BEA eLink OSI TP

User Guide

BEA eLink OSI TP Version 4.0
Document Edition 4.0
December 2000

Copyright

Restricted Rights Legend

Trademarks or Service Marks

BEA eLink OSI TP User Guide

Document Edition	Part Number	Date	Software Version
4.0	870-001041-001	December 2000	BEA eLink OSI TP 4.0

Contents

About This Document

1. Introducing BEA eLink OSI TP

2. Managing Transactions and Buffers

3. Understanding the UDMCONFIG File

4. Configuring BEA eLink OSI TP

5. Using the OSI TP Administration Utility

About This Document

The BEA eLink OSI TP product (hereafter referred to as eLink OSI TP) is a gateway connectivity feature that enables application programs on BEA Tuxedo systems to perform various transactional and non-transactional tasks with application programs that reside on different kinds of computers.

This document covers the following topics:

- Introducing BEA eLink OSI TP describes the BEA eLink platform and BEA eLink OSI TP product.
- Managing Transactions and Buffers provides an overview of how buffers and data translation are managed.
- Understanding the UDMCONFIG File describes the `UDMCONFIG` file sections and parameters.
- Configuring BEA eLink OSI TP provides procedures for configuring the BEA eLink OSI TP product.
- Using the OSI TP Administration Utility provides procedures for using the `osiadmin` utility.
- Error and Informational Messages describes error and informational messages and provides the recommended user action in response to eac message.
- Utilities Reference provides a list of commonly used utilities and explains how they are used.
- Manually Upgrading BEA eLink OSI TP to Version 4.0 describes the procedure for upgrading your `UDMCONFIG` file manually.

What You Need to Know

This document is intended for system administrators who will install the eLink OSI TP software.

e-docs Web Site

BEA Tuxedo product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation or go directly to the "e-docs" Product Documentation page at **http://edocs.bea.com**.

How to Print the Document

A PDF version of this document is available on the OSI TP documentation Home page on the e-docs Web site (and also on the installation CD). You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format. To access the PDFs, open the OSI TP documentation Home page, click the PDF files button and select the document you want to print.

If you do not have the Adobe Acrobat Reader, you can get it for free from the Adobe Web site at **http://www.adobe.com/**.

Related Information

Various other related documentation is available to help you understand the eLink OSI TP product, Tuxedo product, and OLTP technology. Refer to the documentation sources listed in the following sections for additional information about BEA products and OLTP technology.

BEA eLink OSI TP Documentation

The eLink OSI TP documentation consists of the following items:

- *BEA eLink OSI TP Installation Guide*
- *BEA eLink OSI TP User Guide*
- *BEA eLink OSI TP Release Notes*

BEA Tuxedo Information

Information about the BEA Tuxedo Version 7.1 and Version 6.5 products is available on the BEA Tuxedo Online Documentation web site.

Tuxedo Release 7.1

The following BEA Tuxedo Release 7.1 information is available at **http://edocs.bea.com/Tuxedo/tux71/index.htm**:

- *Installation*
- *Getting Started*
- *Administration*
- *Programming*
- *Platforms*

- *Reference*
- *Messages*

Tuxedo Release 6.5

The following Tuxedo Release 6.5 publications are available at **http://edocs.bea.com/Tuxedo/tux65/index.htm**:

- *BEA Tuxedo Administering the BEA Tuxedo System*
- *BEA Tuxedo Application Development Guide*
- *BEA Tuxedo FML Programmer's Guide*
- *BEA Tuxedo Programmer's Guide*
- *BEA Tuxedo Reference Manual*

Other Publications

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

- *The Tuxedo System* (Andrade, Carges, Dwyer, Felts)
- *Tuxedo: An Open Approach to OLTP* (Primatesta)
- *Building Client/Server Applications Using Tuxedo* (Hall)

Customer Support

The BEA eLink OSI TP product is jointly developed by BEA Systems, Inc. and Unisys Corporation. Support services and contacts are described in the following sections.

SURETY Support Services

A SURETY service agreement gives you easy access to Unisys Support Online or Unisys Client Support Centers. These services are described below.

Unisys Customer Support

Unisys customer support gives clients easy access, on the World Wide Web or by telephone, to these service options:

- Support Online
- Client Support Centers
- Tech Advantage

Clients with valid support agreements can visit an electronic database of reported problems and solutions. They can also use e-mail or telephone to contact our product specialists with technical questions. Clients are eligible to receive updates to a product when critical problems have been fixed.

Support Online

Unisys has a World Wide Web site available to customers who have support agreements. If you have any questions or problems with a Unisys product, your first source of information is Support Online, an online technical resource available through the World Wide Web at **http://www.support.unisys.com**.

This technical resource is available 24 hours a day, 7 days a week. It offers a wealth of information about Unisys hardware and software systems. Take some time to explore Support Online and discover the services available to you.

Client Support Centers

Unisys clients with valid support agreements can also access Client Support Centers. If you have unresolved questions or problems concerning Unisys product installation or operation (after referring to the documentation or your system administrator), call the appropriate Client Support Center:

Within the continental United States or Canada, call one of the following toll-free numbers:

1-800-328-0440 (prompt 4) - United States

1-800-387-6181 - Canada

1-800-361-8097 - French Canada

Outside the continental United States, obtain a support center telephone number from your local Unisys customer support representative.

To expedite your request, please have the following information ready before contacting Unisys.

- The name of the hardware (for example, SUN, HP, AIX, UNIXWARE or PC)
- The name and release level of the operating environment (for example UNIX 7.1, or Windows NT 4.0)
- The BEA Tuxedo release level, the BEA eLink OSI TP release level, and the patch levels of both.

Tech Advantage

In addition, Unisys Client Support Centers offer Tech Advantage, a full complement of technical service packages that provide you with cost-effective, fast-cycle support. Tech Advantage services can help you solve your most difficult problems or maximize the effectiveness of your system, regardless of your product choice.

Here are some ways to put Tech Advantage to work for you:

- **System Generation**-Assist in generating or installing system software upgrades and enhancements.
- **Operating System Migration**-Plan and optimize your system migrations.
- **Performance Review**-Evaluate existing systems and networks and provide tuning recommendations.
- **Communications Systems Enhancements**-Assist in configuring, generating, and installing communications systems.
- **Database Management Consulting**-Confirm that systems and database backup procedures are optimized to promote data recovery.

For more information on Tech Advantage, visit the Support Online Web page and follow the link to Tech Advantage or call your Client Support Center.

Reporting a Software Problem with a User Communication Form

A customer representative may provide instructions or solutions from the Client Support Center database. If this help is insufficient, provide your representative with a full description of the problem, including any relevant error messages, and, if possible, a full description of the function you were attempting and your actions immediately prior to the error. The representative will then submit an electronic User Communication Form (UCF) to Unisys on your behalf, and give you the following information:

- A UCF registration number
- Instructions for submitting supporting materials, such as traces and dumps

VAR Customer Support

All clients of Unisys resellers should contact their Value Added Resellers (VARs) for support access details.

Documentation Support

Your feedback on the BEA eLink OSI TP documentation is important to us. Send us e-mail at **docsupport@bea.com** if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the OSI TP documentation.

In your e-mail message, please indicate that you are using the documentation for the BEA OSI TP 4.0 release.

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
blue text	Indicates a hypertext link in PDF or HTML
italics	Indicates emphasis or book titles or variables.
`"string with quotes"`	Indicates a string entry that requires quote marks.
UPPERCASE TEXT	Indicates generic file names, device names, environment variables, and logical operators. *Examples*: LPT1 SIGNON OR

`monospace text`	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

`monospace boldface text`	Identifies significant words in code. *Example*: `void` **`xa_commit`** `( )`
`{ }`	Indicates a set of choices in a syntax line. The braces themselves should never be typed.

Convention	Item

`[ ]`	Indicates optional items in a syntax line. The brackets themselves should never be typed.
	Example:
	`buildclient [-v] [-o name ] [-f file-list]...`
	`[-l file-list]...`

|	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.

. . .	Indicates one of the following in a command line:
	■ That an argument can be repeated several times in a command line
	■ That the statement omits additional optional arguments
	■ That you can enter additional parameters, values, or other information
	The ellipsis itself should never be typed.
	Example:
	`buildclient [-v] [-o name ] [-f file-list]...`
	`[-l file-list]...`

. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

1 Introducing BEA eLink OSI TP

This section covers the following topics:

BEA eLink Solution Overview

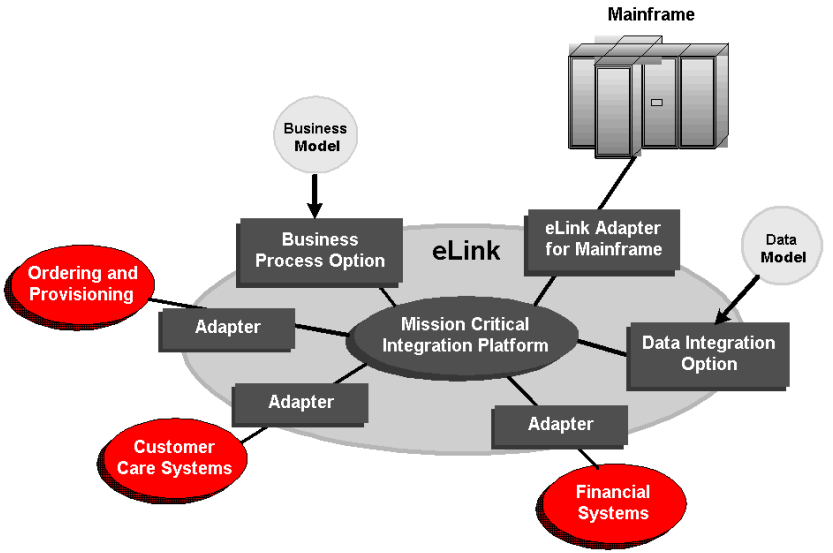
BEA eLink™ provides an open Enterprise Application Integration (EAI) solution that allows applications throughout organizations to communicate seamlessly. Using EAI, you gain the long-term flexibility and investment protection you need to keep up with today's ever-changing business environment.

Typically, companies use packaged applications to automate internal operations, such as financial, manufacturing, or human resources. While they successfully address the needs of these specific areas, these proprietary platforms often do not work together. To compete today, you need a much greater exchange of information. Systems need to communicate at a process level within your own organization, as well as with

customer's and supplier's systems. BEA eLink Platform is the underlying basis of BEA eLink, a family of off-the-shelf enterprise application integration (EAI) products that leverage the BEA transaction platform to integrate existing legacy applications with customer-focused and business-to-business e-commerce initiatives.

BEA eLink Platform provides a proven infrastructure for integrating applications within the enterprise and across the Web. BEA eLink Platform ensures high-performance, secure transactions and transparent access to mission-critical applications and information throughout the enterprise and across the Web. Figure 1-1 illustrates the eLink logical architecture and shows where the eLink Adapters fit into the process.

Figure 1-1 BEA eLink Solution Illustration



The entire BEA eLink family (including all options and adapters) is highly scalable. Multiple instances of BEA eLink components can collaborate so that work is divided between eLink domains. BEA eLink includes Simple Network Management Protocol (SNMP) integration for enterprise management.

The current BEA eLink Platform leverages the BEA Tuxedo infrastructure because it is based on a service-oriented architecture. Both BEA Tuxedo and BEA eLink communicate directly with each other and with other applications through the use of services. Multiple services are grouped into "application servers" or "servers". The

terms Tuxedo services/servers and eLink services/servers can be used interchangeably. Because this document is specifically addressing the eLink family, the terms “eLink service” and “eLink server” are used throughout.

The BEA eLink Platform complies with the Open Group’s X/Open standards including support of the XA standard for two-phase commit processing, the X/Open **ATMI** API, and XPG standards for language internationalization. C, C++, COBOL, and Java are supported. The BEA eLink Platform connects to any RDBMS, OODBMS, file manager or queue manager, including a supplied XA-compliant queueing subsystem.

The following components operate with BEA eLink Platform:

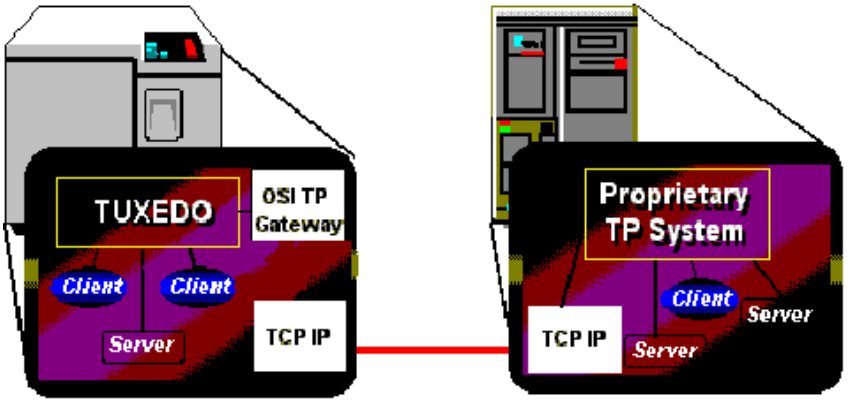
- The **Business Process Option** helps automate tasks in the distributed global business process and dynamically responds to business events and exceptions. The BPO is currently implemented by integrating eLink with technology based on InConcert workflow management software.
- An **eLink Adapter** provides the interface between the BEA eLink Platform and external applications with out-of-the-box functionality.

BEA eLink OSI TP Overview

BEA eLink OSI TP is a gateway connectivity feature that 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 Open Group 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, Microsoft MTS through Unisys OpenTI, and ICL Open VME.

Figure 1-2 BEA eLink OSI TP Communicating with an OSI TP Partner



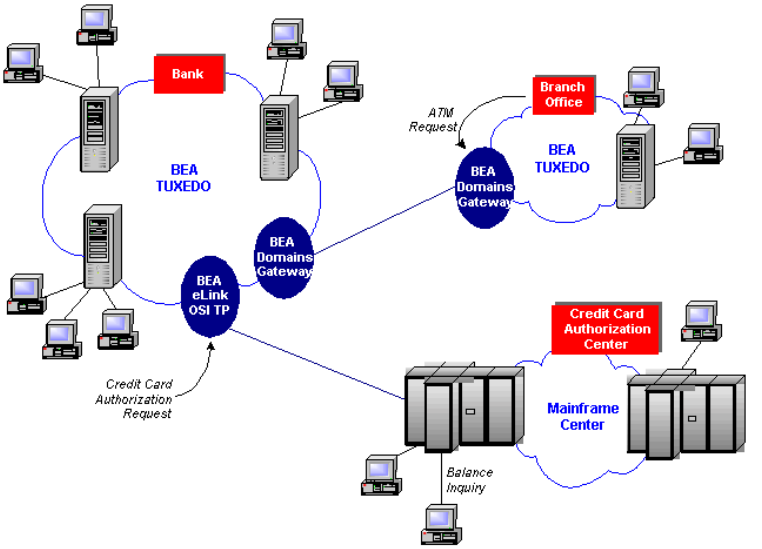
BEA eLink OSI TP is a software product that implements the OSI-TP standard, a set of protocols that is used to:

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

The Open Group 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 eLink 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. The eLink OSI TP software is designed to provide transparent access to remote services that reside outside a BEA Tuxedo application. In addition, eLink OSI TP provides remote application programs access to local services.

Figure 1-3 BEA eLink OSI TP Sample Environment



BEA eLink OSI TP Features

The eLink OSI TP product supports the following features:

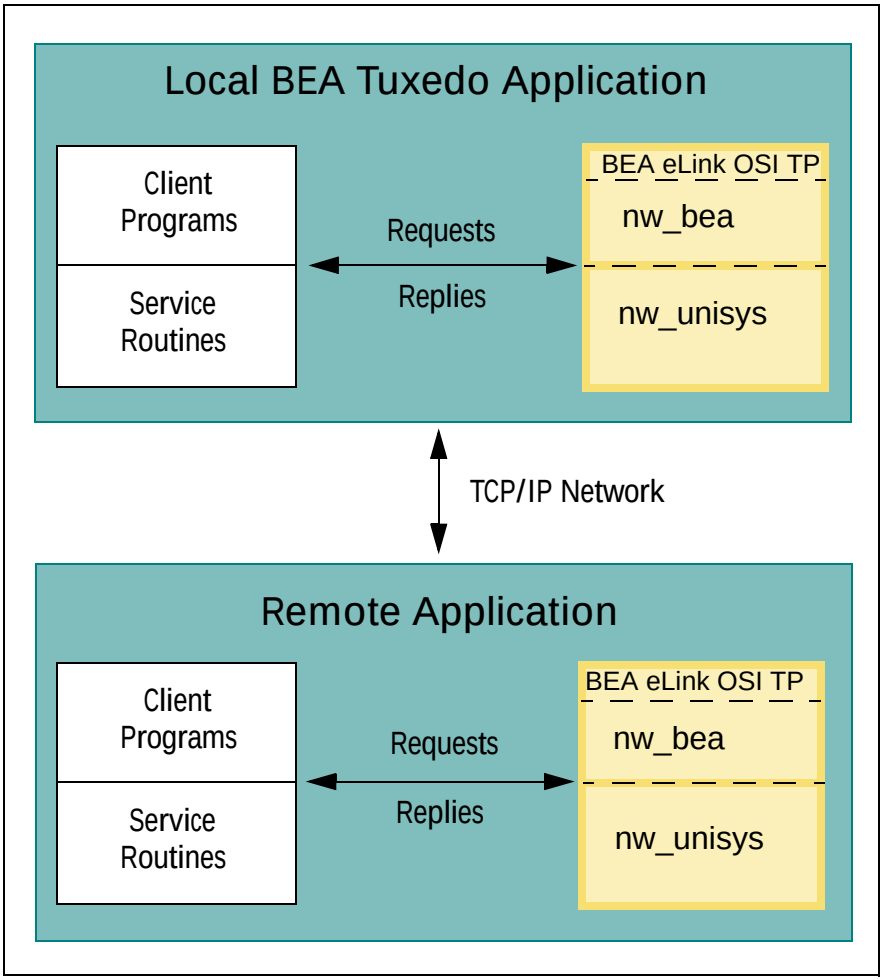
- Tightly-coupled and loosely-coupled transaction processing
- Full array of buffer translation options
- Optional AUTOPREPARE feature that allows requests to remote services to be automatically prepared
- Synchronous and asynchronous service requests and replies initiated from either the BEA Tuxedo 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 configuration support
- ASN.1 based data conversion
- Link-layer security as well as security through access control lists on the local domain
- Multiple simultaneous request/response sessions and conversations
- Connections to remote domains
- Transaction recovery for failures occurring during the second phase of commitment

BEA Tuxedo and eLink OSI TP 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. The location of server programs is kept transparent because remote services are mapped to servers in a section of the configuration file. The eLink OSI TP architecture comprises two distinct internal components, nw-bea and nw-unisys. These two internal components extend the transparent access of the BEA Tuxedo system by sending requests to and receiving requests from remote systems through OSI TP and supporting network software. Figure 1-4 shows how this transparent access works.

Figure 1-4 Routing Service Calls through BEA eLink OSI TP



- When local BEA Tuxedo client programs send requests to remote systems, eLink OSI TP transforms those requests into OSI TP messages. Also, when remote systems respond, eLink OSI TP transforms associated OSI TP messages into replies that local client programs can process.
- When remote client programs send OSI TP messages to local applications, eLink OSI TP transforms those messages into requests that local BEA Tuxedo service

routines can process. Also, when local service routines send replies, eLink OSI TP transforms those replies into OSI TP messages.

- When commitment protocol is sent for global transactions, eLink OSI TP transforms the protocol and manages the transaction commitment.
- When the Tuxedo application type is defined with the `MODEL` parameter set to `SHM` in the `RESOURCES` section of the Tuxedo `UBBCONFIG` file, the OSI TP Domain administration server, `UDMADM`, is optional. However, a warning message appears when the gateway is booted.

The eLink OSI TP software is implemented as an ordinary BEA Tuxedo server group. It accepts standard BEA Tuxedo service requests and returns standard replies. The eLink OSI TP server group consists of the following components:

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

One eLink OSI TP server group acts as a gateway to multiple remote systems using multiple communications targets. Each communications target is a unique OSI TP endpoint called an *association*. There is one association or connection to the remote system for each call to the remote system. Once the call is complete, that association is reused by subsequent calls. There is a connection to the remote system for each simultaneous call made to the remote system. The association is released after a pre-set timer expires, turning unused resources back to the system. This pre-set timer is controlled by parameters specified in the OSI TP tailor file. Refer to Tuning OSI TP-Specific Tables with the TAILOR File for more information.

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

Other products such as remote Unisys A Series Open/OLTP systems, Unisys OS2200 systems, Unisys OpenTI for MTS interoperability, and ICL TPMS for Open VME provide analogous functionality with which local eLink OSI TP servers can interact.

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

To a remote system that supports the Open Group XATMI standard (more specifically, an XATMI application service element), the eLink OSI TP server group appears as a communications resource manager (CRM).

OSI TP Software Overview

Open Systems Interconnection (OSI) is a computer network architecture developed by the International Organization for Standardization (ISO) in cooperation with other standards Organizations. OSI standards define protocols for communication between diverse computer systems in a network.

Benefits of OSI

By conforming to OSI standards, computer hardware and software vendors can enable applications on a local computer system to communicate with applications on remote computer systems that are manufactured by another vendor or have a different architecture from the local system.

OSI Reference Model

The OSI architecture is based upon a framework that divides the networking tasks and requirements into seven layers. The layers are groups of related functions or tasks, intended to make their interfaces and functions easier to understand.

Each layer contains entities that perform specific functions in the communication process. Entities throughout the network that are in the same layer and perform the same functions on different systems are called peer entities. They communicate with each other in a standard way. This is called protocol.

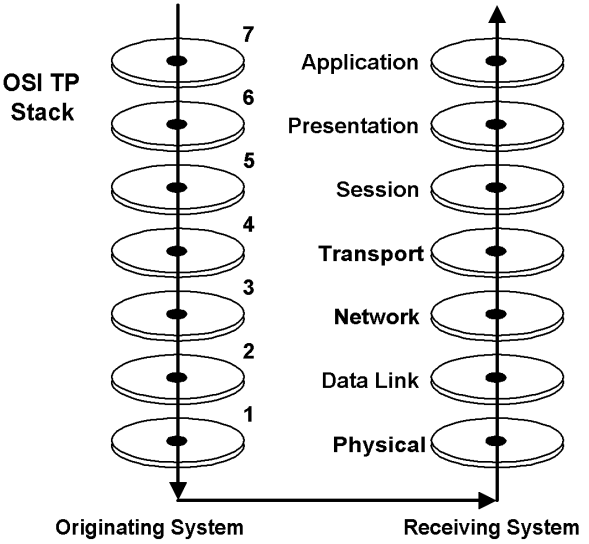
In the OSI Reference Model, peer entities cannot communicate directly. On the sending system, a layer entity uses peer protocols to attach a header containing routing and control information to the message being sent. It then passes this information down to the next layer. That layer adds its own header information and passes it to the next lower layer.

When the message reaches the receiving system, entities in each layer:

- Read and remove the information in the headers added by their peers on the sending system.
- Pass the remaining message up to the next layer.

The following figure illustrates the seven common protocol layers of the OSI Reference Model.

Figure 1-5 OSI Reference Model



Transaction Processing Services

OSI TP works with Tuxedo to provide the following services:

- Marks the beginning of a distributed transaction.
- Coordinates the commitment and rollback of a distributed transaction.
- Performs automatic recovery of all communications paths in the event of a failure.

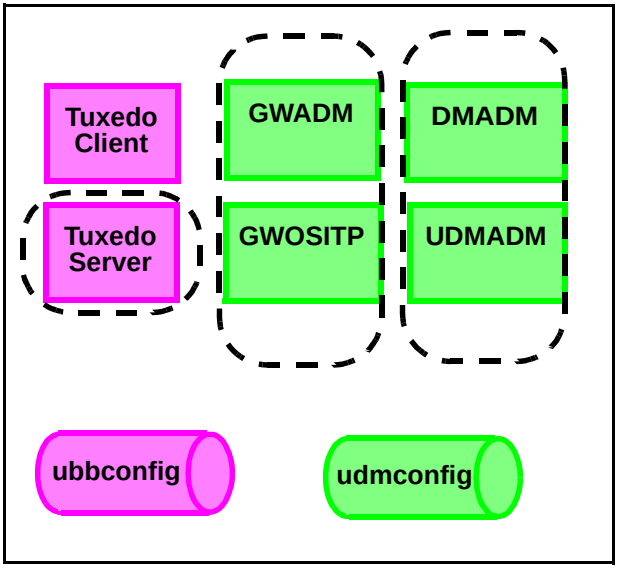
OSI TP Domains Components

The eLink OSI TP software and other eLink products act as gateways between BEA Tuxedo systems and other online transaction processing environments. Connections with remote systems are established by configuring eLink OSI TP as an ordinary BEA Tuxedo server group that identifies remote systems and available services.

The eLink OSI TP gateway 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 eLink OSI TP product.

Figure 1-6 Domain Components



LEGEND:
DMADM = Domain Administration Server
UDMADM = OSI TP Domain Administration Server
GWADM = Gateway Administration Server
GWOSITP = Transactional/OSI TP Domains Instance

BEA eLink OSI TP uses the following administrative servers for domain and gateway configuration and administration:

- `DMADM` - Domain administration server
- `GWADM` - Gateway administration server
- `GWOSITP` - OSI TP Domain Gateway
- `UDMADM` - OSI TP Domain administration server

Note: The gateway, `GWOSITP`, must be started **AFTER** the other servers.

If the eLink OSI TP gateway is used on multiple systems in the same domain, the `UDMADM` will send eLink OSITP-specific configuration information to the other systems in the domain. Refer to Utilities Reference for more detailed information about `UDMADM`.

Specific information on configuring the various sections of a domain are covered in the BEA Tuxedo document, *BEA Tuxedo /Domain Guide.*

2 Managing Transactions and Buffers

This section covers the following topics:

Transaction Management

Transaction management provides coordination for the completion of transactions, whether the transaction is successful or not. Application programmers can request the execution of remote services within a transaction, or users at remote domains can request the execution of local services within a transaction. Domains transaction management coordinates the mapping of remote transactions to local transactions, and the sane termination (commitment or rollback) of these transactions.

