



BEA WebLogic Integration™

WebLogic Integration Solutions Best Practices FAQ

Copyright

Copyright © 2004 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

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

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

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

Trademarks or Service Marks

BEA, Jolt, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Liquid Data for WebLogic, BEA Manager, BEA WebLogic Commerce Server, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Enterprise Security, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic Personalization Server, BEA WebLogic Platform, BEA WebLogic Portal, BEA WebLogic Server, BEA WebLogic Workshop and How Business Becomes E-Business are trademarks of BEA Systems, Inc.

All other trademarks are the property of their respective companies.

Contents

1. Performance Tips

Design-Time Performance Issues	1-1
Which types of processes are faster—stateful or stateless?	1-2
Which controls are faster—process or service?	1-2
Which callback methods are faster—process control callbacks or message broker subscriptions?	1-2
How can I get better performance from parallel nodes in my business processes? ..	1-2
Why do my JMS queues show so many pending messages?	1-2
How can I get rid of pending JMS timer messages?	1-2
How can I increase the transaction timeout period?	1-3
Run-Time Tuning Issues	1-3
What flags should I use when I start WebLogic Server to maximize performance? ..	1-4
How do I tune WebLogic Integration Applications?	1-4
Can I configure the value of max-beans-in-free-pool for the JMS event generator? ..	1-5
Which process tracking levels should I use to optimize performance?	1-5
Do I need to run the Archiver if the process tracking levels are set to none?	1-5
How do I turn off informational Web services messages?	1-6
When should my WebLogic Integration application use the document store?	1-6
Which is the best persistence model for JMS—file-based persistence or JDBC? ...	1-6
What should I watch for when monitoring a WebLogic Integration application running under load?	1-7
How can I reduce the number of transactions timing out?	1-8

Why does my application seem to leak memory? 1-9

2. WebLogic Integration Application Recovery

Recovery Checklist 2-1

What should I check to make sure my WebLogic Integration application is configured properly for recovery? 2-1

How long should I wait to start recovery? 2-3

Performance Tips

The following sections provide answers to questions about designing and tuning high-performance WebLogic Integration applications. It contains the following topics:

- [Design-Time Performance Issues](#)
- [Run-Time Tuning Issues](#)

Design-Time Performance Issues

This section answers the following questions:

- [Which types of processes are faster—stateful or stateless?](#)
- [Which controls are faster—process or service?](#)
- [Which callback methods are faster—process control callbacks or message broker subscriptions?](#)
- [How can I get better performance from parallel nodes in my business processes?](#)
- [Why do my JMS queues show so many pending messages?](#)
- [How can I increase the transaction timeout period?](#)
- [How can I increase the transaction timeout period?](#)

Which types of processes are faster—stateful or stateless?

Stateless processes are significantly faster than stateful processes.

Which controls are faster—process or service?

Process controls are faster than service controls.

Which callback methods are faster—process control callbacks or message broker subscriptions?

You get approximately the same performance when starting a process using the process control or message broker, but it is significantly faster to receive a process control callback than a message broker subscription. The message broker subscription filter mechanism uses a database to map the filter values to process instances. Process control callbacks are routed directly to the process instances.

How can I get better performance from parallel nodes in my business processes?

By default, the branches of a parallel node are transactionally isolated. You can override this behavior by setting the `continueTransaction` property in the source for each parallel element as follows:

```
<parallel continueTransaction="true">
```

For more information, see “Continue Transaction Attribute on Parallel Nodes” in [Business Process](#) in *WebLogic Integration 8.1 Release Notes*.

Why do my JMS queues show so many pending messages?

Timer messages (messages set with a delivery time in the future) show up as “pending messages” in the WebLogic Server console.

How can I get rid of pending JMS timer messages?

For stateful processes, turn off maximum conversation lifetime if you’re not using it:

```
* @jws:conversation-lifetime max-age="0s"
```

For more information on the conversation lifetime feature, see “@jws:conversation-lifetime Annotation” in [Java Web Service Annotations](#) in the “WebLogic Workshop Reference” in WebLogic Workshop Help.

How can I increase the transaction timeout period?

