# BEA Tuxedo

## Glossary

BEA Tuxedo Release 7.1
Document Edition 7.1
May 2000

## Copyright

## Restricted Rights Legend

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

## Trademarks or Service Marks

**BEA Tuxedo Glossary**

| Document Edition | Date | Software Version |
|---|---|---|
| 7.1 | May 2000 | BEA Tuxedo Release 7.1 |

# Contents

## Glossary

abort .... G-1
Access Control List (ACL) .... G-1
access machine .... G-1
ACID properties .... G-1
ACL .... G-2
activate .... G-2
active server .... G-2
ADE .... G-2
administrative API .... G-2
administrative domain .... G-2
administrator .... G-2
advertised .... G-2
AEQ .... G-3
AET .... G-3
allocation .... G-3
alternate facility .... G-3
alternate remote domain .... G-3
AP .... G-3
API .... G-3
applet .... G-3
application .... G-3
application association .... G-3
application context .... G-4
application context name (ACN) .... G-4
application development environment (ADE) .... G-4
application entity .... G-4
application entity qualifier (AEQ) .... G-4
application entity title (AET) .... G-4
application framework .... G-4
application program (AP) .... G-4

application programming interface (API) ........ G-4
Application-to-Transaction Monitor Interface (ATMI) ........ G-5
architecture ........ G-5
asynchronous ........ G-5
asynchronous connections ........ G-5
asynchronous process ........ G-5
asynchronous request ........ G-5
ATMI ........ G-5
auditing ........ G-6
audit trail ........ G-6
authentication ........ G-6
authorization ........ G-6
autoinstall ........ G-6
auto-spawning ........ G-6
availability ........ G-6
backup ........ G-6
BBL ........ G-6
BEA Administration Console ........ G-7
BEA Transaction Processing ........ G-7
BEA Tuxedo application ........ G-7
BEA Tuxedo client ........ G-7
BEA Tuxedo domain ........ G-7
BEA Tuxedo Domains ........ G-7
BEA Tuxedo server ........ G-7
BEA Tuxedo system ........ G-8
bitmap ........ G-8
blocking ........ G-8
blocking mode ........ G-8
BRIDGE ........ G-8
broadcast ........ G-8
broker ........ G-8
buffer types ........ G-8
Bulletin Board ........ G-9
Bulletin Board Liaison (BBL) ........ G-9
byte ........ G-9
C++ ........ G-9
cache ........ G-9
CARRAY buffer ........ G-9
catalog ........ G-9
CCR ........ G-9
certificate authority (CA) ........ G-9

channel ........ G-10
cipher ........ G-10
ciphertext ........ G-10
class library ........ G-10
client ........ G-10
client naming ........ G-10
client program ........ G-10
client/server computing ........ G-10
closed framework ........ G-11
command-line interface ........ G-11
commit ........ G-11
compact disc-read only memory (CD-ROM) ........ G-11
component ........ G-11
concurrency ........ G-11
concurrent ........ G-11
configuration ........ G-11
configuration set ........ G-12
configure ........ G-12
connection ........ G-12
connection-oriented communication ........ G-12
consistent state ........ G-12
conversation ........ G-12
conversational ........ G-12
conversational communication ........ G-12
conversational server ........ G-12
conversational service ........ G-12
cryptography ........ G-13
CSI ........ G-13
current context ........ G-13
daemon ........ G-13
daisy chain ........ G-13
DASD ........ G-13
database ........ G-13
DataBase2 (DB2) ........ G-13
database management system (DBMS) ........ G-13
data-dependent routing ........ G-13
Data Encryption Standard (DES) ........ G-14
data independence ........ G-14
data transfer protocol ........ G-14
DB2 ........ G-14
DBBL ........ G-14

DBMS ........ G-14
DDE ........ G-14
DDE conversation ........ G-14
deadlock ........ G-14
decoding ........ G-15
decryption ........ G-15
default ........ G-15
default context ........ G-15
deferred synchronous communication ........ G-15
deployment ........ G-15
DES ........ G-15
design document ........ G-15
design pattern ........ G-15
dialog ........ G-15
digital certificate ........ G-16
digital signature ........ G-16
digital signature algorithm ........ G-16
Direct Access Storage Device (DASD) ........ G-16
Distinguished Bulletin Board Liaison (DBBL) ........ G-16
distributed application ........ G-17
distributed computing ........ G-17
distributed transaction ........ G-17
distributed transaction processing (DTP) ........ G-17
DLL ........ G-17
domain ........ G-17
domain gateway ........ G-17
Domains-level failback ........ G-17
Domains-level failover ........ G-18
dotted-decimal notation ........ G-18
DSA ........ G-18
DTP ........ G-18
dynamic argument ........ G-18
Dynamic Data Exchange (DDE) ........ G-18
dynamic datatype ........ G-18
dynamic link libraries (DLL) ........ G-18
electronic commerce ........ G-18
encoding ........ G-19
encryption ........ G-19
encryption ........ G-19
environment variable ........ G-19
event ........ G-19

Event Broker/Monitor ........................................................................ G-19
event posting ........................................................................ G-19
event subscription ........................................................................ G-19
extensibility ........................................................................ G-19
External Data Representation (XDR) ........................................................................ G-20
failback ........................................................................ G-20
failover ........................................................................ G-20
field manipulation language (FML) ........................................................................ G-20
field table ........................................................................ G-20
FML ........................................................................ G-21
FML buffer ........................................................................ G-21
foreign access path ........................................................................ G-21
foreign node ........................................................................ G-21
format independence ........................................................................ G-21
framework ........................................................................ G-21
gateway ........................................................................ G-21
gateway group ........................................................................ G-22
gateway server ........................................................................ G-22
global transaction ........................................................................ G-22
Global Transaction Identifier (GTRID) ........................................................................ G-22
graphical user interface (GUI) ........................................................................ G-22
group ........................................................................ G-22
GUI ........................................................................ G-22
handler ........................................................................ G-22
hierarchical database ........................................................................ G-23
hierarchy ........................................................................ G-23
high-level language ........................................................................ G-23
host ........................................................................ G-23
host computer ........................................................................ G-23
hypertext markup language (HTML) ........................................................................ G-23
inactive server ........................................................................ G-23
incoming connection ........................................................................ G-23
information hiding ........................................................................ G-23
infrastructure ........................................................................ G-23
integration ........................................................................ G-24
interaction model ........................................................................ G-24
interactive ........................................................................ G-24
interactive interface ........................................................................ G-24
internationalization ........................................................................ G-24
International Standards Organization (ISO) ........................................................................ G-24
Internet ........................................................................ G-24

Internet Protocol Address (IP Address) ........ G-24
interoperability ........ G-25
intranet ........ G-25
IP Address ........ G-25
ISO ........ G-25
journaling ........ G-25
Kerberos protocol ........ G-25
Kerberos security ........ G-25
keyword ........ G-25
LAN ........ G-25
LAN partition ........ G-25
lazy connection ........ G-26
LDAP ........ G-26
legacy application ........ G-26
link-level encryption (LLE) ........ G-26
link-level failover in Domains ........ G-26
listener ........ G-26
LMID ........ G-27
load balancing ........ G-27
local ........ G-27
local application names ........ G-27
Local Area Network (LAN) ........ G-27
local domain ........ G-27
local gateway ........ G-27
local node ........ G-28
local service ........ G-28
local system ........ G-28
local transaction ........ G-28
location transparency ........ G-28
log file ........ G-28
logical machine ID (LMID) ........ G-28
makefile ........ G-28
managed object ........ G-28
Management Information Base (MIB) ........ G-29
mapping ........ G-29
master node ........ G-29
MD5 ........ G-29
message ........ G-29
message catalog ........ G-29
message digest ........ G-29
message digest algorithm ........ G-30

MIB ........ G-30
middleware ........ G-30
migrate ........ G-30
model ........ G-30
modeling ........ G-30
module ........ G-30
MP model ........ G-30
multicontexted process ........ G-31
multi-domain client ........ G-31
multiple listening addresses ........ G-31
Multiple Virtual Storage (MVS) ........ G-31
multiprocessor ........ G-31
multiprogramming ........ G-31
multithreading ........ G-31
mutual authentication ........ G-31
MVS ........ G-31
MVS console ........ G-32
name server ........ G-32
native client ........ G-32
native node ........ G-32
network ........ G-32
network address ........ G-32
Network Agent (NA) ........ G-32
Network Interface (NI) ........ G-32
Node Manager (NM) ........ G-33
network provider ........ G-33
NLS ........ G-33
node ........ G-33
nonblocking mode ........ G-33
nonmaster node ........ G-33
nonpartitioned ........ G-33
*n*-tier client/server ........ G-33
octet ........ G-33
OLTP ........ G-34
online transaction processing (OLTP) ........ G-34
open framework ........ G-34
open system ........ G-34
Open Systems Interconnect Commitment, Concurrency, and Recovery (OSI CCR) ........ G-34
Open Systems Interconnection (OSI) ........ G-34
Open Systems Interconnect Transaction Processing (OSI TP) ........ G-34

