



THE ENTERPRISE MIDDLEWARE SOLUTION

BEA Connect

OSI TP

User Guide

BEA Connect OSI TP 1.3
Document Edition 1.3
May 1998

Copyright

Copyright © 1998 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, BEA Connect, BEA Jolt, Distributed Application Framework, and Enterprise Middleware Solutions are trademarks of and are developed and licensed by BEA Systems, Inc., Sunnyvale, California. TUXEDO is a registered trademark of Novell, Inc., exclusively licensed to BEA Systems, Inc.

OpenView is a registered trademark of Hewlett-Packard Company. SunNet Manager and Solstice are trademarks of Sun Microsystems, Inc. in the United States and other countries. IBM and NetView are registered trademarks of International Business Machines Corporation. Oracle is a registered trademark of Oracle Corporation. Informix is a registered trademark of Informix. Cabletron Spectrum is a trademark of Cabletron Systems, Inc. UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited.

All other company names may be trademarks of the respective companies with which they are associated.

BEA Connect OSI TP User Guide

| Document Edition | Part Number | Date | Software Version |
|------------------|----------------|----------|--------------------|
| 1.3 | 825-001003-001 | May 1998 | Connect OSI TP 1.3 |

Contents

Preface

| | |
|---|------|
| Purpose of This Manual..... | vii |
| Who Should Read This Manual | vii |
| How This Manual Is Organized | vii |
| How to Use This Manual..... | viii |
| Online Document Considerations | viii |
| Opening the Manual in a Web Browser | viii |
| Printing from a Web Browser | x |
| Document Conventions | x |
| Related Documentation | xi |
| Connect OSI TP Documentation..... | xi |
| Product Manuals..... | xi |
| Other Publications | xi |
| Contact Information..... | xii |
| Documentation Support..... | xii |
| Customer Support..... | xii |

1. Introducing BEA Connect OSI TP

| | |
|--|-----|
| Connect OSI TP Overview | 1-1 |
| Establishing Remote Connections..... | 1-3 |
| Connect OSI TP Features | 1-4 |
| BEA Connect OSI TP and BEA TUXEDO Architecture | 1-4 |
| OSI TP Domains Components | 1-7 |
| Gateway Servers | 1-9 |

2. Installing BEA Connect OSI TP

| | |
|---|------|
| Pre-Installation Considerations..... | 2-1 |
| Configuring the Environment to Install Connect OSI TP | 2-2 |
| Upgrading BEA Connect OSI TP..... | 2-2 |
| Installing BEA Connect OSI TP..... | 2-2 |
| Installing on Unix-based Platforms | 2-3 |
| Installing on a Windows NT Platform..... | 2-6 |
| Distribution Libraries and Executables | 2-13 |
| HP-UX 10.20..... | 2-13 |
| SUN Solaris 2.5.1 | 2-14 |
| UNIXWARE 2.1 | 2-14 |
| Windows NT 4.0 | 2-15 |
| Uninstalling Connect OSI TP on Windows NT | 2-15 |

3. Configuring BEA Connect OSI TP

| | |
|---|------|
| Defining New Gateway Configurations | 3-1 |
| Editing the DMCONFIG File | 3-2 |
| Prerequisites | 3-2 |
| Configuration File Format..... | 3-2 |
| Editing Procedure..... | 3-4 |
| DMCONFIG Parameters | 3-5 |
| *DM_LOCAL_DOMAINS Section | 3-5 |
| *DM_REMOTE_DOMAINS Section | 3-7 |
| *DM_TDOMAIN Section..... | 3-8 |
| *DM_OSITP Section | 3-10 |
| *DM_ACCESS_CONTROL Section | 3-12 |
| *DM_LOCAL_SERVICES Section | 3-13 |
| *DM_REMOTE_SERVICES Section | 3-14 |
| *DM_ROUTING Section | 3-16 |
| Sample Configuration Files..... | 3-18 |
| Modifying an Existing Gateway Configuration | 3-22 |
| Using the dmadm Command..... | 3-22 |
| Defining Connect OSI TP Servers to BEA TUXEDO | 3-22 |
| Sample UBBCONFIG File..... | 3-23 |

4. Security

| | |
|---|-----|
| Where You Specify Security Parameters | 4-2 |
| UBBCONFIG File Security Parameters | 4-3 |
| DMCONFIG File Security Parameters | 4-3 |
| How To Administer Security | 4-5 |
| Adding a Userid and Password | 4-5 |
| Mapping a Userid | 4-5 |
| Removing a Userid's Mapping | 4-6 |
| Deleting a Userid and Password | 4-6 |
| Modifying a Password | 4-7 |

5. Data Translations

| | |
|--|-----|
| Layout Conversion for Buffer Types | 5-2 |
| ASN.1 Encoding..... | 5-2 |
| Summary of Mappings | 5-2 |
| Application Programming | 5-4 |
| Supported ATMI Calls..... | 5-4 |
| Considerations for Selecting Buffers | 5-5 |
| System Programming | 5-6 |
| Defining VIEWS | 5-6 |
| Compiling VIEW Files | 5-6 |

A. Error and Informational Messages

B. Reference Pages

| | |
|-----------------|------|
| addumap | B-2 |
| addusr | B-4 |
| build_dgw..... | B-6 |
| delumap | B-9 |
| delusr | B-11 |
| DMADM..... | B-13 |
| dmadmin..... | B-15 |
| dmconfig | B-34 |
| dmloadcf..... | B-56 |
| dmunloadcf..... | B-59 |

| | |
|----------------|------|
| dmusradd | B-60 |
| dmusrmod..... | B-62 |
| GWADM | B-64 |
| modusr | B-67 |

Preface

Purpose of This Manual

This guide describes the BEA Connect OSI TP™ product and explains how to configure and administer BEA Connect OSI TP.

Who Should Read This Manual

This document is intended for system administrators who will configure and administer BEA Connect OSI TP.

How This Manual Is Organized

The *BEA Connect OSI TP User Guide* is organized as follows:

- ◆ Chapter 1, “Introducing BEA Connect OSI TP,” describes basic BEA Connect OSI TP concepts.
- ◆ Chapter 2, “Installing BEA Connect OSI TP,” describes tasks you must complete to install BEA Connect OSI TP.
- ◆ Chapter 3, “Configuring BEA Connect OSI TP,” explains how to create the configuration file for BEA Connect OSI TP.
- ◆ Chapter 4, “Security,” describes the administrative commands that create, modify, and delete userids and passwords for Connect OSI TP.

-
- ◆ Chapter 5, “Data Translations,” describes the typed buffers that are supported by BEA Connect OSI TP.
 - ◆ Appendix A, “Error and Informational Messages,” provides the message number and description for BEA Connect OSI TP messages.
 - ◆ Appendix B, “Reference Pages,” provides reference pages for several TUXEDO commands.

How to Use This Manual

Online Document Considerations

This document, *BEA Connect OSI TP User Guide*, is designed primarily as an online, hypertext guide. If you are reading this as a paper publication, note that to get full use from this guide you should install and access it as an online document via a Web browser that supports HTML 3.0. Netscape Navigator 2.02 or Microsoft Internet Explorer 3.0 or later are recommended. (Information on how to install the online documentation is available in the *BEA Connect OSI TP Release Notes*.)

Opening the Manual in a Web Browser

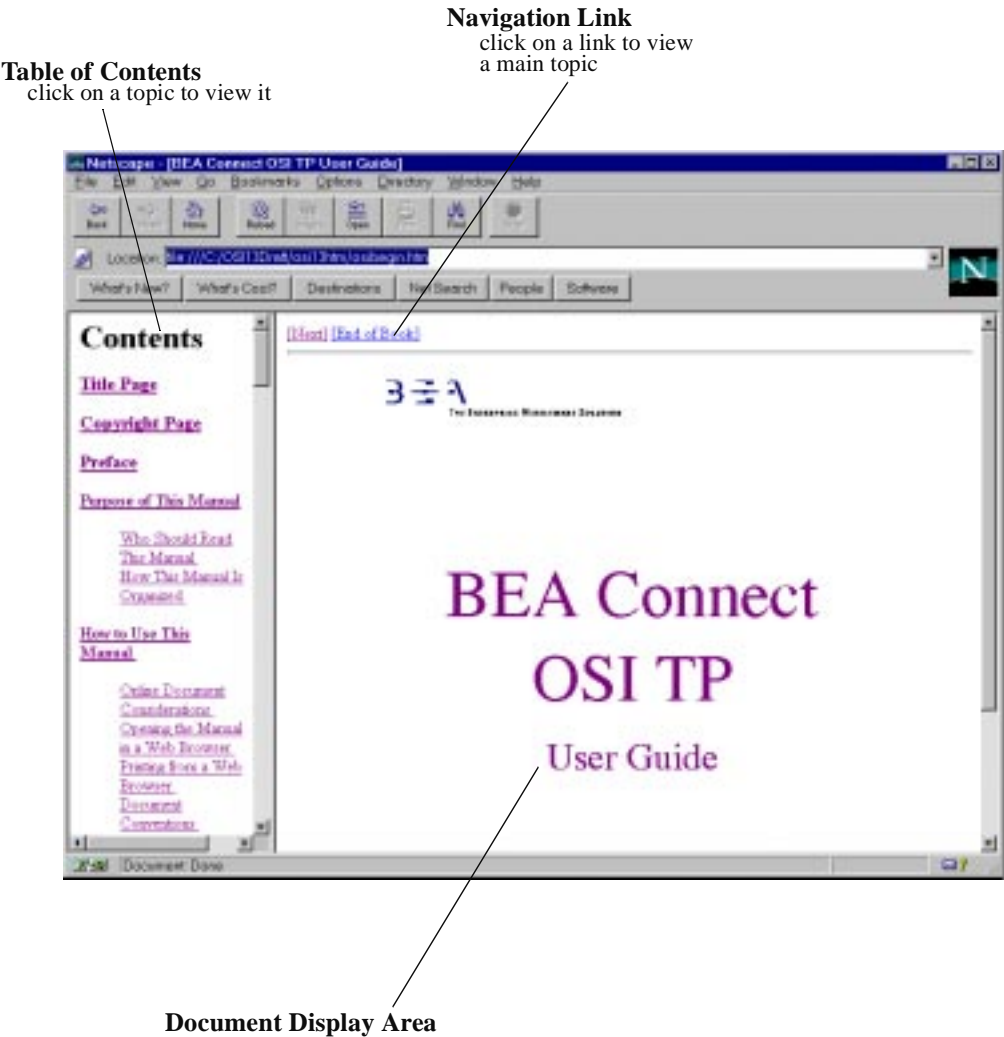
To access the online version of this document, open the following HTML file in a Web browser:

`http://(directory path to Connect OSI TP HTML files)/osibegin.htm`

Note: The online documentation requires a Web browser that supports HTML 3.0. Netscape Navigator 2.02 or Microsoft Internet Explorer 3.0 or later are recommended.

Figure 1 shows the online guide with the clickable navigation bar and table of contents.

Figure 1 BEA Connect OSI TP User Guide Displayed in Netscape Web Browser



Printing from a Web Browser

You can print a hardcopy version of this document, one file at a time, from the Web browser. Before you print, make sure that the chapter or appendix you want is displayed and *selected* in your browser. (To select a file, click anywhere inside the frame you want to print. If your browser offers a Print Preview feature, you can use it to verify which file you are about to print.)

Document Conventions

The following documentation conventions are used throughout this manual:

| Item | Convention | Example |
|---------------------------|--|--|
| Arguments | appear in parentheses and are formatted in a lowercase monospace font. Optional arguments are formatted in italic font. Predefined arguments are formatted in an uppercase font. | <i>(name, 0, value)</i> (ACCTID, 2, 5000) |
| Environment variables | are formatted in an uppercase font. | ENVFILE=\${APPDIR} |
| Key names | are presented in boldface type. | Press Enter to continue. |
| Literals | are formatted in a monospace font. | <code>class extendSample</code> |
| Programs and applications | are formatted with initial caps. | Use the Repository Editor and the Class Library. |
| User input | are formatted in a monospace font. | Type <code>cd TUXDIR</code> |
| Window items | are presented in boldface type. Window items can be window titles, button labels, text edit box names or other parts of the window. | Type your password in the Logon window . Select Export to make the service available to the client. |

Related Documentation

Connect OSI TP Documentation

The Connect OSI TP documentation consists of the following items:

BEA Connect OSI TP User Guide

BEA Connect OSI TP Release Notes

Product Manuals

TUXEDO System 6 Reference Manual

TUXEDO System 6 Programmer's Guide, Volumes 1 and 2

Other Publications

The TUXEDO System (Andrade, Carges, Dywer, Felts)

TUXEDO: An Open Approach to OLTP (Primatesta)

Building Client/Server Applications Using TUXEDO (Hall)

Contact Information

Documentation Support

If you have questions or comments on the documentation, you can contact the BEA Information Engineering Group by e-mail at **docsupport@beasys.com** or by telephone at **+1.408.542.4193**. (For information on how to contact Technical Support, refer to the following section.)

Customer Support

If you have any questions about this version of BEA Connect OSI TP, or if you have problems installing and running BEA Connect OSI TP, contact BEA Customer Support at one of the following telephone numbers or e-mail addresses, or through our Web site (www.beasys.com):

North American Support Center
Sunnyvale, CA, USA
1-888-232-7878
1-408-743-4070
1-408-743-4071 fax
email: support@beasys.com

European Support Center
Paris, France
+33-1-41-45-7090
+33-1-41-45-7009 fax
email: support@beasys.fr

Asia Pacific Support Center
Brisbane, Australia
+61-7-3255-0506 phone
+61-7-3255-0441 fax
email: support@beasys.com.au

Japan Support Center:
Yokohama, Japan
+81-4-5224-1250 phone
+81-4-5224-1251 fax
email: support@beasys.co.jp

When contacting technical support, be prepared to provide the following information:

- ◆ Your name, e-mail address, phone number, and fax number
- ◆ Your company name and company address
- ◆ Your machine type and authorization codes
- ◆ The name and version of the product you are using
- ◆ A description of the problem and the content of pertinent error messages



1 Introducing BEA Connect OSI TP

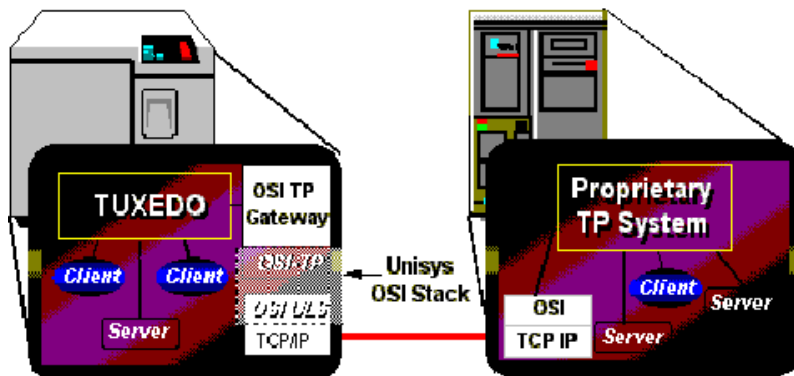
This chapter will cover the following topics:

- ◆ Connect OSI TP Overview
- ◆ Establishing Remote Connections
- ◆ Connect OSI TP Features
- ◆ BEA Connect OSI TP and BEA TUXEDO Architecture

Connect OSI TP Overview

BEA Connect OSI TP is a gateway connectivity feature that, together with Unisys OSI-TP software and Unisys Upper Layer Services (ULS) software, makes it possible for OLTP application programs on BEA TUXEDO systems to perform global transactions and various non-transactional tasks with application programs in the following environments:

- ◆ Other BEA TUXEDO applications. An application (or administrative domain) is a single computer or network of computers that share a single BEA TUXEDO configuration.
- ◆ Other systems that implement the X/Open XATMI standard interface and OSI-TP standard protocols. These include Unisys A Series enterprise servers or Unisys OS-2200 enterprise servers that support Open/OLTP software and OSI-TP.



Unisys OSI Distributed Transaction Processing (OSI-TP) is a software product that implements the OSI-TP standard, a set of protocols that is used to:

- ◆ Establish and support dialogs between application programs on different computers.
- ◆ Facilitate commitment and rollback of global transactions that span multiple computers.

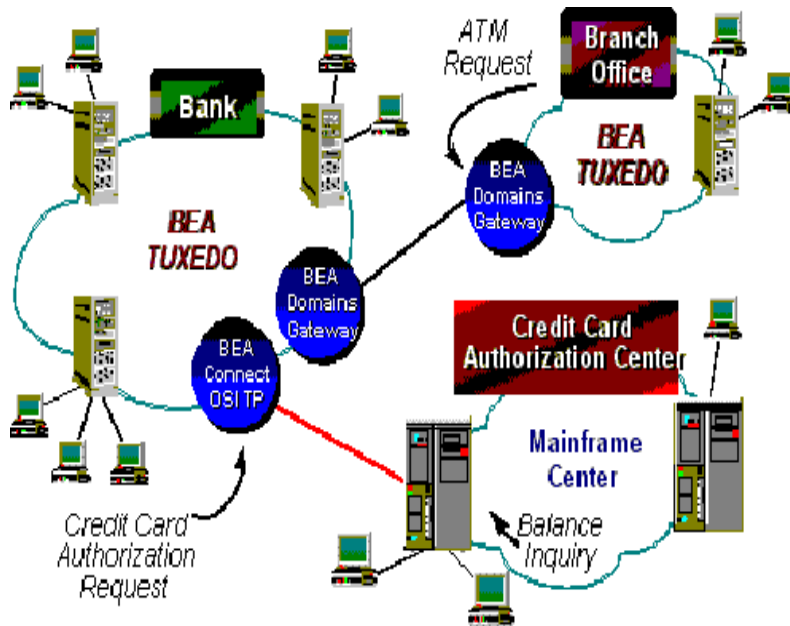
Unisys TransIT CONNECT Upper Layer Services is a software product that implements various OSI application, presentation, and session layer services. ULS is used in conjunction with TCP/IP subnetworks.

The X/Open XATMI standard is an interface that application programs use to communicate with other application programs both inside and outside of global transactions. It supports conversational and request/reply communication styles and is fully implemented by BEA Connect OSI TP.

Data mapping and transformation between the BEA TUXEDO and mainframe environments is easily automated in BEA TUXEDO-based applications with the BEA TUXEDO typed buffer mechanism. This mechanism allows system administrators to predefine how data should be conveyed to the remote application. Application programmers do not need to be aware of this translation; they can simply continue using the buffer types defined for the local application.

BEA Connect OSI TP is designed to provide transparent access to remote services that reside outside a BEA TUXEDO application. In addition, BEA Connect OSI TP provides remote application programs access to local services.

Figure 1-1 BEA Connect OSI TP Sample Environment



Establishing Remote Connections

BEA Connect OSI TP and other BEA Connect products act as gateways between BEA TUXEDO systems and other online transaction processing environments. To establish connections with remote systems, an administrator will:

- ◆ Configure BEA Connect OSI TP as an ordinary BEA TUXEDO server group by updating the BEA TUXEDO configuration.
- ◆ Identify remote systems and available services by updating the BEA Connect OSI TP configuration.

Connect OSI TP Features

BEA Connect OSI TP supports the following features:

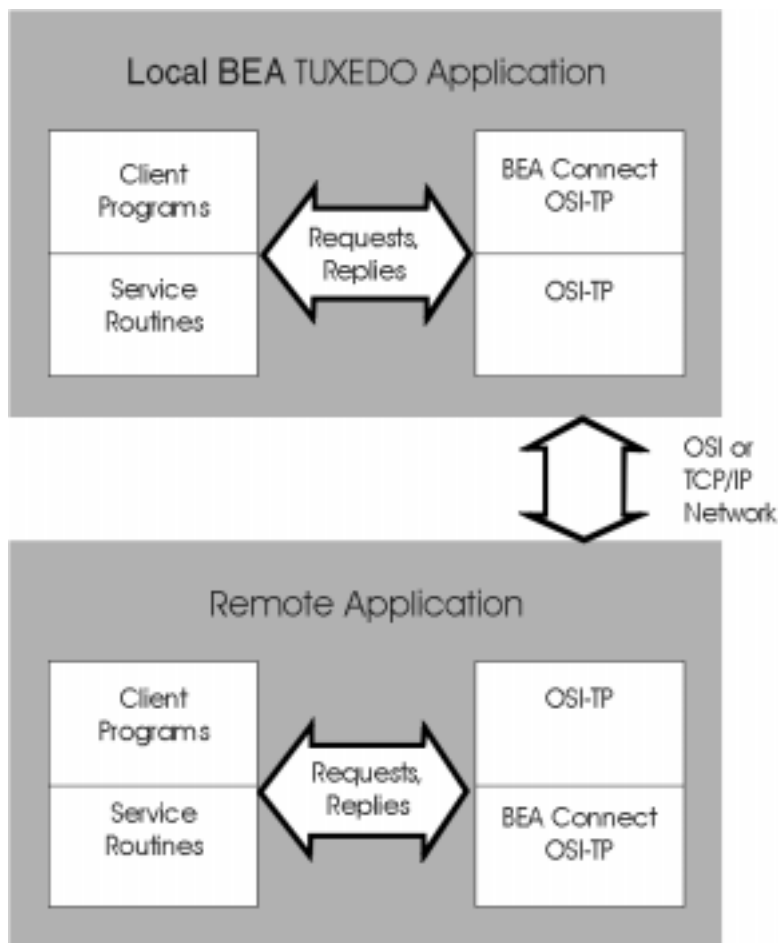
- ◆ Synchronous and asynchronous service requests and replies initiated from either the BEA TUXEDO or mainframe environments.
- ◆ Conversational service requests and replies initiated from either the BEA TUXEDO or the mainframe environments.
- ◆ Global Transactions with OSI TP partners.
- ◆ Event monitoring and reporting.
- ◆ Static and dynamic configuration support.
- ◆ Bundled Unisys Protocol Stack (OSI TP / OSI ULS).
- ◆ ASN.1 based data conversion.
- ◆ Security through Access Control Lists on the local domain.
- ◆ Multiple simultaneous request/response sessions and conversations.
- ◆ Connections to multiple Domain.

BEA Connect OSI TP and BEA TUXEDO Architecture

A BEA TUXEDO application consists of client and server programs that operate across a network of BEA TUXEDO systems. Any client program can request services that are offered by any server program running on any computer in the application. Also, the location of server programs is kept transparent by way of a directory that maps services to servers.

As Figure 1-2 shows, BEA Connect OSI TP extends this transparent access by sending requests to and receiving requests from remote systems through OSI TP and supporting network software.

Figure 1-2 Routing Service Calls through BEA Connect OSI TP



As Figure 1-2 suggests, inside a single application, BEA Connect OSI TP operates between the BEA TUXEDO software and OSI TP.

- ◆ When local BEA TUXEDO client programs send requests to remote systems, BEA Connect OSI TP transforms those requests into OSI TP messages. Also, when remote systems respond, BEA Connect OSI TP transforms associated OSI TP messages into replies that local client programs can process.
- ◆ When remote client programs send requests (that arrive at the local application as OSI TP messages), BEA Connect OSI TP transforms those messages into requests that local BEA TUXEDO service routines can process. Also, when local service routines send replies, BEA Connect OSI TP transforms those replies into OSI TP messages.
- ◆ When commitment protocol is sent for global transactions, BEA Connect OSI TP transforms the protocol and manages the transaction commitment.

BEA Connect OSI TP is implemented as an ordinary BEA TUXEDO server group. It accepts standard BEA TUXEDO service requests and returns standard replies.

The BEA Connect OSI TP server group consists of the following components:

- ◆ A server program that includes the TMS service.
- ◆ An administrative server.

One BEA Connect OSI TP server group acts as a gateway to multiple communications targets. Each communications target is a unique OSI TP endpoint.

Some remote targets, such as remote BEA TUXEDO applications, also support BEA Connect OSI TP. In this situation, BEA Connect OSI TP servers associated with the local gateway communicate with BEA Connect OSI TP servers associated with remote gateways through OSI TP.

Other gateways, such as remote Unisys A Series Open/OLTP systems and OS2200 Open/OLTP systems, provide analogous functionality to which local BEA Connect OSI TP servers can interact.

Although remote systems are identified in the BEA Connect OSI TP configuration, they remain unknown to BEA TUXEDO software. For example, remote systems that are accessible through BEA Connect OSI TP are not identified in the *MACHINES section of the UBBCONFIG file. Also, UBBCONFIG files are not propagated to remote machines that are accessible through BEA Connect OSI TP.

Remote services identified in the BEA Connect OSI TP configuration are advertised to the BEA TUXEDO software directly by BEA Connect OSI TP gateway servers. The remote services are not identified in the *SERVICES section of the UBBCONFIG file.

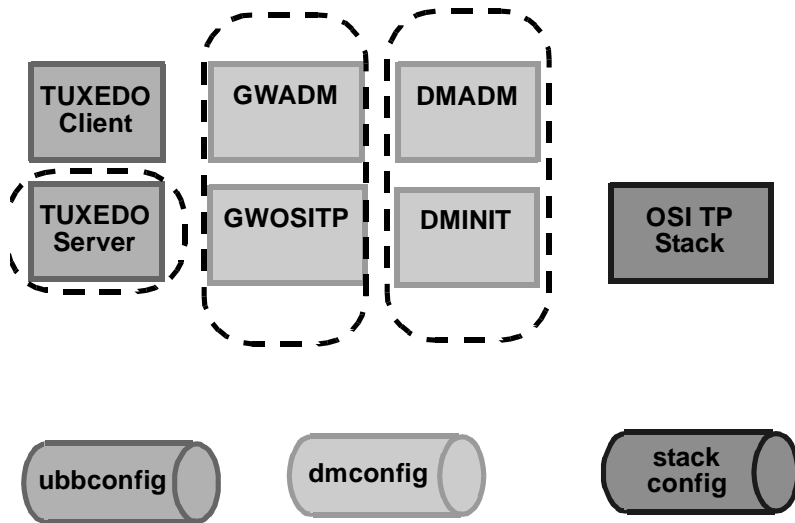
BEA Connect OSI TP maintains its own control information in shared memory, in much the same way that BEA TUXEDO software itself maintains the Bulletin Board. *Although BEA Connect OSI TP accesses the BEA TUXEDO Bulletin Board, BEA TUXEDO does not access BEA Connect OSI TP control information.*

Note: To a remote system that supports the X/Open XATMI standard (more specifically, an XATMI application service element), the BEA Connect OSI TP server group appears as a communications resource manager (CRM).

OSI TP Domains Components

The Connect OSI TP is composed of several elements that can be configured to provide OSI TP solutions. For the most part, the OSI TP domain is much like the other domain gateways. It uses the DMADM and GWADM servers provided with BEA TUXEDO for administration. The following diagram describes each component of the Connect OSI TP product:

Figure 1-3 Domain Components



LEGEND:

DMADM = Domain Administration Server

DINIT = Initialization Server

GWADM = Gateway Administration Server

GWOSITP = Transactional/OSI TP Domains Instance

OSI TP Stack = Unisys-supplied OSI TP Stack

Gateway Servers

- ◆ DMADM - Domain administration server
- ◆ GWADM- Gateway administration server
- ◆ GWOSITP - Transactional /OSI TP Domains Instance

The OSI TP Domain Gateway (GWOSITP) uses the administrative servers provided with BEA TUXEDO for domain and gateway configuration and administration. Specifics of configuring the various sections of a domain are covered in the BEA TUXEDO document *BEA TUXEDO /Domain Guide*, as well as the references page for dmconfig in Appendix B, “Reference Pages”.

2 Installing BEA Connect OSI TP

This chapter contains information for installing and uninstalling the BEA Connect OSI TP product.

Pre-Installation Considerations

Connect OSI TP runs on UNIXWARE, HP, SUN, and Windows NT platforms. Complete the following tasks prior to installing Connect OSI TP:

- ◆ Read the *BEA Connect OSI TP Release Notes*
- ◆ Install and verify the operation of the BEA TUXEDO 6.4 product
- ◆ Install and verify the Unisys OSI-TP Version 2.0.8 stack software
- ◆ The following server platforms are supported for BEA Connect OSI TP:
 - ◆ HP-UX Version 10.20
 - ◆ SUN Solaris Version 2.5.1
 - ◆ UNIXWARE Version 2.1
 - ◆ Microsoft Windows NT Version 4.0

Configuring the Environment to Install Connect OSI TP

Before installing BEA Connect OSI TP, you must configure the environment properly. Ensure that BEA TUXEDO 6.4 and Unisys OSI-TP Version 2.0.8 stack software are configured properly.

Upgrading BEA Connect OSI TP

If you are upgrading from a previous release of the BEA Connect OSI TP product, you must shut down all Domain administrative and gateway servers within an application domain, particularly the following:

- ◆ GWTDOMAIN
- ◆ GWADM
- ◆ DMADM

Warning: Do *not* run `dmadmin`, `dmlaodcf`, and `dmunloadcf` commands until the installation is complete.

BEA Connect OSI TP is a full replacement for previous releases. To upgrade, you will need to reinstall the product.

Installing BEA Connect OSI TP

Connect OSI TP will run on Unix-based platforms (UNIXWARE, HP, and SUN) and Windows NT. Refer to the appropriate platform sections that follow for installation instructions.

Installing on Unix-based Platforms

The following steps will install Connect OSI TP on a Unix-based platform.

1. Log on as root to install Connect OSI TP.

Listing 2-1 Log On as Root

```
$ su -  
Password:
```

2. Mount the CD-ROM. The following example will mount the CD-ROM drive on a UNIXWARE system.

Listing 2-2 Mount the CD-ROM Drive

```
# ls -l /dev/cdrom  
total 0  
brw-rw-rw-  1 root    sys      22,  0 Sep 16 10:55 clb0t010  
# mount -r -F cdfs /dev/cdrom/clb0t010 /mnt  
# cd /mnt  
# ls  
install.sh  uw              winnt
```

3. Run the installation script by typing the following command.

Listing 2-3 Run the Installation Script

```
# sh ./install.sh
```

4. The installation script will prompt you for responses. In the sample installation that follows, responses are in bold.

