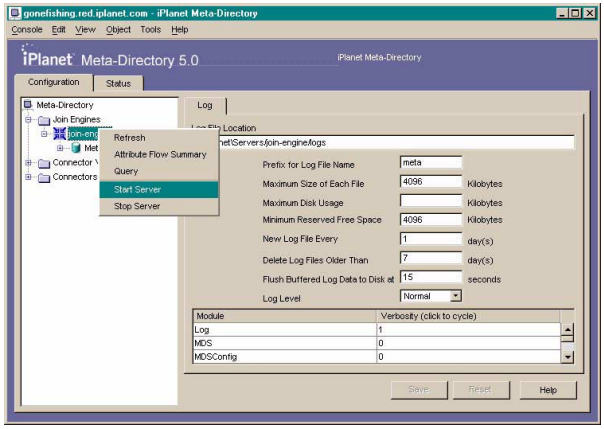
   A message appears stating that the server has been started.

To Start From the Meta-Directory Console Navigation Tree

1. Open iPlanet Console by clicking Start and choosing Programs > iPlanet Server Products > iPlanet Console.
2. Open the branches of the navigation tree until you find the instance of the Meta-Directory component you would like to start.
3. Select the instance and click Open in the upper right corner of the window.

4. Select the instance from the Meta-Directory console navigation tree and right-click on it.

   A context menu appears.



5. Select Start Server.

   A message appears stating that the server has been started.

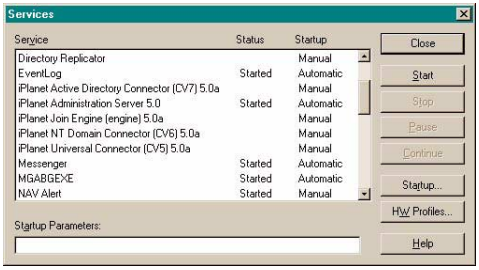
Starting Components Via the Meta-Admin Command-Line Tool

You can also start the join engine or connectors by using the Meta-Admin command-line tool. Because the Meta-Admin Command-line tool works in conjunction with an instance of Administration Server, the Administration Server can be used to start components remotely. For information on this tool, you can read Chapter 16, “Command-Line Administration.” For commands to use with it, you can read Appendix B, “The Meta-Admin Protocol.

Starting Components Via Windows NT Services

Additionally, you can start the join engine or connectors from the Services Control Manager control panel in Windows NT.

1. Click `Start` and choose Settings > Control Panel.
2. In the Control Panel, double-click Services.



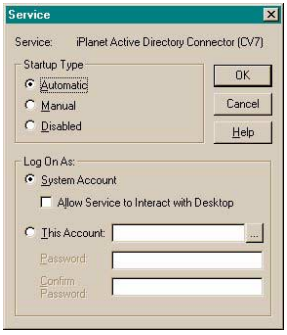
3. Select the iPlanet component you would like to start.
4. Click Start.

Automatic Start-up at Log On

This method of start-up can also be used to automatically start components upon your Windows NT system's startup.

1. Click `Start` and choose Settings.
2. In the Control Panel, double-click Services.
3. Select the component from the Services window that you would like to startup when you log on.

4. Click Startup and the Service window appears.



5. Select Automatic and System Account.
6. Click OK.

Stopping Meta-Directory Components

On Windows NT systems, you can stop Meta-Directory components with the iPlanet Console, the Meta-Admin command-line tool or the Windows NT Service Control Manager. On Solaris systems, you can stop Meta-Directory components with the iPlanet Console or the Meta-Admin command-line tool.

Stopping Components Via the Console

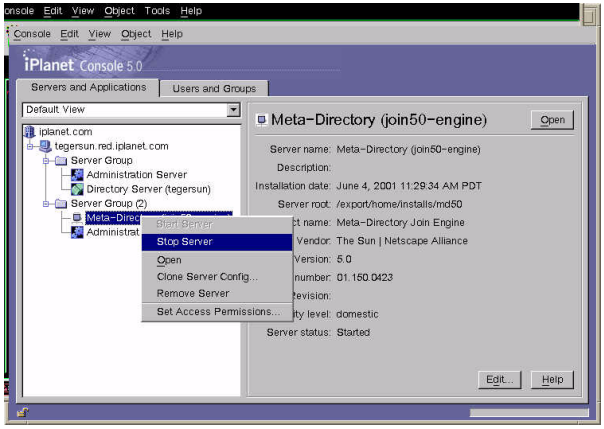
You can stop the join engine or connectors either by way of iPlanet Console or Meta-Directory console.

To Stop From the iPlanet Console Navigation Tree

1. Open iPlanet Console by clicking Start and choosing Programs > iPlanet Server Products > iPlanet Console.
2. Open the branches of the navigation tree until you find the instance of the Meta-Directory component you would like to stop.

3. Select the instance and right-click.

   A context menu appears.



4. Select Stop Server.

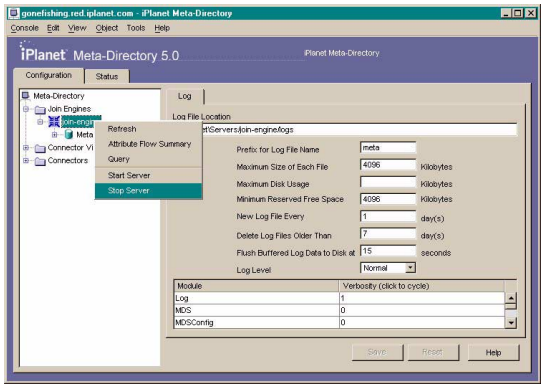
   A message appears stating that the server has been stopped.

To Stop From the Meta-Directory Console Navigation Tree

1. Open iPlanet Console by clicking Start and choosing Programs > iPlanet Server Products > iPlanet Console.
2. Open the branches of the navigation tree until you find the instance of the Meta-Directory component you would like to stop.
3. Select the instance and click Open in the upper right corner of the window.

4. Select the instance from the Meta-Directory console navigation tree and right-click.

   A context menu appears.



5. Select Stop Server.

   A message appears stating that the server has been stopped.

Stopping Components Via the Meta-Admin Command-Line Tool

You can stop the join engine or connectors by using the Meta-Admin command-line tool. As the Meta-Admin Command-line tool works in conjunction with an instance of Administration Server, the Administration Server can be used to start components remotely. For information on this tool, you can read Chapter 16, “Command-Line Administration.” For commands to use with it, see Appendix B, “The Meta-Admin Protocol.”

Stopping Components via Windows NT Services

You can stop the join engine or connectors from the Services Control Manager control panel in Windows NT.

1. Click Start and choose Settings > Control Panel.
2. Choose the iPlanet component you would like to stop.
3. Press Stop.

Checking a Component's Operational Status

The following procedure allows you to check whether a specific component is in operational mode.

1. In the Meta-Directory console, click the Status tab.
2. Select either the join engine or a connector from the navigation tree.
3. Click the Operations tab.

   The service status of the component is at the top of this panel. *Up* refers to a running component while *down* refers to a component that has stopped.

For more information on the Operations panel, see "Operations," on page 251 of Chapter 14, "Monitoring Meta-Directory Components."

Automated Restarts

To automate a server restart, the administrator can prevent the erasure of `start.conf` as explained in "The start.conf File," on page 241. By preventing the erasure of the distinguished name and password, the file can be reused. The first two procedures described are the simplest way to automate a server restart but, offer relatively little or no server security. The last procedure offers a higher level of server security. Once the start.conf file is re-configured, a script can be written to detect a shutdown and restart the component.

Automating a Restart on Windows NT Systems

To automate a server restart on Windows NT, you need to prevent the server from erasing the authentication details in the `start.conf` file. To do this:

1. Select the `start.conf` file of the component you would like to automate in Windows Explorer.
2. Right-click and select Properties.
3. Check Read-only so the file can not be written to.

This procedure allows for the reuse of the authentication details from the last start request. A script can be written to detect whether the component is down and restart if necessary.

Automating a Restart on Solaris Systems

To automate a server restart on the Solaris environment, you need to prevent the server from erasing the authentication details in the `start.conf` file. To do this enter the command:

```
chmod ugo-w start.conf
```

This procedure allows for the reuse of the authentication details from the last start request. A script can be written to detect whether the component is down and restart if necessary.

Automating a Restart With High Security on Windows NT Systems

To allow for automated restarts on Windows NT while achieving the highest level of security possible:

1. Create a login account for your system that will have exclusive permissions to read the `start.conf` file.
2. Configure all four Meta-Directory components (join engine, NT Domain connector, Active Directory connector, and Universal connector) to run as that administration account.

   The four components are configured by selecting from the Windows NT Start menu. Choose Settings > Control Panels > Services. From there select the component to be configured and select Startup. Check Log On As: and use the information created for the trustee account.

NOTE To enable the ability to change startup parameters via the Console, the Administration Server associated with the join engine must be permitted to write to the `start.conf` file. If the Administration Server can read but not write `start.conf`, Console users may start the Meta-Directory server but only with the same parameters already stored in `start.conf`.

Chapter 14

Monitoring Meta-Directory Components

The resources available from the Status tab of Meta-Directory console are used for monitoring operations of the components. For example, you can find out whether a component is enabled, update the connector view or the external data source, refresh data, and specify how activity, usage and data changes statistics are recorded in log files. This chapter contains the following sections:

- Operations
- Logs
- Statistics and Status Messages

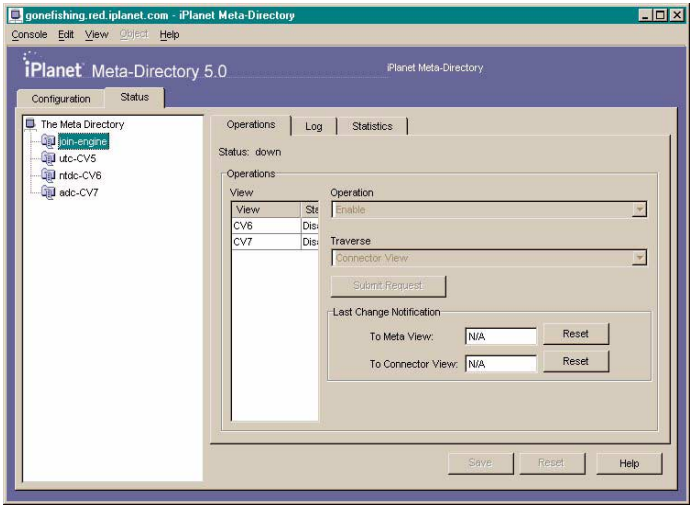
Operations

Within the Status tab of Meta-Directory console, an Operations tab lets you view the component's operational status and initiate requests, such as enabling, disabling, and refreshing the component.

Join Engine Operations

You can view and change the operations of the join engine and its data by selecting the join engine and clicking Operations from the Status tab.

Figure 14-1 The Operations Window for the Join Engine



The following sections explain the fields within the join engine's Operations tab.

Status

The Status field tells you whether the join engine is currently running (up) or not running (down).

View

The View field lists all participating views and whether they are enabled. An enabled view is currently flowing data while a disabled view is not. The participating view can be enabled even though the join engine status is down.

Operation

From the Operation drop-down list, you can choose options to apply to the participating view that is selected in the View field.

Enable Choose this option to allow the selected participating view to flow data. (This option disables the Traverse option.)

Disable Choose this option to stop the selected participating view from flowing data. (This option disables the Traverse option.)

Refresh Choose this option to scan the entries in the connector view or the meta view (depending on your selection in the Traverse drop-down menu) for changes. The changes will be integrated and the link information recreated based on the join rules defined in the participating view.

Refresh Unlinked Choose this option to refresh only unlinked entries in the connector view or the meta view depending on the option selected in the Traverse drop-down menu.

Refresh Groups Choose this option to refresh entries in the connector view or the meta view depending on the option selected in the Traverse drop-down menu. The group filter, which allows you to refresh certain entries of the view, is applied prior to the refresh. More information on the group filter can be found in “Select the Group Filters tab.,” on page 118.

Break Links Choose this option to break all links between the connector view and the meta view. After the links are broken, the join rules need to be re-applied to recreate the link information.

Validate Links Choose this option to check for invalid links. It also attempts to re-join the data based on the join rules defined in the participating view.

Traverse

When chosen with one of the refresh or link operations above, the Traverse option allows you to choose whether to scan the entries in the meta view or the connector view. Once the Operation and Traverse options are chosen, click Submit Request to begin the operation.

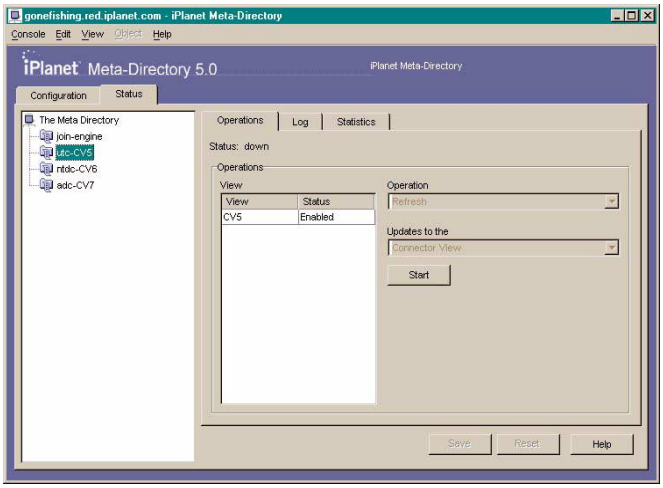
Last Change Notification

The change log assigns a sequential number to changes. The last change notification refers to the last change number the join engine processed. The join engine will only process the change log numbers greater than the number in the last change notification field. The Reset button resets the value.

Connector Operations

You can view and change the operations of a connector and its data by selecting the connector and clicking Operations from the Status tab.

Figure 14-2 The Operations Window for the Connectors



The following sections explain the fields within the connector's Operations tab.

Status

The Status field tells you whether the selected connector is currently running (up) or not running (down).

View

The View field lists the connector's participating view and whether or not it is enabled. An enabled view is currently flowing data while a disabled view is not. The participating view can be enabled even though the connector status is down; data would flow if the connector was running.

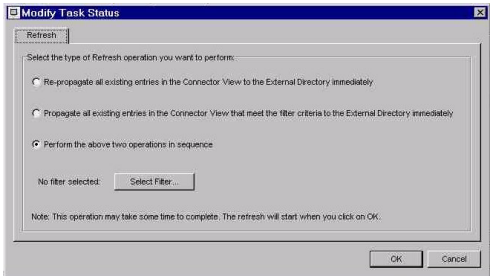
Operation

The only available option in the drop-down list is Refresh which scans all entries based on choices described in Updates To The.

Updates to the

Choose either External Directory or Connector View to work with the Refresh operation. The Modify Task Status dialog box appears after the selection is made and Start is pressed. (If Connector View is choosen, you only have one option.)

Figure 14-3 Modify Task Status Dialog Box



Choose from the following Updates to the External Directory options and click OK:

- Re-propagate all existing entries in the Connector View to the External Directory immediately.
- Propogate all existing entries in the Connector View that meet the filter criteria to the External Directory immediately.

  Select the filter desired. Only those configured for the "NoSubtreesExcept" option are displayed when Select Filter... is chosen, not filters configured for the "AllSubtreesExcept" option.
- Perform the above two operations in sequence.

NOTE After any link operation in either the join engine or connector, the logs should be examined for entry failures by issuing `grep -i fail *.log` in the log directory specified in the console. Any entry failure would need to be fixed by issuing the appropriate command through the Fix-It tool. For example:

- Errors encountered during a refresh operation would need to be manually joined
- Failures encountered during a break links operation would need to be manually unlinked

---

Logs

Log files record most component activities including the opening and closing of connections, activity times, and system interruptions. Each component (join engine, Universal Text Connector, Windows NT Domain connector and Active Directory connector) has the ability to write informational records and error messages to its own log files. You can specify what is written in the log files and use them to monitor Meta-Directory operation. This section describes the log files and how to configure their preferences.