OSI CCR ........ G-34
OSI TP ........ G-35
outgoing connection ........ G-35
out-of-band data ........ G-35
parallel data circuits ........ G-35
partition ........ G-35
PID ........ G-35
PKCS-7 ........ G-35
plaintext ........ G-35
platform ........ G-35
portability ........ G-36
port number ........ G-36
precommit ........ G-36
primary remote domain ........ G-36
principal ........ G-36
private ........ G-36
private MIB ........ G-36
procedure ........ G-36
process ID (PID) ........ G-36
profile ........ G-36
protocol ........ G-37
provider ........ G-37
public key algorithm ........ G-37
Public-Key Cryptography Standard 7 (PKCS-7) ........ G-37
public key encryption ........ G-37
public key security ........ G-37
queue ........ G-38
RC2 ........ G-38
RC4 ........ G-38
record ........ G-38
recover ........ G-38
recovery ........ G-38
relational database ........ G-38
reliability ........ G-39
remote ........ G-39
remote client ........ G-39
remote domain ........ G-39
remote file sharing (RFS) ........ G-39
remote gateway ........ G-39
remote node ........ G-39
remote procedure call (RPC) ........ G-39

remote service........ G-39
remote service fan-out........ G-39
remote service name........ G-40
request........ G-40
request/response communication........ G-40
request/response server........ G-40
request/response service........ G-40
requester........ G-40
requestor........ G-40
resource manager (RM)........ G-40
resource manager instance........ G-41
response time........ G-41
RM........ G-41
roll back........ G-41
rollback........ G-41
RPC........ G-41
RSA........ G-41
RTQ........ G-41
run-time trace........ G-41
scalability........ G-42
scope........ G-42
security........ G-42
security principal........ G-42
server........ G-42
server abstraction........ G-42
server group........ G-42
server ID........ G-42
service........ G-42
service code........ G-43
service request........ G-43
service routine........ G-43
SERVICES section........ G-43
session key........ G-43
SHA-1........ G-43
SHM model........ G-43
single threading........ G-43
socket........ G-44
socket descriptor........ G-44
socket ID........ G-44
socket number........ G-44
SQL........ G-44

standard MIB ..... G-44
StarLAN ..... G-44
state ..... G-44
stateful application ..... G-44
stateless application ..... G-44
STRING buffer ..... G-45
Structured Query Language (SQL) ..... G-45
subroutine ..... G-45
subscriber ..... G-45
symmetric key algorithm ..... G-45
synchronization ..... G-45
synchronous ..... G-45
synchronous communication ..... G-46
synchronous process ..... G-46
system ..... G-46
system administration ..... G-46
system administrator ..... G-46
system operations ..... G-46
task ..... G-46
TCP/IP ..... G-46
TDomain gateway ..... G-46
terminal ..... G-47
threading ..... G-47
three-tier client/server architecture ..... G-47
TLI ..... G-47
tlisten ..... G-47
TLOG ..... G-47
TM ..... G-47
TMS ..... G-47
TP ..... G-47
TP monitor ..... G-47
TP protocols ..... G-47
TPSUT ..... G-47
transaction ..... G-48
transaction coordinator ..... G-48
transaction log (TLOG) ..... G-48
transaction manager (TM) ..... G-48
Transaction Manager Server (TMS) ..... G-48
transactions per second (TPS) ..... G-48
transaction processing (TP) ..... G-49
transaction processing monitor (TP monitor) ..... G-49

transaction processing protocols (TP protocols) .......... G-49
transaction processing service user title (TPSUT) .......... G-49
translation .......... G-49
Transmission Control Protocol/Internet Protocol (TCP/IP) .......... G-49
transport interface .......... G-49
Transport Layer Interface (TLI) .......... G-49
transport provider .......... G-49
TUXCONFIG file .......... G-50
Tuxedo .......... G-50
Tuxedo domain .......... G-50
two-phase commit (2PC) .......... G-50
two-tier client/server .......... G-50
TX interface .......... G-50
type conversion .......... G-50
typed buffer .......... G-50
UBBCONFIG file .......... G-50
unadvertised .......... G-51
uniprocessor .......... G-51
Universal Device List (UDL) .......... G-51
URL .......... G-51
user datagram protocol (UDP) .......... G-51
view .......... G-51
VIEW buffer .......... G-51
VIEW definitions .......... G-51
virtual machine .......... G-51
virtual one-hop network .......... G-52
Volume Table of Contents (VTOC) .......... G-52
VTOC .......... G-52
WAN .......... G-52
Web GUI .......... G-52
Wide Area Network (WAN) .......... G-52
window .......... G-52
Workstation .......... G-52
Workstation client (WSC) .......... G-52
Workstation Handler (WSH) .......... G-53
Workstation Listener (WSL) .......... G-53
wrap .......... G-53
XA .......... G-53
XATMI application service element .......... G-53
XATMI interface .......... G-53
X_C_TYPE buffer .......... G-53

X_COMMON buffer ........ G-54
XDR ........ G-54
X_OCTET buffer ........ G-54
X/Open ........ G-54
X/Open Distributed Transaction Processing (DTP) model ........ G-54

# Glossary

**abort**

Terminate a transaction so that all protected resources, such as database records, have the same value they had at the beginning of the transaction.

**Access Control List (ACL)**

A BEA Tuxedo security feature that controls client access to services by means of lists of clients that are authorized to receive specific services. When ACL security is being used, the BEA Tuxedo system checks the appropriate list whenever a client requests a service, to determine whether the client is authorized to access the service.

**access machine**

The processor within the administrative domain of an application at which a client first accesses the system. For a native client, this is the processor on which it is running. For a workstation client, this is the site at which it contacts the application.

**ACID properties**

The essential characteristics of transaction processing systems include the following:

- Atomicity, which requires that all changes made by a transaction to a database are made permanent, or else all are nullified.
- Consistency, which defines the success of a transaction as the transformation of a database from one valid state to another.
- Isolation, which requires that changes made by a transaction to a database must not be visible to other operations until the transaction completes its work.
- Durability, which guarantees that changes made by a transaction to a database survive future system or media failures.

**ACL**

*See* **Access Control List (ACL)**.

**activate**

To move a server from the inactive (unavailable) state to the state in which it is up and running (active).

**active server**

A BEA Tuxedo server that is either processing a service request or is available to do so.

**ADE**

*See* **application development environment (ADE)**

**administrative API**

An Application Programming Interface that uses the BEA Tuxedo ATMI for the purpose of configuring and controlling an application. It is sometimes referred to as the MIB.

*See also* **application programming interface (API)** and **Management Information Base (MIB)**.

**administrative domain**

That portion of an application that is actively administered at run time by a BBL process. It does not include workstations or host processors.

**administrator**

The person who installs the BEA Tuxedo system, configures and monitors a BEA Tuxedo application, and updates application information (such as computer names and locations).

**advertised**

A service is advertised when a service table entry exists for it in the BEA Tuxedo Bulletin Board. When a Domains gateway server is booted, it advertises all the remote services that it is importing from remote domains in the Bulletin Board of the local domain (that is, the domain on which the gateway server is booted). After a remote service is advertised by a domain gateway server, it remains advertised until either an `unadvertise` command is issued or a MIB request removes the service.

**AEQ**

*See* **application entity qualifier (AEQ)**.

**AET**

*See* **application entity title (AET)**.

**allocation**

The assignment of various types of programs and record categories for system storage locations, such as main storage or disk storage.

**alternate facility**

In distributed transaction programming, a session that a transaction obtains by means of an ALLOCATE command.

**alternate remote domain**

A remote domain that is used when a primary remote domain is unavailable.

**AP**

*See* **application program (AP)**.

**API**

*See* **application programming interface (API)**.

**applet**

An interactive Java program that can be run via a Web page displayed by a Java-capable browser. An applet enhances the Web page's display or enables the user to complete a task.

**application**

A set of one or more local or remote clients that communicate with one or more servers residing on one or more machines running one or more business applications. Each application comprises the servers, services, and associated resource managers defined by a single configuration file (`TUXCONFIG`). One BEA Tuxedo application can communicate with another BEA Tuxedo application through a domain gateway group.

**application association**

The association between a process and a BEA Tuxedo application (domain). A multicontexted process may have associations with multiple BEA Tuxedo domains. It may also have multiple associations with the same domain.

**application context**

A reference to a particular application association. In the BEA Tuxedo system, an application context is set via an explicit call and is then used implicitly by subsequent ATMI calls. Thus, in an implicit context interface, the terms *application context* and *default context* are often used interchangeably.

**application context name (ACN)**

A set of rules that govern associations between application entities.

**application development environment (ADE)**

A set of tools (often presented or accessed via a GUI) to help programmers build applications.

**application entity**

A set of software components that make up a distributed transaction processing application on one computer system.

**application entity qualifier (AEQ)**

The locally unique component of a title that is used to identify an OSI TP application entity.

**application entity title (AET)**

The globally unique component of the title that is used to identify an OSI TP application entity.

**application framework**

The software that provides the infrastructure that makes it possible for sets of applications or other software components to work together.

**application program (AP)**

A single instance of a user program that performs one or more specific tasks. An AP defines transaction boundaries and accesses resources within those boundaries; it interacts with other system components using interfaces specified in the X/Open Distributed Transaction Processing model. An AP is a single thread of control involved in, at most, one global transaction at any time.

**application programming interface (API)**

1) The functions and environment that exist at the application level to support a particular system software product. 2) A set of code that enables a developer to initiate and complete client/server requests within an application. 3) A set of

calling conventions that define how to invoke a service. A set of well-defined programming interfaces (entry points, calling parameters, and return values) by which one software program utilizes the services of another.

**Application-to-Transaction Monitor Interface (ATMI)**

The Application-to-Transaction Monitor Interface is the application programming interface to the BEA Tuxedo system. It includes transaction routines, message handling routines, service interface routines, and buffer management routines.

**architecture**

1) A hardware and software platform (for example, SPARC/Solaris, Intel Pentium/Windows NT, or Intel Pentium/Linux). 2) The structure and interrelationship of components in a system or in an environment.

**asynchronous**

An event that occurs at a time that is unrelated to the time at which another event occurs. The two events are mutually asynchronous. The relationship between the times at which they occur is unpredictable.

**asynchronous connections**

Virtual circuits set up to execute independently of each other, that is, asynchronously. An asynchronous connection does not block the processing of working circuits while attempts are being made to reconnect failed circuits. The `BRIDGE` allows the use of nonfailing network paths by listening and transferring data using multiple network address endpoints.

**asynchronous process**

A process that executes independently of another process. When a request is processed asynchronously, the client application continues to perform other operations while it waits for the service request to be filled.

**asynchronous request**

A request that lets the client do other work while the request is being processed, enhancing parallelism within an application.

**ATMI**

*See* **Application-to-Transaction Monitor Interface (ATMI)**.

