Administrator's Guide

iPlanet Application Server Enterprise Connector for Tuxedo

Version 6.5

806-5509-02
August 2002

Contents

List of Figures

List of Tables

List of Procedures

List of Code Examples

Preface

The *iPlanet Application Server Enterprise Connector for Tuxedo* explains how to install, manage, and configure the iPlanet Application Server Enterprise Connector for Tuxedo.

This preface contains information about the following topics:

- Prerequisites
- Supported Platforms
- System Requirements
- What is in this Guide
- Documentation Conventions
- User Roles
- Online Guides
- Related Information
- Third Party Publications

Prerequisites

This guide is written for system administrators who want to install, manage and configure the iPlanet Application Server Enterprise Connector for Tuxedo

The guide assumes you are familiar with the following topics:

- iPlanet Application Server programming concepts
- The Internet and World Wide Web
- A general understanding of Windows NT or Windows2000 operating system
- A general understanding of Solaris operating system and commands
- System management knowledge of BEA Tuxedo

- Familiarity with BEA Tuxedo programming
- Java programming language

Supported Platforms

The iPlanet Application Server Enterprise Connector for Tuxedo is supported on the following platforms:

Vendor	Platform
Sun	SPARC running Solaris 2.6 and Solaris 2.8.
Microsoft	Windows NT SP5 or Windows 2000

System Requirements

To install the iPlanet Application Server Enterprise Connector for Tuxedo, your system must have the following:

- iPlanet Application Server Version 6.5
- iPlanet Unified Integration Framework Version 6.5
- iPlanet Webserver 6.0 SP2
- Tuxedo Workstation Client 7.1 with Rolling Patch.

What is in this Guide

This guide provides the information you need to understand, set up, and administer all aspects of the iPlanet Application Server Enterprise Connector for Tuxedo.

The following table contains a short summary of the contents of each chapter.

See this chapter:	If you want to do this:
Chapter 1, "Introduction"	Familiarize yourself with general concepts of the iPlanet Application Server Enterprise Connector for Tuxedo.
Chapter 2, "Installation"	Install the connector.
Chapter 3, "The Tuxedo Management Console"	Describes how to configure the Tuxedo Enterprise Connector for use with the BEA Tuxedo system. The concepts you should be familiar are described as well as, how to use The Tuxedo Management Console to import Tuxedo data types and services into the UIF Repository.
Appendix A, "Installed Components"	Provides a descriptive list of components that the installation procedure adds to your system.
Appendix B, "Sample Programs"	Provides a descriptive list of samples provided by iPlanet Application Server Enterprise Connector for Tuxedo.
Appendix C, "Workarounds"	Describes workaround that has been developed to alleviate certain problems that may occur, in particular excess CPU use of time..

Documentation Conventions

File and directory paths are given in Windows format with backslashes separating directory names. For Unix versions, the directory paths are the same, except slashes should be substituted in place of backslashes.

This guide uses URLs of the form:

http://*server.domain/path/file*.html

In these URLs, *server* is the name of the server on which you run your application; *domain* is your Internet domain name; *path* is the directory structure on the server; and *file* is an individual filename. Italics items in URLs are placeholders.

This guide uses the following font conventions:

- The `monospace` font is used for sample code and code listings, API and language elements (such as function names and class names), file names, pathnames, directory names, and HTML tags.

- *Italic* type is used for book titles, emphasis, variables and placeholders, and words used in the literal sense.

User Roles

This section describes the various user roles and their tasks. People with a variety of skills are involved with the setup of the iPlanet Application Server Enterprise Connector for Tuxedo. Some of these users are listed below:

System Administrator	This is the person responsible for the Tuxedo Enterprise Connector installation.
Systems or Business Analyst	This person uses the Tuxedo Management Console to configure the Tuxedo Enterprise Connector and imports the Tuxedo data types and services into the UIF Repository.
Applications Programmer	This person who writes Servlets, JavaServer Pages (JSPs) or Enterprise Java Beans (EJBs) that call the UIF API.
	This person also uses the UIF Repository Browser to determine the available Function Objects (Tuxedo Services).
Tuxedo Programmer	This person develops Tuxedo services using the Tuxedo Application to Transaction Manager Interface (ATMI) interface.

Online Guides

You can find the *iPlanet Application Server Enterprise Connector for Tuxedo Administrator's Guide* online in PDF and HTML formats. To locate these files, use the following URL:

http://docs.iplanet.com/docs/manuals/ias.html

Related Information

In addition to this Administrator's Guide, the Tuxedo Enterprise Connector comes with a Developer's Guide. The Developer's guide explains how to write Java programs (Servlet/JSP/EJB) to access the Tuxedo services that contain the business logic.

The installer copies these publications to *ias/APPS/docs/tux* subdirectory of the root installation directory of the iPlanet Application Server.

You must also refer to the *iPlanet Unified Integration Framework Developer's Guide* under the *ias/APPS/docs/bsp* subdirectory of the root installation of iPlanet Application Server for detailed information about the UIF API and Repository Browser.

In addition to these guides, there is additional information for administrators, users and developers. Use the following URL to view the related documentation:

http://docs.iplanet.com/docs/manuals/ias.html

The following lists the additional documents:

- *iPlanet Application Server Release Notes*
- *iPlanet Application Server Installation Guide*
- *iPlanet Application Server Overview Guide*
- *iPlanet Administration and Deployment Guide*
- *iPlanet Java Programmer's Guide*
- *iPlanet Application Builder Release Notes*
- *iPlanet Application Builder Installation Guide*
- *iPlanet Application Builder User's Guide*

Third Party Publications

The following BEA publications may be useful:

- BEA Tuxedo: Administrating the BEA Tuxedo System
- BEA Tuxedo: Application Development Guide
- BEA Tuxedo: Programmer's Guide
- BEA Tuxedo: Workstation Guide
- BEA Tuxedo: Reference Manual

For more information about Tuxedo technology, refer to the following books:

- Building Client/Server Applications Using Tuxedo - Carl L. Hall
- The TUXEDO System - Andrade, Carges, Dwyer, Felts

Chapter 1

Introduction

The iPlanet Application Server Enterprise Connector for Tuxedo extends the BEA Tuxedo system for Java 2 Enterprise Edition (J2EE) e-commerce applications. Using a consistent Java Application Programming Interface (API), along with the iPlanet Application Server Unified Integration Framework (UIF), the enterprise connector allows you to develop, deploy, and manage application solutions that leverage the Tuxedo transactions (services) in real time without having to learn the Tuxedo Application to Transaction Manager Interface (ATMI).

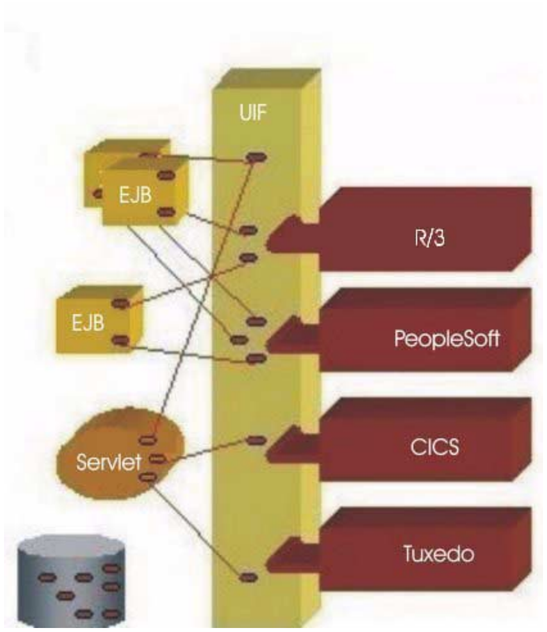
This chapter covers the following topics:

- Unified Integration Framework
- iAS Enterprise Connector for Tuxedo Architecture
- iAS Enterprise Connector for Tuxedo Tools

Unified Integration Framework

The UIF is an application programming framework that provides a single API to access different Enterprise Information Systems (EIS) using the iPlanet Application Server. As shown in Figure 1-1 an enterprise connector is developed for each EIS to allow communication between the UIF and the EIS.

Figure 1-1 The Unified Integration Framework



The framework dramatically reduces development effort by providing a consistent access layer to disparate EISs. The framework provides support for the following features:

- Connection pooling
- Thread management
- Communication and life-cycle management
- Exception management

The framework is multi-threaded to enable high-performance and fault tolerant integration. Application developers, and system integrators can easily build e-solutions accessing the various EISs using the Java Programming Language.

A universal metadata repository is also part of UIF and is used to hold information about EIS data types, business functions, and connection parameters. The EIS system administrator populates the repository using the management console provided with each connector.

Additionally, a universal repository browser allows the application developer to view business functions available and associated data types.

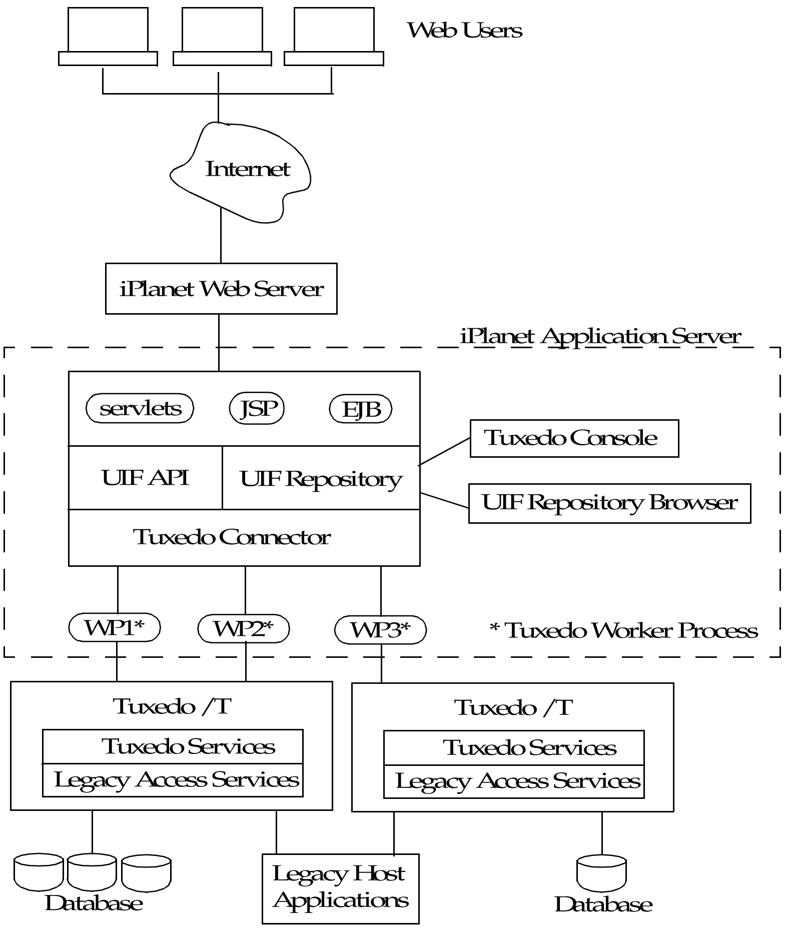
iAS Enterprise Connector for Tuxedo Architecture

The iPlanet Application Server Enterprise Connector for Tuxedo connects Java clients to applications built using the BEA Tuxedo system. The Tuxedo system provides a set of modular services, each offering specific functionality related to the application as a whole. For example, a simple banking application might have services like INQUIRY, WITHDRAW, TRANSFER, and DEPOSIT.

Typically, service requests are implemented in C or COBOL as a sequence of calls to a program library. In order to access these services the Tuxedo client libraries for the specific operating system on each client machine must be installed. The Tuxedo Enterprise Connector avoids this situation. The Tuxedo Enterprise Connector acts as a proxy for the Java clients invoking the Tuxedo services on behalf of the client. TheiPlanet Application Server Enterprise Connector for Tuxedo accepts requests from the Java clients and translates the Java-based request into a Tuxedo request which is forwarded to the Tuxedo system. The Tuxedo system processes the request and returns the information to the iPlanet Application Server Enterprise Connector for Tuxedo which translates it back to the Java client.

Figure 1-2 shows the iAS Enterprise Connector for Tuxedo architecture.

Figure 1-2 The iAS Enterprise Connector for Tuxedo Architecture



iAS Enterprise Connector for Tuxedo Tools

The following tools are available with the iAS Enterprise Connector for Tuxedo:

- Tuxedo Management Console
- UIF Repository Browser

The Tuxedo Management Console is a Java-based Graphical User Interface (GUI) tool which allows browsing and configuring of the iAS Enterprise connector for Tuxedo.

Through the Tuxedo Management Console you can:

- Create a datasource
- Edit a datasource
- Set connection pool parameters
- Set the Tuxedo authentication context
- Set the Tuxedo workstation environment variables

The GUI is also used to import the Tuxedo services definition, FML fields and VIEWs defined in the Tuxedo system into the UIF Repository. Typically, the Tuxedo administrator (domain expert) uses the Management Console to configure the enterprise connector for one or more Tuxedo systems.

The UIF Repository Browser is a Java-based GUI tool, which allows browsing of data in the UIF Repository. You can view the available business functions (Tuxedo Services), configuration parameters, and connection pools defined for a datasource. Typically an application developer uses this information while developing Java code to access the Tuxedo services.

Chapter 2

Installation

This chapter describes how to install and configure the iPlanet Application Server Enterprise Connector for Tuxedo. This chapter covers the following topics:

- About Installation
- Check the Packaging
- Hardware and Software Requirements
- Preparing to Install
- Installing the iAS Enterprise Connector for Tuxedo
- Copying files in Webless Installation
- Verifying the Installation
- Uninstalling
- Troubleshooting

About Installation

The iPlanet Application Server Enterprise Connector for Tuxedo installation procedure adds the following components to your system:

- Tuxedo Enterprise Connector software
- Sample applications
- Documentation

See Appendix A, “Installed Components”, for information about where the installation procedure copies files to your system.

Upgrade or Reinstall Issues

If upgrading an earlier version or reinstalling the current version of the iPlanet Application Server Enterprise Connector for Tuxedo, the installation procedure overwrites the currently installed files. Uninstalling the current version is not required.

Installation of multiple enterprise connectors for multiple EISs (CICS, R/3, PSFT, and Tuxedo) on the same iPlanet Application Server is allowed.

NOTE To avoid problems with overwriting files in use, stop the iPlanet Application Server before installing software. Always backup the system prior to installation including the UIF repository content.

Check the Packaging

The iAS Enterprise Connector for Tuxedo package includes the following items:

- One iPlanet Application Server Unified Integration Framework Install CD-ROM containing installation software, online documentation, and a `README.txt` file
- One iPlanet Application Server Enterprise Connector for Tuxedo Install CD-ROM containing installation software, online documentation, and a `README.txt` file
- One *iPlanet Application Server Enterprise Connector for Tuxedo Administrator's Guide*
- One *iPlanet Application Server Enterprise Connector for Tuxedo Developer's Guide*
- One *iPlanet Unified Integration Framework Developer's Guide*

Hardware and Software Requirements

The following table lists the hardware and software requirements for installing and running iAS Enterprise Connector for Tuxedo.

Table 2-1 Hardware and Software Requirements

Component	Requirement
Computer and Operating System	One of the following systems: • Intel Pentium microprocessor running Microsoft Windows NT version 4.0 SP6 or Windows2000. • Sun SPARC running Solaris 2.6 or Solaris 2.8.
Memory	Per CPU: 256 MB minimum; 512 MB recommended
Available disk space	Total disk space: 200 MB on NT and Windows2000, 400 MB on Solaris, 800 MB on Solaris if installing both NetDynamics and iPlanet Application Server 6.5
CD-ROM drive	CD-ROM drive
Network software	TCP/IP
Browser software	One of the following web browsers: • Netscape Communicator 4.6.1 or 4.7 or later. • Microsoft Internet Explorer 5.0 (Windows NT or Windows2000)
EIS	Tuxedo Workstation Client 7.1 with RollingPatch no. CR042398.
Other software	The following additional software must be installed prior to installing iPlanet Application Enterprise Connector for Tuxedo: • Webserver 6.0 SP2 • iPlanet Application Server 6.5 • Unified Integration Framework 6.5

Preparing to Install

Before installing the iAS Enterprise Connector for Tuxedo:

- Read the iAS Enterprise Connector for Tuxedo Release Notes for any last minute updates.

- Verify the Tuxedo server is properly configured, and has all the prerequisite software installed and running as described in "Hardware and Software Requirements".
- Verify the connectivity between the Tuxedo /WS and Tuxedo /T system before proceeding with the installation. The Tuxedo system comes with the `simpapp` sample application to help verification.
- Stop the iPlanet Application Server if it is running. The installation program may not be able to overwrite files currently in use if the application server is not stopped.

Installing the iAS Enterprise Connector for Tuxedo

This section describes how to install the iAS Enterprise Connector for Tuxedo on the iPlanet Application Server. This section covers the following topics:

- Windows NT/2000 Installation
- Solaris Installation

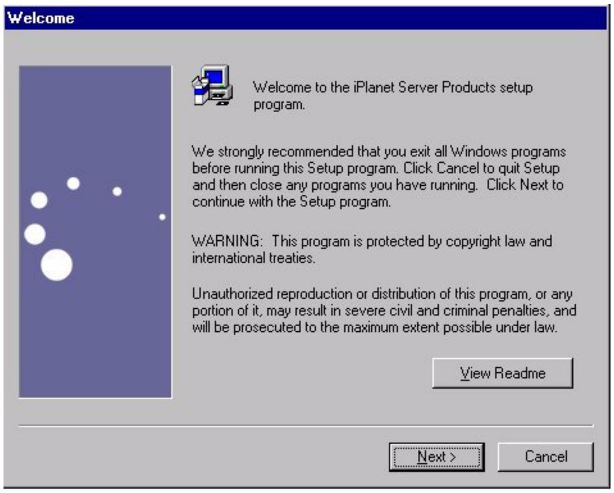
Windows NT/2000 Installation

You must be logged in to Windows NT/2000 as the administrator to install the Tuxedo Enterprise Connector.

To Install the iAS Enterprise Connector for Tuxedo:

1. Read the README.txt file on the CD-ROM.
2. Copy `NT/instux.zip` from the CD-ROM into a temporary local directory.
3. Unzip `instux.zip`, then run `setup.exe`.
4. Read the Welcome message and click Next.

Figure 2-1 Welcome Window



Welcome

Welcome to the iPlanet Server Products setup program.

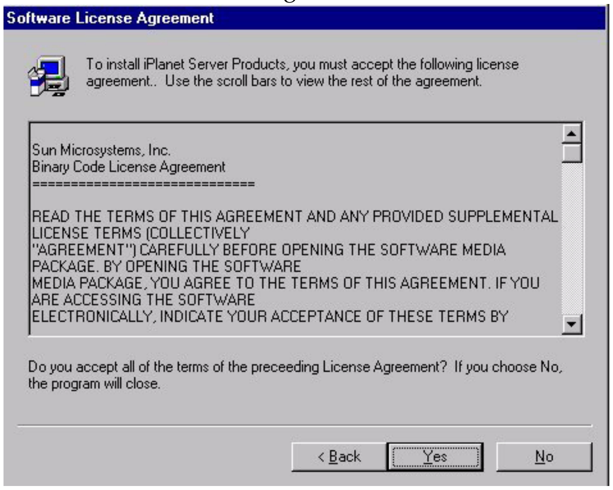
We strongly recommended that you exit all Windows programs before running this Setup program. Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

WARNING: This program is protected by copyright law and international treaties.

Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

5. Read the License Agreement, click Yes to agree and continue.

Figure 2-2 Software License Agreement Window



Software License Agreement

To install iPlanet Server Products, you must accept the following license agreement.. Use the scroll bars to view the rest of the agreement.

```
Sun Microsystems, Inc.
Binary Code License Agreement
============================

READ THE TERMS OF THIS AGREEMENT AND ANY PROVIDED SUPPLEMENTAL
LICENSE TERMS (COLLECTIVELY
"AGREEMENT") CAREFULLY BEFORE OPENING THE SOFTWARE MEDIA
PACKAGE. BY OPENING THE SOFTWARE
MEDIA PACKAGE, YOU AGREE TO THE TERMS OF THIS AGREEMENT. IF YOU
ARE ACCESSING THE SOFTWARE
ELECTRONICALLY, INDICATE YOUR ACCEPTANCE OF THESE TERMS BY
```

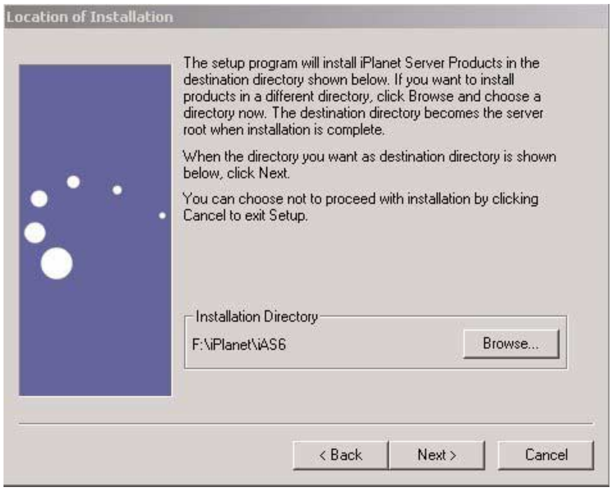
Do you accept all of the terms of the preceeding License Agreement? If you choose No, the program will close.

6. Choose an installation directory and click Next.

The iPlanet Enterprise Connector for Tuxedo must be installed in the same directory as the iPlanet Application Server.

If the installer detects that the iPlanet Application Server and/or the UIF software is not installed, a message is displayed indicating the installation cannot continue. Click OK to exit Setup and install the required software.

Figure 2-3 Location of Installation Window



The setup program will install iPlanet Server Products in the destination directory shown below. If you want to install products in a different directory, click Browse and choose a directory now. The destination directory becomes the server root when installation is complete.

When the directory you want as destination directory is shown below, click Next.

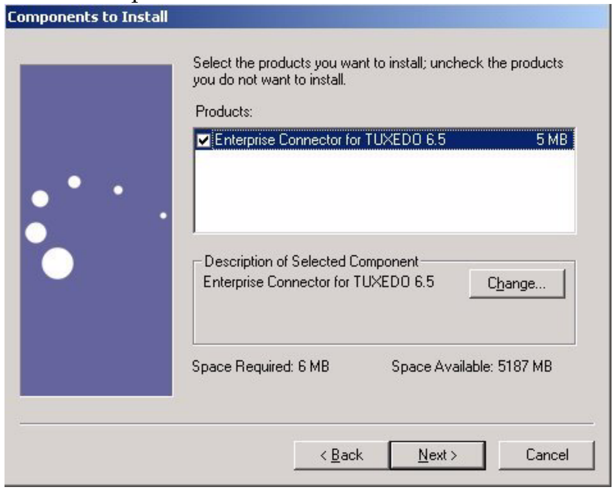
You can choose not to proceed with installation by clicking Cancel to exit Setup.

Installation Directory

F:\iPlanet\iAS6

7. Select the Enterprise Connector for TUXEDO 6.5 checkbox and click Next.

Figure 2-4 Components to Install Window


Components to Install

Select the products you want to install; uncheck the products you do not want to install.

Products:

☑ Enterprise Connector for TUXEDO 6.5 5 MB

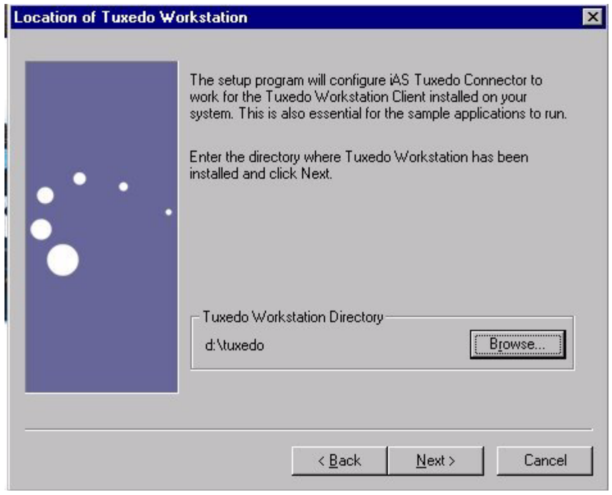
Description of Selected Component
Enterprise Connector for TUXEDO 6.5

Space Required: 6 MB Space Available: 5187 MB

8. Click the Browse button and specify the location of the Tuxedo Workstation client software and then click Next.

   The installer validates the specified directory and displays an error message if its an invalid Tuxedo directory.

Figure 2-5 Location of Tuxedo Workstation Window


Location of Tuxedo Workstation

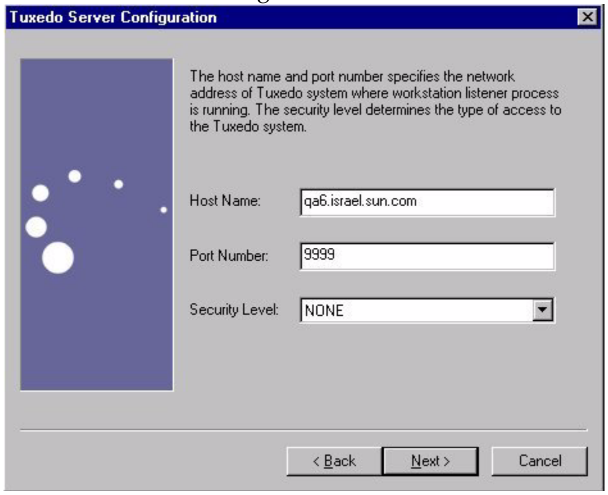
The setup program will configure iAS Tuxedo Connector to work for the Tuxedo Workstation Client installed on your system. This is also essential for the sample applications to run.

Enter the directory where Tuxedo Workstation has been installed and click Next.

Tuxedo Workstation Directory
d:\tuxedo

9. Type in the host name, port number and security level into the Tuxedo Server Configuration dialog box. When completed click Next.

