DIVA Core

DFM (Drop Folder Monitor) User's Guide

Release 8.1

Revision 1.0

August 2021

telestream | DIVA

Copyrights and Trademark Notices

Contents

4 Administering, Operating, and Monitoring DFM

A Configuration Files and Examples

Glossary

Preface

This book describes the DIVA Core Drop Folder Monitor solutions, configuration, administration, and operations.

Audience

This document is intended for System Administrators and DIVA Core users.

Documentation Accessibility

For information about Telestream's commitment to accessibility, visit the Telestream Support Portal located at https://www.telestream.net/telestream-support/diva/support.htm.

Access to Telestream Support

Telestream customers that have purchased support have access to electronic support through the Telestream Support Portal located at https://www.telestream.net/telestream-support/diva/support.htm.

Related Documents

For more information, see the DIVA Core documentation set for this release located at https://www.telestream.net/telestream-support/diva/support.htm.

Document Updates

The following table identifies updates made to this document.

Date	Update
June 2020	Minor formatting changes.
February 2021	Rebranded document to Telestream
	Updated copyright notices

Date	Update
July 2021	Updated document for release 8.1 Updated graphics to reflect new terminology Removed redundant terminology Minor formatting adjustments Replaced Configuration Utility with DIVA Command Updated information for DIVA Core 8.1 release including renaming various components as follows: • *Actor* is now named ***Datahub*** • *Proxy Actor* is now named ***Proxyhub*** • *Production System* is now named ***Network*** • *Source/Destination* is now named ***Server***

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.
blue text	Blue text indicates a link to an outside source, or to another chapter, section, or glossary term in this book.

1

Overview

This chapter describes an overview of the DFM (*Drop Folder Monitor*) component of the DIVA Core system and includes the following information:

DFM Overview

Drop Folder Monitoring allows users and third party applications to deliver content to be archived by copying related files to a folder, an FTP server, or a CIFS share.

Note: Linux-based Datahubs do not support UNC paths for CIFS Source Servers and destinations.

DFM does not require an application, or user, to send the Archive command. Each time a file (*Single File Mode*), or set of files (*File Set Mode*), is placed in a monitored *Drop Folder*, DIVA Core automatically archives the files and creates the related object according to the rules set in the configuration file for that particular Drop Folder.

The flexibility of DFM enables a myriad of possible configuration combinations. Telestream has combined the most commonly used configurations into four basic *DFM Solutions* to make configuration and operations easier. If necessary, you can perform advanced DFM configuration to employ DFM beyond the four solutions described in this book.

DFM supports multiple automated operations. Example configurations have been included to aid in the configuration of specific workflows as follows:

dfm.conf.delete.ini
This configuration file contains the minimum configuration parameters to configure DFM for use as a file deletion solution.

dfm.conf.single.local.ini
This configuration file is a template for configuring DFM for automated local, single file based archive operations.

dfm.conf.ini
This configuration file is a template demonstrating how multiple workflows and Drop Folders can coexist in a single configuration file, and is modular by design.

DFM Features

Drop Folder Monitor is a component of DIVA Core that can be installed with the DIVA Core installation package.

DFM monitors designated Drop Folders and requests DIVA Core to archive objects found in these folders. Drop Folder Monitor can monitor up to 20 local folders, FTP folders, and CIFS folders; each with its own individual configuration. There are two types (*modes*) of Drop Folders; *Single File Mode* and *File Set Mode*.

For example, if the archive contains two media groups, *News* and *Movies*, then you must configure two Drop Folders; one for *News* and one for *Movies*. However, if File Set Mode is used (*or an advanced configuration setup*), you can use a single Drop Folder instead.

After archiving completes, the files are deleted from the folder they have been archived from.

Best practices dictate that files should be copied to the Drop Folder and not moved. After the configured time period, if the file is considered incomplete, it might be deleted. Copying the file, rather than moving it, ensures that a backup copy is available if an unwanted deletion occurs.

When the Drop Folder is configured on an FTP server, you must copy all content (*files and folders to be archived*) to the Drop Folder using FTP. If you copy the content to the Drop Folder using another process (*local copy through the operating system, remote desktop copy, and so on*), there is no guarantee DFM will process the content properly.

Single File Mode Drop Folders

You use *Single File Mode Drop Folders* to automatically archive objects containing a single file. This mode is convenient for MXF, GXF, and so on, with single file assets. There can be many single files in the folder, but each will only be archived as a single file object.

The *Object Name* is the file name including the extension. The *Object Category*, *priority*, and other archive command parameters are defined in a Drop Folder configuration file (*dfm.conf*). The configuration file contains the description of all folders to be monitored by DFM.

When a file appears in a configured Drop Folder, DFM monitors the file size. If the size has not changed after a preconfigured time threshold (*last modification time*), then the file is considered complete, and the archive is initiated. After the file is archived it is automatically deleted from the Drop Folder.

File Set Drop Folders

Caution: The last file created in the File Set Drop Folder must be the metadata file (*with the .mdf extension*). The presence of this file triggers the archive operation. If the metadata file is transferred *before* the actual data files, content loss can occur!

You use *File Set Drop Folders* to automatically archive objects containing more than one file, and is compatible with complex objects.

When monitoring a File Set Drop Folder, DFM checks for the presence of a metadata file (*.mdf*). This file describes the details of the content to be archived, and signals DFM that the content to be archived is ready. The archive process starts only after this file is available to DFM.

Using the metadata file allows specifying the *Object Name*, *Category* and user *Comments* on a per object basis.

Users (*or an application*) must first create a folder within the Drop Folder for the set of files intended to be archived as a single object in DIVA Core. The folder must have the -allow_delete_on_source and -r options configured. After the folder is created, the data files have to be copied to the folder.

The object archiving process is initiated after DFM has opened and parsed the metadata file. After the object is successfully archived, DFM deletes the folder, and all of the files within it, including the .mdf file.

Common DFM Solutions

Telestream offers the four most common DFM Solutions as described in the following sections. Select the solution, or solutions, that meet your requirements, and follow the configuration instructions for those particular solutions.

See Chapter 2 for details and basic configuration of each solution.

Basic DFM Solution for Single Mode Folders

This DFM solution can monitor local disks, FTP transfers, and CIFS folders. DIVA Core will archive each file found in the Drop Folder as a single object. If your requirements include only single files then this solution will work perfectly.

- Only single file objects are supported.
- Supported major configuration parameters include:
  - Manager
  - Drop Folder URL
  - Category Name of objects being archived
  - Server for the Archive request
  - Templates for *File Path Root* and *File Name* generation
  - *Media Name* for the Archive request

If the connection to the disk, FTP or CIFS is unavailable (*or disconnected*) for a period (*that is, when the fileReadyThreshold time has elapsed*), the file will be considered *complete* when it is actually *incomplete*.