**auditing**

A security mechanism that involves saving a secure, tamper-resistant record of requested system operations, along with the identity of the requesting party.

**audit trail**

A manual or computerized means for tracing the transactions affecting the contents of a record.

**authentication**

The validation process used by the server to identify an entity, such as a user or a process, before allowing that user or process to join an application. This process relies on the use of passwords and other security mechanisms.

**authorization**

The process of determining what an entity, such as a user or a process, is allowed to access.

**autoinstall**

A method of creating and installing terminal definitions dynamically as terminals log on, and deleting them at logoff.

**auto-spawning**

The creation, by a BEA Tuxedo application server, of new threads to handle messages as they are received. The number of new threads that can be created is restricted by a configurable limit.

**availability**

Features of transaction processing systems that contribute to the smooth, continued systems operation in the presence of failures.

**backup**

The act of a resource manager in using a log to restore resources to some predetermined state by applying log entries sequentially to the resource until the desired state has been achieved.

**BBL**

*See* **Bulletin Board Liaison (BBL)**.

**BEA Administration Console**

A graphical user interface for the administration of BEA Tuxedo applications, that is accessed through a Web browser.

**BEA Transaction Processing**

*See* **transaction processing (TP)**.

**BEA Tuxedo application**

A business program, built around the BEA Tuxedo system, that is defined and controlled by a single BEA Tuxedo configuration file. Such an application may comprise one or more clients (local or remote), one or more servers, and one or more machines. Multiple BEA Tuxedo applications can communicate with each other through a domain gateway group.

**BEA Tuxedo client**

A program that invokes operations to talk to BEA Tuxedo servers. Remote and native clients are the same. Their requests are handled differently and transparently, depending on whether the client is located on a machine that is running in the BEA Tuxedo domain. BEA Tuxedo native clients are always co-located with the BEA Tuxedo domain in the same machine.

*See also* **client**.

**BEA Tuxedo domain**

A BEA Tuxedo domain consists of a BEA Tuxedo application running one or more business applications. A single domain is defined in one configuration file and is administered as a single entity. A domain can be connected to other BEA Tuxedo domains through the Domains feature.

*See also* **domain**, **TUXCONFIG file**, *and* **UBBCONFIG file**.

**BEA Tuxedo Domains**

A BEA Tuxedo component that extends the BEA Tuxedo system client/server model to provide transaction interoperability across TP domains. This extension preserves the client/server model and the ATMI interface by making access to services on a remote domain (and service requests from a remote domain) transparent to both the application programmer and user.

**BEA Tuxedo server**

A program that performs a task requested of it by a client application.

*See also* **server**.

**BEA Tuxedo system**

A robust e-commerce platform from BEA Systems, Inc. for developing and deploying business-critical client/server applications. It handles distributed transaction processing, application messaging, and the full complement of services necessary to build and run enterprise-wide applications.

**bitmap**

In temporary storage, a control block used by intrapartition transient data to show the VSAM control intervals that have been used and are available. It is updated whenever a control interval or track is assigned to or released from a destination.

**blocking**

The process of combining two or more records into one block.

**blocking mode**

A synchronous style of message delivery, where a program must wait for an action to complete before the program can proceed. Contrast with nonblocking mode.

**BRIDGE**

The BEA Tuxedo system process that maintains virtual circuits to other nodes participating in an application for the purpose of transferring application messages between the nodes.

**broadcast**

To send the same message to every node on a network.

**broker**

A system-level entity whose role is to maintain subscriptions and to cause subscribers' actions to occur when events are posted.

**buffer types**

An abstract name for a message type. BEA Tuxedo provides predefined types for message communication: FML, VIEW, STRING, CARRAY, and XML. These buffer types are transparently encoded and decoded across a network of heterogeneous machines. Applications can define additional types.

**Bulletin Board**

A collection of shared data structures designed to keep track of a running BEA Tuxedo system application. It contains information about servers, services, clients, and transactions pertaining to a BEA Tuxedo application. The Bulletin Board is replicated on each logical native (non-foreign) machine in the application.

**Bulletin Board Liaison (BBL)**

A BEA Tuxedo administrative process responsible for maintaining a copy of the Bulletin Board on a particular processor. When the system is running, one BBL process runs continuously on each logical machine in the application.

**byte**

A sequence of eight adjacent bits that are operated on as a unit.

**C++**

An object-oriented programming language developed at AT&T Bell Laboratories in the early 1980s. C++ is a “hybrid” language based on the non-object-oriented C language.

**cache**

A subset of memory that contains copies of the frequently accessed parts of a larger memory.

**CARRAY buffer**

A data structure that is an array of characters any of which can be the null character. The interpretation of the array is entirely application dependent.

**catalog**

*See* **message catalog**.

**CCR**

The Commitment, Concurrency, and Recovery OSI standard.

**certificate authority (CA)**

A well-known and trusted entity that issues public-key certificates. A certificate authority attests to a user’s real-world identity, much as a Notary Public does so.

**channel**

A functional unit, controlled by a processor, that handles the transfer of data between the processor and local input/output devices.

**cipher**

In cryptography, a coding system used to create encrypted messages.

**ciphertext**

In cryptography, text that is encrypted.

**class library**

A set of client programming tools. These tools can be used in a Java program or Web page-embeddable Java applet.

**client**

A process that generates service requests (that are sent to servers by the BEA Tuxedo system) and receives the servers' responses (which are handled by the BEA Tuxedo system). If the client is located on a machine that is part of the relevant BEA Tuxedo domain (that is, the domain to which the target servers belong), then the client is called a "native client." If the client is located on a machine that is outside that domain, then the client is called a "remote client" or a "Workstation client." (Remote clients communicate with servers through the BEA Tuxedo Workstation component.)

*See also* **domain** and **server**.

**client naming**

A BEA Tuxedo feature that enables client programs to carry both user names and client name values.

**client program**

Source code for a client.

*See also* **client**.

**client/server computing**

A programming model in which application programs are structured as clients or servers to achieve distributed processing. A client program is an application program that requests services to be performed. A server program is an entity that dispatches service routines to satisfy requests from client programs. A service routine is an application program module that performs one or more specific functions on behalf of client programs.

Client/server computing can be configured in a 2-tier or a 3-tier structure. A 2-tier configuration consists of only the client and the server. A 3-tier configuration includes a client, a server, and an intermediate level that acts as a router or a broker.

**closed framework**

A software infrastructure in which developers cannot remove and replace software components easily in a “plug-and-play” fashion.

**command-line interface**

A style of user interface that allows user interaction by entering commands at a system prompt.

**commit**

1) Complete a transaction so that changes are recorded and stable. Protected resources are released. 2) The declaration or process of making a transaction’s updates and messages visible to other transactions. When a transaction commits, all its effects become public and durable. After commitment, the effects of a transaction cannot be reversed automatically.

**compact disc-read only memory (CD-ROM)**

A disk from which data is read optically by laser and on which the data cannot be modified except under special conditions.

**component**

Part of an application.

**concurrency**

The simultaneous execution of more than one function or process.

**concurrent**

Pertaining to the occurrence of two or more activities within a given interval of time. Concurrent processes can alternately use shared common resources.

**configuration**

The set of hardware, hardware options, software, and software setup on a computer or on a network.

**configuration set**

The name or the number used to reference a particular configuration in a configuration partition. Each configuration set describes the services to be used when the configuration is active.

**configure**

To customize hardware and software for a computer or for a network.

**connection**

A half-duplex communication channel between processes.

**connection-oriented communication**

Communication between two BEA Tuxedo system processes over a connection.

**consistent state**

A condition in which shared data is correct and valid.

**conversation**

A dialog over a connection.

**conversational**

Communication characterized by one or more messages exchanged by a requestor and a server such that the server remains dedicated to the communication until the termination of the exchange.

**conversational communication**

*See* **conversational**.

**conversational server**

A server that offers services that require a connection to have a conversation with the requester. The conversation follows an application-established protocol. A conversational service must conform to the startup and termination rules of BEA Tuxedo system services.

**conversational service**

A service routine that is invoked by means of conversational communication from a client program. When the connection is established and the service is invoked, the client and service exchange data in a manner specific to the application. When the service returns, the connection ends.

**cryptography**

The art of protecting information by transforming it (encrypting it) into an unreadable format, called **ciphertext**. Only those who possess a secret key can decipher (or decrypt) the message into **plaintext**.

**CSI**

The API for the BEA TOP END system.

**current context**

Clients may initialize to multiple contexts; however, at any given time, in any particular thread, only one of these contexts may be the current context.

**daemon**

A system process that runs in the background.

**daisy chain**

In intercommunication, the chain of sessions that results when a system requests a resource in a remote system, and the remote system discovers that the resource is in a third system and in turn makes a remote request.

**DASD**

*See* **Direct Access Storage Device (DASD)**.

**database**

A collection of interrelated or independent data items stored together without redundancy to serve one or more applications.

**DataBase2 (DB2)**

IBM relational database.

**database management system (DBMS)**

A program or set of programs that lets users structure and manipulate the data in the tables of a database. A DBMS ensures privacy, recovery, and integrity of data in a multi-user environment.

**data-dependent routing**

1) Routing that directs a request to be processed by a particular group based on the value in a data field of the message. 2) A mechanism in the BEA Tuxedo system in which a service request is mapped to a specific server group based on a value contained in a designated field in the data buffer.

**Data Encryption Standard (DES)**

A symmetric key algorithm adopted by the U.S. government in 1976 as a standard encryption-decryption system for unclassified data in the United States. Several types of DES are offered, including DES-CBC and two-key triple-DES.

- DES-CBC is a 64-bit block cipher run in Cipher Block Chaining (CBC) mode. It provides 56-bit keys; 8 parity bits are stripped from the full 64-bit key.
- Two-key triple-DES is a 128-bit block cipher run in Encrypt-Decrypt-Encrypt (EDE) mode. Two-key triple-DES provides two 56-bit keys (in effect, a 112-bit key).

**data independence**

The ability to request data by a high-level data-management method without concern as to how the data is stored or retrieved.

