Best Practices for Content Manager OnDemand Full Text Search

06/18/2020

Author: Brian Hoyt

Senior Software Engineer

IBM Content Manager OnDemand Development

Contents

Introduction

The introduction of full text search capabilities within Content Manager OnDemand (CMOD) provides customers the ability to intelligently search their content. Prior to the release of the Full Text Search (FTS) feature, the only option was to use the server based text search functionality of CMOD. This option is not ideal because it runs on the same machine as the CMOD server, could only deal with Advanced Function Presentation (AFP), Line, SCS, SCS-Extended, and PDF documents, and was limited to exact matches of the query string. FTS eliminates these limitations by introducing new components that integrate with CMOD to provide a complete solution for the full text indexing and searching of data.

The FTS feature of CMOD is based on the Apache Lucene text search engine library. This is the same text search engine used today by IBM in Db2, Content Manager and FileNet. The FTS feature ships with a new server, the Full Text Search Server, which handles the text extraction, indexing, and searching of data. This allows the processing of full text data to be offloaded to a machine other than your CMOD library and object servers. In addition to AFP, Line, SCS, SCS-Extended, and PDF, this engine can extract and index many other binary formats including Microsoft Office and XML. This text search engine also allows for more advanced queries. Customers can do wildcard searches, fuzzy (or similar) searches, proximity searches, and Boolean searches, just to name a few. A new component, called the FTS Exporter, is included with the server and handles the processing of all updates to the FTS Server. This new component, while it ships with the server, can be run on any supported CMOD platform.

As with some of the other CMOD features, FTS must be separately downloaded and installed. The installation is for the FTS Server component only and is supported on most CMOD platforms. Configuration and administration must be done at the CMOD level through existing OnDemand Administrator client interfaces and at the FTS Server which has its own set of command-line configuration utilities.

The FTS feature supports full text indexing of both new data as well as data that has already been indexed and loaded into CMOD. While configuring FTS for automatic full text indexing can be done through the administrative clients, indexing legacy data must be done through the CMOD command line utilities or the ODWEK Java application programming interfaces (APIs).

Full text searching is enabled through the CMOD folder and allows all CMOD client applications to take advantage of full text queries. Client applications can leverage this capability once the server configuration is complete. Several new CMOD folder field types have been defined in support of FTS. Search score, highlight, and summary are returned, aiding the end user in determining if the document is a good match.

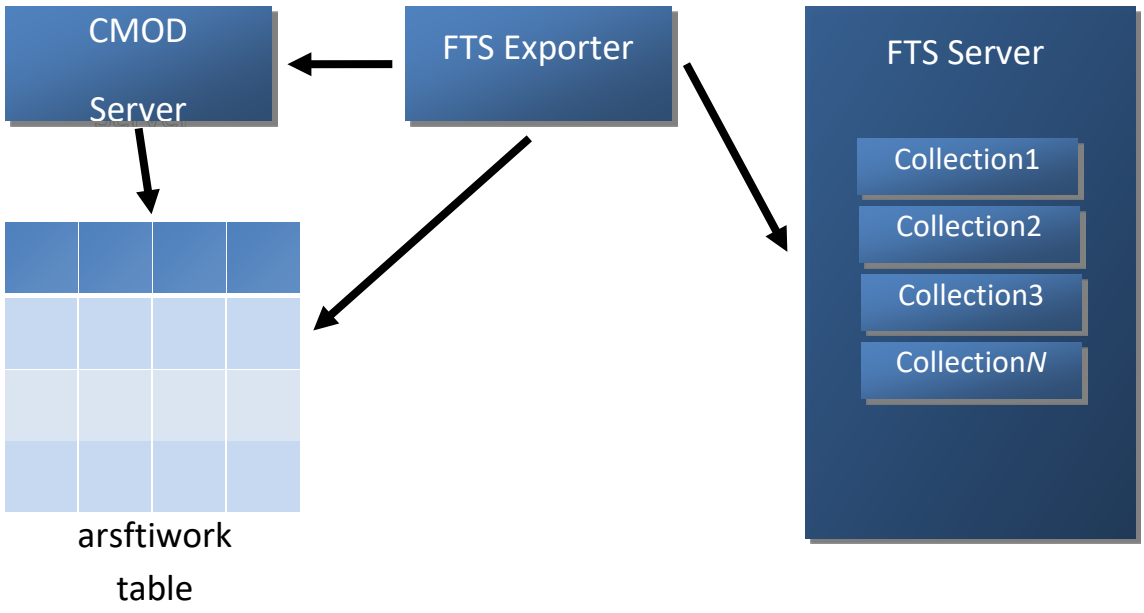
If you have problems when using FTS, enabling trace for both the CMOD Server and the FTS Server is the best tool for problem determination.

Architecture

In order to properly configure and administer an FTS system, a good understanding of the components involved and their interactions is required.

Figure 1 below shows a basic implementation of CMOD with the FTS Server. The components involved in the implementation are the CMOD Server, the FTS Server, and the Exporter.

