IBM FileNet P8
Version 5.0

Performance Tuning Guide

GC19-3394-00

IBM FileNet P8
Version 5.0

Performance Tuning Guide

GC19-3394-00

Note

Before using this information and the product it supports, read the information in "Notices" on page 49.

Contents

About this document

This document provides tuning tips that can help you improve the performance of IBM® FileNet® P8.

Tuning tip organization

If a tuning tip involves an independent software vendor product, and it applies to more than one of the FileNet P8 applications, section for the independent software vendor product, under a heading for the specific vendor. However, if a tuning tip involves an independent software vendor product, and it applies to only one FileNet P8 application, the tuning tip will be in the section for the FileNet P8 application, under a heading for the independent software vendor product and a sub-heading for the specific vendor.

Parameter settings

Many tips in this document do not include suggested parameter settings. However, if the IBM Performance Lab has found certain settings beneficial, the settings are suggested. You can adjust the suggested settings to fit your environment.

Assumptions

This document assumes you are a systems administrator and you have:

- Reviewed the IBM FileNet P8 Hardware and Software Requirements guide for your version of FileNet P8 and have verified that FileNet P8 supports the hardware and software components in your current configuration.
- Installed FileNet P8 according to the IBM FileNet P8 Platform Installation and Upgrade Guide for your version of FileNet P8.
- Configured the FileNet P8 databases with sufficient processing power, memory, and a set of indexes that are appropriate to the application.

Operating systems

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document. For operating system information that is specific to an application, see the topics that are specifically for that application:

Solaris

Processing time for TCP connections

When many requests are being processed, the Java process can run out of available connections and cause “Java.net.BindException: Address already in use” processing errors.

You can prevent Java processing errors and increase TCP socket performance by decreasing the value of the tcp_time_wait_interval parameter or by increasing the value of the tcp_conn_equ_max_q parameter, or by adjusting the values of both parameters accordingly.

Decreasing the TCP wait time

The tcp_time_wait_interval parameter value specifies how long a TCP port is unavailable after both the client and the server have closed a connection. Temporarily disabling the port prevents new programs from inadvertently receiving packets meant for a previous program.

Solaris 8: The default value is 240,000 milliseconds (4 minutes).

Solaris 9 and 10: The default value is 60,000 milliseconds (1 minute).

By decreasing this value, TCP connections can be recycled more quickly.

Use the following command to adjust the TCP wait time:

```
ndd –set /dev/tcp tcp_time_wait_interval 240000
```

Increasing the maximum number of TCP connection requests

The tcp_conn_req_max_q parameter value specifies the maximum number of TCP connection requests that the server can queue up after establishing a successful connection (handshake) with a client.

The parameter default value is 128. By increasing this value, Java can use more TCP connections.

CAUTION Each open TCP connection occupies some memory (usually both bookkeeping tables, and buffer space), and fewer TCP connections are available to other processes. However, the memory footprint is only a minor concern for enterprise-class servers with large amounts of available RAM.

Increasing the tcp_conn_req_max_q parameter value Use the following command to adjust the maximum number of TCP connection requests that the server can queue up after establishing a successful connection (handshake) with a client.

```
ndd –set /dev/tcp tcp_conn_req_max_q 128
```

Windows

Processing time for TCP connections

When many requests are being processed, the Java process can run out of available connections and cause “Java.net.BindException: Address already in use” processing errors.

You can prevent Java processing errors and increase TCP socket performance by decreasing the TcpTimedWaitDelay parameter value, or by increasing the TcpNumConnections and MaxUserPort parameter values, or by adjusting the values of both parameters accordingly.

Decreasing the TCP wait time to release a closed connection

The TcpTimedWaitDelay parameter value specifies the amount of time that must elapse before TCP can release a closed connection (TIME_WAIT state) and reuse its resources.

Decreasing the TcpTimedWaitDelay parameter value decreases the amount of time that a connection stays in the TIME_WAIT state.

Use the Registry Editor to decrease the TcpTimedWaitDelay parameter value, which is located in the following registry path:

> HKLM\System\CurrentControlSet\Services\Tcpip\Parameters

Increasing the maximum number of open TCP connections

The TcpNumConnections parameter value specifies the maximum number of connections that TCP can have open simultaneously.

CAUTION Each open TCP connection occupies some memory (usually both bookkeeping tables and buffer space), and fewer TCP connections are available to other processes. However, the memory footprint is only a minor concern for enterprise-class servers with large amounts of available RAM.

Use the Registry Editor to decrease the TcpNumConnections parameter value, which is located in the following registry path:

> HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters

Increasing the MaxUserPort registry parameter value

The MaxUserPort parameter value specifies the highest port number that TCP can assign when an application requests an available user port. Increase this value when Content Engine requires high throughput transaction processing.

Use the Registry Editor to increase the MaxUserPort parameter value, which is located in the following registry path:

> HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters

Windows 2008

CPU usage

By default, Windows® 2008 starts the Server Manager Console at startup and continually refreshes its information every 2 minutes. At this frequency, Server Manager Console refreshes can cause significant increases in CPU usage.

If you notice regular increases in CPU usage, check the Server Manager Console configuration and consider the following modifications:

- Select “Do not show me this console at login” to stop the Server Manager Console from automatically starting when Windows 2008 starts.
- Click Configure Refresh to disable or modify the Server Manager Console refresh interval.

Databases

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document. For database information that is specific to an application, see the topics that are specifically for that application.

All databases

Review the database indexing tables in this document for performance enhancing database indexes that you can create on one (single) or more (multiple) database columns.

TERMINOLOGY NOTE In FileNet P8 documentation these indexes are typically referred to as “single-property indexes” and “composite-properties indexes”, respectively.

Database indexing tables

The database indexing tables in this document include the following information:

- **Class** – Class [path] in Enterprise Manager that you follow to access the property that is being indexed

  CLASS (PATH) NOTE If a class [path] is not listed, the property is a system property.

- **Database table name** – Location of the property or properties that are being indexed
- **Database column name** – Names of the FileNet P8 properties, as they appear in the database that make up the index
- **FileNet P8 property name** – Symbolic names of the FileNet P8 properties that make up the index
- **Reason to create the index** – Information about how and why to create the index

“*uxx*” prefix

The “u*xx*” prefix represents a variable in the property name, and indicates that the property is a custom property that is either from an installed expansion product or is user-defined.

Tools for creating FileNet P8 indexes

Use Enterprise Manager to create single-property indexes, unless the indexes are for a FileNet P8 system property. If an index is for a FileNet P8 system property or a composite-properties index, use the database management software to create the index.

Use the Microsoft SQL Profiler, the Oracle Statspack, Oracle ADDM, or the DB2 Design Advisor to identify additional properties (columns) that should be indexed.

Performance enhancing indexes

Consider creating one or more of these database indexes to improve system performance regardless of your database system. These indexes are useful to both FileNet P8 applications and custom applications, which include but are not limited to Workplace, Workplace XT, IBM InfoSphere Enterprise Records, and Business Process Framework.

Class	Database table name	Database column name	FileNet P8 property name	Reason to create index
Objectstore > Document Class > Record	docversion	mime_type	MIME Type	Create this single-property index when the number of records in the docversion table exceeds 1,000,000, or when both of the following conditions are true: • The number of items in a folder exceeds 2,000. • Performance degrades across common functionality such as browsing, searching, and creating documents.
N/A	generic	security_policy_id	N/A	Create this single-property index in all of the listed locations when response times degrade or errors occur when a security policy is deleted from an object store that has more than 1,000,000 documents.
N/A	container			
N/A	generic			

Composite properties indexes for continuable searches

To achieve the best performance during continuable (paged) searches:

- Create a composite index on:

  o Any selective columns that can be indexed

  and

  o object_id

  For example, if a continuable search can use an index on selective column “*myProperty*”, then create the following composite properties index: (*myProperty*, object_id).