You will only find the fileReadyThreshold parameter in the dfm.conf.ini, and not in the dfm.single.conf.ini configuration file. However, you can manually add this parameter to the dfm.single.conf.ini file if wanted, or necessary.

Basic DFM Solution for File Set Mode Folders

This DFM Solution offers more flexibility than the Basic Solution. This configuration can monitor local disks, FTP servers, and unsecured CIFS folders containing either single files or file sets. Secured CIFS can also be used if you run the service under a suitable user account.

File Sets include a metadata file containing the details of the files included in the set. Each metadata file found in the Drop Folder is parsed and a new object (*usually with more than one component*) is archived in the DIVA Core system.

- Single and multiple file objects are supported.
- Recursive Archive (*folder and files*) is supported.
- Supported major configuration parameters include:
  - Manager
  - Drop Folder URL
  - *Media Name* for the Archive request

Basic DFM Solution for Delete Mode

This DFM solution monitors only the local disk, and removes files that are obsolete. If a subfolder is empty for more than the configured allowed time period, it will also be removed. This solution is used with the other DFM solutions, and can be included in all DFM configurations.

Supported major configuration parameters include:

- Folder URL
- Interval specifying when the file will be deleted from the folder.

Basic DFM Solution for DIVA Connect

This DFM Solution monitors only the local disk and can use either Single Files or File Sets. Each Metadata File found in the Drop Folder is parsed and a new object (*usually with more than one component*) is archived in the DIVA Core System.

This solution is used with DIVA Connect systems that are configured with inter-site transfer capabilities. Contact your Telestream Sales Support Specialist for detailed information on various DIVA Connect Solutions available.

- Single and multiple file objects are supported
- Recursive Archive (*folders and files*) is supported
- Supported major configuration parameters include:
  - Manager
  - Drop Folder URL
  - *Media Name Pattern* for generation of the *Media Name* for the Archive request
  - *Server Name*
  - *Server Path*

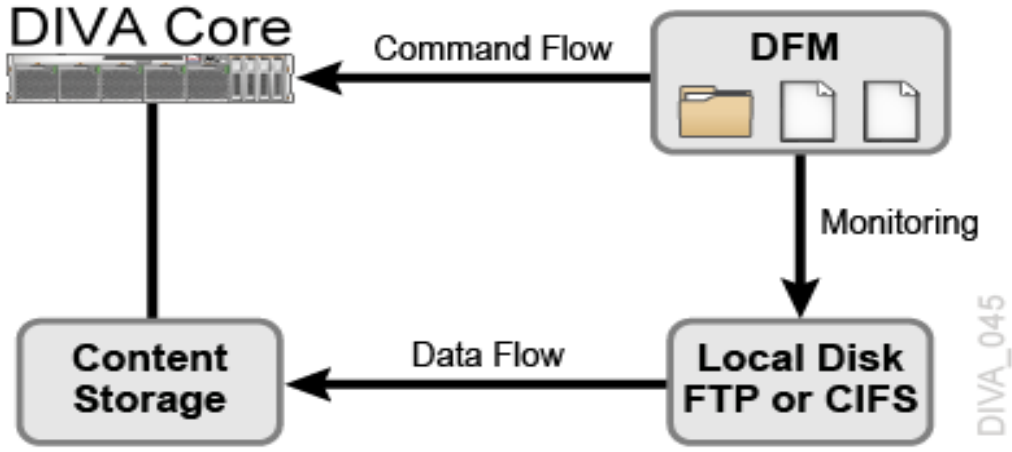
2

Solutions and Basic Configurations

This chapter describes details and configurations of the different DFM solutions, and includes the following information:

DFM Solution for Single Mode Folders

This DFM Solution can monitor a local disk, FTP transfers, and CIFS folders. DIVA Core will archive each file found in the Drop Folder as a single object. As files are dropped into the folder they are archived as single files into DIVA Core. Multiple files can be in the Drop Folder at the same time, but they will be handled by the DFM one at a time.



Supported Servers include:

- Local disks
- FTP folders

  When the Drop Folder is configured on an FTP server, you must copy all content (*files and folders to be archived*) to the Drop Folder using FTP. If you copy the content to the Drop Folder using another process (*local copy through the operating system, remote desktop copy, and so on*), there is no guarantee DFM will process the content properly.
- CIFS folders

  Linux-based Datahubs do not support UNC paths for CIFS Source and Destination Servers.

The following are additional configuration parameters and functionality:

- Only single file objects are supported.
- Supported major configuration parameters include:
  - Manager
  - Drop Folder URL
  - Category Name of objects being archived
  - Server for the Archive request
  - Templates for *File Path Root* and *File Name* generation
  - *Media Name* for the Archive request

Configuring the Basic DFM Single Mode Folder Solution

In the basic DFM solution, DIVA Core will archive each file found in the Drop Folder as single object. You use the following procedure to configure this DFM solution for a Single Mode Folder. *Editing the DFM configuration file is necessary to make the following changes*.

1. Rename the dfm.conf.single.local.ini file to dfm.conf.

   The file is located in the %DIVA_HOME%\Program\conf\dfm\ folder.
2. Specify the *Host* and *Port* using <managerConnetion> as follows (*using your parameters*):

   ```
   <managerConnetion>
    <address host="localhost" port="9000"/>
   </managerConnetion>
   ```
3. Specify the *Windows Service Name* using <serviceName> as follows:

   ```
   <serviceName>DIVA CoreDFM</serviceName>
   ```

   You must write this parameter on one line, and there must not be any symbols or empty spaces at the beginning of the line.
4. Specify the *Folder URL* using <url> as follows (*using your URL*):

   ```
   <url>file:///c:\DROPFOLDER\</url>
   ```
5. Specify the *Category Name* of objects to be archived using <categoryName> as follows (*using your Category Name*):

   ```
   <categoryName>Category</categoryName>
   ```
6. Specify the *Servers* to use in Archive requests using <sourceDestinationDIVAName> as follows using your Server name:

   ```
   <sourceDestinationDIVAName>DISK</sourceDestinationDIVAName>
   ```

7. Specify the *File Path* to the folder containing the files DIVA Core will archive using <archiveFilePathTemplate> as follows (*using your File Path*):

```
<archiveFilePathTemplate platform="DETECT" options="">
 URL_TO_FILE
</archiveFilePathTemplate>
```

8. Specify the template for the generation of the *File Name* using <archiveFileNameTemplate> as follows (*using your Template Name*):

```
<archiveFileNameTemplate platform="DETECT" options="">
 Filename.Ext
</archiveFileNameTemplate>
```

9. Specify the *Media Name* used in the archive request using <divaMediaName> as follows (*using your Media Name*):

```
<divaMediaName>Array1</divaMediaName>
```

Caution: If you do not complete the following Step 10 the installation will fail because it cannot find the trace file.

