

# Oracle® Tuxedo JCA Adapter Inflow Transaction

Users Guide

11g Release 1 (11.1.1.2)

November 2010

ORACLE®

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

# Oracle Tuxedo JCA Adapter Inflow Transaction Guide

Packaging and Contents . . . . .	1
Supported Application Servers and Oracle Tuxedo Versions . . . . .	1
RAR File Name . . . . .	2
RAR File Contents . . . . .	2
Overview . . . . .	3
Configuration . . . . .	4
Oracle Tuxedo JCA Adapter Configuration . . . . .	4
Application Server Resource Adapter Configuration . . . . .	8
Oracle Tuxedo GWTDOMAIN Gateway Configuration . . . . .	20
Oracle Tuxedo JCA Adapter Deployment . . . . .	22
Configure dmconfig File . . . . .	22
Resource Adapter Deployment Descriptor . . . . .	23
Deploy Oracle Tuxedo JCA Adapter . . . . .	37
Configure Activation Spec . . . . .	39
Configuring MDB Using WebSphere Integrated Console . . . . .	39
Deploy MDB To WebSphere . . . . .	39
Oracle Tuxedo Application Domain . . . . .	42
Oracle Tuxedo Configuration . . . . .	42
Programming MDB For Oracle Tuxedo JCA Adapter . . . . .	46
Interface TuxedoMDBService . . . . .	46
Creating an Inbound Connector-Based MDB Using IBM ASTK . . . . .	49
Use J2EE Perspective . . . . .	49
Create EJB Project . . . . .	49
Setup Build Environment . . . . .	49
Create Message-Driven Bean . . . . .	50
Modify EchoBean . . . . .	52

Build .....	53
Create EJB JAR File .....	54
Oracle Tuxedo Transactional Client Source Code .....	57





# Oracle Tuxedo JCA Adapter Inflow Transaction Guide

This document describes the configuration, deployment, and programming of the inflow transaction through *connector-based MDB* (also called *none JMS-based MDB*) feature. This feature allows inflow (inbound) transactions from Oracle Tuxedo to a Java application server. It contains the following topics:

- [Packaging and Contents](#)
- [Overview](#)
- [Configuration](#)
- [Oracle Tuxedo JCA Adapter Deployment](#)
- [Programming MDB For Oracle Tuxedo JCA Adapter](#)

## Packaging and Contents

The Inflow Transaction feature is delivered as an RAR file; you must un-jar the file and modify the Oracle Tuxedo JCA Adapter configuration before it can be installed on an application server.

## Supported Application Servers and Oracle Tuxedo Versions

[Table 1](#) lists supported Oracle Tuxedo and application server versions.

**Table 1 Supported Versions**

Name	Version
WebSphere Application Server	7.0
Oracle Tuxedo	11gR1PS1

## RAR File Name

The RAR file name is `com.oracle.tuxedo.TuxedoAdapter.rar`. After you modify the Oracle Tuxedo JCA Adapter configuration, it can be archived (using any name) be used to configure Oracle Tuxedo JCA Adapter to the application server.

## RAR File Contents

[Table 2](#) lists the RAR file contents.

**Table 2 RAR File Content**

File Name	Description
<code>adapter.properties</code>	Message catalogue of <i>Oracle Tuxedo JCA Adapter</i> .
<code>adapter_ja.properties</code>	Japanese version of message catalogue of <i>Oracle Tuxedo JCA Adapter</i> .
<code>com.bea.core.i18n_1.4.0.0.jar</code>	I18N library
<code>com.bea.core.jatmi_1.3.2.0.jar</code>	JATMI library
<code>com.oracle.tuxedo.adapter_1.2.1.0.jar</code>	Oracle Tuxedo JCA Adapter
<code>dmconfig.xml</code>	Sample <i>dmconfig</i> file for <i>Oracle Tuxedo JCA Adapter</i> configuration.
<code>javax.ejb_3.0.1.jar</code>	EJB 3.0 library
<code>javax.transaction_1.0.0.0_1-1.jar</code>	JTA 1.1 library
<code>tja.xsd</code>	Schema file for <i>Oracle Tuxedo JCA Adapter</i> .



**Table 2 RAR File Content**

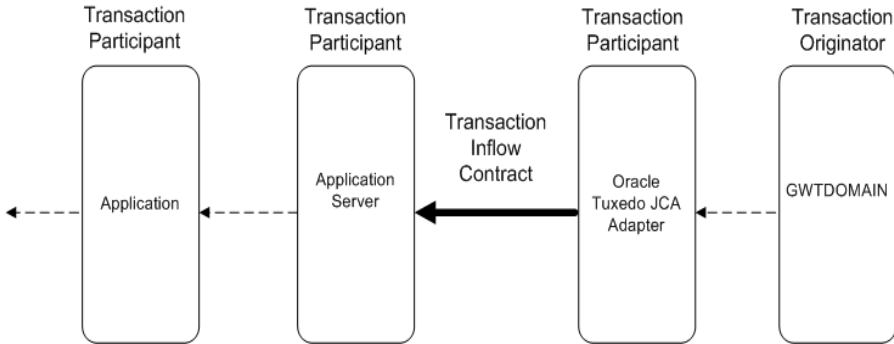
META-INF/MANIFEST.MF	Manifest file
META-INF/client-side.ra.xml	Sample client-side only resource adapter deployment descriptor
META-INF/ra.xml	Sample resource adapter deployment descriptor for connection factory based configuration
META-INF/sample.weblogic-ra.xml	Sample weblogic-ra.xml file for <b>WebLogic Server</b> for connection factory based configuration
META-INF/server.ra.xml	Sample resource adapter deployment descriptor for <i>dmconfig</i> based configuration
META-INF/weblogic-ra.xml	Sample weblogic-ra.xml file for <b>WebLogic Server</b>

## Overview

The Oracle Tuxedo JCA Adapter supports Oracle Tuxedo TDOMAIN protocol including its transaction context format as shown in [Figure 1](#). When the Oracle Tuxedo JCA Adapter receives an inbound request from Oracle Tuxedo, it checks whether there is an associated Oracle Tuxedo transaction context or not. If it does have it, then the Oracle Tuxedo JCA Adapter creates a `javax.transaction.xa.xid` based on the Oracle Tuxedo transaction context.

The Oracle Tuxedo JCA Adapter supplies this XID to an *ExecutionContext* and submits the *Work* instance along with the *ExecutionContext* to the application server *WorkManager* for execution. By propagating an imported transaction to a Java application server this way, the application server and subsequent participants can work as part of the imported transaction.

**Figure 1 Oracle Tuxedo TDOMAIN Protocol**



## Configuration

Configuration can be separated into two parts. The first part is Oracle Tuxedo JCA Adapter configuration, and the second part is configuring the adapter in an application server.

### Oracle Tuxedo JCA Adapter Configuration

An “Exported” service is a Java resource that can be accessed by an Oracle Tuxedo client; in this particular case it is the connector-based MDB. You must configure the “Export” element in the `dmconfig` file for an Oracle Tuxedo client to access resources located in the Java application server.

A single “Export” element in the `dmconfig` file refers to an exported resource to the Oracle Tuxedo client. Listing 1 shows two exported services (`Tolower` and `Echo`), to an Oracle Tuxedo client. The `RemoteName` is the service name the Oracle Tuxedo GWTDOMAIN gateway uses to invoke the service; the `name` attribute is the service name of the resource. The `Type` must be `MDB` for inflow transaction, and the `Source` is the JNDI binding of the MDB.

#### Listing 1 Exported Services Example

---

```
...  
<Export name="Tolower">  
  <RemoteName>TolowerMDB</RemoteName>  
  <SessionName>session_1</SessionName>
```

```

<Type>MDB</Type>
<Source>eis/Tolower</Source>
</Export>
<Export name="Echo">
  <RemoteName>EchoMDB</RemoteName>
  <SessionName>session_1</SessionName>
  <Type>MDB</Type>
  <Source>eis/Echo</Source>
</Export>
...

```

---

Multiple exported services using single MDB is also supported. The purpose is to give greater freedom to the adapter application developer. You can configure them using the same JNDI name specified in the *dmconfig* file `Source` element; however, since there is only one interface implemented by the application for that MDB, the application must do the dispatching itself. [Listing 2](#) shows an example of multiple exported services using single MDB.

---

### Listing 2 Multiple Exported Services Example

---

```

...
<Export name="INFO_SERVICE">
  <RemoteName>INFO</RemoteName>
  <SessionName>session_1</SessionName>
  <Type>MDB</Type>
  <Source>eis/services</Source>
</Export>
<Export name="ACCOUNT_SERVICE">
  <RemoteName>ACCOUNT</RemoteName>
  <SessionName>session_1</SessionName>

```

```
<Type>MDB</Type>
<Source>eis/services</Source>
</Export>
...
```

---

This example exports two services `INFO` and `ACCOUNT` to an Oracle Tuxedo client using the same MDB that binds to JNDI name `eis/services`. In this case you must create and deploy one MDB that dispatches using the service name passed to the MDB. [Listing 3](#) shows an example MDB Code Fragment doing its own dispatching.