The default timeout period for a transaction is 300 seconds (5 minutes). You can increase the amount of elapsed time before a transaction times out by using one of the following approaches:

- Override the transaction timeout value using the procedure described in "Handling Transaction Time-out" in [Default Transactional Behavior in WebLogic Workshop](#) in the *WebLogic Workshop Help*.
- Increase the `trans-timeout-seconds` value for the AsyncDispatcher MDB in `weblogic-ejb-jar.xml`. For more information, see “trans-timeout-seconds” in “2.0 weblogic-ejb-jar.xml in WebLogic Server 8.1” in the [weblogic-ejb-jar.xml Deployment Descriptor Reference](#) in *Programming WebLogic Enterprise JavaBeans*.
- Increase the value of the `transaction-timeout` element of `wlw-config.xml`. For more information, see [wlw-config.xml Configuration File](#) in the *WebLogic Workshop Help*.

[NOTE TO REVIEWERS: THIS INFORMATION IS BASED ON COMMENTS IN CR185603. ARE THERE REASONS A PROGRAMMER WOULD CHOOSE ONE OF THESE APPROACHES OVER THE OTHERS?]

For recovery considerations regarding transaction timeout periods, see “What should I check to make sure my WebLogic Integration application is configured properly for recovery?” on page 2-1.

Run-Time Tuning Issues

This section answers the following questions:

- [What flags should I use when I start WebLogic Server to maximize performance?](#)
- [How do I tune WebLogic Integration Applications?](#)
- [Can I configure the value of max-beans-in-free-pool for the JMS event generator?](#)
- [Which process tracking levels should I use to optimize performance?](#)
- [Do I need to run the Archiver if the process tracking levels are set to none?](#)
- [How do I turn off informational Web services messages?](#)

- When should my WebLogic Integration application use the document store?
- Which is the best persistence model for JMS—file-based persistence or JDBC?
- What should I watch for when monitoring a WebLogic Integration application running under load?
- How can I reduce the number of transactions timing out?
- Why does my application seem to leak memory?

What flags should I use when I start WebLogic Server to maximize performance?

For optimum performance, use the following flags when starting WebLogic Server:

```
production noiterativedev nodebug notestconsole
```

Note: These flags should be set by default when using a generated production domain, but not when using a development domain.

How do I tune WebLogic Integration Applications?

WebLogic Integration projects map onto J2EE resources. For background information on exactly how they map, see *WebLogic Workshop Internals* at the following location:

http://dev2dev.bea.com/products/wlworkshop81/articles/wlw_internals.jsp

For background information on tuning in general, see *WebLogic Server Performance and Tuning* at the following location:

<http://edocs.bea.com/wls/docs81/perform/index.html>

The primary EJB pools you can tune within a WebLogic Integration project are the project beans: the `SyncDispatcher` Stateless Session Bean (SLSB) pool and the `AsyncDispatcher` Message-Driven Bean (MDB) pool. These pools exist for every WebLogic Integration project in an application.

All synchronous requests are routed through the `SyncDispatcher`; all asynchronous (buffered) communication is routed through the `AsyncDispatcher`.

There are two ways to configure the pool size of these beans:

- The preferred method to configure MDB pool size is to configure the EJBs in the project beans directory to use a dedicated thread pool by updating the `weblogic-ejb-jar.xml` file to contain a `dispatch-as` element with the name of the thread pool to use.

For more information, see “dispatch-policy” in “2.0 weblogic-ejb-jar.xml Elements” in the [weblogic-ejb-jar.xml Deployment Descriptor Reference](#) in *Programming WebLogic Enterprise JavaBeans*.

- The alternative method to configure MDB pool size is to set `max-beans-in-free-pool` for the project beans.

For more information about configuring MDB pool size, see “Setting Performance-Related weblogic-ejb-jar.xml Parameters” in [Tuning WebLogic Server EJBs](#) in *WebLogic Server Performance and Tuning*.

Can I configure the value of `max-beans-in-free-pool` for the JMS event generator?

Yes. The JMS event generator by default has `max-beans-in-free-pool` set to 5. You can often set this value higher and improve performance.

For more information, see “Setting Performance-Related weblogic-ejb-jar.xml Parameters” in [Tuning WebLogic Server EJBs](#) in *WebLogic Server Performance and Tuning*.

Which process tracking levels should I use to optimize performance?

Minimize process tracking levels as much as possible to optimize performance. Set tracking to none as the default, and track selected JPDs as needed.

For information on how to configure tracking levels, see “Managing Process Tracking Data” in “About Process Configuration” in [Process Configuration](#) in *Managing WebLogic Integration Solutions*.

Do I need to run the Archiver if the process tracking levels are set to none?

Yes. You should run the archiver process during non-peak hours to clean the process monitor table for stateful processes.