Log Files

Each Meta-Directory component has the ability to record administrative activities and operational errors in an ASCII-text log file. Each log file is located in the logs folder within the component folders named adc-*cvname* (Active Directory connector), ntdc-*cvname* (Windows NT Domain connector), join-engine (join engine) and utc-*cvname* (Universal Text Connector) where *cvname* is the name of the connector view. The component folders, with their log files, are located within the directory where Meta-Directory is installed (the default is `NETSITE_ROOT/iPlanet/Servers`).

Regardless of the component, the naming convention for each log file is:

prefix`-yyyymmdd-nn.log`

In the log's file name, `nn` is a numeric extension (starting with 01). This extension numbers the log files in the order in which they are composed. Because the administrator can set the frequency with which new log files are started (weekly or daily) as well as the maximum amount of disk space used to contain them, the oldest logs in a component's sequence are deleted, as necessary, to avoid exceeding these constraints.

The log format is readable (by people) and analyzable (by software). Each log file consists of a sequence of lines. Each line represents a separate activity as it occurs. Here is a sample log line reporting Meta-Directory activity:

```
[2001/05/01 16:32:26.37 -0700] 2237:07F3 3 DataAccess <- explode_DN
4 RDNs
```

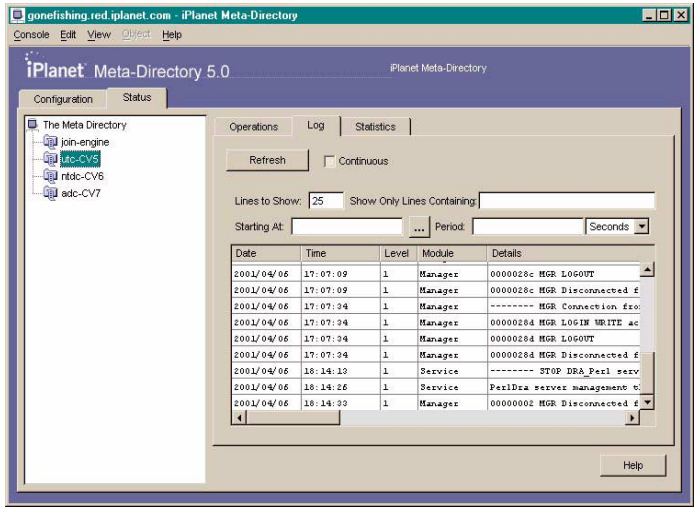
NOTE The recorded time in all log files may be off by a second or two; the event may have actually occurred a second or two later than the time logged due to server activity.

- 2001/05/01 16:32:26.37 refers to the local date and time.
- -0700 refers to the time zone, in this case Pacific Daylight Time.
- 2237 refers to the process number.
- 07F3 refers to the thread number.
- 3 refers to the level of verbosity.
- DataAccess, in this example, is the name of the module in which the activity occurred.
- All information following the module name describes the operation that took place.

Log Window Options

The following sections explain the fields within the Logs tab.

Figure 14-4 The Log Window for Connectors



Refresh

To activate log settings and update the log table to include any new logs, click Refresh. Refresh must be activated after any of the Log window options are modified.

Continuous

To update the log table continuously, choose Continuous.

Lines to Show

To limit the maximum number of lines you would like to have in the log table, enter a number here.

Show Only Lines Containing

To limit the set of lines with a filter, enter a text search string here.

Starting at

Use this field to time-delay the start of a search. Click the "..." button to use a time-date calendar for easy input.

Period

This field is used to specify the time duration for the log search. For instance, if you want to display all logs that occurred up to 90 minutes after the starting time specified above, select Minutes from the drop-down menu and enter `90`.

Modules

Operations recorded to log files refer back to specific functional areas of Meta-Directory or *modules*. Each module concerns itself with a particular task performed by a specific component. Any activity or operation recorded in a log file is recorded to the file which correlates to that module. In addition, Meta-Directory generates debug log files for these modules. These debug logs give more technical and detailed explanations of their respective modules than the customary logs. Table 14-1 and Table 14-3 describe the modules associated with the join engine and the connectors. Table 14-2 and Table 14-4 describe the debug logs. (Because modules work together to perform many tasks, it is often necessary to combine log file information to come to a more complete understanding of the module's functions.)

Table 14-1 Join Engine Modules

Module Name	Description
Log	The Log module controls the logging function from within the application.
MDS	The MDS module controls the participating view and its interaction with the meta view.
MDSConfig	The MDSConfig module controls the updates of, and status for, the LDAP configuration used by the join engine.
DCNS	The DCNS module searches the changelog for new data in the external data source and notifies the join engine or connector so that the appropriate action can be taken.
DataAccess	The DataAccess module performs operations including adding and modifying data and deleting and searching entries on the external data source.

Table 14-2 Join Engine Debug Logs

Debug Log Name	Description
DebugDataAccess	The DebugDataAccess log provides more detailed information about the DataAccess module.
DebugDirectoryAccess	The DebugDirectoryAccess module provides a detailed description of the module that controls LDAP accessory client functions.
DebugDBServer	The DebugDBServer log provides a detailed description of the actions of the database server.
DebugCLOGPlugin	The DebugCLOGPlugin log provides a more detailed description of the actions of the changelog.
DebugMDSConfig	The DebugMDSConfig log provides a more detailed description of the actions of the MDSConfig module.
DebugJoinController	The DebugJoinController log provides information concerning the entry links and ownership that are applied during the join process.
DebugMapper	The Debug Mapper log provides information about the processing of the join process rules.
DebugDCNSServer	The DCNSServer log provides information concerning the registering with the DCNS servers.

Table 14-2 Join Engine Debug Logs *(Continued)*

Debug Log Name	Description
DebugDCNSClient	The DCNSClient log keeps track of change notifications.
DebugMDSLogging	The DebugMDSLogging module provides more detailed information concerning the log module.
DebugMDSUtils	The DebugMDSUtils log provides information about the utilities used internally by the Meta-Directory.
DebugThreadPool	The DebugThreadPool log provides information about the thread pool utility.
DebugOCIWRAPPER	The DebugOCIWRAPPER log provides information about the join engine's interaction with an Oracle database.
DebugDBCLOGPlugin	The DebugDBCLOGPlugin og provides information about the changelog for the database connector.

Table 14-3 Connector Modules

Module Name	Description
Log	The Log module controls logging for the connection between the console and the connectors.
MDSConfig	The MDSConfig module controls the updates of, and status for, the LDAP configuration used by the connectors.
DCNS	The DCNS module searches the changelog for new data in the external data source and notifies the join engine or universal connector so that the appropriate action can be taken.
DataAccess	The DataAccess module performs Add, Modify, Delete, and Search operations, issued by the join engine, on the database. The plugin interacts with the join engine while uploading data from the database connector view to the meta view.
AVL	The AVL tree database module is an internal cache that represents the external database and the connector view.
LdapConfig, REGISTRY	These two modules manage configuration for the Universal connector.
DRAConfiguration	The DRAConfiguration module stores the values of the indirect connector rules: Filter, Attribute Flow and Default Attribute.

Table 14-3 Connector Modules *(Continued)*

Module Name	Description
DDeltas	The Directory Deltas module deals with the movement of data from the connector view to the external data source.
DraDua	The Directory Access module deals with communication with an LDAP data source.
FileIO	The FileIO module deals with certain file i/o utility routines.
LogStuff	The LogStuff module concerns itself with miscellaneous routines of logging.
LDeltas	The Local Deltas module concerns itself with moving data from the external database to the connector view.
Manager	The Manager module concerns itself with the management command processor.
Service	The Service module concerns itself with the Universal Connector's startup routines.
PerlDDM	The Perl Directory Deltas Module concerns itself with processing the Perl Interface.
PerlMain	The PerlMain script module deals with the Universal Connector's startup entry point.
Pldeltas	The Perl Local Deltas script module deals with processing the Perl interface.

Table 14-4 Connector Debug Logs

Debug Log Name	Description
DebugDataAccess	The DebugDataAccess log provides more information about the DataAccess module.
DebugDirectoryAccess	The DebugDirectoryAccess log provides more information about the module that controls LDAP accessory client functions.
DebugDCNSServer	The DCNSServer log provides information concerning the registering with the DCNS servers.
DebugDCNSClient	The DCNSClient log keeps track of change notifications.
DebugCLOGPlugin	The DebugCLOGPlugin log provides more information about the actions of the changelog.

Table 14-4 Connector Debug Logs *(Continued)*

Debug Log Name	Description
DebugMDSConfig	The MDSConfig log provides more information about the updates of, and the status for, the LDAP configuration used by the connectors.
DebugDASearch	This log provides more information about theLDAP search function.
DebugDATraverse	This log provides more information about the LDAP traverse function.
DebugMDSLogging	The DebugMDSLogging log provides more information about the logging module.
DebugMDSUtils	The DebugMDSUtils log provides more information about the utilities used internally by the Meta-Directory.
DebugThreadPool	The DebugThreadPool log provides more information about the actions of the service threads.

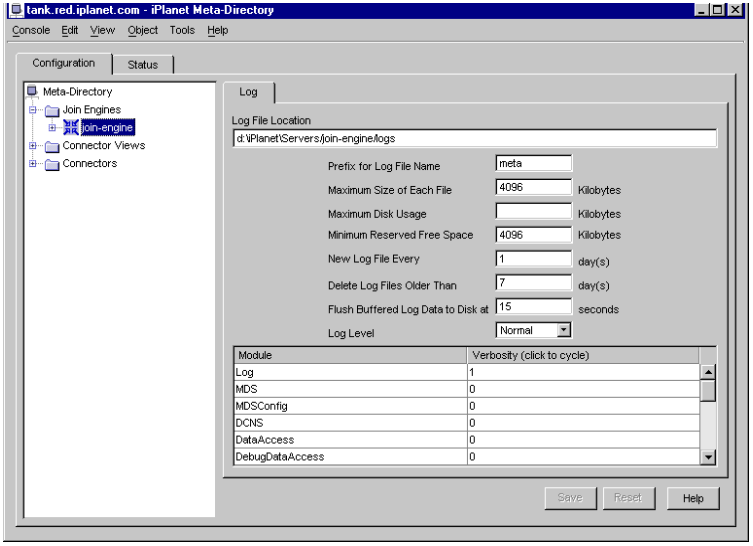
Specifying Log File Preferences

Log file preferences such as name, location, log level and verbosity are set specifically for each component. By selecting the component, you will be able to configure its log file preferences.

Working within the Log Window

1. From the navigation tree in the Meta-Directory console, select a component (join engine or connector instance).

   The Log Window appears.



2. Provide values by entering specifications for the following fields:

Log File Location

Specifies the directory into which the log files are saved. To specify a directory other than the default, enter the full path name in this field.

Prefix for Log File Name

Specifies the word to precede the log file name as:

prefix-yyyymmdd-nn`.log`

For example, if you enter `delta` as the prefix, it would appear as in the following example:

`delta-20010521-01.log`

Maximum Size of Each File

Specifies the maximum size of the log file. After reaching this size, a new log file is created. The default is 4096 KB.

Meta-Directory checks the log file size in five minute intervals. If, between the intervals, the log file reaches and surpasses the maximum size, Meta-Directory will allow this until the maximum file size is checked again on the five minute mark. At that point, a new log file is created.

Maximum Disk Usage

Specifies the maximum disk space set aside for logging. When the maximum disk space is reached, logging is temporarily halted. The default is no maximum. (The field is blank.)

If you specify a maximum available disk space, it must be at least twice the size of the maximum log file size, because the service always stores one backup log file. If the maximum available disk space is less than twice the maximum log file size, the new file cannot reach its maximum size.

Minimum Reserved Free Space

Disk space is reserved for log file storage. By default, the service reserves a minimum of 4096 KB of space.

New Log File Every

Determines the length of time that a log file should record activities before creating a new log file. If the maximum size of the log file is reached first, a new log file is created. The default is 1 day. This value must be less than the value in the Delete Log Files Older Than field.

Delete Log Files Older Than

Determines when log files will be automatically deleted. The default is 7 days. This value must be greater than the value in the New Log File Every field. (Active Directory connector lacks this functionality.)

Flush Buffered Log Data to Disk at

Determines when the log data buffered in memory is to be flushed to the disk.

Log Level

Each component's logging function can be turned on or off via the Log Level. The three levels of logging are as follow:

- Off - Nothing is logged.
- Normal - Necessary information for daily administration is logged. (Disk space requirements are likely to be small and new files are created infrequently.)
- Debug - Maximum amount of information needed for debugging is logged. (Disk space requirements are likely to be large and new files are created frequently.)

You can switch levels without restarting the join engine.

Verbosity

The verbosity of a log refers to the description of an activity within a log line. Unlike the log level which controls the logging capability at the component level, verbosity controls the activity description at a module level within each component. (For example, the log level for join engine might be set to `Normal` while two specific join engine modules, Log and DataAccess, are set to verbosity levels 1 and 3, respectively.) The verbosity levels are as follow:

0 - No logging for specified module

1 - Default (minimum) verbosity level for specified module

2 - Moderate amount of logging for specified module

3 - Maximum amount of logging for specified module

3. Click Save.

Statistics and Status Messages

The Statistics window is the error reporting utility of Meta-Directory. After processing an administrative task, you can check the Statistics window for any success or error messages returned by the component processing the task.

From the Status tab navigation tree in the Meta-Directory console, select a component (join engine or connector instance) and click Statistics. For an explanation of the messages that can be returned, see “Error Responses,” on page 321.

Chapter 15

Administration Tools

Meta-Directory comes with tools to help you find and fix entries that go unlinked as well as customize entry properties. This chapter contains the following sections:

- The Query Tool
- The Fix-It Tool
- Command Line Access of the Query / Fix-It Tool
- The Property Editor

The Query Tool

With the Query Tool, administrators can generate many types of searches including, but not limited to these:

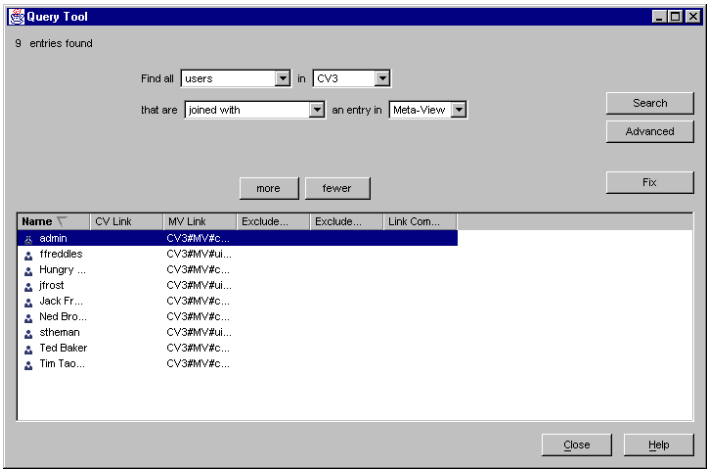
- a search for all entries created in Meta-Directory in the last *n* hours
- a search for all entries from a connector view that are / are not joined into the meta view
- a search of statistics for joined, unjoined, or unassociated entries
- a search for all entries in a connector view that were not automatically joined into the meta view

The following sections provide procedures on accessing the Query Tool and constructing basic and extensive searches.

To Access the Query Tool

1. Open the Meta-Directory console and right-click anywhere in the navigation tree.
2. Choose Query from the context menu that appears.

   The Query Tool opens. Alternatively, from the menu bar, open Tools and choose Query.