**data transfer protocol**

A set of rules for transforming data of a particular buffer type from one representation into another.

**DB2**

*See* **DataBase2 (DB2)**.

**DBBL**

*See* **Distinguished Bulletin Board Liaison (DBBL)**.

**DBMS**

*See* **database management system (DBMS)**.

**DDE**

*See* **Dynamic Data Exchange (DDE)**.

**DDE conversation**

The sending and receiving of DDE messages between a client application and a server application.

**deadlock**

1) Unresolved contention for the use of a resource. 2) An error condition in which processing cannot continue because each of the two elements of the process is waiting for an action or a response from the other.

**decoding**

The conversion of encoded data back to the native format.

*See also* **encoding**.

**decryption**

The process of restoring encrypted data to its original form.

**default**

The value assumed by a program if a value is not supplied by the user.

**default context**

A BEA Tuxedo application association that is referenced by subsequent ATMI calls if tpsetctxt() is not called. The default context may differ from thread to thread. This term is often used interchangeably with **application context**.

**deferred synchronous communication**

A form of asynchronous communication in which one piece of software can send a message to another piece of software, and then continue to work and retrieve the reply to the message at some later time.

**deployment**

The process of placing an application in a distributed environment and making the application available for use. Deployment can include such tasks as installation, configuration, and administration of various parts of the application.

**DES**

*See* **Data Encryption Standard (DES)**.

**design document**

The document written by the system integrator that explains the overall design of the application or the framework to be built.

**design pattern**

A document that encapsulates, in a structured format, solutions to design problems. Design patterns are guides to good design practices.

**dialog**

A process of sending and receiving information.

**digital certificate**

An electronic file used to identify individuals and resources over networks such as the Internet. A digital certificate securely binds the identity of an individual or resource, as verified by a trusted third party known as a Certificate Authority, to a particular public key. Because no two public keys are ever identical, a public key can be used to identify its owner.

BEA Tuxedo public key security recognizes certificates that comply with X.509 Version 3.

**digital signature**

A digital code that can be attached to an electronically transmitted message that uniquely identifies the sender. Like a written signature, the purpose of a digital signature is to ensure that a message is genuine, that it has arrived exactly as it was sent, and that it comes from the stated source.

Digital signatures are especially important for electronic commerce and are a key component of most authentication schemes. The recipient of signed data can use a digital signature to prove to a third party that the signature was, in fact, generated by the signatory. When such proof is provided, the signed data is protected by *nonrepudiation*: the signatory cannot, at a later time, deny authorship of the signature.

**digital signature algorithm**

An algorithm that transforms a message of any length to a digital signature in such a way that it is computationally infeasible to (1) find two messages with the same digital signature, (2) produce a message from a given, predetermined digital signature, or (3) find the digital signature of a given message without knowledge of the sender’s private key. Typically, a digital signature algorithm is implemented by computing a message digest on the message, then encrypting the message digest with the sender’s private key.

An example of a digital signature algorithm is DSA.

**Direct Access Storage Device (DASD)**

A disk, disk drive, or group of disks or drives on an IBM machine.

**Distinguished Bulletin Board Liaison (DBBL)**

A BEA Tuxedo administrative process that runs on the MASTER node of the application and communicates with BBLs to coordinate updates to the Bulletin Board.

**distributed application**

An application that is separated into two or more parts (such as a client and a server) on different computers that communicate through a network.

**distributed computing**

An application design and implementation strategy that separates an application into units that are executed on different computers and communicate through a network. For example, an application can be separated into three distributed units: a user interface unit, a processing unit, and a storage unit.

**distributed transaction**

A transaction involving multiple transaction managers. In a distributed transaction environment, a client application may send requests to several servers resulting in resource updates at multiple resource managers. To complete the transaction, the transaction manager for each participant (client, servers, and resource managers) must be polled to coordinate the commit process for each participant within its domain.

**distributed transaction processing (DTP)**

A form of processing in which multiple application programs update multiple resources (such as databases) in a coordinated manner. Programs and resources can reside on one or more computers access a network.

**DLL**

*See* **dynamic link libraries (DLL)**.

**domain**

*See* **BEA Tuxedo domain**. *See also* **application** and **UBBCONFIG file**.

**domain gateway**

A highly asynchronous multitasking server provided by the BEA Tuxedo system to process service requests to and from remote domains. The gateway makes access to services on a remote domain (and service requests from a remote domain) transparent to both the application programmer and the user.

**Domains-level failback**

The restoration of message traffic to a primary remote domain. The **TDomain gateway** always tries to use the primary domain or the highest-level alternate remote domain defined for a service. When these domains become unavailable (due to circuit failure or other reasons), the gateway transfers message traffic to a lower-priority alternate remote domain, and periodically checks the availability of the primary remote domain and the highest-level alternate remote domain. When possible, the gateway restores message traffic to the primary remote domain or the highest-level remote domain.

**Domains-level failover**

The transfer of message traffic to an alternate remote domain when a primary remote domain fails.

**dotted-decimal notation**

A convention for expressing IP addresses, consisting of a series of four decimal numbers (0-255), separated by periods. Example: 123.205.23.99.

**DSA**

Digital Signature Algorithm. An algorithm used to generate *digital signatures.* DSA is defined in US FIPS 186.

**DTP**

*See* **distributed transaction processing (DTP)**.

**dynamic argument**

A type of argument for which a method can allocate or extend the existing storage.

**Dynamic Data Exchange (DDE)**

A form of communication available on Microsoft Windows, Windows NT, Windows 95, and OS/2 platforms that allows applications to exchange information through a series of messages. Two applications sending and responding to DDE messages are said to have a DDE conversation.

**dynamic datatype**

A datatype whose memory size is not known when the code is compiled; a dynamic datatype’s memory size is known only when the code is executed.

**dynamic link libraries (DLL)**

A collection of functions grouped into a load module that is dynamically linked with an executable program at run time for a Windows or OS/2 application.

**electronic commerce**

The practice of buying and selling goods and services over the Internet.

**encoding**

The conversion of architecture-specific data into a format that can be transmitted between different architectures, such as XDR encoding.

**encryption**

A process used to scramble access codes to secure data from forgery or to prevent unauthorized access.

**encryption**

The process of algorithmically scrambling data to prevent (or hinder) unauthorized disclosure, while still preserving access to the original data by authorized users. To read an encrypted file, the recipient must have access to a secret key or password that enables the recipient to decrypt it. Unencrypted data is called **plaintext**; encrypted data is referred to as **ciphertext**.

**environment variable**

A string of specific value that controls a certain attribute of an application. Environment variables are made available to the application as it begins.

**event**

An indication to a BEA Tuxedo system process of the occurrence of a particular state or condition, for example, disconnection, transaction request mode, connection request, and so forth.

**Event Broker/Monitor**

A BEA Tuxedo system component that monitors the occurrence of defined system and application events and notifies subscribers when an event is detected.

**event posting**

Notification by the BEA Tuxedo system (or by an application) to the Event Broker/Monitor that a defined event has occurred.

**event subscription**

A request made to the Event Broker/Monitor to be notified when a specific event is detected.

**extensibility**

The ease with which a system adapts to meet new requirements. Extensibility also includes the ability to add or change a function or data (data type, file format, database schema, or information model) without:

- Requiring changes to existing functions, data, and interfaces
- Introducing unwanted side effects, such as degradation of performance, reliability, maintainability, portability, and so on

**External Data Representation (XDR)**

A canonical data format defined by SUN Microsystems and used to transfer data between heterogeneous hardware nodes.

**failback**

Restoration of message traffic to a higher-priority circuit. The `BRIDGE` process always tries to use the highest-priority circuit defined for the node; when traffic is flowing on a lower-priority circuit, whether due to circuit failure or just non-availability, the `BRIDGE` periodically checks higher-priority circuits to find one that is usable. When a higher priority circuit becomes available again, the data flow is returned to it. This mechanism is called “failback.”

**failover**

Seamless transfer of message traffic to a lower-priority circuit on the occasion of the failure of a higher-priority circuit.

Some operating system and hardware bundles transparently detect a problem on one network card and replace it with another. When this replacement is done quickly, application-level TCP virtual circuits have no indication that a fault has occurred.

In the BEA Tuxedo system, data flows over the highest available priority circuit. If all network groups have the same priority, data travels over all networks simultaneously. If all circuits at the current priority fail, data is sent over the next lower priority circuit. This is called “failover.”

**field manipulation language (FML)**

A set of C language functions for defining and manipulating storage structures called field buffers. Cooperating processes can send and receive data in fielded buffers.

**field table**

A file that consists of FML field names and their identifiers. The field table enables users to refer to fields by logical names rather than by system field identifiers.

**FML**

*See* **field manipulation language (FML)**.

**FML buffer**

A buffer of self-describing data items accessed through the field manipulation language API.

**foreign access path**

A physical connection between a native BEA Tuxedo system node and a foreign node. At least one gateway server must reside on the BEA Tuxedo node.

**foreign node**

A node in the network that does not have access to the configuration's Bulletin Board, or that cannot execute the full complement of BEA Tuxedo system software.

*See also* **field manipulation language (FML)**.

**format independence**

The ability to send data to a device without having to be concerned with the format in which the data is displayed. The same data may appear in different formats on different devices.

**framework**

The software environment tailored to the needs of a specific domain. Frameworks include a collection of software components that programmers use to build applications for the domain the framework addresses. Frameworks can contain specialized APIs, services, and tools, which reduce the knowledge a user or programmer needs to accomplish a specific task.

**gateway**

For the BEA Tuxedo system, any communication mechanism between different environments (for example, between native and foreign nodes). A software program that allows dissimilar systems to communicate and exchange information. A gateway normally handles communication between systems and performs all necessary protocol translation such that the end applications communicate transparently.

**gateway group**

A collection of processes that provide communication services to and from remote domains. The group consists of the following: `GWADM`, the gateway administrative server, and a gateway process, for example, `GWTDOMAIN`.