- Add an “ORDER BY *myProperty*” clause to the search.

By making these adjustments, the database SQL select statement that Content Engine issues uses an ORDER BY *uxx_myProperty*, object_id clause, which can result in a database query plan that uses the composite index and results in much faster paging results.

Indexing object value properties

Consider creating single-value property indexes when you use object-value properties (OVPs) that are associated with a large table, such as the docversion table.

Generally, if the association properties values are mostly null, there is little overhead in index maintenance, and little affect on document ingestion.

Class	Database column name	FileNet P8 property name	Reason to create index
Database table name			
class [path]	*uxx_property name*	Property Name	When object-value properties are associated with a large table, they can benefit from indexes.
table name			

Memory and disk I/O

Memory and disk I/O limitations are the main contributors to database-related performance issues.

Use the appropriate operating system tools to monitor memory usage, CPU usage, and disk performance. Use vmstat and iostat on UNIX®-based systems.

Compute statistics for optimal performance

As content is added, removed, and changed in a database, it is important to run a statistics update periodically for optimal performance.

Reorganize indexes and tables

As large amounts of content are added, removed, or changed in a database, the associated tables, and indexes need to be periodically reorganized for optimal performance.

See the database vendor documentation to determine if a REORG is necessary, and if so, how to complete the process.

DB2

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document and that use an IBM DB2 database. For database information that is specific to an application, see the topics that are specifically for that application.

Performance enhancing indexes

Consider creating one or more of these database indexes to improve system performance when the database is DB2. These indexes are useful to both FileNet P8 applications and custom applications, which include but are not limited to Workplace, Workplace XT, IBM InfoSphere Enterprise Records, and Business Process Framework.

Class Database table name	Database column name	FileNet P8 property name	Reason to create index
N/A Relationship	tail_id ----------- head_id	N/A	Create this composite-properties index to improve the performance of insubfolder and infolder searches. For example, when Workplace and Workplace XT use these query constructs to retrieve stored searches or properties of stored searches. InfoSphere Enterprise Records also uses these query constructs.

Improving the performance of FileNet Content Manager applications

DB2_WORKLOAD is a DB2 Windows registry variable that represents pre-defined groups of (other) Windows registry variables that have been compiled to optimize performance of certain systems

As of DB2 9.7 (and DB2 9.5 fix pack 5), “FILENET_CM” is a valid setting for the DB2_WORKLOAD registry value and represents optimally set registry variables that have been compiled to increase the performance of applications that use Content Manager.

The FILENET_CM registry variable includes the DB2_EVALUNCOMMITTED and DB2_SKIPINSERTED registry variables, which help to improve concurrent query and ingestion workloads. Consider enabling the DB2_EVALUNCOMMITTED and DB2_SKIPINSERTED registry variables. When these registry values are enabled, unnecessary locking is avoided, which can improve performance and, in some cases, prevent deadlock or query timeout errors.

Enabling the DB2_WORKLOAD Windows registry variables

Enable the DB2_WORKLOAD Windows registry variable by entering the following command in the DB2 command-line processor:

```
db2set DB2_WORKLOAD=FILENET_CM
```

Oracle

Maximum number of concurrent connections

The QUEUESIZE parameter value in the listener.ora file specifies the maximum number of concurrent connections that the listener can receive.

Increasing the number of concurrent connections

Increasing this parameter value helps prevent listener backlog full conditions, which generate the Oracle ORA – 12541: "No listener" and ORA-12535: “TNS:operation timed out” errors.

Increasing the QUEUESIZE parameter value

To set the QUEUESIZE parameter, edit the address line in the listener.ora file as follows:

```
(ADDRESS = (PROTOCOL = TCP)(HOST = servername)(PORT = portnumber)(QUEUESIZE=100))
```

The listener.ora file resides on the Oracle database server:

```
$ORACLE_HOME/network/admin
```

For more information, see “Handling Large Volumes of Concurrent Connection Requests” in the Oracle online help at:

Version 10 g
http://download.oracle.com/docs/cd/B19306_01/network.102/b14212/listenercfg.htm#NETAG0101

Version 11 g
http://download.oracle.com/docs/cd/B28359_01/network.111/b28316/listenercfg.htm#CHDEBAJE

Connectivity parameters

Increasing the parameter values for the parameters in the following table can prevent database connectivity issues when FileNet P8 requires many simultaneous connections to the Oracle database.

The values must enable the Oracle database instance to sustain the total number of possible simultaneous connections for all object stores in the database instance.

Parameter name	Description
db_file_multiblock_read_count	Minimizes disk I/O during table scans by specifying the maximum number of blocks read in one I/O operation during a sequential scan.
job_queue_processes	Specifies the maximum number of processes that can be created for the execution of jobs.
parallel_max_servers	Specifies the maximum number of parallel execution processes and parallel recovery processes for an instance.
processes	Specifies the maximum number of operating system user processes that can simultaneously connect to an Oracle server. This value accomodates all background processes such as Job Queue (SNP) and parallel execution (Pnnn) processes.
sessions	Specifies the total number of user and system sessions. **NOTE** The default number is greater than PROCESSES to allow for recursive sessions. The sessions default is derived (1.1 * PROCESSES + 5).

Microsoft SQL Server

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document and that use a Microsoft SQL Server database. For database information that is specific to an application, see the topics that are specifically for that application.

Large content uploads and downloads

Uploading and downloading documents that are 300 MB or larger can cause errors in Content Engine, Workplace, and other content-based applications written to the Content Engine APIs.

To prevent or fix this problem add the following custom property to the data source configuration:

Object Store DataSource non-XA:

packetSize=32767

Object Store DataSource XA:

packetSize=32767

WebSphere

Complete the following procedure to update the packet size on WebSphere.

1. From the WebSphere Administrative Console, select Console > JDBC > Data Sources, and then click an object store data source.
2. Select Custom Properties.
3. From the Custom Properties page, click New to add the following property:

   Name: packetSize, Value: 32767

WebLogic

Complete the following procedure to update the packet size on WebLogic.

4. From the WebLogic Administrative Console, click an object store data source.
5. Select Connection Pool.
6. Add the following two properties in the Properties field:

   packetSize=32767

JBoss

Complete the following procedure to update the packet size on JBoss.

1. Edit an object store data source XML file (*your_datasource*-ds.xml and *your_datasource*-xa-ds.xml) located in ./server/yourserver/deploy.
2. Add the following two lines to the XML file:

   XA

   <packetSize>32767</packetSize>

   Non-XA

   <packetSize>32767</packetSize>

Application servers

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document. For application server information that is specific to an application, see the topics that are specifically for that application.

WebSphere, WebLogic, and JBoss

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document and that use either the WebSphere, WebLogic, or JBoss application server. For application server information that is specific to an application, see the topics that are specifically for that application.

Connection pools

WebSphere, WebLogic, and JBoss all use connection pooling in their data source configurations to minimize resource usage when connecting to the database.

If an error occurs during high-volume conditions, consider increasing the maximum capacity of the connection pools.

Adjusting the connection pool maximum capacity setting

Use the following formulas as a guide to determining the connection thread pool capacity.

- For Global Configuration Database (GCD) connection pools, both distributed (XA) and non-distributed (non-XA):
  - Maximum connections = 10* (# object stores) + (# execution threads) +10
  - Minimum connections = 20 + (# execution threads)
- For each individual object store connection pools:
  - Maximum and minimum connections = 20 + (# execution threads)

NOTE Execution threads are the maximum number of expected concurrent client connections. In the global configuration database data source, execution threads are the maximum number of expected concurrent client connections for all object stores in a domain. In the object store data source, they are the maximum number of expected concurrent client connections for that object store.