Table 2-2 Tuxedo Server Configuration Window



The host name and port number specifies the network address of Tuxedo system where workstation listener process is running. The security level determines the type of access to the Tuxedo system.

Host Name: qa6.israel.sun.com

Port Number: 9999

Security Level: NONE

Set the host name and port number to the hostname and workstation listener port intended to connect to Tuxedo system. This is equivalent to the WSNADDR environment variable set before running a workstation client. WSNADDR is constructed as *//<hostname>:<port>*. This information is mandatory because the iPlanet Application Server Enterprise Connector for Tuxedo connects to the Tuxedo system via the Tuxedo Workstation client.

Set the security level to any of the three constants based on security level configured on the Tuxedo system as shown in Table 2-3. This is determined by the SECURITY parameter value in the RESOURCES section of the Tuxedo configuration file. Consult the Tuxedo system administrator to obtain the configured security level.

If the Tuxedo system is configured with SECURITY it is necessary to pass the security information. The information is passed to the BEA Tuxedo system via a TPINIT buffer while the iPlanet Application Server Enterprise Connector for Tuxedo is attaching to the Tuxedo system.

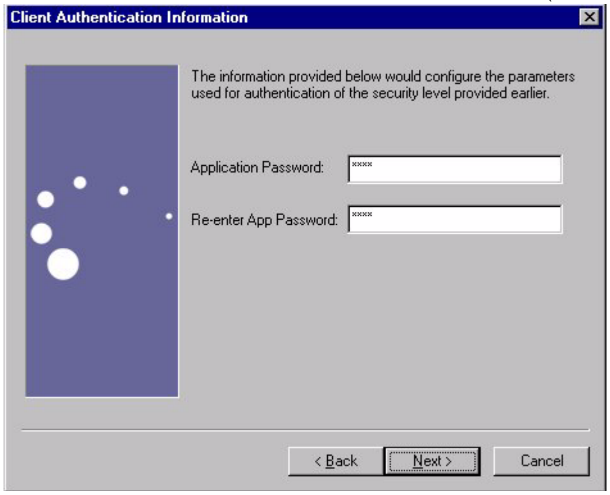
Table 2-3 Tuxedo Security Level

Constant	Type of Authorization
NONE	The security level is configured to NONE. No additional information is needed by the Tuxedo Workstation client to connect to Tuxedo system. This is the default value used by the installer. If this option is selected the installer doesn't prompt for additional information.
APP_PW	The security level is configured to APP_PW. This security level requires every client to provide an application password as part of joining the Tuxedo system. When this option is selected during installation, the installer prompts for an application password. The information is passed inside the TPINIT data structure when tpinit() is called by the iPlanet Application Server Enterprise Connector for Tuxedo.
USER_AUTH/ACL/ MANDATORY_ACL	The security level is configured to USER_ACL, ACL or MANDATORY_ACL. These levels require an application password as well as user specific data (user name, client name, and user password) which is passed to an authentication service before the client attaches to Tuxedo system. When this option is selected during installation, the installer prompts for an application password, user name, client name and user password. This information is passed inside the TPINIT data structure when tpinit() is called by the iPlanet Application Server Enterprise Connector for Tuxedo.

Refer to BEA Tuxedo Programmer's Guide for more details on security levels.

10. (Optional) If the Security Level is set to APP_PW specify an application password and click on the Next button.

Figure 2-6 Client Authentication Information Window (for APP_PW option only)



Client Authentication Information

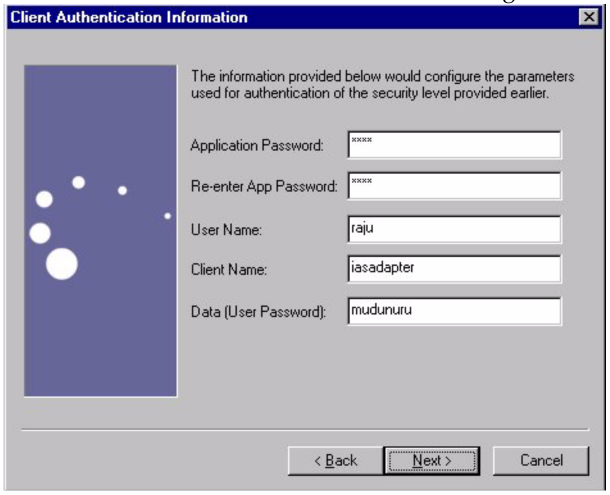
The information provided below would configure the parameters used for authentication of the security level provided earlier.

Application Password: ****

Re-enter App Password: ****

11. (Optional) If the Security Level is set to USER_AUTH/ACL/MANDATORY_ACL specify an application password, user name, client name, and user password. When completed click Next.

Figure 2-7 Client Authentication Information Dialog Box



Client Authentication Information

The information provided below would configure the parameters used for authentication of the security level provided earlier.

Application Password: ****

Re-enter App Password: ****

User Name: raju

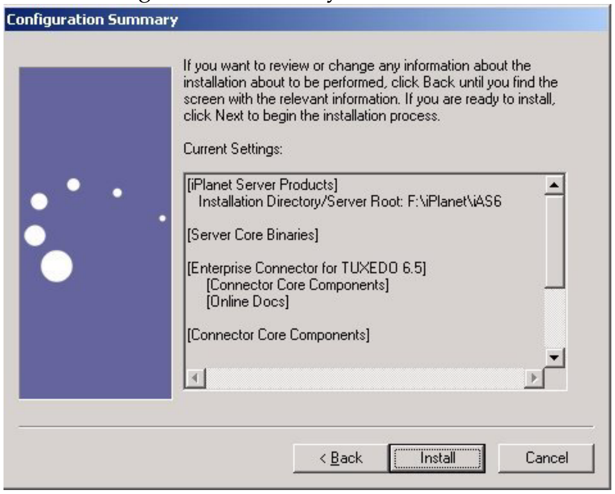
Client Name: iasadapter

Data (User Password): mudunuru

This option is for USER_ACL/ACL/MANDATORY_ACL only.

12. Check the Configuration Summary setting and, if they are correct, click Install.

Figure 2-8 Configuration Summary Window



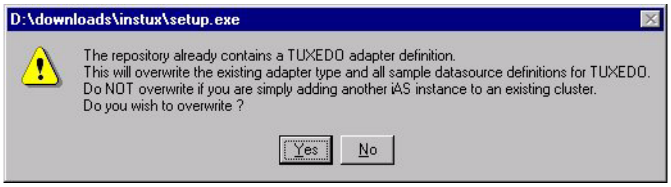
If you want to review or change any information about the installation about to be performed, click Back until you find the screen with the relevant information. If you are ready to install, click Next to begin the installation process.

Current Settings:

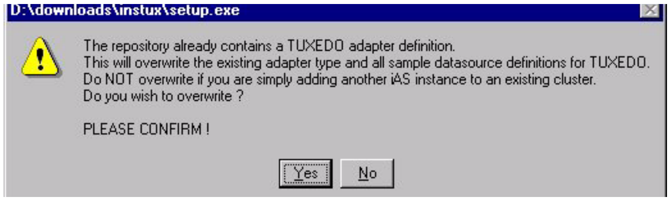
```
[iPlanet Server Products]
  Installation Directory/Server Root: F:\iPlanet\iAS6

[Server Core Binaries]

[Enterprise Connector for TUXEDO 6.5]
    [Connector Core Components]
    [Online Docs]

[Connector Core Components]
```

13. (Optional) These are the possible warning window(s) that can appear, if the installer detects that the iPlanet Application Server Enterprise Connector for Tuxedo is already installed. Choose Yes to overwrite the existing adapter type and all sample datasource definitions shipped with Tuxedo connector. Repository nodes `bsp.adapterTypes.Tuxedo`, `bsp.dataSources.Tuxedo-OnlineBank`, and `bsp.dataSources.Tuxedo-SimpApp` are overwritten by the installer. The user defined datasources are not overwritten. Select No if you are simply adding another iPlanet Application Server instance to an existing cluster.

Figure 2-9 Warning Windows



The repository already contains a TUXEDO adapter definition.
This will overwrite the existing adapter type and all sample datasource definitions for TUXEDO.
Do NOT overwrite if you are simply adding another iAS instance to an existing cluster.
Do you wish to overwrite ?


D:\downloads\instux\setup.exe

The repository already contains a TUXEDO adapter definition.
This will overwrite the existing adapter type and all sample datasource definitions for TUXEDO.
Do NOT overwrite if you are simply adding another iAS instance to an existing cluster.
Do you wish to overwrite ?

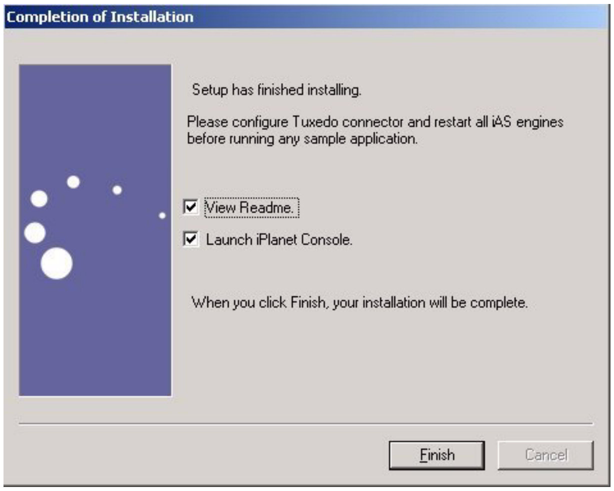
PLEASE CONFIRM !

14. Read the Completion of Installation window and click Finish. The iPlanet Application Server Enterprise Connector for Tuxedo is now installed.

    Configure your installation by setting the environment variables described in "Configuring the Installation.

    The machine must be rebooted and the iPlanet Application Server restarted before using the iPlanet Application Server Enterprise Connector for Tuxedo.

Figure 2-10 Completion of Installation Window


Completion of Installation

Setup has finished installing.

Please configure Tuxedo connector and restart all iAS engines before running any sample application.

[✓] View Readme.
[✓] Launch iPlanet Console.

When you click Finish, your installation will be complete.

Configuring the Installation

To configure the iAS Enterprise Connector for Tuxedo

1. Add `<tuxedo /ws rootdir>\bin` to `path` environment variable.
2. Add `<tuxedo /ws rootdir>\locale\C` to NLSPATH environment variable.
3. Set `TUXDIR` environment variable to Tuxedo workstation client installation root directory.

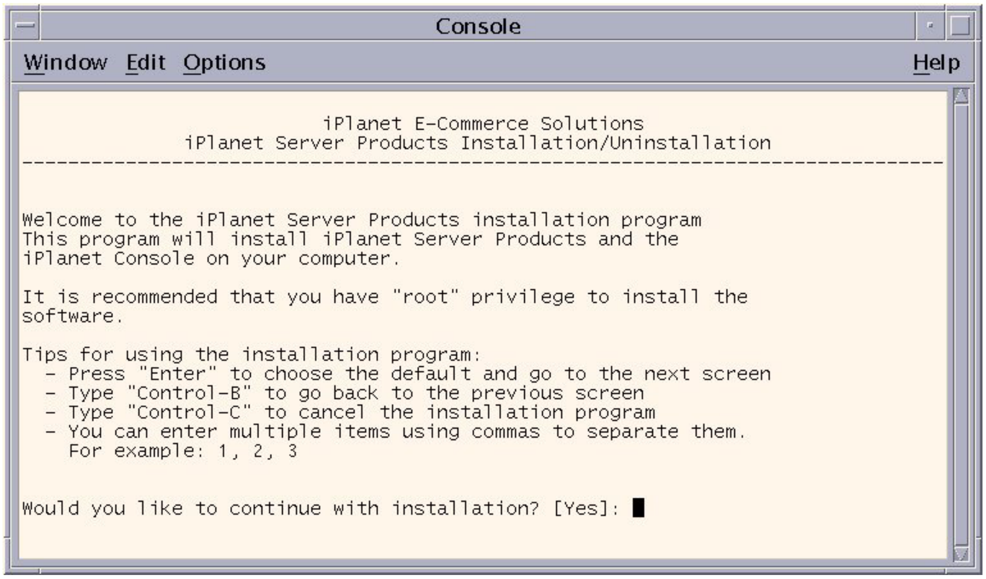
NOTE Refer to BEA Tuxedo documentation for additional information.

Solaris Installation

The following procedure describes how to install the iPlanet Application Server Enterprise Connector for Tuxedo on a Solaris machines:

1. Insert the iPlanet Application Server Enterprise Connector for Tuxedo Install CD-ROM into the CD-ROM drive.
2. Mount the CD-ROM, for example, `/cdrom/cdrom0`.
3. Read the README.txt file on the CD-ROM.
4. Copy the `Solaris/instux.tar` file from the CD-ROM into a temporary directory.
5. Untar the `instux.tar` file and run the `./setup` command.
6. The following welcome screen appears either press enter or type Yes to continue.

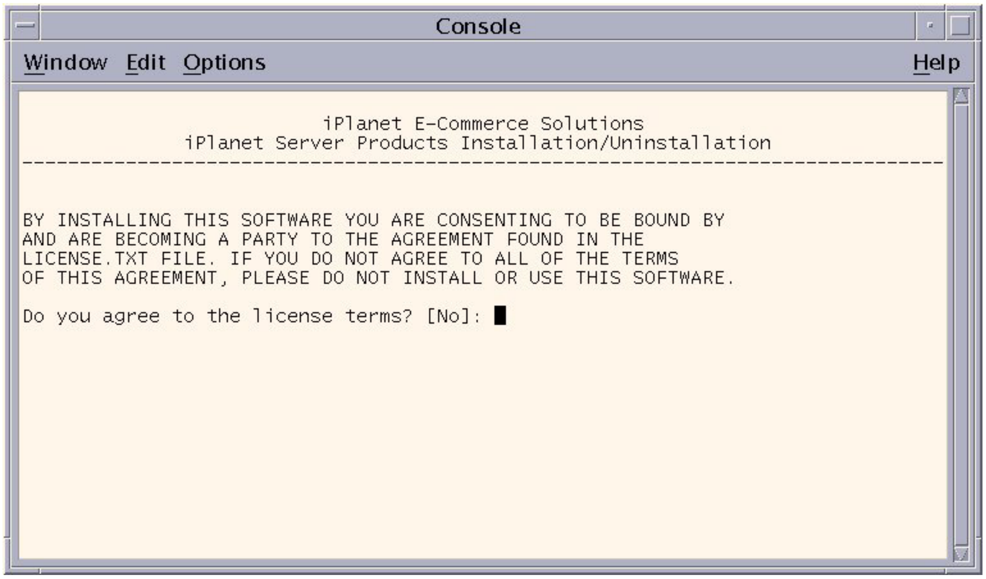
Figure 2-11 Welcome Screen



```
                 iPlanet E-Commerce Solutions
             iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


Welcome to the iPlanet Server Products installation program
This program will install iPlanet Server Products and the
iPlanet Console on your computer.

It is recommended that you have "root" privilege to install the
software.

Tips for using the installation program:
  - Press "Enter" to choose the default and go to the next screen
  - Type "Control-B" to go back to the previous screen
  - Type "Control-C" to cancel the installation program
  - You can enter multiple items using commas to separate them.
    For example: 1, 2, 3


Would you like to continue with installation? [Yes]:
```

7. The following license agreement screen appears.

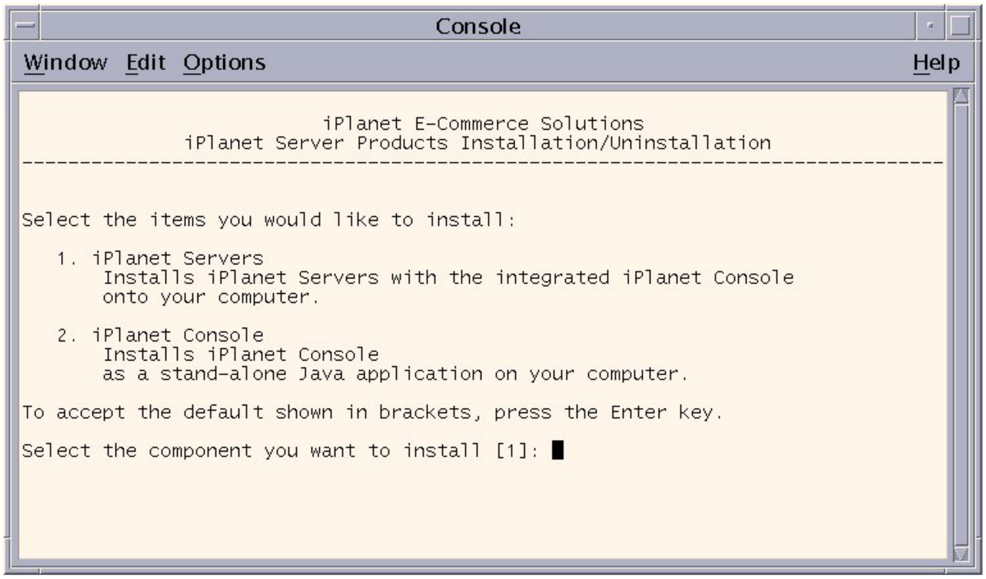
Figure 2-12 License.



```
                 iPlanet E-Commerce Solutions
          iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


BY INSTALLING THIS SOFTWARE YOU ARE CONSENTING TO BE BOUND BY
AND ARE BECOMING A PARTY TO THE AGREEMENT FOUND IN THE
LICENSE.TXT FILE. IF YOU DO NOT AGREE TO ALL OF THE TERMS
OF THIS AGREEMENT, PLEASE DO NOT INSTALL OR USE THIS SOFTWARE.

Do you agree to the license terms? [No]:
```

If you agree with the terms of the license agreement type *Y* and then press return.

8. The following Component Installation screen appears.

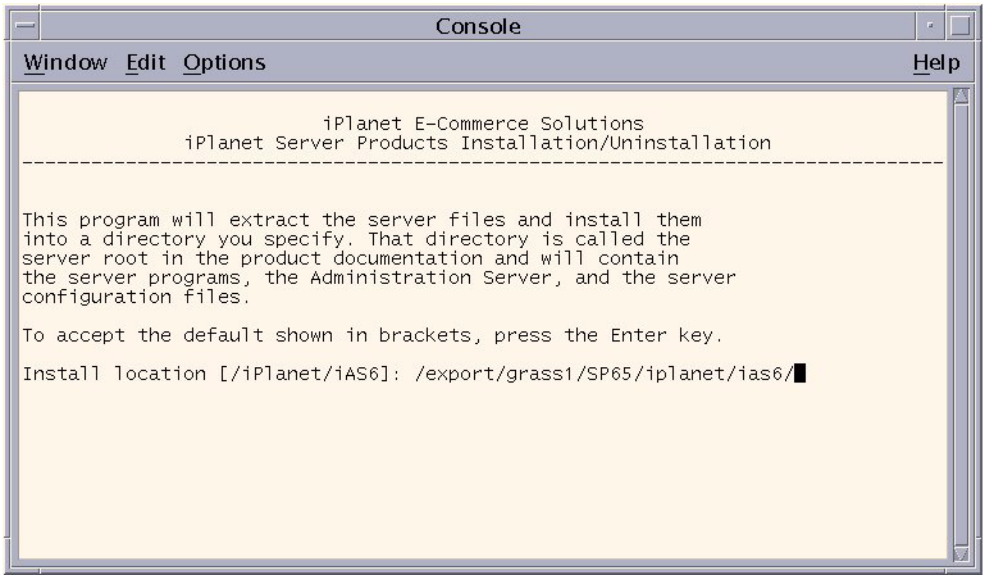


```
                    iPlanet E-Commerce Solutions
             iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


Select the items you would like to install:

   1. iPlanet Servers
       Installs iPlanet Servers with the integrated iPlanet Console
       onto your computer.

   2. iPlanet Console
       Installs iPlanet Console
       as a stand-alone Java application on your computer.

To accept the default shown in brackets, press the Enter key.

Select the component you want to install [1]:
```

9. Press enter to select *iPlanet Servers.*

   The Installation Directory screen is displayed.

Figure 2-13 Installation Directory.

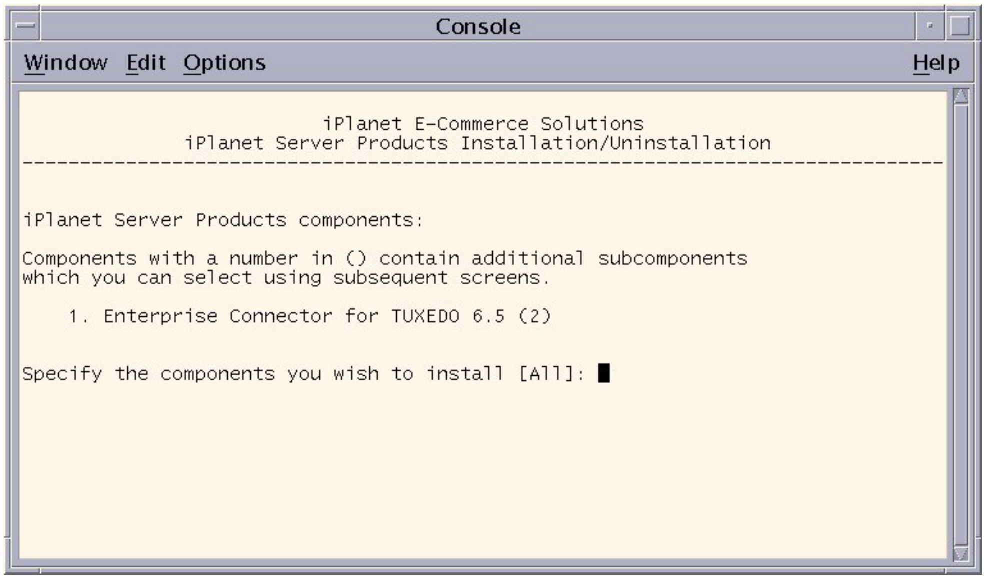


```
                    iPlanet E-Commerce Solutions
             iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


This program will extract the server files and install them
into a directory you specify. That directory is called the
server root in the product documentation and will contain
the server programs, the Administration Server, and the server
configuration files.

To accept the default shown in brackets, press the Enter key.

Install location [/iPlanet/iAS6]: /export/grass1/SP65/iplanet/ias6/
```

10. Specify the iPlanet Application Server installation directory.

The *Components to Install* screen is displayed.

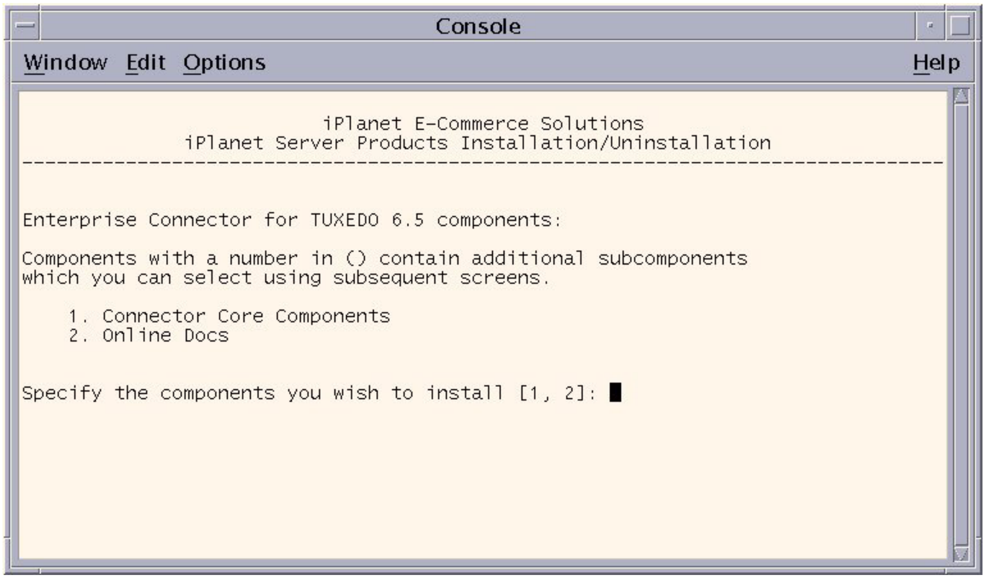
Figure 2-14 Components to Install



```
                    iPlanet E-Commerce Solutions
           iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


iPlanet Server Products components:

Components with a number in () contain additional subcomponents
which you can select using subsequent screens.

    1. Enterprise Connector for TUXEDO 6.5 (2)


Specify the components you wish to install [All]:
```

11. Press return to display the subcomponents to install screen.

Figure 2-15 Subcomponents to Install

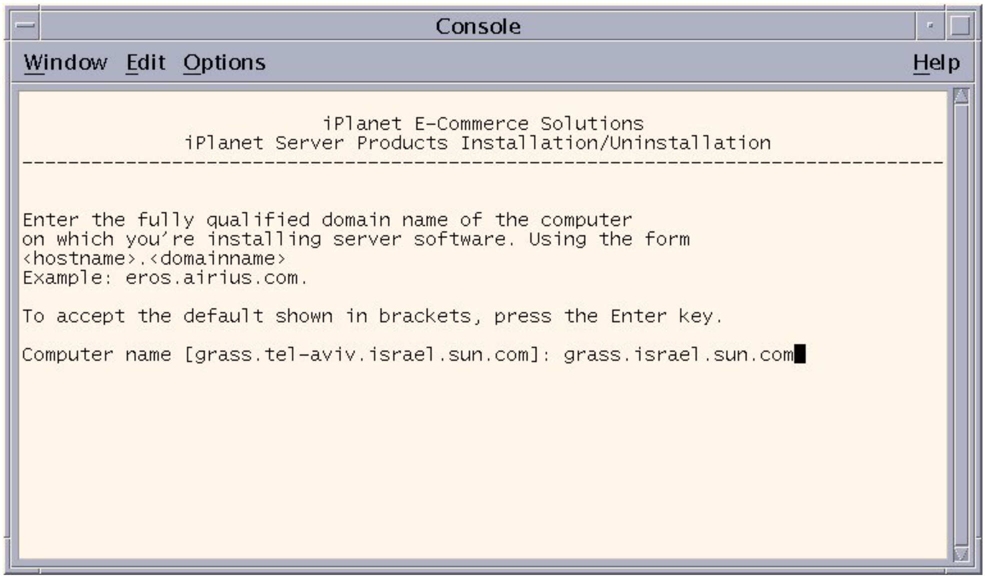


```
                  iPlanet E-Commerce Solutions
            iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


Enterprise Connector for TUXEDO 6.5 components:

Components with a number in () contain additional subcomponents
which you can select using subsequent screens.

    1. Connector Core Components
    2. Online Docs


Specify the components you wish to install [1, 2]:
```

12. Press return to install the connector core components and the documentation.

The screen to specify the domain name of your computers is displayed.