Figure 1 Full Text Search components



FTS Server

The FTS Server provides a full document processing pipeline that includes text extraction from popular binary formats, a wide range of encoding support, and language processing in 23 languages. The flow of data during indexing depends on the configuration and environment. In a single server configuration, document content and properties are sent from the repository to the FTS Server. Then, documents are run through preprocessing steps before they are indexed. Preprocessing steps include text extraction, language identification, tokenization, and language analysis. After preprocessing is complete, documents are sent for indexing.

The first step to indexing is to extract the text from the document. This has to be done with text extraction engines. The FTS Server ships with text extractors for many varied document types, including Microsoft Office formats and XML. While the FTS Server contains text extractors for many data types,

AFP, Line, SCS, and SCS-Extended are not included. For these four data types, text extraction occurs within the Exporter. The resulting extracted text is then sent to the FTS Server.

Note: Documents of type image are not supported.

Once the text has been extracted it is now ready for preprocessing. During preprocessing, the FTS Server determines what language the document is in as well as tokenization and language analysis. After preprocessing has completed, the indexes are created for the documents. The indexes are stored into logical groupings called *collections*. See Segmentation below for more information on *collections*.

Content Manager OnDemand Server

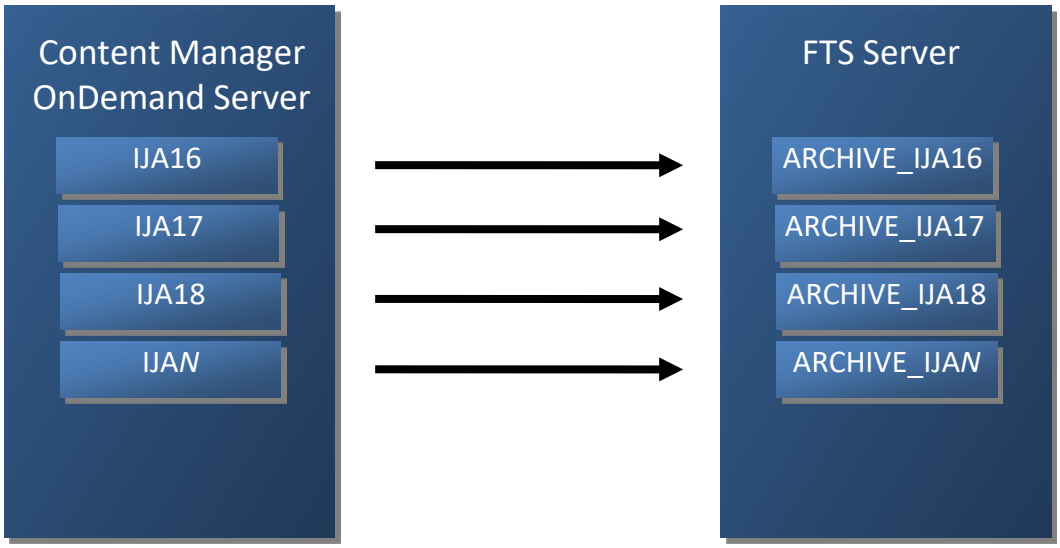
Indexing

The process of full text indexing a document can be lengthy. Because of this, an integration architecture was needed which would not introduce a significant amount of overhead to existing CMOD loading processes. The solution was to keep the process of full text indexing of the data separate from existing CMOD load processes. This was accomplished by creating a new table (**arsftiwork**) in the CMOD database which is used to hold FTS work items. When loading data, the CMOD load process simply adds a work item to this table. A new tool was developed to process these work items. This tool is called the FTS Exporter. The Exporter handles all tasks related to adding, updating and deleting documents to and from the full text index.

Segmentation

The data for an application group is stored in one or more *collections*. FTS uses the same data segmentation model as CMOD. Each time a new data table is created within CMOD, a new *collection* is created for the data's full text index. This means FTS *collections* maintain a one to one relationship with CMOD data tables. *Collections* are created with the following naming convention, *InstanceName_TableName*.

Figure 2 Content Manager OnDemand segment table to FTS collection mapping



This allows the FTS index to scale horizontally. During a query operation, CMOD can narrow the scope of documents that need to be searched. If the end user specifies a date range in addition to their full text search criteria, the CMOD segment tables are referenced to determine which *collections* need to be queried.