### Listing 3 MDB Code Fragment

---

```
...
public Reply service(TPServiceInformation mydata)
    throws TuxedoReplyException
{
    String serviceName = mydata.getServiceName();
    if (serviceName.equals("ACCOUNT_SERVICE")) {
        doAccount1(mydata);
    }
    else if (serviceName.equals("INFO_SERVICE")) {
        doInfo(mydata);
    }
    else {
        /* throws an exception */
    }
}
```

---

[Listing 4](#) shows a complete Oracle Tuxedo JCA Adapter configuration file example.

---

#### Listing 4 Oracle Tuxedo JCA Adapter Configuration File Example

```
<?xml version="1.0" encoding="UTF-8"?><TuxedoConnector>
  <LocalAccessPoint name="JDOM">
    <AccessPointId>JDOM_ID</AccessPointId>
    <NetworkAddress>//localhost:10801</NetworkAddress>
  </LocalAccessPoint>
  <RemoteAccessPoint name="TDOM1">
    <AccessPointId>TDOM1_ID</AccessPointId>
    <NetworkAddress>//localhost:12478</NetworkAddress>
  </RemoteAccessPoint>
  <SessionProfile name="profile_1">
    <BlockTime>60000</BlockTime>
    <ConnectionPolicy>ON_STARTUP</ConnectionPolicy>
  </SessionProfile>
  <Session name="session_1">
    <LocalAccessPointName>JDOM</LocalAccessPointName>
    <RemoteAccessPointName>TDOM1</RemoteAccessPointName>
    <ProfileName>profile_1</ProfileName>
  </Session>
  <Export name="Tolower">
    <RemoteName>TolowerMDB</RemoteName>
    <SessionName>session_1</SessionName>
    <Type>MDB</Type>
    <Source>eis/tolower</Source>
  </Export>
```

```
<Export name="Echo">
  <RemoteName>EchoMDB</RemoteName>
  <SessionName>session_1</SessionName>
  <Type>MDB</Type>
  <Source>eis/echo</Source>
</Export>
<Export name="INFO_SERVICE">
  <RemoteName>INFO</RemoteName>
  <SessionName>session_1</SessionName>
  <Type>MDB</Type>
  <Source>eis/services</Source>
</Export>
<Export name="ACCOUNT_SERVICE">
  <RemoteName>ACCOUNT</RemoteName>
  <SessionName>session_1</SessionName>
  <Type>MDB</Type>
  <Source>eis/services</Source>
</Export>
</TuxedoConnector>
```

---

## Application Server Resource Adapter Configuration

You must configure the Resource Adapter Deployment Descriptor (`ra.xml`). The name, `ra.xml`, cannot be changed. Every RAR file must contain one `ra.xml` file. For inflow transactions using MDB to work, you must configure the `inbound-resourceadapter` element. This element is used to describe the interface and activation specification specific to the Oracle Tuxedo JCA Adapter.

The `inbound-resourceadapter` element is fixed. The `source` property is the only property that you can configure. If configured, the JCA container requires the `source` property to be specified in the EJB descriptor (`ejb-jar.xml`), file.

[Listing 5](#) shows an `ra.xml` file example. You can use the `ra.xml` file distributed with the Oracle Tuxedo JCA Adapter as a base and customize it as needed.

### Listing 5 ra.xml File Example

---

```
<?xml version="1.0" encoding="UTF-8"?>
<connector xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/connector_1_5.xsd"
  version="1.5">
  <display-name>Tuxedo JCA Adapter</display-name>
  <vendor-name>Oracle</vendor-name>
  <eis-type>Tuxedo</eis-type>
  <resourceadapter-version>11gR1(11.1.1.2.1)</resourceadapter-version>
  <license>
    <description>Tuxedo SALT license</description>
    <license-required>>false</license-required>
  </license>
  <resourceadapter>

  <resourceadapter-class>com.oracle.tuxedo.adapter.TuxedoResourceAdapter</re
sourceadapter-class>

  <!--
    The following is the list of properties name can be configured as
    adapter-wise configuration.

    traceLevel - java.lang.String - a numerical value
```

## <Default ? Font>Oracle Tuxedo JCA Adapter Inflow Transaction Guide

```
    xaAffinity - java.lang.String - transaction affinity to a remote
domain, "true" or "false", default to true

    keyFileName - java.lang.String - encryption key file name

    throwFailureReplyException - java.lang.Boolean - default to ture

    appManagedLocalTxTimeout - java.lang.Integer - Application managed
transaction or AUTOTRAN timeout

                                                defaults to 300 seconds

    fieldTable16Class - java.lang.String - a comma-separated list of fully
qualified FML classes

    fieldTable32class - java.lang.String - a comma-separated list of fully
qualified FML32 classes

    viewFile16Class - java.lang.String - a comma-separated list of fully
qualified VIEW classes

    viewFile32Class - java.lang.String - a comma-separated list of fully
qualified VIEW32 classes

    tpusrFile - java.lang.String - path name to the TPUSR file

    remoteMBEncoding - java.lang.String - remote Tuxedo encoding name for
multi-byte language

    mBEncodingMapFile - java.lang.String - path name to Multi-byte encoding
name mapping

    autoTran - java.lang.Boolean- enable adapter-wise AUTOTRAN,
default to false

-->
<outbound-resourceadapter>
    <connection-definition>

<managedconnectionfactory-class>com.oracle.tuxedo.adapter.spi.TuxedoManage
dConnectionFactory</managedconnectionfactory-class>
```

```
<!--
```

The following is the list of properties that you can use



to configure the connection pool or connection factory.

You must either configure `localAccessPointSpec` or `connectionFactoryName` if transaction is used.

These property described here is serving as template, user should not

configure them here, instead user should configure them either through WebSphere console

or `weblogic-ra.xml` side file.

```
-->
```

```
<config-property>
```

```
  <description>factory-wise AUTOTRAN setting, default to false,
  overrides adapter-wise setting</description>
```

```
  <config-property-name>autoTran</config-property-name>
```

```
  <config-property-type>java.lang.Boolean</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

```
  <description>factory-wise Failure Reply Exception setting, default
  to true, overrides adapter-wise setting</description>
```

```
<config-property-name>throwFailureReplyException</config-property-name>
```

```
  <config-property-type>java.lang.Boolean</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

```
  <description>factory-wise application managed transaction or
  AUTOTRAN time out, overrides adapter-wise setting</description>
```

```
<config-property-name>appManagedLocalTxTimeout</config-property-name>
```

```
  <config-property-type>java.lang.Integer</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

## <Default ? Font>Oracle Tuxedo JCA Adapter Inflow Transaction Guide

```
<description>connection factory or pool name, this is required if
XA or local application managed
        transaction is required</description>
<config-property-name>connectionFactoryName</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>application password in either clear text or cipher
text using com.oracle.tuxedo.tools.EncryptPassword tool</description>
  <config-property-name>applicationPassword</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>local access point specification of the format
//hostname:port/domainId=DOMAINID</description>
  <config-property-name>localAccessPointSpec</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise SSL to configure whether mutual
authentication is required, default to false</description>
<config-property-name>mutualAuthenticationRequired</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
</config-property>
<config-property>
  <description>factory-wise SSL for configuring identity key store
file name, must be configured if SSL is desired</description>
```

```

<config-property-name>identityKeyStoreFileName</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise SSL setting for private key alias used
in the key store, must be configured if SSL is desired</description>
    <config-property-name>privateKeyAlias</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise trusted key store file name, must be
configured if SSL is desired</description>

<config-property-name>trustedKeyStoreFileName</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise password for identityKeyStore in clear
text</description>

<config-property-name>identityKeyStorePassPhrase</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise password for privateKeyAlias in clear
text</description>

<config-property-name>privateKeyAliasPassPhrase</config-property-name>
    <config-property-type>java.lang.String</config-property-type>

```

## <Default ? Font>Oracle Tuxedo JCA Adapter Inflow Transaction Guide

```
</config-property>
<config-property>
  <description>factory-wise password for trustedKeyStore in clear
text</description>
<config-property-name>trustedKeyStorePassPhrase</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise RemoteAccessPoint specification of the
format //hostname:port/domainId=DOMAINID</description>
<config-property-name>remoteAccessPointSpec</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise allow anonymous access to Tuxedo, default
to false</description>
  <config-property-name>rapAllowAnonymous</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
</config-property>
<config-property>
  <description>factory-wise application key value for anonymous user,
default to -1</description>
<config-property-name>rapDefaultApplicationKey</config-property-name>
  <config-property-type>java.lang.Integer</config-property-type>
</config-property>
<config-property>
```

```

        <description>factory-wise application key fully qualified class
name for AppKey generator</description>

<config-property-name>rapApplicationKeyClass</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise custom application key
parameter</description>