**gateway server**

A server process, resident on a native BEA Tuxedo system node, that communicates with one or more foreign machines.

**global transaction**

1) A transaction that spans one or more resource managers comprising local transactions. The Transaction Manager name for a transaction that uses multiple servers or multiple resource manager interfaces and is coordinated as an atomic unit of work. 2) The BEA Tuxedo name for a transaction that uses multiple servers or multiple resource manager interfaces and is coordinated as an atomic unit of work.

**Global Transaction Identifier (GTRID)**

A data structure, the value of which uniquely identifies a global transaction.

**graphical user interface (GUI)**

A high-level interface that uses windows and menus with graphic symbols instead of typed system commands to provide an interactive environment for a user. The BEA Administration Console, available through the World Wide Web, enables an authorized user to configure and control a BEA Tuxedo application.

**group**

A collection of servers or services on a machine, often associated with a resource manager. A group is an administrative unit used for booting, shutting down, and migrating servers and services.

**GUI**

*See* **graphical user interface (GUI)**.

**handler**

A request that originates on a remote computer. Handlers are registered in the local BEA Tuxedo Bulletin Board as client programs.

*See also* **Workstation Handler (WSH)**.

**hierarchical database**

A database organized in the form of a tree structure that predetermines the access paths to data stored in the database. DL/I, IMS, and SQL/DS are hierarchical database managers.

**hierarchy**

In a database, a tree of segments beginning with the root and proceeding downward to dependent segment types. No segment type can be dependent on more than one other segment type.

**high-level language**

A programming language.

**host**

A computer that is attached to a network and provides services other than acting as a communication switch.

**host computer**

The primary or controlling computer in a data communication system.

**hypertext markup language (HTML)**

The language used for writing pages for the World Wide Web.

**inactive server**

A server that is not currently available to process requests.

**incoming connection**

A connection to the local gateway that is initiated by a domain gateway on a remote domain.

**information hiding**

A software design technique in which a piece of code contains only the information it needs to do its job.

**infrastructure**

A common underlying computing base. The infrastructure is a set of components (fundamental services) that support another higher-level set of components in a given system. The higher-level components are typically more directly associated with providing the specific function of the overall system.

**integration**

The ability of applications to share information or to process independently by requesting services and satisfying service requests. In a well-integrated system, all of the parts have a purpose, and the parts combine effectively to achieve the purpose of the overall system.

**interaction model**

A description of how the clients and servers in a distributed application or application framework work with each other.

**interactive**

Pertaining to an application in which each entry entails a response from a system or program, as in an inquiry system or airline reservation system. An interactive system may also be conversational, implying a continuous dialog between the user and the system.

**interactive interface**

A system facility that controls how different users see and work with the system by means of user profiles. When signing on, the interactive interface makes available those parts of the system authorized by the profile. The interactive interface has sets of selection and data entry panels through which users communicate with the system.

**internationalization**

A mechanism that allows customization of a system's text messages and data formats into an application's language and format of choice.

**International Standards Organization (ISO)**

An international organization whose membership includes standards and research groups from various nations. ISO establishes standards for computer network communications and many other technologies.

**Internet**

The world's largest network, the Internet is based on the TCP/IP protocol suite and is universally accessible.

**Internet Protocol Address (IP Address)**

A numeric value that uniquely identifies a node in a TCP/IP network. IP addresses are usually expressed in "dotted decimal notation," a series of four decimal numbers (0-255), separated by periods. Example: 123.205.23.99.

**interoperability**

The ability to exchange requests between entities.

**intranet**

A set of internal company or group-specific networks protected by firewalls and connected by IP routers. An intranet appears to its users as a single network.

**IP Address**

*See* **Internet Protocol Address (IP Address)**.

**ISO**

*See* **International Standards Organization (ISO)**.

**journaling**

The recording of information in any journal (including the system log) for possible subsequent processing by the user. The primary purpose of journaling is to enable forward recovery of the data sets. A data set can be reconstructed by applying transactions in the journal against a previous version of the data set. Journaling can be used for any other user-defined purpose, such as auditing, accounting, or performance analysis.

**Kerberos protocol**

The private key authentication protocol developed as part of Project Athena at the Massachusetts Institute of Technology.

**Kerberos security**

The security system that provides authentication, mutual authentication, and protection against replay and sequencing attacks.

**keyword**

1) A symbol that identifies a parameter. 2) A part of a command operand that consists of a specific character string.

**LAN**

*See* **Local Area Network (LAN)**.

**LAN partition**

The failure of a LAN connecting the machines of an application, resulting in a loss of message communication between the machines. A partitioned site is one that no longer has access to the master node.

**lazy connection**

A connection between a domain gateway and a remote domain that is not established until the remote domain receives a request for a remote service. A lazy connection keeps initialization overhead low for configurations involving many domains.

When a domain gateway server is booted, no connections are made to any remote domains. All remote services are assumed to be available and are advertised in the BEA Tuxedo Bulletin Board. When the first request for a service in a particular remote domain is made, the gateway server receives the request and tries to establish the connection. If the connection is made, the request flows to the remote domain and the connection remains active. If the connection fails, the client receives a failure message and the service remains advertised.

**LDAP**

Lightweight Directory Access Protocol. A set of protocols for accessing information directories. LDAP is based on the standards contained within the X.500 standard, but is significantly simpler. And unlike X.500, LDAP supports TCP/IP, which is necessary for any type of Internet access. Because it is a simpler version of X.500, LDAP is sometimes called “X.500-lite.”

**legacy application**

An existing application that needs to be modified or wrapped so that it can gain access to a BEA Tuxedo domain.

*See also* **domain**.

**link-level encryption (LLE)**

The encryption of messages moving over network links. Encryption is performed just before date is transmitted on the network, and decryption is performed just after data is received.

LLE operates over Workstation client, domain gateway, bridge, and administrative network links. It employs a symmetric key encryption technique (specifically, RC4), which uses the same key for encryption and decryption.

**link-level failover in Domains**

Link-level failover is a mechanism that ensures that an alternate network link becomes active when a primary link fails.

**listener**

*See* **Workstation Listener (WSL)**.

### LMID

*See* **logical machine ID (LMID)**.

### load balancing

The practice of distributing service requests among all the servers in a given domain to achieve the most efficient handling of those requests. Specifically, the system identifies the server currently doing the smallest amount of work and sends requests to that server's queue for processing.

When a service request is routed to a domain gateway, the gateway implements two algorithms, a load-balancing algorithm and a data-dependent routing algorithm, to find the proper remote domains to which the request should be sent. Load-balancing and data-dependent routing algorithms are based on the remote service table entries and remote domain table entries in the gateway shared memory.

### local

In data communication, pertaining to devices that are accessed directly (that is, without use of a telecommunication line).

### local application names

The DDE Listener uses the application name supplied in the DDE Initiate message to determine if the client is looking for a local or a remote DDE application. The syntax for a local DDE application includes only the name of the application. For example, if the client is looking for Microsoft Excel on the local computer, the application name would be EXCEL.

### Local Area Network (LAN)

A high-speed network that spans a limited distance, such as a building or a cluster of buildings. LANs can be connected to wide area networks (WANs) with bridge devices.

### local domain

View of an application (that is, a subset of the application's services) that is available to other domains. Because a *local domain* is always represented by a *domain gateway group*, the terms are used interchangeably.

### local gateway

A specific gateway group (for example, `GWADM` and `GWTDOMAIN`) within a local BEA Tuxedo application. Multiple local gateways may be running within a single BEA Tuxedo application.

**local node**

The computer that is connected to a user’s workstation.

**local service**

A service of a local domain that is available to remote domains through a domain gateway group.

**local system**

In a multisystem environment, the system on which an application program is being executed. A local application can process data from databases located on either the same (local) system or another (remote) system.

**local transaction**

A local resource manager transaction that is active on behalf of a global transaction.

**location transparency**

The ability to define a resource so that its name implies no specific network address or physical location.

**log file**

A message file that describes events that occur during an operation. Log files are updated frequently during an operation and are useful for tracing system operations and errors.

**logical machine ID (LMID)**

The logical name assigned (in the configuration file) to a processing element used in a transaction manager application.

**makefile**

A file, referenced by the make command, that tells the make command how to create each of the files needed to generate a complete program. The makefile contains a list of source files, object files, and dependency information.

**managed object**

An entity (such as a process, a piece of hardware, or system performance) that is defined in the MIB and is controlled by a management device.

**Management Information Base (MIB)**

A BEA Tuxedo system component that provides a complete definition of the classes and their attributes that make up the BEA Tuxedo system. The BEA Tuxedo System Management Information Base comprises a generic MIB and a specific MIB for each major component, such as Domains and Workstation. Configuration and administration of the BEA Tuxedo system can be done programmatically by using the ATMI to set or change the value of an attribute.

**mapping**

The process of associating local values or entities with values or entities that are meaningful on remote systems.

**master node**

The `MASTER` node for an application as designated in the `RESOURCES` section of the configuration file. It contains the master copy of the `TUXCONFIG` binary configuration file. Administration of the running system is done from the `MASTER` node.

**MD5**

Message Digest 5. An algorithm defined in RFC 1321 that takes as input a message of arbitrary length and produces as output a 128-bit *message digest*, or *hash value*, of the input. The MD5 algorithm is intended for digital signature applications, where a large file must be compressed in a secure manner before being encrypted with a private key under a public-key cryptosystem such as PKCS.

**message**

Data that contains information and/or instructions for the recipient.

*See also* **request**.

**message catalog**

With respect to internationalization, a file or storage area containing program messages, command prompts, and responses to prompts for a particular native language, territory, and codeset.

**message digest**

The representation of text in the form of a single string of digits, created with a formula called a *one-way hash function*. Encrypting a message digest with a private key creates a digital signature, which is an electronic means of **authentication**.