In the BEA Tuxedo system, a transaction tree is a two-level tree where the root is the group coordinating a global transaction and the machines are other groups involved in the transaction. Each group performs its part of the global transaction independently

from the parts done by other groups. Each group, therefore, implicitly defines a transaction branch. If this BEA Tuxedo transaction branch is controlled by the eLink OSI TP domain, it may contain multiple actual OSI TP transaction branches. The eLink OSI TP gateway controls the mapping between the single BEA Tuxedo transaction branch and the many OSI TP transaction branches. The BEA Tuxedo system uses Transaction Manager Servers (TMS) to coordinate the completion of the global transaction and make sure each branch completes.

Domains transaction management can be summarized as follows:

- Gateways generate mappings from a BEA Tuxedo transaction to a network transaction. A new mapping is generated per BEA Tuxedo transaction or per incoming network transaction branch.
- Each instantiation handles its own representation of the network transaction tree. Instantiations observe the hierarchical nature of the inter-domain communication.

Tightly-Coupled and Loosely-Coupled Transactions

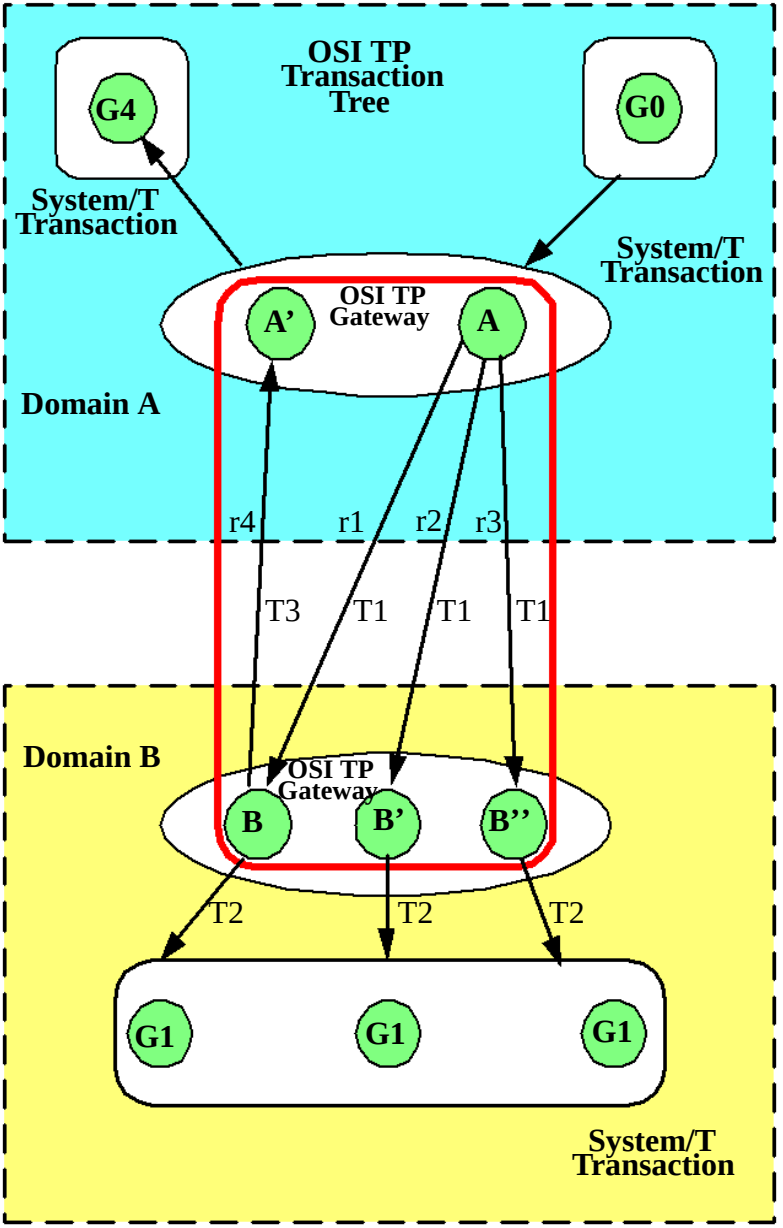
In the Open Group DTP Model, a Transaction Manager (TM) can construct transaction trees by defining either tightly-coupled or loosely-coupled relationships with a Resource Manager (RM). The coupling of the relationships are determined by the way the local service is defined in the `UDMCONFIG` file.

A tightly-coupled relationship is one in which the same transaction identifier, XID, is used by all processes participating in the same global transaction and accessing the same RM. This relationship maximizes data sharing between processes; XA-compliant RMs expect to share locks for resources used by processes having the same XID.

The BEA Tuxedo system achieves the tightly-coupled relationship through the group concept; that is, work done by a group on behalf of a given global transaction belongs to the same transaction branch; all the processes are given the same XID. In a loosely-coupled relationship, the TM generates a transaction branch for each part of the work in support of the global transaction. The RM handles each transaction branch separately; there is no sharing of data or of locks between the transaction branches. Deadlocks between transaction branches can occur. A deadlock results in the rollback of the global transaction. In the BEA Tuxedo system, when different groups participate in the same global transaction each group defines a transaction branch; this results in a loosely-coupled relationship. The eLink OSI TP instantiation is user configurable and can provide a tightly-coupled integration that solves this deadlock problem by minimizing the number of transaction branches required in the interoperation between two domains.

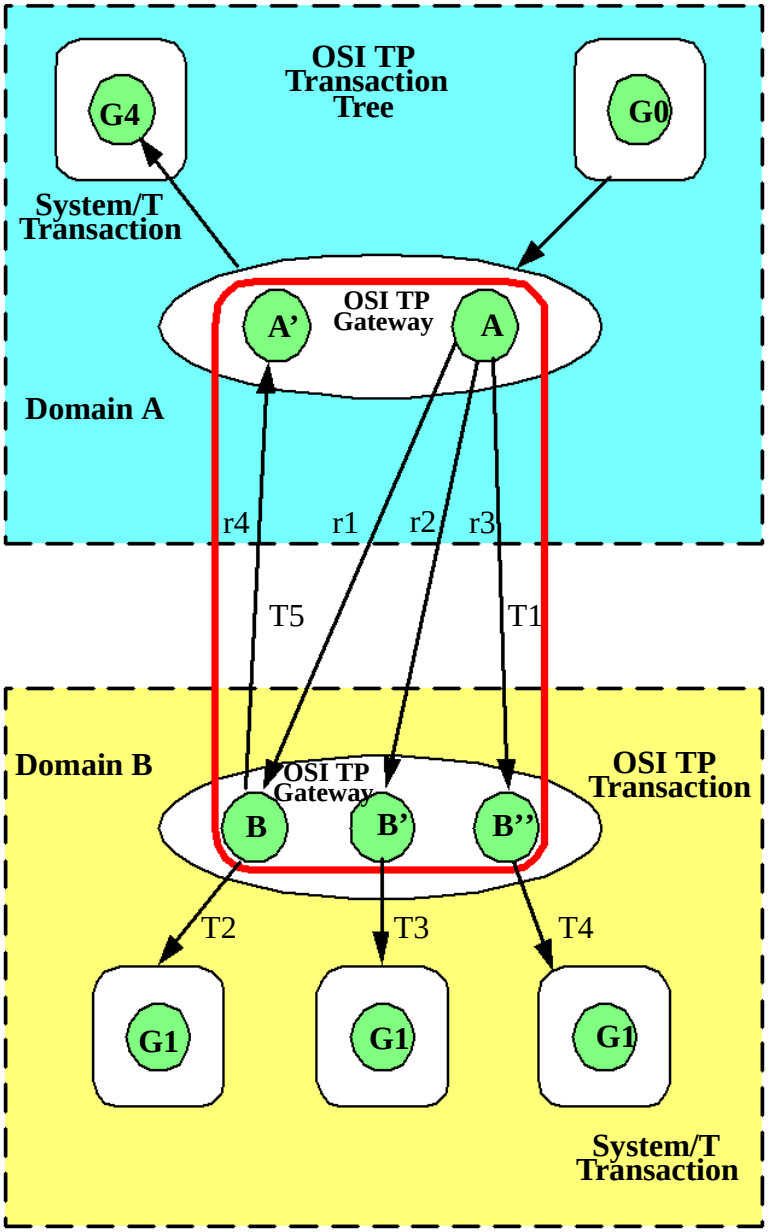
Following are diagrams showing loosely-coupled and tightly-coupled integrations and an explanation of each diagram.

Figure 2-1 Example of a Tightly-Coupled Integration



The transaction tree for the tightly-coupled integration shown in Figure 2-1 eliminates the probability for intra-transaction deadlock by minimizing the number of transaction branches required in the interoperation between two domains. Application A makes three requests (r1, r2 and r3) to a remote Domain B. OSI TP sends all three requests mapped to the same OSI TP transaction, T1. On Domain B, the eLink OSI TP gateway checks the `COUPLING` flag for the remote service and discovers that for service B the `COUPLING=TIGHT`. In this case all requests for service B belong to the same BEA Tuxedo system transaction. Each request for service B is added to the previous requests and all will have the same BEA Tuxedo `XID` indicated by T2. Resources in group G1 will not be isolated and changes made by any instantiation of service B for this transaction will be “seen” by the others. Request r4 is mapped to identifier T2 on Domain B, but the Tuxedo domain generates a new branch in its transaction tree (r4: B to A'). This is a new transaction branch on Domain A, and therefore, the gateway generate a new mapping T3, to a new BEA Tuxedo system transaction. The gateway group on Domain A also coordinates group G4, so the hierarchical nature of inter-domain communication is fully enforced with this mapping; group G4 cannot commit before group G1.

Figure 2-2 Example of a Loosely-Coupled Integration



The transaction tree for the loosely-coupled integration shown in Figure 2-2 shows group G0 in Domain A coordinating the global transaction started by the client. In this case application A sends three requests (r1, r2, and r3) to a remote Domain B. Like the tightly-coupled case, all three branches are represented by OSI TP transaction T1. On Domain B, the eLink OSI TP gateway checks the `COUPLING` flag for the remote service and sees that service B is `COUPLING=LOOSE`. In this case, the eLink OSI TP gateway generates three BEA Tuxedo system transactions: T2, T3 and T4. Any changes made to G1 are isolated. For example, any changes made by service B can not be “seen” by service B'. When B calls back the A', a new transaction, T5, is generated.

Global Transactions Across Domains

A global transaction in a single BEA Tuxedo application follows a two-level transaction tree, but a global transaction across domains follows a more complex transaction tree. There are two reasons for this:

- A transaction across domains may involve more domains than can be known from the root domain (where the transaction is controlled), so the structure of the transaction tree cannot be fully known.
- The root domain of a transaction across domains may not be directly connected to all domains in the transaction as would be required for a two-level transaction tree.

The commitment protocol across domains must be hierarchical to handle the complex transaction tree structure. For example, a loop-back service request is made from one domain (Domain A) to another domain (Domain B) and then comes back to be processed in the original domain. The service in Domain B requests another service in Domain A. The transaction tree has two branches at the network level: a branch b1 from A to B and a branch b2 from B to A. Domain A cannot commit the work done on branch b2 before receiving commit instructions from B.

The eLink OSI TP instantiation optimizes `GTRID` mapping by optionally implementing a tightly-coupled relationship. In eLink OSI TP, service requests issued on behalf of the same global transaction are mapped to the same network transaction branch. Therefore, incoming service requests can be mapped to a single BEA Tuxedo transaction. However, the hierarchical structure of inter-domain communication and the inter-domain transaction tree must still be maintained. See Figure 2-1 for an example of a tightly-coupled relationship.

The optimization that eLink OSI TP introduces applies only to a single domain. When two or more domains are involved in the transaction, the network transaction tree contains at least one branch per domain interaction. Therefore, across domains, the network transaction tree remains loosely-coupled. There will be as many branches as there are domains involved in the transaction (even if all branches access the same resource manager instance). Domains gateway groups implement a loosely-coupled relationship because they generate different transaction branches for inter-domain transactions. See Figure 2-2 for an example of a loosely-coupled relationship.

Notice that the gateway still must perform mappings between a BEA Tuxedo transaction and a network transaction, and that the hierarchical nature of the communication between domains must be strictly enforced. The figure shows that requests r1, r2, and r3 are mapped to a single eLink OSI TP transaction branch. Therefore, on Domain B only one BEA Tuxedo transaction needs to be generated; request r4 is mapped to an identifier on Domain B, but eLink OSI TP generates a new branch in its transaction tree (r4: B to A'). This is a new transaction branch on Domain A, and therefore, the gateway generates a mapping to a new BEA Tuxedo transaction. The graph shows that gateway group GW on Domain A also coordinates group G4. Hence, the hierarchical nature of inter-domain communication is fully enforced with this mapping: group G4 cannot commit before group G1.

Transaction Recovery

OSI TP can recover an entire transaction of individual dialogues if one or more failures occur during the second phase of the two-phase commit process. Failures that occur before the second phase of commitment cause the transaction to roll back automatically. Three types of failure can occur after the second phase of commitment begins. For these types of failures, the following transaction recovery actions can occur:

Table 2-1 Transaction Recovery Actions for Failures

Type of Failure	Transaction Recovery Action
Communications Failure	eLink OSI TP automatically re-establishes communications so the transaction can be completed

Type of Failure	Transaction Recovery Action
Software Failure	eLink OSI TP maintains communications with the other hosts involved in the transactions until the application is restored. When the application is restored, OSI TP informs the application of the active transactions that need to be completed and the state of each transaction. The application must direct OSI TP to commit or roll back each transaction.
System Failure	The application informs eLink OSI TP of the active transactions that need to be completed and the state of each transaction from the secured data log. eLink OSI TP then establishes communications with the other hosts involved so that transactions can be completed.

Buffer and Data Translation

The eLink OSI TP software uses typed buffers to transmit and receive data. Full buffer translation is supported for the following buffer types:

- CARRAY
- X_OCTET

Note: Null X_OCTET buffers are *not* supported.

- FML and FML32
- STRING
- VIEW and VIEW32
- X_C_TYPE
- X_COMMON

The following sections introduce procedures that eLink OSI TP follows to process and convert data buffers.

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 eLink OSI TP supports this combination. Interoperability using eLink OSI TP requires that remote systems support XATMI ASE. Therefore, Tuxedo-specific buffer types, such as `STRING, VIEW, FML` and `CARRAY` may need to be converted into XATMI standard types. BEA eLink 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.

Buffers and Records

The following terms are used to describe input and output data:

Buffer

Input or output data as it exists inside the local BEA Tuxedo region. This includes all the buffer types that BEA Tuxedo software supports-both BEA Tuxedo ATMI buffer types and X/Open XATMI buffer types.

Record

Input or output data as it exists outside the local BEA Tuxedo region on different kinds of Open/OLTP systems.

These terms make it easier to understand how eLink OSI TP handles input and output data.

Buffers Received from Local Programs

The eLink OSI TP gateway processes records from local programs in the following manner.

1. When eLink OSI TP receives a buffer from a local program, it automatically determines the buffer's type.

   The eLink OSI TP product automatically "types" input buffers that local client programs send to remote services.

   The eLink OSI TP product automatically "types" output buffers that local services return to remote client programs.

2. After eLink OSI TP determines a buffer's type, it refers to the configuration file (`UDMCONFIG`) to determine whether the buffer needs to be converted to a different format.

   Client requests sent to remote services may need to be converted to record formats that are meaningful to those services.

   Server responses returned to remote client programs may need to be converted to record formats that are meaningful to those programs.

3. If the configuration indicates that conversion is required, eLink OSI TP transforms the buffer into the record format that is specified in the configuration.

Records Received from Remote Programs

The eLink OSI TP gateway processes records from remote programs in the following manner.

1. When eLink OSI TP receives a record from a remote system, it refers to the domain configuration (`UDMCONFIG`) to determine the record's type and whether the record needs to be converted to a different format.

   Client requests from remote client programs may need to be converted to buffer formats that are acceptable to local service routines.

   Server responses returned from remote services may need to be converted to buffer formats that are acceptable to local client programs.

2. If the configuration indicates that conversion is required, eLink OSI TP transforms the record into the buffer format that is specified in the configuration.

Managing Parameters for Buffer and Record Conversion

The eLink OSI TP product provides four configuration parameters you can use to map buffers and records. These parameters are optional.

The following buffer configuration parameters are specified in the `DM_LOCAL_SERVICES` and/or the `DM_REMOTE_SERVICES` sections of the domain configuration file (`UDMCONFIG`) as appropriate.

INBUFTYPE
: Identifies the type, and in some cases the format, of a buffer received from a Tuxedo client or server. This restricts the buffer type naming space of data types accepted by this service to a single buffer type.

OUTBUFTYPE
: Identifies the type, and in some cases the format, of a buffer to be sent to a Tuxedo client or server. Use this parameter to specify the type and format to translate the incoming message to.

INRECTYPE
: Identifies the type, and in some cases the format, of a buffer to be sent to a remote gateway. This parameter is used for buffer and record translation.

OUTRECTYPE
: Identifies the type, and in some cases the format, of a buffer received from a remote gateway.

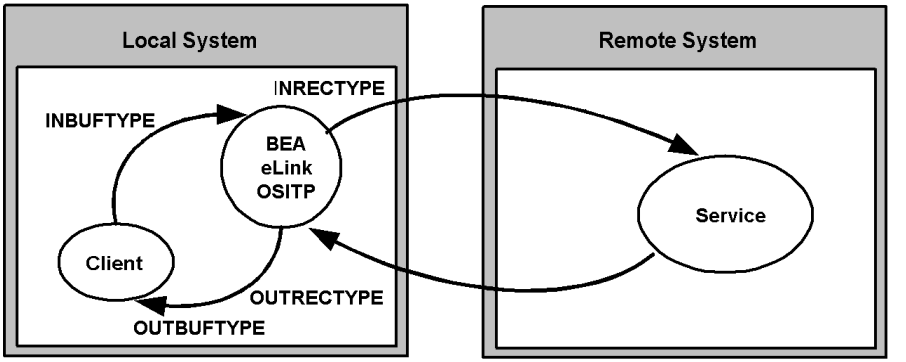
The definitions of these four parameters depend on whether the service requests originate locally or remotely. The following sections describe these parameters in relation to where the service request originates.

Parameters for Locally Originated Calls

This section describes in more detail how eLink OSI TP handles service calls that originate locally, within the immediate BEA Tuxedo region. It also explains how the INBUFTYPE, INRECTYPE, OUTRECTYPE, and OUTBUFTYPE parameters can be used to manage the conversion of buffers and records that flow between local client programs and remote services.

In the following figure, a local BEA Tuxedo client program issues a service call that a local eLink OSI TP gateway routes to a remote server through eLink OSI TP.

Figure 2-3 Figure 3-1 How Parameters Are Mapped During Locally Originated Calls



In this situation, the four configuration parameters that are shown in the figure have the following meanings:

- The INBUFTYPE parameter describes the BEA Tuxedo input buffer that the local client program provides to the eLink OSI TP gateway through BEA Tuxedo software. When this parameter is specified, the data type and subtype are verified.
- The INRECTYPE parameter describes the input record that is sent to the service on the remote system.
- The OUTRECTYPE parameter describes the output record that is received from the service on the remote system.
- The OUTBUFTYPE parameter describes the BEA Tuxedo output buffer that is returned to the local client program.

Guidelines for Mapping Input Buffers to Input Records

The following sections provide detailed information explaining how to use the INBUFTYPE and INRECTYPE parameters for service calls that originate locally (where local client programs call remote services).

INBUFTYPE

The INBUFTYPE parameter is used to specify the request buffer type that is provided to a local eLink OSI TP gateway when a local client program issues a service request. Tuxedo uses this information to restrict the buffer type from the local client to only the types defined by the INBUFTYPE parameter.

INRECTYPE

The INRECTYPE parameter is used to specify the type, and in some cases the format, of the request record that a particular remote service requires. The eLink OSI TP gateway uses this information to convert BEA Tuxedo request buffers into records that remote services can process.

The INRECTYPE parameter may be omitted if the request buffer is identical, in type and structure, to the request record the remote service expects.

You must specify the INRECTYPE parameter when one of the cases described in the following table is true.

Case	Explanation
The remote service uses an input record that is structured differently from the client program's request buffer	The remote service uses a record that is structured differently from the client program's VIEW, X_C_TYPE, or X_COMMON buffer. For example, the remote service may expect structure members to be sequenced differently.
The remote service uses a request record that differs from the client program's request buffer in both type and structure.	The client program uses a BEA Tuxedo FML buffer and the remote service expects a corresponding record with an appropriate structure.

Guidelines for Mapping Output Records to Output Buffers

The following sections provide detailed information explaining how to use the OUTRECTYPE and OUTBUFTYPE parameters for service calls that originate locally (where local client programs call remote services and receive output from those services).

OUTBUFTYPE

The OUTBUFTYPE parameter is used to specify the type, and in some cases the structure, of the reply buffer that a local client program expects. The eLink OSI TP gateway uses this information to map reply records from remote services to the appropriate kinds of reply buffers.The eLink OSI TP maps the incoming record to the type and subtype defined by the OUTBUFTYPE parameter.

OUTRECTYPE

The OUTRECTYPE parameter is used to specify the type, and in some cases the format, of the reply record that a particular remote service returns to the local eLink OSI TP gateway. The eLink OSI TP maps the incoming record to the type and subtype defined by the OUTBUFTYPE parameter.

The OUTBUFTYPE parameter may be omitted if the remote service returns a reply record that is identical, in type and structure, to the reply buffer the local client program expects.

You must specify the OUTBUFTYPE parameter when one of the cases described in the following table is true.

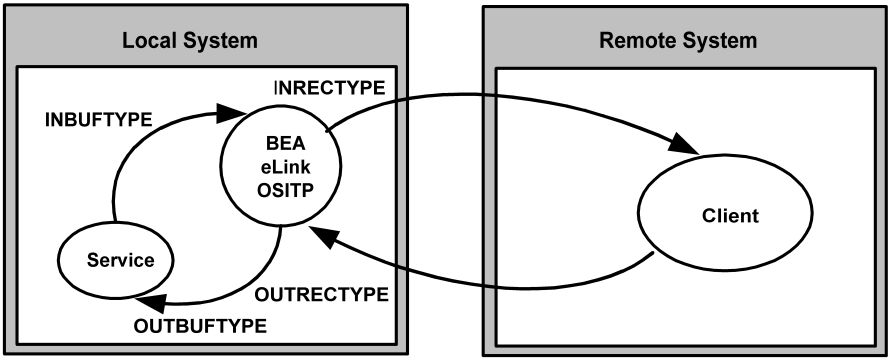
Case	Explanation
The remote service returns a reply record that is structured differently from the reply buffer the local client program expects.	The remote service returns a record that is structured differently from the client program's VIEW, X_C_TYPE, or X_COMMON buffer. For example, the structure members of the output record may be sequenced differently than the structure members of the output buffer.
The remote service returns a reply record that differs in both type and structure from the reply buffer the client program expects.	The remote service returns a particular record and the local client program expects a corresponding BEA Tuxedo FML buffer.

Parameters for Remotely Originated Calls

This section describes how eLink OSI TP handles service calls that originate on remote computers, outside the local BEA Tuxedo region. It also explains how the `INRECTYPE, INBUFTYPE, OUTBUFTYPE,` and `OUTRECTYPE` parameters can be used to manage the conversion of buffers and records that flow between remote client programs and local services.

In the following figure, a remote client program issues a service request that a remote eLink OSI TP gateway routes to the local eLink OSI TP gateway. The gateway receives the request from the network and passes the request to a local BEA Tuxedo server.

Figure 2-4 How Parameters Are Mapped During Remotely Originated Calls



In this situation, the four configuration parameters that are shown in the figure have the following meanings:

- The `OUTRECTYPE` parameter describes the record that the remote client sends to the eLink OSI TP gateway.
- The `OUTBUFTYPE` parameter describes the BEA Tuxedo buffer that is provided to the local server.
- The `INBUFTYPE` parameter describes the BEA Tuxedo buffer that the local server returns to the eLink OSI TP gateway.
- The `INRECTYPE` parameter describes the record that the local eLink OSI TP gateway returns to the remote client program.

Guidelines for Mapping Input Records to Input Buffers

This section provides detailed information explaining how to use the INRECTYPE and INBUFTYPE parameters for service calls that originate on remote systems (where remote client programs call local services).

INBUFTYPE

The INBUFTYPE parameter is used to specify the type, and in some cases the structure, of the reply buffer that the eLink OSI TP gateway expects from a local server. This restricts the buffer type naming space of data types accepted by this service to a single buffer type. Because the gateway determines the type of buffer automatically at runtime this parameter is described here for conceptual completeness only.

INRECTYPE

The INRECTYPE parameter is used to specify the type, and in some cases the format, of the reply record that the local eLink OSI TP gateway sends to the remote client. You can omit the INRECTYPE parameter if the local server program sends a reply buffer that is identical in type and structure to the reply record the remote client expects.

You must specify the INRECTYPE parameter when one of the cases described in the following table is true.

Case	Explanation
The remote client program requires a reply record that is structured differently from the reply buffer the local service provides.	The remote client program sends a record that is structured differently than the local service's VIEW, X_C_TYPE, or X_COMMON buffer. For example, the structure members of the input record may be sequenced differently than the structure members of the input buffer.
The remote client program requires a reply record that differs in both type and structure from the reply buffer the local service provides.	The remote client program requires a particular record and the local service provides a corresponding BEA Tuxedo FML buffer.

Guidelines for Mapping Output Buffers to Output Records

This section provide detailed information explaining how to use the OUTBUFTYPE and OUTRECTYPE parameters for service calls that originate on remote computers (where remote client programs call local services and receive output from those services).

OUTBUFTYPE

The OUTBUFTYPE parameter specifies the request buffer type that the local eLink OSI TP gateway provides to the local server. The eLink OSI TP gateway uses this information to convert request records from remote clients into buffers that local server programs can process.