Figure 2-16 Specifying Domain Name of Computer.

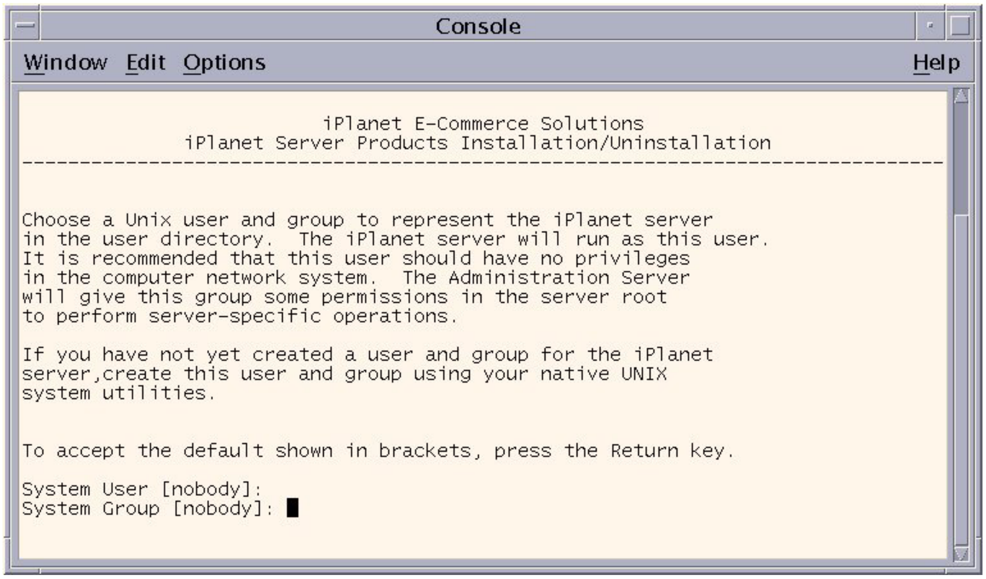


```
                  iPlanet E-Commerce Solutions
            iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


Enter the fully qualified domain name of the computer
on which you're installing server software. Using the form
<hostname>.<domainname>
Example: eros.airius.com.

To accept the default shown in brackets, press the Enter key.

Computer name [grass.tel-aviv.israel.sun.com]: grass.israel.sun.com
```

13. Type in the name domain name of your computer and press return.

    The User Name screen is displayed.

Figure 2-17 User Name and Group



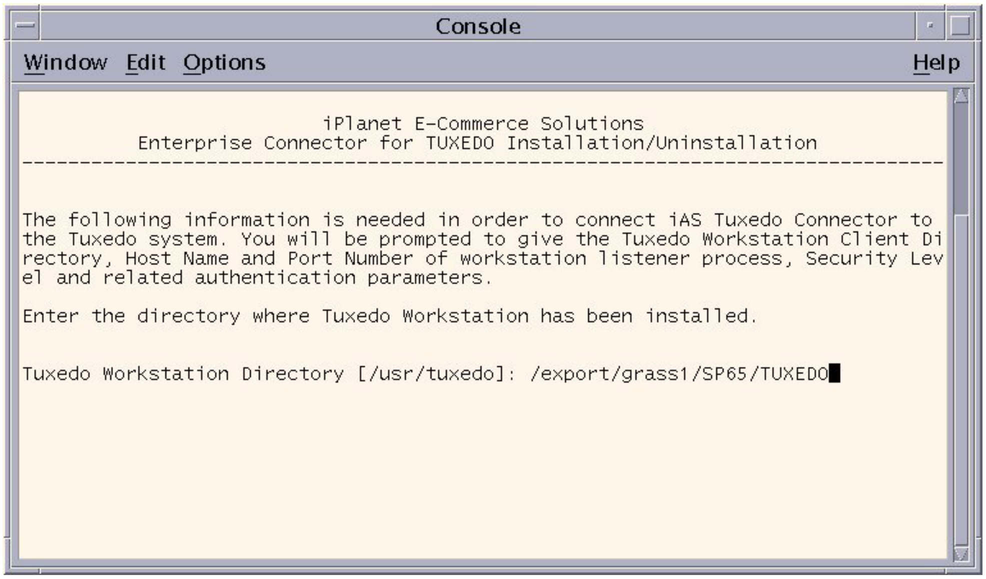
```
                         iPlanet E-Commerce Solutions
              iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


Choose a Unix user and group to represent the iPlanet server
in the user directory.  The iPlanet server will run as this user.
It is recommended that this user should have no privileges
in the computer network system.  The Administration Server
will give this group some permissions in the server root
to perform server-specific operations.

If you have not yet created a user and group for the iPlanet
server,create this user and group using your native UNIX
system utilities.


To accept the default shown in brackets, press the Return key.

System User [nobody]:
System Group [nobody]: █
```

14. Specify the Unix user and group name under which your iPlanet Application Server Enterprise Connector for Tuxedo will run.

    The user and group name must have already been set up prior to running an installation.

15. Press enter to display the Tuxedo Workstation Directory screen.

Figure 2-18 Tuxedo Workstation Directory.



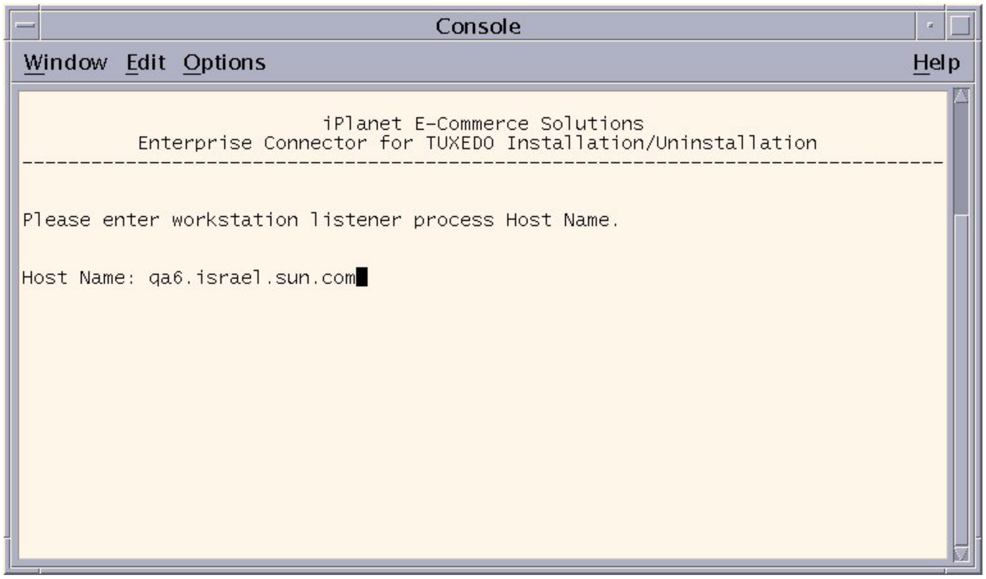
```
                    iPlanet E-Commerce Solutions
          Enterprise Connector for TUXEDO Installation/Uninstallation
--------------------------------------------------------------------------------


The following information is needed in order to connect iAS Tuxedo Connector to
the Tuxedo system. You will be prompted to give the Tuxedo Workstation Client Di
rectory, Host Name and Port Number of workstation listener process, Security Lev
el and related authentication parameters.

Enter the directory where Tuxedo Workstation has been installed.


Tuxedo Workstation Directory [/usr/tuxedo]: /export/grass1/SP65/TUXEDO
```

Specify the Tuxedo Workstation client installation directory and press Enter to continue.

The installer validates the directory specified and displays an error message if its an invalid Tuxedo directory.

16. The installer prompts for host name, port number and security level parameters.

Figure 2-19 Host Name



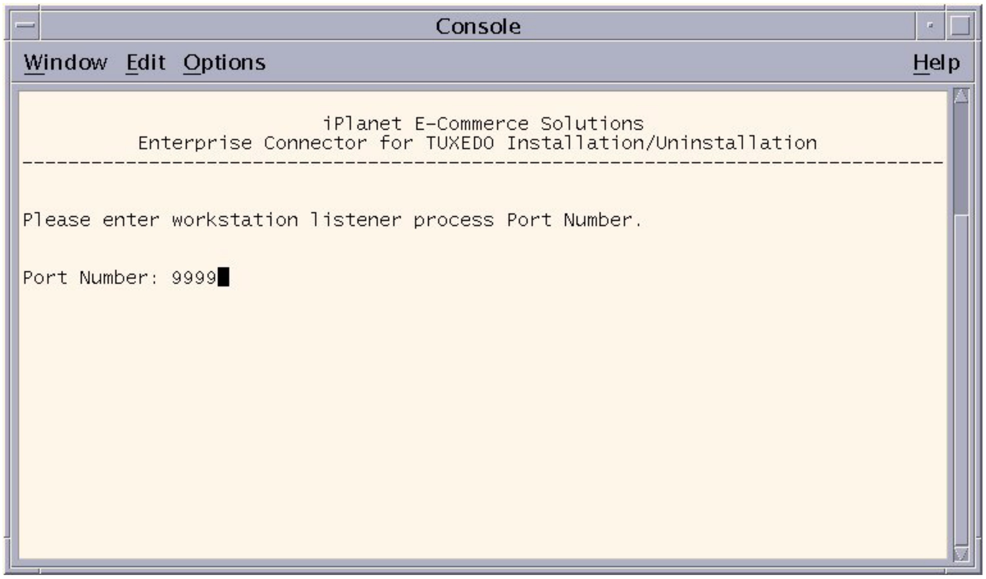
```
                    iPlanet E-Commerce Solutions
          Enterprise Connector for TUXEDO Installation/Uninstallation
--------------------------------------------------------------------------------


Please enter workstation listener process Host Name.


Host Name: qa6.israel.sun.com
```

17. Specify the host name and press return.

    The host name and workstation listener port parameters are equivalent to the WSNADDR environment variable you typically set before running the Tuxedo workstation client. WSNADDR is constructed as *//<hostname>:<port>*. This information is mandatory because the Tuxedo Enterprise Connector connects to the Tuxedo system via the Tuxedo workstation client.

18. Specify the workstation listener port you intend to connect to the Tuxedo system.

Figure 2-20 Workstation Listener Process Port Number

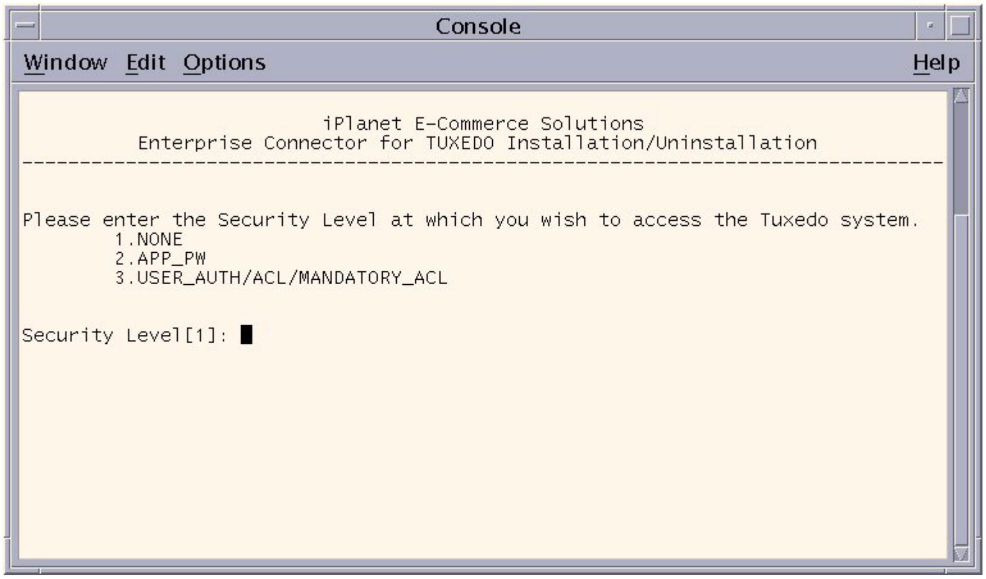


```
                    iPlanet E-Commerce Solutions
        Enterprise Connector for TUXEDO Installation/Uninstallation
--------------------------------------------------------------------------------


Please enter workstation listener process Port Number.


Port Number: 9999
```

19. Type in the Workstation Listener process port number and press return.

The Security screen is displayed.

Figure 2-21 Security



```
                    iPlanet E-Commerce Solutions
        Enterprise Connector for TUXEDO Installation/Uninstallation
--------------------------------------------------------------------------------


Please enter the Security Level at which you wish to access the Tuxedo system.
        1.NONE
        2.APP_PW
        3.USER_AUTH/ACL/MANDATORY_ACL


Security Level[1]:
```

20. Specify the security level based on the security level configured on the Tuxedo system. This is determined by the SECURITY parameter value in the RESOURCES section of the Tuxedo configuration file. Consult the Tuxedo system administrator to obtain the configured security level.

    Security information must be passed if the Tuxedo system is configured with SECURITY. The information is passed to the BEA Tuxedo system via a TPINIT buffer while the Tuxedo Enterprise Connector is attaching to the Tuxedo system.

Table 2-4 Security Options

Constant	Type of Authentication
NONE	The security level is configured to NONE. No additional information is needed by the Tuxedo Workstation client to connect to Tuxedo system. This is the default value used by the installer. If this option is selected the installer doesn't prompt for additional information.
APP_PW	The security level is configured to APP_PW. This security level requires every client to provide an application password as part of joining the Tuxedo system. When this option is selected during installation, the installer prompts for an application password. The information is passed inside the TPINIT data structure when tpinit() is called by the iPlanet Application Server Enterprise Connector for Tuxedo.
USER_AUTH/ACL/MANDATORY_ACL	The security level is configured to USER_ACL, ACL or MANDATORY_ACL. These levels require an application password as well as user specific data (user name, client name, user password) which is passed to an authentication service before the client attaches to Tuxedo system. When this option is selected during installation, the installer prompts for an application password, user name, client name and user password. This information is passed inside the TPINIT data structure when tpinit() is called by the iPlanet Application Server Enterprise Connector for Tuxedo.

21. (Optional) The following screen is displayed if the Security Level is set to APP_PW. Enter the application password and press the Enter key.

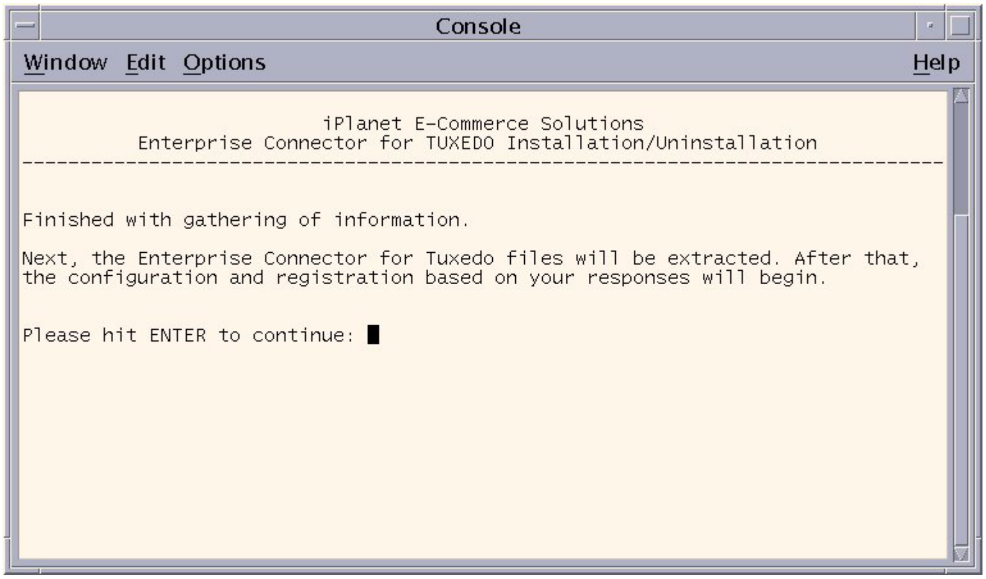
```
iPlanet E-Commerce Solutions
Enterprise Connector for TUXEDO Installation/Uninstallation
----------------------------------------------------------------

Please enter the Security Level at which you wish to access the
Tuxedo domain.
        1.NONE
        2.APP_PW
        3.USER_AUTH/ACL/MANDATORY_ACL

Security Level[1]: 2

Please enter Application Password.

Application Password:
```

22. (Optional) Installer prompts for the application password, user name, client name, and data (user password) if the Security Level is set to USER_AUTH/ ACL/MANDATORY_ACL. Enter the application password, user name, client name, and user password and press the Enter key.

```
iPlanet E-Commerce Solutions
Enterprise Connector for TUXEDO Installation/Uninstallation
----------------------------------------------------------------

Application Password:

User Name [raju]:

Client Name [iastuxadapter]:

Data (User Password):
```

23. Press Enter key to continue.

Figure 2-22 Configuration and Registration



```
                    iPlanet E-Commerce Solutions
          Enterprise Connector for TUXEDO Installation/Uninstallation
--------------------------------------------------------------------------------


Finished with gathering of information.

Next, the Enterprise Connector for Tuxedo files will be extracted. After that,
the configuration and registration based on your responses will begin.


Please hit ENTER to continue:
```

24. (Optional) The following warning screen(s) appears, if the installer detects that the iPlanet Application Server Enterprise Connector for Tuxedo is already installed. Choose Yes to overwrite the existing adapter type and all sample datasource definitions shipped with iAS Enterprise Connector for Tuxedo. Repository nodes `bsp.adapterTypes.Tuxedo`, `bsp.dataSources.Tuxedo-OnlineBank`, and `bsp.dataSources.Tuxedo-SimpApp` are overwritten by the installer. The user defined datasources are not overwritten. Select No if you are simply adding another iPlanet Application Server instance to an existing cluster.

```
The repository already contains a TUXEDO adapter definition.
This will overwrite the existing adapter type and all sample
datasource definitions for the TUXEDO adapter.
Do *not* overwrite if you are simply adding another iAS instance
to an existing cluster.
Do you wish to overwrite ?

Confirm? [No]:
```

```
The repository already contains a TUXEDO adapter definition.
This will overwrite the existing adapter type and all sample
datasource definitions for the TUXEDO adapter.
Do *not* overwrite if you are simply adding another iAS instance
to an existing cluster.
Do you wish to overwrite ?

PLEASE RE-CONFIRM !

Confirm? [No]:
```

25. When all files are installed, the Completion of Installation screen appears. Press Return to continue. The iPlanet Application Server Enterprise Connector for Tuxedo is installed.

    Configure your installation by setting the environment variables described in "Configuring the Installation".

    You must restart the iPlanet Application Server before using the iPlanet Application Server Enterprise Connector for Tuxedo.

Configuring the Installation

To configure the iAS Enterprise Connector for Tuxedo

1. Modify the script iasenv.ksh that is located in `<ias_root>/ias/env` as follows:

   Add $TUXDIR/lib to LD_LIBRARY_PATH.

2. Add `<tuxedo /ws rootdir>/locale/C` to `NLSPATH` environment variable.
3. Set *TUXDIR* environment variable to Tuxedo workstation client installation root directory.

NOTE Refer to BEA Tuxedo documentation for additional information.

NOTE There is no need to set the following Tuxedo environment variables:

- *WSNADDR*
- *WSDEVICE*

These variables are set internally by the iPlanet Application Server Enterprise Connector for Tuxedo based on information provided during installation or while creating a new datasource using the Tuxedo Management Console.

It is highly recommended to have the Tuxedo Server (Tuxedo /T system) configured for both simple application and bank application to run the samples supplied with iAS Enterprise Connector for Tuxedo.

Copying files in Webless Installation

In a webless installation, the web server and iPlanet Application Server reside on separate machine. In this configuration mode the iAS Enterprise Connector for Tuxedo installation procedure will not automatically copy the HTML and image files pertaining to connector samples to web server document root directory. In order to run the samples shipped with iAS Enterprise Connector for Tuxedo, a set of HTML and image files must be copied manually from the iPlanet Application Server instance to web server instance.

To run the samples in a webless installation

1. Create the `tuxSamples` directory under the web server document root directory. For example, `<iWS_installdir>/docs/tuxSamples`.
2. Copy `index.html` file from `<iAS_rootdir>/ias/APPS/adapters/tux/htmldocs/tuxSamples` directory to `<iWS_installdir>/docs/tuxSamples` directory on web server machine.
3. Create `TuxBank` directory under `<iWS_installdir>/docs/GXApp` directory on web server machine.
4. Create `images` directory under `<iWS_installdir>/docs/GXApp/TuxBank` directory on web server machine.
5. Create `TuxSimpApp` directory under `<iWS_installdir>/docs/GXApp` directory on web server machine.

6. Copy `sindex.html` file from `<iAS_rootdir>/ias/APPS/adapters/tux/htmldocs/GXApp/TuxSimpApp to <iWS installdir>/docs/GXApp/TuxSimpApp.`

7. Copy `bindex.html` file from `<iAS_rootdir>/ias/APPS/adapters/tux/htmldocs/GXApp/TuxBank to <iWS installdir>/docs/GXApp/TuxBank.`

8. Copy all image files under `<iAS_rootdir>/ias/APPS/adapters/tux/htmldocs/GXApp/TuxBank/images directory to <iWS installdir>/docs/GXApp/TuxBank/images` directory.

Verifying the Installation

To verify the iAS Enterprise Connector for Tuxedo installation

1. Verify the Tuxedo server (the Tuxedo /T) has been properly installed and configured according to the instructions in the BEA Tuxedo documentation.

2. Verify the workstation listener is running on the Tuxedo server.

3. On the Tuxedo server, run the Tuxedo Simple Application (`simpapp`) server program, according to the instructions in the BEA Tuxedo documentation. This program offers the *TOUPPER* service, which converts the input string to upper case and returns it to the client.

4. Verify the Tuxedo workstation client software (Tuxedo /WS) has been properly installed and configured according to the instructions in the BEA Tuxedo documentation. The Tuxedo /WS software must be installed on the system where the iPlanet Application Server is installed.

5. Verify the connection between the Tuxedo server and Tuxedo workstation client software by running the sample application `simpapp` provided by the Tuxedo system.

6. Verify the environment variables are set correctly as described in "Configuring the Installation".

7. Verify the iPlanet Application Server engines (`kas`, `kxs`, and `kjs`) are running.

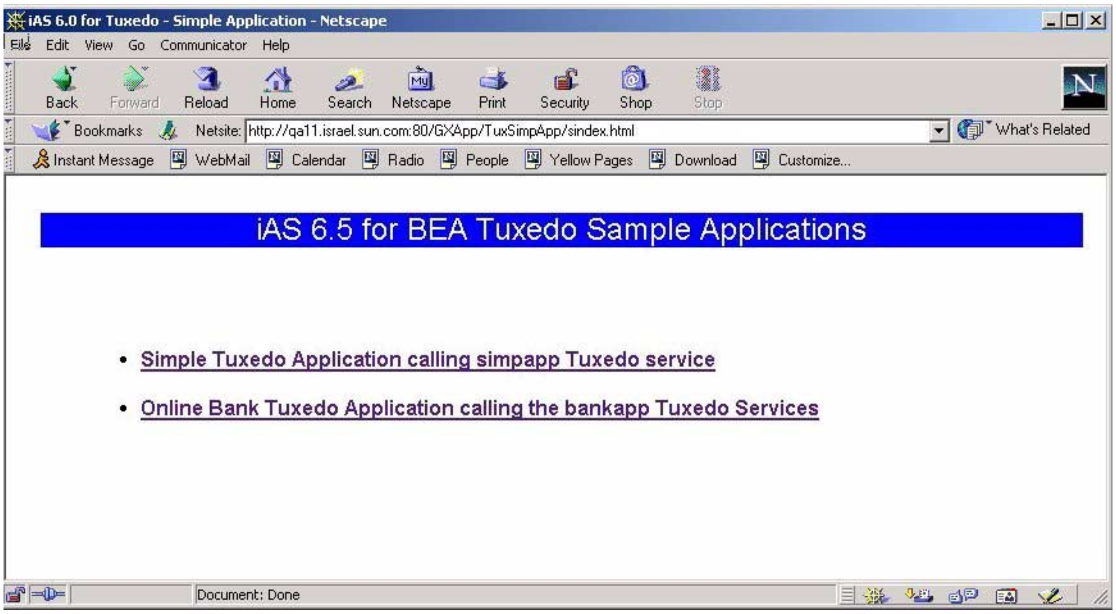
8. Start the Tuxedo Connector Sample application.

Windows NT: From the Start/Programs menu, choose iPlanet Application Server 6.5 / TUXEDO Connector 6.5 - Sample Applications entry.

Unix: Visit `http://<hostname:port>/tuxSamples/index.html` URL. Where hostname is your web server's hostname. Example, `http://horse:80/tuxSamples/index.html`.

Figure 2-23 shows the Tuxedo samples Home page.