**message digest algorithm**

A method of reducing a message of any length to a string of a fixed length, called the *message digest* or *hash value*. Message digest algorithms have the property that it is computationally infeasible to find a message that corresponds to a given message digest, or to find two different messages that produce the same message digest. Examples of message digest algorithms are MD5 and SHA-1.

**MIB**

*See* **Management Information Base (MIB)**.

**middleware**

A set of services for building distributed client/server applications, such as services for locating other programs in the network, establishing communication with those programs, and passing information between applications. Middleware services can also be used to resolve disparities between different computing platforms and to provide a uniform authorization model in networks that comprise multiple vendors and multiple operating systems.

**migrate**

Relocate a server or group of servers from one LMID to another. Migration must be planned for and specified in the configuration file.

**model**

A paradigm in which details have been abstracted for the sake of simplification.

**modeling**

A design technique used in developing architecture, simulations, and computer systems.

**module**

A piece of code that contains information on a particular topic and on the topic's related interfaces. For example, code that describes a banking withdrawal operation could be stored in a module.

**MP model**

A BEA Tuxedo application configuration that runs on more than one computer. The multiple machines that make up an MP configuration might include two or more uniprocessors, one or more multiprocessors, or a combination of uniprocessors and multiprocessors.

**multicontexted process**

A BEA Tuxedo process that is connected to more than one application and/or has more than one connection to the same application.

**multi-domain client**

A BEA Tuxedo client that is associated with more than one BEA Tuxedo applications.

**multiple listening addresses**

When addresses are available on separate networks, even if one virtual circuit is disrupted, another circuit can continue undisturbed. Only a failure on all configured networks makes reconnection of the `BRIDGE` processes impossible. For example, when a high-priority network fails, its load can be switched to an alternate network with a lower priority. When the higher-priority network returns to service, the network load returns to it.

**Multiple Virtual Storage (MVS)**

One of IBM’s principal mainframe operating systems. MVS/XA: Extended Architecture. MVS/ESA: Enterprise Systems Architecture.

**multiprocessor**

A computer that has more than one processing element, each with its own private memory.

**multiprogramming**

The concurrent execution of two or more computer programs.

**multithreading**

Use of a process by several transactions.

**mutual authentication**

The process by which peers authenticate themselves to each other. In normal authentication, a server accepts a client’s proof of identity, but does not authenticate itself to the client. In mutual authentication, the client proves its identity to the server and demands that the server prove its identity to the client before communication proceeds.

**MVS**

*See* **Multiple Virtual Storage (MVS)**.

**MVS console**

A terminal used by the MVS operating system and other software components to display informational and error messages for the system operator. MVS consoles can also be used by the system operator to enter commands or to inquire about and modify system status.

**name server**

A software component of the BEA Tuxedo system that transparently maps service names to physical addresses so that users can communicate with services by name rather than by internal identifier.

**native client**

*See* **client**.

**native node**

A machine in a BEA Tuxedo configuration that contains the full complement of BEA Tuxedo software, and that has access to the same Bulletin Board as all other native nodes in the configuration (that is, it is part of the administrative domain of the application).

**network**

1) An interconnected group of nodes. 2) The assembly of equipment through which connections are made between data stations. 3) The communication path used to communicate with a server.

**network address**

A unique identifier assigned to each network connection made. This identifier may either be assigned randomly by the provider or created in response to a request for a specific identifier from the process establishing the connection. The format of the network address is provider-specific.

**Network Agent (NA)**

BEA TOP END term that is equivalent to the BEA Tuxedo term **Workstation Handler (WSH)**.

**Network Interface (NI)**

BEA TOP END term that is equivalent to the BEA Tuxedo term **BRIDGE**.

**Node Manager (NM)**

BEA TOP END term that is equivalent to the BEA Tuxedo term **Bulletin Board Liaison (BBL)**.

**network provider**

The protocol used at the transport level and below to communicate data across a network. Network providers are typically accessed from programs through a transport interface. Examples of network providers are TCP/IP and StarLAN.

**NLS**

The UNIX system network listener service.

**node**

A point on a network. The term is also used to refer to a computer (for example, a single instantiation of the UNIX operating system) that participates in a BEA Tuxedo system application. It is possible, however, to have more than one node in a multiprocessor system.

**nonblocking mode**

An asynchronous style of message delivery, that does not require a program to wait for an action to complete before proceeding.

**nonmaster node**

Any node of a BEA Tuxedo application that is not designated the `MASTER` node.

**nonpartitioned**

A term used to identify portions of a partitioned network that continue to be able to communicate with the DBBL on the master node.

***n*-tier client/server**

An application development approach that distributes application logic across three or more environments: the desktop computer, one or more application servers, and a database server. The main advantage of the *n*-tier client/server model is that it extends the benefits of client/server architecture to the enterprise level. Other advantages include added manageability, scalability, security, and higher performance.

**octet**

1) A byte that consists of eight bits. 2) A byte composed of eight binary elements.

**OLTP**

*See* **online transaction processing (OLTP)**.

**online transaction processing (OLTP)**

1) A form of data processing in which users at terminals or workstations send messages to application programs, which update databases in real time. 2) The execution of units of work in an environment that appears to the user as immediate; real-time; usually having internal recoverability, history-keeping, and consistency-assurance features.

**open framework**

A software infrastructure in which developers can easily remove and replace software components in a “plug-and-play” fashion.

**open system**

A system that implements specified common standards across different computer vendors. Implementing open system standards for communication allows computers from different vendors to communicate with each other.

**Open Systems Interconnect Commitment, Concurrency, and Recovery (OSI CCR)**

1) This protocol is an ISO standard (ISOIIEC 9804) for services and protocols that are used to commit or roll back global transaction branches. 2) A software implementation of the ISOIIEC 9804 standard.

**Open Systems Interconnection (OSI)**

A consortium that facilitates communication among different types of computer systems.

**Open Systems Interconnect Transaction Processing (OSI TP)**

1) An ISO standard (ISOIIEC 10026-2) for services and protocols that are used to establish dialogs and pass messages between clients and service routines on different computers. 2) A software implementation of the ISOIIEC 10026-2 standard.

**OSI CCR**

*See* **Open Systems Interconnect Commitment, Concurrency, and Recovery (OSI CCR)**.

**OSI TP**

*See* **Open Systems Interconnect Transaction Processing (OSI TP)**.

**outgoing connection**

A connection from the local gateway that has been generated as a result of one of the following: an automatic retry of the connection, an initial request to a remote domain, or a `dmadmin`(1) `connect` command sequence issued by the administrator.

**out-of-band data**

Data delivered by the BEA Tuxedo system outside the normal client/server communications channels supported by the BEA Tuxedo system.

**parallel data circuits**

Parallel data circuits enable data to flow simultaneously on more than one circuit. When you configure parallel data circuits, network traffic is scheduled over the circuit with the largest network group number (`NETGRPNO`). When this circuit is busy, the traffic is scheduled automatically over the circuit with the next lower network group number. When all circuits are busy, data is queued until a circuit is available.

**partition**

A state in which one or more active nodes of a networked application are unable to contact other active nodes because of a problem such as a LAN failure.

**PID**

*See* **process ID (PID)**.

**PKCS-7**

*See* **Public-Key Cryptography Standard 7 (PKCS-7)**.

**plaintext**

In cryptography, text that is not encrypted.

**platform**

The combination of hardware, operating system, and windowing system software that supports an application.

**portability**

The ease with which developers can move an application from one platform to another without costly reengineering.

**port number**

The entity on a TCP/IP host that identifies a logical communications channel and distinguishes one connection from another. A TCP/IP server “listens” for incoming connection requests at a designated port. A TCP/IP client initiates a connection with the server by specifying the host’s IP address and the server’s designated port number.

**precommit**

The process (used by resource managers to make data recoverable) of copying data changes to stable storage.

**primary remote domain**

The remote domain with the highest priority. It is used whenever it is available.

**principal**

In security, an authenticated user.

**private**

A field that is not visible outside its class definition.

**private MIB**

A MIB that is defined under a private MIB directory.

**procedure**

A sequence of instructions (to a computer) for performing a particular task.

**process ID (PID)**

A unique number that identifies a process.

**profile**

A set of information about a client or a user. The profile provides information the server might require to recognize the client or the user.

**protocol**

1) A set of rules that govern the format and timing of messages sent and received over a communication link. TCP/IP is an example of a network protocol. 2) The set of "rules" followed by two systems to communicate and exchange information.

**provider**

The communications product supplying networking facilities through level 4 of the OSI communications protocol.

**public key algorithm**

An algorithm for encrypting or decrypting data with a public or private key. A private key is typically used to encrypt a message digest; in such an application, the public key algorithm is called a *message digest encryption algorithm*. A public key is typically used to encrypt a content-encryption key, or session key; in such an application, the public key algorithm is called a *key-encryption algorithm*. An example of a public key algorithm is RSA.

**Public-Key Cryptography Standard 7 (PKCS-7)**

One of a set of Public-Key Cryptography Standards developed by RSA Laboratories in cooperation with an informal consortium, originally including Apple, Microsoft, DEC, Lotus, Sun and MIT. PKCS-7 defines a general syntax for messages that include cryptographic enhancements such as digital signatures and encryption. BEA Tuxedo public key security complies with the PKCS-7 standard.

**public key encryption**

A technique that uses a pair of asymmetric keys for encryption and decryption. Each pair of keys consists of a public key and a private key. The public key is made public by distributing it widely. The private key is never distributed; it is always kept secret.

**public key security**

BEA Tuxedo security built on the public key encryption technology. Uses public key encryption to establish end-to-end digital signing and data privacy between BEA Tuxedo application clients and servers. Complies with the PKCS-7 standard.

**queue**

A simple data structure for managing the time-staged delivery of requests to servers. Queued elements may be sorted in some order of priority. Clients insert items in the queue and servers remove items from the queue, as soon as possible, in batch, or periodically.

**RC2**

Rivest's Cipher 2. A variable key-size block cipher with a key size range of 40 to 128 bits. It is faster than DES and is exportable with a key size of 40 bits. A 56-bit key size is allowed for foreign subsidiaries and overseas offices of United States companies. In the United States, RC2 can be used with keys of virtually unlimited length, although BEA Tuxedo public key security restricts the key length to 128 bits.