<config-property-name>rapApplicationKeyClassParam</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise session profile block timeout value,
default to 60000 milliseconds</description>
    <config-property-name>spBlockTime</config-property-name>
    <config-property-type>java.lang.Integer</config-property-type>
</config-property>
<config-property>
    <description>factory-wise whether allows interoperate with 6.5
Tuxedo Domain, default to false</description>
    <config-property-name>spInteroperate</config-property-name>
    <config-property-type>java.lang.Boolean</config-property-type>
</config-property>
<config-property>
    <description>factory-wise security setting, legal values: NONE,
DM_PW, APP_PW</description>
    <config-property-name>spSecurity</config-property-name>

```

## <Default ? Font>Oracle Tuxedo JCA Adapter Inflow Transaction Guide

```
<config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise credential propagation policy, either
LOCAL or GLOBAL</description>
  <config-property-name>spCredentialPolicy</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise number of seconds that session waits
between automatic connection establishment,
          default to 60 seconds. A value of 0 disabled connection
retry</description>
  <config-property-name>spRetryInterval</config-property-name>
  <config-property-type>java.lang.Long</config-property-type>
</config-property>
<config-property>
  <description>factory-wise maximum number of times adapter will try
to establish a session connection to
          remote Tuxedo access point. Default value is
Long.MAX_VALUE.</description>
  <config-property-name>spMaxRetries</config-property-name>
  <config-property-type>java.lang.Long</config-property-type>
</config-property>
<config-property>
  <description>factory-wise compression threshold, default to
Integer.MAX_VALUE</description>
  <config-property-name>spCompressionLimit</config-property-name>
  <config-property-type>java.lang.Integer</config-property-type>
```

```

</config-property>
<config-property>
  <description>factory-wise minimum encryption strength requirement,
  legal values are 0, 40, 56, 128, 256.
      Default value is 0.</description>
  <config-property-name>spMinEncryptBits</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise maximum encryption strength requirement,
  legal values are 0, 40, 56, 128, 256.
      Default value is 128.</description>
  <config-property-name>spMaxEncryptBits</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise the maximum idle time before sending
  application level keep alive.
      It is measured in millisecond, and roundup to seconds.
  Default value is 0.</description>
  <config-property-name>spKeeyAlive</config-property-name>
  <config-property-type>java.lang.Long</config-property-type>
</config-property>
<config-property>
  <description>factory-wise how long adapter will wait for
  acknowledgement before adapter decides the
      connection already lost. Measurement in millisecond,
  and its default value is 10 seconds.

```

## <Default ? Font>Oracle Tuxedo JCA Adapter Inflow Transaction Guide

A value of 0 will disable the wait, and thus will not close the connection</description>

```
<config-property-name>spKeepAliveWait</config-property-name>
```

```
<config-property-type>java.lang.Long</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

<description>factory-wise valid Tuxedo service names in a comma-separated list. If not specified then

default import will be used and will grant all service request to remote Tuxedo domain</description>

```
<config-property-name>impResourceName</config-property-name>
```

```
<config-property-type>java.lang.String</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

<description>Exported resource, types of resource supported are EJB, POJO, MDB.</description>

```
<config-property-name>exportSpec</config-property-name>
```

```
<config-property-type>java.lang.String</config-property-type>
```

```
</config-property>
```

```
<connectionfactory-interface>javax.resource.cci.ConnectionFactory</connectionfactory-interface>
```

```
<connectionfactory-impl-class>com.oracle.tuxedo.adapter.cci.TuxedoConnectionFactory</connectionfactory-impl-class>
```

```
<connection-interface>javax.resource.cci.Connection</connection-interface>
```

```
<connection-impl-class>com.oracle.tuxedo.adapter.cci.TuxedoJCAConnection</connection-impl-class>
```



```

        </connection-definition>
<!--
        <transaction-support>NoTransaction</transaction-support>
        <transaction-support>LocalTransaction</transaction-support>
-->
        <transaction-support>XATransaction</transaction-support>
        <authentication-mechanism>

<authentication-mechanism-type>BasicPassword</authentication-mechanism-type>

<credential-interface>javax.resource.spi.security.PasswordCredential</credential-interface>

        </authentication-mechanism>
        <reauthentication-support>>false</reauthentication-support>
</outbound-resourceadapter>
<inbound-resourceadapter>
    <messageadapter>
        <messagelistener>

<messagelistener-type>com.oracle.tuxedo.adapter.intf.TuxedoMDBService</messagelistener-type>

        <activation-spec>

<activation-spec-class>com.oracle.tuxedo.adapter.spi.TuxedoActivationSpec</activation-spec-class>

            <required-config-property>
                <config-property-name>source</config-property-name>
            </required-config-property>
        </activation-spec>

```

```
        </messagelistener>
    </messageadapter>
</inbound-resourceadapter>
</resourceadapter>
</connector>
```

---

The inbound-resourceadapter element contains the interface class that must be implemented in the connector-based MDB and the activation specification class.

The fully qualified interface name is `com.oracle.tuxedo.adapter.intf.TuxedoMDBService`. The fully qualified activation specification is `com.oracle.tuxedo.adapter.spi.TuxedoActivationSpec`. *You must not change any one of these two values in the Oracle Tuxedo JCA Adapter `ra.xml` file.*

## Oracle Tuxedo GWTDOMAIN Gateway Configuration

You must also configure the Oracle Tuxedo GWTDOMAIN gateway to communicate with the Oracle Tuxedo JCA Adapter. [Listing 6](#) shows a Tuxedo /Domain configuration file example.

### Listing 6 Tuxedo /Domain Configuration File Example

---

```
#
*DM_RESOURCES
#
VERSION=U22
#
#
#
*DM_LOCAL_DOMAINS
#
```

```

# NOTE: Remove DYNAMIC_RAP line if you are not running with Tuxedo 11.1.1.2.0
#
"TDOM1"      GWGRP=GROUP3
              TYPE=TDOMAIN
              DOMAINID="TDOM1_ID"
              BLOCKTIME=60
              SECURITY=NONE
              DMTLOGDEV="C:\test\JCA\inflow_tx\tdom\DMTLOG"
              DYNAMIC_RAP="YES"

#
*DM_REMOTE_DOMAINS
#
#
JDOM        TYPE=TDOMAIN
              DOMAINID="JDOM_ID"

#
#
*DM_TDOMAIN
#
TDOM1      NWADDR="//localhost:12478"
JDOM       NWADDR="//localhost:10801"
#
#
*DM_LOCAL_SERVICES
#

```

```
#Exported
#
#
*DM_REMOTE_SERVICES
#
#Imported
#
TolowerMDB
EchoMDB
INFO
ACCOUNT
```

---

In this example, Oracle Tuxedo *imports* the services TolowerMDB, EchoMDB, INFO, and ACCOUNT; while the Oracle Tuxedo JCA Adapter *exports* them.

## Oracle Tuxedo JCA Adapter Deployment

On the WebSphere Integrated Solution Console, enter <https://localhost:9047/ibm/console/logon.jsp> (where 9047 is the port number your application server is listening on).

### Configure *dmconfig* File

Before deploying the Oracle Tuxedo JCA Adapter for WebSphere application server, a `dmconfig` configuration file must be created. [Listing 7](#) shows a `dmconfig` file example.

#### Listing 7 `dmconfig` File Example

---

```
<?xml version="1.0" encoding="UTF-8"?><TuxedoConnector>
  <LocalAccessPoint name="JDOM">
    <AccessPointId>JDOM_ID</AccessPointId>
```

```

    <NetworkAddress>//localhost:10801</NetworkAddress>
</LocalAccessPoint>
<RemoteAccessPoint name="TDOM1">
    <AccessPointId>TDOM1_ID</AccessPointId>
    <NetworkAddress>//localhost:12478</NetworkAddress>
</RemoteAccessPoint>
<SessionProfile name="profile_1">
    <BlockTime>60000</BlockTime>
    <ConnectionPolicy>ON_STARTUP</ConnectionPolicy>
</SessionProfile>
<Session name="session_1">
    <LocalAccessPointName>JDOM</LocalAccessPointName>
    <RemoteAccessPointName>TDOM1</RemoteAccessPointName>
    <ProfileName>profile_1</ProfileName>
</Session>
<Export name="ECHOMDB">
    <RemoteName>ECHO</RemoteName>
    <SessionName>session_1</SessionName>
    <Type>MDB</Type>
    <Source>eis/echo</Source>
</Export>
</TuxedoConnector>

```

---

## Resource Adapter Deployment Descriptor

You can either create the Resource Adapter Deployment Descriptor from scratch or modify an existing one. [Listing 8](#) shows a *Deploy Descriptor* example.

## Listing 8 Deploy Descriptor Example

---

```
<?xml version="1.0" encoding="UTF-8"?>
<connector xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/connector_1_5.xsd"
    version="1.5">
    <display-name>Tuxedo JCA Adapter</display-name>
    <vendor-name>Oracle</vendor-name>
    <eis-type>Tuxedo</eis-type>
    <resourceadapter-version>11gR1(11.1.1.2.1)</resourceadapter-version>
    <license>
        <description>Tuxedo SALT license</description>
        <license-required>>false</license-required>
    </license>
    <resourceadapter>