10. Rename the %DIVA_HOME%\Program\conf\dfm\dfm.trace.ini file to dfm.trace.
11. Install the DFM service using the following command:

```
%DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat install
```

12. Start the DFM service using the following command, or open the Windows Services panel and start the service from there.

```
%DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat start
```

DFM Solution for File Set Folders on Local Disk and FTP Servers

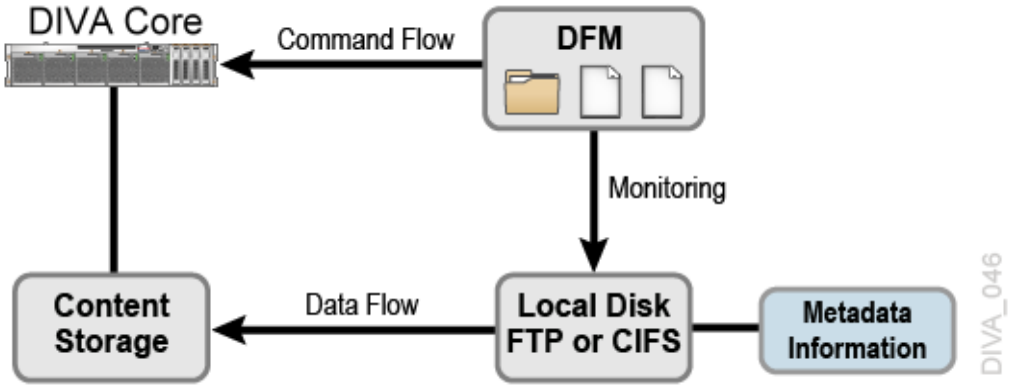
This DFM solution offers more flexibility than the basic solution. This configuration can monitor the local disk, FTP server, and CIFS folders that contain either single files or file sets.

DFM supports single and set based folder types. Each file dropped into a Single type folder results in a separate archive per file. File Set type folders provide a means of archiving objects comprised of multiple files. File Set mode archive operations require the use of metadata files, which contain the details of the file set comprising a specific archive operation (*see* Metadata File). Files for separate archive operations can coexist in a File Set mode folder, because the scope of files included in each archive operation is determined by the file list in each MDF file placed in the File Set mode folder.

Caution: The last file created in the File Set Drop Folder must be the metadata file (*with the .mdf extension*). The presence of this file triggers the archive operation. If the metadata file is transferred *before* the actual data files, content loss can occur!

AXF format files are, at the file system level, single files, but contain many files internally. You can archive AXF files into DIVA Core, but this requires the use of a File Set mode folder and an MDF file. The MDF file contains the list of files within the AXF file, and the target parent attributes of the object used when archived.

See Appendix A for sample metadata files in DFM format.



Supported Servers include:

- Local disks
- FTP folders

  When the Drop Folder is configured on an FTP server, you must copy all content (*files and folders to be archived*) to the Drop Folder using FTP. If you copy the content to the Drop Folder using another process (*local copy through the operating system, remote desktop copy, and so on*), there is no guarantee DFM will process the content properly.
- Unsecured CIFS folders

  Secured CIFS is possible if the Windows DFM Service is run under a suitable user account.

The following are additional configuration parameters and functionality:

- Single and multiple file objects are supported.
- Recursive Archive (*folders and files*) is supported
- Supported major configuration parameters include:
  - Manager
  - Drop Folder URL
  - *Media Name* for the Archive request

Configuring Recursive Archive

DFM Recursive Archive is enabled when the DFM folder is configured to use File Set Mode. In this mode the Archive File Set is defined in the specific object configuration .mdf file.

This feature defines a special character, asterisk, that can be used in the DFM metadata file (*the .mdf file*). Applying an asterisk to the end of the folder path causes all files under this path to be archived, but does not include subfolders.

Using the -r option in the configuration of the drop folder achieves different results than using the asterisk option. When you include the -r option in the request, the Datahub will recurse into all subfolders whether the asterisk is used or not.

Conversely, if the -r option is not used, the Datahub only selects the files in the immediate folder. In this case, use of the asterisk will still cause archiving of all files in the immediate folder action to occur. When using this option, be aware of the request and Server configurations to achieve the desired results.

You can configure the -r option in either of the two following locations:

- In the request ***Options*** field when creating the request. This method is only applicable to the current request. You enter the -r option in the ***Options*** field to use this method.
- In the Server ***Connect Options*** field in DIVA Command. This method is applicable to all requests to this Server. You include the -r option in the ***Connect Options*** field with your other request options to use this method.

Note: Recursive Archive does consider the use of the -file_order FILE_FIRST and -file_order DIRS_FIRST command options.

The following are several simple examples (*excerpts from the .mdf file*), which enables recursive archive for the specific folders and files:

Example 1:

```
<fileList>
 folder1/*
 test/folder2/*
 folder3/test.txt
</fileList>
```

The results of this configuration are as follows:

- Every file under folder1 and test/folder2 will be archived, but not subfolders.

  If you used the -r option in the request ***Options***, or on the Server Configuration, all subfolders and their files are also archived.
- For folder3/test.txt, only the test.txt file located in folder3 will be included.

Example 2:

```
<fileList>
 *
</fileList>
```

The results of this configuration are as follows:

- Every file in the specified folder will be archived, but not subfolders.
- If you used the -r option in the request ***Options***, or on the Server Configuration, all subfolders and their files are also archived.

Example 3:

```
<fileList>
 folder1/*
</fileList>
```

The results of this configuration are as follows:

- Every file under folder1 will be archived, but not subfolders.
- If the -r option was used in the request ***Options***, or on the Server Configuration, all folder1 subfolders and their files will also be archived.

Configuring the DFM Solution for File Set Folders

You use the following procedure to configure the DFM solution for a File Set folder, default filter for local disks, and FTP:

1. Rename the dfm.conf.set.local.ftp.ini file to dfm.conf.

The file is located in the %DIVA_HOME%\Program\conf\dfm\ folder.

2. Specify the *Host* and *Port* using <managerConnetion> as follows (*using your parameters*):

```
<managerConnetion>
 <address host="localhost" port="9000"/>
</managerConnetion>
```

3. Specify the *Windows Service Name* using <serviceName> as follows:

```
<serviceName>DIVA CoreDFM</serviceName>
```

You must write this parameter on one line, and there must not be any symbols or empty spaces at the beginning of the line.