**RC4**

Rivest's Cipher 4. A variable key-size stream cipher with byte-oriented operations. RC4 is a symmetric or private key system that is about 10 times faster than DES and is exportable with a key size of 56 bits. In the United States, RC4 can be used with keys of virtually unlimited length, although BEA Tuxedo link-level encryption restricts the key length to 128 bits.

**record**

Input or output data as it exists outside the local or remote BEA Tuxedo regions, or on different kinds of systems.

**recover**

A request from a coordinator or a participant to complete an identified transaction.

**recovery**

In transaction systems, after a failure, the ability to restore the system to the most recently committed, and therefore consistent, state. In distributed systems, recovery may involve resynchronizing several distributed components. Once a system has been recovered, processing can resume, and transactions aborted as a result of the failure can be resubmitted.

**relational database**

A database accessed and organized according to the relationships between data items. Relationships are expressed by means of tables that allow the accessing of items with matching attributes. The access path is determined at the time of access.

**reliability**

The extent to which a system (or part of a system) produces the correct output on repeated trials (without unintended side effects), while meeting the performance specification.

**remote**

Describes a service or a computer that is available to a client over the network.

**remote client**

*See* **client**.

**remote domain**

View of an application that is accessed through a local domain gateway group.

**remote file sharing (RFS)**

A UNIX system capability that provides access to remote files over a network.

**remote gateway**

The functionality of a specific gateway group within a remote BEA Tuxedo application.

**remote node**

Any computer in the network other than the computer to which the user’s workstation is connected.

**remote procedure call (RPC)**

A local procedure call that is executed in a non-local program or address space. Enables application logic to be split between a client and a server in the way that best uses available resources.

**remote service**

A service of a remote domain that is made available to the local application through a domain gateway group.

**remote service fan-out**

A configuration in which a remote service that is imported from multiple remote domains is advertised only once by a local domain gateway in the BEA Tuxedo Bulletin Board. A remote service can be fanned-out by a gateway server that imports the same service name from multiple remote domains. Fan-out is achieved through the gateway shared memory.

**remote service name**

The name (between 1 and 16 characters long) of a service offered by a remote system that can be accessed through a remote gateway.

**request**

A message sent by a client that identifies an operation to be performed.

*See also* **client** and **server**.

**request/response communication**

Communication characterized by a single request matched to a single response message. When this type of communication is used, a client requests a task, and a server performs the task and sends a response to the client. Request/response communication may be conducted synchronously or asynchronously.

**request/response server**

A server that offers request/response services.

**request/response service**

A service initiated by a request from a client. The service routine receives a single request and provides (at most) a single reply. A request/response service is handled like a procedure and has the following properties: it is executed until completion, it does not have any dialog with the requestor, and it sends back a return-value to the requester. For a requester, the execution of a request/response service can be synchronous or asynchronous.

**requester**

A generic term for a client or server that is acting as a client.

**requestor**

A process that receives messages from clients, converts these messages to a common internal form, determines the appropriate server or servers for the transaction request, and forwards the request to a server.

**resource manager (RM)**

An interface and associated software that provides access to a collection of information and processes; for example, a database management system. Resource managers provide transaction capabilities and permanence of actions; they are the entities accessed and controlled within a global transaction.

*See also* **transaction manager (TM)**.

**resource manager instance**

A particular instance or occurrence of a resource manager. There may be many occurrences or instances of the same or different resource managers within a global transaction, each managing different data. Each resource manager instance is considered to be autonomous, in full control of local access (for both local and global transactions), administration, and so forth.

**response time**

The elapsed time between entering an inquiry or request and receiving a response.

**RM**

*See* **resource manager (RM)**.

**roll back**

To terminate a transaction in such a way that all resources updated within a transaction revert to their original (that is, pretransactional) state.

**rollback**

The event that ends a transaction and nullifies or undoes all changes to resources that were specified during that transaction.

**RPC**

*See* **remote procedure call (RPC)**.

**RSA**

A public key algorithm invented by Rivest, Shamir, and Adleman. RSA is widely employed for digital signatures and encryption. The RSA algorithm is asymmetric: a private key is closely held by its owner, while a corresponding public key may be widely disseminated.

**RTQ**

Component of the BEA TOP END system that is equivalent to the /Q component of the BEA Tuxedo system.

**run-time trace**

A BEA Tuxedo feature that enables users to monitor application-to-application transactions and, if necessary, troubleshoot distributed applications under development or during production. Also, it allows users to pinpoint problems to any hardware, operating system, network, or application code.

**scalability**

The extent to which developers can apply a solution to problems of different sizes. Ideally, a solution should work well across the entire range of complexity. In practice, however, there are usually simpler solutions for problems of lower complexity.

**scope**

To use a class to enforce a particular use for an application.

**security**

The protection of information from unauthorized modification or disclosure and the protection of resources from unauthorized use.

**security principal**

An entity that is known to, and can be authenticated by, the security system.

**server**

1) A process that receives service requests from a requester and dispatches the service routine that acts on the request. 2) A software module that accepts requests from clients and other servers and returns responses. 3) A software program that provides a service in a client/server architecture.

*See also* **client**.

**server abstraction**

Applications combine their service routines with the BEA Tuxedo system `main()` in building a server process. The BEA Tuxedo system's `main()` provides server initialization and termination, and receives incoming requests and dispatches them to service routines. All of this processing is transparent to applications.

**server group**

*See* **group**.

**server ID**

An identifier for a single server. No two servers can operate at the same time with the same server ID.

**service**

1) An application routine available for request by a client in the system. 2) A module of application code that carries out a service request.

**service code**

The name associated with a service offered remotely through a tlisten process.

**service request**

A request initiated by a requester process that asks for the invocation of a service.

**service routine**

An application module that performs one or more specific services on behalf of clients. The structure of service routines (the mechanism by which they are called and terminated) is defined by the XATMI interface specification.

**SERVICES section**

The section of the configuration file in which services are defined.

**session key**

Used with the symmetric key algorithm in which encryption and decryption involve the same key: the session key. Data encrypted with a session key can only be decrypted with the same session key.

**SHA-1**

Secure Hash Algorithm 1. An algorithm specified in the Secure Hash Standard that takes as input a message of any length greater than 264 bits and produces as output a 160-bit *message digest,* or *hash value,* of the input. It is slightly slower than MD5, but the larger message digest makes it more secure against brute-force collision and inversion attacks. The SHA-1 algorithm is intended for digital signature applications, where a large file must be compressed in a secure manner before being encrypted with a private key under a public-key cryptosystem such as PKCS.

**SHM model**

A BEA Tuxedo application that runs entirely on a single computer, even if that computer is a symmetric multiprocessor.

**single threading**

The complete execution of a program. Processing of one transaction is completed before another transaction is started.

**socket**

An endpoint of communication to which a name may be bound. The socket interface is a network access method supported by the BEA Tuxedo system. The logical end point of a TCP/IP connection. An application accesses a TCP/IP connection through a socket.

**socket descriptor**

A TCP/IP-assigned number that uniquely identifies a socket and TCP/IP connection. An application must specify the socket descriptor in TCP/IP API calls to identify the socket/connection.

**socket ID**

*See* **socket descriptor**.

**socket number**

*See* **socket descriptor**.

**SQL**

*See* **Structured Query Language (SQL)**.

**standard MIB**

A MIB developed as a standard by the Internet community. Examples of the standard MIBs are MIB I and MIB II.

**StarLAN**

An AT&T LAN product.

**state**

The situation of a conversation from the point of view of one of the participating transactions. The conversation state determines the commands that a transaction can validly issue. The state of each transaction changes dynamically in the course of conversation.

**stateful application**

An application that retains state information in memory after a service or an operation has been performed.

**stateless application**

An application that flushes state information from memory after a service or an operation has been performed.

### STRING buffer

A data structure that is an array of non-null characters terminated by the null character. It is a self-describing buffer.

### Structured Query Language (SQL)

A non-procedural language for defining and accessing relational databases. SQL has become the industry standard database language.

### subroutine

A sequenced set of instructions that can be used in one or more programs and at one or more points in a program. The execution of a subroutine is usually invoked by a call.

### subscriber

An application program that subscribes to an event or set of events, and declares what action should take place when an event is posted.

### symmetric key algorithm

An algorithm for encrypting or decrypting data with the same key, known as a *session key*. A random number generator creates a new session key for each communication, which makes it difficult for a would-be attacker to reuse previous communications.

Examples of symmetric key algorithms are DES, RC2, and RC4.

### synchronization

A coordinated commitment control process between communicating transactions that ensures that all logically related updates to recoverable resources are completed or that all are backed out.

### synchronous

1) Pertaining to an event that happens, exists, or arises at precisely the same time as another event. 2) Pertaining to an operation that occurs regularly or predictably with regard to the occurrence of a specified event in another process; for example the calling of an input/output routine that receives control at a precoded location in a program.

**synchronous communication**

A method of transmitting data using a timing signal. With this form of communication, when one piece of software sends a message to another piece of software, the originating software must wait until the service provider completes the request before it can continue.

**synchronous process**

A process that cannot be executed independently of other processes. When a request is processed synchronously, the client must wait until the service provider completes the request before continuing.

**system**

BEA TOP END term that is equivalent to **BEA Tuxedo domain**.

**system administration**

1) The preparation of an instance of a system for use in a particular setting or installation. 2) Changing the system as the installation changes. This term is often used in a way that includes system operations as well.

**system administrator**

*See* **administrator**.

**system operations**

Tasks that need to be performed regularly for a system. Examples are backing up and restoring data and logs, monitoring the system for error conditions, and so forth.

**task**

One execution of a transaction.

**TCP/IP**

*See* **Transmission Control Protocol/Internet Protocol (TCP/IP)**.

**TDomain gateway**

The domain gateway that handles communication between two BEA Tuxedo domains.