Adjusting the connection pool maximum capacity on WebSphere 6.0.x

Complete the following procedure to adjust the connection pool maximum capacity setting for WebSphere 6.0.x.

1. Navigate to Resources > JDBC Providers.
2. Select Connection Pool Name > Data Sources.
3. Select Data Source Name > Connection Pool Properties.
4. Adjust the Maximum Capacity setting.

Adjusting the connection pool maximum capacity on WebSphere 6.1.x

1. Navigate to Resources > JDBC.
2. Adjust the Maximum Capacity setting.

Thread pools

If you encounter errors when the application server handles a high volume of requests, or you encounter errors that are related to insufficient thread creation, consider increasing the minimum and maximum thread pool settings.

Increasing the minimum and maximum thread pool settings

Increasing the minimum thread pool size enables the system to pre-allocate resources when the system is started.

Increasing the minimum and maximum thread pool settings on WebSphere 6.0.x and 6.1

Use the following procedure to increase the minimum and maximum thread pool settings on WebSphere 6.0.x and 6.1.

1. Navigate to Servers > Application Servers > ***ServerName*** > Additional Properties > Thread Pools > WebContainers.
2. Adjust either the minimum setting or the maximum setting, or adjust both.

WebSphere

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document and that use the WebSphere application server. For application server information that is specific to an application, see the topics that are specifically for that application.

Lightweight third party authentication (LTPA) timeout: Token validity

The LTPA “Timeout” setting specifies how long an LTPA token is valid (in minutes). The token contains this expiration time so that any server that receives the token can verify that the token is valid before proceeding further. When the defined amount of time has passed, all user tokens expire regardless of session activity.

Consider increasing this value if one or both of the following conditions occur:

- Batch processes or other operations run uninterrupted for longer than the currently configured LTPA timeout value.
- The following message is in the server.log file:

  “org.omg.CORBA.NO_PERMISSION: Credentials have expired”

Adjusting the LTPA timeout setting – WebSphere 6.1.x

Complete the following procedure to adjust the LTPA timeout setting on WebSphere 6.1.x.

1. Navigate to Secure administration, applications, and infrastructure > Authe and Authorization Service.
2. Under Authentication, select Authentication mechanisms and expiration. ntication
3. Enter the security cache timeout value in the “Timeout value for forwarded credentials between servers” field.

Adjusting the LTPA timeout setting – WebSphere 7.0.x

Complete the following procedure to adjust the LTPA timeout setting on WebSphere 7.0.x.

1. Navigate to **Security > Global security > Authentication (Authentication mechanisms and expiration)**
2. Under Authentication, select **LTPA**.
3. Enter the security cache timeout value in the “Timeout value for forwarded credentials between servers” field.

Client-side security cache timeout: Cache refresh

The client-side security cache timeout setting specifies how often security-related caches are refreshed (in minutes). When the defined amount of time has passed, the security-related cache is refreshed and any new requests for information result in a database lookup.

The default value is 600. Increasing this value to 3600 provides a modest improvement in server response times. (However, increasing the value further does not provide further improvement.) If a security cache refresh is not required every 600 seconds, consider increasing the cache timeout value.

Adjusting the security cache timeout setting – WebSphere 6.1.x

Complete the following procedure to adjust the security cache timeout setting on WebSphere 6.1.x.

1. Navigate to Secure administration, applications, and infrastructure > Authentication and Authorization Service.
2. Under Authentication, click Authentication mechanisms and expiration.
3. Enter the security cache timeout value in the “Authentication cache timeout” field.

Adjusting the security cache timeout setting – WebSphere 7.x

Complete the following procedure to adjust the security cache timeout setting on WebSphere 7.x.

1. Navigate to **Security > Global security > Authentication (Authentication mechanisms and expiration)**.
2. Under Authentication, click **Authentication cache settings**.
3. Enter the security cache timeout value in the “Authentication cache timeout” field.

Performance monitoring infrastructure PMI

By default, the WebSphere Performance Monitoring Infrastructure is enabled. Disabling this option provides moderate response time and CPU utilization improvements. If you are not troubleshooting performance or scalability issues, consider disabling this option.

Disabling the WebSphere Performance Infrastructure

Complete the following procedure to disable the WebSphere Performance Infrastructure.

1. Navigate to Monitoring and Tuning > Performance Monitoring Infrastructure (PMI).
2. Select the server, and then deselect the Enable PMI option.

Service logs

Service logs are messages that are written to the System.out stream. Service logs contain extended service information that is normally not of interest, but can be important when analyzing problems.

On occasion, service logs can cause excessive disk I/O even when WebSphere is operating as expected. If this I/O affects other applications that are co-located with the application server, consider:

- Moving the service logs to a separate set of physical disks

or

- Disabling the service logs

Disabling service logs

1. Navigate to Troubleshooting > Logs and Trace > ***ServerName*** > IBM Service Logs.
2. Deselect the “Enable service log” option.

Local host lookups

The following JVM property improves the performance of host name look-ups by caching the host name that is returned from the underlying operating system:

com.ibm.cacheLocalHost=true

The JVM always uses the cached host name, so this tuning tip might not be appropriate for situations where the local host name can change for a Java application.

Configuring the com.ibm.cacheLocalHost property

1. In the WebSphere Administrative Console, select Application Servers > ***ServerName*** > Process Definition > Control > JVM Properties > Custom Properties > New.
2. Enter the following information:
   - Name: com.ibm.cacheLocalHost
   - Value: true

WebLogic

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document and that use a WebLogic application server. For application server information that is specific to an application, see the topics that are specifically for that application.

Server and client Java Virtual Machine switches

Sun Microsystems has a version of Java Virtual Machine (JVM) for server-side performance and a version for client-side performance. These two solutions share the Java HotSpot runtime environment but have different compilers, which are suited to the distinctly different performance characteristics of clients and servers.

The server-side JVM has shown significant performance improvements over the client-side JVM in highly scalable server environments. If the application server is running in production mode, the server compiler is already being used.

JVM tuning

For additional information about application-specific JVM tuning, check the appropriate application-specific sections.

Tuning for improved response time and CPU utilization

To improve overall throughput, consider using a generational garbage collection strategy, which is also known as a “generational concurrent garbage collection.”

A generational garbage collection strategy works best with workloads that are designed to reduce or eliminate garbage collection pauses and are designed to trade some memory throughput to accomplish the reduction or elimination.

Applications with these kinds of workloads often use gigabyte-sized heaps. Following are some characteristics of workloads that could benefit from generational garbage collection:

- Allocates many short-lived objects
- Are transaction-based, where objects in the transaction are deleted after the transaction is committed
- Tend to develop a fragmented heap

NOTE To verify the effectiveness of the garbage collection strategy, run the application, and measure either response times or the throughput relative to garbage collection pause times, or both.

JVM switch	Description
WebSphere format -Xgcpolicy:*gcType* For example: -Xgcpolicy:gencon	Defines a specific garbage collector algorithm. Consider setting the garbage collector algorithm to “gencon”, a generational, concurrent garbage collector. With this type of garbage collector, objects are allocated in the young generation (the nursery). When the young generation is full, JVM stops all Java threads and moves the live objects in the young generation to the old generation.
JRockit format -Xgc:gcType For example: -Xgc:gencon	
Sun HotSpot Generational garbage collection is the default setting.	

High-volume ingestion scenarios

During high-volume ingestion scenarios, Content Engine creates a large amount of short-lived objects. Consider providing ample space to your new or tenure generation. Monitor your JVM by using VerboseGC or WebSphere Performance Management Infrastructure and adjust the size of these generations appropriately.