Listing 2-4 Sample UNIXWARE Installation

```
01) uw/uw21
```

```
Install which platform's binaries? [01-l, q to quit, l for list]: 01
```

```
** You have chosen to install from uw/uw21 **
```

```
BEA Connect OSI TP Release 1.3 for TUXEDO System Release 6.4
```

```
This directory contains the BEA Connect OSI TP SDK for  
UnixWare 2.1 on Intel i386.
```

```
Please refer to the product documentation for release notes and  
further information.
```

```
Is this correct? [y,n,q]: y
```

```
To terminate the installation at any time  
press the interrupt key,  
typically <del>, <break>, or <ctrl+c>.
```

```
The following packages are available:
```

```
1      tuxositp      BEA Connect OSI TP
```

```
Select the package(s) you wish to install (or 'all' to install  
all packages) (default: all) [?,??,q]: all
```

```
BEA Connect OSI TP  
(i386) Release 6.4
```

```
Copyright (c) 1998 BEA Systems, Inc.
```

```
Portions of this software (c) 1995 Novell, Inc.
```

```
All Rights Reserved. Distributed under license by BEA Systems, Inc.  
TUXEDO (r) is a registered trademark of Novell, Inc. in the  
United States and other countries.
```

Checking UnixWare version...
...finished

The following installation options are available:

- 1 both Install the complete TUXEDO System - client and server
- 2 client Install the client only

Select an option (default: both) [?,??,q]: 1

Directory where TUXEDO files are to be installed [?,q]: **/usr/tuxedo**

Determining if sufficient space is available ...
6931 blocks are required
2982512 blocks are available to /usr/tux64conn

Using /usr/tux64conn as the TUXEDO base directory

Unloading /mnt/uw/uw21/tuxositp/ws.z ...
bin/GWOSITP
lib/libgwo.a
lib/libgwo.so.60
lib/libtasnl.a
lib/libtasnl.so.60
locale/C/LIBGWO.text
locale/C/LIBGWO_CAT
6960 blocks
... finished

Changing file permissions...
...done

Installation of BEA Connect OSI TP was successful

Please don't forget to fill out and send in your registration card
#

5. Unmount the CD-ROM by typing a command similar to the following.

Listing 2-5 Unmount the CD-ROM

```
# cd /  
# umount /mnt  
# exit  
$ exit
```

6. Use a text editor to add the following line of text to the `udataobj/DMTYPE` file.
If a line beginning with `OSITP` exists, then delete it and replace it with the following new line of text.

Listing 2-6 Edit udataobj/DMTYPE File

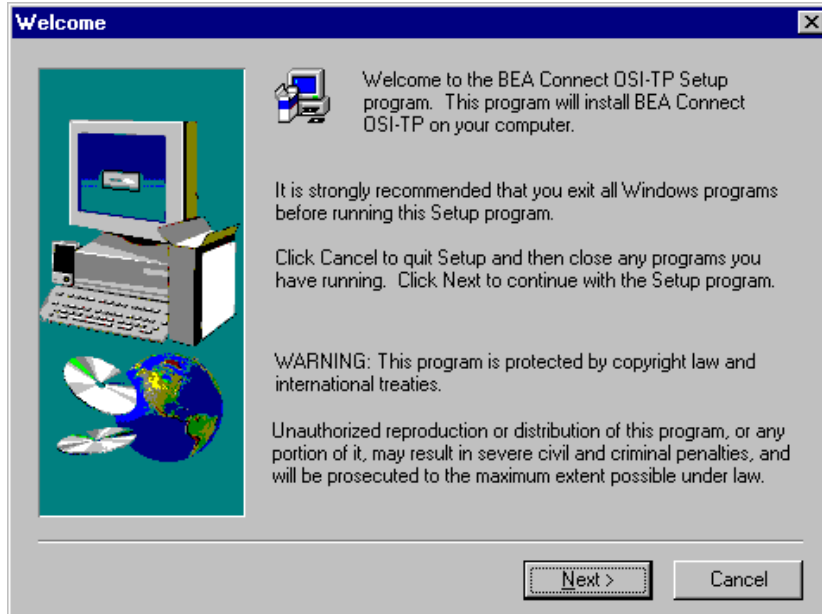
```
OSITP:-lgwo:-ltasn1 -lnwi -lnws -lnwi -lxaptp::
```

Installing on a Windows NT Platform

The following steps will install Connect OSI TP on a Windows NT system.

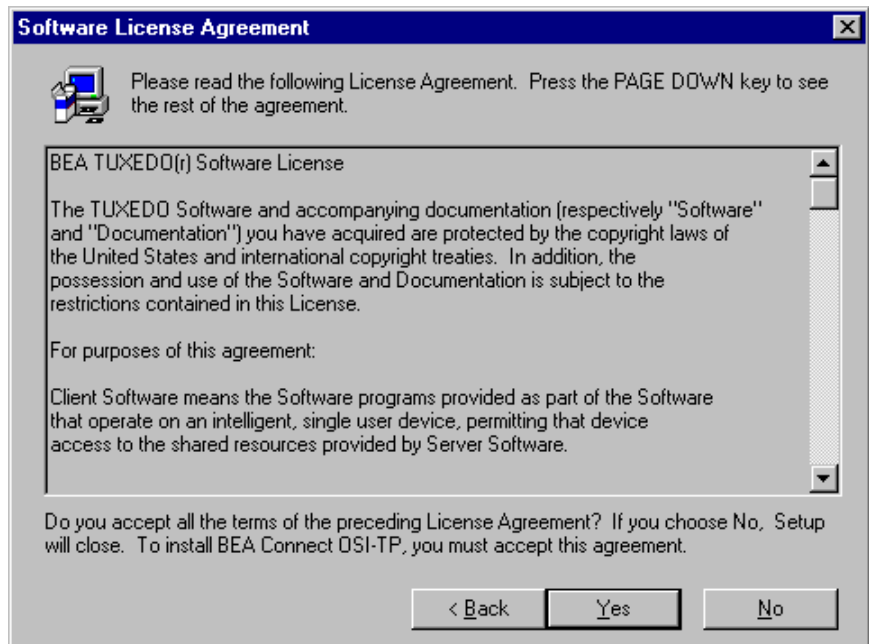
1. Insert the product CD-ROM and click on the **Run** option from the **Start menu**. The **Run** window displays. Click on the **Browse** button to select the CD-ROM drive. Change directories to the winnt directory and select the Setup.exe program. Click **OK** to run the executable and begin the installation. The following **Welcome** screen displays. Click **Next** to continue with the installation.

Figure 2-1 Welcome Screen



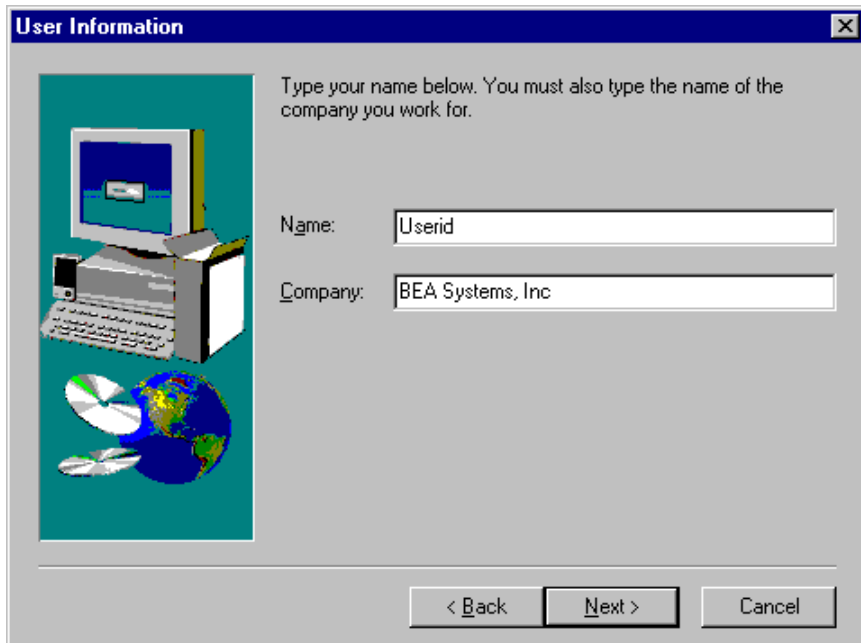
2. The BEA Software License Agreement displays. Click **Yes** to accept the terms of the agreement and continue with the product installation. Click **No** to exit the installation process.

Figure 2-2 BEA Software License Agreement



3. The **User Information** screen displays after the License Agreement. Enter the name of the BEA TUXEDO System Administrator in the **Name** field. Enter the name of your company in the **Company** field. Click **Next** to continue with the installation.

Figure 2-3 User Information



User Information

Type your name below. You must also type the name of the company you work for.

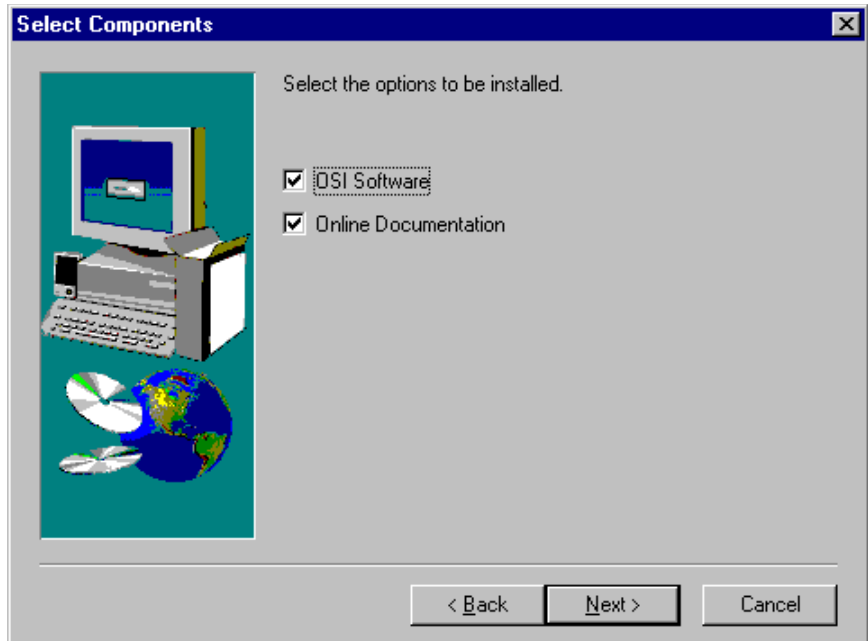
Name:

Company:

< Back Next > Cancel

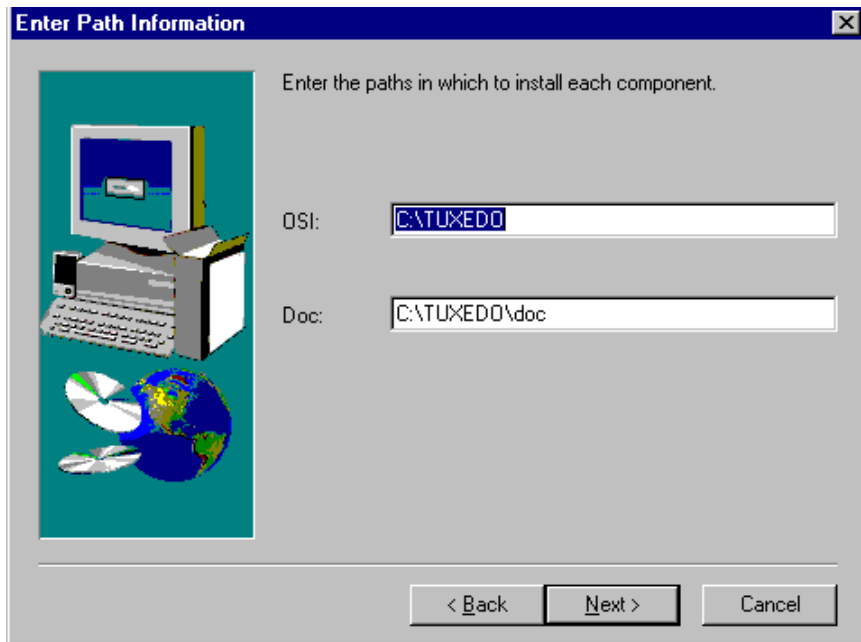
4. The **Select Components** screen displays after the **User Information** screen. Click the check box to the left of the **OSI Software** option to install the Connect OSI TP product. Click the check box to the left of the **Online Documentation** option to load the Connect OSI TP online documentation. Click **Next** to continue with the installation.

Figure 2-4 Component Selection



5. The **Enter Path Information** screen displays next. Enter the directory path where you want to install the BEA Connect OSI TP product in the **OSI** field. Enter the directory path for the BEA Connect OSI TP online documentation in the **Doc** field. Click **Next** to continue with the installation.

Figure 2-5 Path Definition



Enter Path Information

Enter the paths in which to install each component.

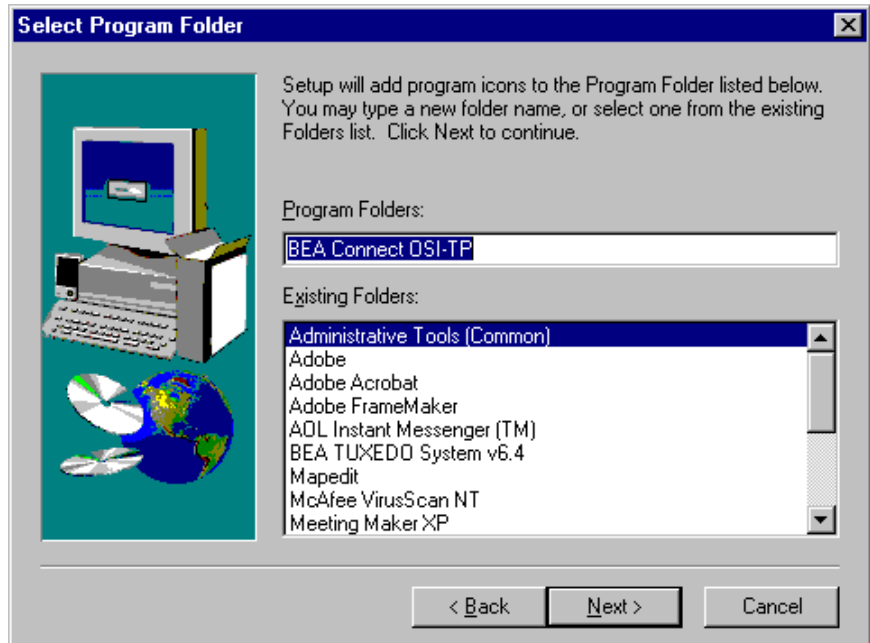
OSI: C:\TUXEDO

Doc: C:\TUXEDO\doc

< Back Next > Cancel

6. The **Select Program Folder** screen displays next prompting you to name a folder for the Connect OSI TP files. Enter a folder name in the **Program Folders** field or select a name from the **Existing Folders** list. Program icons for Connect OSI TP will be added in the designated folder. Click **Next** to continue with the installation.

Figure 2-6 Folder Selection



7. A progress bar displays showing the status of the installation.
8. The **Setup Complete** screen displays notifying you that the BEA Connect OSI TP product is installed on your system. Click **Finish** to complete the Setup process.
9. Use a text editor such as Microsoft NotePad to add the following line to the `udataobj/DMTYPE` file. If a line beginning with `OSITP` exists in the file, then delete it and replace it with the following new text.

Listing 2-7 Edit udataobj/DMTYPE File

```
OSITP;%TUXDIR%\lib\libgwo.lib;%TUXDIR%\lib\libtasn1.lib;%TUXDIR%\lib\libnwi.lib %TUXDIR%\lib\libnws.lib %TUXDIR%\lib\libnwi.lib; ;
```

Distribution Libraries and Executables

The BEA Connect OSI TP CD-ROM contains the following libraries and executable programs. After installing the Connect OSI TP software, verify that these libraries and programs are installed on your system.

HP-UX 10.20

Verify that the following files are installed by BEA Connect OSI TP:

| Directory | Files |
|-----------|--|
| /bin | GWOSITP |
| /lib | libgwo.a libgwo.sl libtasn1.a libtasn1.sl |
| /locale/C | LIBGWO.text LIBGWO_CAT |

SUN Solaris 2.5.1

Verify that the following files are installed by BEA Connect OSI TP:

| Directory | Files |
|-----------|--|
| /bin | GWOSITP |
| /lib | libgwo.a libgwo.so libtasn1.a libtasn1.so |
| /locale/C | LIBGWO.text LIBGWO_CAT |

UNIXWARE 2.1

Verify that the following files are installed by BEA Connect OSI TP:

| Directory | Files |
|-----------|--|
| /bin | GWOSITP |
| /lib | libgwo.a libgwo.so libtasn1.a libtasn1.so |
| /locale/C | LIBGWO.text LIBGWO_CAT |

Windows NT 4.0

Verify that the following files are installed by BEA Connect OSI TP:

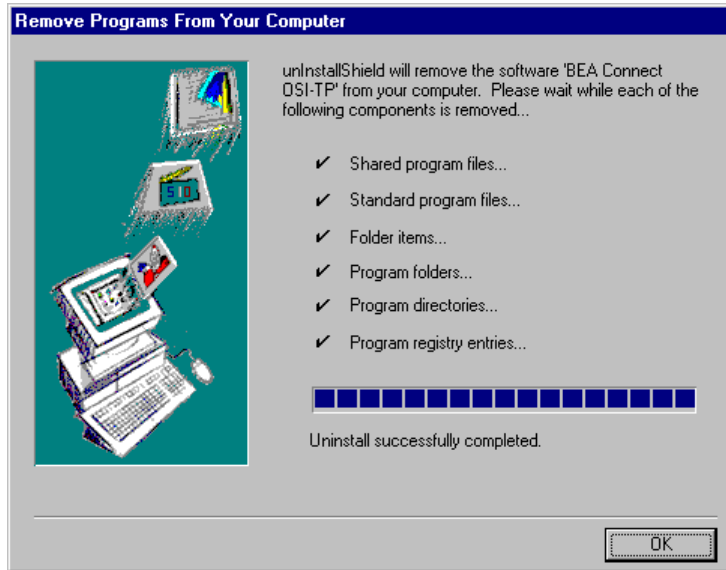
| Directory | Files |
|-----------|---|
| /bin | gwositp.exe |
| /lib | libgwo.dll libgwo.lib libtasn1.dll libtasn1.lib libgwo.text libgwo_cat |

Uninstalling Connect OSI TP on Windows NT

The following steps will uninstall the BEA Connect OSI TP product on a Windows NT system.

1. Access the **Control Panel** window from the **Start>Settings>Control Panel** menu option.
2. Double click on the **Add/Remove Programs** option from the Control Panel listings to access the **Add/Remove Programs Properties** window.
3. In the **Add/Remove Programs Properties** window, select BEA Connect OSI TP from the program list and click on the **Add/Remove** button.
4. The uninstall process for BEA Connect OSI TP will begin. The **Remove Programs From Your Computer** screen displays. Click **OK** to complete the uninstall process.

Figure 2-7 Removing Programs From Your Computer



3 Configuring BEA Connect OSI TP

Once the installation of Connect OSI TP is complete, you will want to configure Connect OSI TP. The proper configuration of Connect OSI TP sets up the gateway configuration. Configuring the Connect OSI TP gateway includes:

- ◆ Defining new gateway configurations or modifying existing gateway configuration.
- ◆ Defining the Connect OSI TP administrative and gateway servers to the BEA TUXEDO system.
- ◆ Verifying the configuration of Connect OSI TP.

Defining New Gateway Configurations

To define a new gateway configuration, complete the following steps.

1. Edit the DMCONFIG file.
2. Process the DMCONFIG file by running the `dmloadcf` utility.

Editing the DMCONFIG File

The configuration specified in the DMCONFIG file controls much of the operation of the BEA Connect OSI TP gateway. A sample of this file is provided in the installation directory of your BEA Connect OSI TP product software.

DMCONFIG is the ASCII version of a TUXEDO System/Domain domain configuration file. The DMCONFIG file is parsed and loaded into a binary version by the `dmloadcf` utility. The binary configuration file, called the BDMCONFIG file, contains information used by domain gateways to initialize the context required for communications with other domains. `dmadmin` uses the binary file (or a copy of it) in its monitoring activity. There will be one BDMCONFIG file for each TUXEDO System/Domain application that uses the /Domain feature.

A DMCONFIG file, and its binary BDMCONFIG counterpart, are analogous to the UBBCONFIG and TUXCONFIG files of a non-/Domain System/T application. The DMCONFIG file extends the definition of a non-/Domain System/T application so that the application becomes a domain.

Prerequisites

The BEA Connect OSI TP product software must be installed and accessible to your text editor. You must have file permission to access the `install` directory and modify the sample DMCONFIG file. In addition, the following prerequisites must be met to successfully complete the editing procedure:

- ◆ The `$TUXDIR/udataobj/DMTYPE` file defining the valid domain types must exist. Refer to the Appendix B, “Reference Pages,” for information on the `dmloadcf` command.
- ◆ The effective user identifier of the person running `dmloadcf` must match the UID in the RESOURCES section of the TUXCONFIG file. Refer to the `dmloadcf` reference page in Appendix B, “Reference Pages”.

Configuration File Format

The format of a domain configuration file is as follows:

- ◆ The file is made up of eight possible specification sections. Lines beginning with an asterisk (*) indicate the beginning of a specification section. Each such line

contains the name of the section immediately following the *. Allowable section names are: DM_LOCAL_DOMAINS, DM_REMOTE_DOMAINS, DM_LOCAL_SERVICES, DM_REMOTE_SERVICES, DM_ROUTING, DM_ACCESS_CONTROL and DM_domtype, where domtype is OSITP or TDOMAIN. The DM_LOCAL_DOMAINS section must precede the DM_REMOTE_DOMAINS section.

- ◆ Parameters are generally specified by: *KEYWORD* = *value*. This sets *KEYWORD* to *value*. Valid keywords are described in the following sections. *KEYWORDS* are reserved; they cannot be used as *values* unless they are quoted.
- ◆ Lines beginning with the reserved word, DEFAULT:, contain parameter specifications that apply to any lines that follow them in the section in which they appear. Default specifications can be used in all sections. They can appear more than once in the same section. The format for these lines is:

DEFAULT: [*KEYWORD1* = *value1* [*KEYWORD2* = *value2* [...]]]

The values set on this line remain in effect until reset by another DEFAULT: line, or until the end of the section is reached. These values can also be overridden on non-DEFAULT: lines by placing the optional parameter setting on the line. If on a non-DEFAULT: line, the parameter setting is valid for that line only; lines that follow revert to the default setting. If DEFAULT: appears on a line by itself, all previously set defaults are cleared and their values revert to the system defaults.

- ◆ If a value is *numeric*, standard C notation is used to denote the base (that is, 0x prefix for base 16 (hexadecimal), 0 prefix for base 8 (octal), and no prefix for base 10 (decimal)). The range of values acceptable for a numeric parameter are given under the description of that parameter.
- ◆ If a value is an *identifier*, standard C rules are used. An *identifier* must start with an alphabetic character or underscore and contain only alphanumeric characters or underscores. The maximum allowable length of an identifier is 30 (not including the terminating null). An identifier cannot be the same as any *KEYWORD*.
- ◆ A value that is neither an integer number nor an identifier must be enclosed in double quotes. Certain special characters can be escaped inside a string using a backslash. “\” translates to a single backslash. “\\” translates to a double quote. “\n” translates to a new line. “\t” translates to a tab. “\f” translates to a form feed. “\x” (where ‘x’ is any character other than one of the previously mentioned special characters) translates to ‘x’.

- ◆ Input fields are separated by at least one space (or tab) character.
- ◆ "#" introduces a comment. A new line ends a comment.
- ◆ Blank lines and comments are ignored.
- ◆ Comments can be freely attached to the end of any line.
- ◆ Lines are continued by placing at least one tab after the new line. Comments cannot be continued.

Editing Procedure

The following steps will assist you in editing the DMCONFIG file.

Note: Because BEA Connect OSI TP may be installed on a variety of platforms, the procedures in this section make only general references to command entries. Many steps show UNIX command examples. Be sure to use the proper syntax for your platform when making command-line entries.

1. Enter the platform command to gain access to the `install` directory, for example:

```
cd install
```
2. Gain access to the `examples` subdirectory, for example:

```
cd examples
```
3. Enter the command to invoke the text editor and gain access to the DMCONFIG file, for example:

```
edit DMCONFIG
```
4. Edit the DMCONFIG file as necessary. Refer to the parameter descriptions in this section for details about defining your BEA Connect OSI TP configuration.
5. When editing is complete, save and exit the DMCONFIG file. For example, type the following command.

```
$exit
```
6. Load the file with the `dmloadcf` utility. This parses and loads a binary BDMCONFIG configuration file (refer to `dmloadcf` reference page in Appendix B, "Reference Pages").

DMCONFIG Parameters

The following paragraphs describe the significant parameters within specific sections of the DMCNFIG file that define new gateway configurations.

*DM_LOCAL_DOMAINS Section

This section identifies local domains and their associated gateway groups. The section must have an entry for each gateway group (Local Domain). Each entry specifies the parameters required for the domain gateway processes running in that group.

FORMAT

*DM_LOCAL_DOMAINS entries have the following format.

LDOM required parameters [optional parameters]

where:

LDOM is an *identifier* value used to name each local domain.

LDOM must be unique within a particular configuration. As you will see in the description of the *DM_LOCAL_SERVICES section, *LDOM* is the identifier that connects local services with a particular gateway group.

REQUIRED PARAMETERS

The following parameters are required.

GWGRP = *identifier*

specifies the name of the gateway server group (the name provided in the TUXCONFIG file) representing this local domain. There is a one-to-one relationship between a *DOMAINID* and the name of the gateway server group, that is, each GWGRP must have its own, unique DOMAINID.

TYPE = *identifier*

is used for grouping local domain into classes. TYPE can be set to one of the following values: TDOMAIN or OSITP. The TDOMAIN value indicates that this local domain can only communicate with another TUXEDO System/Domain. The OSITP value indicates that this local domain communicates with another TP Domain via the OSI-TP protocol. Domain types must be defined in the \$TUXDIR/udataobj/DMTYPE file.

DOMAINID = *string*

is used to identify the local domain. DOMAINID must be unique across both local and remote domains. The value of *string* can be a sequence of characters (for example, "BA.CENTRAL01"), or a sequence of hexadecimal digits preceded by "0x" (for example, "0x0002FF98C000B9D6"). DOMAINID must be 32 octets or fewer in length. If the value is a string, it must be 31 characters or fewer.

DMTLOGDEV = *string*

specifies the TUXEDO file system that contains the Domain transaction log (DMTLOG) for this machine. The DMTLOG is stored as a TUXEDO System VTOC table on the device. If this parameter is not specified, the domain gateway group is not allowed to process requests in transaction mode. Local domains running on the same machine can share the same DMTLOGDEV file system, but each local domain must have its own log (a table in the DMTLOGDEV) named as specified by the DMTLOGNAME keyword.

OPTIONAL PARAMETERS

The following optional parameters describe resources and limits used in the operation of domain gateways:

AUDITLOG = *string*

specifies the name of the audit log file for this local domain. The audit log feature is activated from the `dmadmin` command and records all the operations within this local domain. If the audit log feature is active and this parameter is not specified, the file `DMmmddyy.LOG` (where `mm`=month, `dd`=day, and `yy`=year) is created in the directory specified by the `$APPDIR` environment variable or the `APPDIR` keyword of the `*MACHINES` section of the `TUXCONFIG` file.

BLOCKTIME = *numeric*

specifies the maximum wait time allowed for a blocking call. The value sets a multiplier of the `SCANUNIT` parameters specified in the `TUXCONFIG` file. The value `SCANUNIT * BLOCKTIME` must be greater than or equal to `SCANUNIT` and less than 32,768 seconds. If this parameter is not specified, the default value is set to the value of the `BLOCKTIME` parameter specified in the `TUXCONFIG` file. A timeout always implies a failure of the affected request. Notice that the timeout specified for transactions in the `TUXCONFIG` will always be used when the request is issued within a transaction.

DMTLOGNAME = *identifier*

specifies the name of the domain transaction log for this domain. This name must be unique when the same DMTLOGDEV is used for several local domains. If not specified, the default is the string ‘DMTLOG’. The name must be 30 characters or less.

DMTLOGSIZE = *numeric*

specifies the numeric size, in pages, of the Domain transaction log for this machine. It must be greater than 0 and less than the amount of available space on the TUXEDO file system. If not specified, the default is 100 pages.

MAXDATALEN = *numeric*

specifies a maximum amount of data (in bytes) that can be sent to or from any services advertised by this local domain. There is no limit if this parameter is not specified.

MAXRDOM = *numeric*

specifies the maximum number of connections (or dialogues if the domain is of type *OSITP*) allowed per gateway. There is no limit if this parameter is not specified.

MAXRDTRAN = *numeric*

specifies the maximum number of domains that can be involved in a transaction. It must be greater than 0 and less than 32,768. If not specified, the default is 16.

MAXTRAN = *numeric*

specifies the maximum number of simultaneous global transactions allowed on this local domain. It must be greater than or equal to 0 and less than or equal to the MAXGTT parameter specified in the TUXCONFIG file. If not specified, the default is the value of MAXGTT.

MAXSENDLEN = *numeric*

specifies the maximum length (in bytes) of messages sent or received by this local domain. If this parameter is set all messages sent or received will be broken up into packets of no more than MAXSENDLEN bytes. There is no limit if this parameter is not specified.

***DM_REMOTE_DOMAINS Section**

This section identifies the known set of remote domains and their characteristics.

FORMAT

*DM_REMOTE_DOMAINS entries have the following format.

RDOM required parameters

where:

RDOM is an *identifier* value used to identify each remote domain known to this configuration.

RDOM must be unique within the configuration.

REQUIRED PARAMETERS

The following parameters are required.

TYPE = *identifier*

is used for grouping remote domain into classes. TYPE can be set to one of the following values: TDOMAIN or OSITP. The TDOMAIN value indicates that this remote domain can only communicate with another TUXEDO System/Domain Domain. The OSITP value indicates that this remote domain communicates with another TP domain via the OSI-TP protocol.