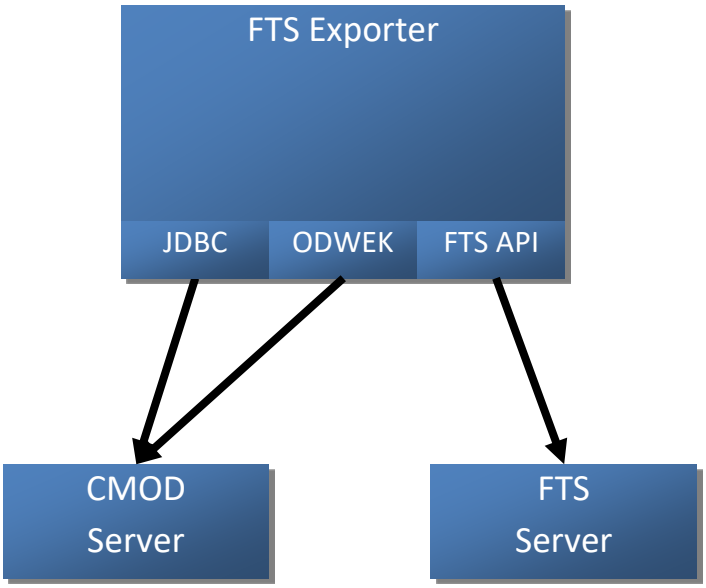
Searching

Searching for content using the full text index involves both servers, CMOD and FTS. When a full text search string is specified (see the Folder section on page 15 for more information on full text folder field types), a query is issued to the FTS Server on all *collections* that match the date range. If no date range has been specified in the query, then all *collections* for the specified application group are queried. Four new folder field types were added in support of FTS. **Score**, **Highlight**, **Summary**, and **Full Text Search**. At a minimum, **Full Text Search** must be specified, as this field is where the end user specifies their query string. **Score**, **Highlight**, and **Summary**, if created, result in the FTS Server returning more information for each matching document. **Score** is a value between 1 and 100 and is only relevant in relation to the other found hits. **Highlight** contains the context of the matching text (similar to Google), and **Summary** is the first 80 characters of the document.

Exporter

The FTS Exporter is a Java application that ships with the CMOD server.

Figure 3 Exporter protocols to servers



For data types other than AFP, Line, SCS, and SCS-Extended, the Exporter simply retrieves the specified documents from the server and sends them to the FTS server for processing. For AFP, Line, SCS, and SCS-Extended, the text from the documents must first be extracted. This process can be CPU intensive and therefore it is recommended that the Exporter be run from a machine other than the CMOD library server.

When started, the Exporter reads and processes work items from the **arsftiwork** table. The Exporter must have the following authorities on the **arsftiwork** table: **SELECT**, **UPDATE** and **DELETE**. Some work items require the Exporter to read from the **arsseg** table of CMOD. When accessing the **arsseg** table, the Exporter must have the following authorities: **SELECT**. All access to the CMOD tables is done through JDBC. Depending on the work item, the Exporter retrieves content from CMOD. To accomplish this, the Content Manager OnDemand Web Enablement Kit (ODWEK) Java APIs are used. Once the documents are retrieved, they are sent to the FTS Server for indexing. This is accomplished using the FTS Java client APIs. For all document types other than AFP, Line, SCS, and SCS-Extended, the data is sent as is; for AFP, Line, SCS, and SCS-Extended, the text of these documents must be extracted first.

Note: Only a single instance of the Exporter per CMOD instance is supported.

Installation

System requirements

Hardware requirements

The FTS Server must meet the following minimum hardware requirements:

Disk space

To determine the disk space that is required for a *collection*, add up the text size of all documents that will be indexed and multiply by four. To determine the text size for AFP and Line data documents, extract a sample document by using **arsdoc get** and then using **arsxafp** to determine the text size.

For example, on an IBM i system, the following qsh commands can be used for this process:

```
arsdoc get -c -h QUSROND -G INVNEW1 -X 5123-4-0-1FAA-
   20121018000000-20121018000000-5124 -o /tmp/INVNEW1.OUT

/QIBM/ProdData/OnDemand/bin/arsxafp /tmp/INVNEW1.OUT >
   /tmp/INVNEW1.TXT

wc /tmp/INVNEW1.TXT
     411    1178   19899 /tmp/INVNEW1.TXT
```

The text size of the document is 19899 bytes.

Memory

2 GB RAM

Optimal memory size is ten times the text size of the largest document. See **Disk space** requirements for a suggestion on sizing the largest document.

On AIX and Linux configure swap space of at least 8 GB.

Processor

1 processor

Important: Actual disk space, memory, and processor consumption depends on the number of *collections*, the number of documents per *collection*, the number of concurrently indexed *collections*, the required indexing throughput, the query load, and so on. For more information, see the topics in the Capacity planning section of this document.

Operating system

The FTS Server is supported on the following operating systems. See CMOD requirements for supported versions and hardware platforms:

- AIX
- SUSE Linux ES
- Red Hat Enterprise Linux
- Windows Server

Resource limit requirements on AIX and Linux

Before the FTS Server is installed on AIX and Linux ensure that operating system resource limits (ulimits) are set correctly.

The values of the **`fsize`** and **`nofiles`** parameters must be set to unlimited (-1) or 65536 to ensure correct system operation.

`fsize`
: Specifies the maximum file size.

`nofiles`
: Specifies the maximum number of files that are allowed for a process

The startup script for the FTS Server checks the values of the **`fsize`** and **`nofiles`** parameters. If the parameters are not set correctly, the startup script attempts to change them (for the current session only). If the script is unable to set the required parameter values, a warning is returned.

To manually check the values of the **`fsize`** and **`nofiles`** parameters and set their values if necessary:

1. Check the current values of the parameters by running the following commands:

Parameter	Command	Correct value
`fsize`	`ulimit -f`	`-1`
`nofiles`	`ulimit -Hn`	`-1` (AIX) 65536 (Linux)

2. If either parameter is not set correctly:
    a. Log on as root user.
    b. Open and edit the `/etc/security/limits` file. The file extension depends on the operating system.
    c. Save the file.
    d. Log out of the current session and log back on for the changes to take effect.

Capacity planning

Capacity planning helps determine the hardware resources for a particular implementation.