JRockit tuning for improved response time and CPU utilization

To improve response time and CPU utilization consider tuning the following JVM switches. Start by using the values that are listed in the examples and then adjust them accordingly.

JVM switch	Description
-XXtlaSize	Defines the size of thread local areas (TLAs), which are used for object allocation. The default value is 2 KB. Format: -XXltaSize:*size*[k|K] [m|M] [g|G] For example: java -XXtlaSize:64k
-XXlargeObjectLimit	Defines the size of a large object. Objects that exceed the defined size limit are considered large and are not allocated in TLAs. Format: -XXlargeObjectLimit:*size*[k|K] [m|M] [g|G] For Example: java -XXlargeObjectLimit:64k

JVM switch	Description
-Xns	Defines the nursery size. Consider setting the JRockit nursery size to 25% of the total heap. Format -Xns:size[k|K] [m|M] [g|G] For Example: java -Xns:300m

Heap size

The JVM heap is a repository of active objects, inactive objects, and free memory. When the JVM heap runs out of memory, all processing in the JVM stops until garbage collection completes.

The JVM heap size is important because it controls how often, and for how long garbage collection runs. If garbage collection runs too infrequently, the extra time that is required to complete garbage collection can negatively affect FileNet P8 performance.

When you tune the JVM heap size, consider the following two objectives:

- Minimizing the amount of time that is required to complete garbage collection
- Maximizing FileNet P8 functionality to handle a significant workload

By balancing the JVM heap with the number of instances in the application server cluster, you can minimize the number of lengthy garbage collection pauses, and the frequency of full garbage collections that are caused by insufficient heap space.

The JVM heap size is a configurable parameter that you can set when you start the JVM. The lowest acceptable value for the minimum heap size is 512 MB, and the lowest acceptable value for the maximum heap size is 1GB.

Native out-of-memory errors

Out-of-memory errors can occur when the JVM does not have an adequate amount of native memory space.

Confirm enough native memory is available to the JVM, which includes memory for the maximum heap size, and additional memory required by the JVM, such as the Just-In-Time compiler, and other native code. To be safe, the amount of native memory allocated for the JVM process should be no more than 60% of the per process memory limit as dictated by the operating system and application server platform.

The required amount of additional memory required by the JVM process varies for each environment. However, 25 percent of the maximum heap size is a reasonable guideline. For example, if the maximum heap size is set to 1 GB, allow the native memory for the JVM process to at least 1.25 GB.

By applying these rules, an example of a JVM running on 32-bit Windows with 2GB of memory available to any single process might use the following settings as the smallest minimum and largest maximum allowed:

-Xms=512m

-Xmx=1228m

JVM out-of-memory errors

IBM

Under certain conditions, a FileNet P8 application server can encounter out-of-memory errors when the heap still had ample memory available. For example:

> E_UNEXPECTED_EXCEPTION: An unexpected exception was detected. Caused by: java.lang.OutOfMemoryError: JVMCI015:OutOfMemoryError, cannot create more threads due to memory or resource constraints.

Situations similar this example can occur when the memory inside the heap becomes fragmented or is otherwise poorly allocated.

One way to avoid similar situations is to increase the minimum heap size to 50% of the maximum heap size assuring sufficient memory is allocated when the JVM is started. Another option is to ensure that an adequate amount of memory is set aside in the heap to load the JVM code. This area of memory can be called the class cache. The amount of memory for the class cache is controlled by the Xinitsh JVM parameter, which sets the initial system size of the heap.

For JVMs that run on 32-bit operating systems, consider a value of 4m. JVMs that run on 64-bit operating systems are set to 8m by default, which is typically large enough to avoid this problem*.* To use this option, add the following switch to the application server command-line arguments: -Xinitsh<*size>.*

For example, -Xinitsh4m.

JRockit

Under certain conditions a FileNet P8 application server can throw out-of-memory errors when the heap still had ample memory available. An example of these errors is the following message: “Error starting thread: Not enough storage is available to process this command."

The cause of this message can be either improper sizing of the JRockit heap generations or memory fragmentation caused by system class placement in the heap, or both. Unlike the Sun Hotspot and IBM JVMs, JRockit does not have permanent system class heap space, which can cause fragmentation problems in certain scenarios.

To avoid this issue, consider setting your JRockit nursery size to 25% of the total heap. Depending upon the workload, this setting might require some additional adjustments. You can monitor the nursery size by using VerboseGC collection and analysis.

Directory servers

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document.

Tivoli

The performance tuning topics in this section apply to all IBM FileNet applications that are covered in this document and that use a Tivoli directory server.

Optimizing the Tivoli directory server database

To improve performance and query speeds, optimize the directory server database:

- On a regular schedule
- After significant database updates

To optimize the directory server database, use the Optimize Database task in the IBM Tivoli Directory Server Configuration tool. This tool uses the DB2 runstats command and updates statistical information that is used by the query optimizer for all the directory server tables.

Optimizing the directory server database using the Tivoli Optimize Database task

1. Stop the directory server.
2. Start the Configuration Tool by typing the following command on the command-line:

   ```
   idsxcfg
   ```

3. Click Optimize Database in the left navigation pane.
4. In the Optimize Database window, click Optimize.
5. When the optimization process finishes, exit the Configuration tool.
6. Restart the directory server.

Content Engine

The performance tuning topics in this section apply specifically to IBM FileNet P8 Content Engine.

Databases

The performance tuning topics in this section apply specifically to IBM FileNet P8 Content Engine regardless of the database type.

Consider creating the following indexes to improve Content Engine performance during multi-value property searches.

Database Table Name	Database Column Name	FileNet P8 Property Name	Tool	Description/ Reason to create index
ListOfString	element_value ----------- parent_prop_id ----------- parent_id	*Property Name*	Database	Create a composite-properties index to improve Content Engine performance during multi-value property searches.
ListOfBoolean				
ListOfInteger32				
ListOfFloat64				
ListOfDatetime				
ListOfId				
ListOfString	parent_ id ----------- parent_prop_id ----------- element_value	*Property Name*	Database	Create a composite-properties index to improve Content Engine performance during multi-value property searches that use wildcard values, such as LIKE %value%.

Create these indexes only if the table has at least several hundred rows and has enough distinct values relative to the number of rows in the table. (You can determine the cardinality by reviewing the table statistics or by using a group by query, for example select count(*), element_value from ListOfString group by element_value order by 1).

Example: ListOfString index

For example, use the following CREATE INDEX statement to create an index that improves Content Engine performance during multi-value searches:

```
create index I_MV on ListOfString (element_value, parent_prop_id, parent_id)
```

Example: ListOfString index for searches that begin with %wildcard% values

Use the following CREATE INDEX statement to create an index that improves Content Engine performance during multi-value property searches when an index on (element_value, parent_prop_id, parent_id) cannot be used because the search begins with a wild card value, such as LIKE %value%:

```
create index I_MV2 on ListOfString (parent_id, parent_prop_id, element_value)
```

NOTES

- Each database type has its own limit on index byte size, which can require extra work to create the index for string data types. By default, the ListOfString's element_value column is of character size 4000 (1333 on Oracle), might be longer than the maximum index length for the database.
- 4000 characters consume 8000 bytes during value insertion on Microsoft SQL Server and 8000 bytes during column creation on DB2. On an Oracle database, the varchar2 allocation is tripled. As a result, e999 bytes are consumed for a 1333 character size column.

Use the guidelines in the following table. If the database does not allow the index to be created, reduce the size of the ListOfTable element_value column. However, reducing this value might not work if the values stored in the column are too large. If only a few values are too large, consider changing the data in those columns to allow the index to be created.