DOMAINID = *string*

is used to identify a remote domain. DOMAINID must be 32 octets or fewer in length. If the value is a string, it must be 31 characters or fewer.

DOMAINID must be unique across remote domains. The value of *string* can be a sequence of characters or a sequence of hexadecimal digits preceded by "0x".

OPTIONAL PARAMETERS

There are no optional parameters for this section.

*DM_TDOMAIN Section

This section defines the addressing information required by domains of type TDOMAIN. This section should have an entry per local domain if requests from remote domains to local services are accepted on that local domain (gateway group), and an entry per remote domain accessible by the defined local domains.

FORMAT

*DM_TDOMAIN entries have the following format.

DOM required parameters [optional parameters]

where:

DOM is an *identifier* value used to identify either a local domain (LDOM) or a remote domain (RDOM) in the *DM_LOCAL_DOMAINS section or in the *DM_REMOTE_DOMAINS section.

The *DOM* identifier must match a previously defined *LDOM* in the *DM_LOCAL_DOMAINS sections or *RDOM* in the *DM_REMOTE_DOMAINS section.

REQUIRED PARAMETERS

The following parameter is required.

NWADDR = *string*

This parameter specifies the network address used by a local or a remote domain to accept connections from other TUXEDO System/Domain Domains. If *string* has the form “0xhex-digits”, it must contain an even number of valid hexadecimal digits.

OPTIONAL PARAMETERS

The following parameter is optional:

NWDEVICE = *string*

Specifies the device file name to be used when binding to the listening address of a local or a remote domain. If the networking functionality is TLI-based, the device name must be an absolute path name. If the networking functionality is Sockets-based, this parameter does not need to be specified.

Notice that multiple entries for a particular domain may be defined in this table. Currently, only the first address is used and the remaining entries are ignored.

*DM_OSITP Section

This section defines the addressing information required by domains of type OSITP. This section should have one entry per gateway group (local domain), and one entry per remote domain of type OSITP.

FORMAT

*DM_OSITP entries have the following format.

DOM required parameters [optional parameters]

where:

DOM is an *identifier* value used to identify a local domain (LDM) or a remote domain (RDM) in the *DM_LOCAL_DOMAINS section or in the *DM_REMOTE_DOMAINS section.

The *DOM* identifier must match a previously defined *LDM* in the *DM_LOCAL_DOMAINS sections or *RDM* in the *DM_REMOTE_DOMAINS section.

REQUIRED PARAMETERS

The following are required parameters.

APT = *string*

This parameter specifies an OSI Application Process Title (APT). An APT may be a name (i.e., the Directory Name of an Application Process Title) or an object identifier (i.e., a sequence of integer values separated by periods).

AEQ = *string*

This parameter specifies an OSI Application Entity Qualifier (AEQ). An AEQ is an integer.

NWDEVICE = *string*

Specifies the device name to be used when binding to an XAP dialogue instance. It may be an absolute path name of a device file name or just an identifier of a device. The value of the string is specific to the XAP-TP provider.

XATMI_ENCODING = { *CAE* | *PRELIMINARY* | *OLTP_TM2200* }

This parameter specifies the version of the XATMI protocol used to communicate with a remote application. Valid values are:

CAE

PRELIMINARY

OLTP_TM2200

OPTIONAL PARAMETERS

The following are optional parameters.

AET = *string*

This parameter specifies an OSI Application Entity Title (AET). An AET is formed from an Application Process Title (APT) and an Application Entity Qualifier (AEQ), i.e. in ASN.1 AET is defined as a SEQUENCE {APT, AEQ} where APT and AET are of type ANY. Three main formats are accepted for the value of *string*:

{object identifier}, {integer}

The first element represents the APT defined as an object identifier (i.e., a sequence of integer values separated by periods) and the second element represents an AEQ defined as an integer constant, e.g., AET = "{1.3.15.0.3},{1}".

ACN = {*XATMI* / *UDT*}

This parameter specifies the object identifier of the Application Context Name (ACN) used by this domain. Current allowed application contexts are: the XATMI-ASE (XATMI) and the UDT-ASE (UDT). If this parameter is not specified, the ACN is set to the object identifier of the XATMI-ASE Application Context.

APID = *integer*

This parameter specifies an OSI Application Process Invocation Identifier (APID).

AEID = *integer*

This parameter specifies an OSI Application Entity Invocation Identifier (AEID).

PROFILE = *identifier*

This parameter specifies the OSI TP profile used by this domain and is used to determine the required OSI TP functional units. PROFILE can be set to one

of the following values: ATP11, ATP21, ATP31, ATP12, ATP22, and ATP32. The UDT ASE application context allows the use of any of these profiles. The XATMI-ASE application context only allows profiles ATP11, ATP21 and ATP31. Profiles ATP11, ATP21 and ATP31 use the Dialogue, Polarized Control and Handshake functional units. Profiles ATP12, ATP22 and ATP32 use the Dialogue, Shared Control, and Handshake functional units. Profiles ATP11 and ATP12 do not use OSI TP transactions (the Commit functional unit is not used). Profiles ATP21 and ATP22 require the Commit, Unchained Transactions, and Recovery functional units. Profiles ATP31 and ATP32 require the Commit, Chained Transactions, and Recovery functional units. By default, the ATP21 profile is always selected.

URCH = *string*

This parameter specifies the user portion of the OSITP Recovery Context Handle. It may be required by the XAP-TP provider in order to perform recovery of distributed transactions after a communications line or system failure.

MAX_LISTENING_EP = *integer*

This parameter specifies the number of endpoints currently waiting for incoming connections. The default is three (3).

*DM_ACCESS_CONTROL Section

This section specifies the access control lists used by local domain.

FORMAT

*DM_ACCESS_CONTROL entries have the following format.

ACL_NAME required parameters

where:

ACL_NAME is a (*identifier*) name used to identify a particular access control list; it must be 15 characters or less in length.

REQUIRED PARAMETERS

The following parameter is required.

ACLIST = *identifier* [*,identifier*]

where an ACLIST is composed of one or more remote domain names (RDOM) separated by commas. The wildcard character (*) can be used to specify that all the remote domains defined in the *DM_REMOTE_DOMAINS section can access a local domain.

*DM_LOCAL_SERVICES Section

This section provides information on the services exported by each local domain. This section is optional and if it is not specified then all local domains defined in the *DM_LOCAL_DOMAINS section accept requests to all of the services advertised by the TUXEDO System/Domain application. If this section is defined then it should be used to restrict the set of local services that can be requested from a remote domain.

FORMAT

*DM_LOCAL_SERVICES entries have the following format.

service [optional parameters]

where:

service is the (*identifier*) local name of the exported service, and it must be 15 characters or fewer in length.

This name corresponds to a name advertised by one or more servers running with the local TUXEDO System/Domain application. Notice that exported services inherit the default or special properties specified for the service in an entry in the SERVICES section of the TUXCONFIG file. Some of these parameters are: LOAD, PRIO, AUTOTRAN, ROUTING, BUFTYPE, and TRANTIME.

REQUIRED PARAMETERS

There are no required parameters for *DM_LOCAL_SERVICES.

OPTIONAL PARAMETERS

The following parameters are optional.

ACL = identifier

specifies the name of the access control list (ACL) to be used by the local domain to restrict requests made to this service by remote domains. The name of the ACL is defined in the *DM_ACCESS_CONTROL section. If this parameter is not specified then access control will not be performed for requests to this service.

LDOM = identifier

specifies the name identifying the local domain exporting this service. If this keyword is not specified then all the local domains defined in the *DM_LOCAL_DOMAINS section will accept requests to this local service.

RNAME = string

specifies the name exported to remote domains. This name will be used by the remote domains for request to this service. If this parameter is not specified, the local service name is supposed to be the name used by any remote domain. For OSI TP, this maps to the TPSUT on the remote system.

*DM_REMOTE_SERVICES Section

This section provides information on services “imported” and available on remote domains.

FORMAT

*DM_REMOTE_SERVICES entries have the following format.

service [optional parameters]

where:

service is the (*identifier*) name used by the local TUXEDO System/Domain application for a particular remote service.

Remote services are associated with a particular remote domain.

REQUIRED PARAMETERS

There are no required parameters for the *DM_REMOTE_SERVICES section.

OPTIONAL PARAMETERS

The following parameters are optional.

CONV = { Y | N }

specifies whether or not the remote service is a conversational service. Use Y to specify the remote service is a conversational service. Use N to specify the remote service is not a conversational service. The default value is N.

LDOM = *identifier*

specifies the name of a local domain in charge of routing requests to this remote service. The gateway group associated with the local domain advertises *service* in the TUXEDO System/Domain Bulletin Board. If this parameter is not specified then all the local domains will be able to accept requests to this remote service. The service request will be then redirected to a remote domain of the same type (see the following definition for RDOM keyword).

RDOM = *identifier*

specifies the name of the remote domain responsible for the actual execution of this service. If this parameter is not specified and a routing criteria (see the following definition for ROUTING keyword) is not specified, then the local domain assumes that any remote domain of the same type accepts this service and it selects a known domain (a domain to which a connection already exists) or remote domain from the **DM_REMOTE_DOMAINS section.

RNAME = *string*

specifies the actual service name expected by the remote domain. If this parameter is not specified, the remote service name is the same as the name specified in *service*. For OSI TP, this maps to the TPSUT on the remote system.

ROUTING = *identifier*

when more than one remote domain offers the same service, a local domain can perform data-dependent routing if this optional parameter is specified. The *identifier* specifies the name of the routing criteria used for this data-dependent routing. If not specified, data-dependent routing is not done for this service. *identifier* must be 15 characters or less in length. If multiple entries exist for the same service name but with different RDOM parameters, the ROUTING parameter should be the same for all of these entries.

TRANTIME = *integer*

specifies the default time-out value in seconds for a transaction automatically started for the associated service. The value must be greater than or equal to 0 and less than 2147483648. The default is 30 seconds. A value of 0 implies the maximum time-out value for the machine.

*DM_ROUTING Section

This section provides information for data-dependent routing of service requests using FML, VIEW, X_C_TYPE, and X_COMMON typed buffers.

FORMAT

*DM_ROUTING entries have the following format.

CRITERION_NAME required parameters

where:

CRITERION_NAME is the (*identifier*) name of the routing entry that was specified on the services entry.

CRITERION_NAME must be 15 characters or less in length.

REQUIRED PARAMETERS

The following parameters are required.

FIELD = *identifier*

specifies the name of the routing field. It must be 30 characters or less. This field is assumed to be a field name that is identified in an FML field table (for FML buffers) or an FML view table (for VIEW, X_C_TYPE, or X_COMMON buffers). The FLDTBLDIR and FIELDTBLS environment variables are used to locate FML field tables, and the VIEWDIR and VIEWFILES environment variables are used to locate FML view tables.

RANGES = *string*

specifies the ranges and associated remote domain names (RDOM) for the routing field. *string* must be enclosed in double quotes. The format of *string* is a comma-separated ordered list of range/RDOM pairs (see the following “Sample Configuration Files” section).

A range is either a single value (signed numeric value or character string in single quotes), or a range of the form “lower - upper” (where lower and upper are both signed numeric values or character strings in single quotes). Note that “lower” must be less than or equal to “upper.” To embed a single quote in a character string value (as in O’Brien, for example), it must be preceded by two backslashes (‘O\\’Brien’).

The value MIN can be used to indicate the minimum value for the data type of the associated FIELD; for strings and arrays, it is the null string; for character fields, it is 0; for numeric values, it is the minimum numeric value that can be stored in the field.

The value MAX can be used to indicate the maximum value for the data type of the associated FIELD; for strings and arrays, it is effectively an unlimited string of octal-255 characters; for a character field, it is a single octal-255 character; for numeric values, it is the maximum numeric value that can be stored in the field.

Thus, “MIN - -5” is all numbers less than or equal to -5 and “6 - MAX” is all numbers greater than or equal to 6. The meta-character “*” (wildcard) in the position of a range indicates any values not covered by the other ranges previously seen in the entry; only one wildcard range is allowed per entry and it should be last (ranges following it will be ignored).

The routing field can be of any data type supported in FML. A numeric routing field must have numeric range values and a string routing field must have string range values.

String range values for string, array, and character field types must be placed inside a pair of single quotes and cannot be preceded by a sign. Short and long integer values are a string of digits, optionally preceded by a plus or minus sign. Floating point numbers are of the form accepted by the C compiler or atof () : an optional sign, then a string of digits optionally containing a decimal point, then an optional e or E followed by an optional sign or space, followed by an integer.

When a field value matches a range, the associated RDOM value specifies the remote domain to which the request should be routed. A RDOM value of “*” indicates that the request can go to any remote domain known by the gateway group.

Within a range/RDOM pair, the range is separated from the RDOM by a “:”.

BUFTYPE = *~type1[:subtype1[,subtype2 . . .]][;type2[:subtype3[, . . .]]] . . . ~*
is a list of types and subtypes of data buffers for which this routing entry is valid. The types are restricted to be either FML, VIEW, X_C_TYPE, or X_COMMON. No subtype can be specified for type FML and subtypes are required for the other types (“*” is not allowed). Duplicate type/subtype pairs cannot be specified for the same routing criterion name; more than one routing entry can have the same criterion name as long as the type/subtype pairs are unique. This parameter is required. If multiple buffer types are specified for a single routing entry, the data types of the routing field for each buffer type must be the same.

If the field value is not set (for FML buffers), or does not match any specific range and a wildcard range has not been specified, an error is returned to the application process that requested the execution of the remote service.

OPTIONAL PARAMETERS

There are no optional parameters for the *DM_ROUTING section.

Sample Configuration Files

The following configuration file defines a 5-site domain configuration. The example shows 4 Bank Branch domains communicating with a Central Bank Branch. Three of the Bank Branches run within other TUXEDO System/Domain domains. The fourth Branch runs under the control of another TP Domain and OSI TP is used in the communication with that domain.

The example shows the TUXEDO System/Domain Domain configuration file from the Central Bank point of view.

Listing 3-1 Sample Configuration File at Central Bank

```
# TUXEDO DOMAIN CONFIGURATION FILE FOR THE CENTRAL BANK
#
#
*DM_LOCAL_DOMAINS
# <local domain name> <Gateway Group name> <domain type> <domain
id> <log device>
#                               [<audit log>] [<blocktime>]
```

```
#          [<log name>] [<log offset>] [<log size>]
#          [<maxrdom>] [<maxrdtran>] [<maxtran>]
#          [<maxdatalen>] [<security>]
#          [<tuxconfig>] [<tuxoffset>]
#
#
DEFAULT: SECURITY = NONE

c01      GWGRP = bankg1
        TYPE = TDOMAIN
        DOMAINID = "BA.CENTRAL01"
        DMTLOGDEV = "/usr/apps/bank/DMTLOG"
        DMTLOGNAME = "DMTLG_C01"

c02      GWGRP = bankg2
        TYPE = OSITP
        DOMAINID = "BA.CENTRAL01"
        DMTLOGDEV = "/usr/apps/bank/DMTLOG"
        DMTLOGNAME = "DMTLG_C02"
        NWDEVICE = "OSITP"
        URCH = "ABCD"

#
*DM_REMOTE_DOMAINS
#<remote domain name>  <domain type> <domain id>
#
b01      TYPE = TDOMAIN
        DOMAINID = "BA.BANK01"

b02      TYPE = TDOMAIN
        DOMAINID = "BA.BANK02"

b03      TYPE = TDOMAIN
        DOMAINID = "BA.BANK03"

b04      TYPE = OSITP
        DOMAINID = "BA.BANK04"
        NWDEVICE = "/dev/osi"
        URCH = "ABCD"

*DM_TDOMAIN
#
# <local or remote domain name>  <network address> [<nwdevice>]
#
# Local network addresses
c01      NWADDR = "0x0002ff98c00b9d6d"  NWDEVICE = "/dev/tcp"
c01      NWADDR = "newyork01.65432"      NWDEVICE = "/dev/starlan"
# Remote network addresses
b01      NWADDR = "0x00020401c00b6d05" NWDEVICE = "/dev/tcp"
b02      NWADDR = "dallas.65432"  NWDEVICE = "/dev/starlan"
b03      NWADDR = "0x00021094c00b6d9c" NWDEVICE = "/dev/tcp"
```

```
*DM_OSITP
#
#<local or remote domain name> <apt> <aeq>
#                                     [<aet>] [<acn>] [<apid>] [<aeid>]
#                                     [<profile>]
#
c02      APT = "BA.CENTRAL01"
         AEQ = "TUXEDO.R.4.2.1"
         AET = "{1.3.15.0.3},{1}"
         ACN = "XATMI"

b04      APT = "BA.BANK04"
         AEQ = "TUXEDO.R.4.2.1"
         AET = "{1.3.15.0.4},{1}"
         ACN = "XATMI"

*DM_LOCAL_SERVICES
#<service_name> [<Local Domain name>] [<access control>]
#<exported svcname>]
#
#
open_act  ACL = branch
close_act ACL = branch
credit
debit
balance
loan      LDOM = c02ACL = loans

*DM_REMOTE_SERVICES
#<service_name>      [<Remote domain name>] [<local domain name>]
#                   [<remote svcname>] [<routing>] [<conv>] [<trantime>]
#
#
tlr_add   LDOM = c01  ROUTING = ACCOUNT
tlr_bal   LDOM = c01  ROUTING = ACCOUNT
tlr_add   RDOM = b04  LDOM = c02 RNAME = "TPSU002"
tlr_bal   RDOM = b04  LDOM = c02 RNAME = "TPSU003"

*DM_ROUTING
# <routing criteria><field> <typed buffer> <ranges>
#
ACCOUNTFIELD = branchid BUFTYPE = "VIEW:account"
              RANGES = "MIN - 1000:b01, 1001-3000:b02, *:b03"

*DM_ACCESS_CONTROL
#<acl name>      <Remote domain list>
#
branch ACLIST = b01, b02, b03
loans  ACLIST = b04
```

This example shows the TUXEDO System/Domain Domain Configuration file required at one of the Bank Branches (BANK01).

Listing 3-2 Sample Configuration File at Branch Bank

```
#TUXEDO DOMAIN CONFIGURATION FILE FOR A BANK BRANCH
#
#
*DM_LOCAL_DOMAINS
#
b01GWGRP = auth
TYPE = TDOMAIN
DOMAINID = "BA.BANK01"
DMTLOGDEV = "/usr/apps/bank/DMTLOG"

*DM_REMOTE_DOMAINS
#
c01  TYPE = TDOMAIN
      DOMAINID = "BA.CENTRAL01"

*DM_TDOMAIN
#
b01  NWADDR = "0x00021094c00b689c"  NWDEVICE = "/dev/tcp"
c01  NWADDR = "0x0002ff98c00b9d6d"  NWDEVICE = "/dev/tcp"
*DM_LOCAL_SERVICES
#
tlr_add  ACL = central
tlr_bal  ACL = central

*DM_REMOTE_SERVICES
#
OPA001  RNAME = "open_act"
CLA001  RNAME = "close_act"
CRD001  RNAME = "credit"
DBT001  RNAME = "debit"
BAL001  RNAME = "balance"

*DM_ACCESS_CONTROL
#
central  ACLIST = c01
```

Modifying an Existing Gateway Configuration

To modify an existing gateway configuration, complete the following steps.

1. Use the `dmadmin` command to access the binary `BDMCONFIG` file.
2. Process the `DMCONFIG` file by running the `dmloadcf` utility.

Using the `dmadmin` Command

You can use the `dmadmin` command or the BEA TUXEDO graphical user interface to administer Connect OSI TP.

You can administer domain gateway groups defined for a BEA TUXEDO System/T application using `dmadmin`, an interactive command interpreter. The `dmadmin` command can operate in two modes: administration mode and configuration mode. Configuration mode allows the administrator to change a configuration while the application is running. Refer to the *BEA TUXEDO/T Domain Manual* for information about the `dmadmin` command.

Defining Connect OSI TP Servers to BEA TUXEDO

To establish a gateway configuration, the BEA TUXEDO system must recognize the Connect OSI TP administrative and gateway servers. To define the Connect OSI TP servers for BEA TUXEDO, edit the `UBBCONFIG` file to define the Connect OSI TP administrative and gateway servers to the BEA TUXEDO system.

Sample UBBCONFIG File

The following file is a sample UBBCONFIG file that defines Connect OSI TP administrative and gateway servers to the BEA TUXEDO system.

Listing 3-3 Sample UBBCONFIG File for Connect OSI TP

```
### Sample ubbconfig file showing Connect OSI TP modifications

*RESOURCES
    # No OSI TP entries in the RESOURCES section.
*MACHINES
    # No OSI TP entries in the MACHINES section.
*GROUPS

### A group must exist for each OSI TP Gateway.
#
OSIGRP                # Name the group as you wish
    GRPNO=1           # Choose a free group number
    LMID=MACHINE      # Specify the machine hosting the gateway.
    TMSNAME="TMS"     # Two or more TMS servers are required.
    TMSCOUNT=2

*SERVERS

### One DMADM server must be configured.
#
DMADM
    SRVID=1           # Choose a unique server id.
    SRVGRP=OSIGRP     # Place in any convenient group.

### One GWADM server must be configured for each gateway.
#

GWADM
    SRVID=2           # Choose a unique server id.
    SRVGRP=OSIGRP     # Place into the group created for this gateway.

### One GWOSITP server must be configured for each gateway.
```

3 *CONFIGURING BEA CONNECT OSI TP*

```
GWOSITP
    SRVID=3          # Choose a unique server id.
    SRVGRP=OSIGRP# Place into the group created for this gateway.
```

```
*SERVICES
```

```
    # No OSI TP entries in the SERVICES section.
```

Refer to the *BEA TUXEDO Reference Manual*.

4 Security

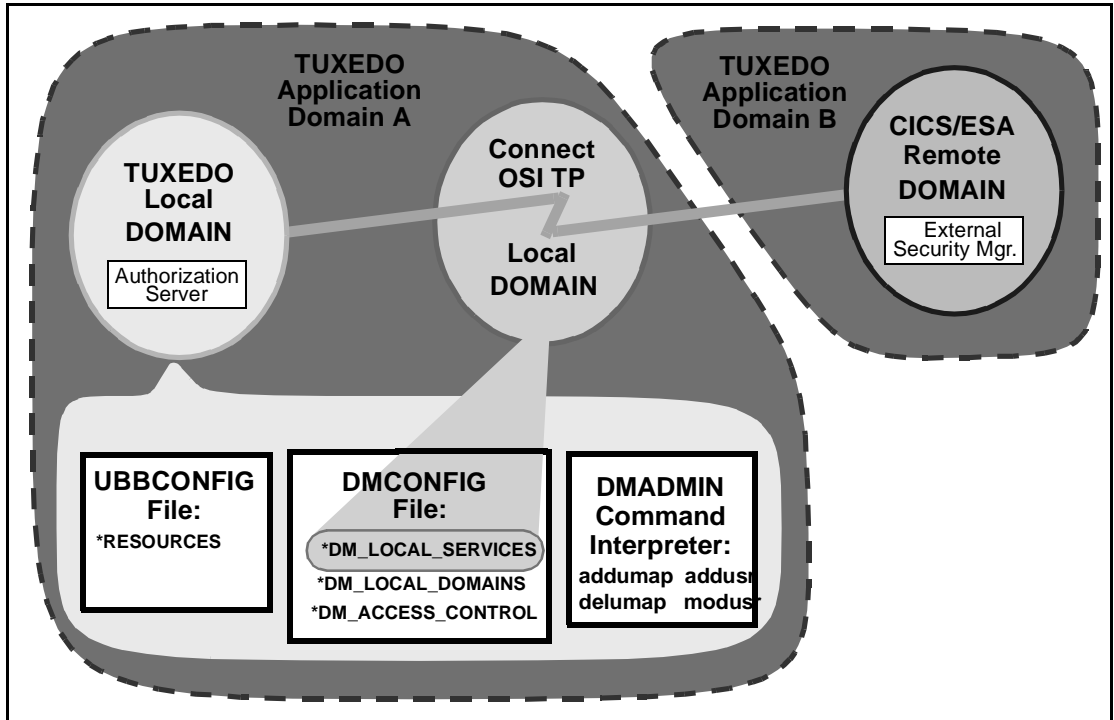
In order for any security checking to occur, each domain must have a security mechanism in place. For the TUXEDO domain, this is the Authorization Server. Figure 4-1 shows these elements.

Note: A domain without an operational security mechanism in place accepts all transaction requests by treating userids as "trusted users."

There are four sections in two BEA TUXEDO configuration files in which you specify parameters bearing on security. The two configuration files are DMCONFIG and UBBCONFIG.

Userid and password mapping between domains also bears on security. There are five DMADMIN subcommands which you use to enter userids and passwords, set up mappings, remove mappings, remove userids and passwords, and modify passwords.

Figure 4-1 Connect OSI TP Security Elements



Where You Specify Security Parameters

The configuration sections where security is specified are:

- ◆ *RESOURCES section of the UBBCONFIG file
- ◆ *DM_LOCAL_DOMAINS section of the DMCONFIG file
- ◆ *DM_LOCAL_SERVICES section of the DMCONFIG file
- ◆ *DM_ACCESS_CONTROL section of the DMCONFIG file

UBBCONFIG File Security Parameters

The `*RESOURCES` section in this file contains a `SECURITY` parameter which works in conjunction with the `SECURITY` parameter in the `DMCONFIG` file to establish how Connect OSI TP controls access to the local TUXEDO domain. This parameter takes the form:

```
SECURITY = value
```

where *value* is:

| | |
|---------------|---|
| NONE | No security is enforced (default) |
| APP_PW | Remote domain-initiated <code>dmloadcf</code> and <code>dmunloadcf</code> require password authorization through the local application <code>AUTHSVC</code> security process. |
| USER_AUTH | Same as <code>APP_PW</code> , but additional authorization is required on a per-user basis. |
| ACL | Same as <code>USER_AUTH</code> , but additional access-control checks are done on service names, queue names, and event names. If no <code>ACL</code> exists for a given name, access is granted. |
| MANDATORY_ACL | Same as <code>ACL</code> , but if no <code>ACL</code> exists for a given name, access is denied. |

In most cases, the `UBBCONFIG` file has already been configured and you do not need to establish the `SECURITY` parameter settings, but examining this file enables you to ascertain how Connect OSI TP enforces security.

If this parameter is set to `NONE`, no security is enforced. If set to `APP_PW`, the local TUXEDO domain's Authorization Server prompts for the application password. If set to `USER_AUTH`, `ACL`, or `MANDATORY_ACL`, the qualified security is enforced as specified.

DMCONFIG File Security Parameters

Three sections in the `DMCONFIG` file contain parameters affecting Connect OSI TP control of access to the local TUXEDO domain:

- ◆ `*DM_LOCAL_DOMAINS` section contains a `SECURITY` parameter which specifies the type of security enforced for the TUXEDO local domain.

- ◆ *DM_ACCESS_CONTROL section contains local access control lists used by the TUXEDO domain to associate local resources with remote OSI environments permitted to have access to them.

Caution: Do not delete the DMCONFIG file. Tables of remote users, remote passwords, and remote mappings are stored in this file. If deleted, all security information must be re-entered.

*DM_LOCAL_DOMAINS SECTION

The SECURITY parameter settings in this section work in conjunction with the SECURITY parameter in the *RESOURCES section of the TUXEDO domain's UBBCONFIG file to establish how Connect OSI TP controls access to the TUXEDO local domain. The parameter takes the form:

SECURITY = *value*

where *value* is:

| | |
|------------|-------------------------------------|
| NONE | No security is enforced. |
| APP_PW | No security is enforced. |
| DM_USER_PW | User password security is enforced. |

If this parameter is set to NONE or APP_PW, the Connect OSI TP domain takes no action with regard to security. If this parameter is set to DM_USR_PW, the Connect OSI TP domain enforces security according to the setting in the TUXEDO domain's UBBCONFIG file (refer to "UBBCONFIG File Security Parameters" on page 4-3).

*DM_LOCAL_SERVICES SECTION

The ACL parameter in this section works in conjunction with the ACL_NAME defined in the *DM_ACCESS_CONTROL section to restrict requests made to the local services by remote domains.

How To Administer Security

After setting up and/or checking the security settings for the TUXEDO domain and the OSI domain, you must relate the security information in both domains to each other. To do this, use the `addusr` and `addumap` subcommands provided with the `dmadmin` command interpreter.

Once the user security information in both domains is mapped, you can perform administration on the affected security files in each domain. To do this, use the `delumap`, `modusr`, and `delusr` subcommands.

The following paragraphs discuss how you enter these commands. Refer to the associated reference pages in Appendix B, “Reference Pages” for detailed information about each subcommand.

Adding a Userid and Password

Use the `addusr` subcommand to define a TUXEDO local domain’s user ID and password to the remote domain’s External Security Manager. Enter the following command:

```
addusr -d local_domain_id -R remote_domain_id -u remote_userid
```

where:

- | | |
|-----------------|---|
| <code>-d</code> | adds the name of the local domain. |
| <code>-R</code> | adds the name of the remote domain. |
| <code>-u</code> | adds the remote user name to be added (the system prompts for the user’s password). |

Mapping a Userid

Use the `addumap` subcommand to map a local domain userid to a remote domain userid. The userid must be added before it can be mapped. Enter the following command:

```
addumap -d local_domain_id -R remote_domain_id  
-p local_principal_userid -u remote_userid
```

where:

- d maps the name of the local domain.
- R maps the name of the remote domain.
- p maps the local userid.
- u maps the remote userid.

Removing a Userid's Mapping

Use the `delumap` subcommand to remove the mapping for a local domain userid to a remote domain userid. Enter the following command:

```
delumap -d local_domain_id -R remote_domain_id  
-p local_principal_userid -u remote_userid
```

where:

- d deletes the name of the local domain.
- R deletes the name of the remote domain.
- p deletes the local userid.
- u deletes the remote userid.

Deleting a Userid and Password

Use the `delusr` subcommand to remove a local TUXEDO domain's user ID and password from the remote domain's External Security Manager. The mapping for a userid must be removed before the userid can be removed. Enter the following command:

```
delusr -d local_domain_id -R remote_domain_id -u remote_userid
```

where:

- d deletes the name of the local domain.
- R deletes the name of the remote domain.