<resourceadapter-class>com.oracle.tuxedo.adapter.TuxedoResourceAdapter</re
sourceadapter-class>

    <!--
        The following is the list of properties name can be configured as
        adapter-wise configuration.

        traceLevel - java.lang.String - a numerical value
        xaAffinity - java.lang.String - transaction affinity to a remote
        domain, "true" or "false", default to true
        keyFileName - java.lang.String - encryption key file name
        throwFailureReplyException - java.lang.Boolean - default to ture
        appManagedLocalTxTimeout - java.lang.Integer - Application managed
        transaction or AUTOTRAN timeout
```

defaults to 300 seconds

fieldTable16Class - java.lang.String - a comma-separated list of fully qualified FML classes

fieldTable32class - java.lang.String - a comma-separated list of fully qualified FML32 classes

viewFile16Class - java.lang.String - a comma-separated list of fully qualified VIEW classes

viewFile32Class - java.lang.String - a comma-separated list of fully qualified VIEW32 classes

tpusrFile - java.lang.String - path name to the TPUSR file

remoteMBEncoding - java.lang.String - remote Tuxedo encoding name for multi-byte language

mBEncodingMapFile - java.lang.String - path name to Multi-byte encoding name mapping

autoTran - java.lang.Boolean- enable adapter-wise AUTOTRAN, default to false

-->

<config-property>

<config-property-name>traceLevel</config-property-name>

<config-property-type>java.lang.String</config-property-type>

<config-property-value>2000000</config-property-value>

</config-property>

<!--

<config-property>

<config-property-name>xaAffinity</config-property-name>

<config-property-type>java.lang.String</config-property-type>

<config-property-value>>true</config-property-value>

</config-property>

-->

```
<config-property>
  <config-property-name>dmconfig</config-property-name>
  <config-property-type>java.lang.String</config-property-type>

<config-property-value>C:\test\JCA\inflow_tx\adapter\dmconfig.xml</config-
property-value>
</config-property>
<config-property>
  <config-property-name>keyFileName</config-property-name>
  <config-property-type>java.lang.String</config-property-type>

<config-property-value>C:\test\JCA\inflow_tx\adapter\foo.key</config-prope
rty-value>
</config-property>
<config-property>
  <config-property-name>debugAdapter</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
  <config-property-value>true</config-property-value>
</config-property>
<config-property>
  <config-property-name>debugJatmi</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
  <config-property-value>true</config-property-value>
</config-property>
<config-property>
  <config-property-name>debugConfig</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
  <config-property-value>true</config-property-value>
```



```

</config-property>
<config-property>
  <config-property-name>debugSession</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
  <config-property-value>true</config-property-value>
</config-property>
<config-property>
  <config-property-name>debugXa</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
  <config-property-value>true</config-property-value>
</config-property>
<config-property>
  <config-property-name>debugPdu</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
  <config-property-value>true</config-property-value>
</config-property>
<config-property>
  <config-property-name>debugSec</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
  <config-property-value>true</config-property-value>
</config-property>
<!--
-->
  <outbound-resourceadapter>
    <connection-definition>

<managedconnectionfactory-class>com.oracle.tuxedo.adapter.spi.TuxedoManage
dConnectionFactory</managedconnectionfactory-class>

```

```
<!--  
    The following is the list of properties that you can use to  
    to configure connection pool or connection factory.  
    User must either configure localAccessPointSpec or  
    connectionFactoryName if transaction is used.  
    These property described here is serving as template, user should not  
    configure them here, instead user should configure them either  
    through WebSphere console  
    or weblogic-ra.xml side file.  
-->  
  
<config-property>  
    <description>factory-wise AUTOTRAN setting, default to false,  
    overrides adapter-wise setting</description>  
    <config-property-name>autoTran</config-property-name>  
    <config-property-type>java.lang.Boolean</config-property-type>  
</config-property>  
  
<config-property>  
    <description>factory-wise Failure Reply Exception setting, default  
    to true, overrides adapter-wise setting</description>  
  
<config-property-name>throwFailureReplyException</config-property-name>  
    <config-property-type>java.lang.Boolean</config-property-type>  
</config-property>  
  
<config-property>  
    <description>factory-wise application managed transaction or  
    AUTOTRAN time out, overrides adapter-wise setting</description>  
  
<config-property-name>appManagedLocalTxTimeout</config-property-name>  
    <config-property-type>java.lang.Integer</config-property-type>
```

```

</config-property>
<config-property>
  <description>connection factory or pool name, this is required if
XA or local application managed
      transaction is required</description>
  <config-property-name>connectionFactoryName</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>application password in either clear text or cipher
text using com.oracle.tuxedo.tools.EncryptPassword tool</description>
  <config-property-name>applicationPassword</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>local access point specification of the format
//hostname:port/domainId=DOMAINID</description>
  <config-property-name>localAccessPointSpec</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise SSL to configure whether mutual
authentication is required, default to false</description>
<config-property-name>mutualAuthenticationRequired</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
</config-property>
<config-property>

```

```
    <description>factory-wise SSL for configuring identity key store  
file name, must be configured if SSL is desired</description>
```

```
<config-property-name>identityKeyStoreFileName</config-property-name>
```

```
    <config-property-type>java.lang.String</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

```
    <description>factory-wise SSL setting for private key alias used  
in the key store, must be configured if SSL is desired</description>
```

```
    <config-property-name>privateKeyAlias</config-property-name>
```

```
    <config-property-type>java.lang.String</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

```
    <description>factory-wise trusted key store file name, must be  
configured if SSL is desired</description>
```

```
<config-property-name>trustedKeyStoreFileName</config-property-name>
```

```
    <config-property-type>java.lang.String</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

```
    <description>factory-wise password for identityKeyStore in clear  
text</description>
```

```
<config-property-name>identityKeyStorePassPhrase</config-property-name>
```

```
    <config-property-type>java.lang.String</config-property-type>
```

```
</config-property>
```

```
<config-property>
```

```
    <description>factory-wise password for privateKeyAlias in clear  
text</description>
```

```

<config-property-name>privateKeyAliasPassPhrase</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise password for trustedKeyStore in clear
text</description>

<config-property-name>trustedKeyStorePassPhrase</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise RemoteAccessPoint specification of the
format //hostname:port/domainId=DOMAINID</description>
    <config-property-name>remoteAccessPointSpec</config-property-name>
    <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
    <description>factory-wise allow anonymous access to Tuxedo, default
to false</description>
    <config-property-name>rapAllowAnonymous</config-property-name>
    <config-property-type>java.lang.Boolean</config-property-type>
</config-property>
<config-property>
    <description>factory-wise application key value for anonymous user,
default to -1</description>

<config-property-name>rapDefaultApplicationKey</config-property-name>
    <config-property-type>java.lang.Integer</config-property-type>

```

## <Default ? Font>Oracle Tuxedo JCA Adapter Inflow Transaction Guide

```
</config-property>
<config-property>
  <description>factory-wise application key fully qualified class
name for AppKey generator</description>
<config-property-name>rapApplicationKeyClass</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise custom application key
parameter</description>
<config-property-name>rapApplicationKeyClassParam</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise session profile block timeout value,
default to 60000 milliseconds</description>
  <config-property-name>spBlockTime</config-property-name>
  <config-property-type>java.lang.Integer</config-property-type>
</config-property>
<config-property>
  <description>factory-wise whether allows interoperate with 6.5
Tuxedo Domain, default to false</description>
  <config-property-name>spInteroperate</config-property-name>
  <config-property-type>java.lang.Boolean</config-property-type>
</config-property>
<config-property>
```

```

        <description>factory-wise security setting, legal values: NONE,
DM_PW, APP_PW</description>
        <config-property-name>spSecurity</config-property-name>
        <config-property-type>java.lang.String</config-property-type>
    </config-property>
</config-property>
        <description>factory-wise credential propagation policy, either
LOCAL or GLOBAL</description>
        <config-property-name>spCredentialPolicy</config-property-name>
        <config-property-type>java.lang.String</config-property-type>
    </config-property>
</config-property>
        <description>factory-wise number of seconds that session waits
between automatic connection establishment,
                default to 60 seconds. A value of 0 disabled connection
retry</description>
        <config-property-name>spRetryInterval</config-property-name>
        <config-property-type>java.lang.Long</config-property-type>
    </config-property>
</config-property>
        <description>factory-wise maximum number of times adapter will try
to establish a session connection to
                remote Tuxedo access point. Default value is
Long.MAX_VALUE.</description>
        <config-property-name>spMaxRetries</config-property-name>
        <config-property-type>java.lang.Long</config-property-type>
    </config-property>
</config-property>

```

```
<description>factory-wise compression threshold, default to
Integer.MAX_VALUE</description>
  <config-property-name>spCompressionLimit</config-property-name>
  <config-property-type>java.lang.Integer</config-property-type>
</config-property>
<config-property>
  <description>factory-wise minimum encryption strength requirement,
legal values are 0, 40, 56, 128, 256.
      Default value is 0.</description>
  <config-property-name>spMinEncryptBits</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise maximum encryption strength requirement,
legal values are 0, 40, 56, 128, 256.
      Default value is 128.</description>
  <config-property-name>spMaxEncryptBits</config-property-name>
  <config-property-type>java.lang.String</config-property-type>
</config-property>
<config-property>
  <description>factory-wise the maximum idle time before sending
application level keep alive.
      It is measured in millisecond, and roundup to seconds.
Default value is 0.</description>
  <config-property-name>spKeepAlive</config-property-name>
  <config-property-type>java.lang.Long</config-property-type>
</config-property>
<config-property>
```



<description>factory-wise how long adapter will wait for acknowledgement before adapter decides the

connection already lost. Measurement in millisecond, and its default value is 10 seconds.