Database	Index MaxBytes	Byte size of element_value	Index allowed?	Notes
Microsoft SQL 2008	900	8000	No	
Oracle 10 g	75% block size	3999	Under certain conditions	The Oracle 10 g block size is also minus overhead, which is small.
DB2 9	8k	8000	Yes	8k is one-quarter the table space size. The DB2 8k block size is also minus overhead, which is small.

DB2

The performance tuning topics in this section apply specifically to IBM FileNet P8 Content Engine when using an IBM DB2 database.

New (empty) object stores

After you create an object store, it is important that you monitor the database. As new content is added to the object store, the DB2 optimizer continually requires current table and index statistics to select the most efficient access plan. Ensuring that the DB2 optimizer has current statistics is especially important during large ingestions at a high rate, such as migration. It is also important for proof-of-concept testing, stress testing, and testing custom applications.

This section provides information about:

- Monitoring databases using the FileNet System Manager Dashboard
- Enabling automatic statistics
- Using the DB2 Task Center to gather statistics manually

Monitoring databases using the FileNet System Manager Dashboard

For each object store, Content Engine provides counts and durations for the various database operations and server operations. This information can provide valuable insight into database performance.

You can find these counts and durations by using the FileNet P8 System Manager Dashboard tool.

- To view counts and durations for database operations, select the following nodes for a given machine and Content Engine server:

  User > *object store name* > Database

- To view counts and durations for server operations select the RPC node for a given machine and Content Engine server.

Enabling automatic statistics

Click the following link for instructions about enabling automatic statistics from the command-line:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.admin.perf.doc/doc/c0011762.html

Improvements are possible when manually gathering statistics using the DB2 Task Center within the first 1,000,000 documents or the first 100,000 folders.

Using the DB2 Task Center to gather statistics manually

Use the DB2 Task Center to update statistics at regular intervals and send notifications. Scheduling information is stored in the DB2 tools Catalog database.

For example, something similar to the following script would be listed on the Command Script tab of the Task properties. The following could then be scheduled to run every 20 minutes during a migration.

```
CONNECT TO OS1DB;

RUNSTATS ON TABLE CEUSER.CONTAINER ON ALL COLUMNS AND INDEXES ALL ALLOW WRITE ACCESS ;

COMMIT WORK;

RUNSTATS ON TABLE CEUSER.CONTENT ON ALL COLUMNS AND INDEXES ALL ALLOW WRITE ACCESS ;
```

```
COMMIT WORK;

RUNSTATS ON TABLE CEUSER.DOCVERSION ON ALL COLUMNS AND INDEXES ALL ALLOW WRITE ACCESS ;

COMMIT WORK;

RUNSTATS ON TABLE CEUSER.GENERIC ON ALL COLUMNS AND INDEXES ALL ALLOW WRITE ACCESS ;

COMMIT WORK;

RUNSTATS ON TABLE CEUSER.HIERARCHYINDEX ON ALL COLUMNS AND INDEXES ALL ALLOW WRITE ACCESS ;

COMMIT WORK;

RUNSTATS ON TABLE CEUSER.RELATIONSHIP ON ALL COLUMNS AND INDEXES ALL ALLOW WRITE ACCESS ;

COMMIT WORK;

RUNSTATS ON TABLE CEUSER.SECURITYDESC ON ALL COLUMNS AND INDEXES ALL ALLOW WRITE ACCESS ;

COMMIT WORK;

CONNECT RESET;
```

For more information about using the Task Center to schedule tasks, see the DB2 documentation:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.admin.gui.doc/doc/t0022277.html

Full-text indexing for content-based retrieval

FileNet P8 parameters

Adjusting one or more of the following Enterprise Manager parameters for Content Search Engine can help improve content-based retrieval performance.

FileNet P8 domain parameter name	Description
Thread count	Specifies the number of background threads to process index requests. Increase this value on machines that have multiple CPUs, or on machines with configurations that distribute processing across multiple CSE servers. Typically, this is 2 threads per CSE server CPUs.
Optimization Interval	Specifies the number of document, annotation, folder (and so on) batches to index before optimizing on the index information.
Max Batch Size	Specifies the maximum number of objects to index for submitting to Content Search Engine in a single batch. The default value is 8000. Consider setting 4000 for smaller-sized documents and the English locale, and 1000 for larger-sized documents and the Uni locale.
Max Objects per Collection	Specifies the approximate number of objects to be placed in any Content Search Engine collection. Use the default value.
Lease Duration	Specifies the number of seconds that are required to complete an index request. If an index request takes longer than the lease time, the index request is considered abandoned and is re-executed. Use the default value.
Dispatcher Wait Interval	Specifies the number of seconds that the dispatcher waits before looking for work. Valid settings range from 5 to 600 seconds. If the value of this parameter is set too low, some content might not get migrated to the storage area by the time the Content Search Engine tries to index it.

Adjusting Content Search Engine parameters

1. Navigate to and open EM.
2. Right-click the EM domain root.
3. Select Properties, and then click the Verity Server tab.

Stop words

To improve Content Search Engine throughput by up to 30%, update the stop words list to include the most obvious case variations of a word. For example, the word “the” has three obvious variations: “The”, “the”, and “THE”.

Application servers

The performance tuning topics in this section apply specifically to IBM FileNet P8 Content Engine.

WebLogic

The performance tuning topics in this section apply specifically to IBM FileNet P8 Content Engine when using a WebLogic application server.

Group membership search

Because Content Engine uses WebLogic LDAP configuration for authentication but not authorization, the Consider changing the WebLogic "Group Membership Searching" setting from its default value of Unlimited to Limited, which will prevent WebLogic from performing unnecessary group membership searches that might cause high CPU usage on the LDAP repository server.

NOTE Verify the companion parameter "Group Membership Search Level" is set to its default: 0.

Adjusting the group membership searching setting

1. Logon to the WebLogic Administrative Console.
2. Select Security Realms.
3. Select myrealm.
4. Select Providers.
5. Select the [Provider Name].
6. Select Provider Specifics.
7. Click Lock and Edit.
8. Change the "Group Membership Searching" to Limited.
9. Click Activate Changes.

Handling many folders, documents, and objects

Consider the following tuning options when planning a FileNet P8 environment, developing efficient custom applications, and configuring Workplace.

NOTE These tuning options are not meant to replace any application-specific performance studies or any specific recommendations that you might have received from IBM for FileNet P8.

Folders and subfolders

Retrieving the contents of a folder can be a memory-intensive operation. The methods that retrieve folder contents return a list of all folders and objects contained in the folder and store that list in memory. If many containees exist, then the server can run out of memory and the application can fail or take an excessive amount of time to display the results. To avoid this problem, a custom application can use the search API methods to filter the list of sub-folders by using a query. For example, define a search query to return only those folder names that start with “Case 2”.

Workplace does not provide an option to limit the number of folders returned. Each time a user opens an object store or folder, Workplace retrieves a list of all the folders contained at that level along with several properties for each folder. For example, any time a user opens an object store, a list of all the top-level folders in that object store is retrieved from Content Engine for display. If the object store or folder contains an excessive number of items, both performance and usability can be adversely affected. Requiring the user to browse through many folders is not recommended for usability reasons.

While the FileNet P8 architecture does not impose a limit for the number of supported folders at a given level, you can optimize performance by using no more than 100-200 folders at a given level. For applications that require many folders, consider using the search APIs and filtering to limit the results. In addition to performance considerations, limiting the number of folders at any given level enables the use of a more user-friendly interface by providing a browsing-style interface that does not require the user to enter a filter condition.