Searches

To request a search for entries within a meta view or a connector view specify search criteria using the tool's drop-down lists. The choices will complete the search request as follows.

Find all *group choice* in *view choice* that are *associated in a particular way with* an entry in *view choice*.

NOTE You can conduct an even more basic search by clicking on Fewer. This allows a search using only the first two drop-down lists.

To Do a Search

1. From the Content Selection Filters drop down list (“Find all”), select the grouping you would like to search (entries, users, groups, or users and groups).
2. From the Views drop-down list (“in”), select either a connector view or a meta view.

   This menu shows all configured views in Meta-Directory.
3. From the Linkage Selection Filters drop-down list (“that are”), select how you would like to perform your search.

   The choices consist of the possible ways that entries can be linked. They include: *unassociated with*, *joined with*, *joined manually with*, *joined automatically with*, *excluded from*, *able to delete*, *not able to delete*, *members with*, or *are non members with*.
4. From the Corresponding View drop-down list (“an entry in”), select the corresponding view.

   If you selected meta view from the Views drop-down list, you will choose from participating connector views. If you selected a connector view from the Views drop-down list, your only choice will be the meta view.
5. Click Search. The results appear in the bottom pane.

Extensive Searches

The Query Tool allows the administrator to build broader and more complex searches by clicking the `More` button and including or excluding entries in other meta or participating views.

To construct a more extensive search, build your query using the first two drop-down lists as many times as needed in the following form:

> Find all *group choice* in *view choice* that are *associated in a particular way with* an entry in *view choice* **[and/or** that are *associated in a particular way with* an entry in *view choice***] [and/or** that are *associated in a particular way with* an entry in *view choice***] [and/or** that are *associated in a particular way with* an entry in *view choice***]** et cetera.

You chose *and* or *or* to further refine the search request based on the variables that follow. The rest of the search request is constructed as described in “Searches,” on page 268. However, `More` can be clicked an infinite number of times, adding to the complexity of the search criteria.

Constructing a Search With an LDAP Filter

To construct a search using an LDAP filter entry, click `Advanced` from the Query Tool. A new field appears that enables you to enter an LDAP filter. The new LDAP filter query form is:

> Find all *group choice* in *view choice* matching the LDAP filter *{filter}* that are *associated in a particular way with* an entry in *view choice.*

An LDAP filter or SQL Where clause can be entered in the new field; for example:

```
telephonenumber=4*
```

The rest of the query form is construct as described in “Searches,” on page 268.

NOTE The filter will be ignored if the data is not valid.

Customizing the Query Tool

The Query Tool window can be customized based on the administrator’s needs. Right-click anywhere on the window to open a context menu, which provides the following customization options:

- View selected entry
- Customize column headers for meta view
- Customize content selection filters
- Customize linkage selection filters

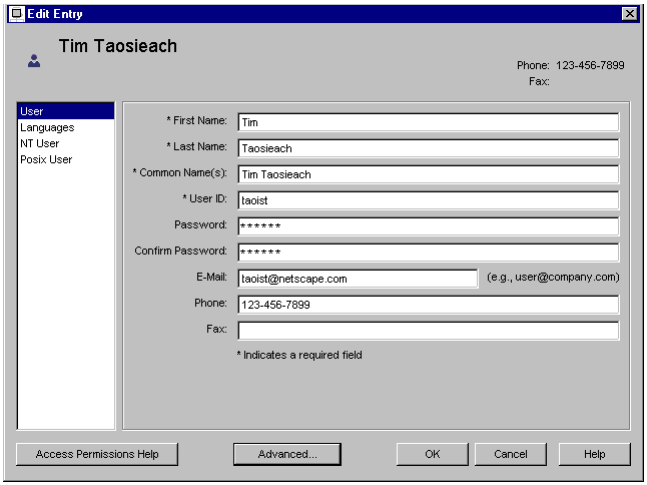
The sections that follow provide procedures for customization.

Viewing a Selected Entry

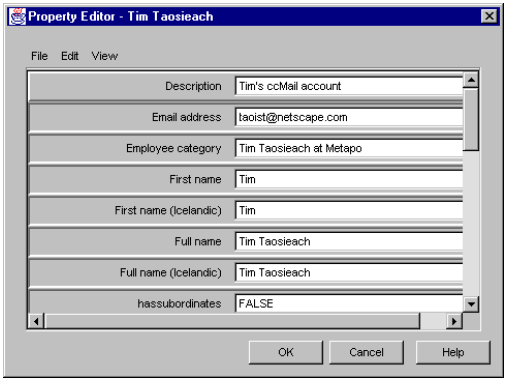
This option lets you view and edit the properties of a chosen entry listed as the result of a search request.

1. Select an entry from the list in the Query Tool dialog box.
2. Right click in the Query Tool dialog box and select `View selected entry` from the context menu.
3. Either the Edit Entry or Property Editor dialog box will appear. The Edit Entry box will appear if any of the following is true:
    - The entry is LDAP-based.

- The entry has a standard option class.
- You have invoked the Query Tool from the console.



In any other circumstances, the Property Editor dialog box will appear.



NOTE If the Edit Entry dialog box appears, you can access the Property Editor dialog by clicking Advanced.

4. Modify the fields in either dialog box as discussed in “The Property Editor,” on page 286.

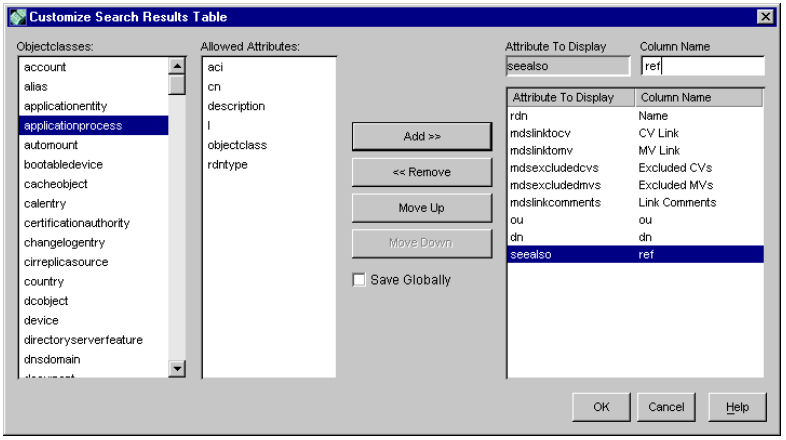
5. Click OK to return to the Query Tool.

NOTE For Database connectors only, the Property Editor is named the Database Entry Viewer. The Database Entry Viewer functions in the same way as the Property Editor.

Customizing Column Headers

The columns in the search result portion of the Query Tool can be customized by renaming, adding, or deleting the column headers.

1. Right click in the Query Tool dialog box to open a context menu.
2. Select Customize column headers for meta view to open the Customize Search Results Table.



3. Select the object class from the Objectclasses column that includes the attribute you would like to chose as a column name.
4. Select the attribute itself from the Allowed Attributes column. (Use the Control key to select more than one attribute.)
5. Click Add to add the attribute selected.

   The attribute you added now appears in the Attribute To Display field on the right side of the dialog box. For each attribute, you can enter a name in the Column Name field to represent the attribute. It is this name that will appear as the column header when you return to the Query Tool.

6. Optional: Use the following buttons and checkbox as needed.

   - Click Remove to remove unwanted attributes displayed in the Attribute To Display field and return them to the Allowed Attributes list.
   - Click Move Up and Move Down to arrange attributes in the preferred order for your displayed headings. (Moving the column name up will move the coulmn itself to the left.)
   - Click Save Globally to save the headers for everyone who accesses the system.

7. Click OK.

   The Query Tool dialog box reappears with the attributes and column names you specified. Clicking Search again will re-assemble your results based on the new columns.

Sorting the Display of Entries

You can sort entries resulting from the search request in accordance with the column labels that appear above the display of data. The labels function as a toggle; alternately sorting the data in ascending and descending order.

To sort the display, click a label. For instance, to sort entries by the meta view link, click MV Link. The sorting mark (a triangle pointing in the direction the column is sorted) appears beside MV Link in the column label.

To Specify Language Preferences

Meta-Directory provides `dn`, `rdn`, and `rdtype` attributes to allow language preferences. Each attribute supports only one language, so you could set one attribute's preferred language to Spanish (es), for example, while the default for all others is English (en). This entry might appear as shown in the following example:

```
dn: cn=Fred G. Garcia, ou=people, o=siroe.com
cn: Fred Garcia
cn;lang-es: Frederico Garcia
location: USA
location;lang-es: Los Estados Unidos
sn: Garcia
mail: Garcia@siroe.com
```

Suppose that you set up in the Customize Search Results Table, the attributes and column names in Table 15-1. Any language can be used in the column names because the names are text strings of your choice. Some names appear in English; some are in Spanish. These column names are copied directly into the column headers in the search results table in the Query Tool dialog box.

Table 15-1 Column Specifications

Attribute	Column Name
sn	Last Name
cn	Whole Name
location	Country
mail	Email Addresses
dn	Entry Name
rdn	Nombre
rdntype	Nombre Con Idioma

When you request a search, the Query Tool finds each qualifying entry and places its attributes in the correct columns based on your specifications in the Customize Search Results Table dialog box. The search results appear in the columns whose names are shown in Table 15-2.

Table 15-2 Search Results for the Sample Entry

Value	Column Name
`Garcia`	Last Name
`Frederico Garcia`	Whole Name
`Los Estados Unidos`	Country
`Garcia@siroe.com`	Email Addresses
`cn=Fred Garcia, ou=people, o=siroe.com`	Entry Name
`Fred Garcia`	Nombre
`Frederico Garcia`	Nombre Con Idioma

Table 15-3 shows how the Query Tool decides the language subtype to choose.

Table 15-3 Query Tool Chooses the Language Subtype

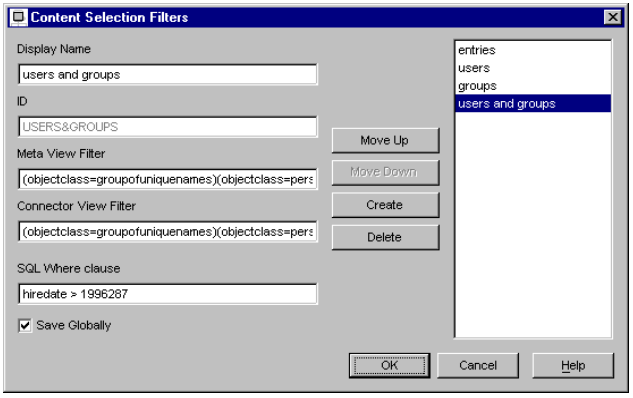
Attribute	Column Name	Operation
sn	Last Name	This attribute has no language-specific version in the entry, so the default is English and Garcia is displayed under this column.
cn	Whole Name	This attribute does have language subtypes in the entry. As the preferred language is `es` (Spanish) and the `cn` attribute has an "es" version, "Frederico Garcia" is extracted.
location	Country	This attribute also has a Spanish version, which is "Los Estados Unidos", so it is extracted and displayed under this column.
mail	Email Addresses	Since there is only an English version for `mail`, both English values are displayed.
dn	Entry Name	This attribute is fabricated. The `dn` is extracted from the entry and displayed under this column. Since there is only one `dn`, languages are not considered.
rdn	Nombre	This attribute is fabricated. The `rdn` (the first part of the `dn`) is extracted from the entry and displayed under this column. In this case, `rdn` is "Fred Garcia". There is only one `rdn` so language is not considered. The `rdn` is based on an attribute that also resides in the entry, which in this case is `cn`. In this entry, `cn` has different language subtypes. For those who choose French as their preferred language, this mechanism may not be helpful, since they may want to see only French, and not the English name that constitutes the `rdn`.
rdntype	Nombre Con Idioma	This attribute is fabricated, and addresses the issue associated with the `rdn` attribute. `rdntype` examines the attribute that the `rdn` is based on and extracts all values for the attribute, respecting language preferences. In this case, the `rdn` is based on `cn`, and the language preference is `es`, so the tool extracts Frederico Garcia.

Customizing Content Selection Filters

Content selection filters can be created or modified using this option. The entries added or edited in this window affect the selections available in the Query Tool Content Selection Filters drop-down list (“Find all”). The filters are used against the data in the entries of the context view (`cn`, `sn`, `Location`, `Phonenumber`, etc.).

Accessing the Content Selection Filters Window

1. Right-click in the Query Tool dialog box to open the context menu.
2. Select Customize content selection filters to open the Content Selection Filters dialog box.



To Create a New Content Selection Filter

1. In the Content Selection Filters dialog box, click Create.

   A default of NEW FILTER appears.

2. Provide values for the following fields:

Field	Description
Display Name	This field carries the name of the filter. It appears in the Query Tool content selection filters drop-down list.
ID	This read-only field is available only when a new filter is created. It is the unique identifier of the filter.
Meta View Filter	Specify the object class by which you want entries filtered in the meta view. If you do not specify a filter in this field, the Query Tool interprets the field as `(objectclass=*)`.

Field	Description
Connector View Filter	Specify the object class by which you want entries filtered in the connector view. If you do not specify a filter in this field, the Query Tool interprets the field as `(objectclass=*)`.
SQL Where Clause	If the data you are searching is in an SQL-accessible database, specify the where clause of an SQL query.

3. Click OK to save changes and return to the Query Tool dialog box.

   The new filter appears in the content selection filters drop-down list.

To Edit a Content Selection Filter

1. In the Content Selection Filters dialog box, select an entry from the right pane.

   The display name, associated ID and values for the meta view and connector view filters appear in the corresponding fields.

2. Modify the information as needed.

3. Optional: Use the following buttons and checkbox as needed.

   - Click Delete to remove the filter.
   - Click Move Up and Move Down to arrange filters in the preferred order for your content selection filters drop-down filter list.
   - Click Save Globally to save the filter for everyone who accesses the system.

4. Click OK to save changes and return to the Query Tool dialog box. The changes are reflected in the content selection filters drop-down list.

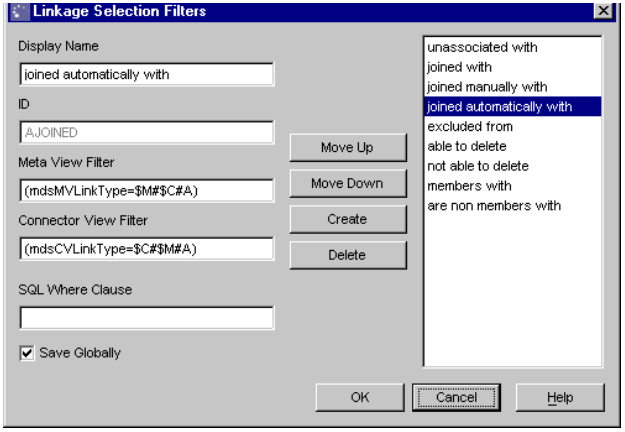
Customizing Linkage Selection Filters

You can create or modify linkage selection filters using the Linkage Selection Filters dialog box. The filters you add or edit affect the selections available in the Query Tool Linkage Selection Filters drop-down list (“that are”). The filters in that list allow you to search for entries by their linkage information.

Accessing the Linkage Selection Filters Window

1. Right-click in the Query Tool dialog box to open the context menu.

2. Select Customize linkage selection filters to open the Linkage Selection Filters dialog box.



To Create a New Linkage Selection Filter

1. In the Linkage Selection Filters dialog box, click Create.

   A default of NEW FILTER appears in the list box at the right and in the Display Name field.

2. Type the new filter's name in the Display Name field, and provide input for the remaining fields as described in "To Create a New Content Selection Filter," on page 276.

3. Click OK to save changes and return to the Query Tool dialog box.

   Your new filter appears in the Linkage Selection Filters drop-down list.