A value of 0 will disable the wait, and thus will not close the connection</description>

<config-property-name>spKeepAliveWait</config-property-name>

<config-property-type>java.lang.Long</config-property-type>

</config-property>

<config-property>

<description>factory-wise valid Tuxedo service names in a comma-separated list. If not specified then

default import will be used and will grant all service request to remote Tuxedo domain</description>

<config-property-name>impResourceName</config-property-name>

<config-property-type>java.lang.String</config-property-type>

</config-property>

<config-property>

<description>Exported resources. Types of resource supported are</description>

<config-property-name>exportSpec</config-property-name>

<config-property-type>java.lang.String</config-property-type>

</config-property>

<connectionfactory-interface>javax.resource.cci.ConnectionFactory</connectionfactory-interface>

<connectionfactory-impl-class>com.oracle.tuxedo.adapter.cci.TuxedoConnectionFactory</connectionfactory-impl-class>

## <Default ? Font>Oracle Tuxedo JCA Adapter Inflow Transaction Guide

```
<connection-interface>javax.resource.cci.Connection</connection-interface>

<connection-impl-class>com.oracle.tuxedo.adapter.cci.TuxedoJCAConnection</
connection-impl-class>

    </connection-definition>
<!--
    <transaction-support>NoTransaction</transaction-support>
    <transaction-support>LocalTransaction</transaction-support>
-->
    <transaction-support>XATransaction</transaction-support>
    <authentication-mechanism>

<authentication-mechanism-type>BasicPassword</authentication-mechanism-ty
pe>

<credential-interface>javax.resource.spi.security.PasswordCredential</cred
ential-interface>
    </authentication-mechanism>
    <reauthentication-support>>false</reauthentication-support>
</outbound-resourceadapter>
<inbound-resourceadapter>
    <messageadapter>
        <messagelistener>

<messagelistener-type>com.oracle.tuxedo.adapter.intf.TuxedoMDBService</mes
sagelistener-type>
    <activation-spec>
```

```
<activation-spec-class>com.oracle.tuxedo.adapter.spi.TuxedoActivationSpec</activation-spec-class>
    <required-config-property>
        <config-property-name>source</config-property-name>
    </required-config-property>
</activation-spec>
</message-listener>
</message-adapter>
</inbound-resource-adapter>
</resource-adapter>
</connector>
```

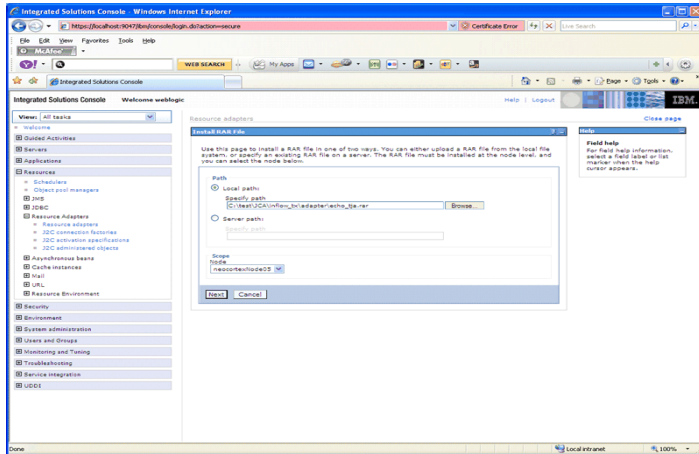
---

## Deploy Oracle Tuxedo JCA Adapter

After jarring the resource adapter with deployment descriptor to create a Resource Archive, you can deploy it to a WebSphere application server.

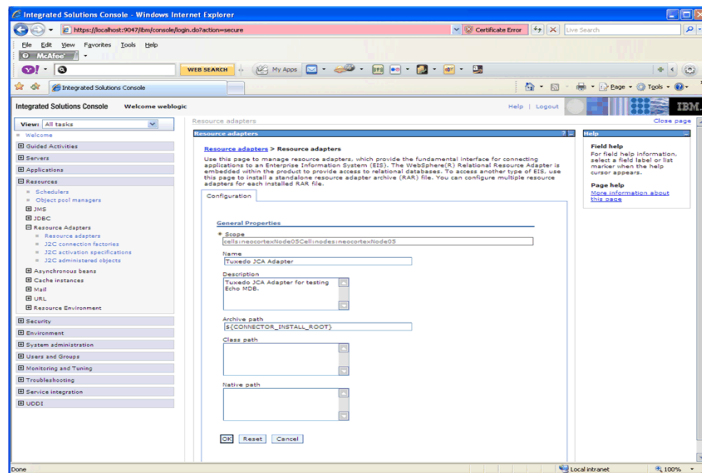
After logging in to the WebSphere Integrated Solution Console select **Resource** from the left window pane; select **Resource Adapters** as shown in [Figure 2](#). The **Resource adapter** window appears. Click **Browse** to find your RAR file.

Figure 2 WebSphere Integrated Solution Console



Click Next; the **General Properties** page appears as shown in Figure 3. Enter the appropriate description in the **Description** text entry box.

Figure 3 General Properties Page

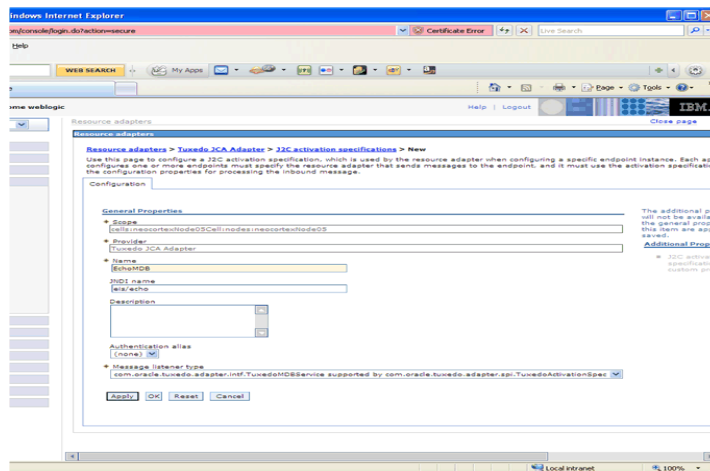


Click **OK**, then click **Save**.

## Configure Activation Spec

From **Resource adapter > Tuxedo JCA Adapter**, select **J2C activation specification** under **Additional Properties**; the **J2C activation specification** page appears as shown in [Figure 4](#). Select **New**; enter a name for the activation specification and its JNDI name (this example uses EchoMDB as name and eis/echo as its JNDI name. This JNDI name is the JNDI name EchoMDB uses.

Figure 4 J2C Activation Specification Page



Click **OK** to complete specification.

## Configuring MDB Using WebSphere Integrated Console

Start your WebSphere application server and log in to WebSphere using the Integrated Solution Console. The console port number usually is 904X where “X” can be any digit.

**Note:** You can find a logs/server1/SystemOut.log. Look for “TCP Channel TCP\_3 is listening.”

## Deploy MDB To WebSphere

### Configure a Shared Library

On the left pane of console select **Environment**; this expands the menu item with a sub-menu. Select **Shared Library**; the **Shared Library** screen appears.

Click **New**; the configuration screen appears. Fill in **Name** with any name you like. For this example, enter **EchoMDBenv** in the **Name** text entry box. In the Classpath window enter the full path name of the following two JAR files.

- com.bea.core.jatmi\_1.3.2.0.jar
- com.oracle.tuxedo.adapter\_1.2.1.0.jar

Use the Enter key as a separator then click **Save**.

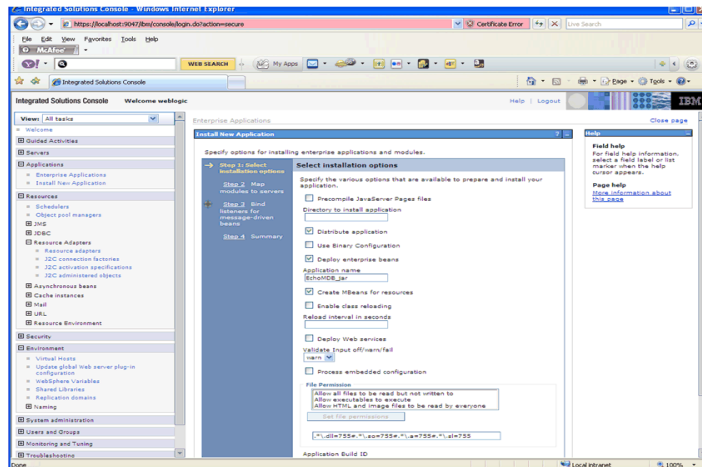
## Install MDB

On left pane of the console, select **Applications**, then select **Install New Application**. The **Enterprise Application** menu appears.