Although the number of folders displayed per page in Workplace is configurable via Site Preferences, the number of folders or subfolders retrieved by Workplace is not configurable. That is, any folder opened through Workplace will result in the retrieval of all subfolders at that folder level, regardless of whether they are displayed through the user interface.

Documents and custom objects

As previously noted, retrieving the contents of a folder can be a memory-intensive operation for custom applications and for Workplace. Each time a user opens a folder, Workplace retrieves the entire contents of the folder along with several properties for each item. If the folder contains an excessive number of documents or custom objects, system performance can be adversely affected.

Workplace does provide filtering options for items other than folders. You can use the Folder Filter options in Site Preferences and User Preferences to limit the number of documents and custom objects retrieved by Workplace. The Folder Filter options limit how many documents and custom objects are retrieved per folder access, with a default value of 100 per object type. If a folder contains more items than specified by the Folder Filter options, then the remaining items can be retrieved by entering criteria in the Browse page Filter folder field to retrieve only those objects that match the criteria. Overhead from the initial retrieval operation can be significantly reduced by applying folder filtering. For example, set the filter criteria so that only document names added by a specific user or documents containing specific words in the title will be returned, rather than all items in the folder.

Possible tuning options

To minimize the affects on performance, consider the following:

- When filing in a folder, the application ensures that unique containment names are created for each object so that auto-unique naming is not overused. The Auto Unique setting when filing an object can seriously affect performance, and even cause timeouts, when it is used for many objects that are filed in the same folder using a duplicate containment name.
- When possible, limit the number of subfolders at every folder level. Create a hierarchy of subfolder levels rather than using many folders at the same level. Accessing additional levels of folders does not hinder the performance of the system, but accessing many folders at the same level does affect performance. In addition to performance considerations, the user experience can be improved by limiting the number of items displayed and by not requiring the user to enter filter criteria.
- Use the FileNet P8 search APIs to filter the search results when the application requirements call for large numbers of folders or objects within folders.

  Some operations can return a large result set that affects performance. You can limit the result set size by requesting only the object types or properties that are needed, rather than requesting all items.

  - On demand queries can be more efficient than using other methods to obtain the containees.
  - Use FROM and WHERE clauses to more precisely define the result set.
  - Use JOIN and SQL parameters to target the results.
  - Limit the number of results rows returned.
  - Use multiple queries. Several queries can be faster than a single method that retrieves all containees.

File storage area paths

Although EM has the option of creating file storage areas with local or UNC paths, it is recommended that you use a local path in all of the following configurations to help prevent performance degradation.

- A single Content Engine with a local or a SAN disk, or both
- Multiple Content Engines using GPFS (/vol/fs1 or F:\)
- Multiple Content Engines that use NFS and a consistent path on all UNIX systems (/vol/fs1)
- Multiple Content Engines that map UNC to a consistent drive on all Windows systems (F:\:-> \\fs1\fs1)

Content streaming

Inline Content Retrieval Limit and Put/Get Block Size parameters: Uploads and retrievals

Inline Content Retrieval Limit (a Content Engine parameter that is set in Enterprise Manager) and FileNet.Content.GetBlockSize KB (a client, such as Application Engine Workplace, Workplace XT, eForms and so on, parameter that is set in the filenet.properties file) work together to define the maximum amount of data that can be retrieved from a file store, database, or fixed content device in a single RPC.

The default setting for the Inline Content Retrieval Parameter is 1 MB with a maximum value of 10 MB. The FileNet.Content.GetBlockSize KB setting cannot exceed the amount defined for the Inline Content Retrieval Limit parameter. For example, if the Inline Content Retrieval Limit is 5 MB, GetBlockSize KB cannot be greater than 5 MB. If it is, the maximum amount of data received in each RPC will still be only 5 MB.

FileNet.Content.PutBlockSize KB is also a parameter in the filenet.properties file. It defines how much data can be sent to Content Engine in a single RPC. It has a preset maximum value of 10 MB.

Increasing the values of these two filenet.properties parameters from their default setting (1024K) makes large content uploads and retrievals faster, but it also increases the memory footprint on the Content Engine server and client. Too many concurrent large creations and retrievals might cause memory contention.

Decreasing the values of these 2 filenet.properties parameters lowers the memory footprint but also causes extra round trips to the server and reduces performance. For example, if you set these values to 512K and you retrieve 1024K of content, the server will make two round trips instead of one.

Adjust these values according to your use case and workload.

Adjusting the Content Retrieval Limit parameter

1. Log on as object store administrator.
2. Expand the Sites node.
3. Right-click a site, and select Properties.
4. Go to the Content tab.
5. Adjust the Inline Content Retrieval Limit (kilobytes) field.
6. Click OK or Apply when you are done.

Adjusting the Put/Get Block Size parameters when an application server hosts the Content Engine client

1. Place these lines in the filenet.properties file:

   FileNet.Content.PutBlockSize KB=1024

   FileNet.Content.GetBlockSize KB=1024

2. Save the file to the following location, for example:

   WebSphere – C:\Program Files\IBM\WebSphere\AppServer\profiles\ *your_profile*

   WebLogic – C:\bea\user_projects\domains\your_domain

   NOTE This location corresponds with the directory in which you start the application server.

ABOUT THE FILENET.PROPERTIES FILE The filenet.properties file provides a way to edit certain preset configuration parameters. Because the configuration parameters are preset, the filenet.properties file does not exist until a preset configuration value that is managed by this file requires an adjustment. If the filenet.properties files have already been created, it will be located in one of the following locations. If it has not been created, place it in one of the following locations:

- The current directory (Java system property user.dir)
- The user's home directory (Java system property user.home)
- The Java JVM home directory (Java system property java.home)
- The classpath

Adjusting the Put/Get Block Size parameters when the Content Engine client is a stand-alone Java application

For stand-alone Java client applications, a switch needs to be applied to the application executable. For example: java.exe –DFileNet.Content.PutBlockSize KB=1024.

Content upload destination

When content is stored in file stores and fixed stores, the content is uploaded into an inbound folder located in the same location as the local file directory, as specified for the file store. Each file store has only one inbound folder; and if a large volume of content uploads is expected, the inbound folder can become congested.

Better throughput can be achieved by providing multiple file stores, which enables content to be uploaded into multiple inbound folders. Multiple inbound folder effectively distributes the demands on the operating system administration of the file directory information and, if located on separate physical drives, the disk I/O. Multiple file storage areas can be made transparent by utilizing the Storage Policy feature. This feature provides load-balancing capabilities for content storage by transparently spreading the content across multiple storage areas.

When content is stored in database storage, content is uploaded to the folder that is designated for temporary file storage. By default, this is the same as the installation location of the Content Engine server. To lessen contention with other system processes and increase throughput, you can direct uploaded content to a dedicated portion of the file system. This can be done by using the Temporary Directory property on the Content Configuration tab in the object store properties.

Business Process Management

Process Engine

Client tuning – JVMs

Use the following tuning tips, according to the application server, if Process Engine:

- Becomes unresponsive after extended durations of high workload conditions.

  WebSphere – This failure manifests in the form of webcontainer threads in a 'Blocked State' due to SocketTimeouts.

  Other applications servers – Drops in throughput might be observed.

- Shows poor performance under high workload conditions.

Process Task Manager

Minimum and maximum heap size for high volume Component Manager activities

Consider increasing the minimum and maximum heap size to avoid out-of-memory errors during high volume Component Manager activities:

1. Open Process Task Manager.
2. Select Component Manager.
3. Click the Advanced tab.
4. Set the value of the JRE Parameters field to at least –Xms1024m –Xmx1024m.