This information describes typical settings for an initial setup of the FTS Server.

Estimating disk consumption

FTS Server storage components include program files (JARs, JVM, and so on), server configuration files, collection configuration files, and collection indexes.

Program files

The following directories under the *FTS_Home* directory are not modified after installation and do not consume additional disk space:

- bin
- lib
- resource
- stellent

FTS Server configuration files consume a fixed amount of disk space plus a small amount of disk space per *collection*. The text index size grows as more documents are indexed into a *collection*.

Text index

Although the amount of disk space usage depends on the text in each document, this usage is approximately linear to the original size of the indexed data. Typically, the size of the index on the disk is between 50 - 150% of the original text size. For example, 100,000 documents of size 20 KB each can require about (100,000 x 20 KB x 75%) = 1500 MB of disk space.

The size of the index is not limited. However, it is important to consider that when large amounts of data are added to or removed from a text index, the text index structure is merged to improve query performance. The time for completion depends on the size of the index. Together with absolute throughput, which depends on data type and index format, this results in practical limits on the total text index size. For query performance, the biggest impact is the number of matching results, not the size of the text index.

During the indexing process, the server requires additional disk space for temporary storage. The maximal required disk space is approximately four times the total size of the text of the documents that will be indexed.

Index location

The location of the collection configuration files for all collections is determined by the **`defaultDataDirectory`** parameter, which is set by using the FTS Server command line configuration tool. By default, the collection configuration directory is *FTS_Home*`/config/collections`. The collection data (index) files are in a subdirectory under the collection configuration directory: **`defaultDataDirectory`**`/`*`collection_name`*`/data/text`.

When creating large indexes, specify a collection directory in a location with sufficient disk space. When creating multiple large indexes, consider storing them on separate or striped disk devices, in particular if concurrent index updates are scheduled.

Log files

FTS server generates trace and command-line tool log files in the *FTS_Home*`/log` directory by default.

Ensure that the target location has sufficient free disk space for the log files (at least 100 MB). Otherwise, the text search server stops logging and sends an error.

Heap memory consumption

Configuration considerations include heap memory consumption, queue sizes, and file size limits.

During indexing and searching, the FTS Server consumes heap memory for storing the indexed documents, preprocessing and indexing queues, and index memory structures. To optimize the performance of the FTS Server, it is very important that the maximal heap memory size in the JVM, the queue size, and file size limits are configured appropriately. You can configure the maximum heap size by using the configuration tool.

The maxHeapSize parameter sets the maximum heap size for the FTS Server. The default is 1.5 GB. This value must be a number between 1.5 GB and the amount of available memory.

When setting the maximum heap size to a value greater than 2 GB, file size limits for text, XML and binary documents need to be increased for new collections. For each 8.3 MB of heap memory over 2 GB, the values of the file size limits (60 MB by default) need to be increased by 1 MB (up to 400 MB):

```
60 MB + (heap memory - 2 GB)/8.3
```

Table 1 Examples of maximum heap size versus file size limit settings

Maximum heap size	File size limits
2 GB	60 MB
3 GB	180 MB
4 GB	300 MB

Use the configuration tool to specify the **maxBinaryTextSize**, **maxTextSize**, and **maxXmlTextSize** parameters.

Consider the ratio between the input and output queue's memory size and the heap memory. The queue size is determined by the memory consumption of the documents in the queue. If you intend to process large documents (for example, 20 MB each), consider increasing the queue memory size and increasing the heap size. The input queue size, as well as the output queue size, should not be greater than 5% of the maximum heap size.

Changing the configuration

The FTS Server configuration settings are modified by using the configuration tool. Run the `configTool` command-line program from the *FTS_Home*`/bin` directory. For changes to take effect, the FTS Server must be restarted.

Installing the FTS Server

The CMOD FTS Server can be downloaded from IBM Passport Advantage.

To install the FTS Server:

1. Copy the setup file to a directory on the computer where the FTS Server will be installed.

Operating system	Setup file
AIX	`odftsaix.bin`
SunOS	`odftssun.bin`
Linux	`odftslinux.bin`
zLinux	`odftslinux390.bin`
Windows	`odftswin.exe`

2. From the directory that contains the setup file, run the appropriate setup file:

Operating system	Run the installation program as follows:
AIX and Linux	Log in as a user with read, write, and execute permission for the installation directory and enter: `./setup_file_name -i console`
Windows	Log in as a user with administrator authority. In a command window, enter `./setup_file_name -i console`.

Note: On Windows, the FTS Server is installed as a service. You start and stop the FTS Server service by using the Microsoft Services window.

Removing FTS Server

Follow these instructions to remove the FTS Server.

To remove the FTS Server:

1. Start the appropriate program.

Operating system	Command
AIX and Linux	From the *FTS_Home*/`_uninst900` directory, enter the following command: `./uninstallodfts`
Windows	Uninstall the application from the Control Panel, Uninstall a program screen.

Configuration and administration

CMOD Server

In order to enable the CMOD server to support full text index and search, first download and install the FTS Server feature. See the section above for Installation. After completing the installation and configuration of the FTS Server, the next step is to enable the CMOD server for full text indexing and searching followed by configuring CMOD application groups and folders.

