BEA Tuxedo®

Interoperability

Version 10.0
Document Released: September 28, 2007

Contents

Interoperability and Coexistence

The following sections describe how BEA Tuxedo 10.0 interoperates with older releases of the BEA Tuxedo software, BEA WebLogic Enterprise, and third-party products:

Interoperability Defined

Interoperability, as defined in this discussion, is the ability of the current release of BEA Tuxedo software to communicate over a network connection with BEA Tuxedo release 9.0 or earlier software *or* with BEA WebLogic Enterprise release 5.1 software. In addition, *intradomain interoperability* and *interdomain interoperability* have the following meanings:

- Intradomain interoperability

  Involves one machine in a multiple-machine BEA Tuxedo domain (application) running BEA Tuxedo release 10.0 software, and another machine in the same domain running BEA

Tuxedo 9.1 or earlier software *or* BEA WebLogic Enterprise 5.1 software. Machines in a multiple-machine domain configuration communicate via Tuxedo Bridge processes.

  – In a multiple-machine Tuxedo domain running Tuxedo 9.1 or earlier system software, the *master* machine (and *master backup* machine if so configured) must run the highest release of the Tuxedo system software in the domain. Accordingly, the Tuxedo domain just described qualifies as a "BEA Tuxedo domain running BEA Tuxedo release 10.0 software."

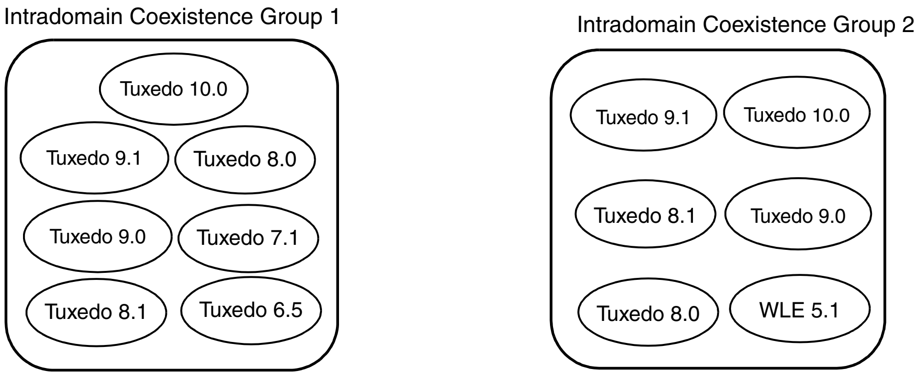
- Interdomain interoperability

  Involves one BEA Tuxedo domain running BEA Tuxedo release 10.0 software, and another domain running BEA Tuxedo release 9.1 or earlier software *or* BEA WebLogic Enterprise release 5.1 software. Domains involved in a multiple-domain (Domains) configuration communicate via Tuxedo domain gateway processes.

Intradomain Interoperability

Message exchange and protocol compatibility exist in each of the following two *intradomain* groups:

Figure 1-1 Intradomain Groups



BEA Tuxedo 10.0 can coexist in the same domain with Tuxedo 9.1, 9.0, 8.1, 8.0, 7.1, and 6.5. BEA Tuxedo 10.0 can also coexist in the same domain with Tuxedo 9.1, 9.0, 8.1 and WebLogic Enterprise 5.1.

In both of these environments, the propagation of transaction context (transactional state information) and security context (user identity) between application clients and servers is fully supported. Also, administration is fully supported in both of these environments.

Interdomain Interoperability

Message exchange and protocol compatibility exist in each of the following three *interdomain* scenarios:

Figure 1-2 Interdomain Scenario 1

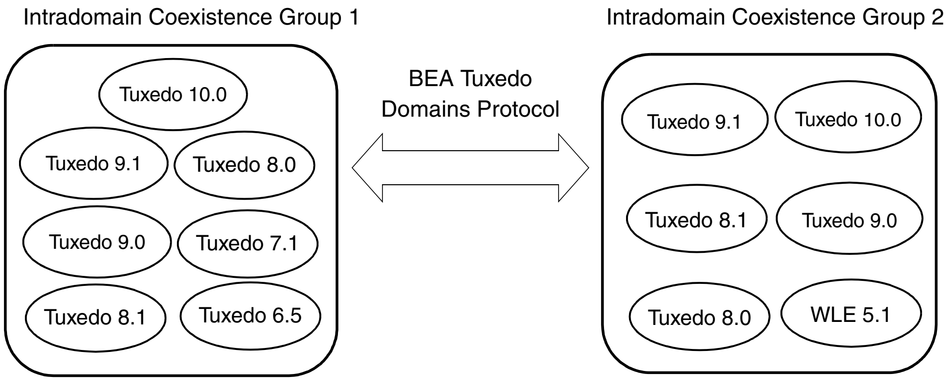


Figure 1-3 Interdomain Scenario 2

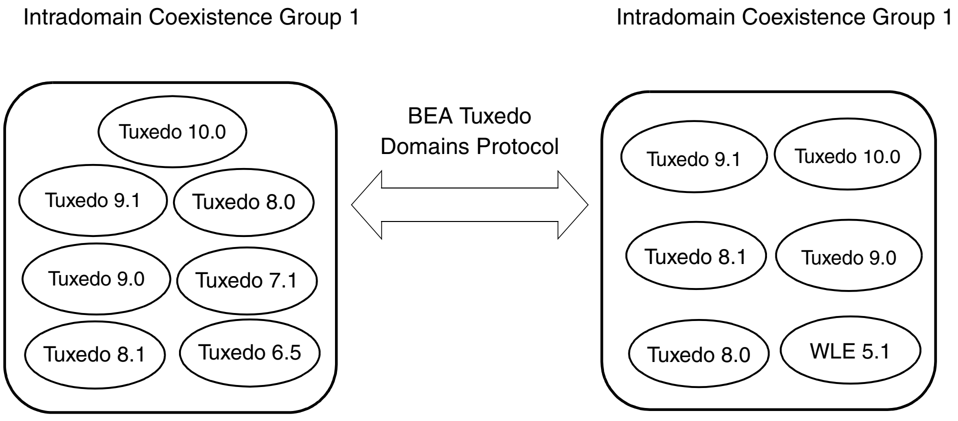
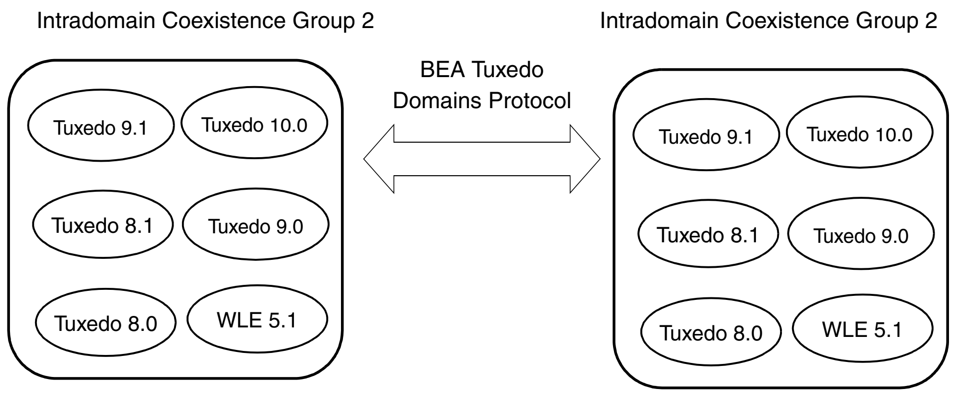


Figure 1-4 Interdomain Scenario 3



In each of these scenarios, a Tuxedo domain (TDomain) gateway process running on a machine in the one domain communicates over a network connection with a TDomain gateway process running on a machine in the other domain. The following pairs of communicating TDomain gateway processes are supported.

Table 1-1 Communicating TDomain Gateway Processes

A TDomain process in any of these releases . . .		Can communicate with a TDomain process in any of these releases . . .
BEA Tuxedo 10. 0	Connection Matrix	BEA Tuxedo 10.0
BEA Tuxedo 9.1		BEA Tuxedo 9.1
BEA Tuxedo 9.0		BEA Tuxedo 9.0
BEA Tuxedo 8.1		BEA Tuxedo 8.1
BEA Tuxedo 8.0		BEA Tuxedo 8.0
BEA Tuxedo 7.1		BEA Tuxedo 7.1
BEA Tuxedo 6.5		BEA Tuxedo 6.5
BEA WebLogic Enterprise 5.1		BEA WebLogic Enterprise 5.1

Of course, the interdomain capabilities available through a pair of communicating TDomain processes are limited to the capabilities available to the TDomain process running in the earlier release of Tuxedo or WebLogic Enterprise software.

In all of these scenarios, administration, transaction context propagation, and security context propagation between domains is fully supported *except* when the master machine in the one domain is running Tuxedo 7.1 or 6.5 software and the master machine in the other domain is running WebLogic Enterprise 5.1 software. In any of these exception cases, administration is supported and transaction context propagation is supported, but security context propagation is not supported.

BEA Tuxedo 10.0 supports interdomain interoperability with Tuxedo 9.1, 9.0, 8.1 CORBA domains and with WebLogic Enterprise 5.1 CORBA domains. This capability includes the ability to advertise CORBA C++ factories across domain boundaries.

For interdomain transactional requests between Tuxedo 6.5 and other Tuxedo releases (6.5, 7.1, 8.0, 8.1, 9.0, 9.1, and 10.0), the following patch level upgrades must be applied:

- to at least patch level 446 for Tuxedo 6.5
- to at least patch level 291 for Tuxedo 7.1
- to at least patch level 261 for Tuxedo 8.0
- to at least patch level 090 for Tuxedo 8.1
- 9.0
- Tuxedo 9.1 General Available version is acceptable
- Tuxedo 10.0 General Available version is acceptable

This introduced a requirement for simultaneous patch upgrade for all interoperating domains involved. However, this simultaneous upgrade requirement might cause practical problems for applications that do staggered upgrades of patches. To mitigate such practical problems, `TM_GWT_OLDSECCHECK` is introduced to `GWTDOMAIN` and needs to be used until all interoperating domains are at least at the patch levels mentioned previously. Once all interoperating domains are upgraded, this environment variable can be eliminated from all domains.

Client-Server Interoperability

To support customer migration, the following client-server interoperability is supported for BEA Tuxedo 10.0.

Table 1-2 Client-Server Interoperability

This component . . .	Can interoperate with . . .
BEA Tuxedo 10.0 ATMI server	• ATMI clients running in Tuxedo 6.5, 7.1, 8.0, 8.1, 9.0, and 9.1 • ATMI clients running in WebLogic Enterprise release 5.1 • Jolt clients running in Jolt 1.2, 1.2.1, 8.0, 8.1, 9.0 and 9.1(via Jolt server 10.0)
BEA Tuxedo 9.1 ATMI server	• ATMI clients running in Tuxedo 6.5, 7.1, 8.0, 8.1 and 9.0 • ATMI clients running in WebLogic Enterprise release 5.1 • Jolt clients running in Jolt 1.2, 1.2.1, 8.0, 8.1 and 9.0 (via Jolt server 9.1)
BEA Tuxedo 9.0 ATMI server	• ATMI clients running in Tuxedo 6.5, 7.1, 8.0 and 8.1 • ATMI clients running in WebLogic Enterprise release 5.1 • Jolt clients running in Jolt 1.2, 1.2.1, 8.0 and 8.1 (via Jolt server 9.0)
BEA Tuxedo 8.1 ATMI server	• ATMI clients running in Tuxedo 6.5, 7.1, and 8.0 • ATMI clients running in WebLogic Enterprise release 5.1 • Jolt clients running in Jolt 1.2, 1.2.1, and 8.0 (via Jolt server 8.1)
BEA Tuxedo 10.0 CORBA server	• CORBA clients running in Tuxedo 8.1, 9.0 and 9.1 • CORBA clients running in WebLogic Enterprise 5.1
BEA Tuxedo 9.1 CORBA server	• CORBA clients running in Tuxedo 8.1 and 9.0 • CORBA clients running in WebLogic Enterprise 5.1