To Edit a Linkage Selection Filter

1. Select an entry from the list box at the right.

   The display name, associated ID and values for the meta view and connector view filters appear in the corresponding fields.

2. Modify the information as needed.

3. Optional: Use the following buttons and checkbox as needed.

   - Click Delete to remove the filter.

- Click Move Up and Move Down to arrange filters in the preferred order for your Linkage Selection Filters drop-down filter list.
- Check Save Globally to save the filter for everyone who accesses the system.

4. Click OK to save changes and return to the Query Tool dialog box. The changes are reflected in the Linkage Selection Filters drop-down list.

Query Tool Keyboard Shortcuts

The Query Tool provides a set of keyboard shortcuts. Table 15-4 provides a list of these shortcuts and what they do.

Table 15-4 Keyboard Shortcuts for Use With the Query Tool

Key Combination	Function
`Ctrl-f`	Launches the Fix-It Tool as discussed in “The Fix-It Tool,” on page 280.
`Ctrl-h`	Launches help.
`Ctrl-l`	Launches the Linkage Filters Editor dialog box.
`Ctrl-m`	Clicks the More button.
`Ctrl-o`	Launches the Content Selection Filters dialog box.
`Ctrl-q`	Exits the Query Tool.
`Ctrl-s`	Starts the search.
`Ctrl-w`	Clicks the Fewer button.
`Ctrl-z`	Launches the Customize Search Results Table dialog box.
`Ctrl-space`	Shows all of the currently selected entry.
`Ctrl`	Enables you to select entries one by one from the Query Tool search results table.
`Shift`	Enables you to select contiguous blocks of entries from the Query Tool search results table.
`Enter`	If search results are already displayed, launches the Fix-It Tool. If no search results yet, the search starts.

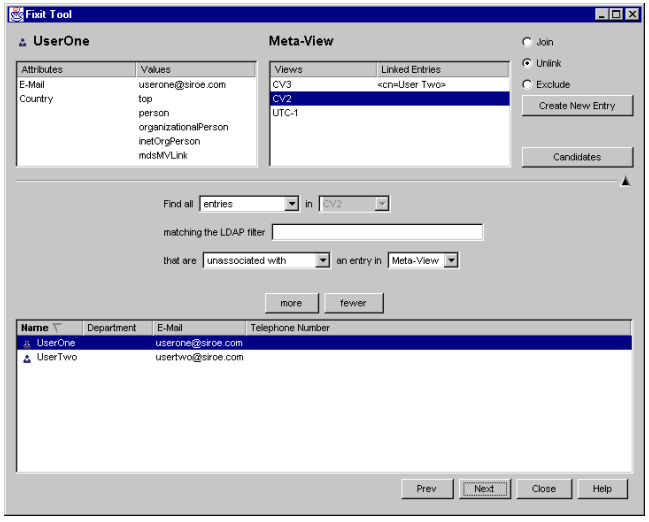
The Fix-It Tool

With the Fix-It Tool, you can create or delete manual links between entries when automated linking breaks. This allows an administrator to establish the linkages for hard to match entries or to review Meta-Directory's predetermined linking. You can also mark or unmark entries as excluded, and assign or remove ownership of an entry.

Accessing the Fix-It Tool

Select an entry from the search results table of the Query Tool and click Fix. The Fix-It Tool window appears showing attributes and values for the selected entry in the upper left hand corner. It's associated views and the partial DN of the selected entry (relative to the view's base DN) appears in the upper right hand corner. To view attributes and values for other source entries, click on the desired entry. The Fix-It Tool loads all the entries selected from the Query Tool dialog box.

Figure 15-1 Fix-It Tool



To Link an Existing Entry to a New Entry

1. Select the view entry for which you want to establish a link with a new entry.
2. Click Create New Entry.

   The New Entry DN Selector dialog box appears.
3. Select a DN from the drop-down list (populated from your DN rules) or enter your own, and click OK.

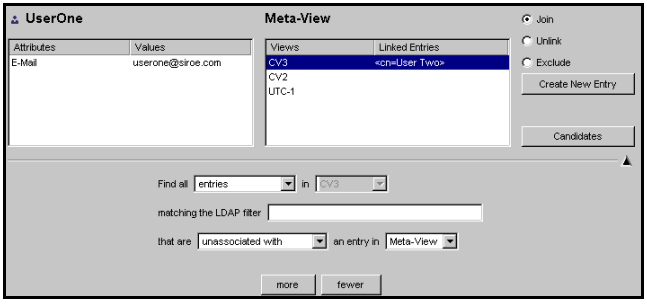
   Your input appears as a linked entry for the view you selected.

To Join, Unlink, or Exclude Entries

1. Select an existing entry from the Views column.
2. Click Candidates. The list of join candidates appears in the bottom pane, based on the join rules you have configured.

   You can also expand the search criteria by doing the following:

   a. Click on the triangle button below the Candidates button. A search panel appears that is preloaded with the search filter the join engine uses to retrieve join candidates. (The button is a toggle that closes the search panel when you click it again.)



   b. Change the fields in the search panel as desired, and click Candidates.

3. Select an entry from the candidates list and right-click.

   A context menu appears.

4. Select “Load this entry.”

   This adds the entry to the Fix-It Tool’s set of entries. (If you selected a set of entries from the Query Tool and then invoked the Fix-It Tool, the entry you are loading is positioned following the current entry.) You can still click `Prev` or `Next` to cycle through the list of entries.

5. Click one of the following radio buttons:

Manaual Operations	Description
Join	Links the current entry and the entry selected in the Candidates table, undoing any existing exclusion. The join will appear as a linked entry.
Unlink	Breaks any links or exclusions from the current entry to the entry in the view selected. Linked entries will disappear for the particular view.
Exclude	Excludes the current view from the view selected, breaking any existing link. The Linked Entries column will show “Exclude” for this view.

To Refresh a Current Entry

1. Right-click on the Fix-It Tool screen.

   A context menu appears.

2. Select Refresh.

   The status will change in the Linked Entries column.

To view and modify an entry

1. Select the candidate you want to view and right-click.

   A context menu appears.

2. Select View this entry.

   The Edit Entry dialog box appears.

3. View and change the entry as described in “Viewing a Selected Entry,” on page 270.

Customizing the Fix-It Tool

Customize the Fix-It Tool by doing the following:

1. Right-click on the Fix-It Tool screen.

   A context menu appears.

2. Select from the following choices:

Customization Choice	Description
One click joining mode	This feature joins a highlighted candidate with the source entry when you press Enter or click.
Auto load join candidates mode	This feature automatically searches for candidates when you switch to a new view or entry.
Auto advance entry mode	This feature automatically loads the next entry in the set. This is the default if you invoked the Fix-It Tool from the Query Tool.
Auto advance view mode	This feature automatically loads the next view. If no next view exists, the next entry in the set loads.
Auto advance unlinked view mode	This feature automatically advances to the next view with which the source entry is unlinked. It skips views with which the source entry is linked or excluded. If there are no more views, this option proceeds to the next entry.
Customize column headers	The procedure for customizing column headers is the same as for the Query Tool. To customize column headers, see “Customizing Column Headers,” on page 272.

Fix-It Tool Shortcuts

The Fix-It Tool provides a set of keyboard maps to enhance productivity. Table 15-5 provides list of these shortcuts.

Table 15-5 Fix-It Tool Shorcuts

Key Combination	Function
`Ctrl-e`	Excludes the current entry from the currently selected view.
`Ctrl-g`	Drops down or hides the search panel.
`Ctrl-h`	Invokes help.
`Ctrl-i`	Sets the “Auto advance view” mode.
`Ctrl-j`	Joins the current selected entry with the source entry.
`Ctrl-k`	Sets the “Auto advance unlinked view” mode.
`Ctrl-l`	Loads the currently selected entry into the Fix-It Tool as the source entry.
`Ctrl-m`	Clicks on the More button.
`Ctrl-n`	Invokes the Query Tool if not currently visible, or shifts it to the foreground if currently visible.
`Ctrl-o`	Toggles the one-click joining mode.
`Ctrl-q`	Exits the Fix-It Tool
`Ctrl-r`	Refreshes the current source entry. (Refetches from the Directory Server.)
`Ctrl-s`	Performs a candidates search.
`Ctrl-t`	Invokes the Customize Search Results Table dialog box.
`Ctrl-u`	Unlinks the current entry from the entry to which it is linked.
`Ctrl-w`	Clicks on the Fewer button.
`Ctrl-y`	Sets the “Auto advance entry” mode.
`Ctrl-z`	Invokes the Customize Search Results Table dialog box.
`Ctrl-space`	Shows all of the currently selected entry.
`Enter`	Joins the current selected entry with the source entry.
`Shift-Up`	Go up a view in the view list.
`Shift-Down`	Go down a view in the view list.

Table 15-5 Fix-It Tool Shorcuts *(Continued)*

Key Combination	Function
`Right Arrow`	Go to the next view, entry, or unlinked view based on the mode.
`Left Arrow`	Go to the previous view, entry, or unlinked view based on the mode.

Command Line Access of the Query/Fix-It Tool

You can invoke the Query / Fix-It Tool from the command line as well as the interface; the tool operates the same.

NOTE The exception to this rule is that the Property Editor dialog box appears rather than the Edit Entry dialog box. (See "Viewing a Selected Entry," on page 270 for more information.)

To Invoke the Query/Fix-It Tool

1. Locate either the `fixittool` (if you have a Solaris system) or `fixittool.bat` (if you have a Windows NT system) executable in the following location:

   `NETSITE_HOME``/bin/meta50/bin/...`

2. Run the command as follows:

   `fixittool` *binddn bindpw jeadminurl* where:

binddn	Specifies the data server bind DN to be bound to the data server URL.
bindpw	Specifies the password associated with the data server bind DN.
jeadminurl	Specifies the admin server URL of the join engine.

   If you provide all three of the options, the Query Tool appears directly. Otherwise, the standard console login dialog box appears.

The Property Editor

The Property Editor is used to alter the class properties for members within a connector view or a meta view.

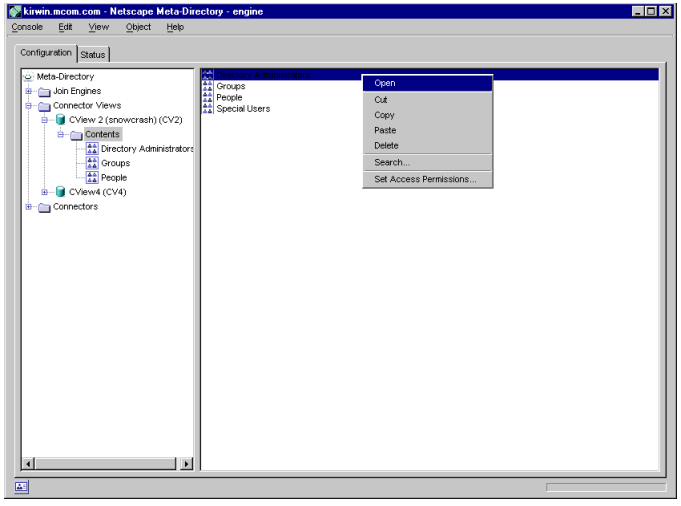
To Alter the Properties for Members

1. Click the Contents object of a particular connector or meta view.

   The members for the view appear on the right pane.

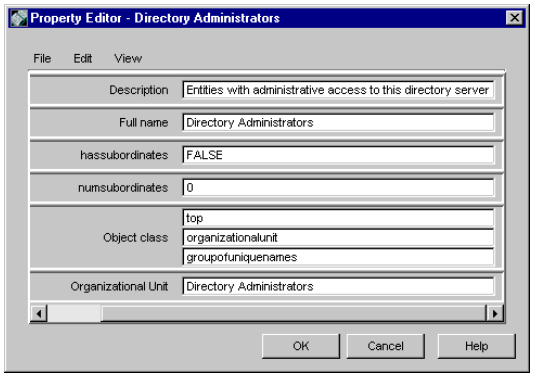
2. Click on a member class and right-click.

   A context menu appears.



3. Select Open.

The Property Editor dialog box appears.



You can use this dialog box to add, modify, or remove object classes and attribute values. You can also use an LDIF file for these purposes. Click Help for more information, or see the *iPlanet Directory Server Administrator's Guide.*

Chapter 16

Command-Line Administration

Meta-Directory provides a tool, the Meta-Admin tool, that lets you administer the Meta-Directory components from the command line. This tool is used in conjunction with the Meta-Admin protocol, which lets you perform the following administrative processes on a Meta-Directory component:

- Start a specified component.
- Stop a specified component.
- Get the status of a specified component.
- Get the statistics of a specified component.

The Meta-Admin Tool

The Meta-Admin Tool is a command-line utility that you can use to administer any running Meta-Directory component. The Meta-Admin Tool communicates with the Meta-Directory components through HTTP or HTTPS `POST` requests. Your requests are routed through the Administration Server CGI scripts to the specified component.

You specify the administrative tasks you want to perform using an LDIF file. The Meta-Admin Tool reads the LDIF file and translates your commands into `POST` requests, which it sends to the appropriate CGI scripts.

In the LDIF input you supply, each request you make contains a set of parameters whose values describe the administration task you want to perform. In response to your requests, the CGI scripts return an HTTP document to the Meta-Admin Tool. The tool translates these responses into success and error messages.

When you run it, the Meta-Admin Tool does the following:

1. Reads requests (in LDIF format) from `stdin`.
2. Sends the requests to the Meta-Directory components, via the Administration Server CGI scripts.
3. Echoes the Meta-Directory component responses to `stdout`.
4. Outputs any error messages to `stderr`.
5. Returns, as an exit status, the number of requests that failed.

The Meta-Admin Tool Syntax

The Meta-Admin Tool is a Perl script that is installed in *NETSITE_ROOT*`/bin/meta50/bin`. The script makes use of several Perl extension modules, including and the `Meta_Admin` module.

The tool uses nsPerl5.005_03, which is installed when you install the Meta-Directory software package. As a note, the Perl implementation must be set to "`use Meta_Admin`".

On Solaris systems, you execute the tool from either a shell or shell script. On Windows systems, you can execute the tool from either the Command Prompt window or from a batch file.

For example, you can issue the following commands from a command prompt to run the Meta-Admin Tool:

```
% cd NETSITE_ROOT/bin/meta50/bin
% ./meta-admin [options] < input.ldif
```

Here, it is assumed that *NETSITE_ROOT* is set to the directory root of your installed Meta-Directory package, *options* are any command-line options you want to specify, and *input.ldif* is the LDIF input you supply to the tool.

Meta-Admin Tool Command-Line Options

The following two tables define the command-line options that you can use with the Meta-Admin Tool. Table 16-1 explains the command-line options that you can use to modify your administrative requests:

Table 16-1 Request-related options for the Meta-Admin Tool

Option	Description
`-b` *URL*	Resolves all relative (incomplete) `admin-URL` values relative to the specified base *URL*.
`-p` *URL*	Sets the proxy to be used for the requests. The program also loads proxy settings from the environment, unless the `-p` option is also used.
`-P`	Specifies that proxy settings are not loaded from the environment.
`-t` *timeout*	Sets the time-out value; that is, the amount of time to wait for a response from the remote server before failing. The default unit for the time-out value is seconds. You may append "`m`" or "`h`" to the time-out value to make it minutes or hours, respectively. The default time-out is 180 seconds (3 minutes).
`-i` *time*	Sets the `If-Modified-since` header in the request. If *time* is the name of a file, use the modification timestamp for this file. If *time* is not a file, it is parsed as a literal date. For recognized date formats, refer to the `HTTP:DATE` manpage.