OUTRECTYPE

The OUTRECTYPE parameter is used to specify the type, and in some cases the format, of the request record a particular remote client program sends to the eLink OSI TP gateway. The eLink OSI TP maps the incoming record to the type and subtype defined by the OUTRECTYPE parameter.

The OUTBUFTYPE parameter may be omitted if the local service's request buffer is identical, in type and structure, to the request record the remote client program provides.

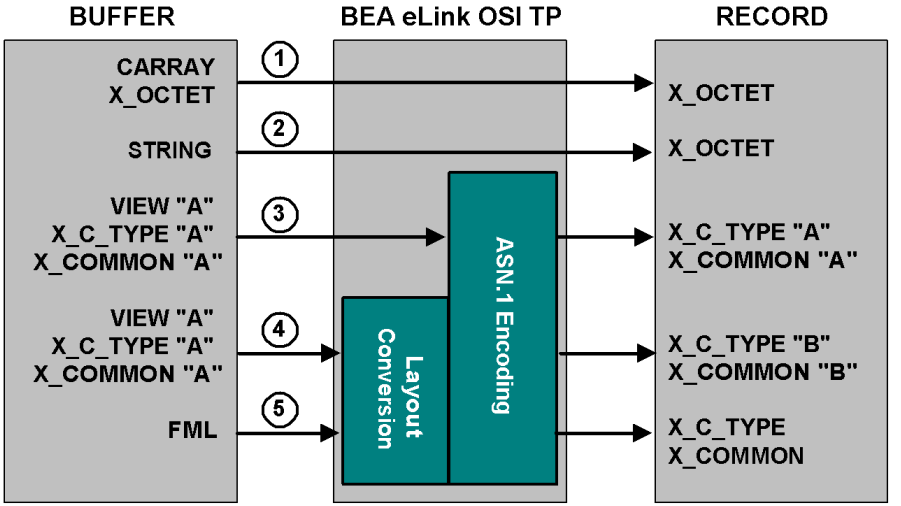
You must specify the OUTBUFTYPE parameter when one of the circumstances described in the following table is true.

Circumstance	Explanation
The remote client program provides a request record that is structured differently from the local service's request buffer.	The remote client program provides a record that is structured differently than the local service's VIEW, X_C_TYPE, or X_COMMON buffer. For example, the local server program may expect structure members to be sequenced differently.
The remote client program provides a request record that differs from the local service's request buffer in both type and structure.	The remote client outputs a record and the local service program expects a corresponding BEA Tuxedo FML buffer.

Mapping Buffers to Records

The following figure shows all the possibilities for mapping buffers to records. The eLink OSI TP gateway is responsible for mapping buffers to records based on information it finds in the eLink OSI TP configuration. This mapping occurs for Tuxedo client requests and Tuxedo server responses.

Figure 2-5 Buffer to Record Mappings



Following are explanations about the mapping possibilities shown in the figure above and some suggestions for setting the `INRECTYPE` parameter. The `INBUFTYPE` parameter is only used for verification purposes and is not discussed here.

1. BEA Tuxedo `CARRAY` input buffers can be copied to `X_OCTET` input records. A `CARRAY` buffer contains raw data that is not converted or translated. The eLink OSI TP gateway automatically sends the `CARRAY` Tuxedo buffer as `X_OCTET` to the remote system; there is no need to set the `INRECTYPE` parameter.

2. BEA Tuxedo STRING input buffers can be mapped to X_OCTET input records. No data conversion or translation is performed. The STRING buffer is copied left to right, up to and including the first NULL character encountered. The eLink OSI TP gateway automatically sends the STRING buffer as X_OCTET; there is no need to set the INRECTYPE parameter.

3. BEA Tuxedo VIEW input buffers can be mapped to X_COMMON or X_C_TYPE input records. Specify the desired record type and the name of this VIEW definition with the INRECTYPE parameter. The eLink OSI TP gateway translates the VIEW into the correct X_COMMON or X_C_TYPE input record.

4. BEA Tuxedo VIEW, X_COMMON or X_C_TYPE input buffers can be mapped to X_COMMON or X_C_TYPE input records-in any combination. However, in this situation, the data structure that the remote service expects (designated as X_COMMON 'B' mapping possibilities in Figure 3-3) differs from the data structure the client program uses (designated as VIEW 'A' in Figure 3-3). Consequently, you must

   a. Create a VIEW definition for the data structure that the remote service expects.

   b. Specify the desired record type and the name of this VIEW definition with the INRECTYPE parameter.

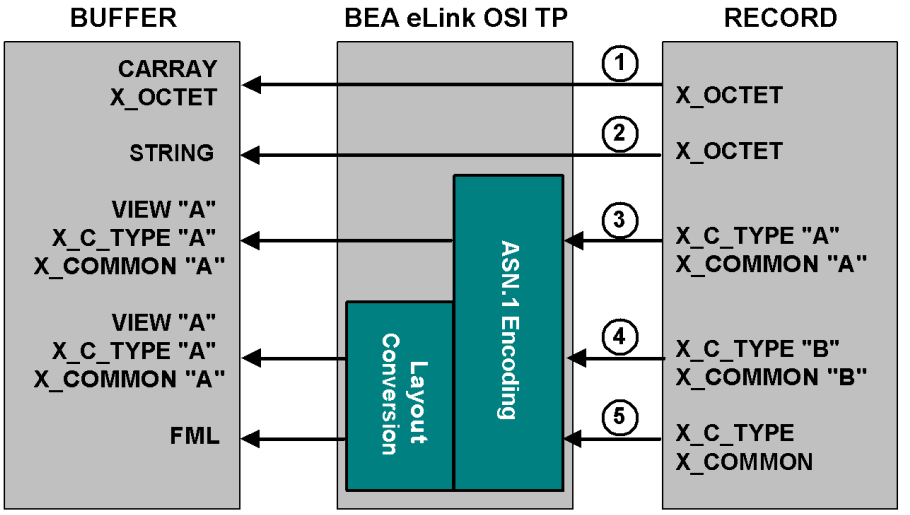
5. Before a BEA Tuxedo FML input buffer can be sent to a remote service that does not support FML, it must be mapped to one of the following input record types: X_C_TYPE or X_COMMON. Also, you must create a VIEW definition for the input data structure that the remote service expects. Set INRECTYPE to VIEW:viewname. Refer to the BEA Tuxedo online documentation for more detailed information about FML translation.

   Note: If the source and target VIEW names are different, FML fields must be specified for all VIEW to VIEW conversions that eLink OSI TP performs (For example, VIEW=V10.V --> X_C_TYPE=V10.V does not require FML mapping fields). In other words, any VIEW that is to be used in a VIEW to different VIEW, VIEW to FML, or FML to VIEW conversion must be defined with appropriate FML fields (no dashes in the FNAME column of the VIEW definition). In order for the FML fields to match, you must compile the VIEWs without the -n option specified.

Mapping Records to Buffers

The following figure shows all the possibilities for mapping records to buffers. The eLink OSI TP gateway is responsible for mapping records to buffers, based on information it finds in the eLink OSI TP configuration. This mapping occurs for remote client requests and remote server responses.

Figure 2-6 Record to Buffer Mappings



Following are explanations about the mapping possibilities shown in the figure above and some suggestions for setting the OUTBUFTYPE parameter (for service calls that originate locally. These suggestions use the OUTBUFTYPE parameter, which controls data translation.

1. Incoming X_OCTET output records can be copied to CARRAY or X_OCTET output buffers. A CARRAY buffer contains raw data that is not converted or translated. Set the OUTBUFTYPE to either CARRAY or X_OCTET; the OUTRECTYPE does not need to be set.

2. Incoming X_OCTET output records can also be copied to STRING output buffers. This creates a string that goes through no conversion and no translation. The resultant buffer is the length of the original X_OCTET buffer. Since all characters are copied, if the X_OCTET buffer contains null characters, it affects the buffer when later handled as a STRING. The OUTBUFTYPE should be set to STRING.

3. Incoming output records can be mapped to identical BEA Tuxedo VIEW output buffers. In this situation, the data structure that the remote service returns is identical to the data structure the local client program expects. There is no need to create a new VIEW definition. The OUTRECTYPE parameter can be set to VIEW:viewname, for greater type checking, but it is not mandatory.

4. Incoming X_C_TYPE and X_COMMON output records can be mapped to VIEW output buffers-in any combination. However, in this situation, the data structure that the remote service returns (designated as X_C_TYPE 'B' in Figure 2-6) differs from the data structure the client program expects (designated as VIEW 'A' in Figure 2-6). To facilitate the conversion process, perform the following tasks.

   - Create a VIEW definition for the data structure that the remote service returns.
   - If the name given to the VIEW definition is different from the name that the remote service returns (that is, ATMI buffer subtype), specify the output record type and the name of X_C_TYPE (or X_COMMON)'B' with the OUTBUFTYPE parameter. (By doing this, you override the value the eLink OSI TP requester automatically detects.)
   - Specify the output buffer type and the name of an existing view (VIEW 'A' in the figure) specified in the OUTBUFTYPE parameter.

   Note: FML Field definitions may be required to map VIEW 'B' to VIEW 'A'.

5. Incoming X_COMMON or X_C_TYPE output records can be mapped to FML output buffers. To facilitate the conversion process, you must perform the following tasks.

   - Create a VIEW definition that describes the data structure that the remote service returns.
   - If the name given to the VIEW definition is different from the name that the remote service returns (that is, the ATMI buffer subtype), specify the output record type and the name of your VIEW definition with the OUTBUFTYPE parameter. (By doing this, you override the value the eLink OSI TP requester automatically detects.)

- If verification of the FML buffer is desired, set the OUTBUFTYPE to FML or FML32 in the UDMCONFIG followed by a colon(:). (Example: OUTBUFTYPE="FML32:")

Note: FML fields must be specified for all FML to VIEW conversions that eLink OSI TP performs. In other words, any VIEW that is to be used in an FML to VIEW conversion must be defined with appropriate FML fields (no dashes in the FBNAME column of the VIEW definition). In order for the FML fields to match, you must compile the VIEWs without the -n option specified.

Special Cases and Examples of Buffer Conversion

Following are some examples of special cases and considerations for buffer conversion.

- If the incoming buffer (INBUFTYPE) or (OUTRECTYPE) = STRING, CARRAY, or X-OCTET, the conversion type must be STRING, CARRAY, or X-OCTET.
- Conversion must be between 16-bit buffers, or between 32-bit buffers. You can not convert a 16-bit buffer to a 32-bit buffer, or vice versa.

  Examples:

  a.
  ```
  OUTRECTYPE="VIEW:v10"
  OUTBUFTYPE="VIEW:v12"/* OK */
  ```

  b.
  ```
  OUTRECTYPE="VIEW:v10"
  OUTBUFTYPE="VIEW32:v13"/* ERROR */
  ```

  c.
  ```
  OUTRECTYPE="VIEW32:v13"
  OUTBUFTYPE="FML32:"/* OK */
  ```

  d.
  ```
  OUTRECTYPE="VIEW32:v13"
  OUTBUFTYPE="FML:"/* ERROR */
  ```

- If INBUFTYPE = FML / FML32,

  INRECTYPE **must** be configured.

- INRECTYPE and OUTRECTYPE cannot be FML / FML32.

```
INRECTYPE="FML:"/* ERROR */
OUTRECTYPE="FML32:"/* ERROR */
```

- For INBUFTYPE and OUTBUFTYPE, configure buffer type FML as follows:

  ```
  INBUFTYPE="FML:"
  OUTBUFTYPE="FML32:"
  ```

  Note: A colon is required at the end of keywords FML and FML32.

- INBUFTYPE and OUTRECTYPE must be configured to the same type/subtype as the incoming buffer type/subtype.
- Packed decimal type dec_t cannot be used in views participating in a conversion when either the source or the destination buffer is FML or FML32 or when the source and destination are two different views.
- For views participating in conversion, substitute view fields of type int with fields of type long or short.
- If a View member contains system default values, that member will not be transferred during buffer conversion. To force the transfer, use "NONE" in the default Null column of the source View file.

UDMCONFIG Examples for Conversion to View32 and FML

In the following example, incoming buffer X_COMMON:v10 gets converted to VIEW:v12 before the request is sent to the service.

Listing 2-1 Conversion to View32

```
*DM_REMOTE_SERVICES
TOUPPER12
RDOM=DALNT2
LDOM=DALNT19220
PRIO=66
RNAME="TOUPPER12"
INBUFTYPE="X_COMMON:v10"
INRECTYPE="VIEW:v12"
```

In the following example, incoming buffer `VIEW:v12` gets converted to FML, before the request is sent to the local service, and FML gets converted to `X_C_TYPE:v16` before the reply is returned to the remote client.

Listing 2-2 Conversion to FML

```
*DM_LOCAL_SERVICES
OUTRECTYPE=”VIEW:v12”
OUTBUFTYPE=”FML:”
INBUFTYPE=”FML:”
INRECTYPE=”X_C_TYPE:v16”
```

For more information about FML, refer to the BEA Tuxedo Online Documentation at **http://edocs.bea.com/Tuxedo.**

3 Understanding the UDMCONFIG File

Before you configure BEA eLink OSI TP and set up the gateway configuration, it is helpful to understand the `UDMCONFIG` file.

This section covers the following topics:

- Overview of the UDMCONFIG File
- UDMCONFIG File Format
- UDMCONFIG File Sections

For detailed instructions on how to configure eLink OSI TP by modifying the `UDMCONFIG` file, refer to Configuring BEA eLink OSI TP.

Overview of the UDMCONFIG File

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

`UDMCONFIG` is the ASCII version of a Tuxedo System/Domain domain configuration file. The `UDMCONFIG` file is parsed and loaded into two binary versions by the `udmloadcf` utility. The binary configuration files, called the `BDMCONFIG` and `BUDMCONFIG` files, contain information used by domain gateways to initialize the context required for communications with other domains. In its monitoring activity, `dmadmin` uses the binary file (or a copy of it). There is one `BDMCONFIG` file for each

Tuxedo System/Domain application that uses the /Domain feature. Refer to Processing a Configuration File with the Udmloadcf Utility for more information about the binary configuration files.

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

OSI TP Application Addresses Used in the UDMCONFIG File

OSI TP application address information is used for several parameters in the UDMCONFIG file. The address of a Tuxedo application using OSI TP consists of a collection of the names of each of the components described in the following table. These names must be coordinated with the remote domain OSI TP implementation.

Table 3-1 OSI TP Application Components

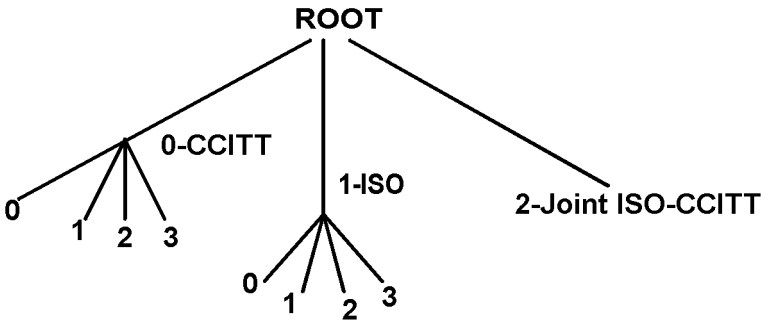
Component	Description
Application Entity Title (AET)	A dotted integer based on the ISO Object Identifier Based NameForm that uniquely identifies the OSI TP node. See the following description of the AET, Creating an Application Entity Title and Figure 3-1.
Presentation Selector (P_SEL)	A logical name for the address of the software that provides the presentation layer services for OSI protocols.
Session Selector (S_SEL)	A logical name for the address of the software that provides the session layer services for OSI protocols.
Transport Selector (T_SEL)	A logical name for the address of the software that provides the transport layer services for OSI protocols.
Network Address (NWADDR)	A globally unique computer system address used to identify the OSI TP node.

Creating an Application Entity Title

The Application Entity Title equals the APT (application process title) plus the AEQ (application entity qualifier). Each OSI TP node in your network must have a unique AET. If your site is participating in a global OSI network, you need to contact the OSI registration authority for a valid OSI Object-ID, otherwise, create your own unique AET as described below.

If your site is in a closed network, create an Object ID with at least 3 "arcs" as shown in Figure 3-1. Each arc of the dotted integer (1.3.192.59.192.213 in the example below) represents an identifier for the Object ID. A valid OSI TP Object ID has either 0 or 1 for the first arc, and 0, 1, 2, 3 for the second arc. It is recommended to use 1.3 (IP address) for the first and second arcs respectively.

Figure 3-1 Example of an Application Entity Title



Example: 1.3.192.59.192.213

Key:

0-CCITT	1-ISO
0-Recommendation	0-Standard
1-Question	1-Registration Authority
2-Administration	2-Member Body
3-Network Operator	3-Identified Organization

UDMCONFIG 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 *. Refer to UDMCONFIG File Sections for more information about the sections.
- Parameters are generally specified by: *KEYWORD* = *value*. This sets *KEYWORD* to *value*. Valid keywords are described in the following sections. *KEYWORD*s are reserved; they cannot be used as *value*s unless they are quoted.

Caution: Enter all parameters on separate lines. The `NWADDR` parameter is the only exception; it may list multiple network address values in a single line.

- Lines beginning with the reserved word, DEFAULT:, contain parameter specifications that apply to all the following lines in the section in which they appear. Default specifications can be used in any of the sections. Defaults 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 by specifying another value for the parameter on a non-DEFAULT: line. The override 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 in 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 included in a string by preceding them with 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

UDMCONFIG File Sections

The `UDMCONFIG` file consists of the following sections and parameters that define new gateway configurations.

- DM_LOCAL_DOMAINS Section

  Note: The `DM_LOCAL_DOMAINS` section must precede the `DM_REMOTE_DOMAINS` section.
- DM_REMOTE_DOMAINS Section
- DM_OSITPX Section
- DM_ACCESS_CONTROL Section

- DM_LOCAL_SERVICES Section
- DM_REMOTE_SERVICES Section
- DM_ROUTING Section

Following is a sample configuration file and detailed descriptions of the UDMCONFIG file sections and the parameters applicable to each section.

Sample Configuration File

Listing 3-1 Sample UDMCONFIG File

```
*DM_LOCAL_DOMAINS

dalnt8
GWGRP       = OSIGRP
TYPE        = OSITPX
DOMAINID    = "dalnt8"
BLOCKTIME   = 30
DMTLOGDEV   = "D:\tuxedo\log\DMLOG"
SECURITY    = DM_PW  # turns link layer security on


*DM_REMOTE_DOMAINS

dal2200 TYPE=OSITPX DOMAINID="dal2200"
openti  TYPE=OSITPX DOMAINID="openti"
icl2    TYPE=OSITPX DOMAINID="icl2"
aseries1 TYPE=OSITPX DOMAINID="aseries1"

*DM_OSITPX

dalnt8
   AET="{1.3.132.61.146},{3}"
   TAILOR_PATH="d:\tuxedo\configs\tailor.txt"
   NWADDR="//dalnt8:102"
   DNS_RESOLUTION=STARTUP # this is the default

dal2200
   AET="{1.3.132.61.46},{3}"
   XATMI_ENCODING="OLTP_TM2200"
   NWADDR="132.61.146.3";"132.61.147.1" #redundant IP addresses
   T_SEL="OSITP"
```

```
openti
   AET="{1.3.122.62.103},{209}"
   NWADDR="122.62.103.209:2001"
   OPTIONS=SECURITY_SUPPORTED

icl2
   AET="{1.3.142.60.203},{4}"
   NWADDR="142.60.203.4"
   T_SEL="ICLTP"
   S_SEL="SSEL"
   P_SEL="PSEL"

aseries1
   AET="{1.3.123.55.222},{51}"
   NWADDR="123.55.222.51"
   XATMI_ENCODING="PRELIMINARY"
   T_SEL="0x5453"
   S_SEL="0x3F5C3F"

*DM_ACCESS_CONTROL
mylist  ACLIST = dalnt8, dal2200

*DM_LOCAL_SERVICES
TOUPPERF
  INRECTYPE="VIEW:view10"
  OUTBUFTYPE="FML:"
  COUPLING=LOOSE   #this is the default

TOUPPERF32
  INRECTYPE="VIEW:view10a"
  OUTBUFTYPE="FML32:"
  COUPLING=TIGHT


TOUPPERV
  INBUFTYPE="X_C_TYPE:v10"
  INRECTYPE="VIEW:upper"
  COUPLING=LOOSE

TOUPPERC OUTRECTYPE="X_OCTET" OUTBUFTYPE="CARRAY"
        INRECTYPE="X_OCTET"
        COUPLING=TIGHT

TOUPPERS  OUTRECTYPE="X_OCTET" OUTBUFTYPE="STRING"
        INRECTYPE="X_OCTET"

TOUPPERX  OUTRECTYPE="STRING" OUTBUFTYPE="STRING"
         INRECTYPE="X_OCTET"
```

```
*DM_REMOTE_SERVICES
DEFAULT: TRANTIME=300

ECHOXOCT RNAME="ECHOSRVR" OUTBUFTYPE="X_COMMON:ECHOVIEW"
RDOM=dal2200 LDOM=dalnt8
ECHOXCOM RNAME="ECHOSRVR" RDOM=openti LDOM=dalnt8 AUTOPREPARE=Y

ECHOXCTYPE RNAME="ECHOSRVR"
         INBUFTYPE="X_C_TYPE:ECHOVIEW"
         INRECTYPE="X_COMMON:ECHOVIEW"
         OUTBUFTYPE="X_C_TYPE:ECHOVIEW"
         OUTRECTYPE="X_COMMON:ECHOVIEW"
         RDOM=aseries1
         LDOM=dalnt8
         CONV=Y
ECHOVIEW RNAME="ECHOSRVR"
         INBUFTYPE="VIEW:ECHOVIEW"
         INRECTYPE="X_COMMON:ECHOVIEW"
         OUTBUFTYPE="VIEW:ECHOVIEW"
         OUTRECTYPE="X_COMMON:ECHOVIEW"
         RDOM=icl2
         LDOM=dalnt8
         REM_TPSUT="tpmvs"

*DM_ROUTING
ACCOUNT FIELD = branchid BUFTYPE = "View:account"
        RANGE = "MIN - 1000:aseries1, 1001-3000:openti, *:dal2200"
```

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 described in the DM_LOCAL_SERVICES section, *LDOM* is the identifier that connects local services with a particular gateway group.

Valid Parameters

Following is a list of valid parameters for the DM_LOCAL_DOMAINS section:

Parameter	Required/ Optional	Description
AUDITLOG	Optional	Name of the audit log
BLOCKTIME	Optional	Maximum wait time allowed for a blocking call
DMTLOGDEV	Optional	Tuxedo file system that contains the domain transaction log
DMTLOGNAME	Optional	Name of the domain transaction log
DMTLOGSIZE	Optional	Size of the domain transaction log
DOMAINID	Required	Local domain
GWGRP	Required	Name of gateway server group
MAXRDTRAN	Optional	Maximum number of remote domains that can be involved in a transaction
MAXTRAN	Optional	Maximum number of simultaneous global transactions allowed on local domain
SECURITY	Optional	Link-level of security for local domain
TYPE	Required	Classification of local domain

Parameter Definitions

Following is more detailed information about each of the DM_LOCAL_DOMAINS section parameters:

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. `BLOCKTIME` may need to be increased due to remote network latency or if security is turned on. 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` is always used when the request is issued within a transaction.

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.

DMTLOGNAME = *"string"*

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.

DOMAINID = *"string"*

identifies 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 31 characters or fewer.

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.

MAXRDTRAN = *numeric*

specifies the maximum number of remote 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. MAXGTT is the maximum number of transactions for all the domains on a given machine. If not specified, the default is the value of MAXGTT.

SECURITY = {NONE | DM_PW}

specifies whether link-level security for the local domain is turned on. NONE is the default and it indicates that no security is used. When this parameter is set to DM_PW, incoming connections from remote domains are authenticated using the passwords defined in the *DM_PASSWORDS section of the BDMCONFIG file.

Note: This parameter must appear **AFTER** the TYPE=OSITPX parameter.

TYPE = *identifier*

groups local domains into classes. TYPE can be set to one of the following values: TDOMAIN or OSITPX. The TDOMAIN value indicates that this local domain can only communicate with another Tuxedo System/Domain. The OSITPX 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. The type, OSITPX, uses a DMTYPE of

OSITP. The eLink OSI TP install automatically updates the DMTYPE file with the required type needed.

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.

Valid Parameters

Following is a list of valid parameters for the DM_REMOTE_DOMAINS section:

Parameter	Required/ Optional	Description
DOMAINID	Required	ID of remote domain.
TYPE	Required	Class of remote domain

Parameter Definitions

Following is more detailed information about each of the DM_REMOTE_DOMAINS section parameters:

DOMAINID = *"string"*

identifies 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".

TYPE = *identifier*
groups remote domains into classes. `TYPE` can be set to one of the following values: `TDOMAIN` or `OSITPX`. The `TDOMAIN` value indicates that this remote domain can only communicate with another Tuxedo System/Domain Domain. The `OSITPX` value indicates that this remote domain communicates with another TP domain via the OSI TP protocol.

DM_OSITPX Section

This section defines the addressing information required by domains of type `OSITPX`. This section should have at least one entry per gateway group (local domain), and at least one entry per remote domain of type `OSITPX`. The bridged configuration can have multiple gateways in a local domain.

Format

`DM_OSITPX` entries have the following format.

DOM required parameters [*optional parameters*]

where

DOM is an *identifier* value used to identify 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.

Valid Parameters

Following is a list of valid parameters for the `DM_OSITPX` section:

Parameter	Required/ Optional	Description
AET	Required for LDOMS and RDOMS	Application entity title
DNS_RESOLUTION	Optional for RDOMS	Indicator for when DNS name is resolved
NWADDR	Required for LDOMS and RDOMS	List of IP addresses with their optional port numbers or a DNS name and its optional port number
OPTIONS	Optional for RDOMS	Optional flags to turn on OSI TP features such as security
P_SEL	Optional for LDOMS and RDOMS	Logical name of address for software that provides presentation layer services
S_SEL	Optional for LDOMS and RDOMS	Logical name of address for software that provides session layer services
T_SEL	Strongly recommended for LDOMS and optional for RDOMS	Logical name of address for software that provides transport layer services
TAILOR_PATH	Optional for LDOMS	Path to optional OSI TP tailor file
XATMI_ENCODING	Optional for RDOMS	Version of XATMI protocol

Parameter Definitions

Following is more detailed information about each of the DM_OSITPX section parameters:

AET = "*string*"

indicates the application entity title that this `LDOM` or `RDOM` uses. This address must be unique among all hosts communicating in the OSI TP network. This number matches the local AE Title on the remote (OLTP) node. Refer to OSI TP Domains Components for more information about AETs.