Cached Entry Timeout parameter: User and group cache

The Cached Entry Timeout parameter specifies how often (in hours) Process Engine refreshes user and group cache. Caching user and group information causes Process Engine to make fewer calls to Content Engine (and the LDAP server) and provides better throughput. However, system response times might increase when the user and group cache refreshes. For example, caching systems that have roster security, queues with complex security specifications, or step definitions that refer to large user groups (several hundred members or more) could affect system performance. Therefore, because user and group cache refreshes can degrade system performance, controlling the frequency of refreshes also maximizes Process Engine performance.

The default value is 4 hours. Base this setting on how often the directory service changes (additions and deletions of users and groups, or modifications to group memberships).

If changes rarely occur on the Directory Service, set the Cached Entry Timeout to 0 and manually update the cache as needed. If changes occur frequently, set the Cached Entry Timeout parameter to the number of hours that Process Engine should execute a refresh.

Adjusting the Cached Entry Timeout parameter

The frequency that Process Engine executes a refresh is specified by the Cached Entry Timeout parameter value in the Advanced tab, in the Process Engine server's node properties, in Process Task Manager.

Manually refreshing the user and group cache

If the directory service rarely changes, and you want to update the cache only as necessary, set the Cached Memory Timeout to zero and manually update the cache as needed by using the flushusercache command.

Scheduling cache refresh

If you want cache refreshes to run during certain off-peak hours, set the Cached Entry Timeout parameter to 0 and use the appropriate operating system tools to run the vwtool env i f m y command as a scheduled task.

Flush Granularity option: Cache refresh

Refreshing the user environment cache is part of the routine Process Engine server operations. You control the refresh period by setting the Cached Entry Timeout parameter in Process Task Manager.

During the cache refresh period, a difference in system performance might be noticeable; throughput might drop and response times might rise. This situation can occur if, for example, the system contains queues with complex security specifications that refer to large groups (several hundred members or more). This situation can also occur with roster security, and in general when dealing with large groups in security specifications or step definitions.

The administrator can add that setting to the vwserver.ini file (flushGranularity= number of seconds) and the engine uses it after the Process Engine system is restarted.

The default value for flushGranularity is two seconds. The optimal setting for flushGranularity differs depending upon the complexity of ACLs on the queues, speed of the directory service, the setting for the user environment cache timeout (in the Process Engine Task Manager), and system workload.

- For a predictable but potentially more severe effect on the system, use a low value for flushGranularity.
- For a sporadic but dampened effect on the system, use a higher value for flushGranularity.

Process Simulator (PS) and Process Analyzer (PA)

Max Simulation Jobs parameter: Concurrent simulation runs

The FileNet P8 default settings might not be optimal for your hardware setup. For example, if your server hardware contains more processors than are configured by the default settings, you can increase the Max Simulation Jobs value so that the server can handle a larger number of concurrent simulation runs.

NOTE Increasing this value will most likely require an increase in the Java max memory.

Java max memory

Increase this value if you experience memory errors when running a simulation that has a high number of active work items at any moment or when concurrently running many simulations concurrently.

Tuning Process Analyzer for optimal performance

The performance of the PA is measured by the publishing rate. The higher the rate, the better the performance. Factors affecting the publishing rate are number of publisher threads, cache size, database, and publishing interval. Each factor is described below.

Publisher threads

The PA uses multiple threads to publish event data. On a multiple CPU system, increasing the number of publisher threads can improve the publishing rate, as long as the disk I/O and the database are not bottlenecks, the CPU utilization is not approaching 100%, and sufficient memory is available.

NOTE Increasing the number of publisher threads will most likely require an increase in the Java max memory.

Cache size

A low cache hit ratio can affect the publishing rate. If the cache-hit ratio is low (that is, 50%), increase the size of the corresponding cache until the ratio improves or until the memory limit is reached.

Databases

PA monitors and tracks the average time for updating the PA database, processing the work item cube, and processing the workflow cube. PA updates the database and processes the work item and workflow cubes (that is, the “work in progress” cubes) at the end of every publishing interval (default: every 5 minutes). If any of these three operations takes longer than one minute, you should monitor the PA database and tune it as necessary.

The processing of historical cubes can take a very long time (hours). You should configure the historical cube processing to occur at the earliest off-hour time to allow sufficient time to process the cubes; the default processing time for the historical cubes is 11:00 PM.

If the historical cube processing is taking too long, consider upgrading your disk hardware or reduce the size of the historical fact tables by using the PA Compression Wizard.

Publishing interval

In general, the PA processes events for the amount of time specified in the publishing interval; the default is every 5 minutes. After the publishing interval is reached, the PA spends time updating the PA database and process the “work in progress” cubes. When the updating and processing completes the PA switches back into its event-processing mode and the process repeats.

If PA has a large backlog of events to process and you have already tried the other performance tuning options, consider increasing the publishing interval to give PA more time to process events.

NOTE Increasing the publishing interval directly affects how often the “work in progress” cubes are updated and processed. Configure the publishing interval relative to your needs for having up-to-date “work in progress” cubes. For example, if it is sufficient to have the “work in progress” cubes updated every hour then change the publishing interval to 60 minutes.

IBM InfoSphere Enterprise Records

The performance tuning topics in this section apply specifically to IBM InfoSphere Enterprise Records.

Databases

The performance tuning topics in this section apply specifically to IBM InfoSphere Enterprise Records.

All databases

The performance tuning topics in this section apply specifically to IBM InfoSphere Enterprise Records regardless of the database type.

Disposition Sweep and Auto Destroy

Consider creating the following indexes to improve the performance of IBM InfoSphere Enterprise Records.

Class ------------ Database table name	Database column name	FileNet P8 property name	Reason to create index
Objectstore > FPOS > Document Class > Record docversion	*uxx_* cutoffdate	Cut Off Date	*(File plan object stores only)* Create this single-property index when performance degrades during the following actions: • Sweep performs cutoff processing • Sweep resets disposition • The user has changed to a different schedule
Objectstore > FPOS > Document Class > Record docversion	*uxx_* cutoffinherited from	Cutoff Inherited From	*(File plan object stores only)* Create this single-property index when performance degrades during the following actions. • Sweep performs cutoff processing • Sweep resets disposition • The user has changed to a different schedule
Objectstore > FPOS > Document Class > Record docversion	*uxx_* currentphase action	Current Phase Action	*(File plan object stores only)* Create this single-property index when performance degrades during the following actions: • Sweep performs a phase transition • Sweep launches a cutoff workflow • Sweep performs complex phase processing (such as during a change in a phase or during alternate retentions)

Class **------------** **Database table name**	**Database column name**	**FileNet P8 property name**	**Reason to create index**
Objectstore > FPOS > Document Class > Record docversion	*uxx_*current phase execution date	Current Phase Execution Date	*(File plan object stores only)* Create this single-property index when performance degrades during the following actions: • Sweep performs a phase transition • Sweep launches a cutoff workflow • Sweep performs complex phase processing (such as during a change in a phase or during alternate retentions)
Objectstore > FPOS > Document Class > Record docversion	*uxx_*exportstatus	Export Status	*(File plan object stores only)* Create this single-property index when performance degrades during the following actions: • Sweep performs a phase transition • Sweep launches a cutoff workflow • Sweep performs complex phase processing (such as during a change in a phase or during alternate retentions)
Objectstore > FPOS > Document Class > Record docversion	*uxx_*isdeleted	Is Deleted	*(File plan object stores only)* Create this single-property index when performance degrades during the following actions: • Sweep performs a phase transition • Sweep launches a cutoff workflow • Sweep performs complex phase processing (such as during a change in a phase or during alternate retentions)

Class **------------** **Database table name**	**Database column name**	**FileNet P8 property name**	**Reason to create index**
Objectstore > FPOS > Document Class > Record docversion	*uxx*_onhold	On Hold	*(File plan object stores only)* Create this single-property index when performance degrades during the following actions: • Sweep performs a phase transition • Sweep launches a cutoff workflow • Sweep performs complex phase processing (such as during a change in a phase or during alternate retentions)
Objectstore > FPOS > Document Class docversion	*uxx*_record information	Record Information	*(File plan object stores only)* Create this single-property index when performance degrades during the following actions: • Sweep performs a phase transition • Sweep launches a cutoff workflow • Sweep performs complex phase processing (such as during a change in a phase or during alternate retentions)
Objectstore > ROS > Document Class Docversion	*uxx*_record information	Record Information	*(Record object stores only)* For record object stores, create this single-property index during the following actions: • CBR searches from Hold Sweep • CBR searches from the Administration Web application. • Selecting the Get action for the Record Information property from Workplace or Workplace XT.