`-H` *header*	Sends an HTTP header with each request. A server can be specified. For example:
	`meta-admin`
	`-H 'referrer: http://other.url'`
	`-H 'Host:` *your_host*`' <` *ldif.input*

Table 16-2 describes the command-line options for controlling the output of the Meta-Admin Tool.

Table 16-2 Options that control the output for the Meta-Admin Tool

Output Options	Description
`-a`	Sets the output mode to ASCII text and outputs a blank line between responses. If this option is not supplied, response content is output in binary mode with nothing added.
`-u`	Prints the request method and absolute URL as requests are made.

Table 16-2 Options that control the output for the Meta-Admin Tool *(Continued)*

Output Options	Description
`-U`	Prints the request headers in addition to request method and absolute URL.
`-s`	Prints response status code. This option is always on for `HEAD` requests.
`-S`	Prints response status chain showing redirect and authorization requests handled by the library.
`-e`	Prints response headers. This option is always on for `HEAD` requests.
`-d`	Specifies that the content of a response not be printed.
`-o`	Processes HTML content in various ways before outputting it. If the content type of the response is not HTML, then this option has no effect. The legal format values are: `text`, `ps`, `links`, `html`, and `dump`. • `text` specfies that the HTML output be formatted as plain `latin1` text. • `ps` specifies that the HTML output be formatted as Postscript. • `links` outputs all links found in the HTML document. Relative links are expanded to absolute links. • `html` reformats the HTML code. • `dump` format dumps the HTML syntax tree.
`-v`	Prints the version number of the program and quits.
`-h`	Prints the usage message and quits.
`-x`	Provides extra debugging output.

Meta-Admin Tool Input

To administer the Meta-Directory components using the Meta-Admin Tool, you create an LDIF file which contains the commands for the administrative tasks you want to perform. You then invoke the Meta-Admin Tool, passing to it as input the LDIF file you've created. The tool reads the LDIF file and sends requests to the appropriate components as dictated by the request records in your file.

The LDIF file contains a sequence of attribute-value pairs. That is, each line of the input stream typically contains a parameter name, a colon, and the parameter's value. Each input record in the LDIF file describes a request for a single administrative task. Using LDIF format, you structure your administrative requests, where each request has attributes and values that conform to the Meta-Admin protocol.

NOTE You use the attributes and values defined in the Meta-Admin protocol to structure the request commands that you input into the Meta-Admin Tool.

For complete details on the Meta-Admin protocol and its use, refer to Appendix B, "The Meta-Admin Protocol."

The input file can contain a single request or it can contain a sequence of requests; however, any input following the first empty line is ignored.

You could use example script below to start a component, stop the component, read component statistics, and so on. The example has several administration request, each commented out with a hash mark in the original. To perform a desired task, just remove the hash marks next to the commands you want to perform and run the script through the Meta-Admin Tool.

Code Example 16-1 Sample Meta-Admin Tool input script

```
admin-URL: http://admin:admin@wrks4us.iplanet.com:5000
configuration: LDAP://config.admin-domain.com:5001/
  cn%3Djoin50-engine%2Ccn%3DMeta-Directories%2Ccn%3DSystem%2C
  ou%3D5%2Cou%3DMeta-Directory%2Cou%3DGlobal%20Preferences%2C
  ou%3Dadmin.domain%2Co%3DNetscapeRoot
authenticationDetails: simple "cn=Directory Manager" "iplanet"
#
#admin-path: join50-engine/Tasks/start
#
#admin-path: join50-engine/Tasks/stop
#
#admin-path: join50-engine/Tasks/read/status
#
#admin-path: join50-engine/Tasks/read/statistics
#
```

If this script were saved with the file name `admin_input.txt`, you could run the script with the Meta-Admin Tool using the following command:

```
meta-admin < admin_input.txt
```

Using the Meta-Admin protocol, you can create a custom client that you can use to perform a predetermined set of administration tasks. To note an example, the Meta-Directory Console is a client of this protocol.

Meta-Admin Tool Responses

After successfully processing a request, the Meta-Admin Tool will output any data returned by the request. While the Meta-Admin Tool might display progress indicators as it performs your requested tasks, a request which is successfully processed will not necessarily return any special success message.

The following text is an example of what you might see returned from a `statistics` request:

Code Example 16-2 Meta-Admin Tool output from a statistics request

```
NMC_Status: 0
NMC_ErrInfo: connecting ...
NMC_ErrInfo: communicating ...
NMC_Description: <3d95f080c204e3af489@join50-engine>
NMC_Description: LOGIN WRITE
NMC_Description: OK
NMC_Description: STATUS
         40 total number of changes
         20 total number of CV to MV changes
         20 total number of MV to CV changes
          5 total number of CV to MV adds
          0 total number of CV to MV add errors
         15 total number of CV to MV modifies
          0 total number of CV to MV modify errors
          0 total number of CV to MV deletes
          0 total number of CV to MV delete errors
          0 total number of MV to CV adds
          0 total number of MV to CV add errors
         20 total number of MV to CV modifies
          0 total number of MV to CV modify errors
          0 total number of MV to CV deletes
          0 total number of MV to CV delete errors
          1 total number of ConnectorViews
          1 total number of ConnectorViews running
          1 total number of MetaViews
          0 telnet-only mode
          1 current manager associations
         53 total number of manager associations
          0 total number of manager login authentication failures
```

Code Example 16-2 Meta-Admin Tool output from a statistics request *(Continued)*

```
  978976517 date/time of last startup (8 Jan 2001 09:55:17 -0000)
        259 secs since last Meta-Directory activity
NMC_Description: OK
NMC_Description: LOGOUT
```

It is possible for you to make a request that results in an error message being returned by the Meta-Admin Tool. For example, if the Meta-Directory CGI software is absent or damaged, if the tool does not support the request you make, or if the client is not permitted to perform the requested action, you will receive an error message in response to your request.

For details on the responses you receive, review Appendix B, “The Meta-Admin Protocol.” In particular, see the section titled “Protocol Responses.”

Managing Servers and Permissions

In order to access Directory Server or an Oracle database through the Meta-Directory interface, it needs to be configured as part of Meta-Directory's navigation tree. As well, access permissions need to be set or modified. This chapter contains the following sections:

- Configuring Data Servers
- Setting Access Permissions

Configuring Data Servers

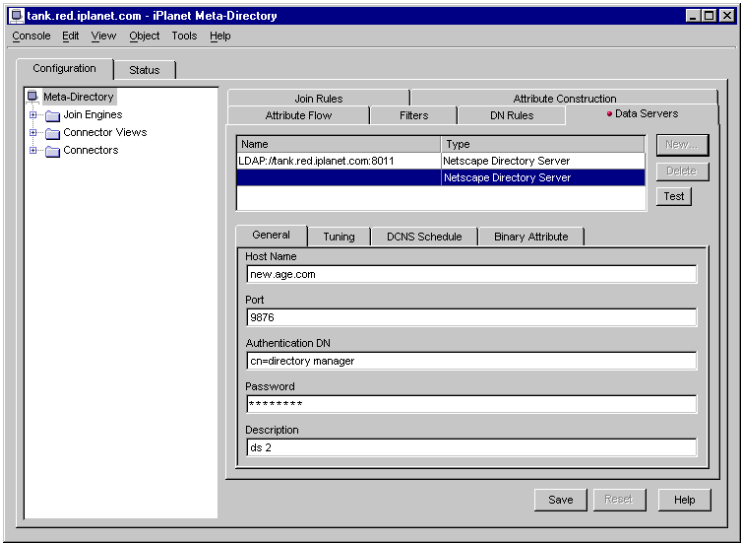
Meta-Directory has the ability to connect to a number of servers for a number of functions. These servers can be added as external data sources or to hold LDAP-configured connector views or meta views. Installing a Directory Server or an Oracle database as an addition to the Meta-Directory navigation tree is accomplished from the Data Servers panel. Typically, you would choose the server type and then configure it.

Accessing the Data Servers Window

1. From the Meta-Directory console, choose the Configuration tab.
2. Select Meta-Directory in the navigation tree.

3. Click the Data Servers tab.

   The Data Servers window appears.



To Add a New Data Server

1. In the Data Servers window, click New.

   The Data Server Type dialog box appears.

2. Select the type of server (Directory Server or Oracle) you want to add and click OK.

   The server appears in the list box and, by default, it is selected. Depending on the type of server being added, either four (Directory Server) or five (Oracle) additional tabs appear at the bottom of the panel. These tabs enable you to name and configure the new server.

NOTE If you choose Oracle in Step 2, go to Chapter 12, “Configuring the Database Connector” to continue configuring your server. If you choose Netscape Directory Server in Step 2, continue with Step 3.

3. On the General tab, provide values or change the defaults as needed for the following fields:

Host Name	Enter the fully-qualified name of the machine where the data source is stored, such as `Server.company22.com`. This is a required field.
Port	Enter the port on which the join engine connects to the host machine (normally port 389). This is a required field.
Authentication DN	Enter the DN of a user that has full administrator access to the Directory Server, such as `cn=directory manager`. This is a required field.
Password	Enter the administrator password. This is a required field.
Description	Enter a brief description to inform other users of the purpose of this data server. This is an optional field.

4. Click Save.

   The name of the new server appears under the Name column in the top window of the panel.

5. Select the new server, and click Test to test the connection.

   A Test Connect Succeeded message appears if the connection was successful. If the test was unsuccessful, make sure the connection information is correct.

6. From the Tuning tab, provide values or change the defaults as needed for the following operational fields:

Maximum Operation Result Time	Enter the maximum amount of time allowed before timing-out an LDAP search with no LDAP results. The suggested minimum value is 3600.
Maximum Number of Retries	Enter the maximum number of times you want the server to attempt to connect after an inital failure. A value of 0 indicates an infinite number of times. This field is associated with the Retry Intervals field.

Retry Intervals	Enter a comma-separated list of numbers, each representing the number of seconds to wait before the next retry should begin. For example, if you provided a value of 10 for Maximum Number of Retries, and 30,300,600,3600 for Retry Intervals, the system would respond as follows: *"If the LDAP server or database becomes unavailable, retry at most 10 times, beginning 30 seconds after the loss of connection is noted, then 5 minutes later, then 10 minutes later, then hourly. If the 10th retry fails, report an error."*
Idle Timeout	Enter the time, in seconds, that should pass before retries are abandoned if the server is idle.

7. Click the Data Change Notification System (DCNS) Schedule tab. This option allows you to schedule when the join engine will look for changes in the change log. Provide values or change the defaults for the following fields:

Field Name	Value
Second Specifier	Enter a value from 0 to 59.
Minute Specifier	Enter a value from 0 to 59.
Hour Specifier	Enter a value from 0 to 23.
Day Specifier	Enter a value from 1 to 31.
Month Specifier	Enter a value from 1 to 12.
Day of the Week Specifier	Enter a value from 0 to 6, where 0 is Sunday and 6 is Saturday.
Maximum Entries	Enter the maximum number of records to be read from the changelog for each cycle.

You can use either a single number as just described or an expression as follows:

Expression	Definition
*	Matches any value.
*/step	Matches any value in steps. For example, */2 matches 0,2,4,6... up to the maximum allowed value for values that start with zero, or it matches 1,3,5,7... up to the maximum value allowed.
x-y	Specifies a range where: • Both x and y are greater than or equal to the minimum allowed value. • y is less than or equal to the maximum allowed value. • x is less than y. The expression matches any value in the range.
x-y/step	Specifies a range as above, but with a step value that is not necessarily 1.
x	Specifies a single number within the allowed range.
x/step	Matches any value starting at x and then at x + step, x + 2*step, and so forth.
x,y,z...	Specifies a comma-separated list of values.
x-y,z-q	Specifies a comma-separated list of ranges.

- Lists can also contain both single values and ranges such as 1,2,5-7,10/5.
- The scheduler operates once every second, so the finest granularity occurs every second.

8. On the Binary Attribute window, check the binary list and, if necessary, make changes to specify which binaries you want the join engine to recognize. These are the changes you can make:
   - To create a new attribute, click New. A blank field appears at the bottom of the drop-down list. Type in the name of the attribute in this field.
   - To edit an attribute, select the attribute from the drop-down list and type over the name.
   - To delete an attribute, select the attribute from the drop-down list and click Delete.
9. Click Save to save the configuration.

To Test a Data Server Connection

1. In the Data Servers window, select the data server whose connection you want to test.
2. Click Test.

   If the connection was successful, a message confirming this appears. If the connection was unsuccessful, check your server's host information and re-test.

To Delete a Data Server

1. In the Data Servers window, select the data server you want to delete.
2. Click Delete.

   The data server and its associated configuration disappear from the list box.

Setting Access Permissions

From iPlanet Console, access permissions can be set for individual servers.

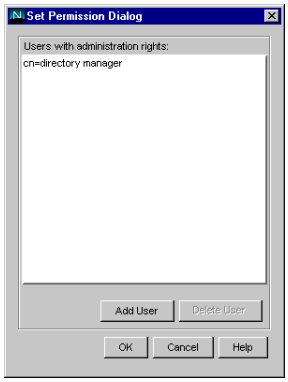
To Set Access Permissions

1. From iPlanet Console, select a Meta-Directory component or server, and right-click.

   A context menu appears.

2. Choose Set Access Permissions.

   The Set Permission Dialog appears with a list of the names of users and groups who currently have access permissions for the selected object. By default, the Configuration Administrators group has unrestricted access to all servers (but not to user data), although its name does not appear on this list.



3. Edit access permission as needed:

   - To allow access permission to additional users or groups, select the user or group name, then click Add User.
   - To deny access permission to a user or group in the list, select the user or group name, then click Delete User.

4. When you have finished adding and deleting users, click OK.

Part 5

Appendices

Appendix A, “The Join Process Operators”

Appendix B, “The Meta-Admin Protocol”

Appendix C, “Troubleshooting Meta-Directory”

Appendix D, “Error Messages”

Appendix A

The Join Process Operators

This appendix defines the syntax used when configuring the selection criteria for join processes, the method with which data is connected between connector and meta views. These processes include join rules, Distinguished Name (DN) mapping rules, constructed attribute definitions, and attribute flow rules.

In configuring these processes, something to note is that there are operators the data must pass through before being accepted in the target entry. First, they pass through the Format Operators where sub-components of the attributes may be created. Then the selection criteria fields are checked via the Requirements Operators. Once requirements are passed, the data is filtered through the Substitution Operators. This appendix contains discussions on the following:

- Format Operators
- Requirements Operators
- Combining Requirements Operators
- Generic Operators
- Quoting

Format Operators

The *format operator* is an optional field which allows an attribute component from the data source to be divided into sub-tokens (special sequences of characters) as it begins the join process. Generally, the original format of the component (consisting of "parameter=value") is sufficient to use as input to the process. However, occasionally, it is desirable to divide it into sub-tokens, dependant upon the configuration of the external data source. For example, you can format an e-mail address attribute such as `given.surname@company-name.com` into separate given-name, surname, and company-name sub-tokens.

Format operators are configured as:

```
{attribute} = %sub-token%,%sub-token%,%sub-token%
```

The schema attribute must be surrounded by curly-brackets (“{“ and “}”) and tokens must be surrounded by percent characters (“%”). An example would be:

```
{description}=%given%,%surname%,%company-name%
```

CAUTION Please note the formatting of the token list. Each sub-token must be separated by a comma with no other spacing. Deviation from this format will result in failure of the process.

Requirements Operators

The *requirements operator* determines whether the data source and/or the target entry meet the conditions set forth by the selection criteria. (Information on how to configure the selection criteria can be found in “Creating the Join Engine Instance,” on page 64”.) If either of them does not meet the required conditions, the entry will not sync and the join process for that particular entry will fail.