Figure 2-23 Tuxedo Samples Index HTML

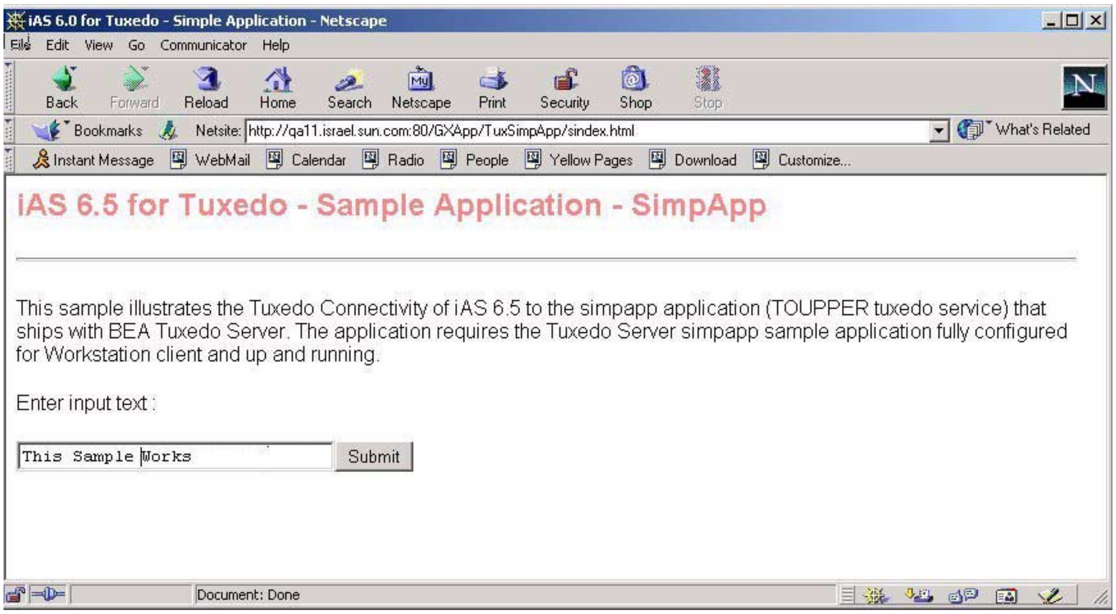

iAS 6.0 for Tuxedo - Simple Application - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Netsite: http://qa11.israel.sun.com:80/GXApp/TuxSimpApp/sindex.html What's Related
Instant Message WebMail Calendar Radio People Yellow Pages Download Customize...
Document: Done

iAS 6.5 for BEA Tuxedo Sample Applications

- Simple Tuxedo Application calling simpapp Tuxedo service
- Online Bank Tuxedo Application calling the bankapp Tuxedo Services

Click the "Simple Tuxedo Application calling simpapp Tuxedo service" link. The input form for sample as shown in Figure 2-24.

Figure 2-24 Simple Tuxedo Application Input Form


iAS 6.0 for Tuxedo - Simple Application - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Netsite: http://qa11.israel.sun.com:80/GXApp/TuxSimpApp/sindex.html What's Related
Instant Message WebMail Calendar Radio People Yellow Pages Download Customize...
Submit
Document: Done

iAS 6.5 for Tuxedo - Sample Application - SimpApp

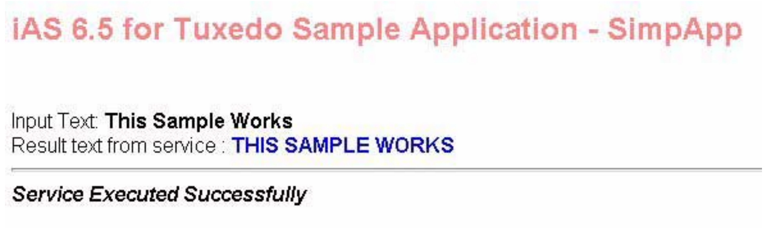
This sample illustrates the Tuxedo Connectivity of iAS 6.5 to the simpapp application (TOUPPER tuxedo service) that ships with BEA Tuxedo Server. The application requires the Tuxedo Server simpapp sample application fully configured for Workstation client and up and running.

Enter input text :

This Sample Works

Enter any string and click the Submit button. A successful message with the entered string in uppercase is displayed, as shown in Figure 2-25. If not, see "Troubleshooting".

Figure 2-25 Simple Application Output



iAS 6.5 for Tuxedo Sample Application - SimpApp

Input Text: **This Sample Works**
Result text from service : **THIS SAMPLE WORKS**

Service Executed Successfully

Uninstalling

This section describes how to uninstall the Tuxedo Enterprise Connector from an iPlanet Application Server. It contains the following topics:

- Removing the Connector from Windows NT/2000
- Removing the Connector from Solaris

Removing the Connector from Windows NT/2000

The procedure for removing the iAS Enterprise Connector for Tuxedo consists of three steps:

- Step 1 - Uninstalling the iAS Enterprise Connector for Tuxedo
- Step 2 - Deleting the Repository Content
- Step 3 - Unregistering the Tuxedo Extension

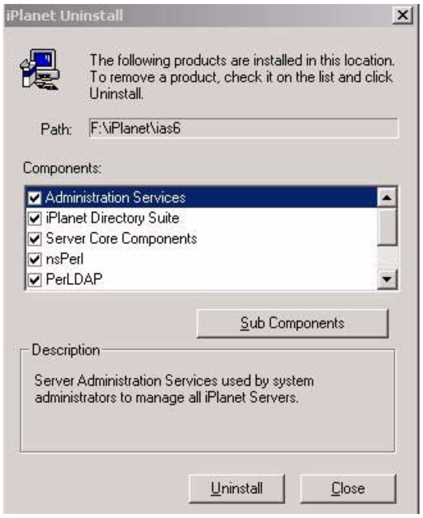
Step 1 - Uninstalling the iAS Enterprise Connector for Tuxedo

To uninstall the iAS Enterprise Connector for Tuxedo

1. Select Start > Programs - iPlanet Application Server 6.5 > Uninstaller.

The iPlanet Uninstall window is displayed. See Figure 2-26.

Figure 2-26 iPlanet Uninstaller Window



iPlanet Uninstall

The following products are installed in this location. To remove a product, check it on the list and click Uninstall.

Path: F:\iPlanet\ias6

Components:

- ✓ Administration Services
- ✓ iPlanet Directory Suite
- ✓ Server Core Components
- ✓ nsPerl
- ✓ PerLDAP

Description

Server Administration Services used by system administrators to manage all iPlanet Servers.

2. Deselect all components for iPlanet Application Server *except* iPlanet Application Server Enterprise Connector for Tuxedo.

The components must be deselected in the following sequence.

CAUTION The uninstaller checks for dependencies between subcomponents and the main component. If you try to deselect a component and its subcomponent are toggled on, a warning message appears.

---

a. Server Core Components

b. Administration Services

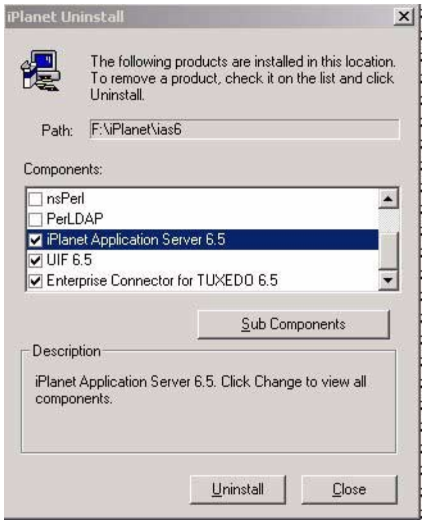
c. iPlanet Directory Suite

d. nsPerl

e. PerLDAP

3. Select iPlanet Application Server 6.5 as shown in Figure 2-27 and then press Sub Components.

Figure 2-27 iPlanet Uninstaller Window (Bottom)



The following products are installed in this location. To remove a product, check it on the list and click Uninstall.

Path: F:\iPlanet\ias6

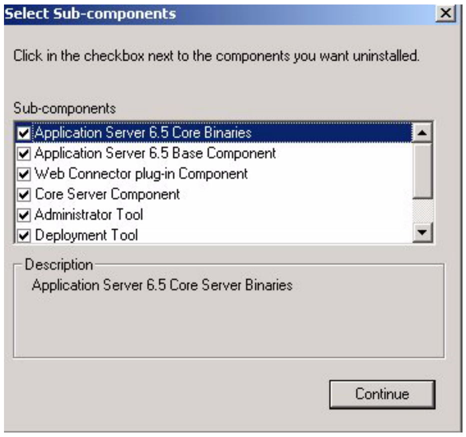
Components:

- ☐ nsPerl
- ☐ PerLDAP
- ☑ iPlanet Application Server 6.5
- ☑ UIF 6.5
- ☑ Enterprise Connector for TUXEDO 6.5

Description

iPlanet Application Server 6.5. Click Change to view all components.

4. Deselect iPlanet Application Server 6.5 Sub Components.



Click in the checkbox next to the components you want uninstalled.

Sub-components

- ☑ Application Server 6.5 Core Binaries
- ☑ Application Server 6.5 Base Component
- ☑ Web Connector plug-in Component
- ☑ Core Server Component
- ☑ Administrator Tool
- ☑ Deployment Tool

Description

Application Server 6.5 Core Server Binaries

The components of the iPlanet Application Server 6.5 must be deselected in the following sequence.

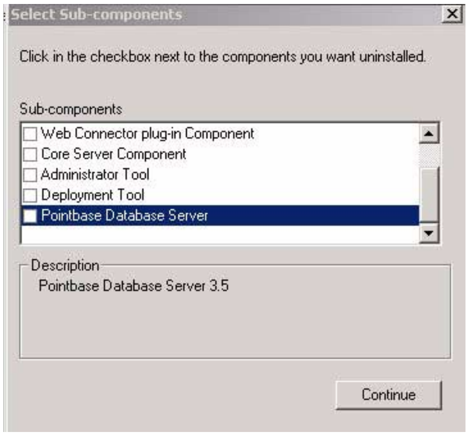
I. Application Server 6.5 Core Binaries

II. Application Server 6.5 Base Component

- **III.** Web Connector plug-in Component
- **IV.** Administrator Tool
- **V.** Core Server Component
- **VI.** Deployment Tool
- **VII.** Pointbase Database Server

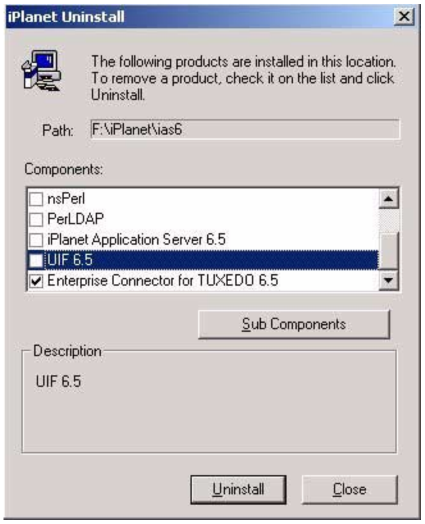
  When you have finished deselecting the check boxes, the Select Sub-components dialog box should be as shown in Figure 2-28.

Figure 2-28 Deselected iPlanet Application Server 6.5 Components


Select Sub-components
Click in the checkbox next to the components you want uninstalled.
Sub-components
Web Connector plug-in Component
Core Server Component
Administrator Tool
Deployment Tool
Pointbase Database Server
Description
Pointbase Database Server 3.5
Continue

5. Click Continue when you have finished deselecting the sub-components to redisplay the main uninstall menu.

Figure 2-29 Main uninstall


iPlanet Uninstall

The following products are installed in this location. To remove a product, check it on the list and click Uninstall.

Path: F:\iPlanet\ias6

Components:

- [ ] nsPerl
- [ ] PerLDAP
- [ ] iPlanet Application Server 6.5
- [ ] UIF 6.5
- [x] Enterprise Connector for TUXEDO 6.5

Description

UIF 6.5

6. Deselect UIF 6.5
7. Click Uninstall.

Step 2 - Deleting the Repository Content

The uninstaller does not automatically delete the repository content related to the iPlanet Application Server Enterprise Connector for Tuxedo. Use the UIF Repository browser to delete the following nodes:

- bsp.adapterTypes.Tuxedo
- bsp.dataSources.<tux_ds_name>

where `<tux_ds_name>` is name of Tuxedo datasource.

In addition to the above repository content, the uninstaller may not delete the following files and directories:

- <iAS InstallDir>/APPS/TuxBank
- <iAS InstallDir>/APPS/TuxSimpApp

Step 3 - Unregistering the Tuxedo Extension

The uninstaller does not automatically unregister the Tuxedo extension in iPlanet Registry database.

To unregister, perform the following steps:

1. Open the iPlanet Registry Editor by typing `kregedit` at the command line.
2. Open the following key:

```
SOFTWARE\iPlanet\Application Server\
6.5\CCS0\EXTENSIONS\TuxedoAdapterExt\cTuxedoAdapterExt\ENABLED
```

3. Set the value of ENABLED to 0.
4. click OK to save your changes.

Removing the Connector from Solaris

The procedure for removing the iAS Enterprise Connector for Tuxedo consists of three steps:

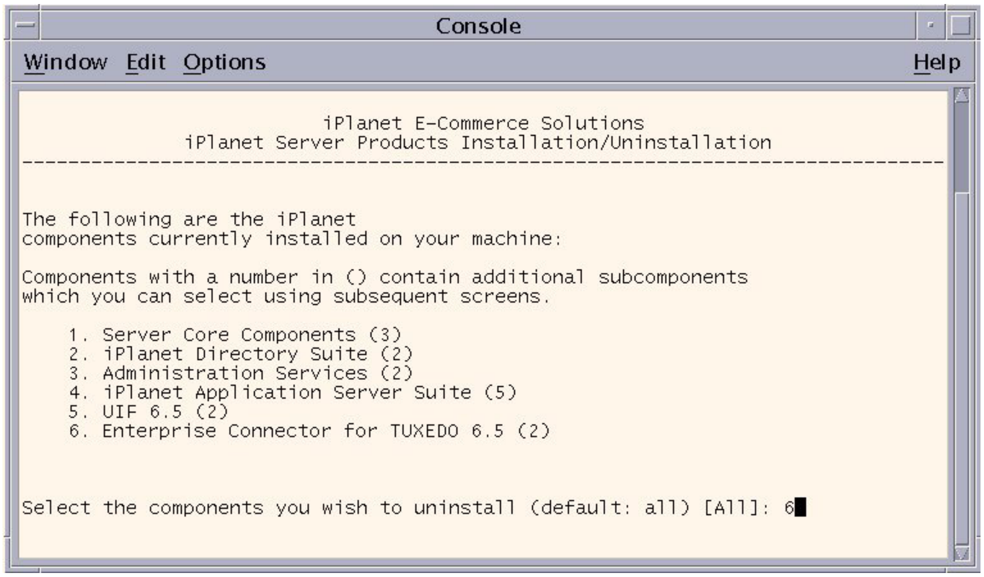
- Step 1 - Uninstallating the iAS Enterprise Connector for Tuxedo
- Step 2 - Deleting the Repository Content
- Step 3 - Unregistering the Tuxedo Extension

Step 1 - Uninstallating the iAS Enterprise Connector for Tuxedo

To uninstall the iAS Enterprise Connector for Tuxedo

1. Go to the iPlanet Application Server installation directory and type `./uninstall`. The following screen is displayed.

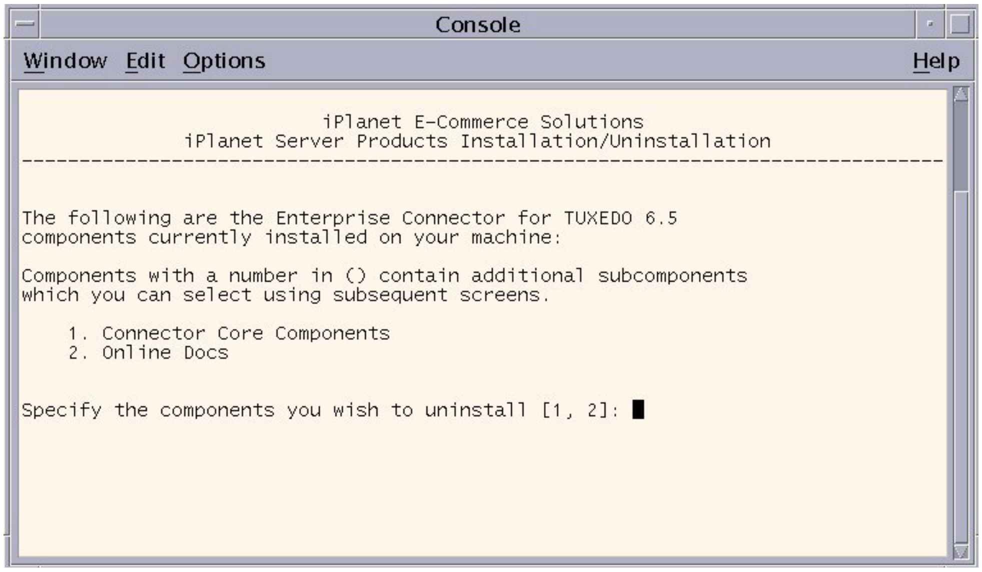
Figure 2-30 Uninstalling iAS Enterprise Connector for Tuxedo



```
                    iPlanet E-Commerce Solutions
           iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


The following are the iPlanet
components currently installed on your machine:

Components with a number in () contain additional subcomponents
which you can select using subsequent screens.

    1. Server Core Components (3)
    2. iPlanet Directory Suite (2)
    3. Administration Services (2)
    4. iPlanet Application Server Suite (5)
    5. UIF 6.5 (2)
    6. Enterprise Connector for TUXEDO 6.5 (2)



Select the components you wish to uninstall (default: all) [All]: 6
```

2. Select the Enterprise Connector for TUXEDO 6.5 entry. The following screen is displayed.

Figure 2-31 UninstallingConnector Components



```
                    iPlanet E-Commerce Solutions
           iPlanet Server Products Installation/Uninstallation
--------------------------------------------------------------------------------


The following are the Enterprise Connector for TUXEDO 6.5
components currently installed on your machine:

Components with a number in () contain additional subcomponents
which you can select using subsequent screens.

    1. Connector Core Components
    2. Online Docs


Specify the components you wish to uninstall [1, 2]:
```

3. Press return.

The iPlanet Enterprise Connector for Tuxedo and its documentation have been removed.

Step 2 - Deleting the Repository Content

The uninstaller does not automatically delete the repository content related to the iPlanet Application Server Enterprise Connector for Tuxedo. Use the UIF Repository browser to delete the following nodes:

- bsp.adapterTypes.Tuxedo
- bsp.dataSources.<tux_ds_name>

where `<tux_ds_name>` is name of Tuxedo datasource.

In addition to the above repository content, the uninstaller may not delete the following files and directories:

- <iAS InstallDir>/APPS/TuxBank
- <iAS InstallDir>/APPS/TuxSimpApp

Step 3 - Unregistering the Tuxedo Extension

The uninstaller does not automatically unregister the Tuxedo extension in iPlanet Registry database.

To unregister, perform the following steps:

1. Open the iPlanet Registry Editor that is located in <ias_root>/ias/bin.

   Typing `./kregedit` at the command line.

2. Open the following key:

   ```
   SOFTWARE\iPlanet\Application Server\
   6.5\CCS0\EXTENSIONS\TuxedoAdapterExt\cTuxedoAdapterExt\ENABLED
   ```

3. Set the value of ENABLED to 0.
4. click OK to save your changes.

Troubleshooting

If a problem occurs during installation or if you cannot run the sample application:

- Review the system configuration as described in “Hardware and Software Requirements”.
- Verify the iPlanet Application Server is installed properly.
- Verify the iPlanet Unified Integration Framework is installed properly.
- Verify the Tuxedo workstation client is installed properly.
- Verify the connection between the Tuxedo server and Tuxedo workstation is working correctly.
- Verify the installation steps were completed correctly, as described in “Installing the iAS Enterprise Connector for Tuxedo”.
- Verify the configuration steps were completed correctly, as described in “Configuring the Installation”.
- Check the *kas*, *kxs*, and *kjs* engines logs for additional information.

Review the following list for possible problems and suggested action(s):

Table 2-5 Troubleshooting Installation

Problem	**Solution**
Installer unable to detect the iPlanet Application Server or Unified Integration Framework components	Check iPlanet Application Server and Unified Integration Framework software has been installed prior to installing the iPlanet Application Server Enterprise Connector for Tuxedo.
Invalid Tuxedo Directory	Not a valid Tuxedo workstation client installation root directory. Check Tuxedo client software has been installed prior to installing the iPlanet Application Server Enterprise Connector for Tuxedo.
Insufficient memory	Close other open programs
Network connection failed	See your network administrator
Insufficient disk space	Remove unnecessary files from the target volume
Unable to load iPlanet Application Server Enterprise Connector for Tuxedo extension	Check the `LD_LIBRARY_PATH for <tuxedo /ws rootdir>/lib` directory on Solaris. Check the `path` environment variable for <tuxedo /ws rootdir>\bin directory on NT.
Unable to locate DLL	Check the `<tuxedo /ws rootdir>\bin` directory is included in `path` environment variable

Table 2-5 Troubleshooting Installation

Problem	Solution
Cannot run the sample applications	Check the iAS Enterprise Connector for Tuxedo and Tuxedo system configuration
Cannot open message catalog `LIBWSC_CAT` error message	Check the `NLSPATH` environment variable is set correctly
Tuxclient `error: WorkerProcess` Init() failed error message	Check the Tuxedo Server is up and running and `WSNADDR` is set correctly
Tuxedo Workstation client tpinit() failed error message	Check the Tuxedo Server is up and running and `WSNADDR` is set correctly

Chapter 3

The Tuxedo Management Console

This chapter describes how to configure the iPlanet Application Server Enterprise Connector for Tuxedo using the Tuxedo Management Console.

This chapter covers the following topics:

- Overview
- Launching the Tuxedo Management Console
- Configuring the Tuxedo Enterprise Connector
- Tuxedo Management Console Operations
- Read Only Nodes
- Deploying the Repository Content
- Setting Parameters for International Applications
- Jolt Migration

Overview

In order to access the Tuxedo service(s) using the iPlanet Application Server Enterprise Connector for Tuxedo, the following information must be set in the iPlanet Application Server:

- Tuxedo /T server information including:
    - The workstation listener's network address through which the Tuxedo workstation clients gain access to the Tuxedo server
    - Security level defined on the Tuxedo /T server
    - Application password, if required

   - User name, client name and user password, if required

- Tuxedo workstation client installation directory (Tuxedo /WS)
- Tuxedo services available on the Tuxedo server. Tuxedo can have many services such as ADD_EMPLOYEE, GET_EMPLOYEE, CHANGE_EMPLOYEE, and ASSIGN_PROJECT. Any or all of these service definitions can be defined in the iPlanet Application Server
- Tuxedo Field Manipulation Language (FML) field definition information, if any Tuxedo service uses FML as an input or output parameter
- Tuxedo VIEW information, if any Tuxedo service uses VIEW buffer type as an input or output parameter

The Tuxedo Management Console allows to import and store the above required information into the UIF Repository. The Tuxedo Enterprise Connector runtime uses this information to process Java-based client requests to invoke the Tuxedo services.

The Tuxedo Management Console is also used to configure connection pool parameters to fine tune the runtime performance of the overall web application.

Launching the Tuxedo Management Console

This section describes how to launch the Tuxedo Management Console. The management console is stand alone Java-based Graphical User Interface (GUI) tool used to configure the Tuxedo enterprise connector.

On Windows NT/2000

To launch the management console, select Start->Programs->iPlanet Application Server 6.5->TUXEDO Connector 6.5 - Management Console or type the following command `tuxConsole.bat` at the DOS prompt.

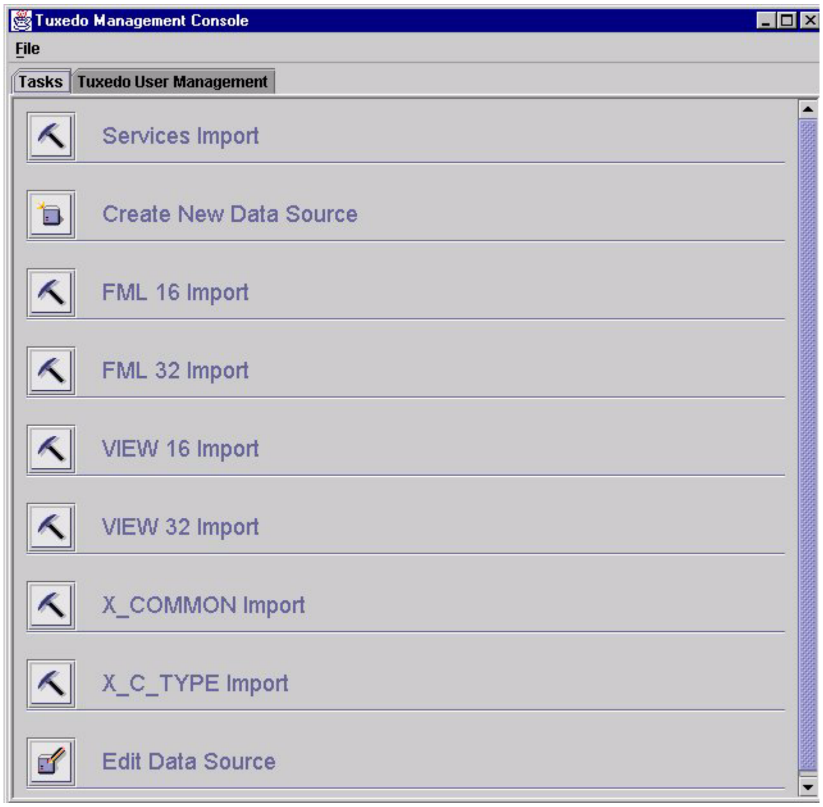
On Solaris

To launch the management console, type the following command at the prompt:

```
<ias directory>/ias/APPS/bin/tuxedoconsole.sh
```

The Tuxedo Management Console main window is displayed as shown in Figure 3-1.