The format accepted for the value of *string* is
"{*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, for example,
`AET = "{1.3.15.0.3},{1}"`.

Note: The braces are part of the syntax and **must** be included within the quotes.

DNS_RESOLUTION = {STARTUP | RUNTIME}

indicates whether the DNS name should be resolved when the gateway is started or at runtime. The DNS name is for the network address defined by `NWADDR`. The runtime option allows support of DHCP networks. When using `DNS_RESOLUTION` as a runtime option, failed services may occur due to delays in resolving the DNS names. The default is `STARTUP`.

NWADDR = "*string*"

indicates the network address that this `LDOM` or `RDOM` uses and, optionally, the port number. The network address may be either an IP address, if using TCP|IP networks, or a DNS name. The default port number is port 102. For local domains, the `NWADDR` specifies which IP address eLink OSI TP will listen on. For remote domains, the `NWADDR` specifies which network messages will be sent on. You may list multiple network addresses by listing each individual address separated by semicolons if the machine is equipped with multiple network cards. Make sure to enter all the IP addresses on one line and separate them with a semi-colon (;). You may wish to configure redundant network paths: up to 8 may be specified.

Examples:

`"#.#.#.#:port-number"`	IP Address
`"//host-name:port-number"`	DNS Name

OPTIONS = SECURITY_SUPPORTED

indicates optional parameters for RDOMs. The `SECURITY_SUPPORTED` value indicates that this remote domain supports the OSITP security extension. This

value provides backward compatibility and is valid only when describing an RDOM.

P_SEL = "*string*" or "`hex digits`"

specifies the logical name for the address of the software that provides the presentation layer services for OSI protocols. The value can be one to 4 ASCII non-control characters (those represented by the hexadecimal numbers 20 to 7E), one to 4 hexadecimal octets, or `NONE` (null). A value of `NONE` is the default. Examples: `"PSEL", "0x3F5C'`

S_SEL = "*string*" or "`hex digits`"

specifies the logical name for the address of the software that provides the session layer services for OSI protocols. The value can be one to 16 ASCII non-control characters (those represented by the hexadecimal numbers 20 to 7E), one to 16 hexadecimal octets, or `NONE` (null). A value of `NONE` is the default. Examples: `"SSEL", "0x3F5C3F'`

T_SEL = "*string*" or "`hex digits`"

represents the logical name for the address of the software that provides the transport layer services for OSI protocols. The value can be one to 32 ASCII non-control characters (those represented by the hexadecimal numbers 20 to 7E), one to 32 hexadecimal octets, or `NONE` (null). Examples: `"OSITP", "0x5453"`

TAILOR_PATH = "*string*"

indicates the full path to the optional OSI TP tailor file used for tuning OSI TP-specific tables. Double quotes are required. If not specified, preset defaults are used. This parameter is valid only when describing an LDOM. Refer to Tuning OSI TP-Specific Tables with the TAILOR File for more information.

XATMI_ENCODING = {CAE | PRELIMINARY | OLTP_TM2200}

specifies the version of the XATMI protocol used to communicate with a remote application. This parameter is only valid for an RDOM. Valid values are:

`CAE` (default)
`PRELIMINARY` (used specifically with Unisys A-series OLTP)
`OLTP_TM2200`

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.

Valid Parameters

Following is a list of valid parameters for the DM_ACCESS_CONTROL section:

Parameter	Required/ Optional	Description
ACLIST	Required	List of remote domain names

Parameter Definitions

Following is more detailed information about the DM_ACCESS_CONTROL section parameter:

ACLIST = *identifier* [,*identifier*]

indicates 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.

Valid Parameters

Following is a list of valid parameters for the DM_LOCAL_SEVICES section:

Parameter	Required/ Optional	Description
ACL	Optional	Name of access control list
COUPLING	Optional	Indicator for type of coupling
INBUFTYPE	Optional	Type and subtype of buffer returned by local service
INRECTYPE	Optional	Type and format of the reply buffer expected by remote client

Parameter	Required/ Optional	Description
LDOM	Optional	Name of local domain exporting a service
OUTRECTYPE	Optional	Type and format of request buffer expected by local service
OUTBUFTYPE	Optional	Type and format of request buffer expected by local service
RNAME	Optional	Name of service exported to remote domains

Parameter Definitions

Following is more detailed information about the DM_LOCAL_SEVICES section parameters:

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 is not performed for requests to this service.

COUPLING = {TIGHT | LOOSE}

specifies service (transaction) coupling to be tight or loose when requests for this local service come from the same remote domain. The default is LOOSE. This means data base updates made by the first request to this local service cannot be seen by the second request to the local service even though they are involved in the same global transaction. By making this value TIGHT, multiple calls to the same service from the same domain are tightly-coupled. Data base updates made by the first request can be seen by the second request. This option is only available when duplicate service requests come from the same RDOM. When the service requests are from different RDOMs, the requests are always loosely-coupled.

INBUFTYPE = type[:subtype]

specifies the type and subtype of the buffer. INBUFTYPE is used to enforce stronger type checking. In the DM_LOCAL_SERVICES section, the TYPE parameters are defined in reference to where the remote request originates.

Refer to Managing Parameters for Buffer and Record Conversion for more information about these parameters.

INRECTYPE = type[:subtype]

specifies the type, and in some cases, the format of the reply buffer that a particular client requires. This parameter can be omitted if the local service sends a buffer that is identical in type and structure to the buffer the remote client expects. If you do not specify `INRECTYPE`, the type of buffer is unchanged. In the `DM_LOCAL_SERVICES` section, the `TYPE` parameters are defined in reference to where the remote request originates. Refer to Managing Parameters for Buffer and Record Conversion for more information about these parameters.

LDOM = identifier

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

OUTBUFTYPE = type[:subtype]

specifies the type, and in some cases, the format of the request buffer that a particular local service expects. This parameter can be omitted if the remote client sends a buffer that is identical in type and structure to the buffer the local service expects. If you do no specify `OUTRECTYPE`, the type of the buffer is unchanged. In the `DM_LOCAL_SERVICES` section, the `TYPE` parameters are defined in reference to where the remote request originates. Refer to Managing Parameters for Buffer and Record Conversion for more information about these parameters.

OUTRECTYPE = type[:subtype]

specifies the type and subtype of the buffer sent by the remote client. This parameter is used to enforce stronger type checking. In the `DM_LOCAL_SERVICES` section, the `TYPE` parameters are defined in reference to where the remote request originates. Refer to Managing Parameters for Buffer and Record Conversion for more information about these parameters.

RNAME = “string”

specifies the name of the service exported to remote domains. This name is used by the remote domains for request to this service. If this parameter is not specified, the local service name is used for the request.

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.

Valid Parameters

Following is a list of valid parameters for the DM_REMOTE_SERVICES section:

Parameter	Required/ Optional	Description
AUTOPREPARE	Optional	Indicator to automatically prepare calls from tpcall
CONV	Optional	Indicator that remote service is conversational
INBUFTYPE	Optional	Type and subtype of buffer sent to the remote service
INRECTYPE	Optional	Type and format of the request buffer expected by remote service
LDOM	Optional	Name of local domain exporting a service
OUTRECTYPE	Optional	Type and format of reply buffer returned by remote client
OUTBUFTYPE	Optional	Type and format of reply buffer expected by local client

Parameter	Required/ Optional	Description
RDOM	Optional	Name of remote domain responsible for execution of service
REM_TPSUT	Optional	TP service user title
RNAME	Optional	Name of service exported to remote domains
ROUTING	Optional	Routing criteria used for data-dependent routing
TPSUT_TYPE	Optional	Type for which the remote TP service user title is to be encoded.
TRANTIME	Optional	Default time-out value in seconds for transaction automatically started for associated service

Parameter Definitions

Following is more detailed information about the DM_REMOTE_SERVICES section parameters:

AUTOPREPARE = {N | Y}

allows a single tpcall() involved in a global transaction to this remote service to automatically prepare the call. This optimization reduces the two-phase commit process to a single step. The remote OSITP domain must support this feature. The default is N.

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.

INBUFTYPE = type[:subtype]

specifies the type and subtype of the buffer allocated by the client. This parameter is used to enforce stronger type checking. In the DM_REMOTE_SERVICES section, the TYPE parameters are defined in reference to where the local request originates. Refer to Managing Parameters for Buffer and Record Conversion for more information about these parameters.

INRECTYPE = type[:subtype]

specifies the type, and in some cases, the format of the request buffer that a particular remote service requires. This parameter can be omitted if the local client sends a buffer that is identical in type and structure to the buffer the remote service expects. If you do not specify `INRECTYPE`, the type of buffer is unchanged. In the `DM_REMOTE_SERVICES` section, the `TYPE` parameters are defined in reference to where the local request originates. Refer to Managing Parameters for Buffer and Record Conversion for more information about these parameters.

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 are able to accept requests to this remote service. The service request is then redirected to a remote domain of the same type (see the following definition for `RDOM` keyword).

OUTBUFTYPE = type[:subtype]

specifies the type, and in some cases, the format of the reply buffer that a particular local client expects. This parameter can be omitted if the remote service returns a buffer that is identical in type and structure to the buffer the local client expects. If you do no specify `OUTRECTYPE`, the type of the buffer is unchanged. In the `DM_REMOTE_SERVICES` section, the `TYPE` parameters are defined in reference to where the local request originates. Refer to Managing Parameters for Buffer and Record Conversion for more information about these parameters.

OUTRECTYPE = type[:subtype]

specifies the type and subtype of the buffer sent by the remote service. This parameter is used to enforce stronger type checking. In the `DM_REMOTE_SERVICES` section, the `TYPE` parameters are defined in reference to where the local request originates. Refer to Managing Parameters for Buffer and Record Conversion for more information about these parameters.

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.

REM_TPSUT = {INTEGER | PRINTABLESTRING}

identifies the TP service user title for the remote system. Some users of OSI TP implementations require this field. It is not required for OS 2200 OLTP-TM2200, OpenTI, A Series Open/OLTP, or BEA eLink OSI TP. If the value is a PRINTABLESTRING, the maximum length is 60 characters. If the value is an INTEGER, the maximum length must fit into a LONG.

RNAME = *identifier*

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

ROUTING = *identifier*

allows a local domain to perform data-dependent routing when more than one remote domain offers the same service. 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.

TPSUT_TYPE = {INTEGER | PRINTABLESTRING}

specifies the type of encoding to be performed on the REM_TPSUT parameter. The default type is PRINTABLESTRING. If TPSUT_TYPE is not specified, the default is used. The INTEGER and PRINTABLESTRING are ASN.1 types.

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.

Valid Parameters

Following is a list of valid parameters for the DM_ROUTING section:

Parameter	Required/ Optional	Description
BUFTYPE	Required	Types and subtypes of data buffers for which this routing entry is valid
FIELD	Required	Name of routing field
RANGES	Required	Ranges and associated remote domain names (RDOM) for routing field

Parameter Definitions

Following is more detailed information about the DM_ROUTING section parameters:

BUFTYPE = *~type1*[*:subtype1{,subtype2* . . .]][;*type2{:subtype3*[, . . .]]] . ..~*
specifies 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.

For `FML` buffers, if the field value is not set 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.

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.

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 are 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. An 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 ":".

4 Configuring BEA eLink OSI TP

After the installation of BEA eLink OSI TP is complete, you must configure the software. The proper configuration of eLink OSI TP sets up the gateway configuration.

This section covers the following topics:

Configuration Prerequisites

The eLink 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 `UDMCONFIG` file.

In addition, the following prerequisites must be met to successfully complete the configuration procedure:

- The `$TUXDIR/udataobj/DMTYPE` file defining the valid domain types must exist so the `udmloadcf` utility can load the binary configuration file and must contain a domain type of `OSITP`. During the installation process if the `DMTYPE` file does not contain an OSI TP entry, the `DMTYPE` file is automatically updated with the required OSITP domain type.
- The effective user identifier of the person running `udmloadcf` must match the `UID` in the `RESOURCES` section of the `TUXCONFIG` file.

Setting Environment Variables

Before you can invoke system commands, you must set several system environment variables. Most of the environment variables required by BEA eLink OSI TP are set when you set up Tuxedo. Refer to your Tuxedo documentation for more information about setting the Tuxedo environment variables.

In addition to the Tuxedo environment variables, eLink OSI TP requires two additional environment variables, `OSIRUNDIR` and `GW_DFLT_TRANTIME`.

You must set `OSIRUNDIR`, before you can boot the gateway or run the `osiadmin` utility. If you do not set the `OSIRUNDIR` environment variable before you boot the gateway, you will receive a message telling you to set `OSIRUNDIR`. This environment variable specifies the path that the eLink gateway uses for runtime files. You can set the `OSIRUNDIR` environment variable through a script, a command line entry, or through the Windows System Properties in the Control Panel. The variable value should include the path and directory as appropriate for your operating system. If the directory does not exist, the system will create it for you.

The default transaction time on the server is determined at startup by an optional environment variable call `GW_DFLT_TRANTIME`. If you do not set this variable, the default value is 5 minutes (300 seconds). This environment variable can be set to a different value at startup, but if the value exceeds the maximum allowed for a `LONG`, then the value is reset to 300 and a `LIBGWO_CAT msg 2204` is sent to the `ULOG` to indicate that the maximum has been exceeded.

Note: The maximum for a `LONG` is `2147483647`.

Defining Gateway Configurations

Whether you are defining a new gateway configuration or modifying an existing one, both processes are similar. Defining a gateway configuration requires the following steps:

1. Define the eLink OSI TP servers in Tuxedo so that the BEA eLink system can recognize the eLink OSI TP administrative and gateway servers. Refer to Defining eLink OSI TP Servers for BEA Tuxedo for more information.
2. Run the `osiadmin` processor if you are upgrading from a previous version of eLink OSI TP. Refer to Using the OSI TP Administration Utility for more information.
3. Determine which parameter information needs to be added or changed to define the gateway. Refer to Understanding the UDMCONFIG File for more information.
4. Edit the OSI TP Tailor file if OSI TP-specific tables need to be tuned. Refer to Tuning OSI TP-Specific Tables with the TAILOR File for more information.
5. Create or edit the `UDMCONFIG` file with new gateway information. Refer to Editing the UDMCONFIG File and Understanding the UDMCONFIG File for more information.
6. Generate a binary version of the `UDMCONFIG` file by running the `udmloadcf` utility. Refer to Processing a Configuration File with the Udmloadcf Utility for more information.

After you perform these steps, you are ready to start the gateway using the Tuxedo `tmboot` command. Refer to the *BEA Tuxedo Online Documentation* for more information about Tuxedo commands.

Defining eLink OSI TP Servers for BEA Tuxedo

To establish a gateway configuration, the BEA Tuxedo system must recognize the eLink OSI TP administrative and gateway servers, DMADM, UDMADM, GWADM, and GWOSITP. You define the eLink OSI TP administrative and gateway servers to the BEA Tuxedo system by editing the UBBCONFIG file.

Perform the following steps to define eLink OSI TP servers for BEA Tuxedo:

1. In the GROUPS section of the UBBCONFIG file, add a server group using the following format:

   ```
   OSIGRP  GRPNO=1  LMID=SITE1
   ```

Note: OSIGRP is used as an example. You may give the group any name you wish.

2. In the SERVERS section of the UBBCONFIG file, add the four server names: DMADM, UDMADM, GWADM, and GWOSITP.

Notes: The DMADM, UDMADM, and GWADM entries should be placed in this order **BEFORE** GWOSITP in the UBBCONFIG file so the admin servers are loaded before the GWOSITP gateway server. UDMADM is optional in a single-host machine environment, but you will receive some warning messages in the ulog file. Refer to Utilities Reference for more detailed information about UDMADM and the other the admin servers.

It is recommended that you set the RESTART parameter in the SERVERS section to Y so that the gateway will automatically restart in case of failure.

Sample UBBCONFIG File

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

Listing 4-1 Sample UBBCONFIG File

```
#----------------------------------
# Connect OSI Test; Client ubbconfig
#----------------------------------

*RESOURCES
#---------------
# Replace IPCKEY
#---------------
IPCKEY       52029
MASTER       SITE1
DOMAINID     FRONTEND
PERM         0660
MAXACCESSERS 40
MAXSERVERS   80
MAXSERVICES  80
MAXCONV      120
MODEL        SHM
LDBAL        Y
MAXGTT       120
MAXBUFTYPE   16
MAXBUFSTYPE  32
SCANUNIT     5
SANITYSCAN   10
DBBLWAIT     5
BBLQUERY     50
BLOCKTIME    15

*MACHINES
#--------------------
# Replace machine name
#--------------------
DALNT45      LMID=SITE1
#----------------------------
# Replace directories as needed
#----------------------------
  TUXDIR="c:\tuxedo"
  APPDIR="D:\dwh\base\ositp\test\client"
  TUXCONFIG="D:\dwh\base\ositp\test\client\tuxconfig"
  TLOGDEVICE="D:\dwh\base\ositp\test\client\TLOG"
  TLOGNAME="TLOG"

*GROUPS
OSIGRP2      GRPNO=2 LMID=SITE1
OSIGRP3      GRPNO=3 LMID=SITE1 TMSNAME="TMS" TMSCOUNT=2

*SERVERS
```

```
DEFAULT: RESTART=Y
DMADM      SRVID=101 SRVGRP=OSIGRP2 CLOPT="-A"
GWADM      SRVID=103 SRVGRP=OSIGRP2 CLOPT="-A"
GWOSITP    SRVID=104 SRVGRP=OSIGRP2 CLOPT="-A" GRACE=0
CRPCSERV   SRVID=8 SRVGRP=OSIGRP3 CLOPT="-A" RQADDR="rpcq"
CCONVSRV   SRVID=9 SRVGRP=OSIGRP3 CLOPT="-A" RQADDR="convq "CONV=Y
               MIN=3 MAX=5

*SERVICES
DEFAULT:      LOAD=50 AUTOTRAN=N
CTOUPPER      PRIO=50
CCONVRTN      PRIO=50
CCONVRTN2     PRIO=50
CCONVRTN3     PRIO=50
CTOUPPER2      PRIO=50
```

Refer to the *BEA Tuxedo Reference Manual* for additional information about the `UBBCONFIG` file.

Example of a Multiple Gateway Configuration

Following is a sample `UBBCONFIG` file and corresponding `UDMCONFIG` file for multiple gateways that reside on the same physical system. Note that use of `LDBAL=Y` in the `UBBCONFIG` file is not required for multiple gateways. Loads are automatically balanced for multiple gateways.

Listing 4-2 UBBCONFIG for Multiple Gateways

```
*RESOURCES
IPCKEY        65952
MASTER        “SITE1”
MODEL         SHM
PERM          0660
LDBAL         N   # not needed for gateway load balancing
MAXACCESSERS  40
MAXSERVERS    80
MAXSERVICES   80
MAXGTT        120
SCANUNIT      5
SANITYSCAN    10
```

```
BLOCKTIME     15
MAXCONV       120

*MACHINES
"SITE1"       LMID="SITE1"

              TUXCONFIG="D:\tuxedo\configs\TUXCONFIG"
              TUXDIR="D:\tuxedo"
              APPDIR="D:\tuxedo\apps"
              TLOGDEVICE="D:\tuxedo\log\TLOG"
              ULOGPFX="D:\tuxedo\log\ULOG"
              TLOGNAME=TLOG
              TLOGSIZE=20
              TYPE="SITE1"
*GROUPS

GROUP1        LMID="SITE1"
              GRPNO=1
GROUP2        LMID="SITE1"
              GRPNO=2

DMGRP         LMID="SITE1"
              GRPNO=3

*SERVERS

DEFAULT:      RQPERM=0666
              RPPERM=0666
              MIN=1
              MAX=1
              CONV=N
              MAXGEN=1
              GRACE=86400
              RESTART=Y
              SYSTEM_ACCESS=FASTPATH


DMADM         SRVGRP=DMGRP
              SRVID=20
              CLOPT="-A"
              RESTART=Y
              MAXGEN=2

GWADM         SRVGRP=GROUP1
              SRVID=21
              CLOPT="-A"
              RESTART=Y
              MAXGEN=2
```

```
GWOSITP         SRVGRP=GROUP1
                SRVID=22
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2

GWADM           SRVGRP=GROUP2
                SRVID=23
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2

GWOSITP         SRVGRP=GROUP2
                SRVID=24
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2


*SERVICES

DEFAULT:  LOAD=50 AUTOTRAN=N
```

Listing 4-3 UDMCONFIG File for Multiple Gateways

```
*DM_RESOURCES
VERSION="SITE1"
#
*DM_LOCAL_DOMAINS

"GW-1"
GWGRP       = GROUP1
TYPE        = OSITPX
DOMAINID    = "GW-1"
BLOCKTIME   = 30
DMTLOGDEV   = "D:\tuxedo\log\DMLOG"

"GW-2"
GWGRP       = GROUP2
TYPE        = OSITPX
DOMAINID    = "GW-2"
BLOCKTIME   = 30
DMTLOGDEV   = "D:\tuxedo\log\DMLOG2"

########################################################################
```

```
*DM_REMOTE_DOMAINS

DEFAULT:

"OPENTI" TYPE="OSITPX" DOMAINID="OPENTI"

#################################################################
*DM_OSITPX

"GW-1"
      AET="{1.3.145.62.103},{2}"
      TAILOR_PATH="d:\tuxedo\configs\tailor.txt"
                              # Inserted from OSITP's config file:
      NWADDR="//SITE1:102"

"GW-2"
      AET="{1.3.145.62.103},{3}"
      TAILOR_PATH="d:\tuxedo\configs\tailor2.txt"
                              # Inserted from OSITP's config file:
      NWADDR="//SITE1:2000" #second gateway must use another
                                    #IP or different port number

"OPENTI"
      AET="{1.3.122.61.203},{20}"
      NWADDR="122.61.203.20"


*DM_LOCAL_SERVICES


*DM_REMOTE_SERVICES
DEFAULT: TRANTIME=300
    # TUXEDO will alternate outgoing calls between the two LDOMs.

callSvc2   RDOM="OPENTI" LDOM="GW-1" PRIO=66
callSvc2   RDOM="OPENTI" LDOM="GW-2" PRIO=66
```

Using the Tuxedo MP Model with the eLink OSI TP Gateway

It is useful to use the Tuxedo MP model (for multiprocessors that do not have global shared memory or for networked applications) when you require two eLink OSI TP systems to exist in the same domain. (Refer to the BEA Tuxedo documentation for

more information about the MODEL parameter.) A practical example of this is setting up a Windows NT cluster. The eLink OSI TP gateway supports active-active failover on an NT cluster. In the MP model case, there are two unique nodes, one defined as the master and a second one defined as a slave or backup system in the case of clustering. There is one UBBCONFIG and one UDMCONFIG that physically exist on the master node. At TMBOOT time, a copy of the TUXCONFIG is propagated to the slave or backup system. When the backup system is booted and the eLink gateway requires configuration information, the UDMADM service retrieves the configuration information from the master system.