`-u` deletes the remote user name to be deleted.

Modifying a Password

Use the `modusr` subcommand to modify a local TUXEDO domain user's password recorded in a remote domain's External Security Manager. Enter the following command:

```
modusr -d local_domain_id -R remote_domain_id -u remote_userid
```

where:

`-d` designates the name of the local domain.

`-R` designates the name of the remote domain.

`-u` designates the remote user name whose password is to be modified (the system prompts for the new password).

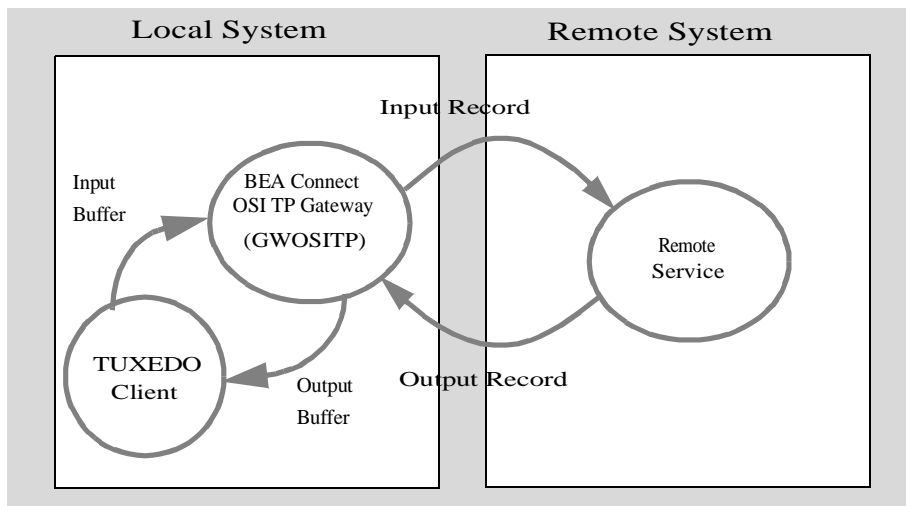
5 Data Translations

BEA Connect OSI TP uses typed buffers to transmit and receive data. BEA Connect OSI TP supports the following buffer types:

- ◆ VIEW
 - ◆ X_C_TYPE
 - ◆ X_COMMON
- ◆ STRING
- ◆ CARRAY
- ◆ X_OCTET

Note: Zero length STRINGS or X_OCTET buffers are *not* supported.

Figure 5-1 Mapping Parameters during Locally Originated Calls



Layout Conversion for Buffer Types

XATMI (X/Open Application Transaction Manager Interface) mappings to OSI TP are defined in the XATMI ASE (Application Service Element). BEA Connect OSI TP supports this combination. Interoperability using Connect OSI TP requires that remote systems support XATMI ASE.

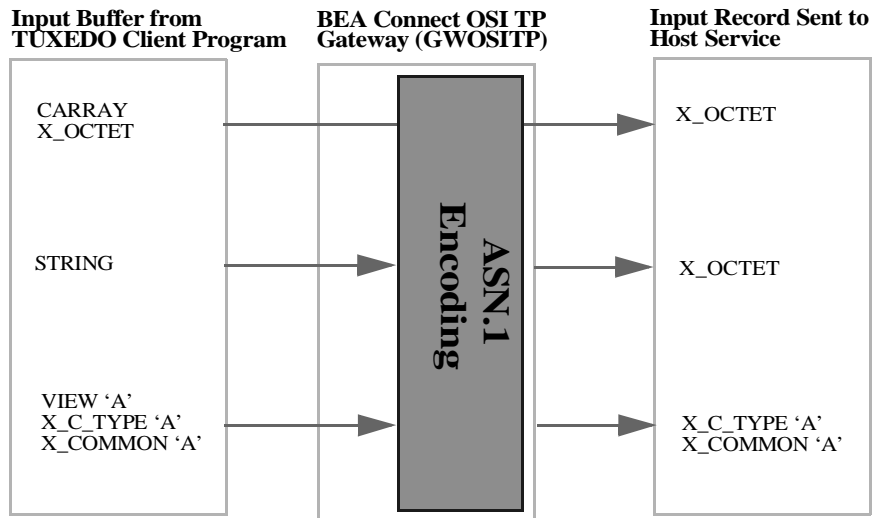
Therefore, TUXEDO-specific buffer types, such as `STRING`, `VIEW`, and `CARRAY` may need to be converted into XATMI standard types. BEA Connect OSI TP Gateways perform these layout conversions implicitly.

ASN.1 Encoding

Abstract Syntax Notation 1 (ASN.1) is an international standard that provides a canonical representation to deal with data representation differences such as byte order, word length, and character sets. The local Gateway (GWOSITP) encodes input from the local client program. It produces an ASN.1 encoded record that is sent to the remote service. When a reply is received, it is decoded before being returned to the client. Similarly, when remote requests for local services are received by the local Gateway, they are decoded from the ASN.1 format. Replies are then encoded for return to the remote client.

Summary of Mappings

The local Gateway (GWOSITP) maps input buffers to input records based on information found in the configuration setting for the local and remote services. The following summarizes all the mapping possibilities:

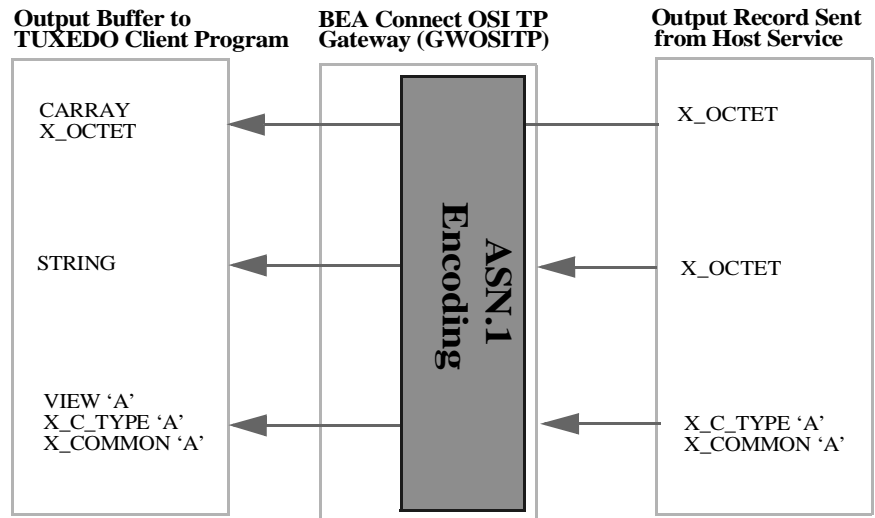
Figure 5-2 Input Mappings

As illustrated, the mapping possibilities include:

- ◆ TUXEDO CARRY input buffer can be copied to X_OCTET input records. A CARRY buffer contains raw data that is ASN.1 encoded.
- ◆ TUXEDO STRING input buffers can be mapped to X_OCTET input records. The buffer goes through ASN.1 encoding.
- ◆ TUXEDO VIEW input buffers can be mapped to X_C_TYPE and X_COMMON. The buffers go through ASN.1 encoding.

While the previous figure illustrates input mapping, output mapping follows exactly the same paths but in reverse.

Figure 5-3 Output Mappings



Application Programming

TUXEDO application programs that send requests through BEA Connect OSI TP are developed like other TUXEDO application programs.

Information on the supported ATMI calls and recommendations for selecting the buffers used in application programs are the major considerations for programmers.

Supported ATMI Calls

The request/response calls supported by BEA Connect OSI TP are the following:

- ◆ tpcall
- ◆ tpacall (including TPNOREPLY)
- ◆ tpgetrply

- ◆ tpcancel
- ◆ tpservice
- ◆ tpreturn
- ◆ tpforward

The conversational calls supported by BEA Connect OSI TP are the following:

- ◆ tpconnect
- ◆ tpsend
- ◆ tprecv
- ◆ tpdiscon
- ◆ tpservice
- ◆ tpreturn

Considerations for Selecting Buffers

Consider the following recommendations when selecting buffers in application programs.

- ◆ Use VIEW for programming action. This couples local and remote record layouts, but it does not couple data representations.
- ◆ Use STRING buffers to represent a single value (field), or variable length (delimited) character fields. Multiple fixed length substrings may be mapped using VIEWS.
- ◆ Avoid CARRAYs, if possible, because CARRAYs most tightly couple local and remote data representations. Acceptable uses include bitmaps and multibyte characters.

System Programming

Because mapping keeps system dependencies out of application code, it may be thought of as a system programming (or an application administration) activity. System programming activity is done once VIEWS have been defined and VIEW files have been compiled.

Defining VIEWS

Work with the application programmers for the host system to determine the input and output record layouts for the remote application programs. Create standard TUXEDO VIEW definitions in files.

Compiling VIEW Files

TUXEDO VIEW definitions in files, compile the VIEW files by running the viewc VIEW compiler.

If necessary, set the VIEWFILES, VIEWDIR, FIELDTBLS, and FLDTBLDIR environment variables using a TUXEDO ENVFILE so that BEA Connect OSI TP servers can locate binary VIEW files and field table files at runtime.

A Error and Informational Messages

BEA Connect OSI TP issue the following error and informational messages.

| | |
|------|---|
| 1000 | "ERROR: Tried to write more than maxlen:len=%ld, maxlen=%ld" |
| 1001 | "ERROR: Can't create file %s in APPDIR for storing OSITP blob" |
| 1002 | "ERROR: Can't write to OSITP blob file %s in APPDIR" |
| 1100 | "ERROR: Unable to retrieve Network context, shutdown Gateway!" |
| 1101 | "ERROR: Unable to retrieve file descriptor, shutdown Gateway!" |
| 1102 | "INFO: Unable to obtain svc info from svcinfo, request rejected!" |
| 1103 | "ERROR: Find network transaction failed, shutdown Gateway!" |
| 1104 | "ERROR: Can not create XAP-TP instance, request rejected!" |
| 1105 | "ERROR: Could not retrieve Network context, shutdown Gateway!" |
| 1106 | "ERROR: Set CNTX_NAME failed, %s" |

| | |
|------|--|
| 1107 | "ERROR: Could not retrieve file descriptor, shutdown Gateway!" |
| 1108 | "WARN: Transaction table full, request rejected!" |
| 1109 | "ERROR: Could not allocate send buffer, request rejected!" |
| 1110 | "ERROR: Unable to set nettxid in shmem" |
| 1111 | "ERROR: Cannot associate with network transaction" |
| 1112 | "ERROR: Set of TTNID failed, %s" |
| 1113 | "ERROR: Send of AP_TP_BEGIN_DIALOGUE_REQ failed, %s" |
| 1114 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1115 | "ERROR: Could not find RDOM %s cache entry" |
| 1116 | "ERROR: Send of %s failed, %s" |
| 1117 | "ERROR: Cannot set remote domain for transaction" |
| 1118 | "INFO: Resumed with BLOCKING timeout" |
| 1119 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1120 | "ERROR: Send of AP_TP_DEFERRED_END_DIALOGUE_REQ failed, %s" |
| 1121 | "ERROR: Bad action state" |
| 1122 | "ERROR: Couldn't retrieve TRAN table from shmem, shutdown Gateway!" |
| 1123 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1124 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1125 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |

| | |
|------|--|
| 1126 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1127 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1128 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1129 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1130 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1131 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1132 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1133 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1134 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1135 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1136 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1137 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1138 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1139 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1140 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 1141 | "ERROR: Internal error, shutdown Gateway!" |

| | |
|------|---|
| 1142 | "WARN: Can't find input message buffer, ACALL event deleted!" |
| 1200 | "ERROR: Malloc failure" |
| 1201 | "ERROR: Malloc failure" |
| 1202 | "ERROR: Malloc failure" |
| 1203 | "ERROR: Malloc failure" |
| 1300 | "ERROR: ioptr_init_char failed" |
| 1301 | "ERROR: ASN.1 ENCODE failed %s" |
| 1302 | "ERROR: ioptr_init_char failed" |
| 1303 | "ERROR: ASN.1 DECODE failed %s" |
| 1304 | "ERROR: tpalloc failed: %s" |
| 1305 | "" |
| 1306 | "ERROR: ioptr_init_char failed" |
| 1307 | "ERROR: ASN.1 ENCODE failed %s" |
| 1308 | "ERROR: ioptr_init_char failed" |
| 1309 | "ERROR: ASN.1 DECODE failed %s" |
| 1310 | "ERROR: tpalloc failed: %s" |
| 1313 | "ERROR: ioptr_init_char failed" |
| 1314 | "ERROR: ASN.1 ENCODE failed %s" |
| 1315 | "ERROR: ioptr_init_char failed" |
| 1316 | "ERROR: ASN.1 DECODE failed %s" |
| 1317 | "ERROR: tpalloc failed: %s" |
| 1319 | "ERROR: ioptr_init_char failed" |
| 1320 | "ERROR: ASN.1 ENCODE failed %s" |
| 1321 | "ERROR: ioptr_init_char failed" |

| | |
|------|--|
| 1322 | "ERROR: ASN.1 DECODE failed %s" |
| 1323 | "ERROR: talloc failed: %s" |
| 1325 | "ERROR: ioptr_init_char failed" |
| 1326 | "ERROR: ASN.1 ENCODE failed %s" |
| 1327 | "ERROR: ioptr_init_char failed" |
| 1328 | "ERROR: ASN.1 DECODE failed %s" |
| 1329 | "ERROR: talloc failed: %s" |
| 1331 | "ERROR: ioptr_init_char failed" |
| 1332 | "ERROR: ASN.1 ENCODE failed %s" |
| 1333 | "ERROR: ioptr_init_char failed" |
| 1334 | "ERROR: ASN.1 DECODE failed %s" |
| 1335 | "ERROR: talloc failed: %s" |
| 1337 | "ERROR: Unable to process typed buffer" |
| 1338 | "ERROR: Unable to process typed buffer" |
| 1339 | "ERROR: Unable to process typed buffer" |
| 1340 | "ERROR: Unable to process typed buffer" |
| 1341 | "ERROR: Unable to process typed buffer" |
| 1342 | "ERROR: Unable to process typed buffer" |
| 1343 | "ERROR: Unable to reconstruct typed buffer" |
| 1344 | "ERROR: Unable to reconstruct typed buffer" |
| 1345 | "ERROR: Unable to reconstruct typed buffer" |
| 1346 | "ERROR: Unable to reconstruct typed buffer" |
| 1347 | "ERROR: Unable to reconstruct typed buffer" |
| 1348 | "ERROR: Unable to reconstruct typed buffer" |
| 1500 | "ERROR: Couldn't retrieve TRAN table from shmem" |

| | |
|------|--|
| 1501 | "ERROR: Unable to get file descriptor for control end point!" |
| 1502 | "ERROR: Unable to find transaction node, shutdown Gateway!" |
| 1503 | "ERROR: Unable to empty send cdata!" |
| 1504 | "ERROR: Send of %s failed, %s" |
| 1505 | "ERROR: Unable to send AP_TP_U_ABORT_REQ, %s" |
| 1506 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1507 | "ERROR: Unrecoverable error on control instance, shutdown Gateway" |
| 1508 | "ERROR: Unable to send AP_TP_U_ABORT_REQ, %s" |
| 1509 | "ERROR: Invalid conversation context" |
| 1510 | "INFO: State check, gtrid(%s)" |
| 1511 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |
| 1512 | "INFO: State check, gtrid(%s)" |
| 1513 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |
| 1514 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 1515 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 1516 | "INFO: State check, gtrid(%s)" |
| 1517 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |
| 1518 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 1519 | "ERROR: Invalid network context descriptor %d" |
| 1520 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1521 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 1522 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 1523 | "ERROR: Find network transaction failed, shutdown Gateway!" |

| | |
|------|---|
| 1524 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 1525 | "INFO: State check, gtrid(%s)" |
| 1526 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |
| 1527 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 1528 | "INFO: State check, gtrid(%s)" |
| 1529 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1530 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 1600 | "ERROR: Invalid action index" |
| 1601 | "ERROR: Unable to obtain remote service information, request rejected!" |
| 1602 | "ERROR: Can not create XAP-TP instance, request rejected!" |
| 1603 | "ERROR: Could not retrieve Network context, shutdown Gateway!" |
| 1604 | "ERROR: Could not retrieve file descriptor, shutdown Gateway!" |
| 1605 | "ERROR: Could not allocate send buffer, request rejected!" |
| 1606 | "ERROR: Find network transaction failed, shutdown Gateway!" |
| 1607 | "ERROR: Transaction table full, request rejected!" |
| 1608 | "ERROR: Could not allocate send buffer, request rejected!" |
| 1609 | "ERROR: Couldn't retrieve TRAN table from shmem" |
| 1610 | "ERROR: Unable to set nettxid in shmem" |
| 1611 | "ERROR: Cannot associate with network transaction" |
| 1612 | "ERROR: Can't set AP_TTNID, ap_errno = %x" |
| 1613 | "ERROR: Send of AP_TP_BEGIN_DIALOGUE_REQ failed, %s" |

| | |
|------|--|
| 1615 | "ERROR: Send of AP_TP_DEFFERRED_END_DIALOGUE_REQ failed, %s" |
| 1616 | "ERROR: Send of AP_TP_DATA_REQ failed, %s" |
| 1617 | "ERROR: Send AP_TP_GRANT_CONTROL_REQ failed, %s" |
| 1618 | "ERROR: Cannot set remote domain for transaction" |
| 1619 | "ERROR: gw_tx_end returned failure" |
| 1621 | "ERROR: Bad action state" |
| 1622 | "ERROR: Invalid action index" |
| 1623 | "ERROR: Invalid incoming conversation context" |
| 1624 | "ERROR: Conversation identifier is NULL" |
| 1625 | "ERROR: Invalid action index" |
| 1626 | "ERROR: Invalid conversation context" |
| 1627 | "ERROR: Conversation identifier is invalid" |
| 1628 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1629 | "ERROR: Send of AP_TP_U_ABORT_REQ failed, %s" |
| 1630 | "ERROR: Could not find RDOM %s cache entry" |
| 1631 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1632 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1633 | "ERROR: gw_tx_end returned failure" |
| 1634 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1635 | "ERROR: gw_tx_end returned failure" |
| 1636 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |

| | |
|------|---|
| 1637 | "ERROR: gw_tx_end returned failure" |
| 1638 | "ERROR: gw_tx_end returned failure" |
| 1639 | "ERROR: Invalid outgoing conversation context" |
| 1640 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1700 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1701 | "ERROR: ap_open failed, %s" |
| 1702 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1703 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1704 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1705 | "WARN: Found transaction without log record" |
| 1706 | "ERROR: Create transaction node failed, shutdown Gateway!" |
| 1707 | "ERROR: ap_bind failed, %s" |
| 1708 | "ERROR: ap_snd failed, %s" |
| 1709 | "ERROR: Unable to restart xap_tp control instance" |
| 1710 | "ERROR: Receive of AP_TP_RESTART_COMPLETE_IND failed, %s" |
| 1711 | "ERROR: Unable to resume control instance" |
| 1712 | "ERROR: Unable to set AP_FLAGS, %s" |
| 1713 | "ERROR: Unable to send AP_TP_RESTART_REQ, %s" |
| 1714 | "WARN: log record refused" |
| 1715 | "ERROR: Unable to reconstruct OSITP TPPM, %s" |

| | |
|------|---|
| 1716 | "ERROR: Unable to send AP_TP_RESTART_COMPLETE_REQ, %s" |
| 1717 | "ERROR: Unable to create action to handle restarted transaction!" |
| 1718 | "ERROR: calloc call failed" |
| 1719 | "ERROR: ap_rcv failed in resume: %s" |
| 1720 | "ERROR: Too many uncompleted transactions from OSITP" |
| 1721 | "INFO: Transaction from OSITP (%s) queued for completion" |
| 1722 | "ERROR: Memory allocation failed in resume" |
| 1723 | "ERROR: No TTNID or DTNID returned in cdata->tp_env with NODE_STATUS_IND" |
| 1724 | "ERROR: Unexpected primitive received in resume: 0x%lx" |
| 1725 | "ERROR: Create transaction node failed, shutdown Gateway!" |
| 1726 | "ERROR: Unable to create action to handle restarted transaction!" |
| 1727 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1728 | "ERROR: Unable to create action to handle restarted transaction!" |
| 1729 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1730 | "WARN: Max number of groups in /T transaction reached, OSI TP transaction orphaned; TTNID: %s" |
| 1731 | "WARN: /T transaction state mismatch, found <%d> - OSI TP transaction orphaned; TTNID: %s" |
| 1732 | "ERROR: /Domain transaction table too small, can't resume transaction, shutdown Gateway!" |
| 1733 | "INFO: Heuristic log for transaction TTNID: %s" |
| 1734 | "ERROR: Resume transaction protocol error, state(0x%lx)!" |

| | |
|------|--|
| 1735 | "ERROR: Unable to create action to handle resumed transaction!" |
| 1737 | "INFO: OSI TP transaction found without corresponding /T transaction, DTNID: %s, state: AP_TP_WCOMMITind" |
| 1738 | "INFO: OSI TP transaction found without corresponding /T transaction, TTNID: %s, state: AP_TP_WCOMMITind" |
| 1739 | "INFO: OSI TP transaction found without corresponding /T transaction, DTNID: %s, state: AP_TP_COMMIT_WDONereq" |
| 1740 | "INFO: OSI TP transaction found without corresponding /T transaction, TTNID: %s, state: AP_TP_WCOMMIT_COMPind" |
| 1741 | "INFO: OSI TP transaction found without corresponding /T transaction, DTNID: %s, state: AP_TP_WCOMMIT_COMPind" |
| 1742 | "INFO: Heuristic log for transaction TTNID: %s" |
| 1743 | "INFO: Heuristic log for transaction DTNID: %s" |
| 1744 | "ERROR: Resume transaction protocol error, state(0x%lx)!" |
| 1745 | "ERROR: Unable to create action to handle resumed transaction!" |
| 1746 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 1747 | "INFO: Transaction from OSITP with DTNID queued for completion" |
| 1748 | "ERROR: Allocate transaction node failed!" |
| 1749 | "ERROR: Allocate transaction node failed!" |
| 1750 | "ERROR: Allocate transaction node failed!" |
| 1751 | "ERROR: Create transaction node failed, shutdown Gateway!" |
| 1752 | "ERROR: Can't find view file, %s" |

| | |
|------|--|
| 1753 | "ERROR: APDU buffer overflow" |
| 1754 | "ERROR: Memory allocation error" |
| 1755 | "ERROR: APDU buffer overflow" |
| 1756 | "ERROR: Memory allocation error" |
| 1757 | "ERROR: APDU buffer overflow" |
| 1758 | "ERROR: Memory allocation error" |
| 1759 | "ERROR: Invalid data type for X_C_TYPE" |
| 1760 | "ERROR: VIEW32 file %s not found" |
| 1761 | "ERROR: Can't find view file %s" |
| 1762 | "ERROR: Missing data element in received Data Stream" |
| 1763 | "ERROR: APDU buffer overflow" |
| 1764 | "ERROR: Missing data element in received Data Stream" |
| 1765 | "ERROR: APDU buffer overflow" |
| 1766 | "ERROR: Missing data element in received Data Stream" |
| 1767 | "ERROR: Invalid data type for %s" |
| 1768 | "INFO: OSI TP transaction found without corresponding /T transaction, TTNID: %s, state: AP_TP_COMMIT_WDONEreq" |
| 1900 | "ERROR: Memory allocation failure" |
| 1901 | "ERROR: Invalid Buffer Type" |
| 1902 | "ERROR: Memory allocation failure" |
| 1903 | "ERROR: Invalid Operation" |
| 1904 | "ERROR: Invalid Buffer Type" |
| 1905 | "ERROR: VIEW file not found" |
| 1906 | "ERROR: Invalid Operation" |

| | |
|------|---------------------------------------|
| 1907 | "ERROR: UDT ENCODE Failed" |
| 1908 | "ERROR: VIEW file %s not found" |
| 1909 | "ERROR: Memory allocation failure" |
| 1910 | "ERROR: Memory allocation failure" |
| 1911 | "ERROR: Memory allocation failure" |
| 1912 | "ERROR: ioptr_init_char failed" |
| 1913 | "ERROR: Encode of U_ASE failed" |
| 1914 | "ERROR: No UDT data present" |
| 1915 | "ERROR: Invalid operation" |
| 1916 | "ERROR: Invalid Buffer Type" |
| 1917 | "ERROR: Memory allocation failure" |
| 1918 | "ERROR: Memory allocation failure" |
| 1919 | "ERROR: Encode failed" |
| 1920 | "ERROR: Can't make object identifier" |
| 1921 | "ERROR: Memory allocation failure" |
| 1922 | "ERROR: Memory allocation failure" |
| 1923 | "ERROR: Memory allocation failure" |
| 1924 | "ERROR: ioptr_init_char failed" |
| 1925 | "ERROR: Encode of U_ASE failed" |
| 1926 | "ERROR: Memory allocation failure" |
| 1927 | "ERROR: Memory allocation failure" |
| 1928 | "ERROR: ioptr_init_char failed" |
| 1929 | "ERROR: Decode of U_ASE failed" |
| 1930 | "ERROR: Can't make object identifier" |
| 1931 | "ERROR: Invalid transfer syntax name" |

| | |
|------|---|
| 1932 | "ERROR: Get of AP_CNTX_NAME failed, %s" |
| 1933 | "ERROR: Can't make object identifier" |
| 1934 | "ERROR: Memory allocation failure" |
| 1935 | "ERROR: Memory allocation failure" |
| 1936 | "ERROR: ioptr_init_char failed" |
| 1937 | "ERROR: Invalid operation, or context" |
| 1938 | "ERROR: XATMI DECODE failed" |
| 1939 | "ERROR: Unable to obtain svc info from svcinfo" |
| 1941 | "ERROR: UDT DECODE failed" |
| 1942 | "ERROR: UDT post recieve function failed" |
| 1945 | "ERROR: Service is denied Access" |
| 1947 | "ERROR: Service is denied Access" |
| 1948 | "ERROR: Invalid operation" |
| 1949 | "ERROR: Undefined buffer type" |
| 1950 | "ERROR: Not a typed buffer" |
| 1951 | "ERROR: Memory allocation error" |
| 1952 | "ERROR: Not a typed buffer" |
| 1953 | "ERROR: Can't find view file" |
| 1954 | "ERROR: Memory allocation error" |
| 1955 | "ERROR: APDU buffer overflow" |
| 1956 | "ERROR: Memory allocation error" |
| 1957 | "ERROR: APDU buffer overflow" |
| 1958 | "ERROR: Memory allocation error" |
| 1959 | "ERROR: Buffer size error" |
| 1960 | "ERROR: Memory allocation error" |

| | |
|------|---|
| 1961 | "ERROR: Memory allocation error" |
| 1962 | "ERROR: Invalid data type for X_COMMON" |
| 1963 | "ERROR: Not a typed buffer" |
| 1964 | "ERROR: Can't find view file" |
| 1965 | "ERROR: Memory allocation error" |
| 1966 | "ERROR: APDU buffer overflow" |
| 1967 | "ERROR: Memory allocation error" |
| 1968 | "ERROR: APDU buffer overflow" |
| 1969 | "ERROR: Memory allocation error" |
| 1970 | "ERROR: Buffer size error" |
| 1971 | "ERROR: Memory allocation error" |
| 1972 | "ERROR: Memory allocation error" |
| 1973 | "ERROR: Invalid data type for X_C_TYPE" |
| 1974 | "ERROR: Undefined buffer type" |
| 1975 | "ERROR: Not a typed buffer" |
| 1976 | "ERROR: Not a typed buffer" |
| 1977 | "ERROR: Can't find view file" |
| 1978 | "ERROR: APDU buffer overflow" |
| 1979 | "ERROR: APDU buffer overflow" |
| 1980 | "ERROR: Invalid data type for X_C_TYPE" |
| 1981 | "ERROR: Not a typed buffer" |
| 1982 | "ERROR: Can't find view file" |
| 1983 | "ERROR: APDU buffer overflow" |
| 1984 | "ERROR: APDU buffer overflow" |
| 1985 | "ERROR: APDU buffer overflow" |

| | |
|------|--|
| 1986 | "ERROR: Invalid data type for X_COMMON" |
| 1987 | "ERROR: Invalid Buffer Type" |
| 1988 | "ERROR: Can't get local services from configuration file" |
| 1990 | "ERROR: Incoming request not allowed access to any local services" |
| 1991 | "ERROR: Invalid buffer type specified" |
| 1992 | "ERROR: Unable to obtain svc info from svcinfo" |
| 1993 | "ERROR: Encode of U_ABORT_RI failed" |
| 1994 | "ERROR: Decode of U_ABORT_RI failed" |
| 1995 | "ERROR: Can't make object identifier" |
| 1996 | "ERROR: Invalid transfer syntax name" |
| 1997 | "ERROR: Can't find LSVC entry by rname %s" |
| 1998 | "ERROR: Can't find RSVC entry by name %s" |
| 1999 | "ERROR: Invalid Buffer Type" |
| 2000 | "ERROR: Memory allocation error" |
| 2001 | "ERROR: Memory allocation error" |
| 2002 | "ERROR: View file must be specified" |
| 2003 | "ERROR: Invalid Buffer Type" |
| 2004 | "ERROR: VIEW32 file is required" |
| 2005 | "ERROR: Missing data element in received Data Stream" |
| 2006 | "ERROR: Missing data element in received Data Stream" |
| 2007 | "ERROR: Missing data element in received Data Stream" |
| 2008 | "ERROR: Missing data element in received Data Stream" |
| 2009 | "ERROR: Missing data element in received Data Stream" |
| 2010 | "ERROR: Missing data element in received Data Stream" |