4. Specify the <folderConfig> node using <type>. The folder should be the *File Set* type, and therefore the **set** value should be specified as follows:

```
<type>set</type>
```

5. Specify the *Folder URL* using <url> as follows (*using your URL*):

```
<url>ftp://diva:diva@localhost:21/dropfolder</url>
```

6. Specify the *Media Name* to use in the archive request using <divaMediaName> as follows (*using your Media Name*):

```
<divaMediaName>Array1</divaMediaName>
```

Caution: If you do not complete the following Step 7 the installation will fail because it cannot find the trace file.

7. Rename the %DIVA_HOME%\Program\conf\dfm\dfm.trace.ini file to dfm.trace.
8. Install the DFM service using the following command:

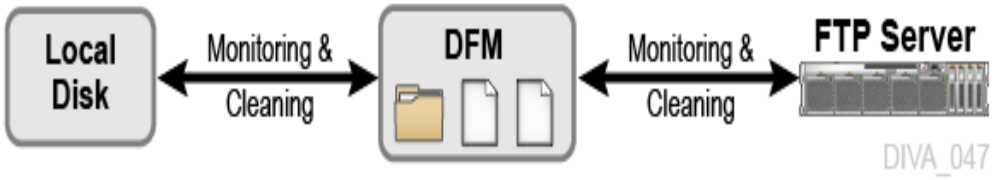
```
%DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat install
```

9. Start the DFM service using the following command, or open the Windows Services panel and start the service from there.

```
%DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat start
```

DFM Solution for Delete Mode

This DFM solution monitors the specified folder, and removes obsolete files. If a subfolder is empty for more than the configured allowed time period, it will also be removed. This solution is used in with the other DFM solutions, and should be included in all DFM configurations.



The following major configuration parameters are supported:

- Folder URL
- Interval specifying when files will be deleted from the folder

Configuring the DFM Solution for Delete Mode

You use the following procedure to configure the DFM solution for Delete Mode:

1. Rename the dfm.conf.delete.ini file to dfm.conf.

   The file is located in the %DIVA_HOME%\Program\conf\dfm\ folder.

2. Specify the interval that identifies when the file will be deleted from the folder using <fileDeleteThreshold> as follows (*the default value is shown*):

   <fileDeleteThreshold>86400</fileDeleteThreshold>

   - This value is the File Delete Threshold in seconds, and for folders only in *Delete Mode*.
   - This interval specifies when a file will be deleted from a DFM folder after copying the file to the folder is complete.
   - The state of completion is identified by the file size. As the file is copied, the size of the file will continue to grow until the copy is complete. Once the file size no longer increases, the system recognizes that the file copy operation is complete.
   - This value must be a positive integer value.
   - The default value of 86400 seconds equals 24 hours.

3. Specify the *Windows Service Name* using <serviceName> as follows:

   <serviceName>DIVA CoreDFM</serviceName>

   You must write this parameter on one line, and there must not be any symbols or empty spaces at the beginning of the line.

4. Specify the <folderConfig> node using <mode> as follows:

   <mode>Delete</mode>

5. Specify the *Folder URL* using <url> as follows (*using your URL*):

   <url>file:///c:\CLEAN_FOLDER\</url>

   Caution: If you do not complete the following Step 6 the installation will fail because it cannot find the trace file.

6. Rename the %DIVA_HOME%\Program\conf\dfm\dfm.trace.ini file to dfm.trace.

7. Install the DFM service using the following command:

   %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat install

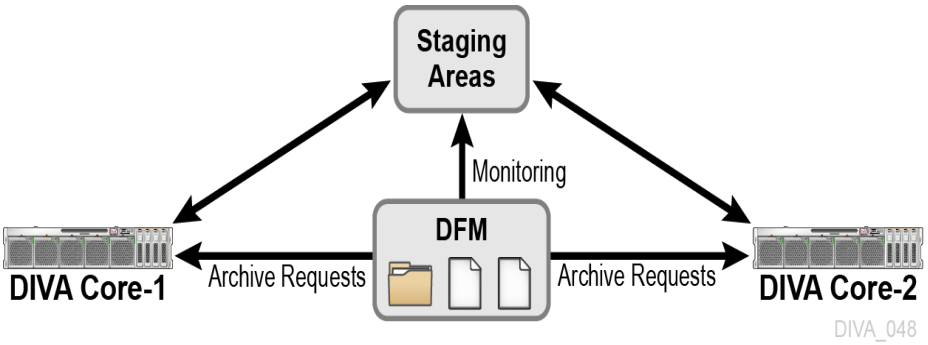
8. Start the DFM service using the following command, or open the Windows Services panel and start the service from there.

   %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat start

DFM Solution for DIVA Connect

This DFM Solution monitors a local disk, and can use either Single Files or File Sets. Each metadata file found in the Drop Folder is parsed, and a new object (*usually with more than one file*) is archived in DIVA Core. DIVA Connect systems configured with inter-site transfer capabilities use this solution.

Contact Telestream Support for detailed information about various available DIVA Connect solutions.



Supported Servers include:

- Local disks

The following are additional configuration parameters and functionality:

- Single and multiple file objects are supported.

  Files must be in non-complex objects.

- Supported major configuration parameters include:
  - Manager
  - Drop Folder URL
  - *Media Name Pattern* for generation of the *Media Name* for the Archive request
  - Server name
  - Server path

Configuring the DFM Solution for DIVA Connect

This DFM Solution monitors the local disk. Each metadata file found in the Drop Folder is parsed, and a new object (*usually with more than one file but not a complex object*) is archived in DIVA Core.

You use the following procedure to configure the DFM Solution for DIVA Connect:

1. Rename the dfm.conf.DIVA Connect.ini file to dfm.conf.

   The file is located in the %DIVA_HOME%\Program\conf\dfm\ folder.

2. Specify the *Host* and *Port* using <managerConnetion> as follows (*using your parameters*):

   ```
   <managerConnetion>
    <address host="localhost" port="9000"/>
   </managerConnetion>
   ```

3. Specify the *Windows Service Name* using <serviceName> as follows:

   ```
   <serviceName>DIVA CoreDFM</serviceName>
   ```

   You must write this parameter on one line, and there must not be any symbols or empty spaces at the beginning of the line.

4. Specify the <folderConfig> node using <type>. The folder should be the *File Set* type, and therefore the **set** value should be specified as follows:

   <type>set</type>

   There are two types of folders: *Single* and *File Set*. If the folder type should be *Single*, then you must specify the single value. If the folder type should be *File Set*, then you must specify the set value as shown in the example.