Windows configurator and the ARS.CFG file

The CMOD server must be enabled for FTS. This is done through the OnDemand Configurator (Windows only) or in the ARS.CFG file (all other platforms).

To enable FTS on a Windows server, select the "**Enable Full Text Index and Search**" check box on the Server (Advanced Options) dialog.

To enable FTS on any other platform, edit the ARS.CFG file and add the following parameter:

- ARS_SUPPORT_FULL_TEXT_INDEX=1
  A value of '**1**' enables the support.

In order to communicate with the FTS Server, a token must be specified. This token can be obtained from the FTS server by running the following command from the <FTS Server install directory>/bin directory:

`configTool.sh printToken` (The command is `configTool.bat` on Windows.)

This will display the communication token to be used for the server. This token must then be specified to the CMOD server. This is done in the Parameters section of the OnDemand Configurator (Windows only) or in the ARS.CFG file (all other platforms). Use the following parameter:

- ARS_FULL_TEXT_INDEX_TOKEN=flqBxTQ=

where flqBxTQ is the token that was returned from the `configTool.`

Application Group

You must configure each CMOD application group for which you plan to use full text index and search.

1. Enable FTS indexing for an application group.
   Update Application Group -> General tab -> Advanced -> Select 'Yes' under Full Text Index.
   Specify FTS Server name and port. Default port is 8191.
   Choose whether or not to automatically index all new loads.

2. Add FTS field.
   Update Application Group -> Field Definition tab.

Enter database field name -> Click add.
On the Field Information tab, change the data type of the field to "Small Int (2)".
Check the "Full Text Index" attribute.

3. Modify permissions.
   Update Application Group -> Permissions tab.
   Add "Full Text Index" permissions for those users and groups who need to index documents.

Folder

The CMOD folder must be configured before any full text searching can occur. Four new folder field types have been added in support of FTS.

- Full Text Index Search
  This field is required. It is the field the end users use to specify their full text search criteria. This field can only be queried.

- Full Text Index Score
  This field is optional. It represents the score of the hit, relative to the other matching hits. It can be queried and displayed in the hit list.

- Full Text Index Highlight
  This field is optional. It returns the text surrounding the matching text. It represents the context in which the text is found. This field can only be displayed in the hit list.

  Note: Highlight is not supported for XML documents.

- Full Text Index Summary
  This field is optional. It returns the first 80 characters of the document. This might or might not be useful depending on the data. For instance, bills and statements typically have identical text for headers and therefore this information could not be used to distinguish hits.

FTS Server

The FTS Server is configured using the command-line tools.

Command-line tools and utilities

The following command-line tools are installed in the *FTS_Home*`/bin` directory.

adminTool

Use the administration tool to manage collections, set trace options, and check the server version.

configTool

Use the configuration tool to set most system parameters and view system properties.

shutdown

Use the shutdown script to shut down the server.

startup

Use the startup script to start the server.

stopwordTool

Use the stop word tool to add or modify the list of stop words (common words that are not indexed).

synonymTool

Use the synonym tool to add or remove synonym dictionaries from indexes.

Configuration tool

The configuration tool is a command-line utility that is used to list and set server, query, and indexing parameters for the FTS Server.

Run the configuration tool from the *FTS_Home*`/bin` directory.

Operating system	Configuration tool
AIX and Linux	`configTool.sh`
Windows	`configTool.bat`

Restriction: Some parameters cannot be modified, and others cannot be modified when the FTS Server is running. Run the configuration tool with the **list** command and the **-details** argument to see which parameters can be modified.

Usage

`configTool` *`command`* `[-`*`command_options`*`] [-configPath` *`value`*`] [-locale` *`value`*`]`

Administration tool

The administration tool is a command-line utility that is used to manage collections, set trace options, and check the server version. For example, use it to check the status of a collection and delete unused collections. The FTS Server must be running for the administration tool to remove *collections*.

Run the administration tool from the *FTS_Home*`/bin` directory.

Operating system	Configuration tool
AIX and Linux	`adminTool.sh`
Windows	`adminTool.bat`

Usage

`adminTool administration_`*`command`* `[-configPath` *`value`*`] [-locale` *`value`*`] [-collectionName` *`value`*`] [-collectionPath` *`value`*`] [-logLevel` *`value`*`]`

Indexing

While both legacy and new data can be full text indexed, the processes for accomplishing each are very different.

New data

Indexing new data is simple with CMOD and FTS. When configured properly, the result of loading data into CMOD automatically creates work items in the **arsftiwork** table. See the Configuration and administration section for details on setting up your CMOD server to automatically full text index all new loads.

Note: If the FTS Exporter is not currently running, the work items are not processed until it is started.

Legacy data

Indexing legacy data requires the use of either the ODWEK Java APIs or the CMOD command line utility called **arsdoc**. Both of these interfaces to CMOD allow for legacy data to be full text indexed.

Content Manager OnDemand Web Enablement Kit Java APIs

The ODWEK Java APIs contain two new methods in support of FTS. The first is *ODFolder.FTIAddHits*(). This method has a single parameter which is a Vector of *ODHits*. The Vector of *ODHit* objects can be produced by using the *search*() methods of the *ODFolder*. All hits contained in the Vector parameter to *FTIAddHits*() are sent to the Exporter for full text indexing.