| | |
|------|--|
| 2011 | "ERROR: Request block is invalid" |
| 2012 | "ERROR: Memory allocation error" |
| 2013 | "ERROR: Request block is invalid" |
| 2014 | "ERROR: Memory allocation error" |
| 2015 | "ERROR: Request block is invalid" |
| 2016 | "ERROR: Memory allocation error" |
| 2017 | "ERROR: Request block is invalid" |
| 2018 | "ERROR: Memory allocation error" |
| 2019 | "ERROR: Request block is invalid" |
| 2020 | "ERROR: Memory allocation error" |
| 2021 | "ERROR: Memory allocation error" |
| 2022 | "ERROR: Memory allocation error" |
| 2023 | "ERROR: Request block is invalid" |
| 2024 | "ERROR: Received type does not match definition" |
| 2025 | "ERROR: ASN.1 DECODE failed, out of short integer range" |
| 2026 | "ERROR: Received type does not match definition" |
| 2027 | "ERROR: ASN.1 DECODE failed, out of short integer range" |
| 2028 | "ERROR: Request block is invalid" |
| 2029 | "ERROR: Received type does not match definition" |
| 2030 | "ERROR: Received type does not match definition" |
| 2031 | "ERROR: Request block is invalid" |
| 2032 | "ERROR: Received type does not match definition" |
| 2033 | "ERROR: Received type does not match definition" |
| 2034 | "ERROR: Request block is invalid" |
| 2035 | "ERROR: Received type does not match definition" |

| | |
|------|--|
| 2036 | "ERROR: Received type does not match definition" |
| 2037 | "ERROR: Request block is invalid" |
| 2038 | "ERROR: Received type does not match definition" |
| 2039 | "ERROR: Received type does not match definition" |
| 2040 | "ERROR: Request block is invalid" |
| 2041 | "ERROR: Memory allocation error" |
| 2042 | "ERROR: Request block is invalid" |
| 2043 | "ERROR: Memory allocation error" |
| 2044 | "ERROR: Request block is invalid" |
| 2045 | "ERROR: Memory allocation error" |
| 2046 | "ERROR: Request block is invalid" |
| 2047 | "ERROR: Memory allocation error" |
| 2048 | "ERROR: Request block is invalid" |
| 2049 | "ERROR: Memory allocation error" |
| 2050 | "ERROR: Request block is invalid" |
| 2051 | "ERROR: Memory allocation error" |
| 2052 | "ERROR: Request block is invalid" |
| 2053 | "ERROR: Memory allocation error" |
| 2054 | "ERROR: Memory allocation error" |
| 2055 | "ERROR: Memory allocation error" |
| 2056 | "ERROR: Memory allocation error" |
| 2057 | "ERROR: Memory allocation error" |
| 2058 | "ERROR: Request block is invalid" |
| 2059 | "ERROR: Memory allocation error" |
| 2060 | "ERROR: Memory allocation error" |

| | |
|------|--|
| 2061 | "ERROR: Memory allocation error" |
| 2062 | "ERROR: Request block is invalid" |
| 2063 | "ERROR: Memory allocation error" |
| 2064 | "ERROR: Memory allocation error" |
| 2065 | "ERROR: Request block is invalid" |
| 2066 | "ERROR: Received type does not match definition" |
| 2067 | "ERROR: ASN.1 DECODE failed, out of short integer range" |
| 2068 | "ERROR: Received type does not match definition" |
| 2069 | "ERROR: ASN.1 DECODE failed, out of short integer range" |
| 2070 | "ERROR: Request block is invalid" |
| 2071 | "ERROR: Received type does not match definition" |
| 2072 | "ERROR: ASN.1 DECODE failed, out of integer range" |
| 2073 | "ERROR: Received type does not match definition" |
| 2074 | "ERROR: ASN.1 DECODE failed, out of integer range" |
| 2075 | "ERROR: Request block is invalid" |
| 2076 | "ERROR: Received type does not match definition" |
| 2077 | "ERROR: Received type does not match definition" |
| 2078 | "ERROR: Request block is invalid" |
| 2079 | "ERROR: Received type does not match definition" |
| 2080 | "ERROR: Received type does not match definition" |
| 2081 | "ERROR: Request block is invalid" |
| 2082 | "ERROR: Received type does not match definition" |
| 2083 | "ERROR: ASN.1 DECODE failed, float range out of local limit" |
| 2084 | "ERROR: Received type does not match definition" |
| 2085 | "ERROR: ASN.1 DECODE failed, float range out of local limit" |

| | |
|------|---|
| 2086 | "ERROR: Request block is invalid" |
| 2087 | "ERROR: Received type does not match definition" |
| 2088 | "ERROR: ASN.1 DECODE failed, double range out of local limit" |
| 2089 | "ERROR: Received type does not match definition" |
| 2090 | "ERROR: ASN.1 DECODE failed, double range out of local limit" |
| 2091 | "ERROR: Request block is invalid" |
| 2092 | "ERROR: Received type does not match definition" |
| 2093 | "ERROR: Received type does not match definition" |
| 2094 | "ERROR: Received type does not match definition" |
| 2095 | "ERROR: Received type does not match definition" |
| 2096 | "ERROR: Request block is invalid" |
| 2097 | "ERROR: Received type does not match definition" |
| 2098 | "ERROR: Received type does not match definition" |
| 2099 | "ERROR: Request block is invalid" |
| 2100 | "ERROR: Received type does not match definition" |
| 2101 | "ERROR: Received type does not match definition" |
| 2102 | "ERROR: Request block is invalid" |
| 2103 | "ERROR: Received type does not match definition" |
| 2104 | "ERROR: Received type does not match definition" |
| 2105 | "ERROR: Request block is invalid" |
| 2106 | "ERROR: Memory allocation error" |
| 2107 | "ERROR: Memory allocation error" |
| 2108 | "ERROR: Memory allocation error" |

| | |
|------|--|
| 2109 | "ERROR: Request block is invalid" |
| 2110 | "ERROR: Memory allocation error" |
| 2111 | "ERROR: Memory allocation error" |
| 2112 | "ERROR: Memory allocation error" |
| 2113 | "ERROR: Request block is invalid" |
| 2114 | "ERROR: Received type does not match definition" |
| 2115 | "ERROR: Received type does not match definition" |
| 2116 | "ERROR: Request block is invalid" |
| 2117 | "ERROR: Received type does not match definition" |
| 2118 | "ERROR: Received type does not match definition" |
| 2119 | "ERROR: Request block is invalid" |
| 2120 | "ERROR: Received type does not match definition" |
| 2121 | "ERROR: Received type does not match definition" |
| 2122 | "ERROR: Request block is invalid" |
| 2123 | "ERROR: Memory allocation error" |
| 2124 | "ERROR: Memory allocation error" |
| 2125 | "ERROR: Memory allocation error" |
| 2126 | "ERROR: Request block is invalid" |
| 2127 | "ERROR: Memory allocation error" |
| 2128 | "ERROR: Memory allocation error" |
| 2129 | "ERROR: Memory allocation error" |
| 2130 | "ERROR: Request block is invalid" |
| 2131 | "ERROR: Received type does not match definition" |
| 2132 | "ERROR: Received type does not match definition" |
| 2133 | "ERROR: Request block is invalid" |

| | |
|------|---|
| 2134 | "ERROR: Received type does not match definition" |
| 2135 | "ERROR: Received type does not match definition" |
| 2136 | "ERROR: Memory allocation error" |
| 2200 | "ERROR: Cannot malloc fd structures" |
| 2201 | "INFO: libgwo build: %s - %s" |
| 2202 | "ERROR: Malloc communication control structure failed" |
| 2203 | "WARN: Specified environmental variable GW_MAX_LISTENING_END_POINT out of valid range, reset to %d" |
| 2204 | "WARN: Specified environmental variable GW_DFLT_TRANTIME out of valid range, ignored!" |
| 2205 | "ERROR: Malloc APT encoding string buffer failed" |
| 2206 | "ERROR: Malloc AEQ encoding string buffer failed" |
| 2207 | "ERROR: Malloc APID encoding string buffer failed" |
| 2208 | "ERROR: Malloc AEID encoding string buffer failed" |
| 2209 | "ERROR: Cannot establish control instance, exiting process" |
| 2210 | "WARN: Could not establish listening instance %d" |
| 2211 | "ERROR: Could not establish any listening instance, exiting process" |
| 2212 | "ERROR: Malloc CID string buffer failed" |
| 2213 | "ERROR: Internal data structure not allocated yet" |
| 2214 | "ERROR: Invalid data index(%d)" |
| 2215 | "ERROR: File descriptor table exhausted" |
| 2216 | "ERROR: Invalid data index(%d)" |
| 2217 | "ERROR: Invalid file descriptor index(%d)" |
| 2218 | "ERROR: Invalid file descriptor(%d)" |

| | |
|------|---|
| 2219 | "ERROR: Internal data structure not allocated yet" |
| 2220 | "ERROR: Invalid data index(%d)" |
| 2221 | "ERROR: Invalid file descriptor index(%d)" |
| 2222 | "ERROR: Invalid file descriptor(%d)" |
| 2223 | "ERROR: Internal data structure not allocated yet" |
| 2224 | "ERROR: Invalid file descriptor index(%d)" |
| 2225 | "ERROR: Invalid data index(%d)" |
| 2226 | "ERROR: Internal data structure not allocated yet" |
| 2227 | "ERROR: Invalid data index(%d)" |
| 2228 | "ERROR: Invalid file descriptor index(%d)" |
| 2229 | "ERROR: Internal data structure not allocated yet" |
| 2230 | "ERROR: Invalid data index(%d)" |
| 2231 | "ERROR: Invalid file descriptor index(%d)" |
| 2232 | "ERROR: Internal data structure not allocated yet" |
| 2233 | "ERROR: Bad increment" |
| 2234 | "ERROR: Memory allocation failure" |
| 2235 | "ERROR: Internal data structure not allocated yet" |
| 2236 | "ERROR: Invalid input parameter" |
| 2237 | "ERROR: Internal protocol error, missing routing information" |
| 2238 | "ERROR: Remote OSITP domain not defined locally" |
| 2239 | "ERROR: Internal data structure not allocated yet" |
| 2240 | "ERROR: Invalid input parameter" |
| 2241 | "ERROR: Internal data structure not allocated yet" |
| 2300 | "ERROR: Bad input arguments, unable to create instance" |
| 2301 | "ERROR: Unable to init env on osi, %s" |

| | |
|------|---|
| 2302 | "ERROR: Unable to set AP_ROLE_ALLOWED, %s" |
| 2304 | "ERROR: Unable to set AP_TP_CATEGORY, %s" |
| 2305 | "ERROR: Unable to set AP_BIND_TPADDR, %s" |
| 2306 | "ERROR: Unable to set AP_FLAGS, %s" |
| 2307 | "ERROR: Could not get RDOM Cache entry" |
| 2308 | "ERROR: RDOM %s OSITP address not configured" |
| 2309 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2310 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2311 | "ERROR: set of COPYENV failed, %s" |
| 2312 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2313 | "ERROR: set of AP_MODE_SEL failed, %s" |
| 2314 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2315 | "ERROR: set of AP_URCH failed, %s" |
| 2316 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2317 | "ERROR: set of AP_CONTROL_ID failed, %s" |
| 2318 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2319 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2320 | "ERROR: Unable to get remote domain information from shmем" |
| 2321 | "ERROR: set of AP_TPFU_SEL failed, %s" |

| | |
|------|---|
| 2322 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2323 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2324 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2325 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2326 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2400 | "ERROR: ap_poll error, %s" |
| 2401 | "ERROR: Could not make new action" |
| 2402 | "WARN: Dropping polled action, no free contexts" |
| 2403 | "ERROR: Could not make new action" |
| 2404 | "WARN: Dropping polled action, no free contexts" |
| 2405 | "ERROR: AP_POLLOUT event has no associated action, fd = %x" |
| 2406 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2407 | "WARN: Could not create listening instance" |
| 2408 | "ERROR: Could not make new action" |
| 2409 | "WARN: Dropping polled action, no free contexts" |
| 2410 | "ERROR: Could not create any listening instance" |
| 2500 | "ERROR: Send AP_TP_BEGIN_DIALOGUE_RSP failed, %s" |
| 2501 | "ERROR: ap_rcv failed, %s" |
| 2502 | "ERROR: Unable to retrieve file descriptor, shutdown Gateway!" |
| 2503 | "ERROR: Could not allocate send buffer, shutdown Gateway!" |

| | |
|------|---|
| 2504 | "ERROR: Cannot create shared mem nettxid" |
| 2505 | "ERROR: Set of TTNID failed, %s" |
| 2506 | "WARN: Transaction table full, rejecting remote request!" |
| 2507 | "ERROR: OSI TP protocol error, shutdown Gateway!" |
| 2508 | "ERROR: Unable to assign id to incoming conversation" |
| 2509 | "ERROR: Decoding AP_TP_U_ABORT_IND data failed!" |
| 2510 | "ERROR: No conversation id associated with this fd" |
| 2511 | "ERROR: No conversation id associated with this fd" |
| 2512 | "ERROR: Invalid Buffer" |
| 2513 | "ERROR: Conversation id not found" |
| 2514 | "ERROR: Conversation id not found" |
| 2515 | "ERROR: Blocking timeout occured, deleting fd" |
| 2516 | "ERROR: Unable to get file descriptor" |
| 2517 | "ERROR: Received unknown op code %x" |
| 2518 | "ERROR: Conversation id not found" |
| 2519 | "ERROR: Conversation id not found" |
| 2520 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2521 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2522 | "ERROR: Received protocol error, sptype = %x" |
| 2523 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2524 | "Received AP_TP_P_ABORT_IND from OSITP" |
| 2525 | "A permanent error condition has been encountered" |
| 2526 | "A transient error condition has been encountered" |
| 2527 | "A protocol error has been encountered" |

| | |
|------|---|
| 2528 | "Transaction request rejected because recipient is already involved in a transaction, or a local error" |
| 2529 | "Two TP_END_DIALOG primitives have collided" |
| 2530 | "A TP_BEGIN_TRANSACTION_REQ and a TP_END_DIALOG_REQ have collided" |
| 2531 | "Invalid diagnostic value" |
| 2532 | "ERROR: Collision of DTNID detected, reject the dialogue!" |
| 2533 | "ERROR: Invalid Buffer" |
| 2534 | "ERROR: No conversation id associated with this fd" |
| 2535 | "ERROR: Invalid Buffer" |
| 2536 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2537 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2538 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2539 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2540 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2541 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 2542 | "ERROR: Protocol error, DiaFSM state(%d), shutdown Gateway!" |
| 2543 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2544 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2545 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2546 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2547 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2548 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2549 | "ERROR: Could not find transaction node, shutdown Gateway!" |

| | |
|------|---|
| 2550 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2551 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2552 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2553 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2554 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2555 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2556 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2557 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2558 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2559 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2560 | "ERROR: Protocol error, dial state(%d), shutdown Gateway!" |
| 2561 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2562 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2563 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2600 | "ERROR: Invalid conversation context" |
| 2601 | "ERROR: Unable to send AP_TP_DATA_REQ, %s" |
| 2602 | "ERROR: Unable to send %s, %s" |
| 2603 | "ERROR: Invalid conversation context" |
| 2604 | "ERROR: Unable to send AP_TP_U_ABORT_REQ, %s" |
| 2605 | "ERROR: Unable to send AP_TP_U_ABORT_REQ, %s" |
| 2606 | "ERROR: TP STATE = %lx" |
| 2607 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |

| | |
|------|---|
| 2608 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2609 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2610 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |
| 2800 | "ERROR: ap_free() failed, reason: %s" |
| 2801 | "ERROR: Unable to retrieve file descriptor!" |
| 2802 | "ERROR: Memory allocation failure" |
| 2803 | "ERROR: Invalid transaction node" |
| 2804 | "ERROR: Could not find transaction node!" |
| 2805 | "ERROR: Unable to obtain remote domain info from shared memory" |
| 2806 | "ERROR: Unable to obtain ositp info from rdom" |
| 2807 | "ERROR: Remote domain name not found" |
| 2808 | "ERROR: Cannot realloc fd structures" |
| 2809 | "ERROR: Could not create new listener" |
| 2810 | "ERROR: Stuctures have not been allocated" |
| 2811 | "ERROR: Memory allocation failure" |
| 2812 | "ERROR: Memory allocation failure" |
| 2813 | "ERROR: Memory allocation failure" |
| 2814 | "ERROR: Memory allocation failure" |
| 2815 | "ERROR: gw_tx_end returned failure" |
| 2816 | "ERROR: Memory allocation failure" |
| 2817 | "ERROR: Memory allocation failure" |
| 2818 | "ERROR: Memory allocation failure" |

| | |
|------|--|
| 2819 | "ERROR: Memory allocation failure" |
| 2820 | "ERROR: Memory allocation failure" |
| 2821 | "ERROR: Invalid input transaction id" |
| 2822 | "ERROR: Exceeding local transaction limit" |
| 2823 | "ERROR: Branch does not belong to this transaction" |
| 2824 | "ERROR: Transaction branch does not exist" |
| 2825 | "ERROR: Invalid input file descriptor" |
| 2826 | "ERROR: Exceeding local transaction limit" |
| 2827 | "ERROR: Transaction branch already exists" |
| 2828 | "ERROR: Invalid input file descriptor" |
| 2829 | "ERROR: Branch does not belong to this transaction" |
| 2830 | "ERROR: Transaction branch does not exist" |
| 2831 | "ERROR: Invalid input file descriptor" |
| 2832 | "ERROR: Exceeding local transaction limit" |
| 2833 | "ERROR: Transaction branch already exists" |
| 2834 | "ERROR: Invalid input file descriptor" |
| 2835 | "ERROR: get of AP_CNTX_NAME failed, %s" |
| 2836 | "ERROR: Can't make object identifier" |
| 2837 | "ERROR: Can't make object identifier" |
| 2838 | "ERROR: Invalid Application Context Name" |
| 2841 | "ERROR: TPFU_SEL_BIT not set" |
| 2842 | "ERROR: TPFU not supported, bit value = %x" |
| 2843 | "ERROR: Memory allocation failure" |
| 2844 | "ERROR: Corrupted or uninitialized file descriptor" |
| 2845 | "ERROR: Allocating Network Transaction table failed" |

| | |
|------|--|
| 2846 | "ERROR: Bad input Dialogue Tree Node Identifier" |
| 2847 | "ERROR: Bad input transaction identification" |
| 2848 | "ERROR: Bad input transaction identification" |
| 2849 | "ERROR: Bad input transaction identification" |
| 2850 | "ERROR: Invalid transaction look up table entry" |
| 2851 | "ERROR: Invalid or corrupted network context descriptor %d" |
| 2852 | "ERROR: Invalid network context descriptor %d" |
| 2853 | "ERROR: Invalid conversation descriptor %d" |
| 2854 | "ERROR: Invalid action descriptor %d" |
| 2855 | "ERROR: Invalid context descriptor %d" |
| 2856 | "ERROR: Invalid action descriptor %d" |
| 2857 | "ERROR: Invalid context descriptor %d" |
| 2858 | "WARN: Possible orphaned conversation %d" |
| 2859 | "ERROR: Invalid action descriptor" |
| 2860 | "ERROR: Invalid file descriptor" |
| 2861 | "ERROR: Invalid file descriptor" |
| 2862 | "ERROR: Invalid action descriptor %d" |
| 2863 | "ERROR: Maximum system context limit already reached" |
| 2864 | "ERROR: Corrupted or uninitialized file descriptor" |
| 2865 | "ERROR: Invalid conversation descriptor" |
| 2866 | "ERROR: Invalid file descriptor" |
| 2867 | "WARN: Remote domain(%s) XATMI encoding not specified, use default" |
| 2868 | "ERROR: Remote OSITP domain %s address not configured" |
| 2869 | "ERROR: Wrong remote domain type(%s)" |

| | |
|------|--|
| 2870 | "ERROR: Can not find correct presentation context identifier" |
| 2871 | "ERROR: Unable to get AP_DCS, %s" |
| 2900 | "ERROR: ap_rcv failed, %s" |
| 2901 | "ERROR: Received protocol error, sptype = %x" |
| 2902 | "ERROR: failed to insert blob segment into transaction table!" |
| 2903 | "ERROR: Event stack overflow, shutdown Gateway!" |
| 2904 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |
| 2905 | "ERROR: Could not find transaction table entry, shutdown Gateway!" |
| 2906 | "ERROR: Cannot allocate RPCRQ buffer" |
| 2907 | "ERROR: Can't get local services from configuration file" |
| 2908 | "ERROR: Could not find transaction table entry, shutdown Gateway!" |
| 2909 | "ERROR: Incoming request not allowed access to any local services" |
| 2910 | "ERROR: Remote domain name not found in configuration" |
| 2911 | "ERROR: Service is denied Access" |
| 2912 | "ERROR: Can't find LSVC entry by rname %s" |
| 2913 | "ERROR: Send AP_TP_HANDSHAKE_AND_GRANT_CONTROL_RSP failed, %s" |
| 2914 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2915 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2916 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2917 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |
| 2918 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |
| 2919 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |

| | |
|------|--|
| 2920 | "ERROR: Protocol error, msg state(%d), shutdown Gateway!" |
| 2921 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2922 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2923 | "ERROR: Memory allocation failure" |
| 2924 | "ERROR: Memory allocation failure" |
| 2925 | "ERROR: ioptr_init_char failed" |
| 2926 | "ERROR: Encode of U_ABORT_RI failed" |
| 2927 | "ERROR: Memory allocation failure" |
| 2928 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2929 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2930 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2931 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2932 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2933 | "ERROR: Could not find transaction node, shutdown Gateway!" |
| 2934 | "ERROR: Create transaction node failed, shutdown Gateway!" |
| 2935 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2936 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2937 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2938 | "ERROR: Lost all control end point context, shutdown GWOSITP!" |
| 2939 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2940 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2941 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2942 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2943 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |

| | |
|------|---|
| 2944 | "ERROR: Protocol error, node state(%d), shutdown Gateway!" |
| 2945 | "WARN: receive %s without TTNID or DTNID, drop the action" |
| 2946 | "WARN: No OSI/TP service provider, shutdown OSITP domain gateway" |

B Reference Pages

This appendix covers the following reference pages, formerly called man pages:

- ◆ addumap
- ◆ addusr
- ◆ build_dgw
- ◆ delumap
- ◆ delusr
- ◆ DMADM
- ◆ dmadmin
- ◆ dmconfig
- ◆ dmloadcf
- ◆ dmusradd
- ◆ dmusrmod
- ◆ GWADM
- ◆ modusr

addumap

Add a local-to-remote mapping for a local/remote domain pair

SYNOPSIS

```
DMCONFIG=dmconfig
addumap -d local domain ID -R remote domain ID
-p local principal name -u remote username
```

DESCRIPTION

addumap can only be executed as a subcommand of dmadmin. The purpose of this page is to describe options for the subcommand and to show examples.

The subcommand allows the administrator to add local-to-remote user mappings for a local/remote domain pair.

Mappings are defined to be inbound, outbound or both when the application is using OSI Domain gateways and SECURITY is set to USER_AUTH, ACL, or MANDATORY_ACL in the ubbconfig file and SECURITY is set to DM_PW or USER_PW in the DMCNFIG file.

The following options are available:

-d *local domain ID*

This is the name of the local domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the DMCNFIG file or through the Graphical Administrative Interface.

-R *remote domain ID*

This is the name of the remote domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the DMCNFIG file or through the Graphical Administrative Interface.

-p *local principal*

The user identification number. The *local principal* must be defined in the ACL user file and must be unique within the list of existing identifiers for the application.

-u *remote username*

The remote user name as defined in the ACL security application (for example, RACF) of the remote domain.

Before running this subcommand the application must be configured using either the Graphical Administrative Interface or `tmloadcf` and `dmloadcf`. `dmadmin addumap` may be run on any active node.

PORTABILITY

This subcommand is available only on non-/WS sites running TUXEDO System /T Release 6.4.

DIAGNOSTICS

The `dmadmin addumap` subcommand exits with a return code of 0 upon successful completion.

EXAMPLE

```
addumap -d ldom -R cics -p tuxusr -u cicsusr
/*maps principal tuxusr with
remote user cicsusr */
```

SEE ALSO

`dmadmin`, `delumap`

addusr

Add a user to the remote domain user and password file

SYNOPSIS

```
DMCONFIG=dmconfig
addusr -d local domain ID -R remote domain ID -u remote username
[-w ]
```

DESCRIPTION

`addusr` can only be executed as a subcommand of `dmadmin`. The purpose of this page is to describe options for the subcommand and to show an example.

The subcommand allows the administrator to add remote usernames and passwords to the remote domain remote user and password table. If `-w` is not specified, the user must be prompted for a password.

The table entries created are used for passing remote user names and passwords to remote OSI domains when the application is using OSI Domain gateways and `SECURITY` is set to `USER_AUTH`, `ACL`, or `MANDATORY` `ACL` in the `ubbconfig` file and `SECURITY` is set to `DM_PW` or `USER_PW` in the `DMCONFIG` file.

The following options are available:

`-d local domain ID`

This is the name of the local domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the `DMCONFIG` file or through the Graphical Administrative Interface.

`-R remote domain ID`

This is the name of the remote domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the `DMCONFIG` file or through the Graphical Administrative Interface.

`-u remote username`

The remote user name to be added.

`-w`

Do not prompt for password.

Before running this subcommand the application must be configured using either the Graphical Administrative Interface or `tmloadcf` and `dmloadcf`. `dmadmin addusr` may be run on any active node.

PORTABILITY

This subcommand is available only on non-/WS sites running TUXEDO System /T Release 6.4.

DIAGNOSTICS

The `dmadmin addusr` subcommand exits with a return code of 0 upon successful completion.

EXAMPLES

```
addusr -d tux -R cics -u cicsusr /*adds remote user cicsusr to cics
                                domain users. The administrator is
                                prompted for a password*/
```

SEE ALSO

`delusr`, `modusr`

build_dgw

Builds a customized domain gateway process.

SYNOPSIS

```
build_dgw [ -c dmtype] [ -o name] [ -v ]
```

DESCRIPTION

Constructs a customized TUXEDO System/T domain gateway module. The files included by the caller should include only the application buffer type switch and any required supporting routines. The command combines the files supplied by the `-c` option with the standard TUXEDO System/T libraries necessary to form a gateway load module. The load module is built by the `cc` command described in UNIX System reference manuals which `build_dgw` invokes. The options to `build_dgw` have the following meaning:

`-c dmtype`

specifies a domain type that characterizes communications access module domain gateway. The value of *dmtype* must appear in the domain type table located in `$TUXDIR/udataobj/DMTYPE`. Each line in this file has the form:

```
dmtype:access_module_lib:comm_libs:tm_typesw_lib:gw_typesw_lib
```

Using the *dmtype* value, `build_dgw` retrieves the corresponding entry from `$TUXDIR/udataobj/DMTYPE`. The *access_module_lib* specifies the libraries used by this particular type of domain instantiation. The *comm_libs* parameter contains a list of the networking communications libraries used by the access module. The *tm_typesw_lib* parameter defines a list of libraries or object modules with the definition of the typed buffers used by the local application. If this parameter is not defined, the gateway will be linked with the default typed buffer definitions. The *gw_typesw_lib* parameter applies only to a gateway of type OSITP and defines a list of libraries or object modules used by the gateway to transform buffers into the protocol required by the remote domain. There should be a one-to-one mapping between the buffer types defined in the *tm_typesw* array (see *typesw* and *tuxtypes*) and the *gw_typesw* array (see *gwtypesw* and *gwtypes*). If this parameter is not defined, the gateway will be linked with the default typed buffer definitions provided with the OSITP instantiation.

Currently, *dmtype* may be set to one of the following values:

TDOMAIN

builds a gateway for communications with another System/T domain. The `build_dgw` command will use the standard TUXEDO System/T `libnws.a` networking library. This is the default option.

OSITP

builds a gateway for communications with an OSI TP domain. The OSITP access module uses the XAP-TP interface. The pathname for the library containing the XAP-TP primitives is provider dependent and should be set according to the provider's specifications.

-o name

specifies the name of the file the output gateway load module is to have. If not supplied, the load module is named `GWTDOMAIN` for the `TDOMAIN` type or `GWOSITP` for the `OSITP` type. Note that the name selected for the load module must also be the name used for the definition of the gateway in the `SERVERS` section of the `TUXCONFIG` file.

-v

specifies that `build_dgw` should work in verbose mode. In particular, it writes the `cc` command to its standard output.

`build_dgw` normally uses the `cc` command to produce the `a.out`. In order to allow for the specification of an alternate compiler, `build_dgw` checks for the existence of a shell variable named `CC`. If `CC` does not exist in `build_dgw`'s environment, or if it is the string "", `build_dgw` will use `cc` as the compiler. If `CC` does exist in the environment, its value is taken to be the name of the compiler to be executed. Likewise, the shell variable `CFLAGS` is taken to contain a set of parameters to be passed to the compiler.

PORTABILITY

`build_dgw` is supported as a TUXEDO System/T-supplied compilation tool on UNIX operating systems only.

EXAMPLES

The following example shows how to build a domain gateway of type `TDOMAIN`.

```
CC=ncc CFLAGS="-I $TUXDIR/include"; export CC CFLAGS
build_dgw -o DGW
```

The following example shows use of `build_dgw` for an OSI TP instantiation:

```
build_dgw -c OSITP -o OTPGW
```

The DMTYPE file will contain the following entries:

```
TDOMAIN:$TUXDIR/lib/libgwt.a:$TUXDIR/lib/libnwi.a  
$TUXDIR/lib/libnws.a::  
OSITP:$TUXDIR/lib/libgwo.a:-l xaptp -l ositp::  
SNA:libgws.a:$TUXDIR/lib/libcpic.a:$SNADIR/lib/libсна.a
```

The paths for the `libxaptp.a` and `libositp.a` libraries are installation and provider dependent. The application administrator must specify the correct pathnames before building a /OSITP gateway instantiation.

SEE ALSO

`cc`, `ld` in UNIX System reference manuals

`gwtypes`, `gwtypesw`, `tuxtypes`, `typesw`

delumap

Delete a local-to-remote mapping for a local/remote domain pair

SYNOPSIS

```
DMCONFIG=dmconfig  
delumap -d local domain ID -R remote domain ID  
-p local principal name -u remote username
```

DESCRIPTION

delumap can only be executed as a subcommand of dmadmin. The purpose of this page is to describe options for the subcommand and to show examples.