5. Specify the *Folder URL* using <url> as follows (*using your URL*):

   <url>file:///c:\DROPFOLDER\</url>

6. Specify the *Media Name Pattern* to use in the archive request using <divaMediaNamePattern> as follows (*using your Media Name Pattern*):

   <divaMediaNamePattern>/data/$GROUP/</divaMediaNamePattern>

7. Specify the *Server Name* using <sourceDestinationDIVAName> as follows (*using your Server Name*):

   <sourceDestinationDIVAName>SourceDestinationServer</sourceDestinationDIVAName>

8. Specify the *Server Path* using <sourceDestinationDIVAPath> as follows (*using your Server path*):

   <sourceDestinationDIVAPath>SourceDestinationServer</sourceDestinationDIVAPath>

   Caution: If you do not complete the following Step 9 the installation will fail because it cannot find the trace file.

9. Rename the %DIVA_HOME%\Program\conf\dfm\dfm.trace.ini file to dfm.trace.
10. Install the DFM service using the following command:

    %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat install

11. Start the DFM service using the following command, or open the Windows Services panel and start the service from there.

    %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat start

Contact Telestream Support for more information and details about the full solution and configuration.

Running Multiple Instances of DFM on the Same Computer

Running multiple instances of DFM on the same computer requires a separate installation of DIVA for each additional DFM instance. It is possible to configure only the DFM component in these additional instances. The <serviceName> parameter must be unique among all of the DFM instances. There is no special licensing required to use DFM in this configuration.

The following commands are for administration operations when running multiple DFM instances on the same computer:

Configuring the Service

The DFM configuration file is Program/conf/dfm.conf in each DIVA folder.

Upgrading the Service

You use the standard DIVA Core upgrade procedure to upgrade DFM.

Installing the Service

DFM module is installed with DIVA Core in the same installation package.

The command line syntax to install the service using a specific configuration file is %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat install -conf Program/conf/dfm/dfm.conf.

Uninstalling the Service

The command line syntax to uninstall the service using a specific configuration file is %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat uninstall -conf Program/conf/dfm/dfm.conf.

Starting the Service

You can execute %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat start -conf Program/conf/dfm/dfm.conf to start the service using a specific configuration file from the command line, or you can start it from the Windows Services panel.

Stopping the Service

You can execute %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat stop -conf Program/conf/dfm/dfm.conf to stop the service running a specific configuration file from the command line, or you can stop it from the Windows Services panel.

Service Logging

The log folder is %DIVA_HOME%\Program\log\dfm\.

The logging configuration is %DIVA_HOME%\Program\conf\dfm\dfm.trace.

Monitoring the Service

You can use the DFM logs, located in %DIVA_HOME%\Program\InterLink\log\dfm\, to obtain the current status.

3

Additional DFM Configuration Options

This chapter describes additional DFM configuration options and includes the following information:

Additional DFM Configuration

The following sections describe additional options for DFM configuration beyond the basic configurations described in Chapter 2.

Incomplete File Threshold: <incompleteThreshold>

If you copy the file into a *File Set Mode* folder, and during the designated incompleteThreshold seconds it was not archived, the file is marked as incomplete. The default value is 86400 seconds (*24 hours*).

<incompleteThreshold>86400</incompleteThreshold>

Processing of Incomplete Files: <incompleteProcessingStrategy>

This parameter identifies how DIVA Core processes incomplete files. Incomplete files are files that were placed into the Drop Folder without the metadata file (*for File Set folders*), and also files that cannot be archived after a specific number of attempts.

DIVA Core uses the following three parameters to identify incomplete files:

- incompleteThreshold
- maxRejectCountInitializing
- maxRejectCountProcessing

The following three options are available to use for the OPTION value in the parameter as shown:

None
This option tells the system to do nothing with an incomplete file.

Delete
This option tells the system to delete the incomplete file.

Rename
This option tells the system to rename the file so it is identifiable as an incomplete file.

<incompleteProcessingStrategy>OPTION</incompleteProcessingStrategy>

MDF Configuration Priority: <mdfConfigPriority>

Note: This parameter only applies to File Set Mode folders.

This parameter can have **Primary** or **Secondary** values, and determines how the system handles the file during the archive process. This can either be by using the metadata file parameters (***Primary***), or the folder specific configuration parameters (***Secondary***). **Primary** is the default value used when this parameter is empty or missing.

<mdfConfigPriority>Primary</mdfConfigPriority>

Additional DFM Configuration for DIVA Core Operations

The following sections describe additional DFM configuration options related to DIVA Core operations.

Delete Before Archive: <deleteBeforeArchive>

Caution: This optional parameter may lead to content deletion in the DIVA Core system, and must be used with caution.

This optional parameter specifies whether DFM should delete the object before the archive request for a specific folder. This parameter can be either true or false. The default value is false.

<deleteBeforeArchive>FALSE</deleteBeforeArchive>

Example:

An object is copied to the Drop Folder and fits into an established Storage Plan. The Storage Plan specifies that the object should be copied to another Server, and archived to a specific folder. Using this option, the object will be copied to the specified Server, but then deleted before the Archive action is performed. Therefore, the object is never actually archived.

Number of Files to Archive Wildcard: <numberFilesToArchiveWildcard>

This parameter defines a threshold on the number of files in the same folder. If the number of files in the folder is larger than the numberFilesArchiveWildcard parameter, DFM sends the archive request to the Manager using the asterisk wildcard as the File List parameter, instead of the actual list of files for the entire folder. The value must be a positive integer to specify the number of files, or if the asterisk wildcard is used in the request, all files will be archived.

```
<numberFilesToArchiveWildcard>*</numberFilesToArchiveWildcard>
```

Object Name Conversion Rules: <objectNameConvertRules>

Note: This parameter only applies to Single File Mode folders.

The Object Name Conversion Rules identify how DIVA Core translates file names into object names. Currently, only the *Simple* conversion method is implemented. The rule syntax for this method is case sensitive and described later.

When you specify a rule using **Name.Ext**, the object name will be the same as the file name. If you only specify **Name**, the file name will be the same as the original file name, but the extension will be removed. This applies to all of the following parameters:

Name.Ext
This value results in the original file name and extension; no conversion of the file name or extension takes place.

name (*all lowercase*)
This value results in the original file name being converted to *all lowercase*, and the file extension is removed.

NAME (*all upper case*)
This value results in the original file name being converted to *all upper case*, and the file extension is removed.

ext (*all lowercase*)
This value results in the file name being removed, the original file name being converted to the extension, and is *all lowercase*.

For example, file_name.ext will become only .ext.