Requirements operators specify token conditions in addition to those mentioned under Format Operators. They are as follows:

- Character arrays start at position 1, not 0.
- Word and character operators can be combined. For example, `'%given%{2}[1]'` identifies the first character of the second word of the given name token.

- Where sets of characters are specified, you can specify a range of values with a dash separator sequence (“-”). For example, a-z means all characters beginning ‘a’ through ‘z’. Also, the terminating character must be the same type as the initial character. For example, ‘a-9’ has no meaning. (If the dash separator is to be used, you must format it with an initial # construction function, i.e.: #a-z.)

Although selection criteria can be configured automatically through the graphical user interface of the Meta-Directory > Configuration window, provided below is a listing of the operators that lie beneath the gui.

Table A-1 Presence functions

Operator	Description	Example
!	Token is empty	%c%!
=@	Token is present	%s%=@

Table A-2 Token length functions

Operator	Description	Example
#=	Token length equals	%s%#=10
#>=	Token length is greater than or equals	%g%#>=2
#>	Token length is greater than	%i%#>1
#<=	Token length is less than or equals	%cn%#<=15
#<	Token length is less than	%i%#<2

Table A-3 Word count functions

Operator	Description	Example
$=	Token word count equals	%cn%$=2
$>=	Token word count is greater than or equal	%s%$>=2
$>	Token word count is greater than	%g%$>2
$<=	Token word count is less than or equals	%cn%$<=2
$<	Token word count is less than	%g%$<2

Table A-4 Token comparison functions

Operator	Description	Example
+	Token begins with string	%cn%+id
-	Token ends with string	%g%-xyz
==	Token matches exactly	%cn%==mailer-daemon
!=	Token does not match	%surname%!=smith
^	Token contains string	%s%^administrator
~	Token does not contain string	%cn%~id

Table A-5 Token construction functions

Operator	Description	Example
.^	Token contains characters	%g%.^z
.~	Token does not contain characters	%s%.~1-9
.#	Token is composed of	%s%.#a-zA-Z
{i}	Token word at index ‘i’	%g%{2}#>=2
[i]	Token character at index ‘i’	%g%[1]==a
[i,c]	Token char range at ‘i’, take ‘c’	%g%[1,4]==reat

Table A-6 Examples of combining requirement operators

Logical Operator	Example
AND	((%given%#>=2)AND(%last%==smith))
OR	((%last%==smith) OR ((%last%==jones))
XOR	((%last%==smith) XOR (%last%==jones)) AND (%given==Tom))
NOT (!)	(!((%last%+Mc) OR (%last%+Oí)))

Combining Requirements Operators

You can combine multiple requirements operators into a single operator by listing them in a semicolon-separated, multi-valued list. Multiple requirements are treated as logical ANDs; that is, all requirements must evaluate to TRUE.

You can specify additional logical operators within individual requirement strings. Each sub-requirement must be encapsulated within parenthesis, and there is no limit to the level of nesting or the number of operators that you can specify. Evaluation of logical operators will be equivalent to the ansi-C precedence and evaluation mechanisms.

Table A-7 Examples of combining requirement operators

Logical Operator	Example
AND	((%given%#>=2)AND(%last%==smith))
OR	((%last%==smith) OR ((%last%==jones))
XOR	((%last%==smith) XOR (%last%==jones)) AND (%given==Tom))
NOT (!)	(!((%last%+Mc) OR (%last%+Oí)))

Substitution Operators

The *substitution operator* builds a search string which looks for entries that match the query in the target view. Substitution rules, using a combination of field codes and token identifiers, produce the resulting string, taking into account criteria from the previous operators. If no matches are found, the join rule fails.

Word, character, and substring operators as defined in Table A-8 are applicable to substitution strings.

Table A-8 Word, character, and substring operators

Operator	Description	Example
{i}	Token word at index ‘i’	%g%{2}#>=2
[i]	Token character at index ‘i’	%g%[1]==a
[i,c]	Token char range at ‘i’, take ‘c’	%g%[1,4]==reat

Generic Operators

Besides the strings described in the sections above, you can apply the following operators to any token value.

Table A-9 Generic operators

Operator	Description	Example
<U>	Uppercase string	%g%<U>
<L>	Lowercase string	%g%<L>
<F>	Friendly string	%g%<F>
<A>	Translate to 7-bit US ASCII	%g%<A>.%s%<A>

Specific syntax guidelines are as follows:

- Combining two operators

  To use two operators together, you need to group them within one set of angle brackets. For instance, to use the <L> and <A> operators together, you need to group them as <LA>, not <L><A>.

- Friendly string

  Friendly string means to capitalize initial and all other characters preceded by a linear white-space character. For instance, `'john joe l.'` becomes `'John Joe L.'`.

- 7-bit ASCII

  Translating to 7-bit ASCII is useful when downgrading from 8-bit and 16-bit character sets to 7-bit RFC-822 addresses. Downgrading rules are not configurable, but will apply intelligence when selecting equivalent 7-bit ASCII characters, the default downgraded character being '-'.

  For example:

  ```
  {L}=%g%.%s%
  ```

  ```
  {L}=%g[1,2].%s[L]
  ```

  ```
  {CN}=%g[F] %i[F] %s[F]
  ```

Quoting

Quoting characters control the parsing and evaluation of a string. Quoting any character with a backslash (\) removes the possibility of programmatic interpretation from certain delimiter and token constructor characters. Some of these characters are shown in Table A-10.

Table A-10 Quoting characters

Character Type	Character
Delimiter	;
Delimiter	%
Token constructor	.
Token constructor	[

Appendix B

The Meta-Admin Protocol

The Meta-Admin protocol offers a repertoire of request commands that you can use to administer a Meta-Directory component, such as the join engine or any instantiated connectors.

Clients that use the Meta-Admin protocol (such as the Meta-Admin Tool) communicate with Meta-Directory components by sending requests through iPlanet Administration Server. Requests to the Administrative Server are formatted as HTTP or HTTPS `POST` requests. In response to each request, the Administrative Server calls a CGI program to carry out the task. A set of CGI programs support each installed Meta-Directory component. The CGI programs can be located using the following directory structure:

NETSITE_ROOT`/bin/`*mdComponent*`/admin/bin`

Here, *mdComponent* is a directory housing an individual Meta-Directory component. The CGI programs located in these directories are Perl scripts that are interpreted by the copy of `nsperl.exe` that's located in the same `admin/bin` directory.

CAUTION Do not directly invoke the `create` and `remove` scripts provided by the protocol; these requests must be used in conjunction with other system calls built into the Meta Directory console. Using these scripts directly will put the Meta-Directory system into an unstable state from which the Meta-Directory console cannot recover.

This appendix describes the protocol itself; for a description on how to use the protocol with the Meta-Admin Tool, see Chapter 16, "Command-Line Administration."

The Protocol Requests

Each request in the protocol makes use of a set of attributes that supply the details needed to carry out the administrative task. You describe each administrative request by supplying the necessary attribute values needed by the request. All character values in the protocol are represented in UTF-8 format.

Attributes can be grouped into two categories:

- **Session attributes** set up the communication between the Administration Server and the component that you are administering with the request.
- **Task attributes** contain the details of the request which you are making. These are the attributes that you use to tailor your administrative requests.

Before you can make a request, you must set up the request by supplying the configuration and authentication information for the component you are administering. You do this using the three *session attributes*: `admin-URL`, `Configuration`, and `authenticationDetails`. These attributes define the information needed to establish communication with the Meta-Directory component, and must be defined for each request you make.

Once you supply the session information, you can specify the details of your individual administrative tasks using the *task attributes* defined in the Meta-Admin protocol. While some request do not require any task attributes, some take optional values, while others require that you supply a value (or set of values) in order for the request command to succeed.

Session Attributes

Before an administrative request can be carried out by a Meta-Directory component, you must specify the component you want to administer and the associated authentication details so you can access that component. Specifically, you must specify values for the following session attributes:

- Server URL
- Configuration Details
- Authentication Details

While these details do not need to be specified with every request, they must be specified, at minimum, in the first request. Once the component and authentication details are supplied, they become default values that subsequent requests can use. If needed, you can specify specific component and authentication values for any request you make; the last one ones specified always become the default.

The Server URL Attribute

Before any administrative tasks can be performed, you must set up the connection with the Administration Server that administers the component taking action. You specify the Administration Server using the `admin-URL` attribute.

The first record of the LDIF file must contain a valid `admin-URL` attribute. After it is specified, the URL attribute is optional for the remaining records in the input file; if the `admin-URL` attribute is omitted from a request, the URL of the preceding request is assumed.

You can use the following attributes in the record to specify individual parts of the URL:

- `admin-username` is used for basic authentication.
- `admin-password` is used for basic authentication.
- `admin-host` specifies the Administration Server's host name or IP address.
- `admin-port` specifies the Administration Server's TCP port number.
- `admin-path` specifies the part of the URL following *host*:*port*/.

The Configuration and Authentication Attributes

You must define the values of the `configuration` and `authenticationDetails` attributes for each LDIF request you make. While these two attributes do not constitute an administrative task by themselves, they contain information needed by the component before it will respond to an administrative request.

configuration - The configuration attribute identifies the LDAP URL of the Meta-Directory component's own configuration entry (under `cn=System`). This attribute takes a single required value.

For example, consider the following `configuration` value:

```
configuration: dn: cn=join-engine, cn=Meta-Directories, cn=System,
  ou=5, ou=Meta-Directory, ou=Global Preferences, ou=iplanet.com,
  o=NetscapeRoot
```

Here, the first `cn` value is the name of the component and the second value is the type of the component (`cn=connectors`). The third, fourth, fifth, sixth, and eighth values of this `dn` are all hardcoded for this release and you must not modify them from this example. The seventh value is the admin domain, which you must supply.

authenticationDetails - The protocol requires an `authenticationDetails` value to access the LDAP server identified by the `configuration` attribute. At minimum, the request should provide a copy of the `mdsAuthenticationDetails` attribute of the configuration Data Server.

This attribute takes one or more values, with each value having the same format as an `mdsAuthenticationDetails` attribute value, as follows:

```
mdsAuthenticationDetails: simple ["username" ["password"]] |
  simple_base64 [base64(username) [base64(password)]]
```

For example:

```
mdsAuthenticationDetails: simple "cn=Manager" "secret12"
mdsAuthenticationDetails: simple_base64 asdlALKJj8== 8AkBcUwHlMM=
```

A zero-length value signifies the absence of an authentication attribute.

Meta-Admin Protocol Requests

Meta-Directory version 5.0 supports the following Meta-Admin protocol requests through the Meta-Admin Tool:

- `start`
- `stop`
- `status`
- `statistics`

The reference section below contains descriptions of the attributes that you need to supply for each task in the protocol. In addition, typical responses are shown are shown for each request.

For an example of how to use these request commands with the Meta-Admin Tool, refer to Chapter 16, “Command-Line Administration.”

start

This request starts the server specified in the `URL` attribute.

Attributes

This request does not require any attributes.

Responses

This request can generate the following responses:

Response	Description
`NMC_Status: 0`	Success response.
`NMC_Status: 1` `NMC_ErrInfo: SCM status NO_SUCH_SERVICE`	Failure response.
`NMC_Status: 2` `NMC_ErrInfo: already started`	The client should assume the agent is started, unless the response indicates a failure.

stop

This request stops the server specified in the `URL` attribute.

Attributes

This request does not require any attributes.

Responses

This request can generate the following responses:

Response	Description
`NMC_Status: 0`	Success response.
`NMC_Status: 2` `NMC_ErrInfo: already stopped`	The client should assume the agent is stopped, unless the response indicates a failure.

read/status

This request informs you whether or not a particular server is up or down. Specify the server in question using the `URL` attribute.

Attributes

This request does not require any attributes.

Responses

This request can generate the following responses:

Response	Description
`NMC_Status: 0` `status: up`	The server is up.
`NMC_Status: 0` `status: down`	The server is down.
`NMC_Status: 1` `NMC_ErrInfo: errno ENOMEM`	Status is unknown, if the request fails.

read/statistics

This request reads and returns a component's status and statistics values. Specify the component using the `URL` attribute.

On Solaris systems, this request might take some time before returning.

Attributes

This request does not require any attributes.

Responses

This request can generate the following responses:

Response	Description

`NMC_Status: 0`
`status: up`
`Total Changes: 2478`
`Total Changes to MV: 253`
`Total Changes to CVs: 996`
`...`
	Successful response.

`NMC_Status: 1` `NMC_ErrInfo: errno ENOMEM`	Status and statistics are unknown, if the request fails.

Protocol Responses

After processing your administrative task, the CGI program will issue a response through an HTTP text document. The client of the protocol (such as the Meta-Admin Tool) can interpret these responses, and output an associated message.

Technically, the client of the protocol should assume a request was successful, even if the if the response only indicates success in the HTTP header (and contains no document) or if it returns a document that contains an `NMC_Status` value other than `0`. If the request returns data, it will be returned in the form of attribute name-value pairs, which have the following format:

name`:` *value*`\n`

While certain requests might return progress indicators, not all requests will do so. Although a request might be quiet, you should assume that your request is being processed until you receive a complete response.

Error Responses

If your request causes an error (for example, if the request has an invalid `configure` parameter), the response will report the failure or warning using the Mission Control `NMC_` name-value pair format, as follows:

`NMC_`*name*`:` *value*

For example, if you send a request to the Meta-Admin Tool whose attributes are not part of the Meta-Admin protocol, the tool could respond with the following:

```
NMC_Status: 1
NMC_ErrType: INCORRECT_USAGE
NMC_ErrInfo: unknown attribute <attribute_name_1>
NMC_ErrInfo: unknown attribute <attribute_name_2>
NMC_ErrInfo: unknown attribute <attribute_name_3>
```

The protocol has a predefined a set of indicators which it can use when sending back the results of your administrative tasks. These are defined as follows:

NMC_Status: Indicates the return status of the administrative CGI. The values returned can be any one of the following:

```
0: Success
1: Failure
2: Warning
3: Unknown
```

NMC_ErrType: The text describes the error type. The values returned can be any one of the following:

```
FILE_ERROR
MEMORY_ERROR
SYSTEM_ERROR
INCORRECT_USAGE
ELEM_MISSING
REGISTRY_DATABASE_ERROR
NETWORK_ERROR
GENERAL_FAILURE
WARNING
APP_ERROR
```

NMC_ErrInfo: Contains a brief text description of the error.

NMC_ErrDetail: Contains a detailed text description of the error.

NMC_Description: Contains a brief text description of the result. This result code can be used when `NMC_Status` equals `0` or `3`.

Appendix C

Troubleshooting Meta-Directory

This appendix describes troubleshooting techniques for Meta-Directory. In addition, it addresses some common configuration problems and how they can be fixed. This chapter contains the following sections:

- Verifying End-to-End Flow
- Viewing Link Information
- Checking Operational Status
- Examining Log Files
- Turning Off UID Uniqueness
- Fixing Join Problems
- Fixing Data Flow Problems
- Fixing Large Synchronization Failures
- Common Problems

Verifying End-to-End Flow

To determine whether a Meta-Directory event occurred as expected, use the Meta-Directory console to systematically verify the attribute flow from the meta view to the connector view and then to the external data source. If the window is not available to verify that an entry has been modified, added, or deleted, invoke the related view in Netscape Communicator by typing the entry's LDAP URL in the location window and pressing `Return`. You can also use any LDAP viewer application.