The second new method of the *ODFolder* object is *FTIReleaseHits*(). This method also takes a Vector of *ODHit* objects as a parameter and is used to remove the indexes from the FTS Server.

Both of these calls produce work items in the **arsftiwork** table.

Note: If the FTS Exporter is not currently running, the work items are not processed until it is started.

ARSDOC

The **arsdoc** command line utility has been enhanced with two new options. The first new option is **fti_add**. Parameters for this option control whether the resulting documents are queried via SQL, the **-i** parameter, or if an entire load is to be full text indexed, the **-X** parameter.

The second new option added to **arsdoc** is **fti_release**. This option takes the same parameters as **fti_add** in order to determine which documents should have their indexes removed from the full text index.

Both of these options result in work items being created in the **arsftiwork** table.

Note: If the FTS Exporter is not currently running, the work items are not processed until it is started.

Exporter

Processing of the work items in the **arsftiwork** table is done by the Exporter. The Exporter begins processing work items, starting with the oldest items. It continues processing these work items until the

table is empty. At this point it goes to sleep for a specified amount of time before waking up and looking for more work items.

Note: The Exporter is a Java application and therefore requires a Java Runtime Environment (JRE).

Usage

```
Usage: com.ibm.cm.od.fti.exporter.Export [index | configure] [options]
  Version: 10.5.0.0
     -configFile      <file> Used to specify parameters [optional]
     -dbEngine        <db engine> Database engine (DB2, MSSQL, ORACLE,
                      AS400)
     -dbHostname      <server> Database server hostname
     -dbPort          <port> Database Port
     -dbUser          <user> Database User Id
     -dbPassword      <passwd> Database Password
     -dbName          <db name> Database name
     -dbOwner         <db owner> Database owner
     -dbSSLParms      <SSL parameters> Parameters for JDBC SSL
                      connection
     -odInstance      <instance>
     -odUser          <user> OnDemand user ID
     -odPassword      <password> OnDemand user password
     -odInstallDir    <path> Path to where OnDemand is installed
                      On IBM i, must be set to /QIBM/UserData/OnDemand
     -odPASEDir       <path> IBM i only, must be set to
                      /QIBM/ProdData/OnDemand
     -pollDelay       <num seconds> Number of seconds between polling
                                    [optional]
     -ftiToken        <FTI authentication token> [optional]
     -tempDir         <path> Temporary work directory [optional]
     -numThreads      <number> Number of worker threads [optional]
```

Note: The pollDelay parameter should be set to a number less than the Content Manager OnDemand Session Inactivity Time Out value.

Note: The numThreads parameter can be set to a value between 1 and 6. This controls the number of threads that can process documents for full text indexing. If your full text index volumes are high and you are seeing idle CPU time on your FTS server and your Exporter server, you may want to increase this value. The default is 1.

Note: If the ftiToken parameter is not specified, the Exporter will use a default value. This default value might not work with your installation of the FTS Server. If this is the case, the proper token can be obtained by running the `configTool` command from the <FTS Server install directory>/bin directory. The format of the command is `configTool.sh printToken`. (On the Windows platform, the command is `configTool.bat`.) The communication token that is displayed by using that command should then be specified for the value of the ftiToken parameter.

The table below contains a list of the parameters required to connect to JDBC and later read/write to the database tables via JDBC. All of these parameters are required.

db engine	Required parameter (**dbEngine**) on command line. This is needed to properly construct the connection string. One of: *DB2*, *MSSQL*, *ORACLE*, *AS400*.
db host	Required parameter (**dbHostname**) on command line
db port	Required parameter (**dbPort**) on command line
db user	Required parameter (**dbUser**) on command line
db password	Required parameter (**dbPassword**) on command line
db name	Required parameter (**dbName**) on command line. When connecting to a z server this is the database location. When connecting to a MP server this is the database name. When connecting to an IBM i server this is the instance name.
db owner	Required parameter (**dbOwner**) on command line. When connecting to an IBM i server this is the instance name.
db SSL	Required parameter for DB SSL connections (dbSSLParms) on command line. For example, the Trust store location and password: *sslTrustStoreLocation=c:\temp\db2_ssl.jks;sslTrustStorePassword=CM0nd emand*

Table 2 Parameters for JDBC

A sample connection string for Db2 would look like: **jdbc:db2://**server1.abc**.com:60000/ARCHIVE**

The dbOwner is prefixed on all table access, such as ROOT.ARSFTIWORK.

The table below contains a list of the parameters required to connect to CMOD through ODWEK.

od instance name	Required parameter (**odInstance**) on command line. This is used to locate connection information in the ARS.INI/Registry.
od host	Read from required HOST variable in ARS.INI/Registry.
od port	Read from required PORT variable in ARS.INI/Registry.
use ssl	Read from optional SSL_CLNT_USE_SSL variable in ARS.INI/Registry.
ssl port	Read from SSL_PORT variable in ARS.INI/Registry.
ssl keyring file	Read from SSL_KEYRING_FILE variable in ARS.INI/Registry.
ssl keyring stash	Read from SSL_KEYRING_STASH variable in ARS.INI/Registry.
od user	Required parameter (**odUser**) on command line.
od password	Required parameter (**odPassword**) on command line.