EXT (*all upper case*)
This value results in the file name being removed, the original file name being converted to the extension, and is *all upper case*.

For example, file_name.ext will become only .EXT.

name.Ext
This value results in the original file name being converted to *all lowercase*, and the original file extension is *left intact*.

name.EXT
This value results in the original file name being converted to *all lowercase*, and the original file extension is converted to *all upper case*.

pre_name_suf.EXT
This value results in the original file name being converted to pre_(original_file_name)_suf, and the original extension is converted to all upper case.

pre_Name_suf
This value results in the original file name being converted to pre_(original_file_name)_suf, and the extension is removed.

For example, the following statement uses the *Simple* method, and leaves the file name and extension unchanged:

<objectNameConvertRule method="simple">Name.Ext</objectNameConvertRule>

Original Server: <originalServer>

In some configurations, the monitored folder is an intermediate location between the video server and the DIVA Core system. You can specify the originalServer parameter to ensure the original content location is set correctly in the DIVA Core system. The original server is displayed in the GUI, and stored in the database as the original content location, instead of the actual *Server* used to archive data from the intermediate location.

In the statement <originalServer>original_server</originalServer>, the value of original_server is the original content location designated by the original host name. For example, the statement <originalServer>VIDEO_SERVER_1</originalServer> identifies VIDEO_SERVER_1 as the original content location.

Original Path: <originalPath>

In some configurations the monitored folder is an intermediate location between the video server and the DIVA Core system. You specify the originalPath parameter to ensure the content path at the original location is set correctly in the DIVA Core system. The original content path is displayed in the GUI, and stored in the database as the path to the original content location, instead of the actual *Root Path* used to archive data from the intermediate location.

In the statement <originalPath>original_path</originalPath>, the value of original_path identifies the content path at the original_server location (*see the previous section describing original_server*).

Media Name: <divaMediaName> and <divaMediaNamePattern>

The <divaMediaNamePattern> is the *Media Name Pattern* used to select the medium for archive requests. Using this pattern allows the GROUP value to be identified.

In the statement <divaMediaNamePattern>folder1/$GROUP/folder3</divaMediaNamePattern>, folder1 and folder3 are actual folder names, while $GROUP indicates the system will identify the group from the .mdf file.

If the pattern was not recognized, then the <divaMediaName> value from the file set's corresponding .mdf file is used as the *Media Name* for the archived request when working with a File Set Mode folder. If the pattern is recognized, then the <divaMediaName> from the configuration file is used.

Additional DFM Configuration for the Monitored Folder

The following sections describe additional configuration options related to the DFM monitored folder.

MDF File Extension: <mdfExtension>

This option identifies the extension for the metadata file for each folder. In the following example, the metadata file extension will be mdf:

<mdfExtension>mdf</mdfExtension>

Server Path: <sourceDestinationDIVAPath>

Note: This parameter only applies to File Set Drop Folders. Because the Datahub and DFM could be on different machines, they might view the folder differently. For example, Datahub may see the folder as a local disk, while DFM may see it as an FTP folder (*or vice versa*).

This field defines the location of the *Drop Folder Root* on the Server. For a dedicated Server (*the Root Path of the Server in DIVA Command points to the root of DFM*), the *File Path Root* of the Server may already contain a path, and then you can leave this field empty.

In the statement <sourceDestinationDIVAPath>path</sourceDestinationDIVAPath>, the value of path is the full path to the root Server folder.

For example, the statement <sourceDestinationDIVAPath></sourceDestinationDIVAPath> identifies DROP_FOLDER as the full path to the root Server folder.

Sample Server Paths for Different Configurations

In File Set Mode, the SourceDestinationDivaPath is added to the relative file path. The following examples are for different common configurations.

See File Root Path and Single File Mode Advanced Configuration for more details and examples.

Example 1:

- **SourceDestinationDivaPath** = E:\DFM
- **Archive File Path Root** = E:\DFM\data\tape1\123456
- **Archive File List** = toto.mdf toto0.data folder1\toto1.data folder1\folder2\toto2.data
- Results
  - E:\DFM\data\tape1\123456\toto.mdf
  - E:\DFM\data\tape1\123456\toto0.data
  - E:\DFM\data\tape1\123456\folder1\toto1.data
  - E:\DFM\data\tape1\123456\folder1\folder2\toto2.data

Example 2:

- **SourceDestinationDivaPath** = F:\DFM\:
- **Archive File Path Root** = F:\DFM\data\tape1\123456
- **Archive File List** = toto.mdf toto0.data folder1\toto1.data folder1\folder2\toto2.data
- Results
  - F:\DFM\data\tape1\123456\toto.mdf
  - F:\DFM\data\tape1\123456\toto0.data
  - F:\DFM\data\tape1\123456\folder1\toto1.data
  - F:\DFM\data\tape1\123456\folder1\folder2\toto2.data

Example 3:

- **SourceDestinationDivaPath** = empty (*no value entered*)
- **Archive File Path Root** = data\tape1\123456

- **Archive File List** = toto.mdf toto0.data folder1\toto1.data folder1\folder2\toto2.data
- Results
  - data\tape1\123456\toto.mdf
  - data\tape1\123456\toto0.data
  - data\tape1\123456\folder1\toto1.data
  - data\tape1\123456\folder1\folder2\toto2.data

Example 4:

- **SourceDestinationDivaPath** = data
- **Archive File Path Root** = data\data\tape1\123456
- **Archive File List** = toto.mdf toto0.data folder1\toto1.data folder1\folder2\toto2.data
- Results
  - data\data\tape1\123456\toto.mdf
  - data\data\tape1\123456\toto0.data
  - data\data\tape1\123456\folder1\toto1.data
  - data\data\tape1\123456\folder1\folder2\toto2.data

Archive File Path Template: <archiveFilePathTemplate>

Note: this parameter is only applicable to Single File Mode Drop Folders.

This parameter specifies the rule for how the Root Path of the archive request will be generated. The platform parameter value can be one of the following:

- **WIN**
- **SOL**
- **CIFS**
- **DETECT**

The format of the value is a combination of folder names, separators, and the following keywords:

PARENTn

This keyword identifies the folder name where the definition is located. For example:

DFM_FILE_FULL_PATH = DFM_URL\PARENTn ...\PARENT_3\PARENT_2\PARENT_1\file_name.ext

URL_TO_PARENTm

This keyword identifies the relative path from DFM_URL to **PARENTm**.

URL_TO_FILE

This keyword identifies the relative path from DFM_URL to the file.

The following is an example using this parameter:

<archiveFilePathTemplate platform="DETECT" options="">URL_TO_FILE</archiveFilePathTemplate>

See File Root Path and Single File Mode Advanced Configuration for more details and examples.