Table 1-2 Client-Server Interoperability

This component . . .	Can interoperate with . . .
BEA Tuxedo 9.0 CORBA server	• CORBA clients running in Tuxedo 8.1 • CORBA clients running in WebLogic Enterprise 5.1
BEA Tuxedo 8.1 CORBA server	• CORBA clients running in Tuxedo 8.0 • CORBA clients running in WebLogic Enterprise 5.1
BEA Tuxedo10.0 ATMI client	• ATMI servers running in Tuxedo 6.5, 7.1, 8.0. 8.1, 9.0, and 9.1 • ATMI servers running in WebLogic Enterprise 5.1
BEA Tuxedo 9.1 ATMI client	• ATMI servers running in Tuxedo 6.5, 7.1, 8.0. 8.1 and 9.0 • ATMI servers running in WebLogic Enterprise 5.1
BEA Tuxedo 9.0 ATMI client	• ATMI servers running in Tuxedo 6.5, 7.1, 8.0 and 8.1 • ATMI servers running in WebLogic Enterprise 5.1
BEA Tuxedo 8.1 ATMI client	• ATMI servers running in Tuxedo 6.5, 7.1, and 8.0 • ATMI servers running in WebLogic Enterprise 5.1
BEA Tuxedo 9.1 .NET client	• ATMI servers running in Tuxedo 6.5, 7.1, 8.0. 8.1 and 9.0 • ATMI servers running in WebLogic Enterprise 5.1
BEA Tuxedo 10.0 CORBA client	• CORBA servers running in Tuxedo 8.1, 9.0, 9.1 and 10.0 • CORBA servers running in WebLogic Enterprise 5.1
BEA Tuxedo 9.1 CORBA client	• CORBA servers running in Tuxedo 8.1 and 9.0 • CORBA servers running in WebLogic Enterprise 5.1
BEA Tuxedo 9.0 CORBA client	• CORBA servers running in Tuxedo 8.1 • CORBA servers running in WebLogic Enterprise 5.1

Table 1-2 Client-Server Interoperability

This component . . .	Can interoperate with . . .
BEA Tuxedo 8.1 CORBA client	• CORBA servers running in Tuxedo 8.0 • CORBA servers running in WebLogic Enterprise 5.1
BEA Jolt10.0 client	• ATMI servers running in Tuxedo 6.5 (via Jolt server 1.2) • ATMI servers running in Tuxedo 7.1 (via Jolt server 1.2.1) • ATMI servers running in Tuxedo 8.0 (via Jolt server 8.0) • ATMI servers running in Tuxedo 8.1 (via Jolt server 8.1) • ATMI servers running in Tuxedo 9.0 (via Jolt server 9.0) • ATMI servers running in Tuxedo 9.1 (via Jolt server 9.1) • ATMI servers running in WebLogic Enterprise 5.1 (via Jolt server 1.2)
BEA Jolt 9.1 client	• ATMI servers running in Tuxedo 6.5 (via Jolt server 1.2) • ATMI servers running in Tuxedo 7.1 (via Jolt server 1.2.1) • ATMI servers running in Tuxedo 8.0 (via Jolt server 8.0) • ATMI servers running in Tuxedo 8.1 (via Jolt server 8.1) • ATMI servers running in Tuxedo 9.0 (via Jolt server 9.0) • ATMI servers running in WebLogic Enterprise 5.1 (via Jolt server 1.2)

Table 1-2 Client-Server Interoperability

This component . . .	Can interoperate with . . .
BEA Jolt 9.0 client	• ATMI servers running in Tuxedo 6.5 (via Jolt server 1.2) • ATMI servers running in Tuxedo 7.1 (via Jolt server 1.2.1) • ATMI servers running in Tuxedo 8.0 (via Jolt server 8.0) • ATMI servers running in Tuxedo 8.1 (via Jolt server 8.1) • ATMI servers running in WebLogic Enterprise 5.1 (via Jolt server 1.2)
BEA Jolt 8.1 client	• ATMI servers running in Tuxedo 6.5 (via Jolt server 1.2) • ATMI servers running in Tuxedo 7.1 (via Jolt server 1.2.1) • ATMI servers running in Tuxedo 8.0 (via Jolt server 8.0) • ATMI servers running in WebLogic Enterprise 5.1 (via Jolt server 1.2)

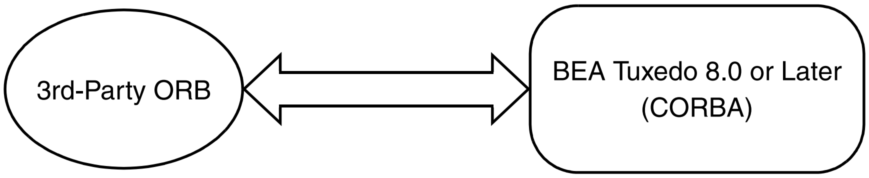
The capabilities available to a client for a particular client-server pair depend on the release of both the application client and the server application. For example, if you have a BEA Tuxedo 10.0 ATMI client interoperating with a BEA Tuxedo 6.5 server application, only BEA Tuxedo 6.5 functionality is available to the client.