For information on how to configure the archiver process, see “Managing Process Tracking Data” in [Process Configuration](#) in *Managing WebLogic Integration Solutions*.

How do I turn off informational Web services messages?

Edit `common/lib/workshopLogCfg.xml` and change all occurrences of `info` to `warn`.

When should my WebLogic Integration application use the document store?

Each application will have a different threshold for the size of documents it is more efficient to pass by reference (using the document store) rather than inline. In general, the more hops a document makes, the better it is to use the document store. You configure the size of files passed by reference by setting the value of the `weblogic.wli.DocumentMaxInlineSize` parameter in the WebLogic Integration application domain's `wlw-config.properties` file.

Which is the best persistence model for JMS—file-based persistence or JDBC?

If you have a fast disk subsystem or a battery backed caching controller, file-based persistence can be quite a bit faster than JDBC.

What should I watch for when monitoring a WebLogic Integration application running under load?

The following table shows parameters of key resources used by your WebLogic Integration application that you should monitor in the WebLogic Server Administration Console, and actions you should take if a problem condition arises.

Table 1-1 Parameters to Monitor in WebLogic Integration Applications

Parameter	Where to Monitor	Problem Condition	Actions Required
Idle thread count	Choose the Configure Execute Queue command on the Advanced Options portion of the Configuration→General tab for the server.	Count approaching 0.	<p>If the value of <code>Thread Count</code> is below 5, change it to a higher value. The default production thread count for the default queue is 25.</p> <p>For more information, see “Tuning the Default Execute Queue Threads” in Tuning WebLogic Server in <i>WebLogic Server Performance and Tuning</i>.</p>
JDBC connection pool usage	Choose the JDBC Connection Pool→Monitoring tab.	Connections in use approaching maximum available.	<p>Increase the <code>MaxCapacity</code> attribute of <code>JDBCConnectionPool</code>.</p> <p>For more information, see “How JDBC Connection Pools Enhance Performance” in Tuning WebLogic Server in <i>WebLogic Server Performance and Tuning</i>.</p>
JMS async queues for project beans	Expand the JMS→Servers node, select a server, and then click Monitor.	Many messages in the queue.	<p>Increase the pool size for the <code>AsyncDispatcher</code>.</p> <p>For more information, see How do I tune WebLogic Integration Applications?.</p>

Table 1-1 Parameters to Monitor in WebLogic Integration Applications

Parameter	Where to Monitor	Problem Condition	Actions Required
wli.internal.tracking.buffer	Choose <i>Domain</i> → Services→JMS→ Servers→cgJMSServer.	Many messages in the queue	Create a thread execute queue named wli.internal.ProcessTracking, and increase the number of threads in that pool. For more information, see Tuning WebLogic Server in WebLogic Server Performance and Tuning .
wli.internal.instance.info.buffer	Choose <i>Domain</i> → Services→JMS→ Servers→cgJMSServer.	Many messages in the queue	Create a thread execute queue named wli.internal.ProcessTracking, and increase the number of threads in that pool. For more information, see Tuning WebLogic Server in WebLogic Server Performance and Tuning .

How can I reduce the number of transactions timing out?

If you have deployed your application as an exploded EAR, you can increase the amount of time that elapses before a transaction times out by increasing the value of `trans-timeout-seconds` in the WebLogic Server Administration Console. You should also change this value in the source for your WebLogic Integration application so that the value of `trans-timeout-seconds` is not reset to its default value when you redeploy the application.

If you have not deployed your application as an exploded EAR, `trans-timeout-seconds` is not accessible through the WebLogic Server Administration Console. You must increase the value of `trans-timeout-seconds` in your application, and then redeploy the application with the increased timeout value.

For information on how to change the value of `trans-timeout-seconds`, see “Edit Deployment Descriptors” in [Implementing Enterprise Java Beans](#) in *Programming WebLogic Enterprise JavaBeans*.

[THIS INFORMATION IS BASED ON CR185603.]

Why does my application seem to leak memory?

The top causes of memory leaks are:

- Operating a WebLogic Integration application with the test console running. This is not actually a leak, but running with the test console on uses a lot of memory.

For more information, see [What flags should I use when I start WebLogic Server to maximize performance?](#).

- Lots of JMS timer messages. This is also not actually a memory leak, but this situation does cause an increasing amount of memory to be used. The memory is recovered when the messages are delivered.