Table 3 Parameters for ODWEK

Note: In order to name the collection properly in FTS, CMOD needs to read the SRVR_INSTANCE variable in the ARS.INI file (for all server platforms but Windows) or Registry (for Windows servers). It is an error if this variable is not found.

Sample Invocations

The Exporter has references to external jar files. These external jar files are installed with the CMOD server in the jars subdirectory, along with the Exporter jar file, and in the www/api subdirectory.

The only exception to this is the required database specific JDBC jar file(s). These jar files need to be added to the classpath either with the CLASSPATH environment variable or the –cp parameter to Java.

Note: IBM Db2 requires an additional license jar file, except on IBM i. Most database JDBC implementations only require a single jar file.

The Exporter uses log4j to control messaging to the console and to a log file for diagnostics. The messaging level and other parameters for log4j are specified in an XML configuration file. A default XML configuration file is shipped in the Content Manager OnDemand jars subdirectory, called `log4j2.xml`. The path where this file resides must be specified in the classpath when invoking the Exporter.

The Exporter can then be run by using the following command:

```
java –cp
/opt/IBM/ondemand/V10.5/jars/*;/opt/IBM/ondemand/V10.5/www/api/*;
/opt/IBM/ondemand/V10.5/jars;<path to jdbc jar files>
com.ibm.cm.od.fti.exporter.Export index –dbEngine DB2 –dbHostname
od1.acme.com –dbPort 60000 –dbUser admin –dbPassword RY6eE20&2q –
dbName ARCHIVE –dbUser ROOT odInstance ARCHIVE –odUser admin –
odPassword euYrT2q1 –odInstallDir /opt/IBM/ondemand/V10.5 –
pollDelay 60
```

where:

- `/opt/IBM/ondemand/V10.5/jars/*` - Expands to all jar files in that directory
- `/opt/IBM/ondemand/V10.5/www/api/*` - Expands to the ODWEK jar file
- `<path to jdbc jar files>` - Path to JDBC file(s)
- `/opt/IBM/ondemand/V10.5/jars` – Path to where log4j config file resides

Note: Alternatively, the location of the log4j xml file and its name could also be specified by using the following parameter to Java: "`-Dlog4j.configurationFile=path/to/log4j2.xml`".

Note: The ODWEK shared library still needs to be added to the operating system shared library path.

In order to reduce the number of parameters on the command line, and to encrypt the passwords, use the **`configure`** action of the Exporter to write parameters and values into a file. This file can then be specified on **`index`** invocations of the Exporter using the `-configFile` parameter. When using a configuration file, any passwords specified are encrypted before they are written. As an example, the following command can be issued to write all of the specified parameters to a configuration file named **`od1.cfg`**.

```
java –cp /opt/IBM/ondemand/V10.5/jars/*;<path to jdbc jar files>
com.ibm.cm.od.fti.exporter.Export configure –configFile od1.cfg –
dbEngine DB2 –dbHostname od1.acme.com –dbPort 60000 –dbUser admin
–dbPassword RY6eE20&2q –dbName ARCHIVE –dbUser ROOT odInstance
ARCHIVE –odUser admin –odPassword euYrT2q1 –odInstallDir
/opt/IBM/ondemand/V10.5 –pollDelay 60
```

Note: In order for the `configure` command to encrypt passwords, the *userid* and *password* specified for the **dbUser** and **dbPassword** must have JDBC access to the CMOD database.

Once this file is created, the invocation of the index function of the Exporter simply becomes:

```
java –cp /opt/IBM/ondemand/V10.5/jars/*;<path to jdbc jar files>
com.ibm.cm.od.fti.exporter.Export index –configFile od1.cfg
```

Additional parameters can be specified with the `index` command along with a configuration file. Parameters specified on the command line take precedence over the parameters loaded from the configuration file.

Search

All CMOD clients use the same process and procedure when searching the full text index. The query is first sent to the CMOD server for processing. If the application group being searched contains a segment date, and if the search criteria specified a date range, that range is used to narrow down which *collections* on the FTS Server need to be searched.

Syntax

The FTS Server supports a rich query language that enables searches such as fuzzy searches, proximity searches, weighted searches, and Boolean searches.

Terms and phrases for queries

Queries can contain terms and operators. A term is a single word such as "*united*". A phrase is a group of words that are contained in quotation marks, such as "computer software". Phrases are searched as exact expressions.

Table 4. Exact match search vs. terms without quotation marks

	Terms without quotation marks	**Exact match (phrases)**
Lemmatization	Done	Not done
Stop words	Removed	Not removed
Synonyms	Expanded	Not expanded
Boolean operators	Supported	Not supported (interpreted as search terms)
Operators: "-" and "%"	Supported	Not supported (interpreted as search terms)

Wildcard characters: "*" and "?"	Supported	Supported

When searching for a word, the base form of the word is also searched. For example, searching for *tests* or *testing* also finds the word *test*.

Boolean searches

Boolean operators allow terms to be combined through logical operators. The following Boolean operators are supported: **AND**, **OR**, **NOT** and "**-**".