Interoperability with Third-Party ORBs

Bootstrapping a BEA Tuxedo CORBA domain establishes communication between a CORBA application client and the domain. Two bootstrapping mechanisms are available: (1) the BEA mechanism using the Bootstrap object and (2) the CORBA Interoperable Naming Service (INS) bootstrapping mechanism specified by the OMG.

Support for INS was added in BEA Tuxedo release 8.0. With the addition of INS, third-party ORBs that use INS are able to interoperate with the BEA Tuxedo CORBA server ORB.

Figure 1-5 Interoperability with Third-Party ORBs



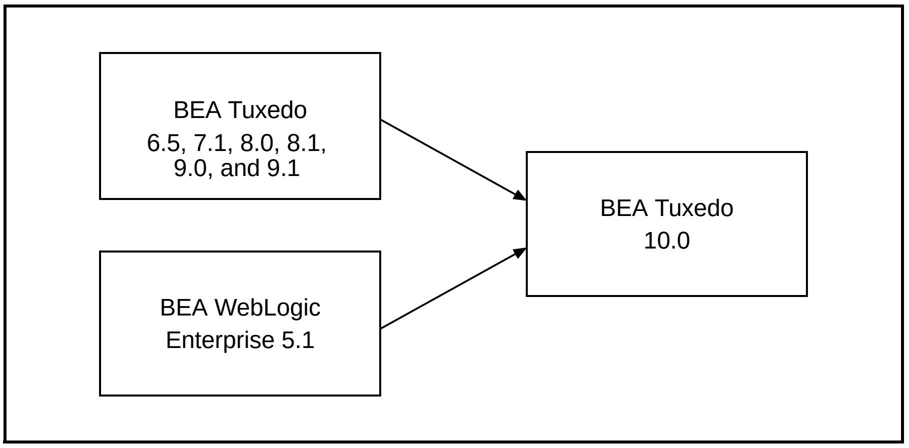
Note: The BEA Tuxedo CORBA client environmental objects continue to be supported in BEA Tuxedo 10.0, just as they were supported in BEA Tuxedo 8.0, 8.1, 9.0, 9.1 and BEA WebLogic Enterprise 5.1.

A CORBA application client uses the BEA Tuxedo Bootstrap object or the INS bootstrapping mechanism to obtain references to the objects in a BEA Tuxedo CORBA domain. BEA client ORBs use the BEA mechanism, and third-party client ORBs use the CORBA INS mechanism. For more information about bootstrapping a BEA Tuxedo domain, see *BEA Tuxedo CORBA Programming Reference*.

Product Upgrades

The following figure shows the existing BEA Tuxedo and BEA WebLogic Enterprise products that can be upgraded to BEA Tuxedo 10.0.

Figure 1-6 Upgrade Paths



If customers can shut down the domain (application) targeted for the upgrade, they should shut down the domain and perform a *simple upgrade*. If customers cannot shut down the domain targeted for the upgrade, they can perform a *hot upgrade*, that is, add the BEA Tuxedo 10.0 system software to the existing BEA Tuxedo or BEA WebLogic Enterprise domain without shutting down the domain.

For instructions on performing a simple upgrade or a hot upgrade, see "Upgrading the BEA Tuxedo System to Release 10.0" in *Installing the BEA Tuxedo System*.

Upward Application Compatibility

Applications developed with BEA Tuxedo 7.1, 8.0, 8.1, 9.0, and 9.1 are upwardly compatible with the BEA Tuxedo 10.0 release; however, relinking may be necessary.

Tuxedo 6.x applications must be recompiled to run on Tuxedo 10.0. Existing WLE 5.1 and earlier applications must be regenerated, recompiled, and relinked to run on Tuxedo 10.0.

Notes: For XML-related applications, you must conform to Xerces C++ 2.5 interface requirements.

On Windows platform, the binary must be relinked if FML-related functions are used.

Interoperability with BEA WebLogic Server

The following sections present interoperability capabilities between BEA Tuxedo and BEA WebLogic Server:

- Interoperability Software Components
- Interoperability Programming Interfaces
- JSL/JSH-Jolt Unidirectional Connectivity
- TDomain-WTC Bidirectional Connectivity
- RMI-over-IIOP Client Direct Connectivity to an EJB
- Summary of Interoperability Capabilities
- Interoperability Sample Applications

Interoperability Software Components

Interoperability between BEA Tuxedo and BEA WebLogic Server is implemented as the following three sets of communicating software processes.

Set	Tuxedo Component	Interoperability Direction	WebLogic Server Component	Interoperability
1	Jolt Server Listener/ Jolt Server Handler	←	BEA Jolt for BEA WebLogic Server	Enables WebLogic Server application servers to call Tuxedo ATMI services.
2	TDomain gateway	↔	WebLogic Tuxedo Connector (WTC)	Enables WebLogic Server application servers to call Tuxedo ATMI services. Enables WebLogic Server application servers to call Tuxedo CORBA C++ objects. Enables Tuxedo ATMI clients or servers to call WebLogic Server application servers. Enables Tuxedo CORBA C++ clients or servers to call WebLogic Server application servers.

Jolt Server Listener

A Jolt Server Listener (JSL) is a listening process, running on the Tuxedo server, that accepts connection requests from Jolt clients and assigns connections to a Jolt Server Handler also running on the Tuxedo server. It also manages the pool of Jolt Server Handler processes, starting them in response to load demands.

Jolt Server Handler

A Jolt Server Handler (JSH) is a gateway process, running on the Tuxedo server, that handles communications between Jolt clients and the Tuxedo ATMI server application. A JSH process resides within the administrative domain of the application and is registered in the local Tuxedo bulletin board as a client.

Each JSH process can manage multiple Jolt clients. A JSH multiplexes all requests and replies with a particular Jolt client over a single connection.