Actions properties

Consider creating the following index to improve the performance of the Get Info actions in IBM InfoSphere Enterprise Records.

Class ------------ Database table name	Database column name	FileNet P8 property name	Reason to create index
N/A docversion	*uxx_* vitalrecordreviewaction -------------------------- *uxx_* currentphaseaction	N/A	Create this composite-properties index to improve the performance of the Get Info actions.

Disposal instruction properties

Consider creating the following index to improve performance when navigating disposal instructions property pages in IBM InfoSphere Enterprise Records.

Class ------------ Database table name	Database column name	FileNet P8 property name	Reason to create index
Objectstore > FPOS > Other Classes > Folder docversion		Disposition Instruction (DoD and Base) or Disposition Schedule (PRO)	Create this single property index to improve response time when navigating disposal instructions property pages.

Location properties

Consider creating the following index to improve performance when retrieving retrieval physical records in IBM InfoSphere Enterprise Records.

Class ------------ Database table name	Database column name	FileNet P8 property name	Reason to create index
N/A docversion	*uxx_* barcodeid ------------------------- *uxx_* rmentitydescription ------------------------- uxx_rmentitytype	N/A	Create this composite-properties index to improve retrieval of physical records. **NOTE** This index is beneficial only when you store physical records. The index is not needed when you store only electronic records.

Class ------------ Database table name	Database column name	FileNet P8 property name	Reason to create index
Objectstore > FPOS > Document Class > Record docversion	*uxx*_location	Location	Create this single–property index to improve retrieval of physical records. **NOTE** This index is beneficial only when you store physical records. The index is not needed when you store only electronic records.
Objectstore > FPOS > Document Class > Record > Marker Record docversion	*uxx*_homelocation	Home Location	Create this single–property index to improve retrieval of physical records. **NOTE** This index is beneficial only when you store physical records. The index is not needed when you store only electronic records.
Objectstore > FPOS > Other Classes > Folder > RM Folder > Record Folder docversion	*uxx*_location	Location	Create this single–property index to improve retrieval of physical records. **NOTE** This index is beneficial only when you store physical records. The index is not needed when you store only electronic records.

Record type properties

Consider creating the following index to improve performance when navigating through record type property pages in IBM InfoSphere Enterprise Records.

Class ------------ Database table name	Database column name	FileNet P8 property name	Reason to create index
Objectstore > FPOS > Document Class > Record docversion		Type (DoD and Base) or Record Type (PRO)	Create this single property index to improve response time when navigating through record type property pages.

SupersedeBy property

Create this single-property index to improve the performance of the InfoSphere Enterprise Records user interface when retrieving values from the InfoSphere Enterprise Records enumeration property "Superseded Records."

Class ------------ Database table name	Database column name	FileNet P8 property name	Reason to create index
Objectstore > FPOS > Document Class docversion	*uxx_* supersededdate	Superseded By	Create this single-property index to improve the performance of the InfoSphere Enterprise Records user interface when retrieving values from the InfoSphere Enterprise Records enumeration property "Superseded Records". **NOTE** This is an object-value property (OVP).

Oracle

The performance tuning topics in this section apply specifically to IBM InfoSphere Enterprise Records when using an Oracle database.

Preventing performance degradation during INSUBFOLDER searches

Before you use the dbms_stats statistics collection mechanism with its default setting, review the benefits and affect of the custom setting options.

With InfoSphere Enterprise Records, certain default settings can cause query plans to degrade the performance of INSUBFOLDER searches. At a minimum, consider using the dbms_stats.gather_schema_stats command and the dbms_stats.gather_table_stats command to specify custom values for estimate_percent and method_opt, for example:

```
exec dbms_stats.gather_schema_stats(ownname => '<SCHEMA>', estimate_percent => 20,
method_opt => 'for all columns size 1', cascade => TRUE);
```

One of the following optional modifications might also improve performance:

- method_opt => "for all indexed columns"
- method_opt => "for all columns size <N>"
- method_opt => "for all indexed columns size <N>"
- Increase the value of <M> in the estimate_percent

NOTE Experiment with the values of estimate_percent and method_opt and use the smallest values that produce the best results.

Application servers

The performance tuning topics in this section apply specifically to IBM InfoSphere Enterprise Records regardless of the application server type.

Thread stack size

The “-Xss” switch value in the startweblogic.cmd specifies how much memory is allocated to the thread stack in kilobytes. Increasing this switch value can help prevent stack overflow errors that cause system fails when threads are computation intensive and need extra memory.

Over time, InfoSphere Enterprise Records user interface operations might fail because of a java.lang.StackOverflowError exception. If you encounter this problem, consider increasing the thread stack size, for example:

MEM_ARGS= -Xss2m

Set the minimum and maximum heap size for the Dynamic Hold Sweep execution

To avoid out-of-memory errors while performing bulk operations in a batch during Dynamic Hold Sweeps, consider increasing the heap size for the InfoSphere Enterprise Records sweeps to 1GB. These values are set in the RecordsManagerSweep.bat/sh file.

NOTES

- The minimum and maximum heap sizes are not set by default in the script.
- The following example is for a server with 2GB available RAM. Substitute the number 1024 for the amount of RAM that you want to allocate for your setup.

Set the minimum and maximum heap sizes

1. Identify the heap size entries.

   In the RecordsManagerSweep script, search for:

   - RecordsManagerSweep.bat :

   ```
   java -jar -Dwasp.location=./lib ./lib/%JarFile%
   ```

   - RecordsManagerSweep.sh :

   ```
   java -jar -Dwasp.location=./lib ./lib/$JarFile
   ```

2. Set the minimum and maximum heap size to at least 1024 MB for sweep execution, for example:

   - RecordsManagerSweep.bat :

   ```
   java -Xms1024m -Xmx1024m -jar -Dwasp.location=./lib ./lib/%JarFile%
   ```

   - RecordsManagerSweep.sh :

   ```
   java -Xms1024m -Xmx1024m -jar -Dwasp.location=./lib ./lib/$JarFile
   ```

3. Save the file.

Improving Sweep and Destroy throughput by tuning write threads

To improve sweep and destroy throughput, consider adjusting the multi-threaded Write feature thread count for each of the following processes so that they each equal the number of logical CPUs on the Content Engine server:

- Disposition Sweep
- Dynamic Holds Sweep
- Auto Destroy

Use the Disposition Sweep console, the Dynamic Holds Sweep console, and the Auto Destroy console respectively, to set the thread count for the Multi-threaded Write feature.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

U.S. Patents Disclosure

This product incorporates technology covered by one or more of the following patents: U.S. Patent Numbers: 6,094,505; 5,768,416; 5,625,465; 5,369,508; 5,258,855.

Product Number: 5724-R76, 5724-R80, 5724-R81, 5724-S19

Printed in USA

GC19-3394-00