Boolean operators must be specified in all uppercase characters. For example, when searching for documents about dogs or cats by specifying the OR Boolean operator, specify the query as `dogs OR cats`, not `dogs or cats`.

Restriction: An exclamation point (!) cannot be used in place of the word NOT.

Precede a term with a minus sign (-) to indicate that the term must be absent from a document for a match to occur. The minus sign acts as a filter to remove documents and must be associated with a query that returns positive results. For example, the following query returns documents that include the term *computer* and not the term *hardware*:

```
computer -hardware
```

Use parentheses to control the Boolean logic in a query. For example, the following query finds documents that contain either WebSphere or Lotus and website:

```
(WebSphere OR Lotus) AND website
```

Wildcard searches

FTS supports wildcard searches. You can place wildcard characters before, within, or after a term.

Use a question mark (?) to perform a single character wildcard search. For example, the following query finds documents that contain the terms, *mare*, *mere*, *mire*, *more*:

```
m?re
```

Use an asterisk (*) to perform a multiple character wildcard search. A multiple character wildcard search looks for 0 or more characters. Wildcard searches find regular characters, not special characters. For example, searching for `www.*.com` finds `www.ibm.com` but not `www.#.com`.

Attention: Using a multiple character wildcard (*) at the beginning of a search term can have a negative impact on the performance of a search query when many matching terms are found. There is a configurable limit on the number of terms that can be returned; an error is returned when the limit (1024 terms by default) is exceeded. To change this limit, use the configuration tool to set a new value for the **queryExpansionLimit** parameter.

Optional terms

Use a percent sign (%) to indicate that a search term is optional. For example, the following query finds documents that include the term *log* and optionally include the term *file*:

```
log %file
```

Fuzzy searches

A fuzzy search query searches for character sequences that are not only the same but similar to the query term. Use the tilde symbol (~) at the end of a term to do a fuzzy search. For example, the following query finds documents that include the terms analytics, analyze, analysis, and so on.

```
analytics~
```

An optional parameter can be used to specify the required similarity. Specify a value greater than 0 and less than 1. The value must be preceded by a 0 and decimal point, for example, 0.8. A value closer to 1 matches terms with a higher similarity. If the parameter is not specified, the default is 0.5.

```
analytics~0.8
```

Restriction: Special characters are not supported in fuzzy search queries.

Proximity searches

A proximity search finds documents that contain terms within a specified number of words of each other. Use the tilde symbol (~) to do a proximity search. For example, the following query finds documents that contain "IBM" and "WebSphere" within seven words of each other.

```
"IBM WebSphere"~7
```

Proximity search is supported for individual terms, not phrases. Also, a word after a sentence break is considered 10 positions apart from the last word of the previous sentence.

Restriction: Special characters are not supported in proximity search queries.

Weighted searches (boosting terms)

Follow a search term by a boost value to influence how documents that contain a specified term are ranked in the search results. Use the caret symbol (^) with a number (the boost factor) at the end of the term. For example, the following query finds documents that include the terms IBM and Germany and increases the relevance of these documents by a factor of five in the search results.

```
ibm Germany^5.0
```

Troubleshooting

FTS Exporter

When having issues with the Exporter, the logging level should be increased. This can be done by changing the level in the log4j configuration file (`log4j2.xml`). By default, only FATAL and ERROR messages will be written to the exporter log file. During problem determination set the 'Root' level to TRACE. Search the log4j2.xml file for the following line and replace "**ERROR**" with "**TRACE**":

```
<Root level="ERROR">
```

The log file name and location can be configured in this file as well. This is accomplished by changing the APP_LOG_ROOT property. Search the log4j2.xml file for the following line and change "**`./logs`**" to point to the directory you would like the exporter.log file to be written to.

```
<Property name="APP_LOG_ROOT">./logs</Property>
```

FTS Server

You can troubleshoot the FTS Server by configuring and viewing logs. The FTS Server generates logging information during server startup, indexing and searching. The log files contain configuration information, warnings, errors, and debugging information that can be useful for monitoring the server and troubleshooting specific issues. The command-line tools also generate log files. By default log files are stored in the *FTS_Home*`/log` directory.

Logging levels

Every message in the log file has an associated level that indicates the message type. Logging levels, in descending order of severity, are defined as follows:

SEVERE	Errors and exceptions that occur during the execution of the server. Typically, SEVERE messages include detailed information with the stack trace that can help identify the cause of the problem.
WARNING	Mild problems that might require the attention of an administrator, such as a missing value for a setting with a default value, or the truncation of a document during indexing.
INFO	Informational messages that are generated during system operation, such as messages about server configuration and the state of server startup.
FINE	Detailed messages for debugging purposes, such as request headers and parsed queries.
FINER	More details, such as the results of document parsing during indexing.
FINEST	The most detailed level. This includes all printouts, entries, and exits from various functions.

Viewing the logging level and log directory

The default logging level is `INFO`, which means that messages of levels SEVERE, `WARNING`, and `INFO` are generated. To view the location of the log files, run the configuration tool with the `list -`

`logFolder` command. To view the current logging level, run the administration tool with the `printLogLevel` command.

Changing the logging level

The logging level impacts the number of messages that are saved to the log files. To change the logging level at runtime, run the administration tool with the `configureTrace -logLevel` command.