BEA Jolt for WebLogic Server

BEA Jolt is a Java-based client API that manages requests to Tuxedo services via a Jolt Service Listener (JSL) running on the Tuxedo server. The Jolt API is embedded within the WebLogic API and is accessible from a servlet or any other BEA WebLogic application.

IIOP Listener

An IIOP Listener (ISL) is a listening process, running on the Tuxedo server, that accepts connection requests from CORBA clients and assigns connections to an IIOP Handler also running on the Tuxedo server. It also manages the pool of IIOP Handler processes, starting them in response to load demands.

IIOP Handler

An IIOP Handler (ISH) is a gateway process, running on the Tuxedo server, that handles IIOP communications between CORBA clients and the Tuxedo server application. An ISH process resides within the administrative domain of the application and is registered in the local BEA Tuxedo bulletin board as a client.

Each ISH process can manage multiple CORBA clients. An ISH multiplexes all requests and replies with a particular CORBA client over a single connection.

TDomain Gateway

The TDomain gateway, implemented by the `GWTDOMAIN` server process, provides interoperability between two or more BEA Tuxedo domains through a specially designed transaction processing protocol that flows over network protocol TCP/IP. Working with the WebLogic Tuxedo Connector gateway, the BEA Tuxedo TDomain gateway can also provide interoperability between Tuxedo domains and WebLogic Server applications.

WebLogic Tuxedo Connector

The WebLogic Tuxedo Connector (WTC) enables bi-directional interoperability between the WebLogic Server and Tuxedo ATMI and CORBA environments. The WTC gateway supports the TDomain gateway protocol.

Tuxedo 10.0 supports the following WebLogic/WTC versions:

- WLS 10.0

- WLS 9.2, 9.1, 9.0
- WLS 8.1
- WLS 7.0

For a complete list of supported WLS versions through all Oracle Tuxedo release, see *Oracle Tuxedo Certified Platform Tables*.

Interoperability Programming Interfaces

Interoperability between BEA Tuxedo and BEA WebLogic Server is achieved using the following application programming interfaces:

- Application-to-Transaction Monitor Interface (ATMI)
- Java Application-to-Transaction Monitor Interface (JATMI)
- Jolt API
- Remote Method Invocation (RMI)
- Remote Method Invocation (RMI) over Internet Inter-ORB Protocol (IIOP) (RMI-over-IIOP)
- CORBA Java

ATMI Interface

ATMI provides an interface for communications, transactions, and data-buffer management that works in all ATMI environments supported by the BEA Tuxedo system. ATMI is described in *Introducing BEA Tuxedo ATMI*.

JATMI Interface

JATMI is the BEA WebLogic Server Java implementation of the BEA Tuxedo ATMI. It allows WebLogic Server application servers to access Tuxedo ATMI services. JATMI is described in *WebLogic Tuxedo Connector Programmer's Guide* at `http://e-docs.bea.com/wls/docs100/wtc_atmi/index.html`.

Jolt Interface

BEA Jolt for BEA WebLogic Server is a Java-based client API that manages requests to BEA Tuxedo services running on the Tuxedo server. The Jolt API is embedded within the WebLogic API and is accessible from a servlet or any other BEA WebLogic application. Jolt API is described in *Using BEA Jolt with BEA WebLogic Server*.

RMI Interface

Remote Method Invocation is a Java-based API set and protocol that allows an object running in one Java virtual machine to invoke methods on an object running in a different Java virtual machine. RMI specifies how distributed Java applications should operate over multiple Java virtual machines. RMI's native protocol is called Java Remote Method Protocol (JRMP).

For more information about RMI, see *Programming WebLogic RMI* at `http://e-docs.bea.com/wls/docs100/rmi/index.html`.

RMI-over-IIOP Interface

RMI-over-IIOP provides interoperability with CORBA objects implemented in any language if all the remote interfaces are originally defined as RMI interfaces. RMI-over-IIOP is also known as RMI-on-IIOP, RMI/IIOP, or RMI-IIOP. The term RMI-over-IIOP is used in the discussions that follow.

With RMI and CORBA, programmers must decide between RMI, with its easy programming features, and CORBA, with its broad interoperability. IBM and Sun's JavaSoft, with the cooperation of the Object Management Group (OMG), jointly developed RMI-over-IIOP to solve this dilemma. JavaSoft includes RMI-over-IIOP in its Java Development Kit (JDK).

With RMI-over-IIOP, Java programmers can create applications in RMI that include CORBA connections. And with CORBA 2.3's support for Objects-by-Value, CORBA programmers can create applications in CORBA that include EJB connections.

Note: For information on Objects-by-Value and supported value types in BEA Tuxedo CORBA, see "Mapping of OMG IDL Statements to C++" in *BEA Tuxedo CORBA Programming Reference*.

With RMI-over-IIOP and CORBA's support for Objects-by-Value, the following client-server interfaces are possible:

- RMI client → RMI-over-IIOP server

- CORBA client $\rightarrow$ RMI-over-IIOP server
- RMI-over-IIOP client $\rightarrow$ RMI server
- RMI-over-IIOP client $\rightarrow$ CORBA server
- RMI-over-IIOP client $\rightarrow$ RMI-over-IIOP server

Note: Of course, for the "RMI-over-IIOP client $\rightarrow$ CORBA server" interface, an RMI-over-IIOP client cannot necessarily access all existing CORBA objects because the semantics of CORBA objects defined in IDL are a superset of those of RMI-over-IIOP objects. Thus, an existing CORBA's object's IDL cannot always be mapped into an RMI-over-IIOP Java interface.