The subcommand allows the administrator to delete local-to-remote user mappings for a local/remote domain pair.

Mappings are defined to be inbound, outbound or both when the application is using OSI Domain gateways and SECURITY is set to USER_AUTH, ACL, or MANDATORY_ACL in the ubbconfig file and SECURITY is set to DM_PW or USER_PW in the DMCNFIG file.

The following options are available:

-d *local domain ID*

This is the name of the local domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the DMCNFIG file or through the Graphical Administrative Interface.

-R *remote domain ID*

This is the name of the remote domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the DMCNFIG file or through the Graphical Administrative Interface.

-p *local principal*

The user identification number. The *local principal* must be defined in the ACL user file and must be unique within the list of existing identifiers for the application.

-u *remote username*

The remote user name as defined in the ACL security application (for example, RACF) of the remote domain. Space is a valid remote username.

Before running this subcommand the application must be configured using either the Graphical Administrative Interface or `tmloadcf` and `dmloadcf`. `dmadmin delumap` may be run on any active node.

PORTABILITY

This subcommand is available only on non-/WS sites running TUXEDO System /T Release 6.4.

DIAGNOSTICS

The `dmadmin delumap` subcommand exits with a return code of 0 upon successful completion.

EXAMPLE

```
delumap -d ldom -R cics -p tuxusr -u cicsusr
/*deletes the mapping of principal
tuxusr with remote user cicsusr */
```

SEE ALSO

`dmadmin`, `addumap`

delusr

Delete a user from the remote domain user and password file

SYNOPSIS

```
DMCONFIG=dmconfig  
delusr -d local domain -R remote domain -u remote username
```

DESCRIPTION

`delusr` can only be executed as a subcommand of `dmadmin`. The purpose of this page is to describe options for the subcommand and to show an example.

The subcommand allows the administrator to remove remote usernames and passwords from the remote domain remote user and password table.

Once the entries are deleted they can no longer be used for mapping remote user names and passwords to local user names and passwords when the application is using OSI Domain gateways and `SECURITY` is set to `USER_AUTH`, `ACL`, or `MANDATORY ACL` in the `ubbconfig` file and `SECURITY` is set to `DM_USER_PW` in the `DMCONFIG` file.

The following options are available:

`-d local domain ID`

This is the name of the local domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the `DMCONFIG` file or through the Graphical Administrative Interface.

`-R remote domain ID`

This is the name of the remote domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the `DMCONFIG` file or through the Graphical Administrative Interface.

`-u remote username`

The remote user name to be deleted.

Before running this subcommand the application must be configured using either the Graphical Administrative Interface or `tmloadcf` and `dmloadcf`. `dmadmin delusr` may be run on any active node.

PORTABILITY

This subcommand is available only on non-/WS sites running TUXEDO System /T Release 6.4.

DIAGNOSTICS

The `dmadmin delusr` subcommand exits with a return code of 0 upon successful completion.

EXAMPLES

```
delusr -d tux -R cics -u cicsusr /*deletes remote user cicsusr to
                                cics domain users. The
                                administrator is prompted for a
                                password*/
```

SEE ALSO

`addusr`, `modusr`

DMADM

/Domain administrative server

SYNOPSIS

```
DMADM SRVGRP = "identifier"
SRVID = "number"
REPLYQ = "N"
```

DESCRIPTION

The /DOMAIN administrative server DMADM is a TUXEDO-supplied server that provides run-time access to the BDMCONFIG file. When DMADM is booted the BDMCONFIG environment variable should be set to the pathname of the file containing the binary version of the DMCONFIG file.

DMADM is described in the *SERVERS section of the UBBCONFIG file as a server running within a group, e.g., DMADMGRP. There should be only one instance of the DMADM running in this group and it must not have a reply queue (REPLYQ must be set to "N").

The following server parameters can also be specified for the DMADM server in the *SERVERS section: SEQUENCE, ENVFILE, MAXGEN, GRACE, RESTART, RQPERM and SYSTEM_ACCESS.

PORTABILITY

DMADM is supported as a TUXEDO-supplied server on UNIX System operating systems.

INTEROPERABILITY

The initial release of OSI Domain can only be installed on a node running TUXEDO Release 6.4.

EXAMPLES

The following example illustrates the definition of the administrative server and a gateway group in the UBBCONFIG file.

```
#
*GROUPS
DMADMGRP  LMID=mach1  GRPNO=1
gwgrp     LMID=mach1  GRPNO=2
```

```
#
*SERVERS
DMADM SRVGRP="DMADMGRP" SRVID=1001 REPLYQ=N RESTART=Y GRACE=0
GWADM SRVGRP="gwgrp" SRVID=1002 REPLYQ=N RESTART=Y GRACE=0
GWTDOMAIN SRVGRP="gwgrp" SRVID=1003 RQADDR="gwgrp" REPLYQ=Y
RESTART=Y MIN=1 MAX=1
```

SEE ALSO

dmadmin, tmboot, dmconfig, GWADM, servopts, ubbconfig

TUXEDO /Domain Guide

TUXEDO Administrator's Guide

dmadmin

TUXEDO System/T Domain Administration Command Interpreter

SYNOPSIS

```
dmadmin [-c]
```

DESCRIPTION

`dmadmin` is an interactive command interpreter used for the administration of domain gateway groups defined for a particular TUXEDO System/T application. `dmadmin` can operate in two modes: administration mode and configuration mode.

`dmadmin` enters *administration* mode when called with no parameters. This is the default. In this mode, `dmadmin` can be run on any active node (excluding workstations) within an active application. Application administrators can use this mode to obtain or change parameters on any active domain gateway group. Application administrators may also use this mode to create, destroy, or reinitialize the `DMTLOG` for a particular local domain. In this case, the domain gateway group associated with that local domain must not be active, and `dmadmin` must be run on the machine assigned to the corresponding gateway group.

`dmadmin` enters *configuration* mode when it is invoked with the `-c` option or when the `config` subcommand is invoked. Application administrators can use this mode to update or add new configuration information to the binary version of the domain configuration file `BDMCONFIG`.

`dmadmin` requires the use of the `DOMAIN` administrative server `DMADM` for the administration of the `BDMCONFIG` file and the gateway administrative server `GWADM` for the re-configuration of active `DOMAIN` gateway groups (there is one `GWADM` per gateway group).

ADMINISTRATION MODE COMMANDS

Once `dmadmin` has been invoked, commands may be entered at the prompt (“>”) according to the following syntax:

```
command [arguments]
```

Several commonly occurring arguments can be given default values via the default command. Commands that accept parameters set via the default command check default to see if a value has been set. If no value is set, an error message is returned.

Once set, a default value remains in effect until the session is ended, unless changed by another default command. Defaults may be overridden by entering an explicit value on the command line, or unset by entering the value “*”. The effect of an override lasts for a single instance of the command.

Output from `dmadmin` commands is paginated according to the pagination command in use (see the definition for the `paginate` subcommand).

Commands may be entered either by their full name or their abbreviation (shown in parentheses) followed by any appropriate arguments. Arguments appearing in square brackets, [], are optional; those in curly braces, {}, indicate a selection from mutually exclusive options. Note that for many commands *local_domain_name* is a required argument, but note also that it can be set with the default command.

The following commands are available in administration mode:

`addumap [options]`

Add local user mappings to remote user mappings for a local/remote domain pair. Mappings are defined to be inbound, outbound or both. See the `addumap` manual page for an explanation of the available options and for examples.

`addusr (addu) [options]`

Add remote usernames and passwords to the remote user and password tables of a remote domain. See the `addusr` manual page for an explanation of the available options and for examples.

`advertise (adv) -d local_domain_name [{ -all | service}]`

Advertise all remote services provided by the named local domain or the specified remote service.

`audit (audit) -d local_domain_name [{off | on}]`

Activate (on) or deactivate (off) the audit trace for the named local domain. If no option is given, then the current setting will be toggled between the values on and off, and the new setting will be printed. The initial setting is off.

`chbktme (chbt) -d local_domain_name -t bktme`

Change the blocking timeout for a particular local domain.

`config (config)`

Enter configuration mode. Commands issued in this mode follow the conventions defined in the section “CONFIGURATION MODE COMMANDS” on page B-20.

`crdmlog (crdlg) -d local_domain_name`

Create the domain transaction log for the named local domain on the current machine (that is, the machine where `dmadmin` is running). The command uses the parameters specified in the `DMCONFIG` file. This command fails if the named local domain is active on the current machine or if the log already exists.

`default (d) [-d local_domain_name]`

Set the corresponding argument to be the default local domain. Defaults may be unset by specifying “*” as an argument.

If the `default` command is entered with no arguments, the current default values are printed.

`delumap [options]`

Delete local to remote user mappings for a local/remote domain pair. See the `delumap` manual page for an explanation of the available options and for examples.

`delusr (delu) [options]`

Delete remote usernames and passwords from the remote user and password tables of a remote domain. See the `delusr` reference page for an explanation of the available options and for examples.

`dsdmlog (dsdlg) -d local_domain_name [-y]`

Destroy the domain transaction log for the named local domain on the current machine (that is, the machine where `dmadmin` is running). An error is returned if a `DMTLOG` is not defined for this local domain, if the local domain is active, or if outstanding transaction records exist in the log. The term outstanding transactions means that a global transaction has been committed but an end-of-transaction has not yet been written. This command prompts for confirmation before proceeding unless the `-y` option is specified.

`echo (e) [{off | on}]`

Echo input command lines when set to `on`. If no option is given, then the current setting is toggled, and the new setting is printed. The initial setting is `off`.

`forgettrans (ft) -d local_domain_name [-t tran_id]`

Forget one or all heuristic log records for the named local domain. If the transaction identifier *tran_id* is specified, then only the heuristic log record for that transaction will be forgotten. The transaction identifier *tran_id* can be obtained from the `printtrans` command or from the `ULOG` file.

`help (h) [command]`

Print help messages. If *command* is specified, the abbreviation, arguments, and description for that command are printed. Omitting all arguments causes the syntax of all commands to be displayed.

`indmlog (indlg) -d local_domain_name [-y]`

Reinitialize the domain transaction log for the named local domain on the current machine (that is, the machine where `dmadmin` is running). An error is returned if a `DMTLOG` is not defined for this local domain, if the local domain is active, or if outstanding transaction records exist in the log. The term outstanding transactions means that a global transaction has been committed but an end-of-transaction has not yet been written. The command prompts for confirmation before proceeding unless the `-y` option is specified.

`modusr (modu) [options]`

Change remote passwords in the password tables of a remote domain. See the reference page “`modusr`” on page B-67 for an explanation of the available options and for examples.

`paginate (page) [{off | on}]`

Paginate output. If no option is given, then the current setting will be toggled, and the new setting is printed. The initial setting is on, unless either standard input or standard output is a non-tty device. Pagination may only be turned on when both standard input and standard output are tty devices. The shell environment variable `PAGER` may be used to override the default command used for paging output. The default paging command is the indigenous one to the native operating system environment, for example, the command `pg` is the default on UNIX System operating environments.

`passwd (passwd) [-r] local_domain_name remote_domain_name`

Prompts the administrator for new passwords for the specified local and remote domains. The `-r` option specifies that existing passwords and new passwords should be encrypted using a new key generated by the system. The password is truncated after at most eight characters.

`printdomain (pd) -d local_domain_name`

Print information about the named local domain. Information printed includes connected remote domains, global information shared by the gateway processes, and additional information that is dependent on the domain type instantiation.

`printstats (stats) -d local_domain_name`
 Print statistical and performance information gathered by the named local domain. The information printed is dependent on the domain gateway type.

`printtrans (pt) -d local_domain_name`
 Print transaction information for the named local domain.

`quit (q)`
 Terminate the session.

`resume (res) -d local_domain_name [{ -all | service}]`
 Resume processing of the specified service or for all remote services handled by the named local domain.

`stats (stats) -d local_domain_name [{ off | on | reset }]`
 Activate (*on*), deactivate (*off*), or reset (*reset*) statistics gathering for the named local domain. If no option is given, then the current setting will be toggled between the values *on* and *off*, and the new setting will be printed. The initial setting is *off*.

`suspend (susp) -d local_domain_name [{ -all | service}]`
 Suspend one or all remote services for the named local domain.

`unadvertise (unadv) -d local_domain_name [{ -all | service}]`
 Unadvertise one or all remote services for the named local domain.

`verbose (v) [{off | on}]`
 Produce output in verbose mode. If no option is given, then the current setting will be toggled, and the new setting is printed. The initial setting is *off*.

`! shellcommand`
 Escape to shell and execute *shellcommand*.

`!!`
 Repeat previous shell command.

`# [text]`
 Lines beginning with "#" are comment lines and are ignored.

`<CR>`
 Repeat the last command.

CONFIGURATION MODE COMMANDS

The `dmadmin` command enters configuration mode when executed with the `-c` option or when the `config` subcommand is used. In this mode, `dmadmin` allows run-time updates to the `BDMCONFIG` file. `dmadmin` manages a buffer that contains input field values to be added or retrieved, and displays output field values and status after each operation completes. The user can update the input buffer using any available text editor.

`dmadmin` first prompts for the desired section followed by a prompt for the desired operation.

The prompt for the section is as follows:

Sections:

- | | |
|-------------------|--------------------|
| 1) LOCAL_DOMAINS | 2) REMOTE_DOMAINS |
| 3) LOCAL_SERVICES | 4) REMOTE_SERVICES |
| 5) ROUTING | 6) ACCESS_CONTROL |
| 7) PASSWORDS | 8) TDOMAIN |
| 9) OSITP | 10) SNA |
| 11) QUIT | |

Enter Section [1]:

The number of the default section appears in square brackets at the end of the prompt. You can accept the default by pressing `RETURN` or `ENTER`. To select another section enter its number, then press `RETURN` or `ENTER`.

`dmadmin` then prompts for the desired operation.

Operations:

- | | |
|----------------|-----------|
| 1) FIRST | 2) NEXT |
| 3) RETRIEVE | 4) ADD |
| 5) UPDATE | 6) DELETE |
| 7) NEW_SECTION | 8) QUIT |

Enter Operation [1]:

The number of the default operation is printed in square brackets at the end of the prompt. Pressing **RETURN** or **ENTER** selects this option. To select another operation enter its number, then press **RETURN** or **ENTER**.

The currently supported operations are:

1. FIRST

Retrieve the first record from the specified section. No key fields are needed (they are ignored if in the input buffer).

2. NEXT

Retrieve the next record from the specified section, based on the key fields in the input buffer.

3. RETRIEVE

Retrieve the indicated record from the specified section by key field(s) (see the following fields description).

4. ADD

Add the indicated record in the specified section. Any fields not specified (unless required) take their default values as specified in `dmconfig`. The current value for all fields is returned in the output buffer. This operation can only be done by the System/T administrator.

5. UPDATE

Update the record specified in the input buffer in the selected section. Any fields not specified in the input buffer remain unchanged. The current value for all fields is returned in the input buffer. This operation can only be done by the System/T administrator.

6. DELETE

Delete the record specified in the input buffer from the selected section. This operation can only be done by the System/T administrator.

7. NEW SECTION

Clear the input buffer (all fields are deleted). After this operation, `dmadmin` immediately prompts for the section again.

8. QUIT

Exit the program gracefully (`dmadmin` is terminated). A value of `q` for any prompt also exits the program.

For configuration operations, the effective user identifier must match the System/T administrator user identifier `UID` for the machine on which this program is executed. When a record is updated or added, all default values and validations used by `dmloadcf` are enforced.

`dmadmin` then prompts whether or not to edit the input buffer.

Enter editor to add/modify fields [n]?

Entering a value of *y* will put the input buffer into a temporary file and execute the text editor. The environment variable `EDITOR` is used to determine which editor to be used; the default is “`ed`”. The input format is in field name/field value pairs and is described in the `CONFIGURATION INPUT FORMAT` section that follows. The field names associated with each `DMCONFIG` section are listed in tables in the subsections that follow. The semantics of the fields and associated ranges, default values, restrictions, etc., are described in `dmconfig`. In most cases, the field name is the same as the `KEYWORD` in the `DMCONFIG` file, prefixed with “`TA_`”. When the user completes editing the input buffer, `dmadmin` reads it. If more than one line occurs for a particular field name, the first occurrence is used and other occurrences are ignored. If any errors occur, a syntax error will be printed and `dmadmin` prompts whether or not to correct the problem.

Enter editor to correct?

If the problem is not corrected (response *n*), then the input buffer will contain no fields. Otherwise, the editor is executed again.

Finally, `dmadmin` asks if the operation should be done.

Perform operation [*y*]?

When the operation completes, `dmadmin` prints the return value as in

Return value `TAOK`

followed by the output buffer fields. The process then begins again with a prompt for the section. All output buffer fields are available in the input buffer unless the buffer is cleared.

Entering break at any time restarts the interaction at the prompt for the section.

When “`QUIT`” is selected, `dmadmin` prompts for authorization to create a backup ASCII version of the configuration:

Unload `BDMCONFIG` file into ASCII backup [*y*]?

If a backup is selected, `dmadmin` prompts for the file name.

Backup filename [`DMCONFIG`]?

On success, `dmadmin` indicates that a backup was created, otherwise an error is printed.

CONFIGURATION INPUT FORMAT

Input packets consist of lines formatted as follows:

```
fldname<tabs>fldval
```

The field name is separated from the field value by one or more tabs (or spaces).

Lengthy field values can be continued on the next line by having the continuation line begin with one or more tabs (which are dropped when read back into `dmadmin`).

Empty lines consisting of a single newline character are ignored.

To enter an unprintable character in the field value or to start a field value with a tab, use a backslash followed by the two-character hexadecimal representation of the desired character (see ASCII in a UNIX reference manual). A space, for example, can be entered in the input data as `\20`. A backslash can be entered using two backslash characters. `dmadmin` recognizes all input in this format, but its greatest usefulness is for non-printing characters.

CONFIGURATION LIMITATIONS

The following are general limitations of the dynamic domain re-configuration capability:

- ◆ Values for key fields (as indicated in the following sections) may not be modified. Key fields can be modified, when the system is down, by reloading the configuration file.
- ◆ Dynamic deletions cannot be applied when local domains are active (the corresponding gateway group is running).

RESTRICTIONS FOR CONFIGURATION FIELD IDENTIFIERS/UPDATES

The following sections describe, for each `DMCONFIG` section, what the field identifiers are for each `DMCONFIG` field, what the field type of the identifier is, and when the field can be updated. All applicable field values are returned with the retrieval operations. Fields that are allowed and/or required for adding a record are described in `dmconfig`. The following fields indicated as *key* are key fields that are used to uniquely identify a record within section. These key fields are required to be in the input buffer when updates are done and are not allowed to be updated dynamically. The `Update` column indicates when a field can be updated. The possible values are:

Yes

Can be updated at any time.

NoGW

Cannot be updated dynamically while the gateway group representing the local domain is running.

No

Cannot be updated dynamically while at least one gateway group is running.

CONFIGURING THE DM_LOCAL_DOMAINS SECTION

The following table lists the fields in the *DM_LOCAL_DOMAINS section.

Table B-1 *DM_LOCAL_DOMAINS SECTION

| Field Identifier | Field Type | Update | Notes |
|------------------|------------|--------|---------------------------------|
| TA_LDOM | string | NoGW | key |
| TA_AUDITLOG | string | Yes | |
| TA_BLOCKTIME | numeric | Yes | |
| TA_DOMAINID | string | NoGW | |
| TA_DMTLOGDEV | string | NoGW | |
| TA_DMTLOGNAME | string | NoGW | |
| TA_DMTLOGSIZE | numeric | NoGW | |
| TA_GWGRP | string | NoGW | |
| TA_MAXDATALEN | numeric | Yes | |
| TA_MAXRDOM | numeric | Yes | |
| TA_MAXRDTRAN | numeric | NoGW | |
| TA_MAXTRAN | numeric | NoGW | |
| TA_SECURITY | string | Yes | format: {NONE APP_PW DM_PW} |
| TA_TYPE | string | NoGW | format: {TDOMAIN OSITP SNA} |

CONFIGURING THE DM_REMOTE_DOMAINS SECTION

The following table lists the fields in the *DM_REMOTE_DOMAINS section.

Table B-2 *DM_REMOTE_DOMAINS SECTION

| Field Identifier | Field Type | Update | Notes |
|------------------|------------|--------|----------------------------------|
| TA_RDOM | string | No | key |
| TA_DOMAINID | string | No | |
| TA_TYPE | string | No | format: {TDOMAIN OSITP SNA } |

CONFIGURING THE DM_TDOMAIN SECTION

The *DM_TDOMAIN section contains the network addressing parameters required by TDOMAIN type domains. The following lists the fields in this section:

Table B-3 *DM_TDOMAIN SECTION

| Field Identifier | Field Type | Update | Notes |
|-----------------------|------------|---------|--|
| TA_LDOM or TA_RDOM | string | No/NoGW | key |
| TA_NWADDR | string | No/NoGW | ASCII format (no embedded NULL characters) |
| TA_NWDEVICE | string | No/NoGW | |

If the domain identifier (TA_LDOM) is a local domain identifier, then the TA_NWADDR and TA_NWDEVICE fields can be updated if the gateway group representing that local domain is not running.

CONFIGURING THE DM_OSITP SECTION

The *DM_OSITP section contains the network addressing parameters required by OSITP type domains. The following lists the fields in this section:

Table B-4 *DM_OSITP SECTION

| Field Identifier | Field Type | Update | Notes |
|-----------------------|------------|---------|-------|
| TA_LDOM or TA_RDOM | string | No/NoGW | key |
| TA_APT | string | No/NoGW | |
| TA_AEQ | string | No/NoGW | |
| TA_AET | string | No/NoGW | |
| TA_ACN | string | No/NoGW | |
| TA_APID | string | No/NoGW | |
| TA_AEID | string | No/NoGW | |
| TA_PROFILE | string | No/NoGW | |

If the domain identifier (TA_LDOM) is a local domain identifier, then the other fields in this table can be updated if the gateway group representing that local domain is not running.

CONFIGURING THE DM_LOCAL_SERVICES SECTION

The following table lists the fields in the *DM_LOCAL_SERVICES section.

Table B-5 *DM_LOCAL_SERVICES SECTION

| Field Identifier | Field Type | Update | Notes |
|------------------|------------|--------|-------|
| TA_SERVICENAME | string | No | key |
| TA_LDOM | string | Yes | |
| TA_RNAME | string | Yes | |
| TA_ACLNAME | string | Yes | |

CONFIGURING THE DM_REMOTE_SERVICES SECTION

The following table lists the fields in the *DM_REMOTE_SERVICES section.

Table B-6 *DM_REMOTE_SERVICES SECTION

| Field Identifier | Field Type | Update | Notes |
|------------------|------------|--------|-------------------|
| TA_SERVICENAME | string | No | key |
| TA_RDOM | string | No | key |
| TA_LDOM | string | No | key |
| TA_RNAME | string | Yes | |
| TA_CONV | string | NoGW | format: { Y N } |
| TA_ROUTINGNAME | string | Yes | |
| TA_TRANTIME | numeric | Yes | |

CONFIGURING THE DM_ROUTING SECTION

The following table lists the fields in the *DM_ROUTING section.

Table B-7 *DM_ROUTING SECTION

| Field Identifier | Field Type | Update | Notes |
|------------------|------------|--------|-------|
| TA_ROUTINGNAME | string | No | key |
| TA_FIELD | string | Yes | |
| TA_RANGE | string | Yes | |
| TA_BUFTYPE | string | Yes | |

CONFIGURING THE DM_ACCESS_CONTROL SECTION

The following table lists the fields in the *DM_ACCESS_CONTROL section.

Table B-8 *DM_ACCESS_CONTROL SECTION

| Field Identifier | Field Type | Update | Notes |
|------------------|------------|--------|-------|
| TA_ACLNAME | string | No | key |
| TA_RDOM | string | Yes | |

CONFIGURING THE DM_PASSWORDS SECTION

The following table lists the fields in the *DM_PASSWORDS section.

Table B-9 *DM_PASSWORDS SECTION

| Field Identifier | Field Type | Update | Notes |
|------------------|------------|--------|-----------------------|
| TA_LDOM | string | No | key |
| TA_RDOM | string | No | key |
| TA_LPWD | string | Yes | format: { Y N U } |
| TA_RPWD | string | Yes | format: { Y N U } |

The TA_LPWD and TA_RPWD show the existence of a defined password for the local and/or the remote domain. Passwords are not displayed. If an UPDATE operation is selected, the value of the corresponding field must be set to U. The program will then prompt with echo turned off for the corresponding passwords.

DIAGNOSTICS IN CONFIGURATION MODE

dmadmin fails if it cannot allocate an FML typed buffer, if it cannot determine the /etc/passwd entry for the user, or if it cannot reset the environment variables FIELDTBLS or FLDTBLDIR.

The return value printed by dmadmin after each operation completes indicates the status of the requested operation. There are three classes of return values.

The following return values indicate a problem with permissions or a TUXEDO System/T communications error. They indicate that the operation did not complete successfully.

[TAEPERM]

The calling process specified an ADD, UPDATE, or DELETE operation but it is not running as the System/T administrator. Update operations must be run by the administrator (that is, the user specified in the UID attribute of the RESOURCES section of the TUXCONFIG file).

[TAESYSTEM]

A TUXEDO System/T error has occurred. The exact nature of the error is written to `userlog`.

[TAEOS]

An operating system error has occurred.

[TAETIME]

A blocking timeout occurred. The input buffer is not updated so no information is returned for retrieval operations. The status of update operations can be checked by doing a retrieval on the record that was being updated.

The following return values indicate a problem in doing the operation itself and generally are semantic problems with the application data in the input buffer. The string field `TA_STATUS` will be set in the output buffer and will contain short text describing the problem. The string field `TA_BADFLDNAME` will be set to the field name for the field containing the value that caused the problem (assuming the error can be attributed to a single field).

[TAECONFIG]

An error occurred while reading the `BDMCONFIG` file.

[TAEDUPLICATE]

The operation attempted to add a duplicate record.

[TAEINCONSIS]

A field value or set of field values are inconsistently specified.

[TAENOTFOUND]

The record specified for the operation was not found.

[TAENOSPACE]

The operation attempted to do an update but there was not enough space in the BDMCONFIG file.

[TAERANGE]

A field value is out of range or is invalid.

[TAEREQUIRED]

A field value is required but not present.

[TAESIZE]

A field value for a string field is too long.

[TAEUPDATE]

The operation attempted to do an update that is not allowed.

The following return values indicate that the operation was successful.

[TAOK]

The operation succeeded. No updates were done to the BDMCONFIG file.

[TAUPDATED]

The operation succeeded. Updates were made to the BDMCONFIG file.

When using `dmunloadcf` to print entries in the configuration, optional field values are not printed if they are not set (for strings) or 0 (for integers). These fields will always appear in the output buffer when using `dmadmin`. In this way, it makes it easier for the administrator to retrieve an entry and update a field that previously was not set. The entry will have the field name followed by a tab but no field value.

CONFIGURATION EXAMPLE

In the following example, `dmadmin` is used to add a new remote domain. For illustration purposes, `ed` is used for the editor.

```
$ EDITOR=ed dmadmin
> config
Sections:
    1) LOCAL_DOMAINS          2) REMOTE_DOMAINS
    3) LOCAL_SERVICES         4) REMOTE_SERVICES
    5) ROUTING                 6) ACCESS_CONTROL
    7) PASSWORDS              8) TDOMAIN
    9) OSITP                  10) SNA
   11) QUIT
```

```

Enter Section [1]: 2
Operations:
    1) FIRST                2) NEXT
    3) RETRIEVE             4) ADD
    5) UPDATE               6) DELETE
    7) NEW_SECTION          8) QUIT
Enter Operation [1]: 4
Enter editor to add/modify fields [n]? y
a
TA_RDOM                    B05
TA_DOMAINID                BA.BANK05
TA_TYPE                    TDOMAIN
w
53
q
Perform operation [y]? <return>
Return value TAUPDATED
Buffer contents:
TA_OPERATION                4
TA_SECTION                  2
TA_DOMAINID                 BA.BANK05
TA_RDOM                     B05
TA_TYPE                     TDOMAIN
TA_STATUS                   Update completed successfully
Operations:
    1) FIRST                2) NEXT
    3) RETRIEVE             4) ADD
    5) UPDATE               6) DELETE
    7) NEW_SECTION          8) QUIT
Enter Operation [4]: 7
Sections:
    1) LOCAL_DOMAINS        2) REMOTE_DOMAINS
    3) LOCAL_SERVICES        4) REMOTE_SERVICES
    5) ROUTING               6) ACCESS_CONTROL
    7) PASSWORDS             8) TDOMAIN
    9) OSITP                 10) QUIT
Enter Section [1]: 8
Operations:
    1) FIRST                2) NEXT
    3) RETRIEVE             4) ADD
    5) UPDATE               6) DELETE

```

```

7) NEW_SECTION          8) QUIT
Enter Operation [6]: 4
Enter editor to add/modify fields [n]? y
a
TA_RDOM                  B05
TA_NWADDR                0x00020401c0066d05
TA_NWDEVICE              /dev/tcp
w
55
q
Perform operation [y]? <return>
Return value TAUPDATED
Buffer contents:
TA_OPERATION             4
TA_SECTION               8
TA_RDOM                  B05
TA_NWADDR                0x00020401c0066d05
TA_NWDEVICE              /dev/tcp
TA_STATUS                Update completed successfully
Operations:
1) FIRST                 2) NEXT
3) RETRIEVE              4) ADD
5) UPDATE                6) DELETE
7) NEW_SECTION           8) QUIT
Enter Operation [4]: 8
> quit
The dmadmin program ends.
```

SECURITY

If `dmadmin` is run with the application administrator's UID, it assumes a trusted user and Security is bypassed. If `dmadmin` is run with another user ID, and if the security option is enabled in the TUXCONFIG file, then the corresponding application password is required to start the `dmadmin` program. If standard input is a terminal, then `dmadmin` will prompt the user for the password with echo turned off. If standard input is not a terminal, the password is retrieved from the environment variable, `APP_PW`. If this environment variable is not specified and an application password is required, then `dmadmin` will fail to start.