Listing 4-4 UBBCONFIG File for MP Model

```
UBBCONFIG

*RESOURCES
IPCKEY        65952
MASTER        SITE1,SITE2
MODEL         MP
OPTIONS       LAN
PERM          0660
LDBAL         N   # not needed for gateway load balancing
MAXACCESSERS  40
MAXSERVERS    80
MAXSERVICES   80
MAXGTT        120
SCANUNIT      5
SANITYSCAN    10
BLOCKTIME     15
MAXCONV       120


*MACHINES
"SITE1"       LMID="SITE1"
              TUXCONFIG="D:\tuxedo\configs\TUXCONFIG"
              TUXDIR="D:\tuxedo"
              APPDIR="D:\tuxedo\apps"
              TLOGDEVICE="D:\tuxedo\log\TLOG"
              ULOGPFX="D:\tuxedo\log\ULOG"
              TLOGNAME=TLOG
              TLOGSIZE=20
              TYPE="INTEL"

"SITE2"       LMID="SITE2"
              TUXCONFIG="D:\tuxedo\configs\TUXCONFIG"
              TUXDIR="D:\tuxedo"
```

```
                APPDIR="D:\tuxedo\apps"
                TLOGDEVICE="D:\tuxedo\log\TLOG"
                ULOGPFX="D:\tuxedo\log\ULOG"
                TLOGNAME=TLOG
                TLOGSIZE=20
                TYPE="INTEL"


*GROUPS
GROUP1          LMID="SITE1"
                GRPNO=1
GROUP2          LMID="SITE2"
                GRPNO=2
DMGRP           LMID="SITE1"
                GRPNO=3


*NETWORK
SITE1           NADDR="/SITE1:5020"
                NLSADDR="//SITE1:5021"
SITE2           NADDR="//SITE2:5020"
                NLSADDR="//SITE2:5021"


*SERVERS
DEFAULT:        RQPERM=0666
                REPLYQ=Y
                RPPERM=0666
                MIN=1
                MAX=1
                CONV=N
                MAXGEN=1
                GRACE=86400
                RESTART=N
                SYSTEM_ACCESS=FASTPATH


DMADM           SRVGRP=DMGRP
                SRVID=20
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2

UDMADM          SRVGRP=DMGRP
                SRVID=21
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2
```

```
GWADM           SRVGRP=GROUP1
                SRVID=22
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2

GWOSITP         SRVGRP=GROUP1
                SRVID=23
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2

GWADM           SRVGRP=GROUP2
                SRVID=24
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2

GWOSITP         SRVGRP=GROUP2
                SRVID=25
                CLOPT="-A"
                RESTART=Y
                MAXGEN=2


*SERVICES
DEFAULT:  LOAD=50 AUTOTRAN=N
```

Listing 4-5 UDMCONFIG File for MP Model

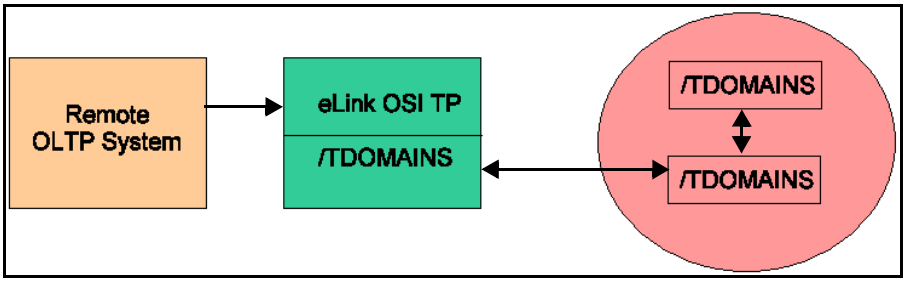
```
UDMCONFIG

*DM_RESOURCES
VERSION="SITE1"
#
*DM_LOCAL_DOMAINS

"GW-1"
GWGRP       = GROUP1
TYPE        = OSITPX
DOMAINID    = "GW-1"
BLOCKTIME   = 30
DMTLOGDEV   = "D:\tuxedo\log\DMLOG1"

"GW-2"
```

```
GWGRP       = GROUP2
TYPE        = OSITPX
DOMAINID    = "GW-2"
BLOCKTIME   = 30
DMTLOGDEV   = "D:\tuxedo\log\DMLOG2"

########################################################################
*DM_REMOTE_DOMAINS

DEFAULT:

"OPENTI" TYPE="OSITPX" DOMAINID="OPENTI"

########################################################################
*DM_OSITPX

"GW-1"
              AET="{1.3.145.62.103},{2}"
              TAILOR_PATH="d:\tuxedo\configs\tailor1.txt"
              # Inserted from OSITPX's config file:
              NWADDR="//SITE1:102"

"GW-2"
              AET="{1.3.145.62.103},{3}"
              TAILOR_PATH="d:\tuxedo\configs\tailor2.txt"
              # Inserted from OSITPX's config file:
              NWADDR="//SITE2:102" # second gateway must use
                                   # another IP or different
                                   # port number

"OPENTI"
              AET="{1.3.122.61.203},{20}"
              NWADDR="122.61.203.20"


########################################################################
#######*DM_LOCAL_SERVICES


########################################################################
*DM_REMOTE_SERVICES
DEFAULT: TRANTIME=300
# Each system will service different applications.

callSvc2   RDOM="OPENTI" LDOM="GW-1" PRIO=66
callSvc3   RDOM="OPENTI" LDOM="GW-2" PRIO=66
```

Using eLink OSI TP as a Pass-Through to Other TUXEDO Systems

For this example, the eLink OSI TP gateway acts as a pass-through to allow access to services on other Tuxedo systems. The eLink OSI TP gateway recieves service requests from a remote OLTP system and then forwards them through the `/TDOMAINS` gateway to a remote Tuxedo system. The systems are as follows:

Figure 4-1 Example of eLink OSI TP Acting as a Pass-Through to Other Tuxedo Systems



Listing 4-6 shows a sample `UBBCONFIG` file and Listing 4-7 shows the corresponding `UDMCONFIG` file for a pass-through configuration.

Listing 4-6 Sample UBBCONFIG File for Pass-Through Configuration

```
*RESOURCES
IPCKEY          65952
MASTER          "SITE1"
MODEL           SHM
PERM            0777

*MACHINES
"SITE1" LMID="SITE1"

                TUXCONFIG="D:\tuxedo\appdir\TUXCONFIG"
                TUXDIR="D:\tuxedo"
                APPDIR="D:\tuxedo\appdir"
                TLOGDEVICE="D:\tuxedo\log\TLOG"
                ULOGPFX="D:\tuxedo\log\ULOG"
                TLOGNAME=TLOG
```

```
                TLOGSIZE=20
                TYPE="SITE1"
*GROUPS

ADMGRP          LMID="SITE1"
                GRPNO=1

OSIGRP          LMID="SITE1"
                GRPNO=2

TDOMGRP         LMID="SITE1"
                GRPNO=3
                OPENINFO=NONE

*SERVERS

DEFAULT:        RQPERM=0666
                RPPERM=0666
                MIN=1
                MAX=1
                CONV=N
                MAXGEN=1
                GRACE=86400
                RESTART=N
                SYSTEM_ACCESS=FASTPATH

DMADM           SRVGRP=ADMGRP
                SRVID=20
                CLOPT="-A"
                RESTART=N

GWADM           SRVGRP=OSIGRP
                SRVID=21
                CLOPT="-A"
                RESTART=N

GWOSITP         SRVGRP=OSIGRP
                SRVID=22
                CLOPT="-A"
                RESTART=Y

GWADM           SRVGRP=TDOMGRP
                SRVID=51
                CONV=N
                CLOPT="-A"
                REPLYQ=N
                RESTART=N

GWTDOMAIN       SRVGRP=TDOMGRP
```

```
              SRVID=52
              CLOPT="-A"
              REPLYQ=N
              RESTART=Y
*SERVICES

DEFAULT:  LOAD=50 AUTOTRAN=N
```

Listing 4-7 Sample UDMCONFIG File for Pass-Through Configuration

```
*DM_RESOURCES
VERSION="SITE1"
#
*DM_LOCAL_DOMAINS

 #  SECURITY=NONE

"osi-local"
             GWGRP       = OSIGRP
             TYPE        = OSITPX
             DOMAINID    = "local"
             BLOCKTIME   = 2000
             AUDITLOG    = "D:\tuxedo\log\AUDIT"
             DMTLOGDEV   = "D:\tuxedo\log\DMLOG"
             DMTLOGSIZE  = 2048
             DMTLOGNAME  = "DMLOG"

"td-local"   GWGRP=TDOMGRP
             TYPE=TDOMAIN
             DOMAINID="td-local"
             DMTLOGDEV="D:\tuxedo\log\TDMLOG"


########################################################################
*DM_REMOTE_DOMAINS

DEFAULT:

"osi-client" TYPE=OSITPX   DOMAINID="osi-client"
"td-backend" TYPE=TDOMAIN DOMAINID="td-tpaix1"

########################################################################
*DM_TDOMAIN

"td-local"        NWADDR="192.63.22.2:5000"
```

```
"td-backend"      NWADDR="192.63.24.74:5000"
#####################################################################
*DM_OSITPX

"osi-local"
              AET="{1.3.192.63.22},{2}"
              TAILOR_PATH="d:\tuxedo\configs\tailor.txt"

# the NWADDR for OSI TP may have the same IP as /TDOMAINS, but
# requires a different port number
              NWADDR="192.63.22.2:102"

"osi-client"
               AET="{1.3.192.23.2},{3}"
              NWADDR="192.23.2.3"
               T_SEL="OSITP"

#####################################################################
*DM_LOCAL_SERVICES
# define the incoming services here, eventhough they reside on
# some remote /TDOMAIN machine.  Include views also on this machine
# for eLink OSI TP to process incoming messages

callSvc1

#####################################################################
*DM_REMOTE_SERVICES
DEFAULT:

# define the actual remote service request here.  It will be
# routed by /TDOMAINS

callSvc1   RDOM="td-backend"   LDOM="td-local" RNAME="callSvc1"
```

Note that the service that resides on the backend `/TDOMAIN` system must be defined as a local service on the eLink OSI TP system, so eLink OSI TP can process the incoming request. It must also be defined as a remote service so that the `/TDOMAIN` gateway can pass the service request to the backend `/TDOMAIN` system. The gateway system must have available the viewfiles and corresponding environment variables that are required by the service, even though the service exists on the backend system.

Setting up Security

In order for any security checking to occur, each domain must have a security mechanism in place. For the TUXEDO domain, the Authorization Server is the security mechanism. Refer to the BEA Tuxedo Online Documentation CD at http://edocs.bea.com/tuxedo/tux65/index.htm for more information about security for TUXEDO domains.

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

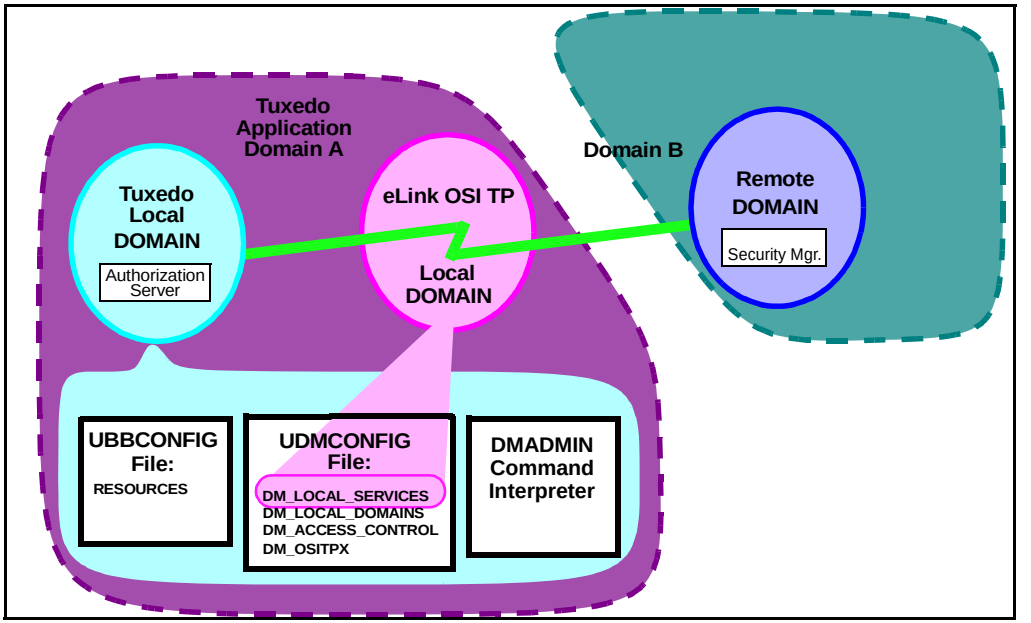
The eLink OSI TP gateway has a new feature to support link layer security. Each domain communicating over OSI TP must have the OSI TP security extensions implemented.

The BEA Tuxedo `UBBCONFIG` and eLink OSI TP `UDMCONFIG` files include five sections in which you specify security parameters:

- `RESOURCES` section of the `UBBCONFIG` file
- `DM_LOCAL_DOMAINS` section of the `UDMCONFIG` file
- `DM_OSITPX` section of the `UDMCONFIG` file
- `DM_ACCESS_CONTROL` section of the `UDMCONFIG` file
- `DM_LOCAL_SERVICES` section of the `UDMCONFIG` file

The following figure shows the relationships between security elements for eLink OSI TP.

Figure 4-2 eLink OSI TP Security Elements



Three sections in the UDMCONFIG file contain parameters affecting how eLink OSI TP controls access to the local TUXEDO domain:

- DM_LOCAL_DOMAINS section contains a SECURITY parameter that specifies the type of security enforced for the TUXEDO local domain.

  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 eLink OSI TP controls access to the Tuxedo local domain. If this parameter is set to NONE or APP_PW, the eLink OSI TP domain takes no action with regard to security. If this parameter is set to DM_PW, the eLink OSI TP domain enforces security according to the security settings in the DM_PASSWORDS section of the BDMCONFIG file.

Caution: Do not delete the Tuxedo BDMCONFIG file. The DM_PW information will be lost if the file is deleted. When new passwords are entered, the GWOSITP service must be shut down and restarted for the passwords to take effect.

- DM_OSITPX section contains an OPTION of SECURITY_SUPPORTED, which indicates that the remote domain supports the OSI TP security extension. The

OSI TP security extension allows OSI TP systems to perform link-layer security. The link layer security feature is activated when the DM_LOCAL_DOMAINS section has SECURITY=DM_PW set and OPTIONS=SECURITY_SUPPORTED is set for the remote 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.
- DM_LOCAL_SERVICES section contains an ACL parameter that 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.

For more information about these parameters, refer to Understanding the UDMCONFIG File

Editing the UDMCONFIG File

If you are upgrading from a previous version of eLink OSI TP, it is recommended that you use the osiadmin utility to update your UDMCONFIG file; however, you may edit the UDMCONFIG file manually. Refer to Using the OSI TP Administration Utility for more information about the osiadmin utility.

To edit the UDMCONFIG file manually, perform the following steps:

1. Find the UDMCONFIG file in your installation directory and open it in any text editor.
2. Edit the UDMCONFIG file as necessary. Refer to the parameter descriptions in this section for details about defining your eLink OSI TP configuration.
3. When editing is complete, save the UDMCONFIG file.

Note: You may want to save the original UDMCONFIG file with a different name or in a different directory.

4. Process the UDMCONFIG file with the udmloadcf utility. This parses the input and creates two binary files: the BDMCONFIG file, which is used by GWOSITP, and the BUDMCONFIG file, which is used by UDMADM.

Refer to Understanding the UDMCONFIG File for more detailed information about the parameters in the UDMCONFIG file.

Steps for Modifying the UDMCONFIG File Parameters

Perform the following steps to modify the UDMCONFIG file parameters:

- Step 1 - Define Local Domains
- Step 2 - Define Remote Domains
- Step 3 - Specify Addressing Information for OSI TP Domains
- Step 4 - Specify Access Control for OSI TP Domains
- Step 5 - Specify Available Local Tuxedo Services
- Step 6 - Specify Available Remote Tuxedo Services
- Step 7 - Specify Routing Information

Step 1 - Define Local Domains

You must define the local domains that use the OSI TP server group you defined in your Tuxedo UBBCONFIG file. Refer to Defining eLink OSI TP Servers for BEA Tuxedo for more information about the UBBCONFIG file.

Perform the following steps to define a local domain in the DM_LOCAL_DOMAINS section of the UDMCONFIG file:

1. Specify the local domain ID with the DOMAINID parameter.
2. Specify the gateway group you defined in the UBBCONFIG file with the GWGRP parameter.
3. Specify the domain type of OSITPX with the TYPE parameter.
4. Specify the size of the domain transaction log with the DMTLOGSIZE parameter.
5. Specify any of the optional DM_LOCAL_DOMAINS parameters that you require: AUDITLOG, BLOCKTIME, DMTLOGDEV, DMTLOGNAME, MAXRDTRAN, MAXTRAN, and SECURITY.

Listing 4-8 Example of DM_LOCAL_DOMAINS Section

```
*DM_LOCAL_DOMAINS

dalnt8
GWGRP       = OSIGRP
TYPE        = OSITPX
DOMAINID    = “dalnt8”
BLOCKTIME   = 30
DMTLOGDEV   = “D:\tuxedo\log\DMLOG”
SECURITY    = DM_PW  # turns link layer security on
```

Refer to Sample Configuration File for more detailed information.

Step 2 - Define Remote Domains

It is recommended that you use the `importcfg` command in the `osiadmin` utility to update remote domains if you are upgrading from a previous version of BEA eLink OSI TP; however, you can manually define remote domains. Refer to Using the OSI TP Administration Utility for more information about the `osiadmin` utility.

Perform the following steps to define remote domains in the `DM_REMOTE_DOMAINS` section of the `UDMCONFIG` file:

1. Specify the remote domain ID with the `DOMAINID` parameter.
2. Specify the `OSITPX` domain type with the `TYPE` parameter.

There are no optional parameters for the `DM_REMOTE_DOMAINS` section.

Listing 4-9 Example of DM_REMOTE_DOMAINS Section

```
*DM_REMOTE_DOMAINS

dal2200 TYPE=OSITPX DOMAINID="dal2200"
openti  TYPE=OSITPX DOMAINID="openti"
icl2    TYPE=OSITPX DOMAINID="icl2"
aseries1 TYPE=OSITPX DOMAINID="aseries1"
```

Refer to DM_REMOTE_DOMAINS Section for more detailed information.

Step 3 - Specify Addressing Information for OSI TP Domains

Perform the following steps to define addressing information for OSI TP domains in the DM_OSITPX section of the UDMCONFIG file:

1. Specify the Application Entity Title for each local and remote OSI TP domain with the AET parameter.
2. Specify the IP address or DNS name and port number for each local and remote OSI TP domain with the NWADDR parameter. If you are using multiple IP addresses, make sure to enter all the addresses on one line, with quotes around each address, and separate them with a semi-colon (;).
3. Specify the logical name for the software that provides transport layer services for all LDOMS and RDOMS with the T_SEL parameter.
4. Specify any of the optional DM_OSITPX parameters that you require: DNS_RESOLUTION, P_SEL, S_SEL, OPTIONS, TAILOR_PATH, and XATMI_ENCODING.

Listing 4-10 DM_OSITPX Section

```
*DM_OSITPX

dalnt8
                AET="{1.3.144.23.103},{208}"
                TAILOR_PATH="d:\tuxedo\configs\tailor.txt"
                NWADDR="//dalnt8:102"
                DNS_RESOLUTION=STARTUP # this is the default

dal2200
                AET="{1.3.132.61.146},{3}"
                XATMI_ENCODING="OLTP_TM2200"
                NWADDR="132.61.146.3;132.61.147.1"
                T_SEL="OSITP"

openti
                AET="{1.3.122.62.103},{209}"
                NWADDR="122.62.103.209"

icl2
               AET="{1.3.142.60.203},{4}"
```

```
                NWADDR="142.60.203.4"
                T_SEL="ICLTP"
                S_SEL="SSEL"
                P_SEL="PSEL"
                SECURITY_SUPPORTED=N  # this is the default


aseries1
                AET="{1.3.123.55.222},{51}"
                NWADDR="123.55.222.51"
                XATMI_ENCODING="PRELIMINARY"
                T_SEL=0x5453
                S_SEL=0x3F5C3F
                OPTIONS=SECURITY_SUPPORTED
```

Refer to DM_OSITPX Section for more detailed information.

Step 4 - Specify Access Control for OSI TP Domains

In the DM_ACCESS_CONTROL section of the UDMCONFIG file, specify a list of all the remote OSI TP domain IDs that can access the local domain with the ACLIST parameter. This parameter is optional.

Listing 4-11 Example of DM_ACCESS_CONTROL Section

```
*DM_ACCESS_CONTROL
mylist  ACLIST = dalnt8, dal2200
```

Refer to DM_ACCESS_CONTROL Section for more detailed information.

Step 5 - Specify Available Local Tuxedo Services

In the DM_LOCAL_SERVICES section of the UDMCONFIG file, specify the Tuxedo services that will be made available to OSI TP applications and define their options with the ACL, COUPLING, INBUFTYPE, INRECTYPE, LDOM, OUTRECTYPE, OUTBUFTYPE, and RNAME parameters. If the local service supports transactions, make sure the group it belongs to contains a TMS name.

These DM_LOCAL_SERVICES parameters are all optional.

Listing 4-12 Example of DM_LOCAL_SERVICES Section

```
*DM_LOCAL_SERVICES
TOUPPERF
  INRECTYPE="VIEW:view10"
  OUTBUFTYPE="FML:"
  COUPLING=LOOSE

TOUPPERF32
  INRECTYPE="VIEW:view10a"
  OUTBUFTYPE="FML32:"
  COUPLING=TIGHT


TOUPPERV
  INBUFTYPE="X_C_TYPE:v10"
  INRECTYPE="VIEW:upper"
  COUPLING=LOOSE

TOUPPERC OUTRECTYPE="X_OCTET" OUTBUFTYPE="CARRAY"
        INRECTYPE="X_OCTET"
        COUPLING=TIGHT

TOUPPERS  OUTRECTYPE="X_OCTET" OUTBUFTYPE="STRING"
        INRECTYPE="X_OCTET"

TOUPPERX  OUTRECTYPE="STRING" OUTBUFTYPE="STRING"
         INRECTYPE="X_OCTET"
```

Refer to DM_LOCAL_SERVICES Section for more detailed information.

Step 6 - Specify Available Remote Tuxedo Services

In the `DM_REMOTE_SERVICES` section of the `UDMCONFIG` file, specify the remote services that can be requested by Tuxedo applications and define their options with the `AUTOPREPARE`, `CONV`, `INBUFTYPE`, `INRECTYPE`, `LDOM`, `OUTRECTYPE`, `OUTBUFTYPE`, `RDOM`, `RNAME`, `ROUTING`, and `TRANTIME` parameters. These parameters are all optional.

Listing 4-13 Example of DM_REMOTE_SERVICES Section

```
*DM_REMOTE_SERVICES
DEFAULT: TRANTIME=300

ECHOXOCT RNAME="ECHOSRVR" OUTBUFTYPE="X_COMMON:ECHOVIEW"
RDOM=dal2200 LDOM=dalnt8
ECHOXCOM RNAME="ECHOSRVR" RDOM=openti LDOM=dalnt8 AUTOPREPARE=Y

ECHOXCTYPE RNAME="ECHOSRVR"
         INBUFTYPE="X_C_TYPE:ECHOVIEW"
         INRECTYPE="X_COMMON:ECHOVIEW"
         OUTBUFTYPE="X_C_TYPE:ECHOVIEW"
         OUTRECTYPE="X_COMMON:ECHOVIEW"
         RDOM=aseries1
         LDOM=dalnt8
         CONV=Y
ECHOVIEW RNAME="ECHOSRVR"
         INBUFTYPE="VIEW:ECHOVIEW"
         INRECTYPE="X_COMMON:ECHOVIEW"
         OUTBUFTYPE="VIEW:ECHOVIEW"
         OUTRECTYPE="X_COMMON:ECHOVIEW"
         RDOM=icl2
         LDOM=dalnt8
         REM_TPSUT="tpmvs"
```

Refer to DM_REMOTE_SERVICES Section for more detailed information.

Step 7 - Specify Routing Information

Perform the following steps to define routing information for service requests in the `DM_ROUTING` section of the `UDMCONFIG` file:

1. Specify the name of the routing field with the `FIELD` parameter.
2. Specify the data buffer type and subtype for which the routing entry is valid with the `BUFTYPE` parameter.
3. Specify the ranges and associated remote domain names for the routing field with the `RANGES` parameter.

Listing 4-14 Example of DM_ROUTING Section

```
*DM_ROUTING
ACCOUNT FIELD = branchid BUFTYPE = "View:account"
        RANGE = "MIN - 1000:aseries1, 1001-3000:openti, *:dal2200"
```

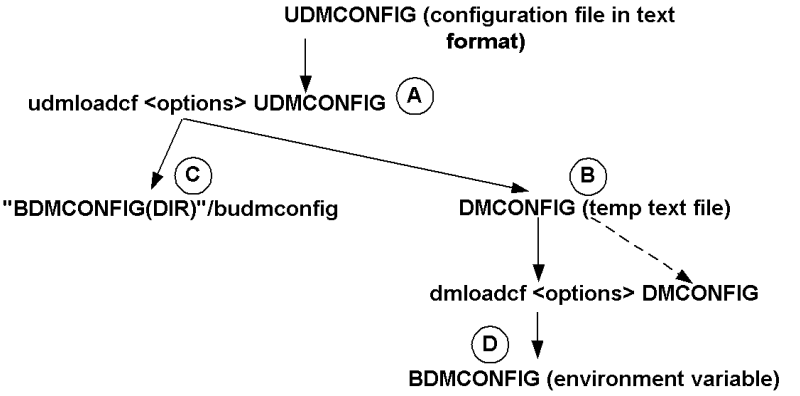
Refer to DM_ROUTING Section for more detailed information.

Processing a Configuration File with the Udmloadcf Utility

The udmloadcf utility compiles the UDMCONFIG file and creates two binary configuration files, BDMCONFIG, which is used by the DMADM server to control the run-time environment, and BUDMCONFIG, which is used by the GWOSITP server at run time.

Figure 4-3 shows how the udmloadcf utility processes the configuration file. A description of the process follows the figure.

Figure 4-3 Udmloadcf Process



When you start udmloadcf from the command line (A), it analyzes the input parameters and saves off any of the parameters ([-c] [-n] [-y] [-b blocks]) that are entered on the command line. If no command line syntax errors are detected, udmloadcf opens the input configuration file. It reads and processes the configuration file line by line. udmloadcf removes new (OSITP 4.0) parameters, saves them internally, and writes the modified information to a temporary output file (B) that will be used as input to BEA's dmloadcf. udmloadcf then takes the internally saved new parameters and creates a new binary file (C) that will be used as input to the access functions used by the gateway at runtime. This file is named by using the directory portion of the environment variable, BDMCONFIG, and adding the filename, BUDMCONFIG. Next, udmloadcf invokes dmloadcf with all of the parameters saved during command line checking to process the temporary DMCONFIG file (which it previously created) and create the BDMCONFIG file (D). Finally, udmloadcf deletes the temporary DMCONFIG file (B), unless the -k option has been specified, in which case it copies the temporary file to a file named dmconfig.sav in the current directory.

Invoking the Udmloadcf Utility

The udmloadcf utility is invoked from a command line with the following syntax:

udmloadcf [-c] [-n] [-y] [-b *blocks*] [-k] *udmconfig_file*

where the following options are valid:

-c
: Prints minimum IPC resources needed for each local domain (gateway group) in this configuration. The BDMCONFIG file is not updated.

-n
: Checks only the syntax of the ASCII DMCONFIG file without actually updating the BDMCONFIG file.

-y
: Suppresses a prompt to create and initialize the BDMCONFIG file. This parameter **must** be entered before the DMCONFIG file name.

-b blocks
: Indicates the number of blocks the device should use to create the Tuxedo file system. If the value of the -b option is large enough to hold the new BDMCONFIG tables, dmloadcf uses the specified value to create the new file system; otherwise, dmloadcf prints an error message and exits. If the -b

option is not specified, `dmloadcf` creates 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.

-k
: Keeps the temporary file used as input to `dmloadcf`. This parameter **must** be entered before the `UDMCONFIG` file name. The temporary file is saved as `dmconfig.sav`.

udmconfig_file
: Specifies the name of the input configuration file to `udmloadcf`.

Getting Help for the Udmloadcf Utility

You can also get help or usage information about the `udmloadcf` utility by entering any one of the following commands:

```
udmloadcf /?
udmloadcf /help
udmloadcf -?
udmloadcf -help
```

How the Udmloadcf Utility Works

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

A Tuxedo `DMTYPE` file is required to define the valid domain types. If this file does not exist, `udmloadcf` exits without performing any updates to the `BDMCONFIG` file.