*See also* **domain gateway**.

**terminal**

1) A computer monitor. 2) A point in a system or communication network at which data can either enter or leave.

**threading**

An operating system capability that allows a single process to be divided into multiple entities, each of which executes independently while sharing a common address space.

**three-tier client/server architecture**

An implementation of *n*-tier client/server architecture.

**TLI**

*See* **Transport Layer Interface (TLI)**.

**tlisten**

A network-independent listener process that runs as a daemon process and provides remote service connections for other BEA Tuxedo system processes.

**TLOG**

*See* **transaction log (TLOG)**.

**TM**

*See* **transaction manager (TM)**.

**TMS**

*See* **Transaction Manager Server (TMS)**.

**TP**

*See* **transaction processing (TP)**.

**TP monitor**

*See* **transaction processing monitor (TP monitor)**.

**TP protocols**

*See* **transaction processing protocols (TP protocols)**.

**TPSUT**

*See* **transaction processing service user title (TPSUT)**.

**transaction**

1) A complete unit of work that transforms a database from one consistent state to another. In DTP, a transaction can include multiple units of work performed on one or more systems. 2) A logical construct through which applications perform work on shared resources (for example, databases). The work done on behalf of the transaction conforms to the four ACID properties: atomicity, consistency, isolation, and durability.

*See also* **distributed transaction processing (DTP)**.

**transaction coordinator**

A system software component that provides the infrastructure that guarantees the integrity and consistency of an operation and the data involved in a transaction.

*See* **transaction manager (TM)**.

**transaction log (TLOG)**

The BEA Tuxedo system log that keeps track of global transactions.

**transaction manager (TM)**

A system software component that manages global transactions on behalf of application programs. A transaction manager coordinates commands from application programs and communication resource managers to start and complete global transactions by communicating with all resource managers that are participating in those transactions. When resource managers fail during global transactions, transaction managers help resource managers decide whether to commit or roll back pending global transactions.

*See also* **transaction coordinator**.

**Transaction Manager Server (TMS)**

A BEA Tuxedo system server process that manages the 2-phase commit protocol and recovery for global transactions.

**transactions per second (TPS)**

A throughput rating used in conjunction with the standard transactions defined by TPC. Often the rating implies the maximum number of transactions that can be processed by the system in which 90% of the transaction’s response times are less than two seconds.

**transaction processing (TP)**

Transaction processing consists of a set of convenient functions that make it easy for you to write code that does the following: 1) Initialize the server application and execute startup and shutdown routines. 2) Tie the server application to BEA Tuxedo domain resources. 3) Perform housekeeping functions.

**transaction processing monitor (TP monitor)**

A class of products that provide a transaction execution environment on top of conventional operating systems.

**transaction processing protocols (TP protocols)**

A set of standard protocols by which transaction processing managers on heterogeneous systems interoperate.

**transaction processing service user title (TPSUT)**

A value that is used to identify OSI TP endpoints that exist within one application entity.

**translation**

The process of changing how intrinsic data types are represented in input data and output data (with respect to word length, byte ordering, and character encoding).

**Transmission Control Protocol/Internet Protocol (TCP/IP)**

The suite of protocols that defines the Internet. Originally designed for the UNIX operating system, TCP/IP software is now available for every major operating system.

**transport interface**

The programming interface used to access a network provider. Transport interfaces are typically network provider independent to an extent.

**Transport Layer Interface (TLI)**

The standard UNIX system user-level interface to data communications features as defined by level 4 of the OSI communications protocol. It is a network access method supported by the BEA Tuxedo system.

**transport provider**

*See* **network provider**.

### TUXCONFIG file

The binary version of the configuration file for a BEA Tuxedo application. This file is accessed by all BEA Tuxedo processes for all configuration information.

*See also* **application**, **client**, **server**, **UBBCONFIG file**.

### Tuxedo

*See* **BEA Tuxedo system**.

### Tuxedo domain

*See* **domain**.

### two-phase commit (2PC)

A method of coordinating a single transaction across more than one DBMS (or other resource manager). It guarantees data integrity by ensuring that transactional updates are committed in all of the participating databases, or are fully rolled back out of all the databases, reverting to the state prior to the start of the transaction.

### two-tier client/server

An application development approach that splits an application into two parts and divides the processing between a desktop workstation and a server machine.

### TX interface

The Transaction Demarcation (TX) API used by application programs to call the transaction manager. Application programs use the TX interface to define the boundaries of global transactions and direct the completion of those transactions.

### type conversion

The process of converting an application program's data buffer or record so that the data is formatted in a manner that is suitable to a target application program.

### typed buffer

A buffer for message communication involving data of a specific type.

*See* **buffer types**.

### UBBCONFIG file

The ASCII version of the configuration file for a BEA Tuxedo application. This is the file from which the `TUXCONFIG` file is generated.

*See also* **application**, **client**, **server**, **TUXCONFIG file**.

**unadvertised**

A service is unadvertised when there is no service table entry for it in the BEA Tuxedo Bulletin Board.

**uniprocessor**

A computer that has only one CPU.

*See also* **multiprocessor**.

**Universal Device List (UDL)**

A system-wide list of devices (either raw disk slices or UNIX files) on which space is allocated for `TUXCONFIG` configuration tables, BEA Tuxedo transaction logs, and, possibly, databases. Its location is specified by the `TUXCONFIG` environment variable.

**URL**

Uniform Resource Locator. The address of a site on the Web. An example of a URL is `www.bea.com`.

**user datagram protocol (UDP)**

The TCP/IP datagram transport layer protocol.

**view**

In the VIEW System Manager, a window that is displayed when you click on an icon that represents a managed host.

**VIEW buffer**

A data structure similar to a C structure. As part of defining this buffer type, a view description file is created. It is a self-describing buffer. VIEW buffers are always accompanied by VIEW definitions.

**VIEW definitions**

Descriptions of data structures that are used for input and output in the BEA Tuxedo environment.

**virtual machine**

The functional equivalent of a computer and its associated devices that is controlled by a user at a terminal.

**virtual one-hop network**

A network in which all nodes can be reached from all other nodes in exactly one transmission. This implies nothing about the physical configuration of the network, which may be a ring, star, bus, or any other valid configuration as long as it appears that all nodes are fully interconnected.

**Volume Table of Contents (VTOC)**

A file that contains the BEA Tuxedo system and possibly database tables.

**VTOC**

*See* **Volume Table of Contents (VTOC)**.

**WAN**

*See* **Wide Area Network (WAN)**.

**Web GUI**

A graphical user interface for the administration of BEA Tuxedo applications, that is accessed through a Web browser. This interface is called the BEA Administration Console.

**Wide Area Network (WAN)**

A public or private data communications system in which data is transmitted primarily over communication lines.

**window**

An area of a user’s screen in a graphical user interface system. A window is a mechanism used by applications for interacting with a user.

**Workstation**

The component of the BEA Tuxedo system that allows application clients to be located on sites on which no server-side components of the BEA Tuxedo system are installed. A Workstation client does not support an administration server, an application server, or a Bulletin Board. All communication between such a client and the rest of the application takes place over a network.

**Workstation client (WSC)**

A client process that runs on a machine on which no BEA Tuxedo server software is installed. Multiple Workstation clients can run simultaneously on Windows, Windows NT, UNIX, and VMS platforms. The ATMI is available to Workstation clients.

**Workstation Handler (WSH)**

A process that manages one or more connections between Workstation clients and native BEA Tuxedo servers. Specifically, a Workstation Handler makes surrogate service requests, manages transactions, and returns replies. WSH processes are started and stopped by a Workstation Listener (WSL). The WSH process resides within the administrative domain of the application. Handlers are registered in the local BEA Tuxedo Bulletin Board as clients.

**Workstation Listener (WSL)**

A process that assigns Workstation Handlers (WSHs) to Workstation clients (WSCs). Once such an assignment is made, the designated WSH manages all service requests from the specified client. The WSL also manages the pool of Workstation Handlers, starting and stopping them in response to load requirements. The WSL process resides within the administrative domain of the application.

**wrap**

To enclose an application in a software layer to make the application available to other applications.

**XA**

The interface between the transaction manager and resource manager.

**XATMI application service element**

Software that maps primitives in the XATMI interface to the OSI TP protocol.

**XATMI interface**

An interface that enables application programs to use request/response communication and conversational communication during global transactions.

**X_C_TYPE buffer**

A non-nested C structure whose elements are any of the following C data types: int, short, long, char, float, double, character string, and octet array. X_C_TYPE is one of three buffers that are defined in the X/Open XATMI standard. It is equivalent to the BEA Tuxedo VIEW buffer.

### X_COMMON buffer

A non-nested C structure whose elements are any of the following C data types: short, long, or char. X_COMMON is one of three buffers that are defined in the X/Open XATMI standard. It is equivalent to the BEA Tuxedo VIEW buffer; however, X_COMMON represents only the subset of field types that are common to both the C and COBOL languages.

### XDR

*See* **External Data Representation (XDR)**.

### X_OCTET buffer

An array of bytes whose structure is defined by an application. X_OCTET is one of three buffers that are defined in the X/Open XATMI standard. It is equivalent to the BEA Tuxedo CARRAY buffer.

### X/Open

The X/Open Company, Ltd., an international private consortium of vendors and users working to establish standards for open systems. BEA Tuxedo products are designed to implement X/Open standards for distributed transaction processing.

### X/Open Distributed Transaction Processing (DTP) model

The distributed transaction processing model specified in standards developed by the X/Open Company, Ltd. The BEA Tuxedo architecture is based on these standards. The model defines four components of a DTP system: Application programs (APs) define the transaction boundaries and perform the actions that make up the transaction (typically database updates).

Resource managers (RMs), such as database management systems, provide access to shared resources. Communication resource managers (CRMs) allow application programs to communicate with each other. Transaction managers (TMs) assign unique identifiers (XIDs) to transactions, monitor the progress of transactions, and handle transaction completion or recovery.