In **Path to the new application** select **Local file system**. Use **Browse** to select the **EchoMDB.jar** file.

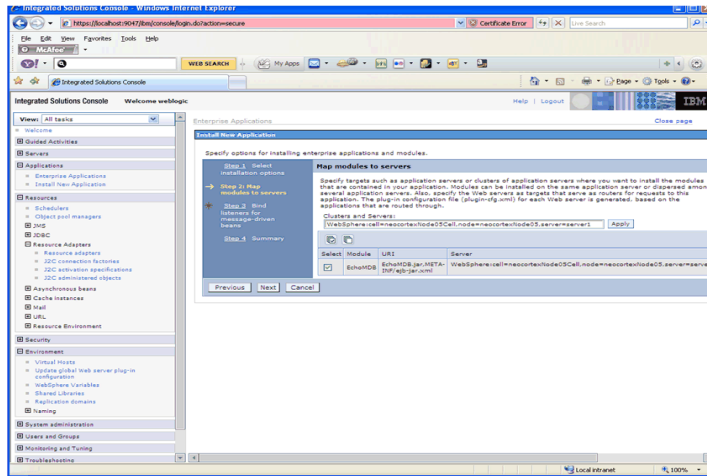
In **How do you want to install the application**, select **Show me all installation options and parameters**. Click **Next**; the **Select installation options** page appears as shown in **Figure 5**. Select **Deploy enterprise beans**, then click **Next**.

**Figure 5 Select installation Options Page**



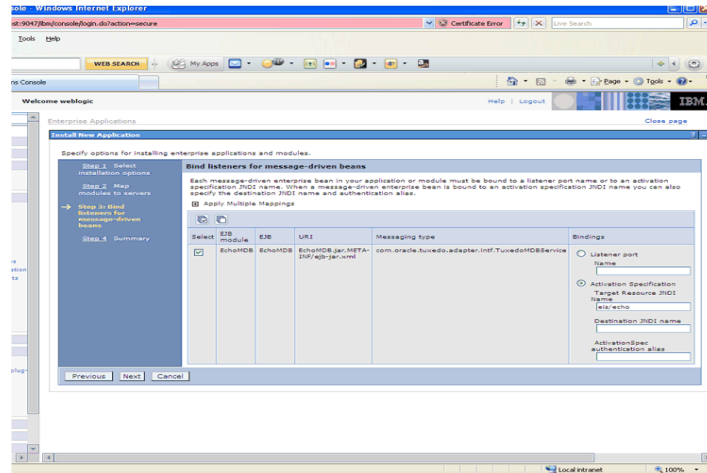
In the “Step 2: Map modules to servers” select the server where you want your Echo MDB be available. Place a check mark to “EchoMDB” then click **Apply**. Click **Next**.

Figure 6 Select EchoMDB Module



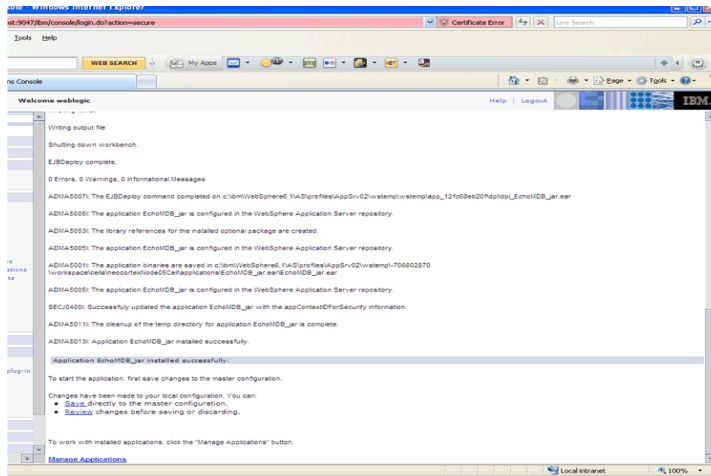
In the “Step 3: Bind listeners for message-driven beans” select **Activation Specification** as shown in Figure 7 then enter the JNDI name for this MDB (in this case, enter `eis/echo`).

Figure 7 Activation Specification



Click **Next**: the Summary page appears as shown in Figure 8. Click **Finish**. The application server compiles and deploy deploys the MDB.

Figure 8 Summary Page



Click Save.

## Activate MDB

From the left pane of the console select **Applications** and then select **Enterprise Applications**. Select **EchoMDB.jar** and click **Start**. **Echo™ EJB** is activated.

# Oracle Tuxedo Application Domain

## Oracle Tuxedo Configuration

Listing 1 shows the Oracle Tuxedo UBBCONFIG file used in this example.

### Listing 1 Oracle Tuxedo UBBCONFIG File Example

```
#
#Ubbconfig domain1
#

*RESOURCES
IPCKEY                    51301
```



```
MASTER                site1
MAXACCESSERS100
MAXSERVERS            25
MAXSERVICES           50
MODEL                  SHM
LDBAL                  N
BLOCKTIME 1
SCANUNIT              5
SECURITY              NONE

*MACHINES
DEFAULT:
    APPDIR="C:\test\JCA\inflow_tx/tdom1"
    TUXCONFIG="C:\test\JCA\inflow_tx/tdom1/TUXCONFIG"
    TUXDIR="c:\tuxedo\tux11g"

"NEOCORTEX"LMID=site1
    MAXWSCLIENTS=2

*GROUPS
GROUP3 LMID=site1 GRPNO=3OPENINFO=NONE
GROUP2 LMID=site1 GRPNO=2OPENINFO=NONE
GROUP1 LMID=site1 GRPNO=1 TMSNAME=TMS TMSCOUNT=3
#GROUP1 LMID=site1 GRPNO=1

*SERVERS
```

## <Default ? Font>Oracle Tuxedo JCA Adapter Inflow Transaction Guide

DEFAULT:

CLOPT="-A" RESTART=Y MAXGEN=5

DMADM SRVGRP=GROUP2SRVID=1

GWADM SRVGRP=GROUP3SRVID=2

GWTDOMAINSRVGRP=GROUP3SRVID=3

ENVFILE="C:\test\JCA\inflow\_tx\tdom1\gwt.env"

simperv SRVGRP=GROUP1SRVID=20

\*SERVICES

TOUPPER\_STR

Tuxedo /Domain Configuration

The following is the /Domain configuration for this sample.

#

\*DM\_RESOURCES

#

VERSION=U22

#

#

\*DM\_LOCAL\_DOMAINS

#

# NOTE: Remove DYNAMIC\_RAP line if you are not running with Tuxedo 11.1.1.2.0

#

"TDOM1" GWGRP=GROUP3

```
TYPE=TDOMAIN
DOMAINID="TDOM1_ID"
BLOCKTIME=60
SECURITY=NONE
DMTLOGDEV="C:\test\JCA\inflow_tx/tdom1/DMTLOG"
DYNAMIC_RAP="YES"

#
*DM_REMOTE_DOMAINS
#
#
JDOM TYPE=TDOMAIN
      DOMAINID="JDOM_ID"

#
#
*DM_TDOMAIN
#
TDOM1 NWADDR="//localhost:12478"
JDOM  NWADDR="//localhost:10801"
#
#
*DM_LOCAL_SERVICES
#
#Exported
#
TOUPPER_STR
```

```
#
*DM_REMOTE_SERVICES
#
#Imported
#
ECHO
```

---

## Programming MDB For Oracle Tuxedo JCA Adapter

### Interface TuxedoMDBService

The Tuxedo JCA Adapter provides an **EJB MDB** interface that you must implement in your **EJB** application code.

**Note:** The **MDB** interface is similar to the existing **EJB** supported by *Oracle Tuxedo JCA Adapter*; however, they are not the same.

[Listing 2](#) shows the interface listing.

#### Listing 2 Interface Listing

---

```
package com.oracle.tuxedo.adapter.intf;
import weblogic.wtc.jatmi.Reply;
import com.oracle.tuxedo.adapter.tdom.TPServiceInformation;
import com.oracle.tuxedo.adapter.TuxedoReplyException;
public interface TuxedoMDBService {
public Reply service(TPServiceInformation service) throws
TuxedoReplyException;
}
```

---

This is different from a *JMS*-based **MDB**, it uses the `service()` interface instead of the `onMessage()` interface. [Listing 3](#) shows an **MDB** code example that implements the “*Tolower*” service for an Oracle Tuxedo client.