Figure 3-1 Tuxedo Management Console Main Window



The management console is used to perform the following operations:

- Create a new datasource
- Edit an existing datasource
- Import Tuxedo FML field definitions
- Import Tuxedo VIEW definitions
- Import Tuxedo Service definitions
- Configure Tuxedo server authentication information
- Configure connection pool parameters

Please refer to "Tuxedo Management Console Operations," on page 98 for detailed descriptions on how to perform a specific operation.

NOTE The management console is a stand alone Java-based GUI tool which uses the iPlanet Directory Server installed with your iPlanet Application Server. The iPlanet Directory Server must be up and running to use the management console. There is no need to start the iPlanet Application Server.

Configuring the Tuxedo Enterprise Connector

This section explains basic configuration concepts, configuration parameters, a procedure to import the Tuxedo services with associated input and output parameters, and setting the connection pool parameters to fine tune the web application.

The following topics are covered:

- Tuxedo Datasources
- Tuxedo Services IDL File
- Importing Tuxedo Data Types and Services
- Connection Pooling
- Tuxedo Authentication Contexts and User Mapping

Tuxedo Datasources

A datasource in UIF represents a single EIS instance and defines the business functions (function objects) available for execution on the datasource. In the context of the Tuxedo Enterprise Connector, a datasource is mapped to the Tuxedo Server and business functions are mapped to Tuxedo services.

Multiple datasources can coexist and each datasource can be configured to connect to a separate Tuxedo server (Tuxedo /T). Also, more than one datasource can be configured to connect to the same Tuxedo server with different sets of Tuxedo services. There is no limit on the number of datasources that can be defined. The management console uses the *Create New Data Source* and *Edit Data Source* features to create a new datasource or to modify an existing datasource.

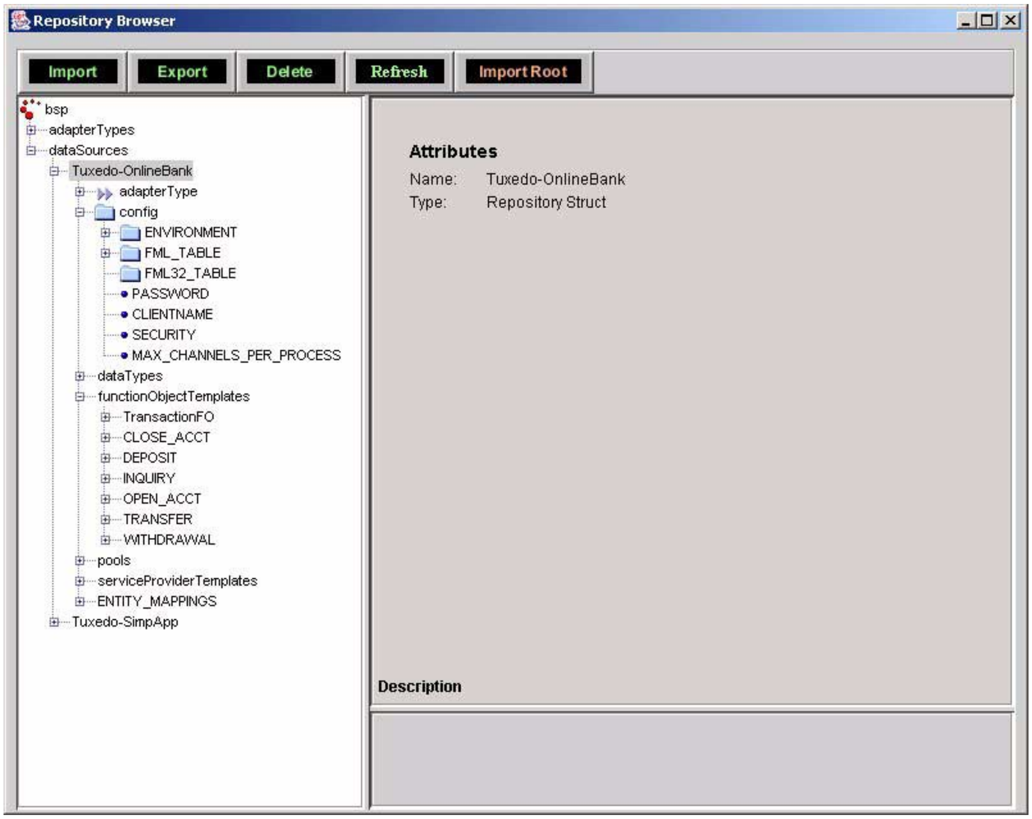
A datasource consists of the EIS configuration, data type definitions, service provider definitions, function objects, pool definitions, and user mapping information. Refer to the *iPlanet Application Server Unified Integration Framework Developer's Guide* for more information.

The Tuxedo Enterprise Connector comes with the following two preconfigured datasources:

- Tuxedo-OnlineBank
- Tuxedo-SimpApp

These datasources are configured to connect to a single Tuxedo server with different services based on information provided during installation.

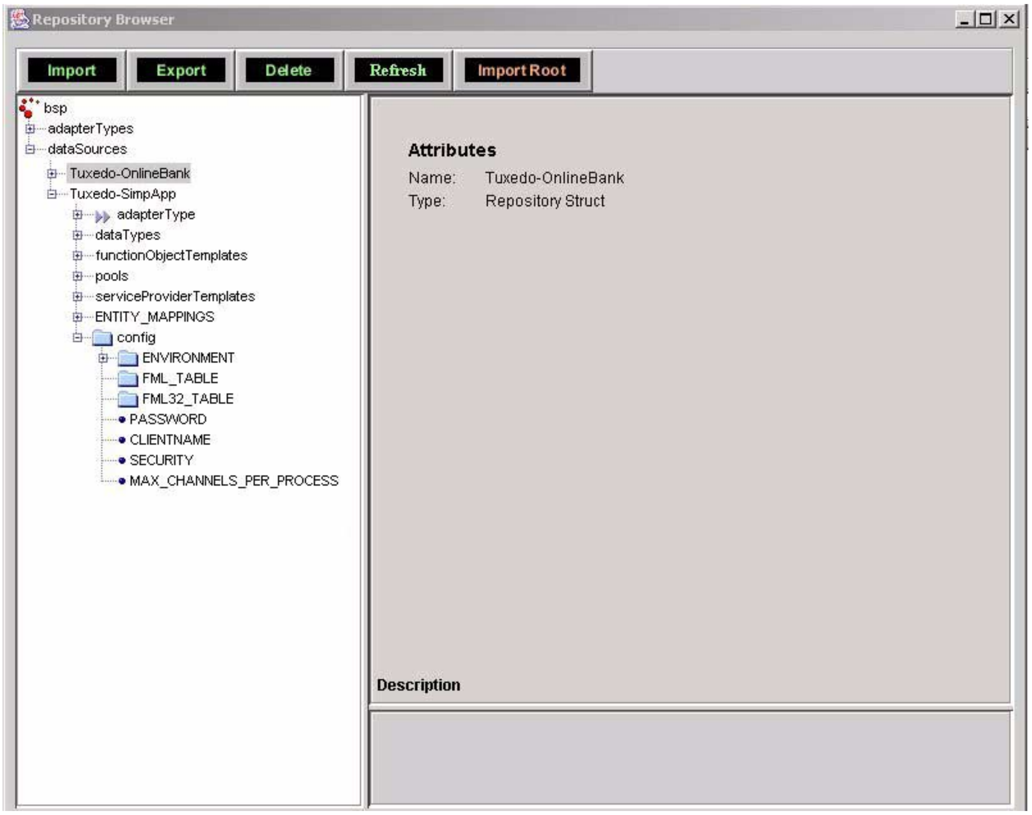
Figure 3-2 Tuxedo-OnlineBank Datasource



```
bsp
  adapterTypes
  dataSources
    Tuxedo-OnlineBank
      adapterType
      config
        ENVIRONMENT
        FML_TABLE
        FML32_TABLE
        PASSWORD
        CLIENTNAME
        SECURITY
        MAX_CHANNELS_PER_PROCESS
      dataTypes
      functionObjectTemplates
        TransactionFO
        CLOSE_ACCT
        DEPOSIT
        INQUIRY
        OPEN_ACCT
        TRANSFER
        WITHDRAWAL
      pools
      serviceProviderTemplates
      ENTITY_MAPPINGS
    Tuxedo-SimpApp
```

Name: Tuxedo-OnlineBank
Type: Repository Struct

Figure 3-3 Tuxedo-SimpApp Datasource



```
bsp
  adapterTypes
  dataSources
    Tuxedo-OnlineBank
    Tuxedo-SimpApp
      adapterType
      dataTypes
      functionObjectTemplates
      pools
      serviceProviderTemplates
      ENTITY_MAPPINGS
      config
        ENVIRONMENT
        FML_TABLE
        FML32_TABLE
        PASSWORD
        CLIENTNAME
        SECURITY
        MAX_CHANNELS_PER_PROCESS
```

Attributes

Name:	Tuxedo-OnlineBank
Type:	Repository Struct

Description

Datasource Configuration Parameters

The following parameters are associated with each datasource:

- TUXDIR
- WSNADDR
- WSDEVICE
- SECURITY
- APPLICATION PASSWORD
- CLIENTNAME
- MAX_CHANNELS_PER_PROCESS

The system prompts for parameter values while creating a new datasource using the management console. Some parameters can be modified later using the *Edit Data Source* feature.

TUXDIR

This is the Tuxedo workstation (Tuxedo /WS) installation root directory and must be installed on the same machine as the iPlanet Application Server.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dataSourceName>.config.ENVIRONMENT.TUXDIR
```

For example, *d:\tuxedows* (NT) and */export/home/tuxedows* (Unix).

WSNADDR

Specifies the workstation listener process network address through which the Tuxedo Enterprise Connector's workstation clients gain access to the Tuxedo server. WSNADDR must be specified in one of the following formats:

```
//host_name:port_number
```

```
//#.#.#.#:port_number
```

You can specify more than one address by including a comma separated list of pathnames for WSNADDR. You can use this advanced syntax to take advantage of Tuxedo server load balancing and fail over functionality. Please refer BEA Tuxedo documentation for additional information.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dataSourceName>.config.ENVIRONMENT.WSNADDR
```

For example, *//horse:9999* and *//192.18.117.141:9999.*

WSDEVICE

Specifies the device used to access the network. It should be set to */dev/tcp* for TCP/IP or left blank.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dataSourceName>.config.ENVIRONMENT.WSDEVICE
```

For example, */dev/tcp.*

SECURITY

Specifies the security level defined on the Tuxedo server. The BEA Tuxedo server provides several levels of security, as follows:

- NONE

- APP_PW
- USER_AUTH
- ACL
- MANDATORY_ACL

The Tuxedo administrator configures the security level. Consult your Tuxedo administrator for the security level setting. Table 3-1 shows the security level mappings..

Table 3-1 Security Level Mapping

Tuxedo Security Level	Security Parameter Value
NONE	TPNOAUTH
APP_PW	TPSYSAUTH
USER_AUTH	TPAPPAUTH
ACL	TPAPPAUTH
MANDATORY_ACL	TPAPPAUTH

- NONE

  No additional information is needed to connect a workstation client to the Tuxedo server.

- APP_PW

  This security level requires an application password which is placed in the *passwd* field of the TPINIT buffer when a workstation client connects to the Tuxedo server.

- USER_AUTH, ACL, and MANDATORY_ACL

  These security levels requires application password and user specific data (user name, client name, and user password) to connect to the Tuxedo server.

  The parameters are placed in the *passwd, usrname, cltname, data*, and *datalen* fields of the TPINIT buffer when a workstation client connects to the Tuxedo server.

  Refer to “User Mapping,” on page 118 for details on how to configure user specific data.

This information is stored in the UIF Repository under the following node:

`bsp.dataSources.<dataSourceName>.config.SECURITY`

For example, *TPNOAUTH*, *TPSYSAUTH*, and *TPAPPAUTH*.

APPLICATION PASSWORD

Specifies the application password and is required when the SECURITY level is set to either TPSYSAUTH or TPAPPAUTH. The setting is ignored with security level TPNOAUTH.

This information is stored in the UIF Repository under the following node:

`bsp.dataSources.<dataSourceName>.config.PASSWORD`

For example, *iastux*.

CLIENTNAME

Specifies the client name. The parameter is set to *iastuxadapter* by default and cannot be modified. The parameter is used only when the security level is set to TPNOAUTH or TPSYSAUTH. The parameter is passed in the *cltname* field of the TPINIT buffer when a client joins the Tuxedo system. The Tuxedo administrator can easily identify the list of all enterprise connector clients who have joined the Tuxedo server and is used for administration purposes.

When the security level is set to TPAPPUTH the client name is defined as part of the user mappings (Tuxedo Authentication) instead of the CLIENTNAME parameter. Refer to "Connection Pooling," on page 91 for details.

This information is stored in the UIF Repository under the following node:

`bsp.dataSources.<dataSourceName>.config.CLIENTNAME`

For example, *iastuxadapter*.

MAX_CHANNELS_PER_PROCESS

The maximum concurrent channels per Tuxedo worker process and must be greater than or equal to 1, and less than or equal to 50. By default its set to 50. Refer to "Connection Pooling," on page 91 for details.

This information is stored in the UIF Repository under the following node:

`bsp.dataSources.<dataSourceName>.config.MAX_CHANNELS_PER_PROCESS`

For example, *50*.

Tuxedo Services IDL File

BEA Tuxedo doesn't provide an API for querying services available on a Tuxedo server. Therefore, the management console cannot automatically discover and import them into the UIF Repository. The definition of these Tuxedo services must be specified manually. The enterprise connector defines a DCE IDL-based file format which allows users to define the Tuxedo services and their associated parameters. The management console uses the IDL files and loads them into the UIF Repository.

IDL File Syntax

The IDL file has three main components:

- Interface
- Services
- Parameters

The file name containing the interface definition have a `.idl` suffix and using the convention makes the file type self documenting.

Interface

The interface is the first component and wraps around the other two. The interface begins with the word *interface* followed by the interface name, an open brace, service definition, a closed brace, and a semicolon to signify closure. Multiple service definitions can be defined within a single interface.

```
interface InterfaceName{
    <services>
        <<parameter>>
};
```

Services

A Service contains the word *void*, the Tuxedo service name, an open parenthesis, the input and output parameters, a closed parenthesis and a semicolon to signify closure. Multiple parameters can be defined for a single service.

```
void ServiceName(
   <parameters>
);
```

Parameters

Parameters contain the in or out keyword, the Tuxedo buffer type, parameter name, and any buffer type specific information.

The parameter block for Tuxedo buffer types STRING, CARRAY, and X_OCTET is defined as follows:

```
[in|out] TYPE PARAM_NAME

TYPE : STRING | CARRAY | X_OCTET
PARAM_NAME : Valid Parameter Name
```

The parameter block for Tuxedo buffer types FML and FML32 is defined as follows:

```
[in|out, FML] TYPE PARAM_NAME
[in|out, FML32] TYPE PARAM_NAME

TYPE : CHAR | STRING | SHORT | LONG | FLOAT | DOUBLE | CARRAY
PARAM_NAME : FML_FIELD_NAME | FML_FIELD_NAME[]

Note:
FML_FIELD_NAME[] format is used to specify the multiple occurence
attribute of FML field.
```

The parameter block for Tuxedo buffer types VIEW, VIEW32, X_COMMON, and X_C_TYPE is defined as follows:

```
[in, TYPE VIEW_NAME] TYPE inputBlock
[out, TYPE VIEW_NAME] TYPE outputBlock

TYPE : VIEW16 | VIEW32 | X_COMMON | X_C_TYPE
VIEW_NAME : Tuxedo View Name

Note:
in must follow inputBlock
out must follow outputBlock
view name - name follows after VIEW clause in Tuxedo view
definition file
```

Sample Services IDL Files

The following is a list of sample services IDL files:

Sample 1

The following listing shows a sample IDL file to demonstrate how-to use STRING, CARRAY, and X_OCTET Tuxedo buffer types as input and output parameters:

```
interface sample1 {

     void STRINGTEST1 (
          [in]  STRING inParam
          [out] STRING outParam
     );

     void STRINGTEST2 (
          [out] STRING outParam
     );

     void STRINGTEST3 (
          [in] STRING inParam
     );


     void CARRAYTEST1 (
          [in]  CARRAY inParam
          [out] CARRAY outParam
     );

     void CARRAYTEST2 (
          [out] CARRAY outParam
     );

     void CARRAYTEST3 (
          [in] CARRAY inParam
     );

     void XOCTETTEST1 (
          [in]  X_OCTET inParam
          [out] X_OCTET outParam
     );

     void XOCTETTEST2 (
          [out] X_OCTET outParam
     );

     void XOCTETTEST3 (
          [in] X_OCTET inParam
     );
};
```

Sample 2

The following listing shows a sample IDL file to demonstrate how-to use FML and FML32 Tuxedo buffer types as input and output parameters:

```
interface sample2 {

   void FML16TEST1 (
        [in, FML]  STRING string16
        [out,FML]  STRING string16[]
    );

    void FML16TEST2 (
        [out,FML]  CHAR char16[]
    );

    void FML16TEST3 (
        [in, FML]  STRING string16
        [in, FML]  SHORT  short16
        [in, FML]  DOUBLE double16
        [in, FML]  FLOAT  float16
        [in, FML]  LONG   long16
    );


   void FML32TEST1 (
        [in, FML32]  STRING string32
        [out,FML32]  STRING string32[]
    );

    void FML32TEST2 (
        [out,FML32]  CARRAY carray32[]
    );

    void FML32TEST3 (
        [in, FML32]  STRING string32
        [in, FML32]  SHORT  short32
        [in, FML32]  DOUBLE double32
        [in, FML32]  FLOAT  float32
        [in, FML32]  LONG   long32
    );
};
```

The above sample uses the following FML field definitions:

```
// FML 16
# name        number  type    flags   comments
*base 1000
char16         1       char    -       this is a char field
string16       2       string  -       this is a string field
short16        3       short   -       this is a short field
long16         4       long    -       this is a long field
float16        5       float   -       this is a float field
double16       6       double  -       this is a double field
carray16       7       carray  -       this is a carray field

// FML32
# name        number  type    flags   comments
*base 2000
char32         1       char    -       this is a char field
string32       2       string  -       this is a string field
short32        3       short   -       this is a short field
long32         4       long    -       this is a long field
float32        5       float   -       this is a float field
double32       6       double  -       this is a double field
carray32       7       carray  -       this is a carray field
```

Sample 3

The following listing shows a sample IDL file to demonstrate how-to use VIEW16, VIEW32, X_COMMON, and X_C_TYPE Tuxedo buffer types as input and output parameters:

```
interface sample3 {
     void VIEW16TEST1 (
         [in,  VIEW16  v16test1]  VIEW16 inputBlock
         [out, VIEW16  v16test1]  VIEW16 outputBlock
     );
     void VIEW16TEST2 (
         [in,  VIEW16  v16test1]  VIEW16 inputBlock
         [out, VIEW16  v16test2]  VIEW16 outputBlock
     );
     void VIEW16TEST3 (
         [in, VIEW16  v16test4]  VIEW16 inputBlock
     );
     void VIEW16TEST4 (
         [out, VIEW16  v16test4]  VIEW16 outputBlock
     );

     void VIEW32TEST1 (
         [in,  VIEW32  v32test1]  VIEW32 inputBlock
         [out, VIEW32  v32test1]  VIEW32 outputBlock
     );
     void VIEW32TEST2 (
         [in,  VIEW32  v32test1]  VIEW32 inputBlock
         [out, VIEW32  v32test2]  VIEW32 outputBlock
     );
     void VIEW32TEST3 (
         [in, VIEW32  v32test4]  VIEW32 inputBlock
     );
     void VIEW32TEST4 (
         [out, VIEW32  v32test4]  VIEW32 outputBlock
     );

     void XCOMMONTEST1 (
         [in,  X_COMMON xcomtest1]  X_COMMON inputBlock
         [out, X_COMMON xcomtest2]  X_COMMON outputBlock
     );

     void XCTYPETEST1 (
         [in,  X_C_TYPE xctest1]  X_C_TYPE inputBlock
         [out, X_C_TYPE xctest2]  X_C_TYPE outputBlock
     );
};
```

The above sample uses the following VIEW definitions (only few elements are shown for each view for simplicity):

```
VIEW v16test1
# type cname   fbname  count flag  size null
string str1    -       1     -     100  -
END
VIEW v16test2
# type cname   fbname  count flag  size null
double double1 -       5     -     -    -
END
VIEW v16test4
# type cname   fbname  count flag  size null
string str1    -       1     -     100  "UIF_TUX_STRING_NULL"
END
VIEW v32test1
# type cname   fbname  count flag  size null
char   char1   -       1     -     -    -
long   long1   -       1     -     -    -
END
VIEW v32test2
# type cname   fbname  count flag  size null
int    int1    -       5     -     -    -
dec_t  dec1    -       5     -     9,2  -
END
VIEW v32test4
# type cname   fbname  count flag  size  null
double double1 -       1     -     -     987.654
END
VIEW xcomtest1
# type cname   fbname  count flag  size null
string str1    -       1     -     100  -
END
VIEW xcomtest2
# type cname   fbname  count flag  size null
long   long1   -       10     -     -    -
END
VIEW xctest1
# type cname   fbname  count flag  size null
short  short1  -       1     -     -    -
END
VIEW xctest2
# type cname   fbname  count flag  size null
char   char1   -       10     -     -    -
END
```

Sample 4

Sample `.idl` file for services defined in the Tuxedo OnlineBank and Simpapp sample applications:

```
interface sample4
{
    void TOUPPER(
        [in]    STRING STRING
        [out]   STRING STRING
    );

    void CLOSE_ACCT(
        [in,FML]    LONG   ACCOUNT_ID
        [out,FML]   STRING SBALANCE
    );
    void DEPOSIT(
        [in,FML]    STRING SAMOUNT
        [in,FML]    LONG ACCOUNT_ID
        [out,FML]   STRING SBALANCE
    );
    void INQUIRY(
        [in,FML]    LONG   ACCOUNT_ID
        [out,FML]   STRING SBALANCE
    );
    void OPEN_ACCT(
        [in,FML]    STRING  SAMOUNT
        [in,FML]    STRING  LAST_NAME
        [in,FML]    STRING  FIRST_NAME
        [in,FML]    CHAR MID_INIT
        [in,FML] CHAR ACCOUNT_TYPE
        [in,FML]    STRING  ADDRESS
        [in,FML]    STRING  PHONE
        [in,FML]    STRING  SSN
        [in,FML]    LONG    BRANCH_ID
        [out,FML]   LONG    ACCOUNT_ID
        [out,FML]   STRING  SBALANCE
    );
    void TRANSFER(
        [in,FML]    LONG   ACCOUNT_ID[]
        [in,FML]    STRING SAMOUNT
        [out,FML]   STRING SBALANCE[]
    );
    void WITHDRAW(
        [in,FML]    LONG   ACCOUNT_ID
        [in,FML]    STRING SAMOUNT
        [out,FML]   STRING SBALANCE
    );
};
```

The Online Bank Services sample uses the following FML field definitions:

```
# Fields for database bankdb
# name                          number  type    flags   comments
ACCOUNT_ID                      110     long    -       -
ACCT_TYPE                       112     char    -       -
ADDRESS                         109     string  -       -
AMOUNT                          117     float   -       -
BALANCE                         105     float   -       -
BRANCH_ID                       104     long    -       -
FIRST_NAME                      114     string  -       -
LAST_ACCT                       106     long    -       -
LAST_NAME                       113     string  -       -
LAST_TELLER                     107     long    -       -
MID_INIT                        115     char    -       -
PHONE                           108     string  -       -
SSN                             111     string  -       -
TELLER_ID                       116     long    -       -
#
# non database fields
SBALANCE                        201     string  -       -
SAMOUNT                         202     string  -       -
XA_TYPE                         203     short   -       -
CURS                            204     string  -       -
SVCHG                           205     string  -       -
VIEWNAME                        206     string  -       -
OPEN_CR                         207     char    -       -
TYPE_CR                         208     char    -       -
STATLIN                         209     string  -       -
```

Importing Tuxedo Data Types and Services

This section describes the basic procedure or steps to import the Tuxedo data types and services into UIF Repository. The metadata describing the Tuxedo data types and services must be present in the UIF Repository prior to starting the web application development using the iAS Enterprise Connector for Tuxedo.

To import Tuxedo data types and services

1. Create a new datasource

   Create a new datasource using the *Create New Data Source* option provided with the management console. Refer to “Create New Data Source” for detailed information.

   The management console must be restarted in order to use the newly created datasource.

2. Create Services IDL file

   Create a Tuxedo services IDL file (as described in “Tuxedo Services IDL File”) which contains the Tuxedo services definitions to be imported into the UIF Repository.

   The enterprise connector supports STRING, CARRAY, X_OCTET, FML, FML32, VIEW, VIEW32, X_COMMON, and X_C_TYPE Tuxedo buffer types. Any Tuxedo service definition which uses these buffer types can be included. There is no limit on the number of services that can be included in an IDL file. Custom buffer types are not supported.

3. Import the FML field table definitions

   This step is mandatory, if any Tuxedo service uses an FML buffer type. Use the *FML 16 Import* feature in the management console to perform this step (refer to FML 16 Import).

   An FML buffer is a self defining Tuxedo data type where each data field carries its own identifier, an occurrence number, and possibly a length indicator.

   The FML field table file must be specified as an input to this step. This is the text file created during the Tuxedo services design and development stage, and contains the FML fields definition. This is the same file given as an input to `mkfldhdr` to create a header file included in the Tuxedo service routine or client programs. There are no restrictions on the number of fields defined in a single file. Create a single file containing the FML field definitions used in all services. Because it is a text file the file can be used across all platforms.

   This information is stored under the following node:

   ```
   bsp.dataSources.<dsName>.config.FML_TABLE
   ```

   The node contents must match the header file contents generated using `mkfldhdr` during the Tuxedo service(s) development stage.

The following listing shows a sample FML field table file:

```
# name         number  type    flags   comments
*base 3000
EMP_EMPNO       1      long    -       -
EMP_NAME        2      string  -       -
EMP_SSN         3      string  -       -
EMP_PHONE       4      string  -       -
EMP_DEPT        5      long    -       -
EMP_SALARY      6      float   -       -
```

4. Import FML32 field table definitions

This step is mandatory, if any Tuxedo service uses the FML32 buffer type. Use the *FML 32 Import* feature in the management console to perform this step (refer to FML 32 Import).

The FML32 data type is similar to FML but allows for larger character fields, more fields, and larger buffers.

The FML32 field table file must be specified as an input to this step. This is the text file created during the Tuxedo services design and development stage, and contains the FML32 fields definition. This is the same file given as an input to `mkfldhdr32` to create a header file included in the Tuxedo service routine or client programs. There are no restrictions on the number of fields defined in a single file. Create a single file containing the FML32 field definitions used in all services. Because it is a text file the file can be used across all platforms.

This information is stored under the following node:

`bsp.dataSources.<dsName>.config.FML32_TABLE`

The node contents must match the header file contents generated using `mkfldhdr32` during the Tuxedo service(s) development stage.

The following listing shows a sample FML32 field table file:

```
# name        number  type    flags   comments
*base 2000
char32         1       char    -       this is a char field
string32       2       string  -       this is a string field
short32        3       short   -       this is a short field
long32         4       long    -       this is a long field
float32        5       float   -       this is a float field
double32       6       double  -       this is a double field
carray32       7       carray  -       this is a carray field
```

5. Import VIEW data types

This step is mandatory, if any Tuxedo services uses the VIEW buffer type. A view object file or view compiled file must be specified as an input to this step. Compiled view files are platform specific and must be generated on the same system the enterprise connector is on. Use the *VIEW 16 Import* feature in the management console to perform this step (refer to VIEW16 Import).

VIEW is a built in Tuxedo buffer type provided to use C structures and COBOL records with the Tuxedo server. VIEW type buffers must have sub types, which designate individual data structures.

To use VIEW buffers, the C structure or COBOL record is first defined in a view file created with a text editor. The each structure or records description is called a view and more than one view can be defined in a single view file. Each view is given a name. The view file is converted into an object format by the view compiler (*viewc*) which takes the view file as an input file. The view compiler generates a compiled view file, and a header file to include in the service routine and client applications. The Tuxedo runtime server uses this compiled view file. Compiled view files will have a `VV` extension on Windows NT and `V` on Unix systems.

This information is stored under the following node:

`bsp.dataSources.<dsName>.dataTypes.VIEW_TABLE`

The following listing shows a sample VIEW file (*test_view.txt*):

```
VIEW v16test1
# type cname   fbname  count flag  size null
char   char1   -       1     -     -    -
string str1    -       1     -     100  -
carray cary1   -       1     -     100  -
long   long1   -       1     -     -    -
short  short1  -       1     -     -    -
int    int1    -       1     -     -    -
float  float1  -       1     -     -    -
double double1 -       1     -     -    -
dec_t  dec1    -       1     -     9,2  -
END
```

Run the `viewc -n test_view.txt` command to generate a view object file `test_view.VV` (Windows NT) or `test_view.V` (Unix) file is the input to this step.

6. Import VIEW32 data types

This step is mandatory, if any Tuxedo services uses the VIEW32 buffer type. A view32 compiled file or view32 object file must be specified as an input to this step. Compiled view files are platform specific and must be generated on the same system the enterprise connector is on. Use the *VIEW 32 Import* feature in the management console to perform this step (refer to VIEW 32 Import).

The VIEW32 buffer type is similar to VIEW but allows for character fields, more fields, and larger overall buffer sizes.

Compiled files are generated by calling the view32 compiler (*viewc32*) on a view32 definition file.

This information is stored under the following node:

`bsp.dataSources.<dsName>.dataTypes.VIEW32_TABLE`

7. Import X_COMMON data types

   This step is mandatory, if any Tuxedo services uses the X_COMMON buffer type. The view object file or view compiled file must be specified as an input to this step. Use the *X_COMMON Import* feature in the management console to perform this step (refer to X_COMMON Import). Compiled view files are system specific and must be generated on the same system the enterprise connector is on.

   The X_COMMON buffer type is similar to VIEW but with field types limited to short, long, and string.

   Compiled files are generated by calling the view compiler *(viewc)* on a X_COMMON definition file.

   This information is stored under the following node:

   ```
   bsp.dataSources.<dsName>.dataTypes.X_COMMON_TABLE
   ```

8. Import X_C_TYPE data types

   This step is mandatory, if any Tuxedo services uses the X_C_TYPE buffer type. The view object file or view compiled file must be specified as an input to this step. Use the *X_C_TYPE Import* feature in the management console to perform this step (refer to X_C_TYPE Import). Compiled view files are platform specific and must be generated on the same system the enterprise connector is on.

   The X_C_TYPE buffer type is equivalent to VIEW.

   Compiled files are generated by calling the view compiler (*viewc*) on a X_C_TYPE definition file.

   This information is stored under the following node:

   ```
   bsp.dataSources.<dsName>.dataTypes.X_C_TYPE_TABLE
   ```

9. Import Tuxedo Services

   You must use the IDL file created in Step 2 as the input. This step populates the Tuxedo services definitions in IDL file into the UIF Repository. Use the *Services Import* feature in the management console to perform this step (refer to Import Tuxedo Services).

   Make sure that all dependent data types are imported before performing this step.

   Each service is mapped to a Function Object under the following node:

   ```
   bsp.dataSources.<dsName>.functionObjectTemplates
   ```

10. Verify the UIF Repository Content

Verify the UIF Repository content using the UIF repository browser. Give special attention to the input and output data blocks defined for each function object and configuration parameters, there should be no unresolved nodes. The unresolved nodes are marked with *X* in RED. Generally, unresolved nodes occur when imported services IDL file without importing the dependent data types, for example, FML and VIEW information.

NOTE The management console uses *<iAS InstallDir>/ias/APPS* directory to write the temporary files generated during the import process. These files are automatically deleted at the end of import process. In order to perform successfully the console operations, the user must have read-write permissions to the above directory.

The metadata is stored in iPlanet Directory Server used with your instance of iPlanet Application Server. You must re-start the iPlanet Application Server to use this data by any Java application.

Connection Pooling

This section describes the pooling concepts and parameters used to configure the enterprise connector. Pooling allows resource sharing with the enterprise connector. Pools are used to share the workstation client connections to the Tuxedo server. The characteristics associated with a pool determines, the number of Tuxedo client worker processes needed and the load on each Tuxedo client worker process. Pools are defined on a per datasource basis. In other words, each datasource defines its own connection pool(s) which can be individually configured using management console.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dataSourceName>.pools.<poolName>
```

The enterprise connector can support more than one connection pool per datasource. A pool named *default* is automatically created with a new datasource, for example *bsp.dataSources.<dataSourceName>.pools.default*.

The number of connection pools defined for a datasource is dependent on the security level setting. For security level, TPNOAUTH (NONE) and TPSYSAUTH (APP_PW) only one pool can be defined. For security level TPAPPAUTH (USER_AUTH, ACL, and MANDATORY_ACL) more than one pool can be specified depending on the user mapping settings. Refer to “Tuxedo Authentication Contexts and User Mapping,” on page 96 for details.

Pool Parameters

The following characteristics are associated with each pool and are configurable:

MaxPoolSize

Maximum number of concurrent requests to be processed.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dsName>.pools.<poolName>.config.MaxPoolSize
```

Default value: *50*

SteadyPoolSize

This should always be the same as MaxPoolSize. The kjs engine fails to load the Tuxedo Enterprise Connector, if this value differs from MaxPoolSize. Any changes to MaxPoolSize must be reflected immediately to SteadyPoolSize.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dsName>.pools.<poolName>.config.SteadyPoolSize
```

Default value: *50*

MaxWait

The maximum time, in seconds, a request waits in the queue for a connection to Tuxedo server before it times out.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dsName>.pools.<poolName>.config.MaxWait
```

Default value: *600*

UnusedMaxLife

The maximum time, in seconds, that a connection remains unused in the pool. Currently the Tuxedo Enterprise Connector requires a higher value to be set for this parameter. On creation of pool, this value is set to 300000. Please ensure that UnuxedMaxLife is always set to either 300000 or above.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dsName>.pools.<poolName>.config.UnusedMaxLife
```

Default value: *300000*

MonitorInterval

The time in seconds, that the monitor thread is executed to monitor the current status of the pool. Default is 30 seconds. Typically, setting this number too low degrades performance.

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dsName>.pools.<poolName>.config.MonitorInterval
```

Default value: *30*

DebugLevel

Determines the message logging type, as follows:

0 - Logging turned off

1 - Logs only callback messages

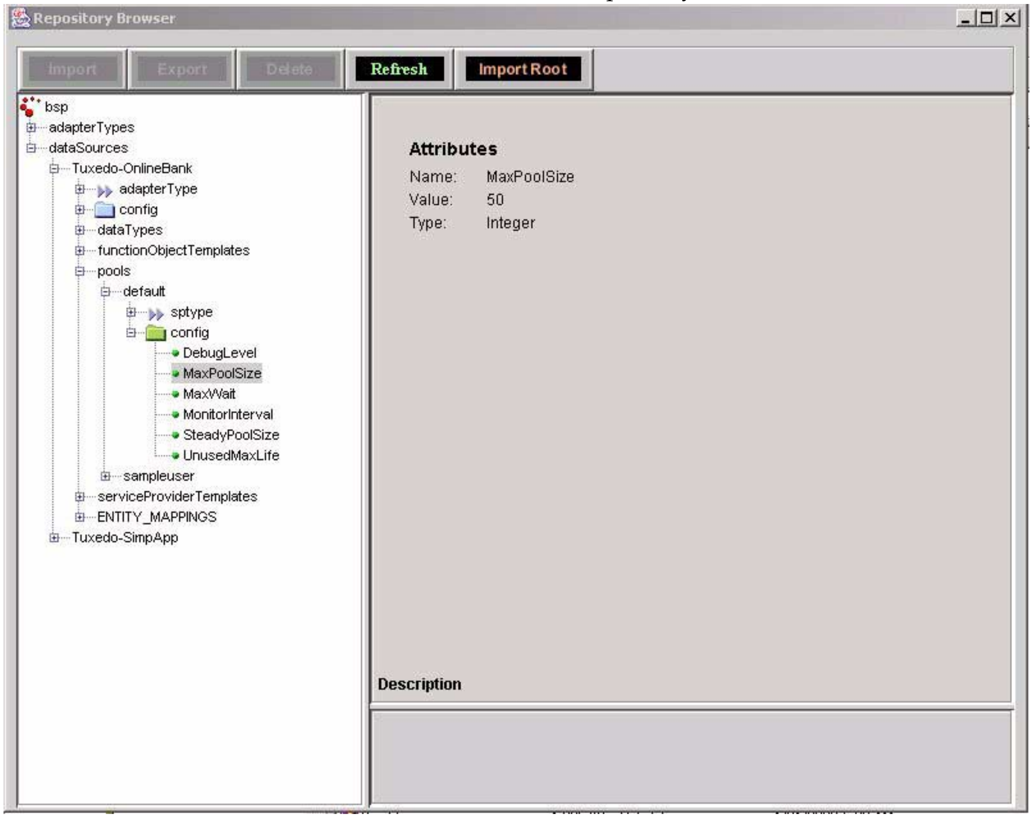
2 - Logs all messages

This information is stored in the UIF Repository under the following node:

```
bsp.dataSources.<dsName>.pools.<poolName>.config.DebugLevel
```

Default value: *0*

Figure 3-4 Pool Parameters in the UIF Repository



Attributes

Name:	MaxPoolSize
Value:	50
Type:	Integer

The value of MaxPoolSize in conjunction with, the value of MAX_CHANNELS_PER_PROCESS defined with the datasource determine the number of Tuxedo client worker processes needed for each pool and loads on each Tuxedo client process. The enterprise connector automatically spawns the necessary number of Tuxedo client processes (MaxPoolSize and MAX_CHANNELS_PER_PROCESS) and load balances the requests across the Tuxedo client worker processes. As shown in the next three examples:

1. A default configuration:

   MaxPoolSize: 50

   SteadyPoolSize: 50

   MAX_CHANNELS_PER_PROCESS: 50

   The enterprise connector spawns one (50/50) Tuxedo client worker process to support 50 non-transactional requests.

2. An advanced configuration:

   `MaxPoolSize: 100`

   `SteadyPoolSize: 100`

   `MAX_CHANNELS_PER_PROCESS: 50`

   The enterprise connector spawns two (100/50) Tuxedo client worker processes to support 100 non-transactional requests. All incoming requests are automatically load balanced across the worker processes.

The above configurations give better performance when all client requests are non-transactional. In other words, none of the requests received by the enterprise connector invokes either BEGIN or COMMIT/ROLLBACK transactional operations provided by the *TransactionFO* function object. In this mode, the enterprise connector, utilizes the Tuxedo server feature "*maximum up to 50 pending responses for non-transactional asynchronous requests allowed per* `tpinit`" and gives better throughput.

Only one concurrent transactional request is allowed per Tuxedo client worker process (`tpinit`) by the Tuxedo server. If the client requests are transactional, or a combination of transactional and non-transactional, a lower value for MAX_CHANNELS_PER_PROCESS is recommended for better throughput. The lower value for MAX_CHANNELS_PER_PROCESS for the same MaxPoolSize, forces a higher number of Tuxedo client worker process and better transactional requests throughput.

3. A more advanced configuration:

   ```
   MaxPoolSize: 100

   SteadyPoolSize: 100

   MAX_CHANNELS_PER_PROCESS: 10
   ```

   The enterprise connector spawns ten (100/10) Tuxedo client worker processes to support a mixture of transactional and non-transactional requests. All incoming requests are automatically load balanced across the worker processes.

Tuxedo Authentication Contexts and User Mapping

The enterprise connector supports multiple Tuxedo authentication contexts. Security level, user mapping, and connection pooling are used to define the Tuxedo workstation client connection behavior. This section is applicable only if the Tuxedo server is configured with security level TPAPPAUTH (USER_AUTH, ACL, or MANDATORY_ACL).

These security levels provide an authentication service that checks for the combination of user name, client name, and user password before the client is allowed to attach to the Tuxedo server. The authentication context information is passed through the TPINIT buffer when a Tuxedo workstation client joining the Tuxedo system. Multiple authentication contexts (users) can exist for a given Tuxedo server but only one authentication context is allowed per `tpinit` process.

A Tuxedo authentication context is defined as using a user name, client name and user password. The definition of these parameters are as follows:

User Name

A user name is assigned by the Tuxedo administrator and is mapped to *usrname* member in the TPINIT buffer. This name may be the same used by the operating system.

Client Name

Client name whose semantics are application defined and is mapped to *cltname* member in the TPINIT buffer. Additionally, it can be used for administration purpose or left blank.

User Password

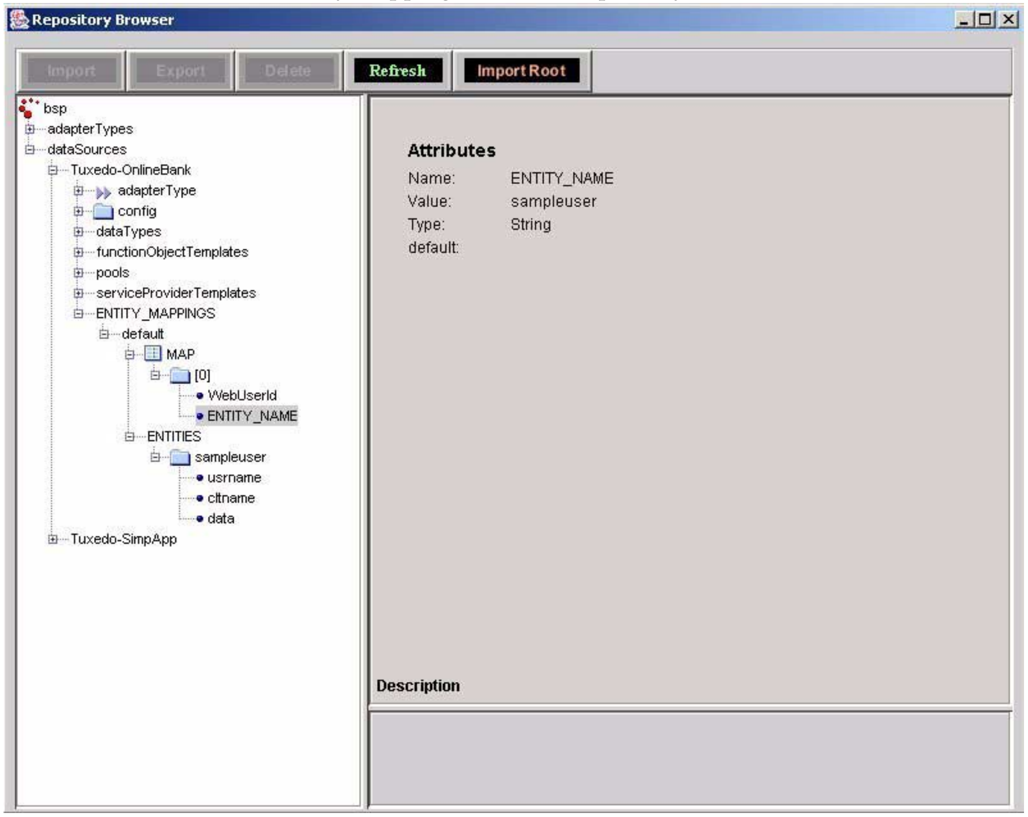
Password for the user/client name combination and is mapped to *data* and *datalen* members in the TPINIT buffer.

Normally, the Tuxedo server administrator uses `tpaddusr`, `tpmodusr`, and `tpdelusr` commands to configure the Tuxedo authentication contexts on a Tuxedo server. Refer to BEA Tuxedo documentation for further details.

The Tuxedo authentication context (user) is mapped to an Entity, as shown in Figure 3-5, in the enterprise connector and uses the following UIF Repository nodes to store the information:

```
bsp.dataSources.<dsName>.ENTITY_MAPPINGS.default.ENTITIES.<entityName>.usrName
bsp.dataSources.<dsName>.ENTITY_MAPPINGS.default.ENTITIES.<entityName>.cltName
bsp.dataSources.<dsName>.ENTITY_MAPPINGS.default.ENTITIES.<entityName>.data
```

Figure 3-5 Entity Mapping in the UIF Repository



Name:	ENTITY_NAME
Value:	sampleuser
Type:	String
default:	

The *Tuxedo User Mapping* feature (refer to *User Mapping*) in the management console is used to define each entity per datasource. The enterprise connector supports an unlimited number of entities per datasource.

A separate pool is created with each entity because only one authentication context is allowed per `tpinit` process. This feature allows an enterprise connector to spawn separate client worker processes to process the client requests for this authentication context.

The pool definitions are found under the following node and is configurable:

```
bsp.dataSources.<dataSourceName>.pools
```

The web user must pass the authentication context either directly or indirectly with each request. Passing the information directly with each web request is a tedious process and error prone. Any changes to the authentication context must be sent to all web users on the system.

To simplify the process, UIF supports *WebUser Entity Mapping*, which allows the enterprise connector administrator to define each web user and assigns an entity (authentication context) per web user. The web user ID is passed with each request instead of the authentication context. Therefore, any change to the authentication context can be carried transparently without effecting the web users.

The application programmer assigns a *WebUserId* to the enterprise connector and is used to determine the authentication context to be used to process each request. A user pool exists with each context. The enterprise connector uses the pool associated with the context to process the request.

The management console provides a GUI-based interface to support creation of Entities, Web Users and associating an entity with each web user.

Tuxedo Management Console Operations

The operations that can be performed using the management console are divided into the following two groups:

- Tasks used to create and update the datasource, import Tuxedo data types, and services.
- User management tasks that establishes the relationship between the web and Tuxedo users.

The main management console window contains two task tabs named, *Tasks* and *Tuxedo User Management*. The Tasks tab contains icons for accessing screens to create a datasource, edit a datasource, import Tuxedo data types and services. The Tuxedo User Management tab allows you to access screens to define Tuxedo authentication contexts and configure pool parameters.

The remainder of this section describes step-by-step procedures to perform a specific task.

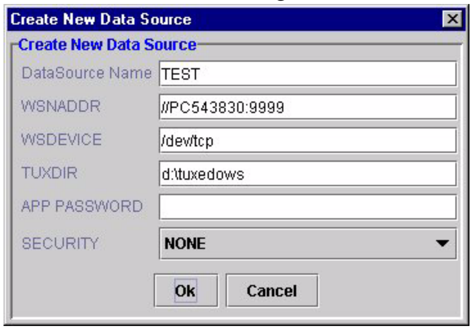
Create New Data Source

This procedure is used to create a new data source.

To Create a New Data Source

1. Choose the *Create New Data Source* feature from the Tuxedo Management Console main window and Figure 3-6 shows the dialog box:

Figure 3-6 Create Data Source Dialog Box


Create New Data Source

Create New Data Source

DataSource Name: TEST
WSNADDR: //PC543830:9999
WSDEVICE: /dev/tcp
TUXDIR: d:\tuxedows
APP PASSWORD:
SECURITY: NONE

2. Enter the following information::

Table 3-2 Create New Data Source dialog Box

Parameter	Definition
Data Source Name	This is the name you assign to the new datasource.

Table 3-2 Create New Data Source dialog Box

Parameter	Definition

WSNADDR	Specify the network address of the workstation listener process through which workstation clients gain access to Tuxedo server. This parameter must be specified in one of the following format:
	`//host_name:port_number`
	`//#.#.#.#:port_number`

WSDEVICE	Specify the device used to access the network. It should be set to `/dev/tcp` for TCP/IP. Optionally can be left blank.
TUXDIR	The Tuxedo workstation (Tuxedo /WS) installation root directory.
APP PASSWORD	Application password. This is required when the SECURITY level is set to either APP_PW, or USER_AUTH/ACL/MANDATORY_ACL. This parameter value is ignored with security level NONE.
SECURITY	Specify the security level defined at Tuxedo server. Valid values: NONE APP_PW USER_AUTH/ACL/MANDATORY_AUTH The default value is NONE. Note that user specific data (user name, client name, and user password) must be defined if the security level is set to USER_AUTH/ACL/MANDATORY_AUTH. Refer to “User Mapping,” on page 118 for details.

3. Press OK.
4. You must restart the management console to use the newly created datasource.

The new Tuxedo datasource is created under the following repository node:

`bsp.dataSources`

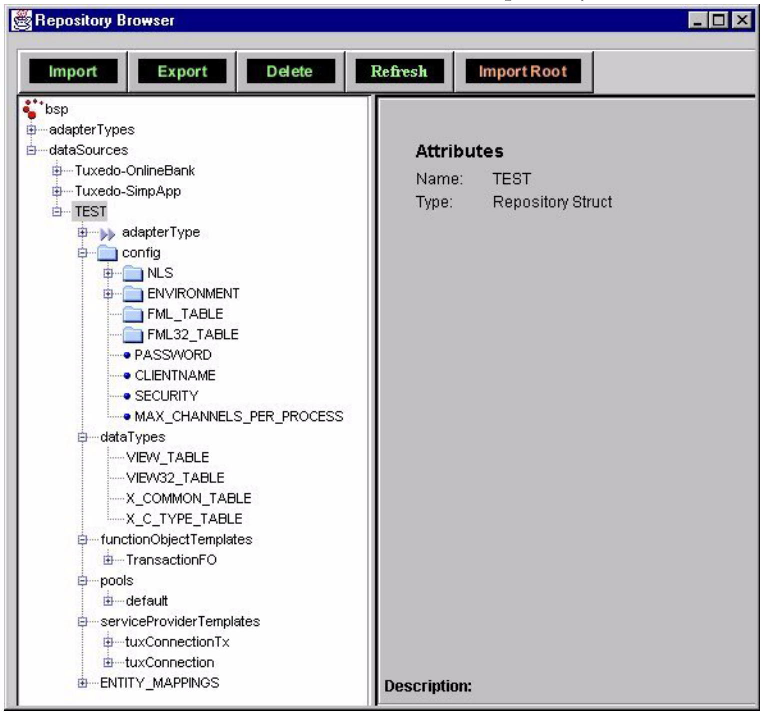
with the following top level child nodes, as shown in Figure 3-7.

- `adapterType`

- `config`
- `dataTypes`
- `functionObjectTemplates`
- `functionObjectTemplates.TransactionFO`
- `pools`
- `pools.default`
- `serviceProviderTemplates`
- `serviceProviderTemplates.tuxConnectionTx`
- `serviceProviderTemplates.tuxConnection`
- ENTITY_MAPPINGS

The *MAX_CHANNELS_PER_PROCESS* is set to 50 with a new datasource. The default function object, *TransactionFO* supports client side transactions in conjunction with *tuxConnectionTx* service provider. Refer to the *iPlanet Application Server Enterprise Connector for Tuxedo Developer's Guide* for details on how to use the *TransactionFO* function object.

Figure 3-7 Tuxedo Datasource in the UIF Repository



Attributes

Name: TEST

Type: Repository Struct

Description:

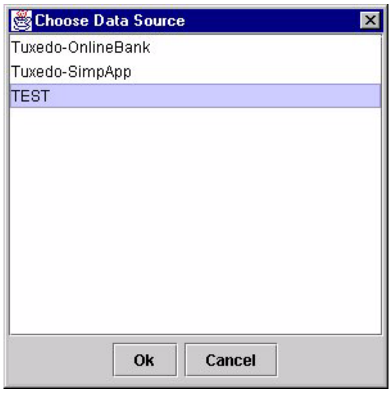
Edit Data Source

The following procedure is used to edit a data source.

To Edit a Data Source

1. Choose the *Edit Data Source* option from the Tuxedo Management Console main window and Figure 3-8 shows the displayed *Choose Data Source* window:

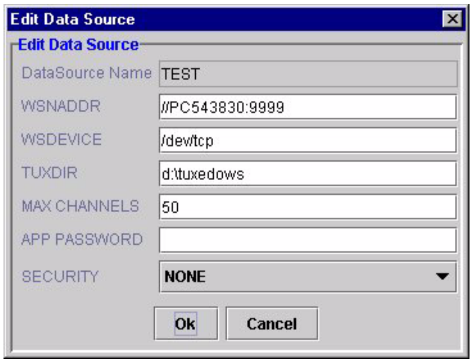
Figure 3-8 Choose Data Source Window



Tuxedo-OnlineBank
Tuxedo-SimpApp
TEST

2. Select the datasource you wish to edit and press OK. Figure 3-9 shows the Edit Data Source dialog box:

Figure 3-9 Edit DataSource Dialog Box



DataSource Name	TEST
WSNADDR	//PC543830:9999
WSDEVICE	/dev/tcp
TUXDIR	d:\tuxedows
MAX CHANNELS	50
APP PASSWORD	
SECURITY	NONE

3. You can modify the following information;:

Table 3-3 Edit Data Source Parameters

Parameter	Definition

WSNADDR	Specify the network address of the workstation listener process through which workstation clients gain access to Tuxedo server. This parameter must be specified in one of the following format:
	`//host_name:port_number`
	`//#.#.#.#:port_number`

Table 3-3 Edit Data Source Parameters

Parameter	Definition
WSDEVICE	Specify the device used to access the network. It should be set to `/dev/tcp` for TCP/IP. Optionally can be left blank.
TUXDIR	The Tuxedo workstation (Tuxedo /WS) installation root directory.
MAX CHANNELS	Specifies the maximum concurrent channels per Tuxedo worker process. Must be greater than or equal 1 and less than or equal to 50. Please refer to "Connection Pooling," on page 91 for detail description.
SECURITY	Specify the security level defined at Tuxedo server. Valid values: NONE APP_PW USER_AUTH/ACL/MANDATORY_AUTH
APP PASSWORD	Application password. This is required when the SECURITY level is set to either APP_PW, or USER_AUTH/ACL/MANDATORY_ACL. This parameter value is ignored with security level NONE.

4. Click OK to save the information.

NOTE The datasource name is read only and cannot be modified. To assign a new name to an existing datasource, you must first delete the existing datasource using the UIF repository browser and create a new datasource.

FML 16 Import

This is used to import FML 16.

To Import FML 16

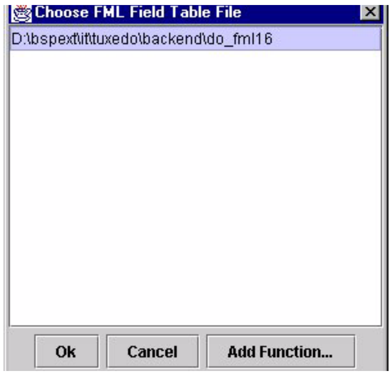
1. Choose the *FML 16 Import* feature from the Tuxedo Management Console main window. The *Choose Data Source* window is displayed as shown in Figure 3-8.

2. Select a datasource from the datasources listed and click on Ok. The *Choose FML Field Table File* window is displayed as shown in Figure 3-10.
3. Click on the *Add Function...* button and the *File Selection* window appears. The current directory (directory from which console started) is shown as the default directory for file selection.
4. Select FML field table definition file(s) and click the Ok button.

FML fields defined in the FML Field Table file(s) are imported under the following repository node:

```
bsp.dataSources.<dsName>.config.FML_TABLE
```

Figure 3-10 Choose FML Field Table File Window


Choose FML Field Table File

D:\bspext\it\tuxedo\backend\do_fml16

The following lists the sample data:

- Content of FML Field Table input file (`fml16flds`).

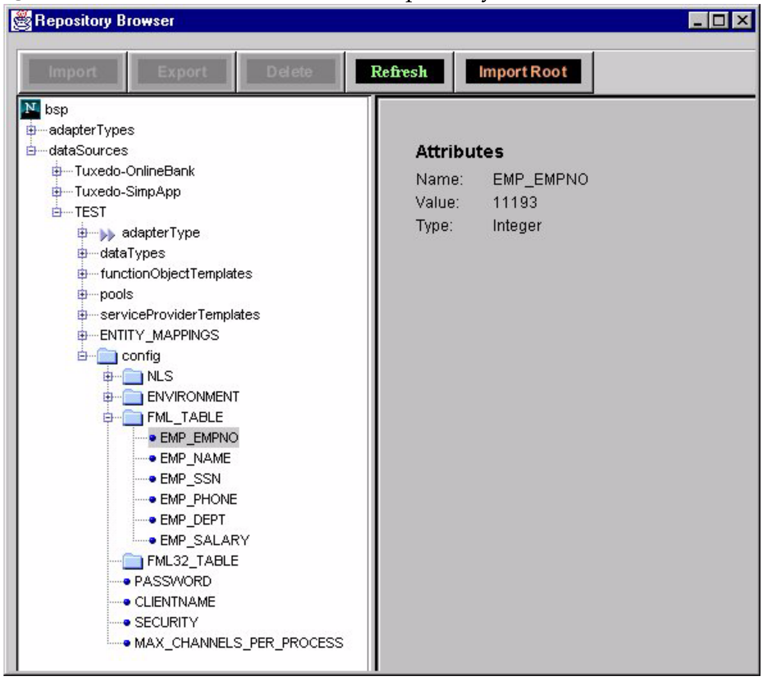
```
# name         number  type    flags   comments
*base 3000
EMP_EMPNO        1     long    -       -
EMP_NAME         2     string  -       -
EMP_SSN          3     string  -       -
EMP_PHONE        4     string  -       -
EMP_DEPT         5     long    -       -
EMP_SALARY       6     float   -       -
```

- Content of header file generated by calling `mkfldhdr fml16flds` command. Not required for import operation. Optionally, can be used to verify the repository information after importing FML fields.

```
#define EMP_EMPNO   ((FLDID)11193)
#define EMP_NAME    ((FLDID)43962)
#define EMP_SSN     ((FLDID)43963)
#define EMP_PHONE   ((FLDID)43964)
#define EMP_DEPT    ((FLDID)11197)
#define EMP_SALARY  ((FLDID)27582)
```

- Figure 3-11 shows the UIF Repository information after importing `fml16flds` using *FML 16 Import* feature.

Figure 3-11 FML Data in the UIF Repository



Attributes

Name:	EMP_EMPNO
Value:	11193
Type:	Integer

FML 32 Import

This is used to import FML 32

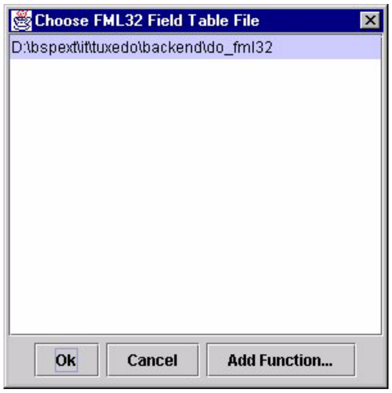
To import FML32

1. Choose the *FML 32 Import* feature from the Tuxedo Management Console main window. The *Choose Data Source* window is displayed as shown in Figure 3-8.
2. Select a datasource from the datasources listed, and click on Ok button. The *Choose FML32 Field Table File* window is displayed as shown in Figure 3-12.
3. Click on the *Add Function...* button and the *File Selection* window appears. The current directory (directory from which console started) is shown as the default directory for file selection.
4. Select FML32 field table definition file(s) and click the Ok button.

FML32 fields defined in the FML32 Field Table file(s) are imported under the following repository node:

```
bsp.dataSources.<dsName>.config.FML32_TABLE
```

Figure 3-12 Choose FML32 Field Table File Window


Choose FML32 Field Table File

D:\bspext\it\tuxedo\backend\do_fml32

The following lists the sample data:

- Content of FML32 Field Table file (`fml32flds`).

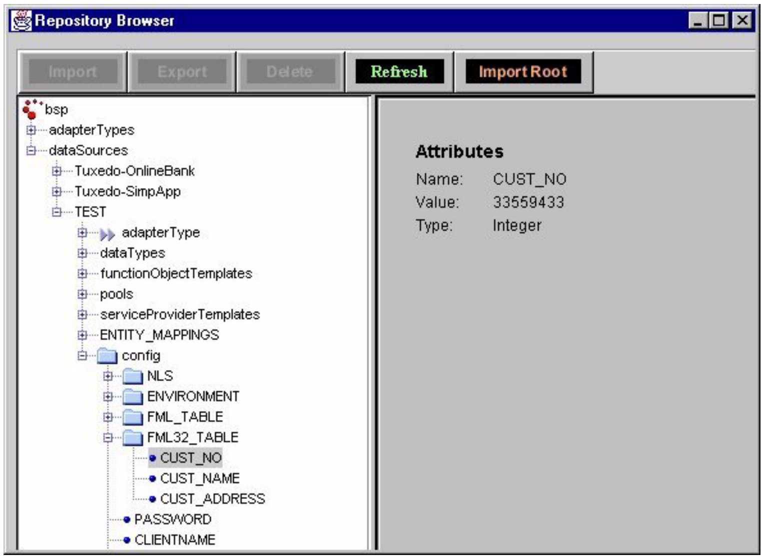
```
# name        number  type    flags   comments
*base 5000
CUST_NO         1     long    -       -
CUST_NAME       2     string  -       -
CUST_ADDRESS    3     string  -       -
```

- Content of header file generated by calling `mkfldhdr32 fml32flds` command. Not required for import operation. Optionally, can be used to verify the repository information after importing FML32 fields.

```
#define CUST_NO          ((FLDID32)33559433)
#define CUST_NAME        ((FLDID32)167777162)
#define CUST_ADDRESS     ((FLDID32)167777163)
```

- Figure 3-13 shows the UIF Repository information after importing `fml32flds` using *FML 32 Import* feature.

Figure 3-13 FML32 Data in the UIF Repository



Name:	CUST_NO
Value:	33559433
Type:	Integer

VIEW16 Import

This is used to import View16 features.

To import VIEW16 features

1. Choose the *VIEW16 Import* feature from the Tuxedo Management Console main window. The *Choose Data Source* window is displayed as shown in Figure 3-8.

2. Select a datasource from the datasources listed, and click on Ok. The *Choose VIEW Object File* window is displayed as shown in Figure 3-14.

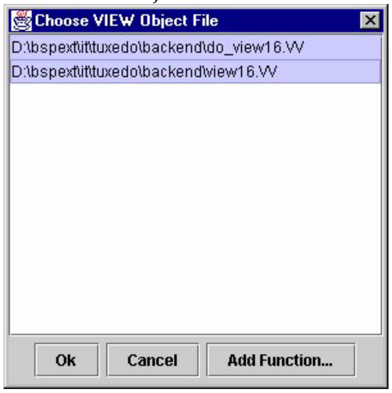
3. Click on the *Add Function...* button and the *File Selection* window appears. The current directory (directory from which console started) is shown as the default directory for file selection.

4. Select VIEW object file(s) and click OK.

This information is imported under the following repository node:

```
bsp.dataSources.<dsName>.dataTypes.VIEW_TABLE
```

Figure 3-14 Choose VIEW Object File Window



D:\bspext\it\tuxedo\backend\do_view16.VV
D:\bspext\it\tuxedo\backend\view16.VV

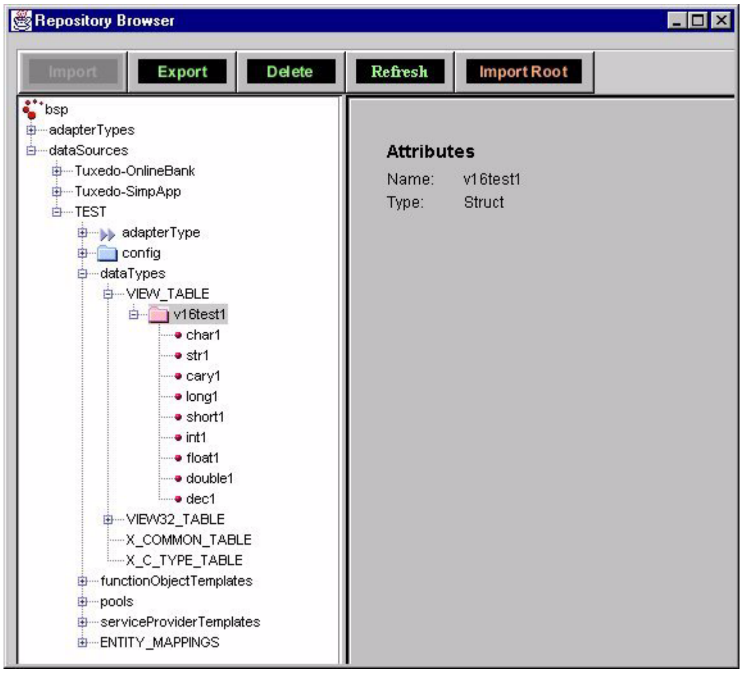
The following lists the sample data:

- Content of VIEW file (`view16.txt`).

```
VIEW v16test1
# type cname   fbname  count flag  size null
char   char1   -       1     -     -    -
string str1    -       1     -     100  -
carray cary1   -       1     -     100  -
long   long1   -       1     -     -    -
short  short1  -       1     -     -    -
int    int1    -       1     -     -    -
float  float1  -       1     -     -    -
double double1 -       1     -     -    -
dec_t  dec1    -       1     -     9,2  -
END
```

- Generate VIEW Object file (`view16.VV` on NT or `view16.V` on Unix) by running `viewc -n view16.txt` command.
- Figure 3-15 shows the UIF Repository information after importing `view16.VV` using *VIEW 16 Import* feature.

Figure 3-15 VIEW Data in the UIF Repository



```
bsp
  adapterTypes
  dataSources
    Tuxedo-OnlineBank
    Tuxedo-SimpApp
    TEST
      adapterType
      config
      dataTypes
        VIEW_TABLE
          v16test1
            char1
            str1
            cary1
            long1
            short1
            int1
            float1
            double1
            dec1
        VIEW32_TABLE
        X_COMMON_TABLE
        X_C_TYPE_TABLE
      functionObjectTemplates
      pools
      serviceProviderTemplates
      ENTITY_MAPPINGS
```

Attributes
Name: v16test1
Type: Struct

VIEW 32 Import

This is used to import VIEW32 features.

To import VIEW32 features

1. Choose the *VIEW 32 Import* feature from the Tuxedo Management Console main window. The *Choose Data Source* window is displayed as shown in Figure 3-8.
2. Select a datasource from the datasources listed, and click on Ok. The *Choose VIEW32 Object File* window is displayed as shown in Figure 3-16.

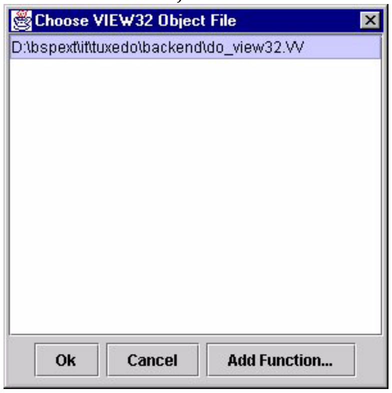
3. Click on the *Add Function...* button and the *File Selection* window appears. The current directory (directory from which console started) is shown as the default directory for file selection.

4. Select VIEW32 object file(s) and click OK.

This information is imported under the following repository node:

```
bsp.dataSources.<dsName>.dataTypes.VIEW32_TABLE
```

Figure 3-16 Choose VIEW32 Object File Window


Choose VIEW32 Object File

D:\bspext\it\tuxedo\backend\do_view32.VV

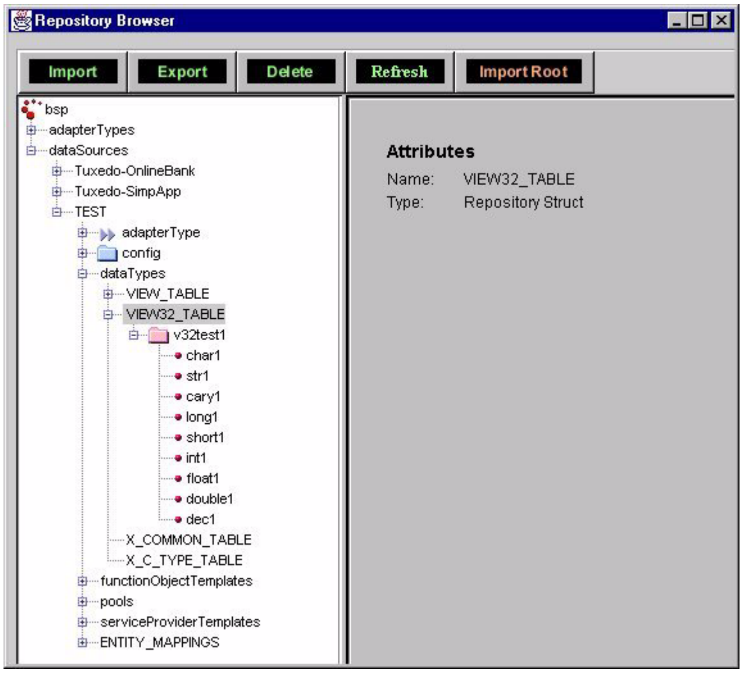
The following lists the sample data:

- Content of VIEW32 file (`view32.txt`).

```
VIEW v32test1
# type cname   fbname  count flag  size null
char   char1   -       1     -     -    -
string str1    -       1     -     100  -
carray cary1   -       1     -     100  -
long   long1   -       1     -     -    -
short  short1  -       1     -     -    -
int    int1    -       1     -     -    -
float  float1  -       1     -     -    -
double double1 -       1     -     -    -
dec_t  dec1    -       1     -     9,2  -
END
```

- Generate VIEW32 Object file (`view32.VV` on NT or `view32.V` on Unix) by running `viewc32 -n view32.txt` command.
- Figure 3-17 shows the UIF Repository information after importing `view32.VV` using *VIEW 32 Import* feature.

Figure 3-17 VIEW32 Data in the UIF Repository



Attributes

Name: VIEW32_TABLE

Type: Repository Struct

X_COMMON Import

This is used to import X_COMMON.

To import X_COMMON

1. Choose the *X_COMMON Import* feature from the Tuxedo Management Console main window. The *Choose Data Source* window is displayed as shown in Figure 3-8.
2. Select a datasource from the datasources listed, and click on Ok. The *Choose X_COMMON Object File* window is displayed as shown in Figure 3-18.

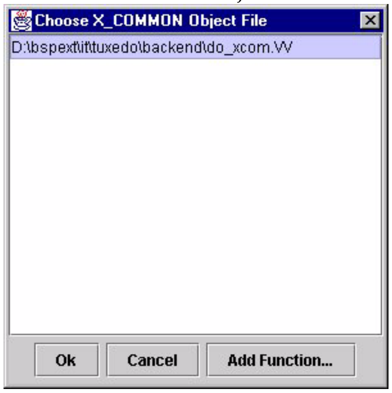
3. Click on the *Add Function...* button and the *File Selection* window appears. The current directory (directory from which console started) is shown as the default directory for file selection.

4. Select X_COMMON object file(s) and click OK.

This information is imported under the following repository node:

`bsp.dataSources.<dsName>.dataTypes.X_COMMON_TABLE`

Figure 3-18 Choose X_COMMON Object File Window



D:\bspext\if\tuxedo\backend\do_xcom.VV

The following lists the sample data:

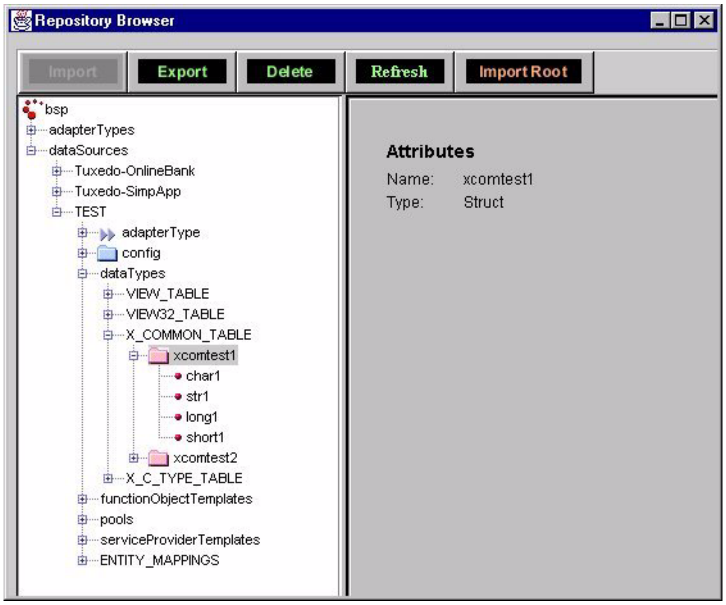
- Content of X_COMMON definition file (`xcomm.txt`).

```
VIEW xcomtest1
# type cname   fbname  count flag  size null
char   char1   -       1     -     -    -
string str1    -       1     -     100  -
long   long1   -       1     -     -    -
short  short1  -       1     -     -    -
END
```

- Generate VIEW Object file (`xcomm.VV` on NT or `xcomm.V` on Unix) by running `viewc -n xcomm.txt` command.

- Figure 3-19 shows the UIF Repository information after importing `xcomm.VV` using *X_COMMON Import* feature.

Figure 3-19 X_COMMON Data in the UIF Repository



Attributes

Name: xcomtest1

Type: Struct

X_C_TYPE Import

This is used to import X_C_TYPE features.

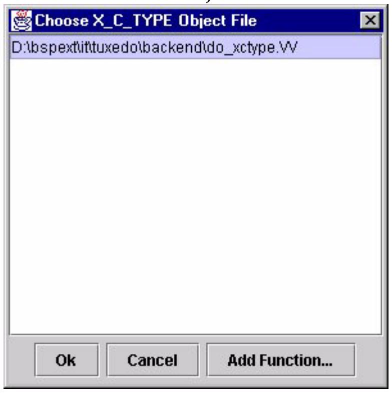
To import X_C_TYPE features

1. Choose the *X_C_TYPE Import* feature from the Tuxedo Management Console main window. The *Choose Data Source* window is displayed as shown in Figure 3-8.
2. Select a datasource from the datasources listed, and click on Ok. The *Choose X_C_TYPE Object File* window is displayed as shown in Figure 3-20.
3. Click on the *Add Function...* button and the *File Selection* window appears. The current directory (directory from which console started) is shown as the default directory for file selection.
4. Select X_C_TYPE object file(s) and click OK.

This information is imported under the following repository node:

```
bsp.dataSources.<dsName>.dataTypes.X_C_TYPE_TABLE
```

Figure 3-20 Choose X_C_TYPE Object File Window



D:\bspext\if\tuxedo\backend\do_xctype.VV

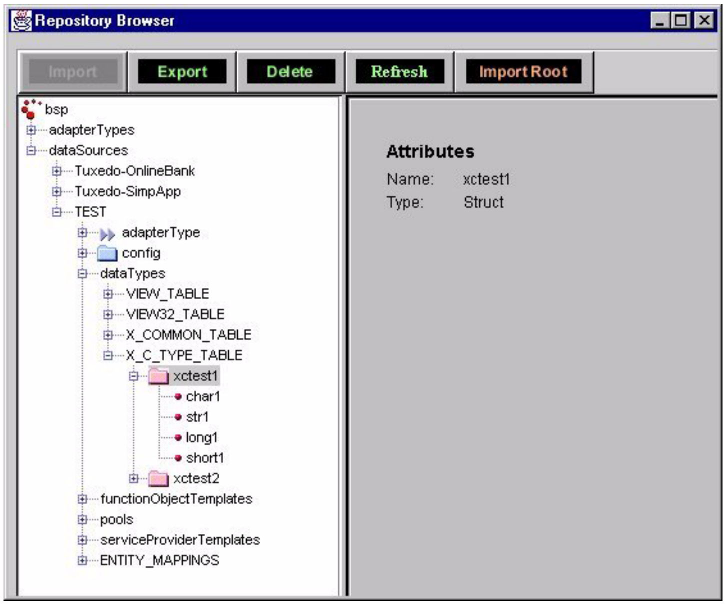
The following lists the sample data:

- Content of X_C_TYPE definition file (`xctype.txt`).