Viewing Link Information

The attributes of an entry indicate whether the Meta-Directory link information between the connector view and the meta view was added correctly. Note that the join engine adds several attributes to each entry that contain useful debugging information.

mdsEntityOwner

You can use the `mdsEntityOwner` attribute for an entry to determine whether the connector view is properly configured, including which view owns the entry. For example:

```
mdsEntityOwner: MV, MYMV
```

If you cannot delete an entry from a connector view, it may be because the meta view owns the entry. Only a view that owns an entry can delete the entry.

mdsLinkToCV

Viewing this attribute is useful in determining where the entry is linked and to what it is linked. It points to a meta view entry's corresponding entry in a connector view. For example:

```
mdsLinktoCV: MYMV#HRCV#cn=thowes
```

Multiple values for this attribute are possible, one for each connector view to which the entry is linked.

mdsLinkToMV

Viewing this attribute is useful in determining where the entry is linked and to what it is linked. It points to a connector view entry's corresponding entry in the meta view. For example:

```
mdsLinktoMV: MYCV#HRMV#cn=thowes
```

Checking Operational Status

To make sure a connector view is working correctly, you can check the Status and Statistics windows in the Meta-Directory console. More information on these windows can be found in Chapter 14, "Monitoring Meta-Directory Components".

Status Window

When a problem occurs between the meta view and a connector view, check the status of the connector view in the Meta-Directory console. The Status window indicates whether the view is enabled, disabled, or performing a refresh operation. For flow problems occurring between the meta view and the connector view:

- If the connector view is enabled and remains enabled, the likely cause is a misconfigured join rule. You can check each of the rules for questionable syntax by using the Join Rule Tester (Configuration > Join Rules > Rule Tester). To find out exactly what failed, you can view the logs.
- If the connector view is enabled but becomes disabled, the likely cause is a bad DN mapping rule, attribute flow rule, or corrupted data. This causes a flow problem to occur between the meta view and the connector view.

Normally, a view is automatically disabled when the join engine encounters an error with one of the rules that would compromise data integrity.

Statistics Window

The Statistics window usually indicates whether or not errors are presently occurring. Here is a guide for what to look for in the Statistics window:

- When many errors appear during a refresh, the likely cause is a bad rule or rule set. Make sure you have added rules to the rule set. You can use the Join Rule Tester to check each of the rules or rule sets for incorrect syntax. You can also check the logs as described above under Status Window.
- When add or modify operations appear without errors, and then an error suddenly appears causing the counter to stop, this indicates that the view was disabled by corrupted data.

NOTE The Windows NT Synchronization Service only reports operational status and not statistics.

Data Server Test Connect Button

This button is a useful tool when you want to check if the configuration for your data server is correct. This button provides feedback on your configuration.

Examining Log Files

You can track the flow of information between connector and meta view by examining the logs.

Most problems with flow involve a failure in the configuration of flow rules. Join engine logs are helpful in determining the cause of the problem if, for example, the join engine couldn't contact a server or if a bind operation failed.

The join engine log reports the transactions between the meta view and connector views. Each transaction is time-stamped and includes the relationship between the meta view and the connector view and a message. When a new entry is added, the join engine detects the changes and initiates action using all the rules defined for the enabled connector views. The join engine evaluates rules in the following order:

- Filter configuration rules
- Join rules

- DN mapping rules
- Attribute flow rules (including constructed attributes)

To see how each rule executed, you can increase the logging level of data access to 3 (Configuration > join-engine > Log > Verbosity). This should enable you to see exactly which rule failed and why it failed, thereby assisting you in correcting your rules grammar. However, for the case of a faulty constructed attribute, the log does not indicate the problem directly. If a constructed attribute fails, the information at the highest logging level (3) only states that the mapping failed for the rule.

For more information about monitoring log files, refer to Chapter 14, “Monitoring Meta-Directory Components.”

Turning Off UID Uniqueness

UID uniqueness is one of the constraint violations that can occur, and is the most common one with Directory Server 4.x. If you have multiple views on one Directory Server, and the entries are using UID, you could have two entries in two different views with the same UID. To Meta-Directory, this is acceptable, but to a Directory Server with the UID plug-in turned on, this is an error.

By default, UID uniqueness is turned on for Directory Server 4.x, but turned off for Directory Server 5.0. For constraint violations in Directory Server 4.x, you should turn off UID uniqueness.

Use the following methodology to resolve a problem that might occur when your data has an attribute UID.

1. Open the Directory Server through the Console.
2. Click on the Configuration tab.
3. Click on the topology tree at Plugins.
4. Scroll down to “uid uniqueness” and click to open.
5. Uncheck the Enable plugin box.
6. Restart the Server.

Fixing Join Problems

If you are experiencing unexplained incorrect joining, make sure that data is not flowing directly from the connector into the meta view.

If the join engine won't install, the most common reason is that the Directory Server is improperly configured. During installation, the join engine must point to a Directory Server instance that contains the configuration directory `o=NetscapeRoot`.

The following error messages, when received during join engine configuration, indicate a Directory Server configuration problem:

`DA_E_INVALID_CREDENTIALS`	The user and/or password are set incorrectly.
`DA_E_NO_SUCH_OBJECT`	The `o=NetscapeRoot` suffix is not set in the Directory Server, or the entry does not exist.
`DA_E_OBJECT_CLASS_VIOLATION`	The Meta-Directory schema was not properly loaded into the Directory Server.
`MDS_CONTROLLER_UNAVAILABLE`	You have not created the 4.x changelog (retro changelog plug-in 5.0), or you have created, configured, and enabled it through instance creation, but have not restarted the Directory Server. Check that the path and suffix/subtree you provided for the changelog is valid.

Fixing Data Flow Problems

Use the following methodology to resolve problems that you might encounter with data flow.

- View the data source and make sure that data exists. To check that the data hasn't already synchronized, use refresh the view to see updates.
- Verify end-to-end attribute flow.
- Check operational status. Is the view enabled? What statistics are being reported for the view?
- Check statistics.
- Make sure the join engine and the connectors are running.

- For indirect connectors where no data appears to synchronize to the connector view, check the scheduled time the data is configured to synchronize. Wait until that time has elapsed, then schedule the task to run more frequently.
- Examine the relevant log files. The join engine log keeps track of errors resulting from rules that aren't configured properly. The Directory Server logs show objectclass violations, constraint violations, or other LDAP problems. You can also increase the log level to check rules (Configuration > join-engine > Log > Verbosity).

Fixing Large Synchronization Failures

Large synchronizations can fail on very slow systems or networks. You can initiate the debug process by increasing the log level to 3 (Configuration > join-engine > Log > Verbosity) if you see the message `USER_TIMELIMIT_EXCEEDED`. You can also go to Configuration > Meta-Directory > Data Servers > Tuning and increase the value in the Maximum Operation Result Time field.

Common Problems

This section discusses common configuration problems and their possible solutions. Tabular information is presented for problems associated with the following components:

- Instances
- Configuration
- Views
- Connectors
- Meta-Directory
- Directory Server

Instances

lists troubleshooting tips for problems you might encounter while creating instances.

Table C-1 Problems with Instance Creation

Problem	Suggested Action
Schema / objectclass violations occur when you try to save configurations through the console.	Check whether you loaded the schema into all directory servers in your system, including the directory server containing the configuration.
The join engine failed, and the log shows `MDS_CONTROLLER_NOT_AVAILABLE`.	Check that you did the following: Enabled the changelogs for all Directory Servers in your system, including the configuration Directory Server. Restarted the Directory Server after enabling the changelog. Provided a valid changelog path for the Directory Server changelog enabling dialog. The path given should have the .dbb files in it, and you should be able to search the changelog suffix. (The default is cn=changelog.) Provided a valid, unique, non-preexisting, well-formed subtree / suffix for your changelog. Can you search for it?
The join engine failed, and the log shows `MDS_CONTROLLER_NOT_AVAILABLE`, but you enabled the Directory Server 5.0 changelog through the Directory Server console.	If you are using Directory Server 5.0 and you enabled the changelog through the Directory Server console (and not Meta-Directory instance creation), make sure to enable the retro changelog, rather than the normal replication changelog.
Some changes never flow.	Is your Directory Server culling changelog entries before the join engine can process them? The changelog cleanup has to be set to infinity, or large enough to not delete changes before the join engine examines them.
The Join Engine or connector instance isn't responding to commands.	Allow the join engine or connector sufficient time to start.

Table C-1 Problems with Instance Creation *(Continued)*

Problem	Suggested Action
The instance creation or view creation failed, and the log shows `NO_SUCH_OBJECT`.	During instance or view creation, did you specify a view location under an existing suffix that has no data nodes under the suffix? For example, the following scenario would fail: Use the Directory Server console and specify `o=MV` suffix, then during instance creation specify `ou=MV1,ou=MvEntries,o=MV` as the base dn. To make this work, you would need to add the entries `o=MV` and `ou=MvEntries`.

Configuration

The following tables list troubleshooting tips for problems you might encounter after configuring iPlanet Meta-Directory.

Table C-2 General Rule Problems

Problem	Suggested Action
Corrupted data, missing values, or objectclass violations are leading to no flow.	Are all your rules being applied in the correct direction? The rules usually will only make sense going one way (meta view to connector view, for example), but not the other way, so if you didn't assign them to the correct direction, nothing will function.
You are not seeing any flow, or your joins are failing.	Make sure all your rules are syntactically correct and free of typos.
Your multi-valued attributes are not flowing correctly.	iPlanet Meta-Directory 5.0 doesn't handle multi-valued attributes very effectively.
The join engine isn't using your rules.	Did you wait long enough for the join engine to re-read the configuration? If the join engine is busy, it might not read the configuration for some time, since it relies on the DCNS queue, and there could be a lot of data changes ahead of your config changes.

Table C-2 General Rule Problems *(Continued)*

Problem	Suggested Action
You added a rule and it didn't show up in other parts of the console.	Did you try refreshing the console after adding your new rule that didn't show up in other parts of the interface? When you invoke a console refresh, it re-reads everything from the LDAP configuration. As a last resort, restart the console.
There are problems with the data flow.	Does your Join Rules tester produce one entry, no entries, or more than one entry? It should produce exactly one entry. More than one entry means that your rule isn't based on a unique attribute.

Table C-3 Problems with Join Rules

Problem	Suggested Action
You are having problems with corrupted data.	Are the optional token assignments for your join rules correct for all entries that the rule applies to? Your token assignments are possibly failing for certain entries. Is the selection criteria for your join rules selecting the correct entries? Your selection criteria is possibly identifying the wrong entries or missing some, etc. Turn the log level for data access to 3 (Configuration > join-engine > Log > Verbosity) and resync or refresh. Look at the more detailed logs for clues as to how the rules were evaluated.
Some entries are not joining.	Is the join filter for your join rules searching on something that uniquely identifies one entry? If the search filter comes up with zero or more than one entry, that entry is skipped.
Nothing is flowing.	The join filter for your join rules must produce a valid LDAP search filter that conforms to the standard and be accepted by the Directory Server. Check the access logs of the Directory Server.

Table C-3 Problems with Join Rules *(Continued)*

Problem	Suggested Action
You are experiencing unexpected joining.	Your join rules must be in the right order. The rules in a rule set are evaluated in order.

Table C-4 Problems with DN Mapping Rules

Problem	Suggested Action
Some entries have mangled DNs when they are created.	Are the optional token assignments for your DN mapping rules correct for all entries that the rule applies to? Your token assignments are possibly failing for certain entries. Turn the log level for data access to 3 (Configuration > join-engine > Log > Verbosity) and resync or refresh. Look at the more detailed logs for clues as to how the rules were evaluated.
You are having problems with mislaid DNs.	Check that your DN Mapping Rules selection criteria are selecting the correct entries. Your selection criteria are possibly identifying the wrong entries or missing some.
You have an entry with an empty or mangled DN (cn=,ou=,o=).	Check that your DN construction is based on a valid, existing attribute for all entries to which this rule applies. Problematic behavior occurs if the DN is based on an attribute that doesn't exist.
An entry is not being added because of an `ALREADY_EXISTS` error.	Check that your DN construction is based on a unique attribute for all entries to which this rule applies. You've created a problem if two entries have the same attribute value upon which the DN is formed. Either your DN is not creating a unique DN and you should redesign it, or you actually want to join or link these two entries. In this case, you're using the wrong type of rule. You're missing a join rule which would do this join or link.

Table C-4 Problems with DN Mapping Rules *(Continued)*

Problem	Suggested Action
Entries are not being added.	Check that your DN construction produces a valid DN that conforms to the DN standard and that it is accepted by the Directory Server.
Entries are not being added in the right place, or are not being added at all.	The DN shouldn't be a complete DN. It needs to be relative to the view's base DN.

Table C-5 Problems with Attribute Flow Rules

Problem	Suggested Action
There is no flow because attribute values are not present, or there is incorrect attribute mapping.	Check that your Attribute Flow selection criteria are selecting the correct entries. Possibly the selection criteria are identifying the wrong entries or missing some, or there is a typo and you created the wrong mapping between the source view and the target view. Another reason may be that you are trying to map to an attribute that is the wrong data type for your source data.
There are missing attributes or no flow due to objectclass violations.	Check that your attribute flow is complete and that you haven't missed any attributes. What objectclasses are you trying to flow? Does your attribute flow produce objectclass compliant entries? Does your attribute flow produce LDAP-compliant entries? Do they have objectclasses? Check which attributes are mandatory for that schema, such as `cn` and `sn` for `inetorgperson`. Check the Directory Server error log for clues about problems with objectclass violations.
There is no flow in a given direction.	Check that your attribute flow rule is configured for the correct direction. Each rule has a specific direction associated with it. Open the rule through the interface and make sure that the direction is correct.

Table C-6 Problems with Constructed Attribute Rules

Problem	Suggested Action
There is no flow because of corrupted data or objectclass violations.	Are the optional token assignments for your constructed attributes correct for all entries that the rule applies to? Your token assignments are possibly failing for certain entries. Is the selection criteria for your constructed attributes selecting the correct entries? Your selection criteria is possibly identifying the wrong entries or missing some, etc. Turn the log level for data access to 3 (Configuration > join-engine > Log > Verbosity) and resync or refresh. Look at the more detailed logs for clues as to how the rules were evaluated.
There is no flow because of corrupted data, objectclass violations, or missing values.	Check that your attribute construction is based on valid, existing attributes with values. Check that you actually mapped your constructed attribute to something in attribute mapping.

Table C-7 Problems with Filter Rules

Problem	Suggested Action
There are missing entries.	Are the entries missing in the flow portion of an excluded subtree? If you've set up a subtree filter, any entries in the excluded subtrees won't flow. Turn the log level for data access to 3 (Configuration > join-engine > Log > Verbosity) and resync or refresh. Look at the more detailed logs for clues as to how the rules were evaluated.

Views

lists troubleshooting tips for problems you might encounter with connector views.

Table C-8 Problems with Views

Problem	Suggested Action
No data is flowing.	Make sure your view is enabled.
You have experienced unexpected or malfunctioning data flow.	Check that your LDAP-based views are in separate subtrees.
	Have you or an application directly modified the data in the connector view? Only MDS should be handling the data.
	Is the subtree of one view a descendant of another? The view subtrees cannot overlap.
No data is flowing.	Have you added the connector view from which you wish to flow as a participating view to the meta view? If the connector view isn't added as a participating view, nothing happens.
You've enabled your view, but no data is flowing and the view keeps getting disabled.	Check DN mapping and attribute flow for corrupt data, then check for faulty join rules.
You have experienced erroneous joining.	Make sure you are not sending data from the connector directly into the meta view.
Your rules aren't being used by the join engine.	Did you apply your rules to the participating view?
Using the MDS console, you were unable to see the contents of a view with many entries.	Did you create a browsing index in the Directory Server console for the suffix of the view with a lot of entries that you are trying to browse? The console can't handle too many entries at once, so it has to rely on the server to give it a Virtual List View, for which you need a special index. See your Directory Server documentation for more information.
No data is flowing, and you see `MDS_S_INSUFFICIENT_CAPABILITIES` in the join engine logs.	Is your participating view's capabilities set up so that changes/entries can flow in the desired directions?
You have experienced a mapping error and the connector view is constantly being disabled.	Is the root entry of your view flowing? You have to create selection criteria to exclude the objectclass of the root entry, or use a filter.