Archive File Name Template: <archiveFileNameTemplate>

Note: This parameter is only applicable to Single File Mode Drop Folders.

This parameter specifies the rule for identifying how the file names of an Archive request are generated. The platform parameter value can be one of the following:

- **WIN**
- **SOL**
- **CIFS**
- **DETECT**

The format of the value is a combination of folder names, separators, and the following keywords:

PARENTn
This keyword identifies the folder name where the definition is located. For example:

DFM_FILE_FULL_PATH = DFM_URL\PARENTn ...\PARENT_3\PARENT_2\PARENT_1\filename.ext

URL_TO_FILE
This keyword identifies the relative path from DFM_URL to the file.

filename.ext
This keyword identifies the original file name.

filename
This keyword identifies the file name without the extension.

In the following example filename and ext represent the keywords of the pattern definition, not the sample file name and extension; filename.ext is not an actual file name.

<archiveFileNameTemplate platform="DETECT" options="">filename.ext</archiveFileNameTemplate>

See File Root Path and Single File Mode Advanced Configuration for more details and examples.

File Filter: <fileFilter>

Note: The <fileFilter> parameter is optional for the Drop Folder configuration. If you do not specify it in the folder configuration, DFM will archive any file in the folder without using any masks. Only one <fileFilter> can be specified per folder.

You use this parameter to filter files to archive. You can specify the type attribute as either **include** or **exclude**.

In *inclusion mode*, *no files are archived* if there are no filters defined. Each filter will allow archiving of files if they match specific rules. The file will only be archived if the file corresponds to one or more masks. If the file does not correspond to any masks then the file will be ignored by DFM, and not deleted or archived.

In *exclusion mode*, *all files are archived* if there are no filters defined. Each filter will restrict the archiving of files if they match specific rules. The file will not be archived if the file corresponds to one or more masks. If the file corresponds to any mask, the file will be ignored by DFM, and not deleted or archived.

You can define the filter as a pattern where the mask has a question mark, indicating any symbol, or an asterisk indicating any string, as a Regular Expression (*rexexp*), or both.

The following parameters are used within the <fileFilter> configuration parameter:

<mask>
This parameter specifies the pattern for the filter.

<regexp>
This parameter specifies the regular expression for the filter.

For details on Regular Expressions visit the following websites:

https://en.wikipedia.org/wiki/Regular_expression

http://www.regular-expressions.info/

Delete Parent and Content Directories: <deleteParentDirectoryAndContentDirectories>

This optional parameter specifies whether DFM should delete the parent and child folders of an MDF file. This is useful when a user places an object into a folder, and DFM is not set to delete the files and folders after all actions are executed on the object. Using this parameter, the object's files and folders are not left in the folder where the object was placed. This parameter can be either TRUE (*this is the default*) or FALSE.

In the following example, the folders *will not* be deleted:

```
<deleteParentDirectoryAndContentDirectories>FALSE</deleteParentDirectoryAndContentDirectories>
```

In the following example, the folders *will* be deleted:

```
<deleteParentDirectoryAndContentDirectories>TRUE</deleteParentDirectoryAndContentDirectories>
```

Folder Configuration Examples

The examples in the following sections have had all comments that are in the actual configuration files removed to make them easier to understand.

Example of a Local Folder in File Set Mode

The *Object Name Conversion Rule* does not apply to Set Mode Folders. Files will be acted on using the original file name and extension.

In the following configuration example:

- The local folder is c:\DROPFOLDER.
- The type of DFM folder is File Set.
- The Priority is 50.
- The Category is category1.
- The Incomplete Threshold is 60 seconds.
- The Server name is LOCAL1.

- The Server path is C:/DROPFOLDER.
- The Media Name is Array1.
- The Incomplete File Processing Strategy is Delete.

  All incomplete folders will be deleted.

```
<folderConfig>
  <url>file:///c:\DROPFOLDER</url>
  <type>set</type>
  <priority>50</priority>
  <categoryName>category1</categoryName>
  <incompleteThreshold>60</incompleteThreshold>
  <sourceDestinationDIVAName>LOCAL1</sourceDestinationDIVAName>
  <sourceDestinationDIVAPath>C:/DROPFOLDER</sourceDestinationDIVAPath>
  <divaMediaName>Array1</divaMediaName>
  <incompleteProcessingStrategy>Delete</incompleteProcessingStrategy>
</folderConfig>
```

Using this configuration, if you copy a set of files into the DFM monitored local folder named C:\DROPFOLDER, and it has 3 files in the set with the names file1.ext, file2.ext, and filesetdata.mdf, they will be archived into a single file in DIVA Core with the following parameters:

- The Server is LOCAL1.
- The Object Name is filesetdata.ext.
- The Category Name is category1.

If the metadata file is not present, the set will be considered incomplete and the files will all be deleted after 60 seconds.

Example of an FTP Folder in Single File Mode

When the DFM Drop Folder is configured on an FTP server, you must copy all content expected to be archived to the DFM Drop Folder using FTP. If you copy the content to the DFM Drop Folder by other means (*local copy through the operating system, remote desktop copy, and so on*) there is no guarantee of proper processing by DFM.

When DFM is used in a Linux environment to monitor an FTP folder, it must be configured as in the following example:

- User: diva
- User Home Directory: /ifs
- Folder to be Monitored: /ifs/folder1
- Correct DFM Configuration: ftp://diva:password@host_ip/folder1
- Incorrect DFM Configuration: ftp://diva:password@host_ip/ifs/folder1

```
<folderConfig>
  <url>ftp://diva:diva@localhost:21/dropfolder</url>
  <type>single</type>
  <priority>20</priority>
  <categoryName>category2</categoryName>
  <incompleteThreshold>120</incompleteThreshold>
  <sourceDestinationDIVAName>FTP1</sourceDestinationDIVAName>
  <sourceDestinationDIVAPath>/video1/data/</sourceDestinationDIVAPath>
  <divaMediaName>Array1</divaMediaName>
  <objectNameConvertRule method="simple">Name</objectNameConvertRule>
  <divaMediaNamePattern>/def/$GROUP/content</divaMediaNamePattern>
</folderConfig>
```

In this example:

- The FTP folder is ftp://diva:diva@localhost:21/dropfolder.
- The Type of DFM folder is Single.
- The Priority is 20.
- The Category is category2.
- The Incomplete Threshold is 120 seconds.
- The Server Name is FTP1.
- The Server Path is /video1/data/.
- The Media Name is Array1.
- The default Object Name Conversion rule for Drop Folders will remove the file extension.
- The divaMediaNamePattern is /def/$GROUP/content.