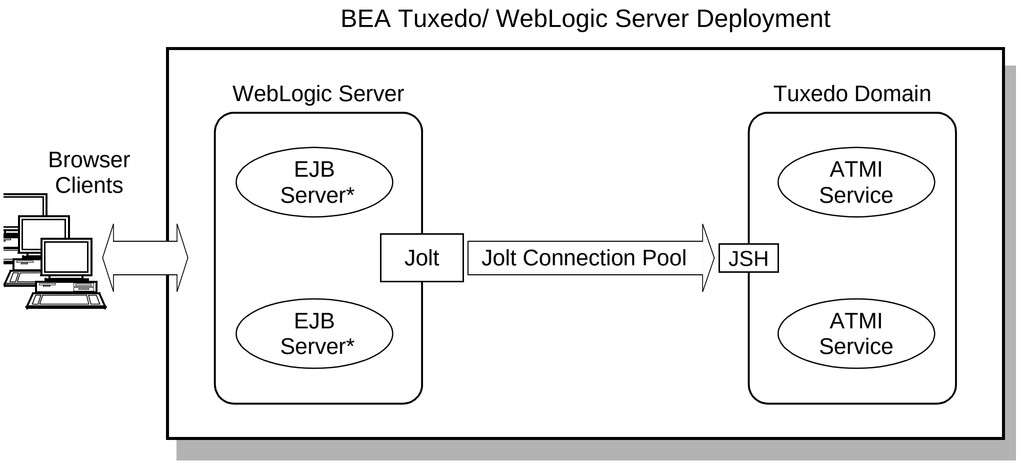
A server binary (i.e., a class file) created using RMI-over-IIOP APIs can be exported as JRMP (RMI's native protocol), IIOP, or both. Exporting an RMI-over-IIOP object to both JRMP and IIOP simultaneously is called *dual export*.

For more information about RMI-over-IIOP, see *Programming WebLogic RMI over IIOP* at `http://e-docs.bea.com/wls/docs100/rmi_iiop/index.html`.

JSL/JSH-Jolt Unidirectional Connectivity

BEA Jolt for WebLogic Server provides unidirectional connectivity from BEA WebLogic Server applications to BEA Tuxedo 8.0 or later ATMI services. With BEA Jolt for WebLogic Server, an application administrator can enable Tuxedo services for the Web, using the WebLogic Server as the front-end HTTP and application server. The following figure shows how this connectivity is implemented.

Figure 2-1 WebLogic Server to BEA Tuxedo Connectivity Using Jolt



* Or other WebLogic Server application servers including JSPs and servlets

BEA Jolt is a Java-based client API that manages requests to BEA Tuxedo services using a Jolt Server Listener running on the Tuxedo server. The Jolt API is accessible to an EJB, a JSP, a servlet, a Java HTML (JHTML), or other BEA WebLogic application server.

Jolt Connection Pooling

WebLogic Server uses a variation of the Jolt session pool called a *servlet session pool*, commonly referred to as simply a *Jolt connection pool*. The Jolt connection pool provides extra functionality that is convenient for use inside an HTTP servlet.

Jolt connection pooling allows WebLogic Server application servers to invoke Tuxedo services in a BEA Tuxedo application. The pooling feature supports connection pool reset in the event of connection pool failure, which eliminates the need to restart WebLogic Server if the connection pool requires a restart.

Jolt Wire-Level Security

The following wire-level security is supported on the network connection between the Java Server Handler and WebLogic Server: 40-bit, 56-bit, or 128-bit LLE. LLE, for Link-Level Encryption, is a Tuxedo-based protocol for establishing data privacy over network links.

Jolt Transaction and Security Context Propagation

Jolt supports transaction demarcation, propagation of security, and connection reset. Jolt provides a mechanism for propagating the security context established in WebLogic Server to the BEA Tuxedo application.

User credentials authenticated by WebLogic Server are mapped to the appropriate security interfaces/protocols. An incoming request does not require re-authentication before invoking Tuxedo ATMI services.

Jolt Documentation

For complete information on using BEA Jolt with WebLogic Server, see *Using BEA Jolt with BEA WebLogic Server*. This document explains the operation of BEA Jolt for WebLogic Server, and describes how to use, configure, and integrate BEA Jolt, BEA Tuxedo ATMI, and BEA WebLogic Server.

TDomain-WTC Bidirectional Connectivity

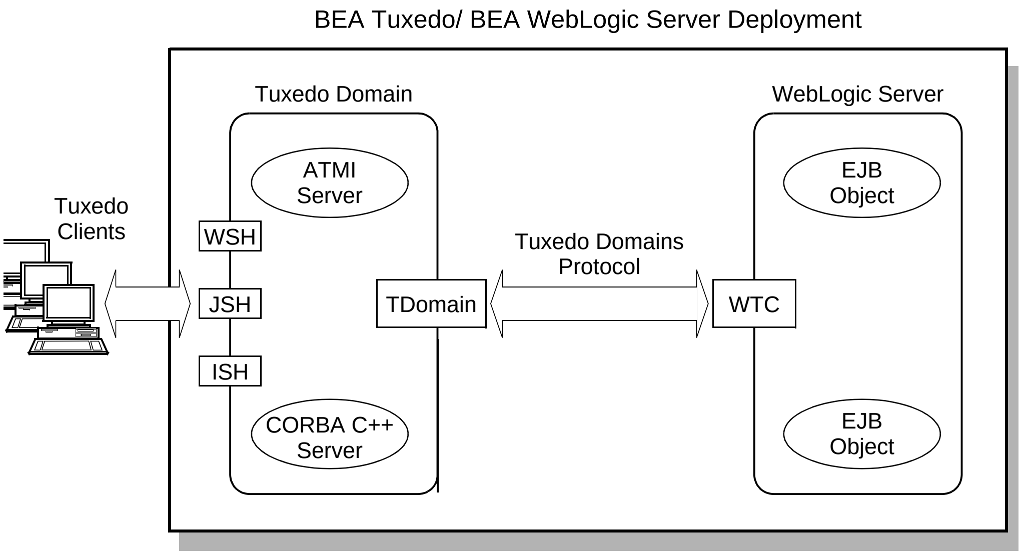
The TDomain and WTC gateways provide bidirectional connectivity *between* ATMI services/ CORBA objects deployed in a BEA Tuxedo application *and* EJB objects deployed in a WebLogic Server application. Together, the gateways allow the following interoperability for a BEA Tuxedo/ WebLogic Server deployment:

- Allow Tuxedo ATMI clients, and Tuxedo ATMI servers acting as clients, to access WebLogic Server EJB servers via the ATMI interface.
- Allow Tuxedo CORBA clients, and Tuxedo CORBA servers acting as clients, to access WebLogic Server EJB servers via RMI-over-IIOP.
- Allow WebLogic Server application servers (EJBs, JSPs, Java servlets) acting as clients to access:
  - Tuxedo ATMI servers via JATMI
  - Tuxedo CORBA servers via CORBA Java or RMI-over-IIOP

BEA Tuxedo to BEA WebLogic Server Connectivity

BEA Tuxedo application clients and servers can invoke EJB objects in a WebLogic Server application, which in turn can invoke other EJB objects, JSPs, or Java servlets. The following figure shows how this connectivity is implemented.

Figure 2-2 Tuxedo to WebLogic Server Connectivity



Note: Tuxedo clients include ATMI clients, Jolt clients, and CORBA C++ clients. For a high-level view of Tuxedo clients, see “Client and Server Components” in *BEA Tuxedo Product Overview*.

The TDomain gateway not only enables Tuxedo domains to share services with other BEA Tuxedo domains, but it enables Tuxedo domains to share services with WebLogic Server 6.1or later installations through the WTC gateway. The WTC gateway supports the TDomain gateway protocol.

The gateways allow Tuxedo ATMI clients, and Tuxedo ATMI servers acting as clients, to access WebLogic Server EJB objects. The TDomain gateway delivers the ATMI client request to the WTC gateway, and the WTC gateway converts the request to an RMI call to access the appropriate EJB object.

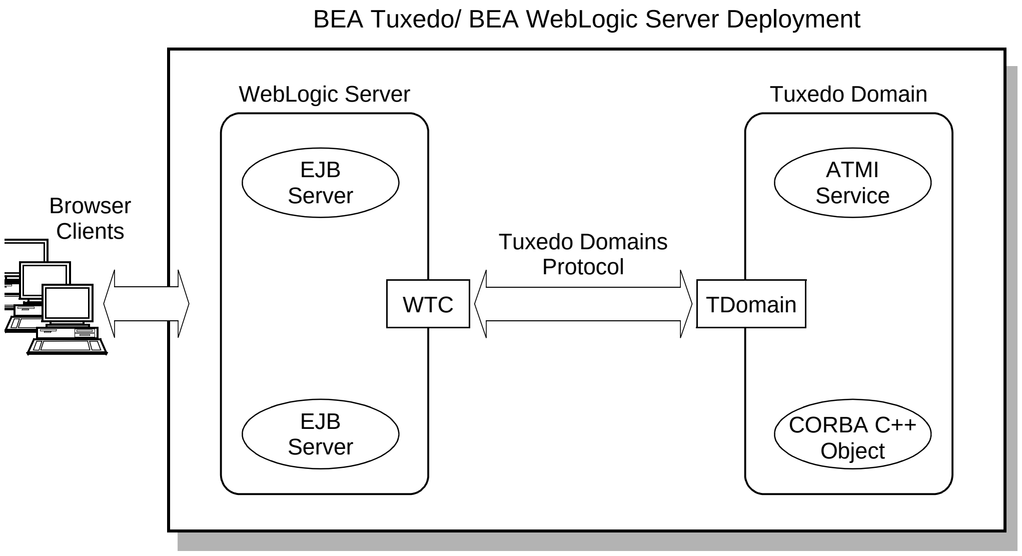
Similarly, the gateways allow Tuxedo CORBA clients, and Tuxedo CORBA servers acting as clients, to access WebLogic Server EJB objects. The TDomain gateway delivers the CORBA RMI-over-IIOP client request to the WTC gateway, and the WTC gateway forwards the request to the appropriate EJB object.

Note: The WTC component delivered with WebLogic Server release 7.0, 8.1, 9.x and 10.0 supports all the functionality included in the WTC 1.0 product. BEA encourages the use of the WTC component delivered with the WebLogic Server product.

BEA WebLogic Server to BEA Tuxedo Connectivity

EJB application servers in a WebLogic Server application can invoke services and CORBA objects in a Tuxedo application using the WTC and TDomain gateways. The following figure shows how this connectivity is implemented.

Figure 2-3 WebLogic Server to Tuxedo Connectivity Using WTC



The WTC and TDomain gateways allow WebLogic Server EJBs, JSPs, or Java servlets acting as clients to access Tuxedo services. The WTC gateway converts the EJB/JSP/servlet JATMI request to an ATMI request, and the TDomain gateway delivers the ATMI request to a Tuxedo ATMI server offering the requested service.

Similarly, the gateways allow WebLogic Server EJBs, JSPs, or Java servlets acting as clients to access Tuxedo CORBA objects. The WTC gateway inserts the EJB/JSP/servlet CORBA Java or RMI-over-IIOP request inside of a Tuxedo GIOP (TGIOP) request message, and the TDomain gateway delivers the TGIOP request to a Tuxedo CORBA server offering the requested object.

Note: The WTC component delivered with WebLogic Server release 7.0, 8.1, 9.x and 10.0 supports all the functionality included in the WTC 1.0 product. BEA encourages the use of the WTC component delivered with the WebLogic Server product.