### Listing 3 MDB Code Example

---

```
package ejbs;

import com.oracle.tuxedo.adapter.TuxedoReplyException;
import com.oracle.tuxedo.adapter.intf.TuxedoMDBService;
import com.oracle.tuxedo.adapter.tdom.TPServiceInformation;
import javax.ejb.MessageDrivenBean;
import javax.ejb.MessageDrivenContext;
import javax.jms.Message;
import weblogic.wtc.jatmi.Reply;
import weblogic.wtc.jatmi.TypedString;

public class TolowerMDBBeanBean
    implements MessageDrivenBean, TuxedoMDBService
{

    public TolowerMDBBeanBean()
    {

    }

    public MessageDrivenContext getMessageDrivenContext()
    {
        return fMessageDrivenCtx;
    }
}
```

```
public void setMessageDrivenContext(MessageDrivenContext ctx)
{
    fMessageDrivenCtx = ctx;
}

public void ejbCreate()
{
}

public void onMessage(Message message)
{
}

public void ejbRemove()
{
}

public Reply service(TPServiceInformation mydata)
    throws TuxedoReplyException
{
    TypedString data = (TypedString)mydata.getServiceData();
    String lowered = data.toString().toLowerCase();
    TypedString return_data = new TypedString(lowered);
    mydata.setReplyBuffer(return_data);
    return mydata;
}

private static final long serialVersionUID = 1L;
```

```
private MessageDrivenContext fMessageDrivenCtx;
}
```

---

## Creating an Inbound Connector-Based MDB Using IBM ASTK

This procedure creates a Connector-Based EJB 2.1 MDB using WebSphere ASTK 6.1. The simple EJB MDB echoes the input string back to the Oracle Tuxedo Client. The name of the project is called EchoMDB.

### Use J2EE Perspective

If you are not already in “J2EE” perspective, do the following to change to “J2EE” perspective.

From menu *Window* select **Open Perspective**, and then select **J2EE**.

### Create EJB Project

From menu “**File**” select **New**, then select **Project...** Expand **EJB** by clicking it, and then highlight **EJB Project**. Click **Next**.

In **EJB Project** menu fill in **Project Name** with “**EchoMDB**”. Click **Next**. The “**Select Project Facets**” menu will be shown.

In “**Select Project Facets**” menu, make sure “**EJB Module**” version is “**2.1**”, “**Java**” version is “**5.0**”, and “**WebSphere EJB (Extended)**” version is “**6.1**”, and make sure these three are selected. Click on “**Next**”.

In “**EJB Module**” menu you uncheck “**create an EJB Client JAR module to hold the client interface and classes**” since inbound EJB is invoked by Oracle Tuxedo JCA Adapter so it is not needed. Click on “**Finish**”.

### Setup Build Environment

Right click on project **EchoMDB** in the **Project Explorer**. Select **Properties** from the context menu, the **Properties for EchoMDB** window appears as shown in [Figure 9](#).

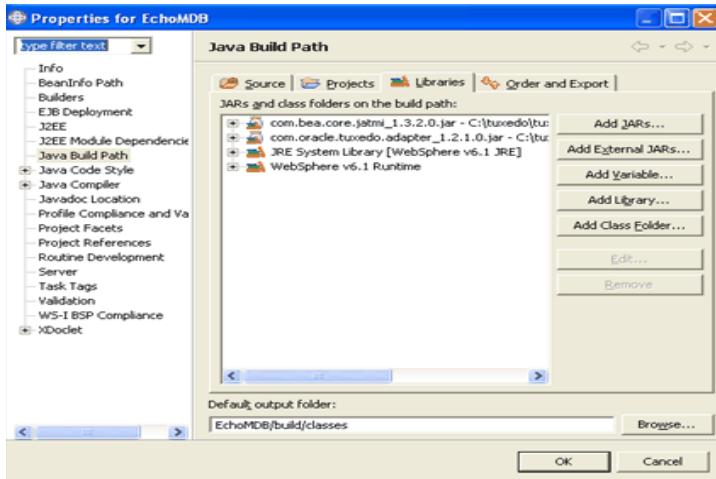
Select **Add External JARs...** from the “**Java Build Path**”. Add the following two Jar files from Oracle Tuxedo JCA Adapter RAR file. (If you have not unzipped the RAR file, do so now.)

```
com.bea.core.jatmi_1.3.2.0.jar
```

```
com.oracle.tuxedo.adapter_1.2.1.0.jar
```

Click **OK**.

**Figure 9 EchoMDB Properties Window**



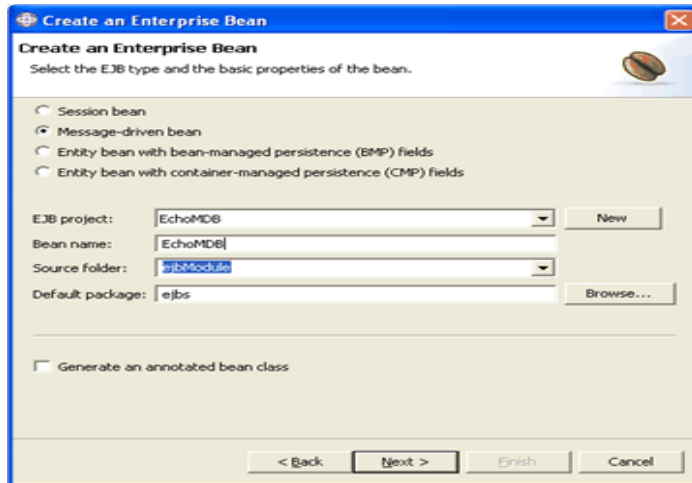
## Create Message-Driven Bean

On the left Window pane under the **Project Explorer**, expand the newly create MDB project EchoMDB. Right click **EchoMDB**, select **New**, and then select **Other**. Select **Enterprise Bean** and click **Next**. The **Create an Enterprise Bean** popup window appears as shown in [Figure 10](#).

In the popup window select **Message-driven bean**. Enter the Bean name with value **EchoMDB**.

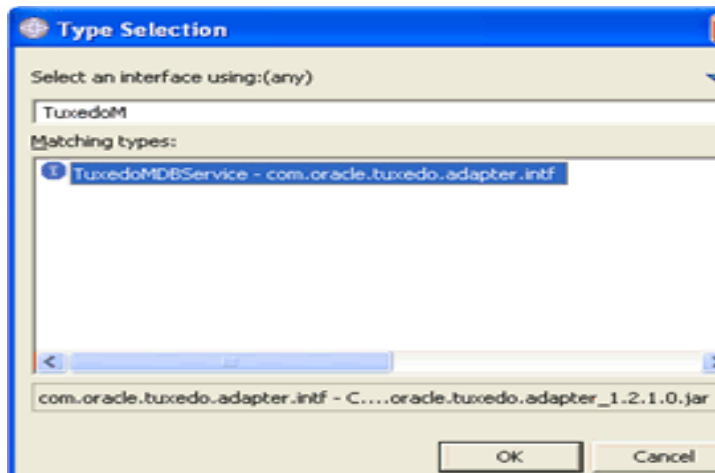


Figure 10 Create an Enterprise Bean Window



Click **Next**. The “Message Driven Bean type” popup window appears as shown in [Figure 11](#). Select **Other Type** and then click **Browse**. Enter **TuxedoMDBService** and select from the list shown in [Figure 11](#), then click **OK**.

Figure 11 Message Driven Bean Type Window

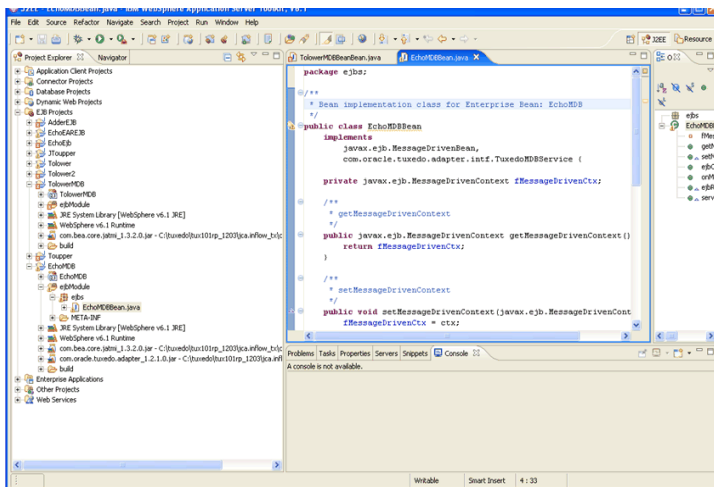


The **Message Driven Bean** type popup window appears. Click **Finish**.

## Modify EchoBean

Expand **ejbModule** in the left window pane until you see **EchoMDBBean.java**. EchoMDBBean.java must be modified to perform the ECHO service. Double click **EchoMDBBean.java** and the edit window pane with default editor appears as shown in [Figure 12](#).

Figure 12 Edit Window



Add the following lines shown in [Listing 1](#) at the top of the class file.

Listing 1 Add New Lines

```
import webllogic.wtc.jatmi.Reply;  
import webllogic.wtc.jatmi.TPException;  
import webllogic.wtc.jatmi.TPReplyException;  
import webllogic.wtc.jatmi.TypedString;  
  
import com.oracle.tuxedo.adapter.TuxedoReplyException;  
import com.oracle.tuxedo.adapter.intf.TuxedoMDBService;  
import com.oracle.tuxedo.adapter.tdom.TPServiceInformation;
```

Edit the method `service()` at the end of the class file as shown in [Listing 2](#).