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

After syntax checking, `udmloadcf` verifies that the file pointed to by `BDMCONFIG` exists, is a valid Tuxedo System file system, and contains `BDMCONFIG` tables. If these conditions are not true and the `-y` option was not entered on the command line, 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 and "Y" indicates that the configuration file should be created.

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

```
"Really overwrite BDMCONFIG file {y, q}?"
```

Prompting is suppressed if the standard input or output are not terminals. Any response other than "y" or "Y" causes udmloadcf to exit without creating the configuration file. If the BDMCONFIG file is not properly initialized and the user has responded with "Y", udmloadcf creates the Tuxedo file system and then creates the BDMCONFIG tables. Any response other than "y" or "Y" causes udmloadcf to exit without overwriting the file.

If the SECURITY parameter is specified in the RESOURCES section of the TUXCONFIG file, then udmloadcf flushes the standard input, turns off terminal echo, and prompts the user for an application password as follows:

```
Enter Application Password?
```

The password is truncated to 8 characters. You cannot load the ASCII DMCONFIG file through the standard input (rather than a file) when the SECURITY parameter is turned on. If the standard input is not a terminal, that is, if the user cannot be prompted for a password, then the environment variable APP_PW is accessed to set the application password. If the environment variable APP_PW is set with terminal as the standard input, then udmloadcf prints an error message, generates a log message, and fails to create the BDMCONFIG file.

Assuming no errors, and if all checks have passed, udmloadcf loads the DMCONFIG file into the BDMCONFIG file and overwrites all existing information found in the BDMCONFIG tables.

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

```
udmloadcf -b 2000 -y bank.udmconfig
```

Diagnostics

If an error is detected in the input, the offending line is printed to the standard error log 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.

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

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

If udmloadcf is run by a person whose effective user ID 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, udmloadcf exits. If the BDMCONFIG file is updated, a userlog message is generated to record this event.

Tuning OSI TP-Specific Tables with the TAILOR File

The OSI TP TAILOR file is external to the DMCONFIG and is used for tuning OSI TP-specific tables. All parameters in the TAILOR file are optional with preset defaults.

Following is a list of valid TAILOR parameters:

Parameter	Default	Description
FreeOldRetryTime	600 seconds	Time in seconds between automatic terminations of old connections
MaxConnections	500	Maximum number of active concurrent calls
MaxRemoteNodes	1000	Maximum number of total remote domains

Parameter	Default	Description
OldAssocTimeout	3600 seconds	Time in seconds denoting an "old" connection (association)
TCPSocketsKeepAlives	N	Toggle for TCP keepalive packets
TCPSocketsLinger	-1	Amount of time TCP/IP socket connection stays open
TCPSocketsListenQueueDepth	5	Number of held TCP/IP connections waiting to be accepted by eLink OSI TP
TCPSocketsNoDelay	N	Toggle to delay sends of TCP/IP packets until an ACK is received from remote machine
TracelpcKey	32800 seconds	IPC key value for the OSI TP log and trace shared memory section

Following is more detailed information about each of the TAILOR file parameters:

FreeOldRetryTimer = *numeric*

Specifies the time in seconds between automatic terminations of old connections (associations). OSI TP reuses established socket connections to a remote domain. The default value is 600 seconds.

MaxConnections = *numeric*

Specifies the maximum number of active concurrent calls. The default value is 500.

MaxRemoteNodes = numeric

Specifies the maximum number of total remote domains, including those added dynamically. The default value is 1000.

OldAssocTimeout = *numeric*

Specifies the time in seconds denoting an "old" connection (association). Any connection to a remote domain that remains unused by a tpcall() for this amount of time is subject to automatic termination. This default value is 3600 seconds.

TCPSocketsKeepAlives = {Y | N}

Specifies whether the TCP keepalive packets are sent. This is useful to insure the integrity of the TCP connection. If `Y` is specified the TCP/IP packets are sent out in a time period specified by the operating system. Most operating systems use a 2 hour time period. The default is `N`.

TCPSocketsLinger = {-1 | 0 | *numeric*}

Specifies the amount of the time that the TCP/IP socket connection will stay open. This can be used to timeout hung connections.

Possible values are:

`-1` (default)	Time that a TCP/IP socket connection stays in the `TIME_WAIT` state is determined by the operating system.
`0`	TCP/IP connection is closed immediately.
`n`	TCP/IP connection stays in the `TIME_WAIT` state for `n` seconds before closing.

TCPSocketsListenQueueDepth = *numeric*

Specifies the number of held TCP/IP connections waiting to be accepted by eLink OSI TP.

Possible values are:

`5` (default)	A minimum of 5 incoming TCP/IP connections are held.
`>5`	More than 5 incoming TCP/IP connections are held. The operating system may only supports a number up to a "reasonable value".

TCPSocketsNoDelay = {Y | N}

Specifies if subsequent sends of TCP/IP packets are held until an `ACK` is received from a remote machine. The default is `N`, the subsequent sends of TCP/IP packets may be held until an `ACK` is received. If `Y` is specified, the TCP/IP packets are sent immediately without waiting for an `ACK` of the previous send.

TracelpcKey = *numeric*

For UNIX systems, specifies the IPC key value for the OSI TP log and trace shared memory segment. If there are multiple local domains, then each doamin must have a unique IPC key value. On Windows NT systems, this value is ignored. The default is 32800.

CHAPTER

5 Using the OSI TP Administration Utility

This section covers the following topics:

About the Osiadmin Processor

Administration of the OSITP functions for BEA eLink OSI TP is provided by the `osiadmin` processor. Administration commands are input from a command line or script file. `osiadmin` can be run regardless of whether the eLink OSI TP process is booted, allowing the user to perform configuration utilities, such as export and import, independently.

The `osiadmin` processor provides the following features:

- Sets internal traces while the OSITP process is up and running.
- Provides a snapshot dump of `NW_BEA` and `NW_UNISYS` internal tables.
- Performs test of OSITP connection to a remote domain.

- Imports configurations from OSI TP 1.3 (`DMCONFIG, OSITP` configuration, and `ULS` configuration) and creates a comparable configuration in proper format for BEA eLink OSI TP 4.0.
- Exports the `dmconfig` file and creates the correct format of the `config-in.txt` file needed for the UNISYS ClearPath systems and creates the necessary OpenTI binary configuration files.
- Administers multiple local domains on the same system through one `osiadmin` processor.
- Sends optional trace messages to the Tuxedo `ULOG`.

Initiating Osiadmin

The `osiadmin` processor can be initiated in interactive, script, or batch mode. The `OSIRUNDIR` environment variable must be set in order to run the `osiadmin` utility. Refer to Setting Environment Variables for more information.

Initiating Osiadmin in Interactive Mode

Enter the following command at the command line:

```
osiadmin [LDOM]
```

The local domain name (`LDOM`) may be specified at `osiadmin` execution time. The `LDOM` is the `LDOM` identifier specified in the `Local_Domains` section of the `UDMCONFIG` file.

Note: The `LDOM` identifier is case sensitive.

For example,

```
osiadmin BA.CENTRAL01
```

where `BA.CENTRAL01` is the `LDOM`.

The LDOM is necessary for CONNECTIONSTATUS, DUMP, TRACE, TESTCONNECTION, and LISTRDOM commands. If the LDOM is not specified on the command line, the user must enter an LDOM command before using any of these commands.

Initiating Osiadmin in Script Mode

Enter the following command at the command line:

```
osiadmin [LDOM]<input_script_name>osiadmin_output
```

where input_script_name is the name of a script file that you created with a series of one or more commands you wish to execute. If the script file is in a different directory, you must also enter the directory path. The script file is a text file that you can create with any text editor. Following is an example run of a script that might be used with osiadmin:

Listing 5-1 Example of an Osiadmin Input Script

```
>page off             Do not prompt for long output
>echo on              Echo input commands
>LDOM BA.CENTRAL01    Set local domain name for
                         following commands
>lr                   List remote domains in
                         BA.CENTRAL01 configuration
 BA.BRANCH01
 BA.BRANCH02
>tc BA.BRANCH01       Verify connection between
                         BA.CENTRAL01 and BA.BRANCH01
<Successfully connected to BA.BRANCH01
>tc BA.BRANCH02       Verify connection between
                         BA.CENTRAL01 and BA.BRANCH02

 Successfully connected to BA.BRANCH02
>exit
```

Initiating Osiadmin in Batch Mode

Enter the following command at the command line:

```
osiadmin -b
```

The "-b" option initiates the `osiadmin` in batch mode with the following conditions: the menu of commands that is normally displayed upon startup of `osiadmin` is not printed, pagination is set to off, command echo is set to on, and interactive prompts (for example, "Overwrite file, Y or N") are suppressed.

What Happens When You Execute an Osiadmin Command?

When you initiate `osiadmin` or enter any of the other commands in interactive mode, the command executes if the syntax is correct. If the syntax is incorrect, an error message displays showing the correct syntax. When the command executes, results are printed to the screen. If the result is more than one page and `PAGINATE=ON` (which is the default), the prompt, `"CONTINUE? Y or N"` displays. Select `Y` to continue displaying another 20 lines or `N` to discontinue displaying the results.

At any time after a command has been executed, you can press `Enter` to redisplay the command syntax as shown by the Help command.

When you initiate `osiadmin` and enter commands using a script, the commands execute in order and the results are printed to the screen as the commands execute.

Using Osiadmin Commands

All `osiadmin` commands are initiated through interactive mode or through a script containing one or more commands. The following sections explain how to get Help for `osiadmin` commands and provide descriptions for each command in detail.

Getting Help for Osiadmin Commands

You can request a list of the valid `osiadmin` commands, request help for specific commands, or display command syntax after a command has been executed.

To display Help in interactive mode:

You can display a list of valid commands by entering the following command at the command line:

```
HELP
```

To display Help for a specific command:

Enter the following command at the command line:

```
HELP command
```

where `command` is any of the valid commands.

The exact syntax for the specified command displays.

Osiadmin Commands

Following are descriptions of the valid `osiadmin` commands.

CONNECTIONSTATUS

Allows a user to display the connection status for one or more `RDOMS` or a pattern containing wildcard characters (*). For example, `CONNECTIONSTATUS *` lists the connection status for all `RDOMS`. `CONNECTIONSTATUS x*` lists the connection status for all `RDOMS` starting with the letter x.

Syntax: `CONNECTIONSTATUS RDOM|pattern`

OR

`CS RDOM|pattern`

DUMP

Produces a dump of internal eLink OSI TP tables including the NW_BEA and the NW_UNISYS portion of code for diagnostic purposes. If a filename is entered, that file is used for the dump. If the filename is not entered, the default dump file, OSITPDUMP.TXT, is used.

Syntax: `DUMP [filename]`

OR

`DU [filename]`

ECHO

Displays input command lines as they are entered when set to ON. If no option is given, the current setting is toggled and the new setting is printed. The initial setting is OFF.

Syntax: `ECHO [OFF|ON]`

OR

`E [OFF|ON]`

EXPORTCFG

Exports eLink OSI TP configuration information from the UDMCONFIG file and creates configuration files readable by OSI-TP on UNISYS ClearPath servers. The names of the files produced are <output directory>TAILOR-IN.TXT and <output directory>CONFIG-IN.TXT. The output directory is optional. The current directory is the default if none is specified.

Syntax: `EXPORTCFG RDOM UDMCONFIG-SOURCE [<output directory>]`

OR

`EXCFG RDOM UDMCONFIG-SOURCE [<output directory>]`

IMPORTCFG

Imports the correct configuration information for the eLink gateway by automatically upgrading configurations from previous versions of the eLink gateway. importcfg specifies the DMCONFIG file, OSITP configuration file, and the ULS configuration file

to be used for input. The DMCONFIG file is used as input to the dmloadcf processor. A UDMCONFIG file is created as the output. The ULS-TAILOR and NEW-TAILOR files are optional and are only valid on Unix. If the output file already exists, a message is sent to confirm before overwriting the file. If no parameters are specified, the user is prompted for the filenames.

Syntax:
```
IMPORTCFG [<outfile> DMCONFIG-SOURCE OSITP-CONFIGFILE
{ULS-TAILOR-SOURCE NEW-TAILOR-SOURCE}]
```

OR

```
IMCFG [<outfile> DMCONFIG-SOURCE OSITP-CONFIGFILE
{ULS-TAILOR-SOURCE NEW-TAILOR-SOURCE}]
```

Note: OSITP-CONFIGFILE is the source configuration file and not the current.cfg file that was required for older versions of BEA eLink OSI TP.

LDOM

Sets the local domain name. The LDOM must be set for the DUMP, CONNECTIONSTATUS, TRACE, TESTCONNECTION, and LISTRDOM commands. The LDOM can also be specified as an argument when executing the osiadmin.

Syntax:
```
LDOM [LDOM name]
```

LISTRDOM

Retrieves and displays a list of all RDOMs.

Syntax:
```
LISTRDOM
```

OR

```
LR
```

PAGINATE

Turns paginate mode ON or OFF. Default is ON.

Syntax:
```
PAGINATE {OFF|ON}
```

OR

```
PAGE   {OFF|ON}
```

TESTCONNECTION

Verifies that the lower layer network is configured properly. OSI TP sends an association request to the specified remote domain and reports the status as the confirmation is received. This command blocks the resources until a successful connection sends an association release to free the resources.

Syntax: `TESTCONNECTION RDOM`

OR

`TC RDOM`

TRACE

Sets trace levels in eLink OSI TP for diagnostics and debugging purposes. When `TRACE` is set to `ON`, messages are written to the Tuxedo `ULOG`. If the `TRACE` command is entered without any options, it returns the current trace level.

Syntax: `TRACE {OFF|ON}`

OR

`TR {OFF|ON}`

A Error and Informational Messages

The BEA eLink OSI TP software issues the following error and informational messages:

1:	<NW_UNISYS messages>	
	DESCRIPTION	General message to log NW_UNISYS messages
	ACTION	Corrective action is given in the messages, otherwise contact Unisys Customer Support.
1000:ERROR	**Tried to write more than maxlen:len=%ld, maxlen=%ld**	
	DESCRIPTION	An internal error has been detected while processing the transaction blob record.
	ACTION	Contact Unisys Customer Support.
1001:ERROR	**Can't create file %s in APPDIR for storing OSITP blob**	
	DESCRIPTION	The Transaction blob file specified could not be created in the APPDIR directory.
	ACTION	Check the file privileges of the APPDIR directory.
1002:ERROR	**Can't write to OSITP blob file %s in APPDIR**	

	DESCRIPTION	The GWOSITP software was unable to write to the Transaction blob file specified.
	ACTION	Check to make sure there is sufficient space available in the APPDIR directory.
1100:ERROR	**Unable to retrieve Network context, shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support.
1101:ERROR	**Unable to retrieve file descriptor, shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support.
1102:INFO	**Unable to obtain svc (%s) info from shared memory, request rejected!**	
	DESCRIPTION	The requested remote service could not be found in Domain remote service section of shared memory.
	ACTION	Use dmunloadcf command to verify that the remote service specified is defined in the current configuration. If this service is not shown to exists in the current configuration, add it to the DM_REMOTE_SERVICES section of the domain configuration file, shutdown the gateway, rerun dmloadcf, and reboot the OSITP gateway process.
1103:ERROR	**Find network transaction failed, shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support
1104:ERROR	**Can not create Network Context, request rejected!**	

	DESCRIPTION	A network context structure could not be created for the current tpcall. It is possible that the necessary memory could not be allocated.
	ACTION	Adjust system parameters to allow more memory and restart the OSITP Gateway process.
1105:ERROR	**Could not retrieve Network context, shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support
1106:ERROR	**Unable to obtain remote domain info from shared memory**	
	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support
1108:WARN	**Transaction table full, request rejected!**	
	DESCRIPTION	The number of current transaction exceeds the number allocated to this gateway as defined by the domain configuration parameter MAXTRAN or the total number of transaction allocated as defined by the TUXCONFIG parameter MAXGTT.
	ACTION	Increase the appropriate Transaction parameter and restart the OSITP Gateway process and domain if necessary.
1110:ERROR	**Unable to set nettxid in shmem**	
	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support
1111:ERROR	**Cannot associate with network transaction**	
	DESCRIPTION	An attempt to start this transaction failed either because this transaction is shown to have already completed or was aborted.

	ACTION	No action is necessary.
1115:ERROR	**Could not find RDOM %s cache entry**	
	DESCRIPTION	The specified Remote domain should have already been placed in Cache. Either the rdom name or the cache memory has been corrupted.
	ACTION	Contact Unisys Customer Support
1117:ERROR	**Cannot set remote domain for transaction**	
	DESCRIPTION	The Maximum number of remote domains for a given transaction has been exceeded.
	ACTION	Increase the MAXRDOM parameter in the domain configuration file, rerun dmloadcf, and restart the OSITP Gateway, or reduce the number of remote domains involved in a single transaction.
1118:INFO	**Resumed with BLOCKING timeout**	
	DESCRIPTION	A blocking timeout has occurred on a request to a remote host.
	ACTION	Examine the ULOG for more information concerning the reason for the failure. Check configuration and increase the BLOCKTIME value, if needed.
1121:ERROR	**Bad action state**	
	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support
1122:ERROR	**Couldn't retrieve TRAN table from shmem, shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support
1123:ERROR	**Unable to obtain local tdomain address from shmem**	

	DESCRIPTION	An internal error has been detected while processing the action.
	ACTION	Contact Unisys Customer Support.
1124:ERROR	**Unable to get remote service information from shared memory**	
	DESCRIPTION	The remote service name specified was not found in the DM_REMOTE_SERVICES shared memory tables.
	ACTION	Check the domain configuration file DM_REMOTE_SERVICES section and verify that the Remote service specified has been defined. If not defined, correct the and restart any necessary processes.
1125:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A tpcall was received and the dialog was not in the FREE state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1126:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A tpconnect was received and the dialog was not in the FREE state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1127:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A tpsend was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1128:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	

	DESCRIPTION	A tpsend was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1129:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A Network reply was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1130:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A Network failure was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1131:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A Network tpcall was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1132:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A Network tpconnect was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1133:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	

	DESCRIPTION	A Network tpsend was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1134:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A MSG reply was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1135:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A MSG tpdisconnect was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1136:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A MSG failure was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1137:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	A tpconnect confirm was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1138:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	

	DESCRIPTION	A tpconnect fail was received and the dialog was not in a valid state. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1139:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1140:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	An invalid dialog state was encountered. An internal processing error has occurred while processing a transaction.
	ACTION	Contact Unisys Customer Support.
1141:ERROR	**Internal error, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred while processing this action.
	ACTION	Look for prior Userlog messages for more information. Contact Unisys Customer Support.
1142:WARN	**Can't find input message buffer, ACALL event deleted!**	
	DESCRIPTION	A reply was received for an ACALL event which has been deleted and no longer exists.
	ACTION	No action is necessary.
1143:ERROR	**More than one tpcall to AUTOPREPARE svc (%s) attempted, request rejected**	
	DESCRIPTION	An attempt to make more than one tpcall within a transaction to an AUTOPREPARE service was attempted. This tpcall was rejected.
	ACTION	Modify the client code or remove the AUTOPREPARE setting on the Service.

1250:ERROR	**copy_local_service: Memory error on Local Service**	
	DESCRIPTION	An internal error has been detected while copying a Local Service.
	ACTION	Contact Unisys Customer Support.
1251:ERROR	**copy_remote_domain: Memory error**	
	DESCRIPTION	An internal error has been detected while copying a Remote Domain.
	ACTION	Contact Unisys Customer Support.
1252:ERROR	**copy_remote_service: Memory error on remote Service**	
	DESCRIPTION	An internal error has been detected while copying a Remote Service.
	ACTION	Contact Unisys Customer Support.
1253:ERROR	**gwnwUtility_getLocalConfig: Memory error**	
	DESCRIPTION	An internal error has been detected while getting the local configuration.
	ACTION	Contact Unisys Customer Support.
1254:ERROR	**gwnwUtility_getLocalConfig: Error walking though LDOM list**	
	DESCRIPTION	An internal error has been detected while walking the LDOM list.
	ACTION	Contact Unisys Customer Support.
1255:ERROR	**read_cfg_in_file: Error allocating receive buffer**	
	DESCRIPTION	An internal error has been detected while allocating the receive buffer.
	ACTION	Contact Unisys Customer Support.
1256:WARNING	**read_cfg_in_file: Can't send request to service UDMADM. Tperrno = %d**	
	DESCRIPTION	The gateway has attempted to call the UDMADM server to obtain the OSITP configuration information. The UDMADM server did not respond.

	ACTION	If running in a Multiple Machine (MP) configuration the UDMADM service is required. In this case you need to modify the ubbconfig file and add UDMADM as a service. See OSI-TP Uses guide for help. Reprocess the file with tmloadcf. Then rerun tmboot. If running in the single machine (SHM) configuration the UDMADM service is not needed and and this message can be ignored.
1257:ERROR	**read_cfg_in_file: Bad version # of BUDMCONFIG file**	
	DESCRIPTION	The binary file "BUDMCONFIG" contains an internal version number which is incompatible with this version of OSITP.
	ACTION	Reprocess your UDMCONFIG file with udmloadcf to create a new binary "BUDMCONFIG" file.
1258:ERROR	**read_cfg_in_file: Memory error on local domains**	
	DESCRIPTION	An internal error has been detected while processing the local domains.
	ACTION	Contact Unisys Customer Support.
1259:ERROR	**read_cfg_in_file: Memory error on Remote domains**	
	DESCRIPTION	An internal error has been detected while processing the remote domains.
	ACTION	Contact Unisys Customer Support.
1261:ERROR	**read_cfg_in_file: Memory error on Local services**	
	DESCRIPTION	An internal error has been detected while processing Local Services.
	ACTION	Contact Unisys Customer Support.
1262:ERROR	**read_cfg_in_file: Memory error on Remote services**	
	DESCRIPTION	An internal error has been detected while processing Remote Services.

	ACTION	Contact Unisys Customer Support.
1263:ERROR	**read_cfg_in_file: fopen error on input file %s**	
	DESCRIPTION	Error opening input file.
	ACTION	Verify file exists and is accessible by you.
1264:ERROR	**read_cfg_in_file: Read error on # of Version number.**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.
1266:ERROR	**read_cfg_in_file: Read error on # of Local domains.**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.
1268:ERROR	**read_cfg_in_file: Read error on local domains**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.
1269:ERROR	**read_cfg_in_file: Read error on # of Remote domains.**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.
1271:ERROR	**read_cfg_in_file: Read error on Remote domains.**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.
1273:ERROR	**read_cfg_in_file: Read error on # of Local services.**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.

1275:ERROR	**read_cfg_in_file: Read error on Local Services.**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.
1276:ERROR	**read_cfg_in_file: Read error on # of Remote services.**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.
1278:ERROR	**read_cfg_in_file: Read error on Remote Services.**	
	DESCRIPTION	Error processing binary configuration file.
	ACTION	Reprocess udmconfig text file with udmloadcf.
1279:ERROR	**BDMCONFIG environment variable is not set.**	
	DESCRIPTION	The environment variable BDMCONFIG is not set.
	ACTION	Set the environment variable BDMCONFIG to the fully qualified name of the BDMCONFIG file.
1325:ERROR	**ioptr_init_char failed**	
	DESCRIPTION	An internal error occurred; gw_nw_buf_encode failed to initialize ioptr structure. The vbuf for which the ioptr was to be initialized is corrupted.
	ACTION	Contact Unisys Customer Support.
1326:ERROR	**ASN.1 ENCODE failed %s**	
	DESCRIPTION	The Gateway process was unable to encode the typed buffer.

	ACTION	Look for further errors which indicate the type and subtype of the buffer which could not be encoded. Check the encoding types and the typed buffer definitions to ensure that they are compatible. Correct any discrepancies and restart the OSITP Gateway process if necessary.
1328:ERROR	**ASN.1 DECODE failed %s**	
	DESCRIPTION	The Gateway process was unable to decode the typed buffer.
	ACTION	Look for further errors which indicate the type and subtype of the buffer which could not be decoded. Check the encoding types and the typed buffer definitions to ensure that they are compatible. Correct any discrepancies and restart the OSITP Gateway process if necessary.
1329:ERROR	**tpalloc failed: %s**	
	DESCRIPTION	An Internal error has occurred while trying to perform a tpalloc.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1341:ERROR	**Unable to process typed buffer, type=%s subtype=%s**	
	DESCRIPTION	The Gateway process was unable to convert the specified buffer to an ASN.1 encoded buffer.
	ACTION	Check the encoding types and the typed buffer definitions to ensure that they are compatible. If buffer conversion is being performed, check to make that the data conversion specified is valid. Correct any discrepancies and restart the OSITP Gateway process if necessary.
1347:ERROR	**Unable to reconstruct typed buffer, type=%s subtype=%s**	

	DESCRIPTION	The Gateway process was unable to convert an ASN1 encoded buffer tothe specified buffer.
	ACTION	Check the encoding types and the typed buffer definitions to ensure that they are compatible. If buffer conversion is being performed, check to make that the data conversion specified is valid. Correct any discrepancies and restart the OSITP Gateway process if necessary.
1350:ERROR	**read_cfg_in_file: fopen error on input file %s**	
	DESCRIPTION	UDMADM had a problem opening the binary configuration file, BUDMCONFIG.
	ACTION	Make sure the binary configuration file, BUDMCONFIG, exists and is accessible to you.
1351:ERROR	**Bad File Handle for binary file**	
	DESCRIPTION	An internal error was detected in UDMADM while trying to process the binary configuration file (BUDMCONFIG).
	ACTION	Contact Unisys Customer Support.
1352:ERROR	**tpalloc failed in UDMADM**	
	DESCRIPTION	An internal error was detected in UDMADM.
	ACTION	Contact Unisys Customer Support.
1353:ERROR	**BDMCONFIG environment variable is not set.**	
	DESCRIPTION	The environment variable BDMCONFIG is not set.
	ACTION	Set the environment variable BDMCONFIG to the fully qualified name of the BDMCONFIG file.
1509:ERROR	**Invalid conversation context**	
	DESCRIPTION	An internal error has been detected.