When running with another user ID (other than the UID of the administrator) only a limited set of commands is available.

ENVIRONMENT VARIABLES

`dmadmin` resets the `FIELDTBLS` and `FLDTBLDIR` environment variables to pick up the `${TUXDIR}/udataobj/dmadmin` field table. Hence, the `TUXDIR` environment variable should be set correctly.

If the application requires security and the standard input to `dmadmin` is not from a terminal, then the `APP_PW` environment variable must be set to the corresponding application password.

The `TUXCONFIG` environment variable should be set to the pathname of the TUXEDO System/T configuration file.

GENERAL DIAGNOSTICS

If the `dmadmin` command is entered before the system has been booted, the following message is displayed:

```
No bulletin board exists. Only logging commands are available.
```

`dmadmin` then prompts for the corresponding commands.

If an incorrect application password is entered or is not available to a shell script through the environment, then a log message is generated, the following message is displayed, and the command terminates:

```
Invalid password entered.
```

INTEROPERABILITY

`dmadmin` for OSI must be installed on TUXEDO System/T R6.4. Other nodes in the same domain with an R6.1 gateway may be TUXEDO System/T R4.2.2 or later.

PORTABILITY

`dmadmin` for OSI is supported as a TUXEDO System/T-supplied administrative tool on UNIX operating systems only.

SEE ALSO

`dmloadcf`, `tadmin`, `dmconfig`, `DMADM`, `addusr`, `delusr`

TUXEDO /Domain Guide

dmconfig

TUXEDO System/T ASCII domain configuration file

DESCRIPTION

`dmconfig` is the ASCII version of a TUXEDO System/Domain domain configuration file; it is also referred to by its environmental variable name: `DMCONFIG`. The `dmconfig` file is parsed and loaded into a binary version by the `dmloadcf` utility. The binary configuration file, called the `BDMCONFIG` file, contains information used by domain gateways to initialize the context required for communications with other domains. `dmadmin` uses the binary file (or a copy of it) in its monitoring activity. There will be one `BDMCONFIG` file for each TUXEDO System/Domain application that uses the `/Domain` feature.

A `DMCONFIG` file, and its binary `BDMCONFIG` counterpart, are analogous to the `UBBCONFIG` and `TUXCONFIG` files of a non-`/Domain` System/T application. The `DMCONFIG` file extends the definition of a non-`/Domain` System/T application so that the application becomes a domain.

Definitions

A TUXEDO System/Domain Application is defined as the environment described in a single `TUXCONFIG` file. A System/T Application can communicate with another System/T Application or with another TP Application via a domain gateway group. In “TUXEDO System/Domain” terms, an Application is the same as a TP Domain.

A Gateway Group is a collection of domain gateway processes that provide communication services with a specific type of TP Domain.

A Domain Gateway is a TUXEDO System/Domain process that relays requests and replies to another TP Domain.

A Local Domain characterizes a part of the application (set or subset of services) that is made available to other domains. A Local Domain is always represented by a Domain Gateway Group, and both terms are used as synonyms.

A Remote Domain is a remote application that is accessed through a Gateway Group. The remote application may be another TUXEDO System/Domain application or an application running under another TP system.

A Remote Service is a service provided by a remote domain that is made available to the local application through a Gateway Group.

A `Local Service` is a service of a local domain that is made available to remote domains through a Gateway Group.

Configuration File Format

The format of a domain configuration file is as follows:

- ◆ The file is made up of eight possible specification sections. Lines beginning with an asterisk (*) indicate the beginning of a specification section. Each such line contains the name of the section immediately following the *. Allowable section names are: `DM_LOCAL_DOMAINS`, `DM_REMOTE_DOMAINS`, `DM_LOCAL_SERVICES`, `DM_REMOTE_SERVICES`, `DM_ROUTING`, `DM_ACCESS_CONTROL` and `DM_domtype`, where *domtype* is `OSITP` or `TDOMAIN`. The `DM_LOCAL_DOMAINS` section must precede the `DM_REMOTE_DOMAINS` section.
- ◆ Parameters are generally specified by: *KEYWORD* = *value*. This sets *KEYWORD* to *value*. Valid keywords are described in the following sections. *KEYWORDS* are reserved; they can not be used as *values* unless they are quoted.

Lines beginning with the reserved word, `DEFAULT:`, contain parameter specifications that apply to any lines that follow them in the section in which they appear. Default specifications can be used in all sections. They can appear more than once in the same section. The format for these lines is:

```
DEFAULT: [ KEYWORD1 = value1 [ KEYWORD2 = value2 [...]] ]
```

The values set on this line remain in effect until reset by another `DEFAULT:` line, or until the end of the section is reached. These values can also be overridden on non-`DEFAULT:` lines by placing the optional parameter setting on the line. If on a non-`DEFAULT:` line, the parameter setting is valid for that line only; lines that follow revert to the default setting. If `DEFAULT:` appears on a line by itself, all previously set defaults are cleared and their values revert to the system defaults.

If a value is *numeric*, standard C notation is used to denote the base (that is, `0x` prefix for base 16 (hexadecimal), `0` prefix for base 8 (octal), and no prefix for base 10 (decimal)). The range of values acceptable for a numeric parameter are given under the description of that parameter.

If a value is an *identifier*, standard C rules are used. An *identifier* must start with an alphabetic character or underscore and contain only alphanumeric characters or underscores. The maximum allowable length of an identifier is 30 (not including the terminating null). An identifier cannot be the same as any *KEYWORD*.

A value that is neither an integer number or an identifier must be enclosed in double quotes. Certain special characters can be escaped inside a string using a backslash. “\\” translates to a single backslash. “\”\”” translates to a double quote. “\n” translates to a newline. “\t” translates to a tab. “\f” translates to a formfeed. “\x” (where ‘x’ is any character other than one of the previously mentioned special characters) translates to ‘x’.

- ◆ Input fields are separated by at least one space (or tab) character.
- ◆ “#” introduces a comment. A newline ends a comment.
- ◆ Blank lines and comments are ignored.
- ◆ Comments can be freely attached to the end of any line.
- ◆ Lines are continued by placing at least one tab after the newline. Comments can not be continued.

VERSION=*string_value*

where *string_value* can be any value. The field is not checked by the software; it is provided simply as a place where the customer can enter a string that may have some documentation value to the application.

The DM_LOCAL_DOMAINS Section

This section identifies local domains and their associated gateway groups. The section must have an entry for each gateway group (Local Domain). Each entry specifies the parameters required for the domain gateway processes running in that group.

Entries have the form:

LDOM required parameters [optional parameters]

where *LDOM* is an *identifier* value used to name each local domain. *LDOM* must be unique within a particular configuration. As you will see in the description of the *DM_LOCAL_SERVICES section, *LDOM* is the identifier that connects local services with a particular gateway group.

The following are the required parameters:

GWGRP = *identifier*

specifies the name of the gateway server group (the name provided in the TUXCONFIG file) representing this local domain. There is a one-to-one relationship between a *DOMAINID* and the name of the gateway server group, that is, each GWGRP must have its own, unique *DOMAINID*.

`TYPE = identifier`

is used for grouping local domain into classes. `TYPE` can be set to one of the following values: `TDOMAIN` or `OSITP`. The `TDOMAIN` value indicates that this local domain can only communicate with another TUXEDO System/Domain. The `OSITP` value indicates that this local domain communicates with another TP Domain via the OSI-TP protocol. Domain types must be defined in the `$TUXDIR/udataobj/DMTYPE` file.

`DOMAINID = string`

is used to identify the local domain. `DOMAINID` must be unique across both local and remote domains. The value of *string* can be a sequence of characters (for example, "BA.CENTRAL01"), or a sequence of hexadecimal digits preceded by "0x" (for example, "0x0002FF98C0000B9D6"). `DOMAINID` must be 32 octets or fewer in length. If the value is a string, it must be 32 characters or fewer (counting the trailing null).

`DMTLOGDEV = string`

specifies the TUXEDO filesystem that contains the Domain transaction log (`DMTLOG`) for this machine. The `DMTLOG` is stored as a TUXEDO System VTOC table on the device. If this parameter is not specified (and it should not be specified if `TYPE=SNADOM`), the domain gateway group is not allowed to process requests in transaction mode. Local domains running on the same machine can share the same `DMTLOGDEV` filesystem, but each local domain must have its own log (a table in the `DMTLOGDEV`) named as specified by the `DMTLOGNAME` keyword.

Optional parameters describe resources and limits used in the operation of domain gateways:

`AUDITLOG = string`

specifies the name of the audit log file for this local domain. The audit log feature is activated from the `dmadmin` command and records all the operations within this local domain. If the audit log feature is active and this parameter is not specified, the file `DMmmddyy.LOG` (where `mm`=month, `dd`=day, and `yy`=year) is created in the directory specified by the `$APPDIR` environment variable or the `APPDIR` keyword of the `*MACHINES` section of the `TUXCONFIG` file.

`BLOCKTIME = numeric`

specifies the maximum wait time allowed for a blocking call. The value sets a multiplier of the `SCANUNIT` parameters specified in the `TUXCONFIG` file. The value `SCANUNIT * BLOCKTIME` must be greater than or equal to `SCANUNIT` and less than 32,768 seconds. If this parameter is not specified, the default

value is set to the value of the `BLOCKTIME` parameter specified in the `TUXCONFIG` file. A timeout always implies a failure of the affected request. Notice that the timeout specified for transactions in the `TUXCONFIG` will always be used when the request is issued within a transaction.

`DMTLOGNAME = identifier`

specifies the name of the domain transaction log for this domain. This name must be unique when the same `DMTLOGDEV` is used for several local domains. If not specified, the default is the string “`DMTLOG`”. The name must be 30 characters or less.

`DMTLOGSIZE = numeric`

specifies the numeric size, in pages, of the Domain transaction log for this machine. It must be greater than 0 and less than the amount of available space on the `TUXEDO` filesystem. If not specified, the default is 100 pages.

`MAXDATALEN = numeric`

specifies a maximum amount of data (in bytes) that can be sent to or from any services advertised by this local domain. There is no limit if this parameter is not specified.

`MAXRDOM = numeric`

specifies the maximum number of connections (or dialogues if the domain is of type `OSITP`) allowed per gateway. There is no limit if this parameter is not specified.

`MAXRDTRAN = numeric`

specifies the maximum number of domains that can be involved in a transaction. It must be greater than 0 and less than 32,768. If not specified, the default is 16.

`MAXTRAN = numeric`

specifies the maximum number of simultaneous global transactions allowed on this local domain. It must be greater than or equal to 0 and less than or equal to the `MAXGTT` parameter specified in the `TUXCONFIG` file. If not specified, the default is the value of `MAXGTT`.

`MAXSENDLEN = numeric`

specifies the maximum length (in bytes) of messages sent or received by this local domain. If this parameter is set all messages sent or received will be broken up into packets of no more than `MAXSENDLEN` bytes. There is no limit if this parameter is not specified.

`SECURITY = value`

specifies the type of application security to be enforced. The following description applies to security in an /SNA Domain.

The combined setting of the `SECURITY` parameter in the `UBBCONFIG` and the `DMCONFIG` file have the following effects:

When the `DMCONFIG SECURITY` parameter is set to `NONE` or `APP_PW`, no action is taken by the /SNA Domain gateway with regard to security.

However, when the `UBBCONFIG` file is configured with `SECURITY` set to `APP_PW`, `dmloadcf` and `dmunloadcf` will require entering of an application password. The application password is validated by an `AUTHSVC` when clients join the application. The `AUTHSVC` is provided by the user application.

If security is to be enforced for each request inbound from CICS to a local /TDOMAIN or outbound to CICS from a local /TDOMAIN, the `UBBCONFIG` file `SECURITY` parameter must be set to one of: `USER_AUTH`, `ACL`, or `MANDATORY_ACL` and the `DMCONFIG` file `SECURITY` parameter must be set to `DM_USER_PW`.

When the `DMCONFIG SECURITY` parameter is set to `USER_PW`, and the `UBBCONFIG` file `SECURITY` parameter is set to `USER_AUTH`, `ACL`, or `MANDATORY_ACL` the following applies:

When a request is being received from CICS the domain access control list in the `DMCONFIG` file for the local service (if present) is checked to see if requests from the remote domain are permitted. If the `DMCONFIG` file does not contain an access control list for the local service, the service is accessible.

If access is permitted, the remote userid is extracted from the conversation startup request from CICS. The remote userid is used to find the local userid and password in the domain security table. This local userid and password will be used for subsequent `ACL` or `MANDATORY_ACL` security checking as and if specified in the `UBBCONFIG` file.

(IMPORTANT: If the application has set `UBBCONFIG SECURITY` to `USER_AUTH`, and is using its own `AUTHSVC`, the least significant 17 bits of the local application password must be able to be stored and subsequently used as a local userid for mapping to remote usernames and passwords in the domain security table.)

When the `DMCONFIG SECURITY` parameter is set to `DM_PW`, and the `UBBCONFIG` file is set to `USER_AUTH`, `ACL`, or `MANDATORY_ACL`, the following applies:

For requests sent from CICS, the remote userid is extracted from the conversation startup request from CICS. If a remote password is also available in the startup request, it is checked. The remote userid is mapped to the remote `DomainID` and password. The remote `DomainID` are used for further `ACL` checking.

The domain administrator can add or delete users from the domain security table. Local userids and passwords corresponding to remote userids and passwords are entered into the domain security table in two ways: first, through data entry panels in the TUXEDO System administration Graphical Administrative Interface, and second, by using the `dmadmin` command `dmusradd`.

The `DM_REMOTE_DOMAINS` Section

This section identifies the known set of remote domains and their characteristics.

Entries have the form:

RDOM *required parameters*

where *RDOM* is an *identifier* value used to identify each remote domain known to this configuration. *RDOM* must be unique within the configuration.

The following parameters are required:

`TYPE` = *identifier*

is used for grouping remote domain into classes. `TYPE` can be set to one of the following values: `TDOMAIN`, `OSITP` or `SNA`. The `TDOMAIN` value indicates that this remote domain can only communicate with another TUXEDO System/Domain Domain. The `OSITP` value indicates that this remote domain communicates with another TP domain via the OSI-TP protocol.

`DOMAINID` = *string*

is used to identify a remote domain. `DOMAINID` must be 32 octets or fewer in length. If the value is a string, it must be 32 characters or fewer (counting the trailing null). `DOMAINID` must be unique across remote domains. The value of *string* can be a sequence of characters or a sequence of hexadecimal digits preceded by “0x”.

There are no optional parameters for this section.

The DM_TDOMAIN Section

This section defines the addressing information required by domains of type `TDOMAIN`. This section should have an entry per local domain if requests from remote domains to local services are accepted on that local domain (gateway group), and an entry per remote domain accessible by the defined local domains.

Entries have the form:

```
DOM    required parameters [optional parameters]
```

where *DOM* is an *identifier* value used to identify either a local domain (*LDOM*) or a remote domain (*RDOM*) in the `*DM_LOCAL_DOMAINS` section or in the `*DM_REMOTE_DOMAINS` section. The *DOM* identifier must match a previously defined *LDOM* in the `*DM_LOCAL_DOMAINS` sections or *RDOM* in the `*DM_REMOTE_DOMAINS` section.

The following parameter is required:

```
NWADDR = string
```

This parameter specifies the network address used by a local or a remote domain to accept connections from other TUXEDO System/Domain Domains. If *string* has the form `'0xhex-digits'`, it must contain an even number of valid hexadecimal digits.

The following parameter is optional:

```
NWDEVICE = string
```

Specifies the device file name to be used when binding to the listening address of a local or a remote domain. If the networking functionality is TLI-based, the device name must be an absolute pathname. If the networking functionality is Sockets-based, this parameter does not need to be specified.

```
NWIDLETIME = numeric
```

Specifies the maximum time allowed for a connection to be idle or unused. When this time is reached, the idle connection is terminated. The numeric value represents a time in minutes. If this keyword is not specified, then idle connections will be maintained until the gateway handling the connection is shutdown.

Notice that multiple entries for a particular domain may be defined in this table. Multiple addresses specified for a remote domain mean that the first address (the first entry in the table for the remote domain) should be used to

establish the connection and the other addresses should be used as back-up addresses in case of failure of the connection setup to the first address. Multiple addresses specified for a local domain mean that multiple listening ports are available on the same or different types of networks.

The DM_OSITP Section

This section defines the addressing information required by domains of type OSITP. This section should have one entry per gateway group (local domain), and one entry per remote domain of type OSITP.

Entries have the form:

```
DOM    required parameters [optional parameters]
```

where *DOM* is an *identifier* value used to identify a local domain (LDM) or a remote domain (RDM) in the *DM_LOCAL_DOMAINS section or in the *DM_REMOTE_DOMAINS section. The *DOM* identifier must match a previously defined *LDM* in the *DM_LOCAL_DOMAINS sections or *RDM* in the *DM_REMOTE_DOMAINS section.

The following are required parameters:

```
APT = string
```

This parameter specifies an OSI Application Process Title (APT). An APT may be a name (i.e., the Directory Name of an Application Process Title) or an object identifier (i.e., a sequence of integer values separated by periods).

```
AEQ = string
```

This parameter specifies an OSI Application Entity Qualifier (AEQ). An AEQ may be a name (i.e., the relative distinguished name of a particular Application Entity) or an integer (i.e., if the APT is an object identifier).

```
NWDEVICE = string
```

Specifies the device name to be used when binding to an XAP dialogue instance. It may be an absolute pathname of a device filename or just an identifier of a device. The value of the string is specific to the XAP-TP provider.

The following are optional parameters:

```
AET = string
```

This parameter specifies an OSI Application Entity Title (AET). An AET is formed from an Application Process Title (APT) and an Application Entity

Qualifier (AEQ), i.e. in ASN.1 AET is defined as a SEQUENCE { APT, AEQ } where APT and AET are of type ANY. Three main formats are accepted for the value of *string*:

encoded string

This is a single value as a hexadecimal octet string which a represents a valid BER encoding of the AET, e.g. AET = “0x06062B80CE0F0107”.

{object identifier}, {integer}

The first element represents the APT defined as an object identifier (i.e., a sequence of integer values separated by periods) and the second element represents an AEQ defined as an integer constant, e.g., AET = “{1.3.15.0.3},{1}”.

{string}, {string}

This format allows the APT and the AEQ to be defined as string constants, e.g., AET = “{BA.CENTRAL01},{TUXEDO}”.

ACN = *{XATMI | UDT}*

This parameter specifies the object identifier of the Application Context Name (ACN) used by this domain. Current allowed application contexts are: the XATMI-ASE (XATMI) and the UDT-ASE (UDT). If this parameter is not specified, the ACN is set to the object identifier of the XATMI-ASE Application Context.

APID = *integer*

This parameter specifies an OSI Application Process Invocation Identifier (APID).

AEID = *integer*

This parameter specifies an OSI Application Entity Invocation Identifier (AEID).

PROFILE = *identifier*

This parameter specifies the OSI TP profile used by this domain and is used to determine the required OSI TP functional units. PROFILE can be set to one of the following values: ATP11, ATP21, ATP31, ATP12, ATP22, and ATP32. The UDT ASE application context allows the use of any of these profiles. The XATMI-ASE application context only allows profiles ATP11, ATP21 and ATP31. Profiles ATP11, ATP21 and ATP31 use the Dialogue, Polarized Control and Handshake functional units. Profiles ATP12, ATP22 and ATP32 use the Dialogue, Shared Control, and Handshake functional units. Profiles ATP11

and ATP12 do not use OSI TP transactions (the Commit functional unit is not used). Profiles ATP21 and ATP22 require the Commit, Unchained Transactions, and Recovery functional units. Profiles ATP31 and ATP32 require the Commit, Chained Transactions, and Recovery functional units. By default, the ATP21 profile is always selected.

URCH = *string*

This parameter specifies the user portion of the OSITP Recovery Context Handle. It may be required by the XAP-TP provider in order to perform recovery of distributed transactions after a communications line or system failure.

The DM_ACCESS_CONTROL Section

This section specifies the access control lists used by local domain. Lines in this section are of the form:

ACL_NAME required parameters

where *ACL_NAME* is a (*identifier*) name used to identify a particular access control list; it must be 15 characters or less in length.

Required parameters are:

ACLIST = *identifier* [,*identifier*]

where an ACLIST is composed of one or more remote domain names (RDOM) separated by commas. The wildcard character (*) can be used to specify that all the remote domains defined in the *DM_REMOTE_DOMAINS section can access a local domain.

The DM_LOCAL_SERVICES Section

This section provides information on the services exported by each local domain. This section is optional and if it is not specified then all local domains defined in the *DM_LOCAL_DOMAINS section accept requests to all of the services advertised by the TUXEDO System/Domain application. If this section is defined then it should be used to restrict the set of local services that can be requested from a remote domain.

Lines within this section have the form:

service [optional parameters]

where *service* is the (*identifier*) local name of the exported service, and it must be 15 characters or fewer in length. This name corresponds to a name advertised by one or more servers running with the local TUXEDO System/Domain application.

Notice that exported services inherit the default or special properties specified for the service in an entry in the SERVICES section of the TUXCONFIG file. Some of these parameters are: LOAD, PRIO, AUTOTRAN, ROUTING, BUFTYPE, and TRANTIME.

Optional parameters are:

ACL = *identifier*

specifies the name of the access control list (ACL) to be used by the local domain to restrict requests made to this service by remote domains. The name of the ACL is defined in the *DM_ACCESS_CONTROL section. If this parameter is not specified then access control will not be performed for requests to this service.

LDOM = *identifier*

specifies the name identifying the local domain exporting this service. If this keyword is not specified then all the local domains defined in the *DM_LOCAL_DOMAINS section will accept requests to this local service.

RNAME = *string*

specifies the name exported to remote domains. This name will be used by the remote domains for request to this service. If this parameter is not specified, the local service name is supposed to be the name used by any remote domain.

The DM_REMOTE_SERVICES Section

This section provides information on services “imported” and available on remote domains. Lines within this *DM_REMOTE_SERVICES section have the form:

```
service    [optional parameters]
```

where *service* is the (*identifier*) name used by the local TUXEDO System/Domain application for a particular remote service. Remote services are associated with a particular remote domain.

Optional Parameters are:

AUTOTRAN= { Y | N }

specifies whether or not a transaction should automatically be started if a request message is received that is not already in transaction mode. The default is No.

BLOCKTIME = *numeric*

specifies the maximum wait time allowed for a reply to this remote service. The value sets a multiplier of the SCANUNIT parameters specified in the TUXCONFIG file. The value SCANUNIT * BLOCKTIME must be greater

than or equal to SCANUNIT and less than 32,768 seconds. A timeout always implies a failure of the affected transaction or request.

CONV = { Y | N }
specifies whether (Y) or not (N) the remote service is a conversational service. The default value is N.

LIDOM = *identifier*
specifies the name of a local domain in charge of routing requests to this remote service. The gateway group associated with the local domain advertises *service* in the TUXEDO System/Domain Bulletin Board. If this parameter is not specified then all the local domains will be able to accept requests to this remote service. The service request will be then redirected to a remote domain of the same type (see RDOM keyword).

LOAD = *integer*
specifies that the remote service imposes a load of integer units. The value of LOAD can be between 1 and 32767 inclusive. If not specified, the default is 50. A higher number indicates a greater load.

PRIO = *integer*
specifies the dequeuing priority of service requests to this remote service. The value of PRIO must be greater than 0 and less than or equal to 100, with 100 being the highest priority. The default is 50.

RDOM = *identifier*
specifies the name of the remote domain responsible for the actual execution of this service. If this parameter is not specified and a routing criteria (see ROUTING keyword) is not specified, then the local domain assumes that any remote domain of the same type accepts this service and it selects a known domain (a domain to which a connection already exists) or remote domain from the `**DM_REMOTE_DOMAINS` section.

RNAME = *string*
specifies the actual service name expected by the remote domain. If this parameter is not specified, the remote service name is the same as the name specified in *service*.

ROUTING = *identifier*
when more than one remote domain offers the same service, a local domain can perform data dependent routing if this optional parameter is specified. The *identifier* specifies the name of the routing criteria used for this data dependent routing. If not specified, data dependent routing is not done for this service. *identifier* must be 15 characters or less in length. If multiple

entries exist for the same service name but with different RDOM parameters, the ROUTING parameter should be the same for all of these entries.

TRANTIME = *integer*

specifies the default timeout value in seconds for a transaction automatically started for the associated service. The value must be greater than or equal to 0 and less than 2147483648. The default is 30 seconds. A value of 0 implies the maximum timeout value for the machine.

The DM_ROUTING Section

This section provides information for data dependent routing of service requests using FML, VIEW, X_C_TYPE, and X_COMMON typed buffers. Lines within the DM_ROUTING section have the form:

CRITERION_NAME required parameters

where *CRITERION_NAME* is the (*identifier*) name of the routing entry that was specified on the services entry. *CRITERION_NAME* must be 15 characters or less in length.

Required parameters are:

FIELD = *identifier*

specifies the name of the routing field. It must be 30 characters or less. This field is assumed to be a field name that is identified in an FML field table (for FML buffers) or an FML view table (for VIEW, X_C_TYPE, or X_COMMON buffers). The FLDTBLDIR and FIELDTBLS environment variables are used to locate FML field tables, and the VIEWDIR and VIEWFILES environment variables are used to locate FML view tables.

RANGES = *string*

specifies the ranges and associated remote domain names (RDOM) for the routing field. *string* must be enclosed in double quotes. The format of *string* is a comma-separated ordered list of range/RDOM pairs (see the following EXAMPLES).

A range is either a single value (signed numeric value or character string in single quotes), or a range of the form “lower - upper” (where lower and upper are both signed numeric values or character strings in single quotes). Note that “lower” must be less than or equal to “upper”. To embed a single quote in a character string value (as in O’Brien, for example), it must be preceded by two backslashes (‘O’Brien’). The value MIN can be used to indicate the minimum value for the data type of the associated FIELD; for strings and

arrays, it is the null string; for character fields, it is 0; for numeric values, it is the minimum numeric value that can be stored in the field. The value `MAX` can be used to indicate the maximum value for the data type of the associated `FIELD`; for strings and arrays, it is effectively an unlimited string of octal-255 characters; for a character field, it is a single octal-255 character; for numeric values, it is the maximum numeric value that can be stored in the field. Thus, “`MIN - -5`” is all numbers less than or equal to `-5` and “`6 - MAX`” is all numbers greater than or equal to `6`. The meta-character “`*`” (wild-card) in the position of a range indicates any values not covered by the other ranges previously seen in the entry; only one wild-card range is allowed per entry and it should be last (ranges following it will be ignored).

The routing field can be of any data type supported in `FML`. A numeric routing field must have numeric range values and a string routing field must have string range values.

String range values for string, array, and character field types must be placed inside a pair of single quotes and can not be preceded by a sign. Short and long integer values are a string of digits, optionally preceded by a plus or minus sign. Floating point numbers are of the form accepted by the C compiler or `atof()`: an optional sign, then a string of digits optionally containing a decimal point, then an optional `e` or `E` followed by an optional sign or space, followed by an integer.

When a field value matches a range, the associated `RDOM` value specifies the remote domain to which the request should be routed. A `RDOM` value of “`*`” indicates that the request can go to any remote domain known by the gateway group.

Within a range/`RDOM` pair, the range is separated from the `RDOM` by a “`:`”.

```
BUFTYPE = ~type1[:subtype1[, subtype2 . . .  
]][:type2[:subtype3[, . . . ]]] . . .~
```

is a list of types and subtypes of data buffers for which this routing entry is valid. The types are restricted to be either `FML`, `VIEW`, `X_C_TYPE`, or `X_COMMON`. No subtype can be specified for type `FML` and subtypes are required for the other types (“`*`” is not allowed). Duplicate type/subtype pairs can not be specified for the same routing criterion name; more than one routing entry can have the same criterion name as long as the type/subtype pairs are unique. This parameter is required. If multiple buffer types are specified for a single routing entry, the data types of the routing field for each buffer type must be the same.

If the field value is not set (for FML buffers), or does not match any specific range and a wild-card range has not been specified, an error is returned to the application process that requested the execution of the remote service.

FILES

The `BDMCONFIG` environment variable is used to find the `BDMCONFIG` configuration file.