### **Listing 2** `service()`

---

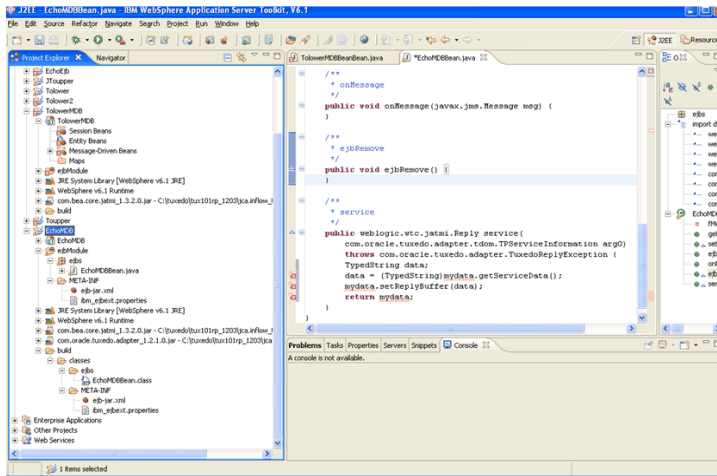
```
public weblogic.wtc.jatmi.Reply service(  
    com.oracle.tuxedo.adapter.tdom.TPServiceInformation mydata)  
    throws com.oracle.tuxedo.adapter.TuxedoReplyException {  
    TypedString data;  
    data = (TypedString)mydata.getServiceData();  
    mydata.setReplyBuffer(data);  
    return mydata;  
}
```

---

## **Build**

Right click project **EchoMDB** in the Project Explorer, and then select **Deploy** as shown in [Figure 13](#). This compiles it into class in the build directory.

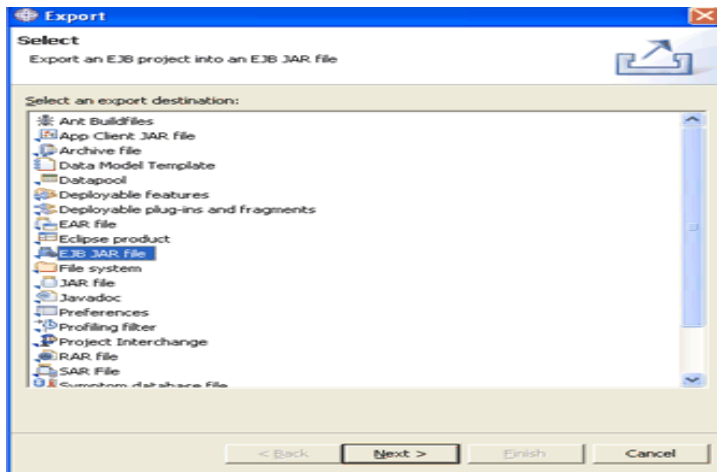
Figure 13 Compile



## Create EJB JAR File

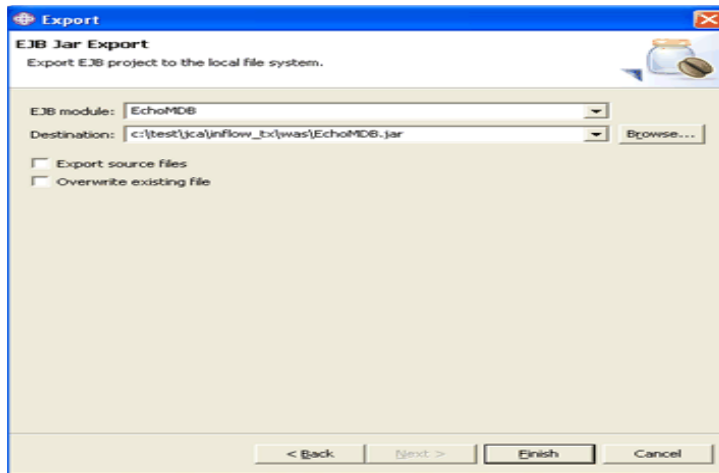
Right click the project **EchoMDB** in the Project Explorer and select **Export**. The **Export** menu popup appears as shown in [Figure 14](#).

Figure 14 Export Popup Window



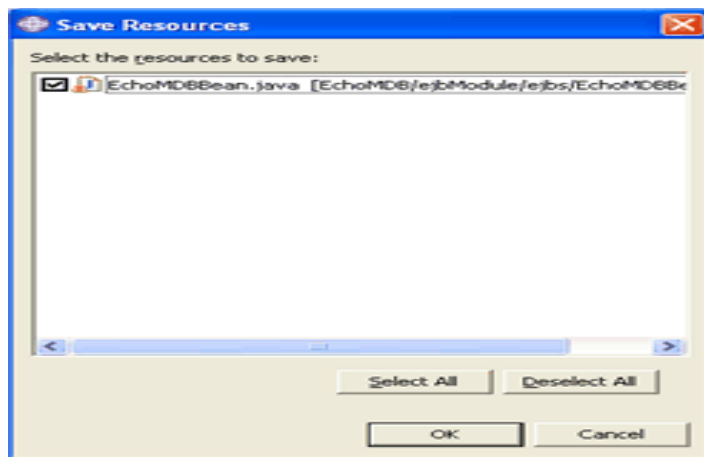
In the Export popup window select **EJB JAR file**. Click **Next**. The **EJB Jar Export** popup window appears as shown in [Figure 15](#). Select **EchoMDB** from the drop down menu, and enter the complete path of the jar file name in the **Destination:** text field. Click **Finish**.

**Figure 15 EJB Jar Export Popup Window**



The **Save Resources** popup window appears as shown in [Figure 16](#) click “OK”.

**Figure 16 Save Resources Popup Window**



For *Oracle Tuxedo JCA Adapter* dispatching-based MDB, you must add `activation-config-property` to its `ejb-jar.xml` file using one of two ways.

1. The first method is to unzip the jar file. After the jar file is unzipped, modify the `META-INF/ejb-jar.xml`, and then re-jar the bean jar file. [Listing 1](#) shows an example `ejb-jar.xml` file suitable to this type of MDB.

### Listing 1 `ejb-jar.xml` File Example

---

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar id="ejb-jar_ID" version="2.1"
xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/ejb-jar_2_1.xsd">
  <display-name>EchoMDB</display-name>
  <enterprise-beans>
    <!-- message driven descriptor -->
    <message-driven id="EchoMDB">
      <ejb-name>EchoMDB</ejb-name>
      <ejb-class>ejbs.EchoMDBBean</ejb-class>
      <!-- message listener interface -->
    </message-driven>
  </enterprise-beans>
  <messaging-type>com.oracle.tuxedo.adapter.intf.TuxedoMDBService</messaging-type>
  <transaction-type>Container</transaction-type>
  <!-- the values for the Activation Spec JavaBean -->
  <activation-config>
    <activation-config-property>
      <activation-config-property-name>source</activation-config-property-name>
```

```

<activation-config-property-value>eis/echo</activation-config-property-val
ue>

        </activation-config-property>
    </activation-config>
</message-driven>
</enterprise-beans>
</ejb-jar>

```

---

Where `eis/echo` is the JNDI name of `EchoMDB`.

2. Similarly, the second method is to modify `ejb-jar.xml` file directly to add `activation-config-property` using ASTK before the MDB is being deployed and exported.

## Oracle Tuxedo Transactional Client Source Code

[Listing 2](#) shows the simple Oracle Tuxedo native client that accesses the `ECHO` service imported from WebSphere application server.

### Listing 2 ECHO Service Imported from WebSphere Application Server

---

```

#include <stdio.h>
#include "atmi.h"
main(int argc, char *argv[])
{
    char *sendbuf, *rcvbuf;
    long sendlen, rcvlen;
    int ret;

    if (tpinit((TPINIT *)NULL) == -1) {

```

```
(void)fprintf(stderr, "Tpinit failed\n");
exit(1);
}
sendlen = strlen(argv[1]);
if ((sendbuf = (char *)tpalloc("STRING", NULL, sendlen + 1)) == NULL) {
    (void)fprintf(stderr, "Error allocating send buffer\n");
    tpterm();
    exit(2);
}
if ((rcvbuf = (char *)tpalloc("STRING", NULL, sendlen + 1)) == NULL) {
    (void)fprintf(stderr, "Error allocating receive buffer\n");
    tpfree(sendbuf);
    tpterm();
    exit(2);
}
(void)strcpy(sendbuf, argv[2]);
tpbegin(45, 0);
ret = tpcall("ECHO", (char *)sendbuf, 0, (char **)&rcvbuf, &rcvlen,
(long)0);
if (ret == -1) {
    tpabort(0);
    tpfree(sendbuf);
    tpfree(rcvbuf);
    tpterm();
    exit(1);
}
userlog("Return string: %s", rcvbuf);
tpcommit(0);
```



```
tpfree(sendbuf);  
tpfree(rcvbuf);  
tpterm();  
return(0);  
}
```

---