	ACTION	Contact Unisys Customer Support.
1511:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.
	ACTION	Contact Unisys Customer Support.
1513:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.
	ACTION	Contact Unisys Customer Support.
1514:ERROR	**Protocol error, dial state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.
	ACTION	Contact Unisys Customer Support.
1515:ERROR	**Protocol error, node state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.
	ACTION	Contact Unisys Customer Support.
1517:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.
	ACTION	Contact Unisys Customer Support.
1518:ERROR	**Protocol error, node state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.
	ACTION	Contact Unisys Customer Support.
1521:ERROR	**Could not find transaction node, shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.
	ACTION	Contact Unisys Customer Support.
1526:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.

	ACTION	Contact Unisys Customer Support.
1530:ERROR	**Could not find transaction node, shutdown Gateway!**	
	DESCRIPTION	An internal error has been detected.
	ACTION	Contact Unisys Customer Support.
1601:ERROR	**Can not create Network Context, request rejected!**	
	DESCRIPTION	A Network Context structure could not be allocated for the requested tpconnect. There is not enough process memory available to allocate the structure necessary for processing the requested tpconnect.
	ACTION	Increase system parameters to allow more process memory and restart the Gateway.
1602:ERROR	**Could not retrieve Network context, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
1603:ERROR	**Could not retrieve Network context, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
1604:INFO	**Unable to obtain svc (%s)info from shared memory, request rejected!**	
	DESCRIPTION	The service requested could not be found in the shared memory DM_REMOTE_SERVICES section. The shared memory has been corrupted.
	ACTION	Contact Unisys Customer Support.
1605:WARN	**Transaction table full, request rejected!**	
	DESCRIPTION	The number of outstanding Transactions has exceeded the Maximum number allowed as defined by the Local Domains MAXTRAN or the Machines definition MAXGTT.

	ACTION	Increase either the MAXTRAN or MAXGTT configuration parameter and restart the Gateway.
1606:WARN	**Unable to create gwnwClientAgent, request rejected!**	
	DESCRIPTION	There is not enough process memory available to allocate the structure necessary for processing the requested tpconnect.
	ACTION	Increase system parameters to allow more process memory and restart the Gateway.
1607:ERROR	**Unable to obtain remote domain info from shared memory**	
	DESCRIPTION	The remote domain could not be found in the shared memory DM_REMOTE_DOMAINS section. The shared memory has been corrupted.
	ACTION	Contact Unisys Customer Support.
1608:ERROR	**Conversation identifier is invalid**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1609:ERROR	**Couldn't retrieve TRAN table from shmem**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1610:ERROR	**Unable to set nettxid in shmem**	
	DESCRIPTION	An attempt to set the network transaction id in shared memory failed. This could indicate that the shared memory tables have been corrupted.
	ACTION	Contact Unisys Customer Support.
1611:ERROR	**Cannot associate with network transaction**	
	DESCRIPTION	An attempt to start this transaction failed either because this transaction is shown to have already completed or was aborted.

	ACTION	No action is necessary.
1612:ERROR	**Cannot set remote domain for transaction**	
	DESCRIPTION	An attempt to set the remote domain for this transaction failed.
	ACTION	Contact Unisys Customer Support.
1613:ERROR	**Invalid action index**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1614:ERROR	**Invalid incoming conversation context**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1615:ERROR	**Conversation identifier is NULL**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1616:ERROR	**Invalid action index**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1617:ERROR	**Invalid conversation context**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1619:INFO	**Resumed with blocking timeout**	
	DESCRIPTION	This conversation incurred a blocking timeout and will be aborted.
	ACTION	Examine the ULOG for more information concerning the reason for the failure. Check configuration and increase the BLOCKTIME value, if needed.

1620:ERROR	**Bad action state**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Software Support.
1621:ERROR	**Could not find RDOM %s cache entry**	
	DESCRIPTION	The remote domain id specified for this service could not be found in the remote domain cache. This could indicate that the configuration file is incorrect or that there has been corruption of the remote domain cache.
	ACTION	Check the domain configuration file and make sure that the remote domain specified is defined in the file correctly. If in error, correct the error, shutdown the gateway, reload the configuration file, and restart the gateway.
1622:ERROR	**Invalid outgoing conversation context**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Software Support.
1705:INFO	**Found transaction without log record**	
	DESCRIPTION	A previous transaction was found, however a log record for this transaction did not exist. This transaction will be rolled back.
	ACTION	No Action necessary.
1706:ERROR	**Create transaction node failed, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1714:WARN	**log record refused**	
	DESCRIPTION	The transaction log record could not be recovered. The transaction will be rolled back.

	ACTION	No action is necessary.
1717:ERROR	**Unable to create action to handle restarted transaction!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1725:ERROR	**Create transaction node failed, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1726:ERROR	**Unable to create action to handle restarted transaction!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1728:ERROR	**Unable to create action to handle restarted transaction!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1752:ERROR	**Can’t find view file, %s**	
	DESCRIPTION	The specified view file could not be found.
	ACTION	Check to make sure that the view file exists and the VIEWFILES or VIEWFILES32 environment variables contain the specified view file name. Also check to make sure that the view file is located in the directory specified by the VIEWDIR or VIEWDIR32 environment variable.
1753:ERROR	**APDU buffer overflow**	
	DESCRIPTION	The APDU buffer allocated was not large enough to hold the data contained in the view buffer.
	ACTION	Contact Unisys Customer Support.
1754:ERROR	**Memory allocation error**	

	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1755:ERROR	**APDU buffer overflow**	
	DESCRIPTION	The APDU buffer allocated was not large enough to hold the data contained in the view buffer.
	ACTION	Contact Unisys Customer Support.
1756:ERROR	**Memory allocation error**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1757:ERROR	**APDU buffer overflow**	
	DESCRIPTION	The APDU buffer allocated was not large enough to hold the data contained in the view buffer.
	ACTION	Contact Unisys Customer Support.
1758:ERROR	**Memory allocation error**	
	DESCRIPTION	An internal error occured; An attempt to allocate a buffer failed.
	ACTION	Contact Unisys Technical Support
1759:ERROR	**Invalid data type for X_C_TYPE**	
	DESCRIPTION	The data type contained in the view is not a valid X_C_TYPE buffer data type. Valid X_C_TYPE data types are short, int, long, char, float, double, decimal, string, and carray.
	ACTION	Look for further error which indicate the name of the VIEW buffer which is in error. Correct the View buffer to contain the correct data types or define the buffer as X_C_TYPE.

1760:ERROR	**VIEW32 file %s not found**	
	DESCRIPTION	The specified view file could not be found.
	ACTION	Check to make sure that the view file exists and the VIEWFILES32 environment variables contain the specified view file name. Also check to make sure that the view file is located in the directory specified by the VIEWDIR32 environment variable.
1761:ERROR	**Can't find view file %s**	
	DESCRIPTION	The specified view file could not be found.
	ACTION	Check to make sure that the view file exists and the VIEWFILES or VIEWFILES32 environment variables contain the specified view file name. Also check to make sure that the view file is located in the directory specified by the VIEWDIR or VIEWDIR32 environment variable.
1762:ERROR	**Missing data element in received Data Stream**	
	DESCRIPTION	The X_C_TYPE buffer did not contain the expected data element as defined by the destination VIEW.
	ACTION	Look for further User Log messages which indicate what the VIEW Type and Subtype was involved, then make sure the source VIEW and destination VIEW definitions are identical. Correct the VIEWs and restart the necessary gateway process.
1763:ERROR	**APDU buffer overflow**	
	DESCRIPTION	The APDU buffer allocated was not large enough to hold the data contained in the view buffer.
	ACTION	Contact Unisys Customer Support.
1764:ERROR	**Missing data element in received Data Stream**	

	DESCRIPTION	The X_C_TYPE buffer did not contain the expected data element as defined by the destination VIEW.
	ACTION	Look for further User Log messages which indicate what the VIEW Type and Subtype was involved, then make sure the source VIEW and destination VIEW definitions are identical. Correct the VIEWs and restart the necessary gateway process.
1765:ERROR	**APDU buffer overflow**	
	DESCRIPTION	The APDU buffer allocated was not large enough to hold the data contained in the view buffer.
	ACTION	Contact Unisys Customer Support.
1766:ERROR	**Missing data element in received Data Stream**	
	DESCRIPTION	The X_C_TYPE buffer did not contain the expected data element as defined by the destination VIEW.
	ACTION	Look for further User Log messages which indicate what the VIEW Type and Subtype was involved, then make sure the source VIEW and destination VIEW definitions are identical. Correct the VIEWs and restart the necessary gateway process.
1767:ERROR	**Invalid data type for %s**	
	DESCRIPTION	The data type contained in the view is not a valid X_C_TYPE buffer data type. Valid X_C_TYPE data types are short, int, long, char, float, double, decimal, string, and carray.
	ACTION	Look for further error which indicate the name of the VIEW buffer which is in error. Correct the View buffer to contain the correct data types or define the buffer as X_C_TYPE.

1901:ERROR	**Invalid Buffer Type**	
	DESCRIPTION	A null buffer was passed to a service call/service reply.
	ACTION	Make sure the buffer is allocated, and the buffer is not NULL
1902:ERROR	**Invalid Buffer Type**	
	DESCRIPTION	Buffer type not defined in the buffer type switch
	ACTION	Examine the application and the configuration to determine which buffer type is being used. GWOSITP only supports standard TUXEDO buffer types over the domain gateway. If the buffer type is VIEW, VIEW32, then also examine the environment variables VIEWDIR, VIEWFILES, VIEWDIR32, and VIEWFILES32
1903:ERROR	**Unable to obtain svc info from svcinfo**	
	DESCRIPTION	An internal error occured; gw_nw_buf_encode failed to obtain remote service information
	ACTION	Contact Unisys Technical Support
1904:ERROR	**Unable to obtain INRECTYPE for remote service**	
	DESCRIPTION	An internal error occured; gw_nw_buf_encode failed to obtain INRECTYPE for remote service
	ACTION	Contact Unisys Technical Support
1905:ERROR	**Can’t find LSVC entry by rname %s**	
	DESCRIPTION	An internal error occured; gw_nw_buf_encode failed to obtain local service information
	ACTION	Contact Unisys Technical Support
1906:ERROR	**Unable to obtain INRECTYPE for local service**	

	DESCRIPTION	An internal error occured; gw_nw_buf_encode failed to obtain INRECTYPE for local service
	ACTION	Contact Unisys Technical Support
1907:ERROR	**Received type(%s:%s) does not match INBUFTYPE(%s:%s)**	
	DESCRIPTION	Buffer received by the gateway does not match INBUFTYPE configuration
	ACTION	Reconfigure INBUFTYPE to match incoming buffer type and subtype
1908:ERROR	**VIEW file %s not found**	
	DESCRIPTION	The specified view file could not be found.
	ACTION	Check to make sure that the view file exists and the VIEWFILES environment variables contain the specified view file name. Also check to make sure that the view file is located in the directory specified by the VIEWDIR environment variable.
1909:ERROR	**Conversion invalid from (%s) to (%s)**	
	DESCRIPTION	Conversion from a 16:bit buffer to 32:bit buffer, or vice versa, is not allowed
	ACTION	Make sure conversion specified is between 16-bit buffers, or between 32-bit buffers
1910:ERROR	**INRECTYPE not configured for (%s)**	
	DESCRIPTION	Buffers of type FML / FML32 cannot be transmitted to a remote system
	ACTION	Configure INRECTYPE to VIEW, X_C_TYPE, or X_COMMON
1911:ERROR	**VIEW file (%s) not found**	
	DESCRIPTION	Could not find the listed view file
	ACTION	Verify VIEWDIR and VIEWFILES environment variables
1912:ERROR	**VIEW file (%s) not found**	

	DESCRIPTION	Could not find the listed view file
	ACTION	Verify VIEWDIR32 and VIEWFILES32 environment variables
1913:ERROR	**Invalid buffer type specified**	
	DESCRIPTION	Buffer type not defined in the buffer type switch
	ACTION	Examine the application and the configuration to determine which buffer type is being used. GWOSITP only supports standard TUXEDO buffer types over the domain gateway. If the buffer type is VIEW, VIEW32, then also examine the environment variables VIEWDIR, VIEWFILES, VIEWDIR32, and VIEWFILES32
1914:ERROR	**Memory allocation failure**	
	DESCRIPTION	An internal error occured; gw_nw_buf_encode failed to allocate XATMI_typed_buffer
	ACTION	Contact Unisys Technical Support
1915:ERROR	**Invalid Buffer Type**	
	DESCRIPTION	Buffer type not defined in the buffer type switch
	ACTION	Examine the application and the configuration to determine which buffer type is being used. GWOSITP only supports standard TUXEDO buffer types over the domain gateway. If the buffer type is VIEW, VIEW32, then also examine the environment variables VIEWDIR, VIEWFILES, VIEWDIR32, and VIEWFILES32
1916:ERROR	**Memory allocation failure**	
	DESCRIPTION	An internal error occured; gw_nw_buf_encode failed to allocate ap_osi_vbuf

	ACTION	Contact Unisys Technical Support
1917:ERROR	**Encode failed**	
	DESCRIPTION	An internal error occured; gw_nw_buf_encode failed to encode typed buffer
	ACTION	Contact Unisys Technical Support
1918:ERROR	**Conversion invalid from (%s) to (%s)**	
	DESCRIPTION	Buffers of type FML and FML32 cannot be transmitted to a remote system
	ACTION	Remove INRECTYPE or reconfigure INRECTYPE to a different type such as a VIEW
1919:ERROR	**Received type(%s:%s) does not match OUTRECTYPE(%s:%s)**	
	DESCRIPTION	Buffer received by the gateway does not match OUTRECTYPE configuration
	ACTION	Reconfigure OUTRECTYPE to match incoming buffer type and subtype
1920:ERROR	**OUTBUFTYPE(%s:%s) invalid for received type(%s)**	
	DESCRIPTION	Received type cannot be converted to OUTBUFTYPE
	ACTION	Configure OUTBUFTYPE to X_OCTET, STRING, or CARRAY
1921:ERROR	**Cannot open %s:%s**	
	DESCRIPTION	Could not find the listed view file
	ACTION	Verify VIEWDIR and VIEWFILES environment variables
1922:ERROR	**FML incompatible field in %s:%s**	
	DESCRIPTION	The listed view contains packed decimal data type. Views containing this type cannot be used for FML-to-view or view-to-view conversions

	ACTION	Remove packed decimal fields from the input FML or view, or send the same view structure to the local gateway that is sent to the remote gateway
1923:ERROR	**Cannot convert view (%s) not mapped to FML**	
	DESCRIPTION	Listed view is not mapped to FML
	ACTION	Recompile the view WITHOUT the '-n' option of the view compiler
1924:ERROR	**tpalloc for %s failed**	
	DESCRIPTION	Allocation of an TUXEDO typed buffer failed
	ACTION	Contact Unisys Technical Support
1925:ERROR	**Fvstof failed for %s:%s**	
	DESCRIPTION	The conversion of a C structure to a fielded buffer failed
	ACTION	Contact Unisys Technical Support
1926:ERROR	**Cannot open %s:%s**	
	DESCRIPTION	Could not find the listed view file
	ACTION	Verify VIEWDIR and VIEWFILES environment variables
1927:ERROR	**Cannot convert view (%s) not mapped to FML**	
	DESCRIPTION	Listed view is not mapped to FML
	ACTION	Recompile the view WITHOUT the '-n' option of the view compiler
1928:ERROR	**tpalloc for %s:%s failed**	
	DESCRIPTION	Allocating a typed buffer for the listed view failed
	ACTION	Contact Unisys Technical Support
1929:ERROR	**Fvftos failed for %s:%s**	

	DESCRIPTION	The conversion of a fielded buffer to a C structure failed
	ACTION	Contact Unisys Technical Support
1930:ERROR	**Unable to obtain svc info from svcinfo**	
	DESCRIPTION	An internal error occured; gw_nw_buf_decode failed to obtain remote service information
	ACTION	Contact Unisys Technical Support
1931:ERROR	**Unable to obtain OUTRECTYPE for remote service**	
	DESCRIPTION	An internal error occured; gw_nw_buf_decode failed to obtain OUTRECTYPE for remote service
	ACTION	Contact Unisys Technical Support
1932:ERROR	**Can't find LSVC entry by rname <%s>**	
	DESCRIPTION	An internal error occured; gw_nw_buf_decode failed to obtain local service information
	ACTION	Contact Unisys Technical Support
1933:ERROR	**Unable to obtain OUTRECTYPE for local service**	
	DESCRIPTION	An internal error occured; gw_nw_buf_decode failed to obtain OUTRECTYPE for local service
	ACTION	Contact Unisys Technical Support
1934:ERROR	**ioptr_init_char failed**	
	DESCRIPTION	An internal error occured; gw_nw_buf_decode failed to initialize ioptr
	ACTION	Contact Unisys Technical Support
1935:ERROR	**Memory allocation failure**	

	DESCRIPTION	An internal error occured; gw_nw_buf_decode failed to allocate XATMI_typed_buffer
	ACTION	Contact Unisys Technical Support
1936:ERROR	**Invalid Buffer Type**	
	DESCRIPTION	Buffer type not defined in the buffer type switch
	ACTION	Examine the application and the configuration to determine which buffer type is being used. GWOSITP only supports standard TUXEDO buffer types over the domain gateway. If the buffer type is VIEW, VIEW32, then also examine the environment variables VIEWDIR, VIEWFILES, VIEWDIR32, and VIEWFILES32
1937:ERROR	**conversion invalid from < %s > to < %s >**	
	DESCRIPTION	Conversion from a 16-bit buffer to 32-bit buffer, or vice versa, is not allowed
	ACTION	Make sure conversion specified is between two 16-bit buffers, or between two 32-bit buffers
1947:ERROR	**Service < %s > is denied Access**	
	DESCRIPTION	The user has been denied access to the specified service. The Remote domain from which this call was initiated is not specified in the Access Control list for this service.
	ACTION	Check the Access Control list and add the Remote Domain if necessary and restart the Gateway process.
1949:ERROR	**Undefined buffer type**	
	DESCRIPTION	An internal processing error has occurred. The ASN.1 buffer type to convert the view to is not X_OCTET, X_COMMON, or X_C_TYPE.

	ACTION	Contact Unisys Customer Support.
1950:ERROR	**Not a typed buffer**	
	DESCRIPTION	An internal processing error has occurred. The buffer to convert to X_OCTET was null.
	ACTION	Contact Unisys Customer Support.
1951:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1952:ERROR	**Not a typed buffer**	
	DESCRIPTION	An internal processing error has occurred. The buffer to convert to X_COMMON was null.
	ACTION	Contact Unisys Customer Support.
1953:ERROR	**Can't find view file**	
	DESCRIPTION	The specified view file could not be found.
	ACTION	Check to make sure that the view file exists and the VIEWFILES or VIEWFILES32 environment variables contain the specified view file name. Also check to make sure that the view file is located in the directory specified by the VIEWDIR or VIEWDIR32 environment variable.
1954:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1955:ERROR	**APDU buffer overflow**	

	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1956:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1957:ERROR	**APDU buffer overflow**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1958:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1959:ERROR	**Buffer size error**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1960:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1961:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.

	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1962:ERROR	**Invalid data type for X_COMMON**	
	DESCRIPTION	The data type contained in the view is not a valid X_COMMON buffer data type. Valid X_COMMON data types are short, long, char, string, and carray.
	ACTION	Look for further error which indicate the name of the VIEW buffer which is in error. Correct the View buffer to contain the correct data types or define the buffer as X_C_TYPE.
1963:ERROR	**Not a typed buffer**	
	DESCRIPTION	An internal processing error has occurred. The buffer to convert to X_OCTET was null.
	ACTION	Contact Unisys Customer Support.
1964:ERROR	**Can't find view file**	
	DESCRIPTION	The specified view file could not be found.
	ACTION	Check to make sure that the view file exists and the VIEWFILES or VIEWFILES32 environment variables contain the specified view file name. Also check to make sure that the view file is located in the directory specified by the VIEWDIR or VIEWDIR32 environment variable.
1965:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1966:ERROR	**APDU buffer overflow**	

	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1967:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1968:ERROR	**APDU buffer overflow**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1969:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1970:ERROR	**Buffer size error**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1971:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1972:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.

	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
1973:ERROR	**Invalid data type for X_C_TYPE**	
	DESCRIPTION	The data type contained in the view is not a valid X_C_TYPE buffer data type. Valid X_C_TYPE data types are short, int, long, char, float, double, decimal, string, and carray.
	ACTION	Look for further error which indicate the name of the VIEW buffer which is in error. Correct the View buffer to contain the correct data types or define the buffer as X_C_TYPE.
1974:ERROR	**Undefined buffer type**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1975:ERROR	**Not a typed buffer**	
	DESCRIPTION	An internal processing error has occurred. The buffer to convert to X_OCTET was null.
	ACTION	Contact Unisys Customer Support.
1976:ERROR	**Not a typed buffer**	
	DESCRIPTION	An internal processing error has occurred. The buffer to convert to X_C_TYPE was null.
	ACTION	Contact Unisys Customer Support.
1977:ERROR	**Can’t find view file**	
	DESCRIPTION	The specified view file could not be found.

	ACTION	Check to make sure that the view file exists and the VIEWFILES or VIEWFILES32 environment variables contain the specified view file name. Also check to make sure that the view file is located in the directory specified by the VIEWDIR or VIEWDIR32 environment variable.
1978:ERROR	**APDU buffer overflow**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1979:ERROR	**APDU buffer overflow**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1980:ERROR	**Invalid data type for X_C_TYPE**	
	DESCRIPTION	The data type contained in the view is not a valid X_C_TYPE buffer data type. Valid X_C_TYPE data types are short, int, long, char, float, double, decimal, string, and carray.
	ACTION	Look for further error which indicate the name of the VIEW buffer which is in error. Correct the View buffer to contain the correct data types or define the buffer as X_C_TYPE.
1981:ERROR	**Not a typed buffer**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1982:ERROR	**Can’t find view file**	
	DESCRIPTION	The specified view file could not be found.

	ACTION	Check to make sure that the view file exists and the VIEWFILES or VIEWFILES32 environment variables contain the specified view file name. Also check to make sure that the view file is located in the directory specified by the VIEWDIR or VIEWDIR32 environment variable.
1983:ERROR	**APDU buffer overflow**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1984:ERROR	**APDU buffer overflow**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1985:ERROR	**APDU buffer overflow**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
1986:ERROR	**Invalid data type for X_COMMON**	
	DESCRIPTION	The data type contained in the view is not a valid X_COMMON buffer data type. Valid X_COMMON data types are short, long, char, string, and carray.
	ACTION	Look for further error which indicate the name of the VIEW buffer which is in error. Correct the View buffer to contain the correct data types or define the buffer as X_C_TYPE.
1987:ERROR	**Invalid Buffer Type**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.

1988:ERROR	**Can't get local service < %s > from configuration file**	
	DESCRIPTION	The service requested could not be found in the shared memory section defined by DM_LOCAL_SERVICES.
	ACTION	Check the configuration files and verify that the service is defined as in the DM_LOCAL_SERVICES section. If necessary, Add the service definition and restart the OSITP Gateway process.
1990:ERROR	**Incoming request not allowed access to any local services**	
	DESCRIPTION	An incoming service call was denied access because the Remote Domain is not specified the ACL List for the local service.
	ACTION	Look for further error messages which specify the local service which has been denied access. If access is required, add the remote domain to the services ACL list.
1997:ERROR	**Can't find LSVC entry by rname %s**	
	DESCRIPTION	The requested service could not be found in the shared memory DM_LOCAL_SERVICES section.
	ACTION	Check the DM_LOCAL_SERVICES section of the configuration file for the requested service. If not found add the service and restart the OSITP Gateway process.
1999:ERROR	**Invalid Buffer Type**	
	DESCRIPTION	The buffer type is not a valid buffer type.
	ACTION	Look for further error which indicate the name of the buffer which is in error.
2000:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.

	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2001:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2002:ERROR	**View file must be specified**	
	DESCRIPTION	The VIEW32 buffer did not have a buffer subtype.
	ACTION	Check the tpalloc call and make sure it specifies a subtype. Look for further User log messages which specify the buffer in error.
2003:ERROR	**Invalid Buffer Type**	
	DESCRIPTION	The buffer to be converted to X_C_TYPE is not a valid tpalloc buffer.
	ACTION	Check the tpalloc call and make sure it specifies a subtype. Look for further User log messages which specify the buffer in error.
2004:ERROR	**VIEW32 file is required**	
	DESCRIPTION	The VIEW32 buffer did not have a buffer subtype.
	ACTION	Check the tpalloc call and make sure it specifies a subtype. Look for further User log messages which specify the buffer in error.
2005:ERROR	**Missing data element in received Data Stream**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2006:ERROR	**Missing data element in received Data Stream**	
	DESCRIPTION	An Internal processing error has occurred.

	ACTION	Contact Unisys Customer Support.
2007:ERROR	**Missing data element in received Data Stream**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2008:ERROR	**Missing data element in received Data Stream**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2009:ERROR	**Missing data element in received Data Stream**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2010:ERROR	**Missing data element in received Data Stream**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2011:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2012:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2013:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2014:ERROR	**Memory allocation error**	

	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2015:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2016:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2017:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2018:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2019:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2020:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.

	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2021:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2022:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2023:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2024:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2025:ERROR	**ASN.1 DECODE failed, out of short integer range**	

	DESCRIPTION	The value to decode was larger than what would fit into a short integer.
	ACTION	Modify the VIEW field to use a long integer, recompile the VIEW as necessary.
2026:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2027:ERROR	**ASN.1 DECODE failed, out of short integer range**	
	DESCRIPTION	The value to decode was larger than what would fit into a short integer.
	ACTION	Modify the VIEW field to use a long integer, recompile the VIEW as necessary.
2028:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2029:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2030:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2031:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2032:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2033:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2034:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2035:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2036:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2037:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2038:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2039:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2040:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2041:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2042:ERROR	**Request block is invalid**	

	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2043:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2044:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2045:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2046:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2047:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2048:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.

2049:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2050:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2051:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2052:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2053:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2054:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2055:ERROR	**Memory allocation error**	

	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2056:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2057:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2058:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2059:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2060:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.