EXAMPLE1

The following configuration file defines a 5-site domain configuration. The example shows 4 Bank Branch domains communicating with a Central Bank Branch. Three of the Bank Branches run within other TUXEDO System/Domain domains. The fourth Branch runs under the control of another TP Domain and OSI-TP is used in the communication with that domain.

```
# TUXEDO DOMAIN CONFIGURATION FILE FOR THE CENTRAL BANK
#
#
*DM_LOCAL_DOMAINS
# <local domain name> <Gateway Group name> <domain type> <domain
id> <log device>
#
#           [<audit log>] [<blocktime>]
#           [<log name>] [<log offset>] [<log size>]
#           [<maxrdom>] [<maxrdtran>] [<maxtran>]
#           [<maxdatalen>] [<security>]
#           [<tuxconfig>] [<tuxoffset>]
#
#
DEFAULT: SECURITY = NONE

c01      GWGRP = bankg1
        TYPE = TDOMAIN
        DOMAINID = "BA.CENTRAL01"
        DMTLOGDEV = "/usr/apps/bank/DMTLOG"
        DMTLOGNAME = "DMTLG_C01"

c02      GWGRP = bankg2
        TYPE = OSITP
        DOMAINID = "BA.CENTRAL01"
        DMTLOGDEV = "/usr/apps/bank/DMTLOG"
        DMTLOGNAME = "DMTLG_C02"
```

```

NWDEVICE = "OSITP"
URCH = "ABCD"

#
*DM_REMOTE_DOMAINS
#<remote domain name> <domain type> <domain id>
#
b01    TYPE = TDOMAIN
        DOMAINID = "BA.BANK01"

b02    TYPE = TDOMAIN
        DOMAINID = "BA.BANK02"

b03    TYPE = TDOMAIN
        DOMAINID = "BA.BANK03"

b04    TYPE = OSITP
        DOMAINID = "BA.BANK04"
        NWDEVICE = "/dev/osi"
        URCH = "ABCD"

*DM_TDOMAIN
#
# <local or remote domain name>    <network address> [<nwdevice>]
#
# Local network addresses
c01    NWADDR = "0x0002ff98c00b9d6d"  NWDEVICE = "/dev/tcp"
c01    NWADDR = "newyork01.65432"      NWDEVICE = "/dev/starlan"
# Remote network addresses
b01    NWADDR = "0x00020401c00b6d05"  NWDEVICE = "/dev/tcp"
b02    NWADDR = "dallas.65432"        NWDEVICE = "/dev/starlan"
b03    NWADDR = "0x00021094c00b6d9c"  NWDEVICE = "/dev/tcp"

*DM_OSITP
#
#<local or remote domain name> <apt> <aeq>
#
#                                     [<aet>] [<acn>] [<apid>] [<aeid>]
#                                     [<profile>]
#
c02    APT = "BA.CENTRAL01"
        AEQ = "TUXEDO.R.4.2.1"
```

```

        AET = "{1.3.15.0.3},{1}"
        ACN = "XATMI"

b04    APT = "BA.BANK04"
        AEQ = "TUXEDO.R.4.2.1"
        AET = "{1.3.15.0.4},{1}"
        ACN = "XATMI"

*DM_LOCAL_SERVICES
#<service_name>  [<Local Domain name>] [<access control>]
[<exported svcname>]
#
#
open_actACL = branch
close_actACL = branch
credit
debit
balance
loan          LDOM = c02ACL = loans

*DM_REMOTE_SERVICES
#<service_name>  [<Remote domain name>] [<local domain name>]
#                [<remote svcname>] [<routing>] [<conv>] [<trantime>]
#
#
tlr_add    LDOM = c01  ROUTING = ACCOUNT
tlr_bal    LDOM = c01  ROUTING = ACCOUNT
tlr_add    RDOM = b04  LDOM = c02 RNAME ="TPSU002"
tlr_bal    RDOM = b04  LDOM = c02 RNAME ="TPSU003"

*DM_ROUTING
# <routing criteria><field> <typed buffer> <ranges>
#
ACCOUNTFIELD = branchid  BUFTYPE ="VIEW:account"
        RANGES ="MIN - 1000:b01, 1001-3000:b02, *:b03"

*DM_ACCESS_CONTROL
#<acl name>      <Remote domain list>
#

```

```
branchACLIST = b01, b02, b03
loans ACLIST = b04
```

EXAMPLE 2

This example shows the TUXEDO System/Domain Domain Configuration file required at one of the Bank Branches (BANK01).

```
#
#TUXEDO DOMAIN CONFIGURATION FILE FOR A BANK BRANCH
#
#
*DM_LOCAL_DOMAINS
#
b01      GWGRP = auth
         TYPE = TDOMAIN
         DOMAINID = "BA.BANK01"
         DMTLOGDEV = "/usr/apps/bank/DMTLOG"

*DM_REMOTE_DOMAINS
#
c01      TYPE = TDOMAIN
         DOMAINID = "BA.CENTRAL01"

*DM_TDOMAIN
#
b01      NWADDR = "0x00021094c00b689c"  NWDEVICE = "/dev/tcp"
c01      NWADDR = "0x0002ff98c00b9d6d"  NWDEVICE = "/dev/tcp"
*DM_LOCAL_SERVICES
#
tlr_add      ACL = central
tlr_bal      ACL = central

*DM_REMOTE_SERVICES
#
OPA001      RNAME = "open_act"
CLA001      RNAME = "close_act"
CRD001      RNAME = "credit"
DBT001      RNAME = "debit"
BAL001      RNAME = "balance"
```

```
*DM_ACCESS_CONTROL
#
central          ACLIST = c01
```

EXAMPLE 3

This example shows the configuration file entries for a BEA Connect SNA application:

```
# TUXEDO DOMAIN CONFIGURATION FILE SAMPLE with BEA Connect SNA
#
*DM_LOCAL_DOMAINS
#<local domain name> <Gateway Group Name> <domain type> <domain id>
<log device>
#
#          [<audit log>] [<blocktime>]
#          [<log name>] [<logoffset>] [<logsize>]
#          [<maxrdomain>] [<maxrdtran>] [<maxtran>]
#          [<maxsendlen>] [<security>]
#          [<tuxconfig>] [<tuxoffset>]
#          [<luname>] {<modename>} [<netid>]
#
#
DEFAULT: SECURITY = NONE

ldom1 GWGRP=bankgrp1
      TYPE=SNADOM
      DOMAINID="TELLER"
      SECURITY=NONE

ldom2 GWGRP=bankgrp2
      TYPE=SNADOM
      DOMAINID="LOANS"

#
*DM_REMOTE_DOMAINS
#
#<remote domain name> <domain type> <domain id>
sna1      TYPE=SNADOM
          DOMAINID=MC

sna2      TYPE=SNADOM
```

```
DOMAINID=VISA
#
*DM_SNADOM
#
# <remote domain name> <luname><netid><modename>
<localLuname><credfile>
#
ldom1    LUNAME="L0F0024A"
         NETID="SNANET1"
         MODENAME="SNA62"
         SECURITY_TYPE="LOCAL"
         SYMDESTNAME="CICSSYSN"
         MAXSNASESS=10

ldom2    LUNAME="L0F0024B"
         NETID="SNANET1"
         MODENAME="SNA62"
         SECURITY_TYPE="LOCAL"
         SYMDESTNAME="CICSSYSN"
         MAXSNASESS=10

sna1     LUNAME="CICSSYSN"
         NETID="SNANET1"
         MODENAME="SNA62"
         SECURITY_TYPE="LOCAL"
         SYMDESTNAME="L0F0024A"

sna2     LUNAME="CICSSYSN"
         NETID="SNANET1"
         MODENAME="SNA62"
         SYMDESTNAME="L0F0024B"

*DM_LOCAL_SERVICES
# <service_name> [<Local Domain name> [<access control>] [<exported
svcname>]
#
#           [<inbuftype>] [<outbuftype>]
#
debit    LDOM=ldom1
         INBUFTYPE=VIEW:cust
         OUTBUFTYPE=VIEW:bill
         RNAME="DEBIT"
```

```

credit          LDOM=ldom1
                INBUFTYPE=VIEW:cust
                OUTBUFTYPE=VIEW:refund
                RNAME="CREDIT"
active          LDOM=ldom2
                INBUFTYPE=STRING
                OUTBUFTYPE=VIEW:cust
                RNAME="ACTIVE"
inactive        LDOM=ldom2
                INBUFTYPE=STRING
                OUTBUFTYPE=VIEW:cust
                RNAME="INACTIVE"

*DM_REMOTE_SERVICES
# <service_name> [<Remote domain name>] [<local domain name>]
#           [<remote svcname>] [<routingname>] [<conv>] [<blocktime>]
#           [<prio>] [<load>] [<autran>] [<trantime>]
#           [<inbuftype>] [<outbuftype>]
#
open            RDOM=sna1          ROUTING=ACCOUNT  RNAME="ACCTOPEN"
close           RDOM=sna1          ROUTING=ACCOUNT  RNAME="ACCTCLSE"
open            RDOM=sna2 LDOM=ldom2      RNAME="ACCTOPEN"
close           RDOM=sna2 LDOM=ldom2      RNAME="ACCTCLSE"

*DM_ROUTING
# <routing criteria> <field> <typed buffer> <ranges>
#
ACCOUNT        FIELD=branchid BUFTYPE="VIEW:account"
                RANGES="MIN - 1000:sna1, *:sna2"
#

```

SEE ALSO

build_dgw, dmadmin, tmboot, tmshutdown, dmloadcf, dmunloadcf
 dmgwopts, GWADM, DMADM

TUXEDO /Domain Guide

TUXEDO Administrator's Guide

TUXEDO Programmer's Guide

dmloadcf

Parse a DMCONFIG file and load binary BDMCONFIG configuration file

SYNOPSIS

```
dmloadcf [-c] [-n] [-y] [-b blocks] {dmconfig_file | - }
```

DESCRIPTION

`dmloadcf` reads a file or the standard input that is in DMCONFIG syntax, checks the syntax, and optionally loads a binary BDMCONFIG configuration file. The BDMCONFIG environment variable points to the path name of the BDMCONFIG file where the information should be stored.

`dmloadcf` prints an error message if it finds any required section of the DMCONFIG file missing. If a syntax error is found while parsing the input file, `dmloadcf` exits without performing any updates to the BDMCONFIG file.

`dmloadcf` requires the existence of the \$TUXDIR/udataobj/DMTYPE file. This file defines the valid domain types. If this file does not exist, `dmloadcf` exits without performing any updates to the BDMCONFIG file.

The effective user identifier of the person running `dmloadcf` must match the UID in the RESOURCES section of the TUXCONFIG file.

The `-c` option to `dmloadcf` causes the program to print minimum IPC resources needed for each local domain (gateway group) in this configuration. The BDMCONFIG file is not updated.

The `-n` option to `dmloadcf` causes the program to do only syntax checking of the ASCII DMCONFIG file without actually updating the BDMCONFIG file.

After syntax checking, `dmloadcf` checks to see if the file pointed to by BDMCONFIG exists, is a valid TUXEDO System file system, and contains BDMCONFIG tables. If these conditions are not true, the user is prompted to create and initialize the file with

```
Initialize BDMCONFIG file: path [y, q]?
```

where *path* is the complete file name of the BDMCONFIG file. Prompting is suppressed if the standard input or output are not terminals, or if the `-y` option is specified on the command line. Any response other than “y” or “Y” will cause `dmloadcf` to exit without creating the configuration file.

If the `BDMCONFIG` file is not properly initialized, and the user has given the go-ahead, `dmloadcf` creates the TUXEDO file system and then creates the `BDMCONFIG` tables. If the `-b` option is specified on the command line, its argument is used as the number of blocks for the device when creating the TUXEDO file system. If the value of the `-b` option is large enough to hold the new `BDMCONFIG` tables, `dmloadcf` will use the specified value to create the new file system; otherwise, `dmloadcf` will print an error message and exit. If the `-b` option is not specified, `dmloadcf` will create a new file system large enough to hold the `BDMCONFIG` tables. The `-b` option is ignored if the file system already exists. The `-b` option is highly recommended if `BDMCONFIG` is a raw device (that has not been initialized) and should be set to the number of blocks on the raw device. The `-b` option is not recommended if `BDMCONFIG` is a regular UNIX file.

If the `BDMCONFIG` file is determined to already have been initialized, `dmloadcf` ensures that the local domain described by that `BDMCONFIG` file is not running. If a local domain is running, `dmloadcf` prints an error message and exits. Otherwise, `dmloadcf`, to confirm that the file should be overwritten, prompts the user with:

```
"Really overwrite BDMCONFIG file [y, q]?"
```

Prompting is suppressed if the standard input or output are not a terminal or if the `-y` option is specified on the command line. Any response other than "y" or "Y" will cause `dmloadcf` to exit without overwriting the file.

If the `SECURITY` parameter is specified in the `RESOURCES` section of the `TUXCONFIG` file, then `dmloadcf` will flush the standard input, turn off terminal echo and prompt the user for an application password as follows:

```
Enter Application Password?
```

The password is truncated to 8 characters. The option to load the ASCII `DMCONFIG` file via the standard input (rather than a file) cannot be used when this `SECURITY` parameter is turned on. If the standard input is not a terminal, that is, if the user cannot be prompted for a password (as with a `here` file, for example), then the environment variable `APP_PW` is accessed to set the application password. If the environment variable `APP_PW` is not set with the standard input not a terminal, then `dmloadcf` will print an error message, generate a log message and fail to load the `BDMCONFIG` file.

Assuming no errors, and if all checks have passed, `dmloadcf` loads the `DMCONFIG` file into the `BDMCONFIG` file. It will overwrite all existing information found in the `BDMCONFIG` tables.

PORTABILITY

`dmloadcf` is supported as a TUXEDO-supplied administrative tool on UNIX operating systems only.

ENVIRONMENT VARIABLES

The environment variable `APP_PW` must be set for applications that require security (the `SECURITY` parameter in the `TUXCONFIG` file is set to `APP_PW`) and `dmloadcf` is run with something other than a terminal as the standard input.

The `BDMCONFIG` environment variable should point to the `BDMCONFIG` file.

EXAMPLES

The following example shows how a binary configuration file is loaded from the `bank.dmconfig` ASCII file. The `BDMCONFIG` device is created (or reinitialized) with 2000 blocks:

```
dmloadcf -b 2000 -y bank.dmconfig
```

DIAGNOSTICS

If an error is detected in the input, the offending line is printed to standard error along with a message indicating the problem. If a syntax error is found in the `DMCONFIG` file or the system is currently running, no information is updated in the `BDMCONFIG` file and `dmloadcf` exits with exit code 1.

If `dmloadcf` is run on an active node, the following error message is displayed:

```
*** dmloadcf cannot run on an active node ***
```

If `dmloadcf` is run by a person whose effective user identifier doesn't match the `UID` specified in the `TUXCONFIG` file, the following error message is displayed:

```
*** UID is not effective user ID ***
```

Upon successful completion, `dmloadcf` exits with exit code 0. If the `BDMCONFIG` file is updated, a `userlog` message is generated to record this event.

SEE ALSO

`dmunloadcf`, `dmconfig`, `ubbconfig`

TUXEDO /Domain Guide

TUXEDO Administrator's Guide

dmunloadcf

Unload binary BDMCONFIG domain configuration file

SYNOPSIS

`dmunloadcf`

DESCRIPTION

`dmunloadcf` translates the BDMCONFIG configuration file from the binary representation into ASCII. This translation is useful for transporting the file in a compact way between machines with different byte orderings and backing up a copy of the file in a compact form for reliability. The ASCII format is the same as is described in `dmconfig`.

`dmunloadcf` reads values from the BDMCONFIG file pointed to by the BDMCONFIG environment variable and writes them to its standard output.

PORTABILITY

`dmunloadcf` is supported as a TUXEDO-supplied administrative tool on UNIX operating systems only.

EXAMPLES

To unload the configuration in `/usr/tuxedo/BDMCONFIG` into the file `bdmconfig.backup`:

```
BDMCONFIG=/usr/tuxedo/BDMCONFIG dmunloadcf > bdmconfig.backup
```

DIAGNOSTICS

`dmunloadcf` checks that the file pointed to by the BDMCONFIG environment variable exists, is a valid TUXEDO file system, and contains BDMCONFIG tables. If any of these conditions is not met, `dmunloadcf` prints an error message and exits with error code 1. Upon successful completion, `dmunloadcf` exits with exit code 0.

SEE ALSO

`dmloadcf`, `dmconfig`

TUXEDO /Domain Guide

dmusradd

Add a user to the domain password file

SYNOPSIS

```
DMCONFIG=dmconfig
dmusradd [ -ld local domain ID ][ -rd remote domain ID ]
[ -lu local uid ] [ -lp local password ]
[ -ru remote username ] [ -rp remote password ]
```

DESCRIPTION

This command allows the administrator to add local userids and passwords and corresponding remote userids and passwords to the domain password table.

The entries created are used for mapping remote user names and passwords to local user names and passwords when the application is using OSI gateways and `SECURITY` is set to `USER_AUTH`, `ACL`, or `MANDATORY ACL` in the `ubbconfig` file and `SECURITY` is set to `DM_PW` or `USER_PW` in the `DMCONFIG` file.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options may exceed this limit.

The following options are available:

`-ld local domain ID`

This is the domainID of the local domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the `DMCONFIG` file or through the Graphical Administrative Interface.

`-rd remote domain ID`

This is the domainID of the remote domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the `DMCONFIG` file or through the Graphical Administrative Interface.

`-lu local uid`

The user identification number. The local *uid* must be a positive decimal number smaller than 128K. *uid* must be unique within the list of existing identifiers for the application. *uid* defaults to the next available (unique) identifier greater than 0. The wildcard character ('*') is a valid *uid*.

`-lp local password`

The local password that is associated with the local uid to be added.

`-ru remote username`

The name of the remote user as defined in the remote domain's security application, for example, RACF. The wildcard character ('*') and space are valid remote usernames.

`-rp remote password`

The remote password that is associated with the local uid to be added.

The administrator is prompted for the local password and for the remote password if not specified on the command line.

Before running this command, the application must be configured using either the Graphical Administrative Interface or `tmloadcf` and `dmloadcf`. `dmusradd` may be run on any active node. If the application is not active, `dmusradd` must be run on the MASTER node.

PORTABILITY

This command is available only on non-/WS sites running TUXEDO System /T Release 6.4.

DIAGNOSTICS

The `dmusradd` command exits with a return code of 0 upon successful completion.

EXAMPLES

1. Associate uid 408 with theirname in CICS security
`dmusradd -ld ldom -rd cics -lu 408 -ru theirname`
2. Associate uid 409 with any remote user
`dmusradd -ld ldom -rd cics -lu 409 -ru '*'`
3. Associate any local user with remote theirname
`dmusradd -ld ldom -rd cics -lu '*' -ru theirname`

SEE ALSO

`dmusrdel`, `dmusrmod`

dmusrmod

Change a user password in the remote domain password file

SYNOPSIS

```
DMCONFIG=dmconfig
dmusrmod [ -ld local domain ID ][ -rd remote domain ID ]
[ -lu local uid ] [ -ru remote username ] [ -rp remote password ]
```

DESCRIPTION

This command allows the administrator to modify the username and password of remote domain users in the corresponding remote domain password file.

Once the entry is modified, the old information can no longer be used for mapping remote user names and passwords to local user names and passwords when the application is using OSI gateways and SECURITY is set to USER_AUTH, ACL, or MANDATORY ACL in the ubbconfig file and SECURITY is set to DM_PW or USER_PW in the DMCONFIG file.

The following options are available:

-ld local domain ID

This is the domainID of the local domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the DMCONFIG file or through the Graphical Administrative Interface.

-rd remote domain ID

This is the domainID of the remote domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the DMCONFIG file or through the Graphical Administrative Interface.

-lu local uid

The user identification number. The local *uid* must be a positive decimal number smaller than 128K. *uid* must be unique within the list of existing identifiers for the application. *uid* defaults to the next available (unique) identifier greater than 0. The wildcard character ('*') is a valid *uid*.

-ru remote username

The name of the remote user that is associated with the local *uid* to be modified.

`-rp remote password`

The remote password that is associated with the local `uid` to be modified.

Before running this command, the application must be configured using either the Graphical Administrative Interface or `tmloadcf` and `dmloadcf`. `dmusrmod` may be run on any active node. If the application is not active, `dmusrmod` must be run on the MASTER node.

PORTABILITY

This command is available only on non-/WS sites running TUXEDO System /T Release 6.4.

DIAGNOSTICS

The `dmusrmod` command exits with a return code of 0 upon successful completion.

EXAMPLES

Change `uid` 408

```
dmusrmod -ld ldom -rd cics -lu 408 -ru newname -rp*****
```

SEE ALSO

`dmusrdel`, `dmusradd`

GWADM

/Domain gateway administrative server

SYNOPSIS

```
GWADM SRVGRP = "identifier" SRVID = "number" REPLYQ = "N"  
      CLOPT = "-A -- [-a { on | off } ] [-s services ]  
      [-t { on | off } ]"
```

DESCRIPTION

The gateway administrative server GWADM is a TUXEDO-supplied server that provides administrative functions for a /Domain gateway group.

GWADM should be defined in the *SERVERS section of the UBBCONFIG file as a server running within a particular gateway group, that is, SRVGRP must be set to the corresponding GRPNAME tag specified in the *GROUPS section. The SVRID parameter is also required and its value must consider the maximum number of gateways allowed within the gateway group.

There should be only one instance of a GWADM per /Domain gateway group, and it should NOT be part of the MSSQ defined for the gateways associated with the group. Also, GWADM should have the REPLYQ attribute set to N.

The CLOPT option is a string of command line options that is passed to the GWADM when it is booted. This string has the following format:

```
CLOPT="-A -- <gateway group runtime parameters>"
```

The following runtime parameters are recognized for a gateway group:

`-a { on | off }`

This option turns *off* or *on* the audit log feature for this local domain. The default is *off*. The `dmadmin` program can be used to change this setting while the gateway group is running (see `dmadmin`).

`-s services`

Specifies the remote *services* that should be initially offered by the domain gateway. The specifications for these services are found in the DMCONFIG file. For example, the specification

`-s x,y,z`

implies that the gateway should initially advertise remote services *x*, *y*, and *z*. Spaces are not allowed between commas and the *-s* option may appear several times.

-t { *on* | *off* }

This option turns *off* or *on* the statistics gathering feature for the local domain. The default is *off*. The *dmadmin* program can be used to change this setting while the gateway group is running (see *dmadmin*).

The GWADM server must be booted before the corresponding gateways.

PORTABILITY

GWADM is supported on TUXEDO-supplied servers, using UNIX System operating systems.

INTEROPERABILITY

The initial release of OSI Domain can only be installed on a node running TUXEDO Release 6.4.

EXAMPLES

The following example illustrates the definition of the administrative server in the UBBCONFIG file.

```
#
*GROUPS
DMADMGRP  GRPNO=1
gwgrp     GRPNO=2
#
*SERVERS
DMADM SRVGRP="DMADMGRP" SRVID=1001 REPLYQ=N RESTART=Y GRACE=0
GWADM SRVGRP="gwgrp" SRVID=1002 REPLYQ=N RESTART=Y GRACE=0
      CLOPT="-A -- -a on -t on"
GWTDOMAIN SRVGRP="gwgrp" SRVID=1003 RQADDR="gwgrp" REPLYQ=Y
RESTART=Y MIN=1 MAX=1
```

SEE ALSO

dmadmin, *tmboot*

dmconfig, *DMADM*, *servopts*, *ubbconfig*

TUXEDO /Domain Guide

TUXEDO Administrator's Guide

modusr

Modify a remote user password

SYNOPSIS

```
DMCONFIG=dmconfig  
modusr -d local domain ID -R remote domain ID -u remote username
```

DESCRIPTION

`modusr` can only be executed as a subcommand of `dmadmin`. The purpose of this page is to describe options for the subcommand and to show an example.

The subcommand allows the administrator to modify passwords in the remote password table. The administrator is prompted for the remote password.

The table entries modified are used for passing remote user names and passwords to remote OSI domains when the application is using OSI gateways and `SECURITY` is set to `USER_AUTH`, `ACL`, or `MANDATORY ACL` in the `ubbconfig` file and `SECURITY` is set to `DM_USER_PW` in the `DMCONFIG` file.

The following options are available:

`-d local domain ID`

This is the name of the local domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the `DMCONFIG` file or through the Graphical Administrative Interface.

`-R remote domain ID`

This is the name of the remote domain gateway with which the ids and passwords are associated. This is the same ID as the one used when creating the domain definitions either in the `DMCONFIG` file or through the Graphical Administrative Interface.

`-u remote username`

The remote user whose password is being modified.

Before running this subcommand the application must be configured using either the Graphical Administrative Interface or `tmloadcf` and `dmloadcf`. `dmadmin modusr` may be run on any active node.

PORTABILITY

This subcommand is available only on non-/WS sites running TUXEDO System /T Release 6.4.

DIAGNOSTICS

The `dmadmin modusr` subcommand exits with a return code of 0 upon successful completion.

EXAMPLES

```
modusr -d tux -R cics -u cicsusr /*modifies remote user's password  
                                sent to CICS. The administrator  
                                is prompted for the password*/
```

SEE ALSO

`delusr`, `addusr`

Index

Symbols

- *DM_ACCESS_CONTROL
 - required parameters 3-12
- *DM_LOCAL_DOMAINS
 - format 3-5
 - optional parameters 3-6
 - required parameters 3-5
- *DM_LOCAL_SERVICES
 - optional parameters 3-13
 - required parameters 3-13
- *DM_OSITP
 - addressing information requirement 3-10
 - format 3-10
 - optional parameters 3-11
 - required parameters 3-10
- *DM_REMOTE_DOMAINS
 - format 3-8
 - optional parameters 3-8
 - required parameters 3-8
- *DM_REMOTE_SERVICES
 - optional parameters 3-15
 - required parameters 3-14
- *DM_ROUTING
 - optional parameters 3-18
 - required parameters 3-16
- *DM_TDOMAIN
 - device name 3-9
 - format 3-9
 - network address 3-9
 - optional parameters 3-9
 - required parameters 3-9

A

- access control list
 - format 3-12
 - local domains 3-12
 - using wildcards 3-13
- addressing
 - format for *DM_OSITP 3-10
 - format for *DM_TDOMAINS 3-9
 - OSITP domains 3-10
 - requirements for TDOMAIN 3-8
- addumap
 - reference page B-2
- addusr
 - reference page B-4
- architecture 1-4
- ASN.1 encoding 5-2
- ATMI calls 5-4
- audit log
 - local domain 3-6

B

- blocking 3-6
- buffers
 - data-dependent routing 3-16
 - defining VIEWS 5-6
 - encoding input 5-2
 - layout conversion 5-2
 - mapping 5-2
 - selecting type 5-5
 - types 5-1
- build_dgw
 - reference page B-6

C

- calls
 - ATMI 5-4
 - conversational 5-5
- classes

- grouping local domains 3-5
- remote domains 3-8
- compile
 - VIEW files 5-6
- components 1-7
- configuration 3-1
 - defining new gateways 3-1
 - remote systems 1-3
- connections
 - maximum per gateway 3-7
- conversational calls 5-5
- customer support xii

D

- data
 - maximum amount 3-7
- delumap
 - reference page B-9
- delusr
 - reference page B-11
- device name 3-9
- distribution libraries 2-13
- DMADM
 - reference page B-13
- dmadmin
 - reference page B-15
- DMCONFIG
 - addressing information for OSITP domains 3-10
 - addressing information for TDOMAIN 3-8
 - defining new gateways 3-1
 - editing 3-2
 - file format 3-2
 - identifying local domains 3-5, 3-7
 - parameters 3-5
 - processing file 3-1
- dmconfig
 - reference page B-34
- dmloadcf
 - reference page B-56
 - utility to process DMCONFIG file 3-1

DMTLOG 3-6
dmunloadcf
 reference page B-59
dmusradd
 reference page B-60
dmusrmod
 reference pages B-62
document conventions x
documentation support xii
Domain transaction log 3-6
 page size 3-7
domains
 local 3-5
 remote 3-7

E

edit
 DMCONFIG 3-2
 DMTYPE file 2-6, 2-12
 UBBCONFIG file 3-22
error messages A-1
executables 2-13

F

features
 product 1-4

G

gateways
 associated to local domains 3-5
 defining new gateways 3-1
 editing DMTYPE 3-1
 maximum connections 3-7
 modifying configuration 3-22
 server group names 3-5
 servers 1-9
GWADM
 reference page B-64

I

- input record
 - mapping to input buffer 5-2
- installation
 - UNIX platforms 2-3
 - Windows NT 2-6

L

- local domain
 - access control list 3-12
 - audit log 3-6
 - class groups 3-5
 - format for *DM_LOCAL_DOMAINS 3-5
 - identifying 3-5
 - maximum amount of data 3-7
 - maximum message length 3-7
 - naming 3-5
 - number of simultaneous global transactions 3-7
 - routing requests to remote service 3-15
 - services exported 3-13
- log
 - transaction 3-6

M

- mapping
 - adding userids 4-5
 - deleting userids 4-6
 - output record to output buffer 5-3
 - typed buffers 5-2
- messages
 - maximum length for local domain 3-7
- modusr
 - reference page B-67

N

- network address 3-9

O

- online documentation viii
- printing x
- web browsers viii

P

- parameters
 - DMCONFIG 3-5
- passwords
 - adding 4-5
 - deleting 4-6
 - modifying 4-7
- platform support 2-1
- pre-installation 2-1
 - configuration 2-2

R

- reference pages B-1
 - addumap B-2
 - addusr B-4
 - build_dgw B-6
 - delumap B-9
 - delusr B-11
 - DMADM B-13
 - dmadmin B-15
 - dmconfig B-34
 - dmloadcf B-56
 - dmunloadcf B-59
 - dmusradd B-60
 - dmusrmod B-62
 - GWADM B-64
 - modusr B-67
- remote domain
 - classes 3-8
 - data-dependent routing 3-15
 - format for *DM_REMOTE_DOMAINS 3-8
 - format for *DM_REMOTE_SERVICES 3-14
 - identifying 3-7, 3-8

- imported services 3-14
- RDOMs in access control list 3-13
- restricting services 3-14
- service execution 3-15
- remote service
 - routing request to 3-15
- remote systems
 - configuration 1-3
 - connections 1-3
 - receive requests 1-6
 - replies 1-6
- remote users
 - adding 4-5
 - deleting 4-6
- requests
 - routing to remote service 3-15
- routing
 - data-dependent 3-15
 - format for *DM_ROUTING 3-16
 - requests to remote service 3-15
 - service requests for typed buffers 3-16

S

- security
 - access control defined in UBBCONFIG 4-3
 - enforced by Connect OSI TP 4-3
 - UBBCONFIG parameters 4-3
- servers
 - defining Connect OSI TP servers for TUXEDO 3-22
 - Domain administration 1-9
 - gateway 1-9
- services
 - execution by remote domain 3-15
 - exported by local domains 3-13
 - format for *DM_LOCAL_SERVICES 3-13
 - imported on remote domains 3-14
 - restricting requests by remote domains 3-14
- stack
 - Unisys-supplied 2-1
- support

customer xii
documentation xii
technical xii

T

TDOMAIN

 addressing information requirement 3-8
transaction log
 name 3-7
transactions
 simultaneous global 3-7
TUXEDO system
 recognizing Connect OSI TP servers 3-22
typed buffers 5-1
 data-dependent routing 3-16

U

UBBCONFIG file

 sample 3-23
 security parameters 4-3
uninstall 2-15
Unisys 1-1
UNIX platforms
 installation 2-3
upgrading Connect OSI TP 2-2
Upper Layer Services (ULS) 1-1
userid
 adding 4-5
 deleting 4-6
 mapping 4-5

V

VIEW files

 compiling 5-6

W

wildcards

access control lists 3-13
Windows NT
installation 2-6

X

XAP
binding to dialogue instance 3-10