Connectors

The following tables list troubleshooting tips for problems you might encounter after configuring connectors.

Table C-9 Problems with the Universal Connector

Problem	Suggested Action
No data is flowing	Is your connector running?
The UTC became corrupted and you couldn't flow data.	Did you interfere with the UTC shutdown? Did you issue `kill -9` or otherwise stop it uncleanly? Try refreshing to the connector view.
You made changes that the UTC didn't flow, or you clicked on refresh and nothing happened.	Make sure to wait for the default 15-minute polling interval for the UTC to flow your changes. Everything the UTC does is based on the polling interval, including refresh. Consequently, you'll have to change the 15-minute default if you want results sooner.
You have pre-existing connector view entries that are not flowing to the foreign data source.	Try a UTC filter refresh, or "touch" every entry by making a small modification to it.
Some attributes aren't reaching the connector view from the FDS, or you are experiencing objectclass violations.	Is your UTC attribute mapping complete? Did you miss an attribute? In this direction, any attributes not listed in the attribute mapping are dropped
Some modifications do not flow bi-directionally.	Do you have a UTC attribute flow defined, thereby forcing attribute level flow? You probably have entry level flow and are modifying the non-owner.
Some attributes flow to the FDS that shouldn't be flowing, and they're being rejected.	Do you have a UTC attribute flow defined for the connector view to FDS that doesn't handle the attributes that are being rejected by the FDS? When going from the connector view to FDS, everything flows.
You are experiencing objectclass violations.	Do you have different object types and your default values / attribute mapping isn't taking this into account? If you have multiple object types, you should be careful with the UTC rules. You may want to create this material in Perl `script/task.cfg`.

Table C-9 Problems with the Universal Connector *(Continued)*

Problem	Suggested Action
The UTC-based connector isn't picking up your configuration changes.	Make sure that you restarted the connector after making the configuration changes (except logging).
Some attributes are being mapped incorrectly, dropped, or mangled.	Is your Perl attribute mapping compatible with your UTC attribute mapping? If you've set up Perl script attribute mapping (like `LineFormat` in `task.cfg`), you need to make sure your UTC mapping takes that into account.
Your UTC filters aren't working correctly.	Make sure your UTC filters are expressed in connector view DNs, regardless of the direction.
Your 5.0 UTC won't run your script.	Does your UTC perl script have a package name in it? There should not be a package name in the script

Table C-10 Problems with the Universal Text Parser

Problem	Suggested Action
The UTP doesn't seem to be working.	Make sure you copied all the required `.pm`'s, `template.pl` and the `*.cfg` files from *NETSITE_ROOT*`/bin/utc50/install/templates/universalparser`. Make sure you renamed one of the `.cfg` files to `task.cfg`. Did you forget to set the `template.pl` script path for the UTC? Does the script path to `template.pl` specified in the UTC UI appear in an `ls` or `dir` listing from the command line?
The UTP isn't working correctly.	The text editor you used to edit `task.cfg` should not remove trailing white spaces from the end of lines. Did you specify DN as your `IndexAttribute` in `task.cfg`? `IndexAttribute` must be a unique attribute, but not DN.

Table C-10 Problems with the Universal Text Parser *(Continued)*

Problem	Suggested Action
The UTP isn't working because of objectclass violations.	Make sure you specified your LineFormat correctly in `task.cfg`. Does the resulting entry conform to `inetOrgPerson` or whichever objectclass you've specified?
The UTP isn't flowing data.	Is `InputFile` set correctly in `task.cfg`? Make sure that `template.pl` can find the input file.
The UTP isn't flowing changes to the external data source.	Make sure you specified your `ImportLineFormat` correctly in `task.cfg`. Is the import function expecting a different format? Is the import utility accessible from `template.pl`? There may be a path problem or a permissions problem, or perhaps the utility doesn't exist or is not executable.

Table C-11 Problems with the NT Domain Connector

Problem	Suggested Action
No data is flowing	Is your connector running?
Changes aren't flowing to NT SAM.	Is the machine you specified as NT Domain host write a PDC?
Changes aren't flowing to or from NT SAM.	Check whether logs like `ntdc-ntacc-*.log` and `ntdc-perl-*.log` are in your NTDC instance. If no logs are present, the accessor never started. Check the host and domain names in `ntdc.conf`.
Writes to NT SAM are failing.	If you are running the NT Domain Connector on a machine other than the PDC, is it running under the Domain Administrator account?
Configuration changes you made to `ntdc.conf` haven't taken effect.	Did you restart the NT Domain Connector after changing `ntdc.conf`?

Table C-11 Problems with the NT Domain Connector *(Continued)*

Problem	Suggested Action
Some pre-existing entries in connector view for the NT Domain Connector aren't flowing to NT SAM.	Make sure you refreshed the meta view.
Entries are not flowing to NT SAM.	Make sure your join engine rules ensure the entry format that the NT Domain Connector is expecting.
Some changes to certain attributes made from the connector / meta view aren't flowing into NT SAM.	Are the attributes you changed on the list of attributes only changeable from NT SAM?
You are unable to modify some entries from the Admin Tool.	Are you flowing user names that are less than 20 characters in length?

Table C-12 Problems with the Database Connector

Problem	Suggested Action
No data is flowing	Is your connector running?
There are missing OCI errors in the database data server UI.	Make sure you installed OCI.
Changes won't flow to or from Oracle, or your database-based view keeps being disabled.	Did your DBA successfully run the SQL scripts against Oracle?
There are flow problems, or you see `DA_E_ALREADY_EXISTS` in the join engine logs.	For each table, make sure the columns you chose for nominated keys uniquely identify the row.
Oracle is rejecting your SQL scripts.	Did you replace any placeholders in the SQL instrumentation scripts with DBA-approved values.
Oracle is rejecting the SQL scripts, or the Database Connector is performing strangely.	Did you change anything in the SQL instrumentation scripts that were marked "Don't Change This"?
Oracle is rejecting the SQL scripts due to authentication problems or this error: "ORA-01031: insufficient privileges."	Did you or your DBA run the SQL instrumentation scripts against Oracle as the System Database Administrator User?

Table C-12 Problems with the Database Connector *(Continued)*

Problem	Suggested Action
Instrumentation scripts fail during the creation of the changetables, or you see this error: “ORA-00957: duplicate column name.”	For your data server, make sure you select column names for Action Column and SynchPoint Column that are unique across entire Oracle instance.
SQL scripts fail with “Cannot remove user.”	Was the Join Engine trying to synchronize Oracle when you ran the Delete Data Server SQL scripts? This is because the join engine is probably connected to Oracle. Remove all views based on the data server from the meta view's participating views list first, and then wait for the join engine to finish before re-running the SQL commands. As a last resort, stop the join engine.
Oracle flow is failing because of objectclass violations.	Did you make a constructed attribute for Objectclass for your DB-based views?
Oracle flow is malfunctioning because of `ALREADY_EXISTS` errors and missing entries.	Are your DN mapping rules for DB-based views based on the Primary or Nominated Key(s)? The DN must be based on the Primary Key / Nominated Key when flowing from the meta view to the connector view. This is also recommended when flowing from the connector view to the meta view.
All modify operations on the Oracle connector view fail.	Did you duplicate mappings between DN Mapping and Attribute Flow for the Database Connector? You should not duplicate mappings. If you’ve mapped `EMPNO` to `employeeNumber` in attribute flow and then made your DN mapping rule look like: `EMPNO=%mv.employeeNumber%` the flow from the meta view to the connector view will be broken.
There is no flow to or from Oracle.	Did you change the structure of any of your instrumented Oracle tables, such as removing a column? If you change the structure of the table, such as deleting an instrumented column, the Database Connector will cease to function. You will need to uninstrument and reinstrument.

Table C-12 Problems with the Database Connector *(Continued)*

Problem	Suggested Action
There is no flow to or from Oracle, or you notice that the changelog is broken (no changes are being logged), or you see trigger-related error messages in Oracle.	Did you run the same DBC SQL script twice? Ask this if the user tells you no flow to / from Oracle is happening or if they notice that the changelog is broken (no changes are being logged) or if they see trigger-related error messages in Oracle. If you run the same script twice, the triggers cease to function.

Meta-Directory

lists general troubleshooting tips for iPlanet Meta-Directory.

Table C-13 Problems with iPlanet Meta-Directory

Problem	Suggested Action
The message `DA_USER_TIMELIMIT_EXCEEDED` appeared during the join process.	Is your data server search time-out parameter high enough?
The join engine is sluggish, and there's increased memory usage.	Is your data server poll interval high enough?
You are experiencing poor performance.	Are your join engine and / or console and / or Directory Server all on the same machine?
You are experiencing several "can't connect" errors, or time-outs.	Are all the machines reachable from one another? Ping all of the machines to make sure they can communicate with each other.
You are experiencing poor flow.	Do you see many errors in the Statistics window, or just a sudden stop? Many errors means faulty rules, and a sudden stop means corrupt data.
There is no data flow.	Make sure that both the join engine and connector are running.

Directory Server

lists general troubleshooting tips for Directory Server.

Table C-14 Problems with Directory Server

Problem	Suggested Action
No data is flowing.	Is your Directory Server running?
The Directory Server times out during large synchronizations.	If you're using Directory Server 4.x, make sure your `ioblocktimeout` parameter in `slapd.conf` set to a high enough value (180000000 or higher).
You are experiencing poor performance, especially during large synchronizations.	Make sure your Directory Server is properly tuned for writes.
You are experiencing poor performance, especially during large synchronizations, and errors such as `DA_E_UNWILLING_TO_PERFORM`, `DA_E_USER_TIMELIMIT_EXCEEDED` might appear, or other time-out messages might appear.	Did you add the indexes for the linkage attributes (`mdslinktocv`, etc.)?
The Directory Server disallows the addition of entries because of conflicting UIDs. You will see constraint violation errors.	Did you turn off the UID uniqueness plug-in?
You are experiencing poor performance and have already added the proper indexes for the linkage attributes.	Did you increase the `allidsthreshold` of the Directory Server?
You are experiencing poor performance and hear disk thrashing on the machine hosting the Directory Server.	Make sure your database cache is big enough to hold all the indexes.
You are experiencing poor performance and hear disk thrashing on the machine hosting the Directory Server, or you see the Directory Server warning at start-up or at the beginning of a database import about your cachesize / dbcachsize being too large for physical memory.	Make sure your Directory Server caches completely fit into physical memory. Do you have a 3:1 ratio of database to entry cache allocation? Are your calculations correct for determining the cache size? Did you factor in 25% memory overhead when computing cache size? Your database cache should not be configured for more than 1.6 gigs.
There are missing entries during the join process and / or time-out errors.	Are your Directory Server front-end settings too low? Check the limit, time-out, and so forth.

Table C-14 Problems with Directory Server *(Continued)*

Problem	Suggested Action
You see the following message: `ldap_modify: No such attribute: additional info: attribute type is unknown. Cannot delete.`	While loading the Meta-Directory schema to a Directory Server instance that does not contain the join engine and meta view configuration information, ldapmodify attempted to delete an entry that does not exist. When ldapmodify applied the Meta-Directory schema, it attempted to delete the attribute type before a new one was added.

Appendix D

Error Messages

This appendix provides a listing of primary error messages for the join engine and Universal Text Connector, along with causes and solutions. The messages are generated to log files.

This appendix contains the following sections:

- Join Engine Messages
- Universal Connector Messages

Join Engine Messages

Table D-1 provides a list of join engine error messages arranged in alphabetical order. The messages are generated to logs found at the following location:

NETSITE_ROOT/join-engine/logs

Table D-1 Join Engine Error Messages

Message	Cause and Solution
DA_E_NO_SUCH_OBJECT	The object or entry was not found in the Directory Server.
DA_E_UNWILLING_TO_PERFORM	One reason this message might appear is that an LDAP version 3 modRDN operation failed. If you rename an entry and additionally move the entry under a new parent, the LDAP protocol command to do so must specify the "newsuperior" or the DN of the parent. Other possible reasons are: • The ldbm database plug-in did not load properly. • An unknown attribute(s) was added to the configuration. • The user's password expired. • An add, rename, or delete was applied against a DSE. • A 1.0-syntax ACI was added. • A modify was applied to a read-only DSE attribute. • You attempted to replace an objectclass or attribute definition in the schema DSE. • An objectclass definition was deleted, which is a superior definition for another objectclass. • A read-only objectclass or attribute was deleted. To correct the problem, see Directory Server Performance Tuning for Meta-Directory in the *iPlanet Meta-Directory Deployment Guide.*
DA_E_USER_TIMELIMIT_EXCEEDED	A search in the Directory Server has timed out. Go to Configuration > Meta-Directory > Data Servers > Tuning. Increase the value in the Maximum Operation Result Time field. If the time-out continues, change the values in the retry fields.

Table D-1 Join Engine Error Messages *(Continued)*

Message	Cause and Solution
DA_S_NO_RESULTS _RETURNED	A search was conducted and no results were returned for the search.
E_DCNS_TOO_LATE	Any message that appears with DCNS in it, including this one, is associated with the Directory Server changelog. For this message, the following events might have occurred: • You may have started the join engine without enabling the changelog. • You enabled the changelog and restarted the Directory Server, but did not subsequently start the join engine. • The changelog wrapped around so that the sync number stored in the iMDS configuration is greater than the last change number in the changelog, thereby resulting in a mismatch. For the first two cases, after enabling the changelog, restart the Directory Server, stop the join engine, restart the iPlanet Console, then start the join engine. For the last case, reset from the console the sync points stored in the iMDS configuration.
MDS_CONTROLLER _UNAVAILABLE	Either you did not configure the changelog, or you did not restart the Directory Server after configuring Meta-Directory.
MDS_S_ENTRY_ALREADY _EXISTS	The entry you are attempting to add already exists in the target view. You may need a join rule to join or link the entry.
MDS_S_INSUFFICIENT _PRIVILEGES	The configured join engine capabilities are prohibiting the requested operation from proceeding. Add more join engine capabilities for the operation you are attempting to perform.
TIMEOUT	This message usually appears with the DA_E_USER_TIMELIMIT_EXCEEDED message, and indicates that an operation, such as a refresh or search, has timed out. Increase the join engine time-out limit.

Universal Connector Messages

provides a list of Universal Connector error messages arranged in alphabetical order. The messages are generated to logs found at the following location:

NETSITE_ROOT/utc-*ViewName*/logs

Table D-2 Universal Connector Error Messages

Message	Cause and Solution
Delta modified Perl address record unapplicable, id 'utcuser1', DN 'cn=utcuser1,ou=CV2'	The entry 'cn=utcuser1,ou=CV2' was detected as unchanged. No action is required.
Directory delta determination postponed pending active delta processing	For flow control, the current synchronization cycle has been skipped because the delta queue has reached its limit. No action is required.
DraDua PerlDra LDAP List error 32 'No such object'	This is usually not an error. It is a generic message indicating that the LDAP search could not return the specified object. Check the log messages in the same area to determine which entry search is causing the 'No such object' error.
Local delta determination postponed pending active delta processing	For flow control, the current synchronization cycle has been skipped because the delta queue has reached its limit. No action is required.
Local delta determination postponed until sync time	The time has not yet arrived for synchronization. No action is required.
Task '1' invalid Perl record index 'xxx', cannot process	The index value the Perl script returned for the Perl record is invalid. Check your Perl script for errors.
Unable to initialize host system. Shutting down...	This message usually indicates that the Perl script you specified for the connector is invalid. Check the file path, and also check for syntax errors in your Perl script.

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W