TDomain-WTC Wire-Level Security

The following wire-level security is supported on the network connection between the TDomain and WTC gateways: 40-bit, 56-bit, or 128-bit LLE. LLE, for Link-Level Encryption, is a Tuxedo-based protocol for establishing data privacy over network links.

TDomain-WTC Transaction and Security Context Propagation

Bidirectional propagation of transaction context and security context between application clients and servers in a BEA Tuxedo/ WebLogic Server deployment is fully supported through the TDomain and WTC gateways.

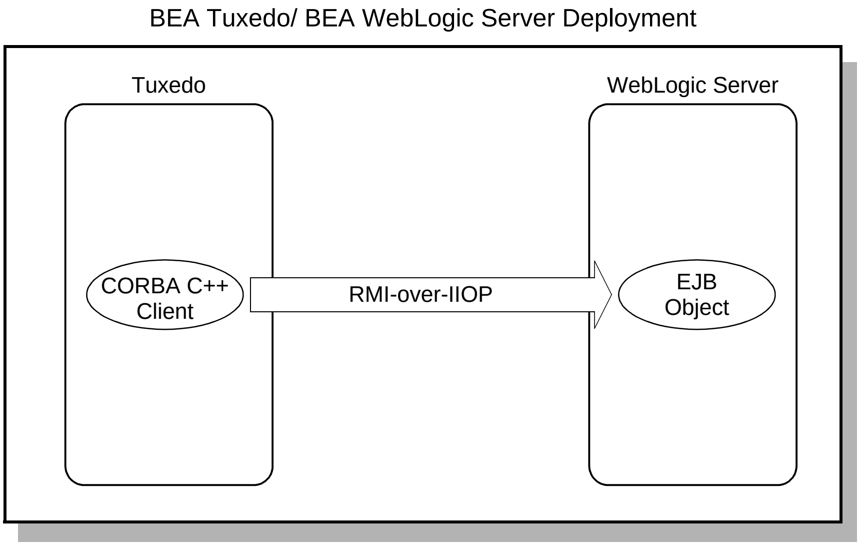
TDomain and WTC Documentation

For details about the Tuxedo Domains gateway, see *Using the BEA Tuxedo Domains Component*. For details about the WTC gateway, see *WebLogic Tuxedo Connector* at `http://e-docs.bea.com/wls/docs100/wtc.html`.

RMI-over-IIOP Client Direct Connectivity to an EJB

In addition to using the TDomain and WTC gateways to achieve connectivity from BEA Tuxedo CORBA to BEA WebLogic Server, Tuxedo CORBA C++ clients or servers can call WebLogic Server application servers *directly* using RMI-over-IIOP and CORBA Interface Definition Language (IDL) interfaces. The following figure demonstrates this type of connectivity.

Figure 2-4 Direct EJB Connectivity Using RMI-over-IIOP and IDL Interfaces



For a sample application describing how a CORBA C++ client application developed in BEA Tuxedo can directly interact with an EJB in WebLogic Server, see *Connectivity Between a BEA Tuxedo CORBA Client and an EJB in WebLogic Server* at `http://edocs.bea.com/tuxedo/tux80/interop/ioptxwls.htm`.

Summary of Interoperability Capabilities

The following table summarizes the interoperability capabilities for a BEA Tuxedo/ WebLogic Server deployment.

Table 2-1 BEA Tuxedo/ WebLogic Server Interoperability Capabilities

This component . . .	Can call a . . .	Through . . .
Tuxedo ATMI client *	WebLogic Server EJB object	WSH ** → TDomain → WTC
Tuxedo Jolt client ***	WebLogic Server EJB object	JSH → TDomain → WTC
Tuxedo CORBA C++ client *	WebLogic Server EJB object	ISH → TDomain → WTC *or* RMI-over-IIOP client direct connectivity to an EJB
Tuxedo ATMI server	WebLogic Server EJB object	TDomain → WTC
Tuxedo CORBA C++ server	WebLogic Server EJB object	TDomain → WTC *or* RMI-over-IIOP client direct connectivity to an EJB
WebLogic Server EJB, JSP, or servlet	Tuxedo ATMI service	WTC → TDomain *or* Jolt for WebLogic Server → JSH
WebLogic Server EJB, JSP, or servlet	Tuxedo CORBA C++ object	WTC → TDomain

* A native Tuxedo ATMI or CORBA C++ client does not use Tuxedo handler gateway processes (WSH, ISH).

** WSH stands for Workstation Handler.

*** The Tuxedo Jolt client connection to a WebLogic Server EJB object has not been tested.

Interoperability Sample Applications

WebLogic Server release 7.0 or later includes a large variety of interoperability sample applications. The sample applications provide client and server programmers with information about the basic concepts of (1) combining Tuxedo ATMI services and WebLogic Server EJB objects in an application and (2) combining Tuxedo CORBA objects and WebLogic Server EJB objects in an application.

For a WebLogic Server 7.0 or 8.1 installation, the ATMI are located in the following directory:

`WL_HOME\samples\server\src\examples\wtc`

Where *`WL_HOME`* represents the top-level directory of the WebLogic Server 7.0 or 8.1 installation (`weblogic700` by default). These examples show how to configure and set up WebLogic Server to work with Tuxedo ATMI servers and clients, using the underlying WTC technology.

For a WebLogic Server 7.0 or 8.1 installation, the RMI-over-IIOP code examples are located in the following directory:

```
WL_HOME\samples\server\src\examples\iiop
```

These examples show how to configure and set up WebLogic Server to work with Tuxedo CORBA servers and clients, using the underlying WTC technology.

For additional information on how to develop interoperability applications employing ATMI, JATMI, CORBA Java, or RMI-over-IIOP API, see *WebLogic Tuxedo Connector* at `http://e-docs.bea.com/wls/docs100/wtc.html`.