```
VIEW xctest1
# type cname   fbname  count flag  size null
char   char1   -       1     -     -    -
string str1    -       1     -     100  -
long   long1   -       1     -     -    -
short  short1  -       1     -     -    -
END
```

- Generate VIEW Object file (`xctype.VV` on NT or`xctype.V` on Unix) by running `viewc -n xctype.txt` command.
- Figure 3-21 shows the UIF Repository information after importing `xctype.VV` using *X_C_TYPE Import* feature.

Figure 3-21 X_C_TYPE Data in the UIF Repository



Attributes

Name: xctest1

Type: Struct

Import Tuxedo Services

This is used to import Tuxedo services.

To Import Tuxedo Services

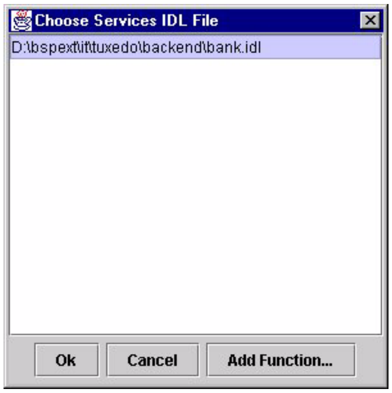
1. Choose the *Services Import* feature from the Tuxedo Management Console main window. The *Choose Data Source* window is displayed as shown in Figure 3-8.
2. Select a datasource from the datasources listed, and click on Ok. The *Choose Services IDL File* window is displayed as shown in Figure 3-22.
3. Click on the *Add Function...* button and the *File Selection* window appears. The current directory (directory from which console started) is shown as the default directory for file selection.
4. Select the IDL file which contains the definition of Tuxedo services and click OK.

This information is imported under the following repository node:

```
bsp.dataSources.<dsName>.functionObjectTemplates
```

There must be an entry for each service.

Figure 3-22 Choose Services IDL File Window



D:\bspext\it\tuxedo\backend\bank.idl

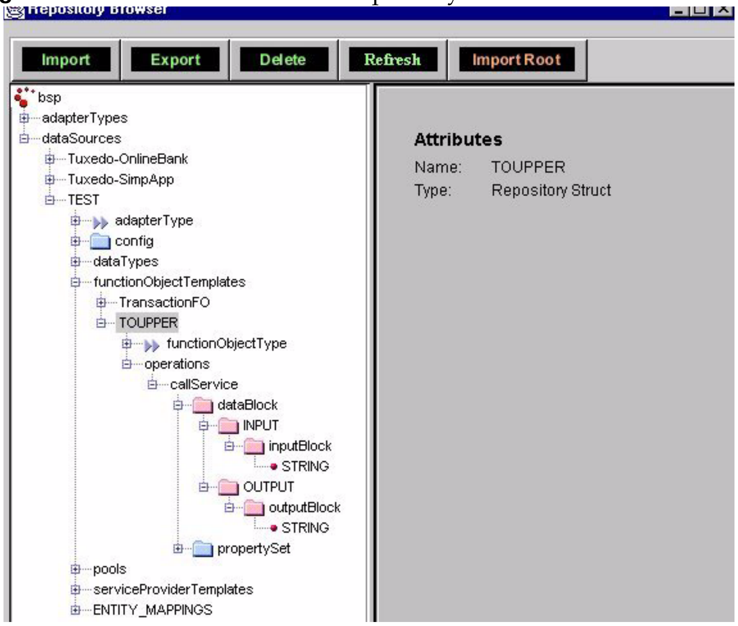
The following lists the sample data:

- Content of Services IDL file (`simple.idl`).

```
interface simple {

    void TOUPPER(
        [in]    STRING STRING
        [out]   STRING STRING
    );
};
```

- Figure 3-23 shows the UIF Repository information after importing `simple.idl` using the *Services Import* feature.

Figure 3-23 Services in the UIF Repository



```
bsp
  adapterTypes
  dataSources
    Tuxedo-OnlineBank
    Tuxedo-SimpApp
    TEST
      adapterType
      config
      dataTypes
      functionObjectTemplates
        TransactionFO
        TOUPPER
          functionObjectType
          operations
            callService
              dataBlock
                INPUT
                  inputBlock
                    STRING
                OUTPUT
                  outputBlock
                    STRING
              propertySet
      pools
      serviceProviderTemplates
      ENTITY_MAPPINGS
```

Attributes

Name: TOUPPER
Type: Repository Struct

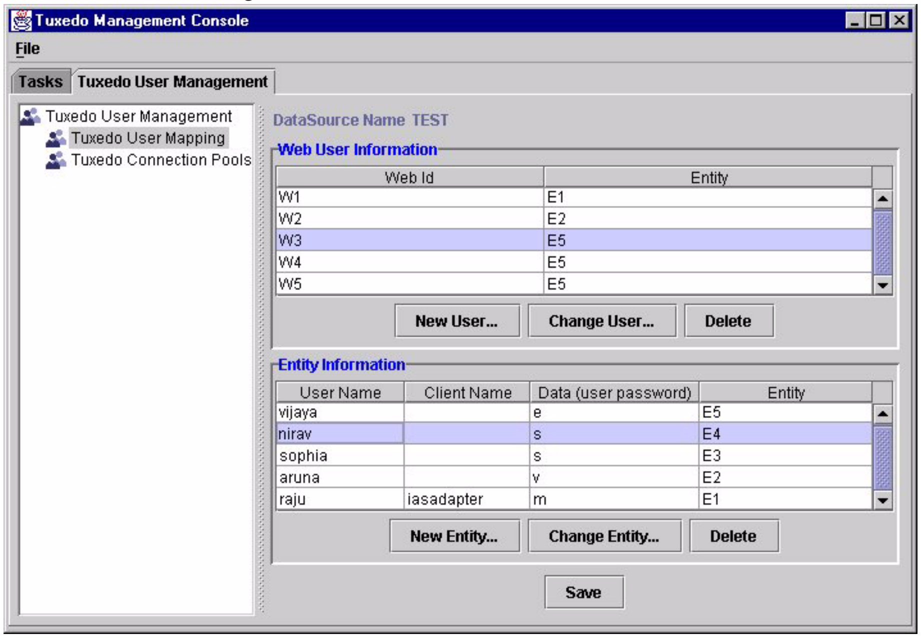
User Mapping

The Tuxedo User Management main screen is used to add, edit, and delete Entity and Web User information for the selected datasource.

To map users

1. Select the Tuxedo User Mapping tab from the Tuxedo Management Console main window.
2. Select *Tuxedo User Mapping* option from Tuxedo User Management tree. The *Choose Data Source* window is displayed as shown in Figure 3-8, if datasource is not selected. Select a datasource from the datasources listed, and click on Ok. The Tuxedo User Management main screen appears, as shown in Figure 3-24.

Figure 3-24 Tuxedo User Management Main Window



DataSource Name TEST

Web User Information

Web Id	Entity
W1	E1
W2	E2
W3	E5
W4	E5
W5	E5

Entity Information

User Name	Client Name	Data (user password)	Entity
vijaya		e	E5
nirav		s	E4
sophia		s	E3
aruna		v	E2
raju	iasadapter	m	E1

Entity Information

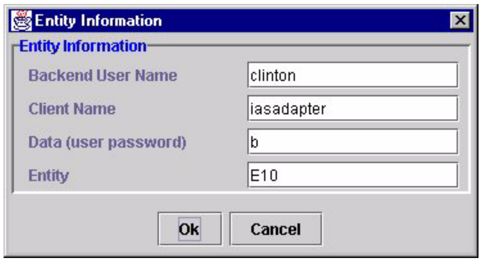
You can add a new entity, change an existing entity, and delete an entity by clicking on the corresponding button.

To add a new entity

1. Click on the *New Entity...* button.

The Entity Information window appears, as shown in Figure 3-25.

Figure 3-25 Entity Information Window



Backend User Name	clinton
Client Name	iasadapter
Data (user password)	b
Entity	E10

2. Type in appropriate data and click the Ok button.

   The new entity information appears in the Entity Information area of User Management main window.

To change entity information

1. Select the entity to be modified in Entity Information area, then click on the *Change Entity...* button.
2. Edit the desired field(s) and click on the Ok button.

   The updated information appears in the Entity Information area of User Management main window.

To delete an entity

1. Select the desired entity and click on the *Delete* button.

   The entity information is removed from Entity Information area of User Management main window.
2. When you have completed all actions in this window, click on the *Save* button. All additions and changes is saved into the UIF Repository.

NOTE An entity cannot be deleted if the web users are associated. You must first delete all associated web users before deleting an Entity.

Web User Information

The Web User Information is used to map a web user's identity to an EIS user's identity (Tuxedo authentication context). The administrator assigns an entity for the web user and the Web User Id is used by the application programmer to specify the authentication context to be used while processing the request.

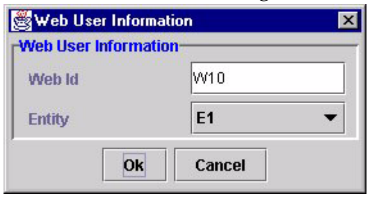
You can add, change, and delete web user by clicking the corresponding button.

To add a web user

1. Click on the *New User...* button.

   The Web User Information dialog box appears, as shown in Figure 3-26.

Figure 3-26 Web User Information Dialog Box



Web Id: W10
Entity: E1

2. Type in the Web ID and select an Entity from the available entities.
3. Click OK. The information is added to Web User Information area of User Management main window.

To change a user

1. Select the desired user from list, click on the *Change User...* button.
2. Edit the desired field(s) and click OK.

   The updated information appears in the Web User Information area of User Management main window.

To delete a web user

1. Select the desired web user and click on the *Delete* button.

   The selected web user is deleted from User Management main window.

2. When you have completed all actions in this window, click *Save*.

   All additions and changes are saved into the UIF Repository.

Changing Datasource

To perform user mapping operations on another datasource:

1. Select *Tuxedo User Mapping* option from Tuxedo User Management tree and click the right mouse button. The *Change Data Source* popup menu appears.
2. Select the *Change Data Source* popup menu. The *Choose Data Source* window is displayed as shown in Figure 3-8.
3. Select a datasource from the datasources listed, and click on Ok. The selected datasource is used for further user mapping operations.

Update Connection Pool Parameters

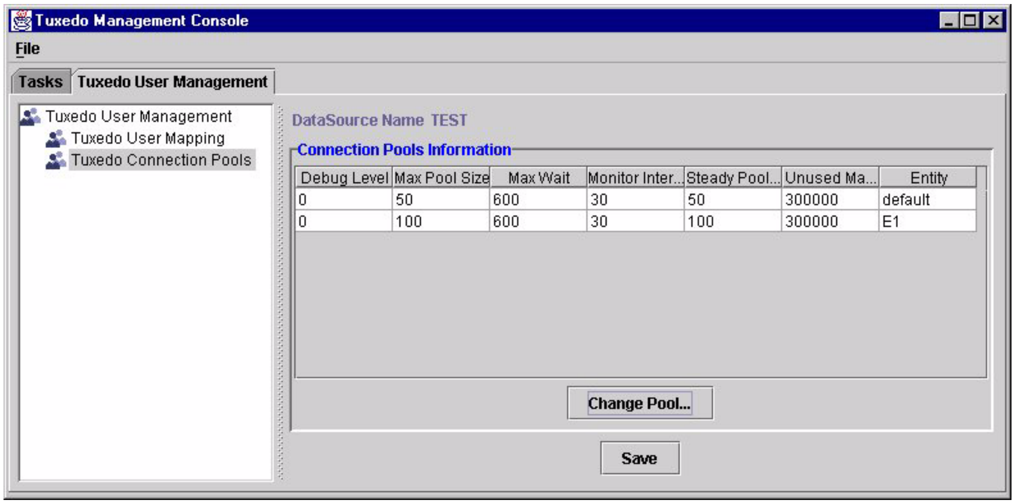
This procedure is used to update Connection Pool parameters.

To update Connection Pool Parameters

1. Choose the *Tuxedo User Mapping* tab from the main management console window.
2. Select *Tuxedo Connection Pools* option from Tuxedo User Management tree. The *Choose Data Source* window is displayed as shown in Figure 3-8, if datasource is not selected before.
3. Select a datasource from the datasources listed, and click OK.

   The Connection Pools Information main screen appears, as shown in Figure 3-27.

Figure 3-27 Connection Pools Information Window



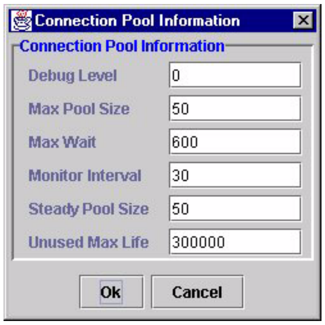
DataSource Name TEST

Connection Pools Information

Debug Level	Max Pool Size	Max Wait	Monitor Inter...	Steady Pool...	Unused Ma...	Entity
0	50	600	30	50	300000	default
0	100	600	30	100	300000	E1

4. Select the desired pool and click on *Change Pool...* button. The *Connection Pool Information* dialog box appears, as shown in Figure 3-28.

Figure 3-28 Edit Connection Pool Dialog Box



Connection Pool Information

Debug Level	0
Max Pool Size	50
Max Wait	600
Monitor Interval	30
Steady Pool Size	50
Unused Max Life	300000

5. You can modify the following information:

Table 3-4 Connection Pool Information

Parameter	Definition
Debug Level	Connection Pool message logging level.
	Valid values:
	0 - Logging turned off
	1 - Logs only callback messages
	2 - Logs all messages
Max Pool Size	Maximum number of concurrent requests to be processed.
	Refer to “Connection Pooling,” on page 91 for detail information.
Max Wait	The maximum time, in seconds, a request waits in the queue for a connection to Tuxedo server before it times out.
Monitor Interval	The time in seconds, the monitor thread prints logging message about pool status.
Steady Pool Size	Should be same as Max Pool Size. The field value is automatically updated for any changes to Max Pool Size field.
	The GUI will ensure that both the Max Pool Size and Steady Pool Size are set to same value. This is the requirement for Tuxedo Enterprise Connector.
Unused Max Life	The maximum time, in seconds, that a connection remains unused in the pool.

Edit the desired field(s) and click OK. The updated information appears in the Connection Pools Information area.

6. Click the *Save* button to store the changes into the UIF Repository.

Changing Datasource

To update a pool defined for another datasource

1. Select *Tuxedo Connection Pools* option from Tuxedo User Management tree and click the right mouse button. The *Change Data Source* popup menu appears.
2. Select the *Change Data Source* popup menu. The *Choose Data Source* window is displayed as shown in Figure 3-8.

3. Select a datasource from the datasources listed, and click on Ok. The Connection Pools Information area is updated with pools defined for the selected datasource.

Read Only Nodes

The following repository nodes are read only. Do not modify the definition of these nodes.

- *bsp.adapterTypes.Tuxedo*
- *bsp.dataSources.<dsName>.adapterType.Tuxedo*
- *bsp.dataSources.<dsName>.config.NLS.charoptimized*
- *bsp.dataSources.<dsName>.config.NLS.charset*
- *bsp.dataSources.<dsName>.functionObjectTemplates.TransactionFO*
- *bsp.dataSources.<dsName>.serviceProviderTemplates*

Always use the management console to make any adjustments to the Tuxedo datasource(s). Avoid importing, manually created or updated datasource XML files, into repository using the UIF repository browser.

Deploying the Repository Content

In addition to deploying your application, you must explicitly deploy the repository content by exporting its contents to XML and importing the XML file on the deployment machine. This is performed using the Repository Browser tool.

Using repository browser, export the datasource content into an XML file. The XML file can be restored later either on the same machine (as part of backup and recovery) or on a different machine (as part of deploying the application) using the Import feature provided with the repository browser.

During the import process, the `repository.dtd` for the enterprise connector must exist in the same directory as the XML file. The `repository.dtd` file for the Tuxedo Enterprise Connector is located in the *ias/APPS/adapters/tux/metadata* subdirectory of the iPlanet Application Server installation root directory. You must use only this `repository.dtd` file while importing XML file(s) related to the Tuxedo Enterprise Connector.

Always backup the datasource(s) into XML file before and after any changes to Tuxedo datasources.

Refer to *Using the Repository Browser* in the *iPlanet Unified Integration Framework Developer's Guide* for additional information on exporting and importing XML data.

NOTE VIEW information is platform specific. Therefore, if the repository content contains VIEW data, the content can only be exchanged across the same platform types. For example, NT to NT, and Solaris to Solaris.

Setting Parameters for International Applications

The following parameters are used with each datasource to store information about the charset to be used while sending configuration data like user name, client name, and password from connector to Tuxedo server.

- `bsp.dataSources.<dsName>.config.NLS.charoptimized`

  Initialized with value 0. This is read-only parameter.

- `bsp.dataSources.<dsName>.config.NLS.charset`

  The charset parameter is initialized with UTF-8. The user authentication data (user name, client name, and data in TPINIT structure) is always passed in UTF-8 format to Tuxedo server, if the iPlanet Application Server is running in international mode. Currently only ASCII characters are supported in TPINIT structure of BEA Tuxedo system. Hence this is read-only parameter.

Even though the management console allows you to enter data in popular legacy character sets such as ASCII, SJIS or ECU_JP (Japanese), GN2312 (Chinese), KCS5601 (Korean), and so on, the *datasource* and *entity* (user mapping) names must be defined using ASCII characters only. To handle multi-byte/double byte data correctly in management console and repository browser, the iPlanet Application Server must operate in INTERNATIONAL mode.

Please contact BEA System for details on support of multi-byte/double byte characters in BEA Tuxedo software.

Jolt Migration

The Tuxedo Enterprise Connector provides a command line tool to import the Tuxedo services into the UIF Repository, if the Tuxedo services definition are already present in the Jolt Bulk Loader format. The command line tool accepts the Jolt bulk loader services file and converts them into XML format and imports it into the UIF Repository. Refer to the BEA Jolt documentation for detailed information on Bulk Loading Tuxedo Services.

To use the command line tool

1. Create a datasource for your application using the management console.
2. Import FML and VIEW data types, if any, used in the Tuxedo services using the management console.
3. Run the following command by passing the Jolt bulk loader services file:

   On Windows NT

   ```
   joltmigrator -f <jolt_bulk_load_file> -s <datasourcename>
   ```

   On Unix

   ```
   <iAS installdir>/ias/APPS/bin/joltmigrator.sh -f
   <jolt_bulk_load_file> -s <datasourcename>
   ```

   Where, *jolt_bulk_load_file* is the Tuxedo services definition file in the Jolt Bulk Load format and the datasource name is the datasource under which the services are to be imported.

The following listing shows a sample Jolt Bulk Loader services file for Tuxedo services used in section "Tuxedo Services IDL File," on page 76.

Code Example 3-1 Tuxedo Service IDL File code

```
service=STRINGTEST1
inbuf=STRING
outbuf=STRING
param=inParam
type=string
access=in
param=outParam
type=string
access=out

service=STRINGTEST2
```

Code Example 3-1 Tuxedo Service IDL File code

```
outbuf=STRING
param=outParam
type=string
access=out

service=STRINGTEST3
inbuf=STRING
param=inParam
type=string
access=in

service=CARRAYTEST1
inbuf=CARRAY
outbuf=CARRAY
param=inParam
type=carray
access=in
param=outParam
type=carray
access=out

service=CARRAYTEST2
outbuf=CARRAY
param=outParam
type=carray
access=out

service=CARRAYTEST3
inbuf=CARRAY
param=inParam
type=carray
access=in

service=XOCTETTEST1
inbuf=X_OCTET
outbuf=X_OCTET
param=inParam
type=carray
access=in
param=outParam
type=carray
access=out

service=XOCTETTEST2
outbuf=X_OCTET
param=outParam
type=carray
access=out

service=XOCTETTEST3
inbuf=X_OCTET
param=inParam
type=carray
access=in
```

Code Example 3-1 Tuxedo Service IDL File code

```
service=FML16TEST1
inbuf=FML
outbuf=FML
param=string16
type=string
access=in
param=string16
type=string
access=out
count=10

service=FML16TEST2
outbuf=FML
param=char16
type=char
access=out
count=10

service=FML16TEST3
inbuf=FML
param=string16
type=string
access=in
param=short16
type=short
access=in
param=double16
type=double
access=in
param=float16
type=float
access=in
param=long16
type=long
access=in

service=FML32TEST1
inbuf=FML32
outbuf=FML32
param=string32
type=string
access=in
param=string32
type=string
access=out
count=10

service=FML32TEST2
outbuf=FML32
param=carray32
type=carray
access=out
count=10
```

Code Example 3-1 Tuxedo Service IDL File code

```
service=FML32TEST3
inbuf=FML32
param=string32
type=string
access=in
param=short32
type=short
access=in
param=double32
type=double
access=in
param=float32
type=float
access=in
param=long32
type=long
access=in

service=VIEW16TEST1
inbuf=VIEW
outbuf=VIEW
inview=v16test1
outview=v16test1

service=VIEW16TEST2
inbuf=VIEW
outbuf=VIEW
inview=v16test1
outview=v16test2

service=VIEW16TEST3
inbuf=VIEW
inview=v16test4

service=VIEW16TEST4
outbuf=VIEW
outview=v16test4

service=VIEW32TEST1
inbuf=VIEW32
outbuf=VIEW32
inview=v32test1
outview=v32test1

service=VIEW32TEST2
inbuf=VIEW32
outbuf=VIEW32
inview=v32test1
outview=v32test2

service=VIEW32TEST3
inbuf=VIEW32
inview=v32test4
```

Code Example 3-1 Tuxedo Service IDL File code

```
service=VIEW32TEST4
outbuf=VIEW32
outview=v32test4

service=XCOMMONTEST1
inbuf=X_COMMON
outbuf=X_COMMON
inview=xcomtest1
outview=xcomtest2

service=XCTYPETEST1
inbuf=X_C_TYPE
outbuf=X_C_TYPE
inview=xctest1
outview=xctest2
```

NOTE `noaccess` value for access param is not supported.

Use count param value other than `1` to indicate FML field is multiple occurrence.

Appendix A

Installed Components

The following table shows the iAS Enterprise Connector for Tuxedo libraries and executable programs. After the iAS Enterprise Connector for Tuxedo software is installed verify that the libraries and programs are installed.

NOTE Please don't execute directly any of the executable programs listed in the following table. The Tuxedo Enterprise Connector runtime or Management Console tool automatically executes, as needed.

Table A-1 Installed Components

Directory	On Windows NT or Win2000	On Solaris
`ias/APPS/bin`	`cTuxedoAdapterExt.dll`	`libcTuxedoAdapterExt.so`
	`jx2tuxnative.dll`	`libjx2tuxnative.so`
	`tuxclient.exe`	`tuxclient`
	`Fml2Xml.exe`	`Fml2Xml`
	`Tuxedo2Xml.exe`	`Tuxedo2Xml`
	`View2Xml.exe`	`View2Xml`
	`fmlminer.bat`	`fmlminer.sh`
	`viewminer.bat`	`viewminer.sh`
	`tuxservicesminer.bat`	`tuxservicesminer.sh`
	`joltmigrator.bat`	`joltmigrator.sh`
	`tuxConsole.bat`	`tuxedoconsole.sh`
`ias/APPS/adapters/tux/*`	`*.gxr, *.reg, *.xml & *.dtd files`	`*.gxr, *.reg, *.xml & *.dtd files`

Table A-1 Installed Components

Directory	On Windows NT or Win2000	On Solaris
`ias/APPS/console/*`	`Management Console related files`	`Management Console related files`
`ias/APPS/ecu/*`	`Management Console related files`	`Management Console related files`
`ias/APPS/netscape/tux`	`*.class files`	`*.class files`
`ias/APPS/TuxBank`	`OnLineBank sample application`	`OnLineBank sample application`
`ias/APPS/TuxSimpApp`	`Simpapp sample application`	`Simpapp sample application`
`ias/APPS/docs/tux`	`TuxAdmin.pdf`	`TuxAdmin.pdf`
	`TuxDev.pdf`	`TuxDev.pdf`
`ias/APPS`	`tuxedoConsole.properties`	`tuxedoConsole.properties`
`ias/kdb/en_US`	`*tux*.properties`	`*tux*.properties`
	Message text files	Message text files

Appendix B

Sample Programs

The Tuxedo Enterprise Connector includes the following sample web-based applications. These applications access the Tuxedo services enabling you to quickly learn techniques for developing and deploying in iPlanet Application Server environment.

- Simple Tuxedo Application Sample
- Online Bank Tuxedo Application Sample

The sample applications are fully functional and J2EE compliant.

Simple Tuxedo Application Sample

The sample illustrates the Tuxedo connectivity of the iPlanet Application Server to the `simpapp` application (TOUPPER Tuxedo service) that ships with BEA Tuxedo system. Typically, the Tuxedo server application is found in the *<tuxedo root dir>/apps/simpapp* directory. This sample Java application requires that the Tuxedo server sample application `simpapp` is fully configured for Tuxedo `/`WS client, and be up and running.

The sample application demonstrates how to use Java Servlet to call a Tuxedo service. The source code is available under *<iAS root dir>/ias/APPS/TuxSimpApp* directory.

Online Bank Tuxedo Application Sample

The sample application illustrates the Tuxedo connectivity of iPlanet Application Server to the `bankapp` application that comes with the BEA Tuxedo system. Typically, the Tuxedo server application is found in the *<tuxedo root dir>/apps/bankapp* directory. In order to run the sample Java application, the Tuxedo server `bankapp` sample application must be configured, and be up and running. Please refer to *Tuxedo Application Development Guide* for instructions on how to configure this application.

The Tuxedo `bankapp` server application contains the services WITHDRAWAL, DEPOSIT, TRANSFER, INQUIRY, CLOSE_ACCT, OPEN_ACCT. The INQUIRY service accepts the integer input FML field ACCOUNT_ID and returns a string result FML field SBALANCE.

The sample application demonstrates techniques to use Servlet, EJB, and JSP J2EE components to develop a web-based application to access the Tuxedo services. The source code for this sample Java application is available under *<iAS root dir>/ias/APPS/TuxBank* directory.

Appendix C

Workarounds

The following workaround has been developed to alleviate certain problems that may occur. The following topics are described:

- Excessive CPU Use of Time

Excessive CPU Use of Time

If the iPlanet Application Server Enterprise Connector for Tuxedo consumes an exorbitant amount of CPU time you can work around the problem using the following method.

Two new variables have been added to the worker so that the Tuxedo client avoids repetitive GETREPLY requests without receiving a reply: SLEEPTIME and MAXSLEEPTIME.

- SLEEPTIME is the time in milliseconds that the worker sleeps (is idle) in between two GETREPLY requests.
- MAXSLEEPTIME is the maximum time in milliseconds that the worker is allowed to remain in SLEEPTIME. This value must be greater than SLEEPTIME.

To Add SLEEPTIME and MAXSLEEPTIME Variables on WinNT/2000

1. Select Start>Settings>Control Panel>System.
2. Select the Environment Tab.
3. Add the following two system variables: SLEEPTIME, MAXSLEEPTIME.

4. Set the value to 1.

To Add SLEEPTIME and MAXSLEEPTIME Variables on Solaris

1. Go to iPlanet/iAS6/ias/env directory
2. Edit the `iasenv.ksh` script and add the following two variables: SLEEPTIME, MAXSLEEPTIME.
3. Set the value to 1.

Index