2061:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2062:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2063:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2064:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2065:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2066:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2067:ERROR	**ASN.1 DECODE failed, out of short integer range**	
	DESCRIPTION	The value to decode was larger than what would fit into a short integer.
	ACTION	Modify the VIEW field to use a long integer, recompile the VIEW as necessary.
2068:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2069:ERROR	**ASN.1 DECODE failed, out of short integer range**	
	DESCRIPTION	The value to decode was larger than what would fit into a short integer.
	ACTION	Modify the VIEW field to use a long integer, recompile the VIEW as necessary.
2070:ERROR	**Request block is invalid**	

	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2071:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2072:ERROR	**ASN.1 DECODE failed, out of integer range**	
	DESCRIPTION	The value to decode was larger than what would fit into a integer.
	ACTION	Modify the VIEW field to use a field which allows larger values, recompile the VIEW as necessary.
2073:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2074:ERROR	**ASN.1 DECODE failed, out of integer range**	
	DESCRIPTION	The value to decode was larger than what would fit into a integer.
	ACTION	Modify the VIEW field to use a field which allows larger values, recompile the VIEW as necessary.
2075:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2076:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2077:ERROR	**Received type does not match definition**	

	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2078:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2079:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2080:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2081:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2082:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2083:ERROR	**ASN.1 DECODE failed, float range out of local limit**	
	DESCRIPTION	The value to decode was larger than what would fit into a float.
	ACTION	Modify the VIEW field to use a double, recompile the VIEW as necessary.
2084:ERROR	**Received type does not match definition**	

	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2085:ERROR	**ASN.1 DECODE failed, float range out of local limit**	
	DESCRIPTION	The value to decode was larger than what would fit into a float.
	ACTION	Modify the VIEW field to use a double, recompile the VIEW as necessary.
2086:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2087:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.

2088:ERROR	**ASN.1 DECODE failed, double range out of local limit**	
	DESCRIPTION	The value to decode was larger than what would fit into a double.
	ACTION	Modify VIEW field to correct the problem.
2089:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2090:ERROR	**ASN.1 DECODE failed, double range out of local limit**	
	DESCRIPTION	The value to decode was larger than what would fit into a double.
	ACTION	Modify VIEW field to correct the problem.
2091:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2092:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2093:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2094:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2095:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2096:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2097:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2098:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2099:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2100:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2101:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2102:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2103:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2104:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2105:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2106:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2107:ERROR	**Memory allocation error**	

	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2108:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2109:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2110:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2111:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2112:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.

2113:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2114:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2115:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2116:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.

2117:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2118:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2119:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2120:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.

	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2121:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2122:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2123:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2124:ERROR	**Memory allocation error**	

	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2125:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2126:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2127:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2128:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2129:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.

2130:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2131:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2132:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2133:ERROR	**Request block is invalid**	
	DESCRIPTION	An Internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.

2134:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2135:ERROR	**Received type does not match definition**	
	DESCRIPTION	The received type does not match the definition specified by the VIEW.
	ACTION	Look for further error messages which specify the VIEW buffer type in error. This can occur when the VIEW definition used by the calling program does not match the VIEW definition used by the receiving process. Check the VIEW definitions of client and server to make sure that are identical. If a VIEW definition has been modified, make sure the associated Gateway process has been restart as VIEW definitions are cached by the Gateways.
2136:ERROR	**Memory allocation error**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for the buffer conversion.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2200:ERROR	**Cannot malloc fd structures**	

	DESCRIPTION	An Internal error has occurred while trying to allocate space for internal structures.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2201:INFO	**libgwo build: %s - %s**	
	DESCRIPTION	This line indicates the time and date that the gwnwinit.c module was built.
	ACTION	No action is necessary.
2202:ERROR	**Malloc communication control structure failed**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for internal structures.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2203:INFO	**GWOSITP is exiting**	
	DESCRIPTION	The OSITP Gateway process is exiting.
	ACTION	No action is necessary.
2204:WARN	**Specified environmental variable GW_DFLT_TRANTIME out of valid range, ignored!**	
	DESCRIPTION	The GW_DFLT_TRANTIME is larger than what will fit into a long integer.
	ACTION	A default value of 300: seconds will be used. If this is not desired, correct the GW_DFLT_TRANTIME environment variable and restart the Gateway.
2212:ERROR	**Malloc CID string buffer failed**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for internal structures.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.

2235:ERROR	**Internal data structure not allocated yet**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2236:ERROR	**Invalid input parameter**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2237:ERROR	**Internal protocol error, missing routing information**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2238:ERROR	**Remote OSITP domain not defined locally**	
	DESCRIPTION	An incoming service call was received from a remote domain which is not defined in the DM_REMOTE_DOMAINS section of the configuration file.
	ACTION	Add the necessary remote domain definition to DM_REMOTE_DOMAIN and DM_OSITP sections of the configuration file and restart the Gateway.
2239:ERROR	**Internal data structure not allocated yet**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2240:ERROR	**Invalid input parameter**	
	DESCRIPTION	A remote domain name was a NULL string.
	ACTION	Contact Unisys Customer Support.
2241:ERROR	**Internal data structure not allocated yet**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.

2242:ERROR	**Remote OSITP domain (%s) not defined in DM_OSITP**	
	DESCRIPTION	An incoming service call was received from the specified remote domain which is not defined in the DM_REMOTE_DOMAINS section of the configuration file.
	ACTION	Add the necessary remote domain definition to DM_REMOTE_DOMAINS and DM_OSITPX sections of the configuration file and restart the Gateway.
2243:ERROR	**Your eLink OSI TP license is either invalid or expired**	
	DESCRIPTION	Check your license file to make sure that is has not expired and that it contains the correct OSI TP License information.
	ACTION	Correct the license file and restart the Gateway. For more help Contact Unisys Customer Support.
2244:ERROR	**Version of Unisys (%d) and BEA (%d) binaries do not match**	
	DESCRIPTION	The versions of the Unisys and BEA dlls and executable are not compatible.
	ACTION	Reinstall the Software and restart the OSITP Gateway. For more help, contact Unisys Customer support.
2300:ERROR	**Bad input arguments, unable to create instance**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2307:ERROR	**Could not get RDOM Cache entry**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2308:ERROR	**RDOM %s OSITP address not configured**	
	DESCRIPTION	The OSITP configuration parameters for the specified RDOM are not configured in the domain configuration file.

	ACTION	Verify that a valid entry for the specified RDOM is present in DM_OSITPX section of the domain configuration file. If not, correct the problem and restart the gateway as necessary.
2320:ERROR	**Unable to get remote domain information from shmem**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2400:ERROR	**gwnwpoll error, %s**	
	DESCRIPTION	A bad return code was received from set_new_action or gw_new_action for the specified message type. This is an internal processing error.
	ACTION	Look for additional log messages which might be related to this error. Contact Unisys Customer Support.
2401:ERROR	**Could not make new action**	
	DESCRIPTION	A bad return code was received by set_new_action from gw_new_action.
	ACTION	Look for additional log messages which might be related to this error. Contact Unisys Customer Support.
2402:WARN	**Dropping polled action, no free contexts**	
	DESCRIPTION	A bad return code was received in set_new_action from gw_nw_CreateInstance or gw_nw_GetNwCtxByAction.
	ACTION	Look for additional log messages which might be related to this error. Contact Unisys Customer Support.
2403:ERROR	**Could not make new action**	
	DESCRIPTION	A bad return code was received by set_new_action from gw_new_action.

	ACTION	Look for additional log messages which might be related to this error. Contact Unisys Customer Support.
2404:WARN	**Dropping polled action, no free contexts**	
	DESCRIPTION	A bad return code was received by set_new_action from gw_nw_SetNwCtx.
	ACTION	Look for additional log messages which might be related to this error. Contact Unisys Customer Support.
2405:ERROR	**queue entry type unknown - %x**	
	DESCRIPTION	A protocol error has occurred in gw_nw_poll. An invalid entry type has been encountered.
	ACTION	Contact Unisys Customer Support.
2406:WARN	**Dropping polled action, context has been free**	
	DESCRIPTION	Network data has been received for a Network Context which no longer exists.
	ACTION	No action necessary.
2500:WARN	**Call to service (%s) aborted due to lack of system Resources**	
	DESCRIPTION	Not enough system resources were available to complete the processing of this service call.
	ACTION	For more information Contact Unisys Customer Support.
2501:ERROR	**Call to service (%s) failed with TPNOENT"**	
	DESCRIPTION	The remote service requested does not exist at the remote domain.
	ACTION	Either the service requested does not exists or the service requested exists but is not the correct type (conversational/non-conversational).
2504:ERROR	**Cannot create shared mem nettxid**	
	DESCRIPTION	An internal processing error has occurred.

	ACTION	Contact Unisys Customer Support.
2506:WARN	**Transaction table full, rejecting remote request!**	
	DESCRIPTION	The number of outstanding tranactions exceeds the maximum allowed under the currenct configuration. This could be the MAXGTT or MAXTRAN value.
	ACTION	To allow more transactions, increase the appropriate configuration parameter and reload the appropriate configuation file and restart the gateway process.
2507:ERROR	**OSI TP protocol error, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2508:ERROR	**Unable to assign id to incoming conversation**	
	DESCRIPTION	The maximum number of conversations allowed, as defined by MAXCONV, has been exceeded.
	ACTION	To allow more conversations, increase the appropriate configuration parameter and reload the appropriate configuation file and restart the gateway process.
2510:ERROR	**No conversation id associated with this fd**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2515:ERROR	**Blocking timeout occurred, deleting fd**	
	DESCRIPTION	A blocking timeout has occurred on a request to a remote host.
	ACTION	Examine the ULOG for more information concerning the reason for the failure. Check configuration and increase the BLOCKTIME value, if needed.

2519:ERROR	**Conversation id not found**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2542:ERROR	**Protocol error, DiaFSM state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2544:ERROR	**Could not find transaction node, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2557:ERROR	**Protocol error, dial state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2558:ERROR	**Protocol error, dial state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2600:ERROR	**Invalid conversation context**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2601:ERROR	**Unable to send**	
	DESCRIPTION	An attempt to send a reply to the called failed.
	ACTION	Look for further Stack vendor errors for more information as to why this failure occurred.
2803:ERROR	**Invalid transaction node**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.

2804:ERROR	**Could not find transaction node!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2805:ERROR	**Unable to obtain remote domain info from shared memory**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2806:ERROR	**Unable to obtain ositp info from rdom**	
	DESCRIPTION	The DM_OSITPX section for a remote domain is not defined in the domain configuration file.
	ACTION	Make the necessary corrections to the domain configuration file, reload the file, and restart the OSITP gateway process.
2808:ERROR	**Cannot realloc fd structures**	
	DESCRIPTION	An Internal error has occurred while trying to allocate space for Network Context structure.
	ACTION	Increase the system parameters to allow more memory per process and restart the OSITP gateway process.
2809:ERROR	**Remote domain name(%s) not found**	
	DESCRIPTION	The specified remote domain is not defined in the domain configuration file.
	ACTION	Add the remote domain information to the domain configuration file, reload the file, and restart the OSITP gateway process.
2815:ERROR	**gw_tx_end returned failure**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2821:ERROR	**Invalid input transaction id**	

	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2822:ERROR	**Exceeding local transaction limit**	
	DESCRIPTION	The number of local transaction allowed, as defined by MAXTRAN, has been exceeded.
	ACTION	To increase the number allowed, Increase the MAXTRAN parameter and restart the OSITP gateway process.
2823:ERROR	**Branch does not belong to this transaction**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2826:ERROR	**Exceeding local transaction limit**	
	DESCRIPTION	The number of local transaction allowed, as defined by MAXTRAN, has been exceeded.
	ACTION	To increase the number allowed, Increase the MAXTRAN parameter and restart the OSITP gateway process.
2829:ERROR	**Branch does not belong to this transaction**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2830:ERROR	**Transaction branch does not exist**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2831:ERROR	**Invalid input file descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2833:ERROR	**Transaction branch already exists**	

	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2834:ERROR	**Invalid input file descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2844:ERROR	**Corrupted or uninitialized file descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2845:ERROR	**Allocating Network Transaction table failed**	
	DESCRIPTION	The memory could not be allocated for the Network Transaction table.
	ACTION	Adjust system parameters to allow more memory or fewer Transactions and restart the OSITP Gateway process.
2847:ERROR	**Bad input transaction identification**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2849:ERROR	**Bad input transaction identification**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2850:ERROR	**Invalid transaction look up table entry**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2851:ERROR	**Invalid or corrupted network context descriptor %d**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.

2853:ERROR	**Invalid conversation descriptor %d**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2854:ERROR	**Invalid action descriptor %d**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2855:ERROR	**Invalid context descriptor %d**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2856:ERROR	**Invalid action descriptor %d**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2857:ERROR	**Invalid context descriptor %d**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2858:WARN	**Possible orphaned conversation %d**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2859:ERROR	**Invalid action descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2860:ERROR	**Invalid file descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.

2861:ERROR	**Invalid file descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2862:ERROR	**Invalid action descriptor %d**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2863:ERROR	**Maximum system context limit already reached**	
	DESCRIPTION	The Maximum number (32767) of system context structures have been created. It is possible that there is a memory leak.
	ACTION	Contact Unisys Customer Support.
2864:ERROR	**Corrupted or uninitialized file descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2865:ERROR	**Invalid conversation descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2866:ERROR	**Invalid file descriptor**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2867:WARN	**Remote domain(%s) XATMI encoding not specified, use default**	
	DESCRIPTION	The XATMI encoding type for the Remote Domain has not been configured in the DM_OSITPX section. The default XATMI encoding type is being used.

	ACTION	No action is necessary, unless the default encoding type is not desired. If the default is not desired, add the XATMI_ENCODING parameter value for the specified remote domain, reload the domain config file, and restart the OSITP gateway process.
2868:ERROR	**Remote OSITP domain %s address not configured**	
	DESCRIPTION	The DM_OSITPX section is not defined for the given Remote Domain.
	ACTION	Add the DM_OSITPX section to the configuration file, reload the domain config file, and restart the OSITP gateway process.
2869:ERROR	**Wrong remote domain type(%s)**	
	DESCRIPTION	The remote domain type is not OSITP.
	ACTION	Contact Unisys Customer Support.
2872:ERROR	**Cannot get status of security file**	
	DESCRIPTION	The security file does not exist.
	ACTION	Check the configuration and make sure the security file exists or reconfigure the system such that security is off.
2873:ERROR	**Cannot find appkey for user name [%s]**	
	DESCRIPTION	The specified username appkey does not exists in the security file.
	ACTION	Add the user appkey to the security file or use a username which is defined in the security file.
2874:ERROR	**Cannot find user name for appkey [%d]**	
	DESCRIPTION	The specified appkey does not exists in the security file.
	ACTION	Add the user appkey to the security file or use a username whose appkey is defined in the security file.

2875:ERROR	**Cannot open security file %s**	
	DESCRIPTION	The security file does not exist.
	ACTION	Check the configuration and make sure the security file exists or reconfigure the system such that security is off.
2902:ERROR	**failed to insert blob segment into transaction table!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support.
2903:ERROR	**Event stack overflow, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2904:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2908:ERROR	**Could not find transaction table entry, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2914:ERROR	**Protocol error, node state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2916:ERROR	**Could not find transaction node, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2917:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.

	ACTION	Contact Unisys Customer Support
2918:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2919:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2920:ERROR	**Protocol error, msg state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2921:ERROR	**Could not find transaction node, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2922:ERROR	**Could not find transaction node, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2928:ERROR	**Could not find transaction node, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2929:ERROR	**Could not find transaction node, shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2935:ERROR	**Protocol error, node state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.

	ACTION	Contact Unisys Customer Support
2936:ERROR	**Protocol error, node state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2937:ERROR	**Protocol error, node state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2939:ERROR	**Protocol error, node state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2940:ERROR	**Protocol error, node state(%d), shutdown Gateway!**	
	DESCRIPTION	An internal processing error has occurred.
	ACTION	Contact Unisys Customer Support
2947:WARN	**No OSI/TP service provider, shutdown OSITP domain gateway**	
	DESCRIPTION	The local OSITP domain is not defined in the DM_OSITPX section of the domain configuration file.
	ACTION	Check the domain configuration file definition and add the DM_OSITPX information for the Local Domain if necessary, reload the domain configuration file and restart the OSITP gateway process.

B Utilities Reference

This section covers the following utilities that are commonly used for eLink OSI TP:

- DMADM
- dmadmin
- GWADM
- UDMADM

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 non-/WS System operating systems.

INTEROPERABILITY The initial release of eLink OSI TP can only be installed on a node running Tuxedo Release 6.5 or Release 7.1.

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
GWOSITP SRVGRP="gwgrp" SRVID=1003 RQADDR="gwgrp" REPLYQ=Y
RESTART=Y MIN=1 MAX=1
```

SEE ALSO *Tuxedo /Domain Guide, Tuxedo Administrator's Guide*

dmadmin

OSITP 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 OSITP 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`.

`chbktime (chbt) –d local_domain_name –t bktime`
: 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-7.

`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.

`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 indigenous to the native operating system environment.

`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. The eLink OSI TP gateway must be shut down and restarted for new passwords to take effect.

`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) OSITP
       9) 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 `udmloadcf` 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 characte. 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_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-3 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_SER-VICES SECTION

The following table lists the fields in the DM_LOCAL_SERVICES section.

Table B-4 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-5 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-6 DM_ROUTING SECTION

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

Table B-6 DM_ROUTING SECTION

Field Identifier	Field Type	Update	Notes
TA_BUFTYPE	string	Yes	

CONFIGURING THE DM_ACCESS_CONTROL SECTION

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

Table B-7 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-8 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 an OSITP 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]
: An OSITP 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) OSITP
       9) 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                         OSITP
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                         OSITP
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) OSITP
       9) 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 OSITP 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.
```

SEE ALSO *Tuxedo /Domain Guide*

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 non-/WS operating systems.

INTEROPERABILITY The initial release of eLink OSI TP can only be installed on a node running Tuxedo Release 6.5 or Release 7.1.

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 RESTART=Y GRACE=0
GWADM SRVGRP="gwgrp" SRVID=1002 RESTART=Y GRACE=0
      CLOPT="-A -- -a on -t on"
GWTDOMAIN SRVGRP="gwgrp" SRVID=1003
RESTART=Y MIN=1 MAX=1
```

SEE ALSO `dmadmin`, `tmboot`

`dmconfig`, `DMADM`, `servopts`, `ubbconfig`

Tuxedo /Domain Guide

Tuxedo Administrator's Guide

UDMADM

/OSI TP domain administrative server.

SYNOPSIS

```
UDMADM SRVGRP = "identifier"
SRVID = "number"
```

DESCRIPTION

The UDMADM administrative server provides run-time access to the BUDMCONFIG file. When you process your DMCONFIG file using udmloadcf, two files that contain the configuration information in binary form are created. One of these, BDMCONFIG, is loaded by Tuxedo into shared memory. The other file, BUDMCONFIG, contains all of the new configuration parameters introduced in OSITP Version 4.0 that are not recognized by the current release of Tuxedo. UDMADM is a specialized Tuxedo server that reads the BUDMCONFIG file into memory, and then waits for calls requesting the configuration information. When the OSITP gateway (GWOSITP) is started up, it makes a call to UDMADM requesting this information. If UDMADM is available, the information is sent back to the gateway and normal gateway initialization continues. If the UDMADM server is not available, the request from the gateway fails, a warning message is logged, and the gateway attempts to read the BUDMCONFIG file directly. If the gateway can not read the BUDMCONFIG file an error is logged and the gateway terminates.

When UDMADM is booted, the BDMCONFIG environment variable is used to set the pathname of the BUDMCONFIG file. UDMADM is described in the *SERVERS section of the UBBCONFIG file as a server running within a group, e.g., UDMADMGRP. There should be only one instance of the UDMADM running in this group.

UDMADM is particularly useful in a multi-platform (MP) or bridged environment. In this environment, the master node provides configuration information to all of the non-master nodes. The master node should be the only place where UDMADM is configured as a server. When the gateway is started on non-master nodes, the configuration information is requested from the master node by calling the service provided by UDMADM. In this environment, the new configuration information is disseminated from the master node to the non-master nodes in the same fashion as the old configuration information.

Note: The UDMADM entry must be placed **BEFORE** GWOSITP in the UBBCONFIG file so the UDMADM server is loaded before the GWOSITP server.

UDMADM is completely optional in a single-host machine (SHM) environment. The ulog file will return several errors when GWOSITP starts up, but these are just warnings. When GWOSITP starts up, it attempts to tpcall the UDMADM to get configuration information from the binary BUDMCONFIG file . If this tpcall fails, the gateway tries to open the new binary file and read in the configuration information by itself. In the case of an SHM environment, the configuration information is local on the machine and the environment variable, BDMCONFIG, must be set. The gateway uses this environment variable to determine the directory in which to look for the BUDMCONFIG file. The gateway strips off the BDMCONFIG file name from that environment variable and just uses the directory portion, then tacks on the BUDMCONFIG file name.

PORTABILITY UDMADM is supported as an OSI TP server on non-/WS System operating systems.

INTEROPERABILITY The initial release of eLink OSI TP can only be installed on a node running Tuxedo Release 6.5 or Release 7.1.

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

```
#
*GROUPS
UDMADMGRP  LMID=mach1 GRPNO=1
gwgrp      LMID=mach1 GRPNO=2
#
*SERVERS
GWADM SRVGRP="gwgrp" SRVID=1002 REPLYQ=N RESTART=Y GRACE=0
UDMADM SRVGRP="UDMADMGRP" SRVID=1010
GWOSITP SRVGRP="gwgrp" SRVID=1003 RQADDR="gwgrp" REPLYQ=Y
RESTART=Y MIN=1 MAX=1
```

C Manually Upgrading BEA eLink OSI TP to Version 4.0

If you are upgrading from a previous version of eLink OSI TP to Version 4.0, you must modify your existing `DMCONFIG` file to create a `UDMCONFIG` file after installing eLink OSI TP.

Note: It is recommended that you use the `osiadmin` utility to upgrade your files, but you may perform this upgrade manually. Refer to the `IMPORTCFG` parameter in Using the OSI TP Administration Utility for more information about upgrading automatically.

This section covers the following topics for manually upgrading your `DMCONFIG` file:

General Checklist for Manually Upgrading Your DMCONFIG File

To manually upgrade your DMCONFIG file for OSI TP Version 4.0, perform the following General Checklist steps. Each step is detailed in the following sections.

1. After installing eLink OSI TP 4.0 locate the original DMCONFIG file, referred to as DMCONFIG.TXT, and the ositp configfile you wish to upgrade and make backup copies of the files before making any modifications.
2. Copy the existing DMCONFIG file to a new file named for example, UDMCONFIG.TXT.
3. Modify the UDMCONFIG file.

Following are the detailed steps for upgrading your UDMCONFIG file:

Step 1 - Backup Existing Configuration Files

It is recommended that you make a backup copy of your original DMCONFIG.TXT file and OSITP CONFIGFILE before making any changes or running any utilities. A backup helps prevent the loss of your current configuration and allows you to restore any information in the event there is a problem. Additionally, OSI TP 4.0 provides a utility, udmloadcf, that processes your new configuration file, UDMCONFIG.TXT, and creates a DMCONFIG file. Enter the following command from your command line or DOS prompt to copy your DMCONFIG.TXT file to DMCONFIG.bak:

```
copy DMCONFIG.TXT DMCONFIG.bak
```

Step 2 - Copy DMCONFIG.TXT to UDMCONFIG.TXT

The `udmloadcf` utility, provided with OSI TP 4.0, processes your new configuration file, `UDMCONFIG.TXT`, and creates a `DMCONFIG` file. Copy your original `DMCONFIG.TXT` file to a new file called `UDMCONFIG.TXT`. This is the configuration file you will edit for the remaining steps. Enter the following command from your command line or DOS prompt to copy your `DMCONFIG.TXT` file to `UDMCONFIG.TXT`:

```
copy DMCONFIG.TXT UDMCONFIG.TXT
```

Step 3 - Modify Parameters in the UDMCONFIG File

You must change the `DM_OSITP` section name to `DM_OSITPX`, and you must modify the `TYPE` field in the `DM_LOCAL_DOMAINS` and `DM_REMOTE_DOMAINS` sections to be `TYPE=OSITPX`. In order to use some of the new features of OSI TP 4.0, you may optionally want to modify the `DM_LOCAL_SERVICES` and `DM_REMOTE_SERVICES` definitions as well. Changes to each of these sections consist of removing obsolete parameters, adding new parameters, and modifying values for existing parameters. You can modify entries in the `UDMCONFIG` file using any text editor.

Following is a summary of the changes that need to be made to your configuration:

Section	Parameter	Change
`DM_OSITP`		Obsolete
`DM_OSITPX`		New
`DM_OSITPX`	`ACN`	Obsolete
	`AEID`	Obsolete

Section	Parameter	Change
	AEQ	Obsolete
	AET	Now required for each domain
	APID	Obsolete
	APT	Obsolete
	DNS_RESOLUTION	New
	MAX_LISTENING_EP	Obsolete
	NWADDR	Now required for each domain
	NWDEVICE	Obsolete
	OPTIONS	New
	P_SEL	New
	PROFILE	Obsolete
	REM_TPSUT	New
	S_SEL	New
	T_SEL	New
	TAILOR_PATH	New
	URCH	Obsolete
DM_LOCAL_DOMAINS	SECURITY	New
	TYPE=OSITP	Obsolete
	TYPE=OSITPX	New
DM_LOCAL_SERVICES	COUPLING	New
	INRECTYPE	New
	OUTRECTYPE	New

Section	Parameter	Change
DM_REMOTE_DOMAINS	TYPE=OSITP	Obsolete
	TYPE=OSITPX	New
DM_REMOTE_SERVICES	AUTOPREPARE	New
	INRECTYPE	New
	OUTRECTYPE	New
	REM_TPSUT	New
	TPSUT_TYPE (must be specified **BEFORE** REM_TPSUT)	New

Refer to Understanding the UDMCONFIG File for more information about these new parameters.

Index

D

E

P

R

S

T

U

V

W

X