To reduce JMS timer messages, see [How can I increase the transaction timeout period?](#)

- Using `ShrinkingEnabled` in the JDBC connection pool. There's an occasional memory leak associated with `ShrinkingEnabled`. Domains generated in WebLogic Integration 8.1 SP1 may have `ShrinkingEnabled` enabled by default.

For more information about the `ShrinkingEnabled` attribute, see “Attributes” in [JDBCConnectionPool](#) in *WebLogic Server Configuration Reference*.

Performance Tips

WebLogic Integration Application Recovery

The following section provides answers to questions about configuring WebLogic Integration applications for recovery. It contains the following topic:

- [Recovery Checklist](#)

Recovery Checklist

This section answers the following questions:

- [What should I check to make sure my WebLogic Integration application is configured properly for recovery?](#)
- [How long should I wait to start recovery?](#)

What should I check to make sure my WebLogic Integration application is configured properly for recovery?

You should check to make sure your WebLogic Integration application is configured in the following manner:

- An XA driver is configured in `cgPool`.
- All JMS servers are targeted to a migratable target. (Migratable targets have the word *migratable* in the target name.)

Note: If you need to change the targeting of one or more JMS servers in an active cluster, you must restart the cluster after you make changes to the targeting configuration.

- `WLAI_DataSource` is not pointing to `cgPool`. If it is, either delete `WLAI_DataSource` or retarget it to another pool.

Note: `WLAI_DataSource` pointing to `cgPool` was the default configuration for domains generated by the Configuration Wizard in WebLogic Integration 8.1 SP1. This is not a problem for domains generated in WebLogic Integration 8.1 SP2.

For more information on retargeting `WLAI_DataSource` to another pool, see “Guidelines for Changing to an XA Configuration” in [How Do I... in Creating WebLogic Configurations Using the Configuration Wizard](#).

- The `WLI Admin` component of `WLI System EJBs` is targeted to the cluster. You can verify the targeting of `WLI Admin` by inspecting its configuration in the `config.xml` file for the `WLI System EJBs` application.

Note: `WLI Admin` can be targeted to the cluster only when JMS servers are targeted to migratable targets.

- Retries and retry intervals are appropriate. Retries multiplied by retry interval should exceed the time it takes for JTA recovery to run.

If you need to tune your retries and retry intervals, you have the following choices:

- Carefully set `retry-count` and `retry-interval` in your JPDs.
- Set the `retry-count` and `retry-interval` of the `async` and `error` queues for each JPD project (WebApp).

Note: This will break explicit retry settings on the JPD, but it is the easiest and recommended approach.

For more details, see [JMSQueue](#) in the *WebLogic Server Configuration Reference*.

- JTA attribute `TimeoutSeconds` in the `config.xml` file is set to a value less than the value of the `JDBCConnectionPool.XA TransactionTimeout` attribute for XA connection pools.

For more information about `TimeoutSeconds`, see [JTA](#) in the *WebLogic Server Configuration Reference*.

- JTA attribute `MaxTransactions` in the `config.xml` file is set to a value large enough to accommodate the number of simultaneous transactions that could occur on a server during recovery. For transaction intensive applications, you should increase the value of this attribute from the default setting.

For more information about `MaxTransactions`, see [Domain-->Configuration-->JTA](#) in the *WebLogic Server Administration Console Online Help*.

[NOTE TO REVIEWERS: THE ITEM ABOVE HAS BEEN ADDED IN RESPONSE TO CR195315.]

- `JDBCConnectionPool` attribute `MaxCapacity` in the `config.xml` file is set to a value greater than the value of the execute queue `ThreadCount` attribute.

For more information about `MaxCapacity`, see [JDBCConnectionPool](#) in the *WebLogic Server Configuration Reference*.

For information about `Server ThreadCount`, see “Tuning the Default Execute Queue Threads” in [Tuning WebLogic Server](#) in *WebLogic Server Performance and Tuning*.

- When using Oracle, the repository tables `dba_2pc_pending`, `dba_2pc_neighbors`, and `dba_pending_transactions` must have the right permissions for recovery to be called. If they do not, you will get a database error and recovery will fail.

How long should I wait to start recovery?

It can take several minutes for in-doubt transactions to show up in Oracle. There may be a race if recovery has started prior to Oracle detecting the loss of the TM. Wait several minutes before starting recovery—either by restarting the server or by doing a JTA migration.

After initiating recovery, if you check the Oracle `dba_2pc_pending` table and see a record associated with the failed server that has a timestamp prior to initiating recovery, recovery will fail. Restart the server.