Using this configuration, if a single file is transferred to the FTP server called localhost, on port 21, with the login name diva, the password diva, in the DFM monitored FTP folder named dropfolder, with the file name file1.ext, it will be archived to a single object with the following parameters:

- Server: FTP1
- Object Name: file1
- Category Name: category2

If the file is not completely present, or if the archive process does not complete, the file will be considered incomplete and deleted after 120 seconds.

Example of a CIFS Folder in File Set Mode

The following example is for when you are using a CIFS folder for the DFM Drop Folder. The *Object Name Conversion Rule* does not apply to Set Mode Folders. Files will be acted on using the original file name and extension.

In the following example:

- The CIFS folder is \\divaserver\dropfolder.
- The Type of DFM folder is File Set.
- The Priority is 50.
- The Category is category3.
- The Incomplete Threshold is 60 seconds.
- The Server Name is CIFS1.
- The Server Path is empty.
- The Media Name is array1.
- The incomplete file processing strategy is Rename.

  All incomplete folders will be renamed to failed_ + (original_file_name).

```
<folderConfig>
  <url>file:///\\divaserver\dropfolder</url>
  <type>set</type>
  <priority>50</priority>
```

```
  <categoryName>category3</categoryName>
  <incompleteThreshold>60</incompleteThreshold>
  <sourceDestinationDIVAName>CIFS1</sourceDestinationDIVAName>
  <sourceDestinationDIVAPath></sourceDestinationDIVAPath>
  <divaMediaName>array1</divaMediaName>
  <incompleteProcessingStrategy>Rename</incompleteProcessingStrategy>
</folderConfig>
```

Using this configuration, if a set of files is copied into the DFM monitored CIFS folder named dropfolder, on the computer named divaserver, and has three files in the set with the names file1.ext, file2.ext, and filesetdata.mdf, they will be archived to a single object with the following parameters:

- Server: CIFS1
- Object Name: filesetdata.ext
- Category Name: category3

If the metadata file is not there, the set will be considered incomplete and the files will all be deleted after 60 seconds.

Service Log Configuration

Use the following parameters to specify the wrapper log file tracing configuration in Program/conf/dfm/dfm.conf.

wrapper.logfile.loglevel
This parameter specifies the level of messages to be logged.

wrapper.logfile.maxsize
This parameter specifies the maximum size of the log file in bytes, kilobytes, or megabytes.

wrapper.logfile.maxfiles
This parameter specifies the maximum log files count.

The valid logging levels are as follows:

NONE
This logging level produces no output.

FATAL
This logging level produces only messages for irrecoverable errors.

ERROR
This logging level produces all error messages.

WARN
This logging level produces all warning messages.

STATUS
This logging level produces all state changes.

INFO
This logging level produces all JVM output and informative messages.

DEBUG
This logging level produces detailed debugging information.

The following is a sample log file configuration:

```
# Log Level for log file output. (See docs for log levels)
wrapper.logfile.loglevel=INFO

# Maximum size that the log file will be allowed to grow to before
# the log is rolled. Size is specified in bytes. The default value
# of 0 disables log rolling. May abbreviate with the 'k' (kb) or
# 'm' (mb) suffix. For example: 10m = 10 megabytes.
wrapper.logfile.maxsize=10m

# Maximum number of rolled log files which will be allowed before
# old files are deleted. The default value of 0 implies no limit.
wrapper.logfile.maxfiles=4
```

Trace Log Configuration

Use the following parameters to specify the tracing configuration for DFM in the file Program/conf/dfm/dfm.trace:

LOGGING_TRACE_LEVEL
This parameter specifies the level of messages to be logged.

LOGGING_MAXFILESIZE
This parameter specifies the maximum size of the log file in bytes, kilobytes, or megabytes.

LOGGING_LIFETIME
The maximum lifetime of a logged item, in hours.

The following is a sample dfm.trace configuration:

```
-----------------------------------------------------
# Levels can be: DEBUG, INFO, WARN, ERROR, FATAL

LOGGING_TRACE_LEVEL=INFO
LOGGING_MAXFILESIZE=7MB
LOGGING_LIFETIME=50

-----------------------------------------------------
```

Advance DFM Configuration

Caution: The parameters in the following sections should not be altered unless required, and you are confident of having the ability to do so correctly. Misconfiguration of these parameters could result in an unstable, or non-functioning system, or both! Telestream recommends that you save the original configuration file to a different file name before making any modifications to the configuration (*for example, dfm.conf.original.ini*).

Full DFM Configuration File

There are several groups of parameters for the configuration file as follows:

Global DFM Parameters
These parameters specify common behavior of DFM.

Folders Default Parameters
These parameters specify some default values for folder specific parameters.

Folder Specific Parameters
These parameters specify folder specific parameters.

Tracing Specific Parameters
These parameters specify tracing parameters.

See Full DFM Configuration File (dfm.conf.ini) to view a sample of the full configuration file.

Global DFM Configuration

The DFM configuration XML provides the following elements for connections configuration:

<managerConnetion>
This parameter specifies how DFM connects to the Manager. You are not required to define these options unless the DIVA Core system being connected to requires connection parameters outside of the default configuration.

For example:

```
<address host="..." port="..." userName="..." password="..." applicationName="..."/>
```

host
This parameter is required and identifies the address of the Manager.

port
This parameter is required and identifies the port number where the Manager is listening. This must be a positive integer value.

userName
This parameter is optional and identifies the user name to log in to the DIVA Core system. If the value is NULL or empty, DIVA Core uses the default user name and privileges from the DIVA Core system for this application.

password
This parameter is optional and identifies the password for the user. The connection is rejected if the provided password is incorrect.

applicationName
This parameter is optional and identifies an arbitrary string that is interpreted by DIVA Core as an application name. This is used for information purposes only, and can be NULL or an empty string.

<serviceName>
This parameter identifies the DFM Service Name for the currently installed instance. Spaces are not allowed within the Service Name. The Service Name will be DIVA Core DFM - serviceName. If you leave serviceName empty, the Service Name DIVA Core DFM is used.

<foibStateFile>
This parameter identifies the file name of the File/Object Information Base dump.

<doibStateFile>
This parameter identifies the file name of the Delete File/Object Information Base dump.

<fileReloadingThreshold>
This parameter identifies the file reloading threshold in seconds. This interval specifies how often the File Monitor module reloads the state of the DFM folders. This setting must be a positive integer value between 1 and 10,000,000.

<fileReadyThreshold>
This parameter identifies the time interval, in seconds. If the last modification time to a file remains unchanged after this interval, the file is marked as *copied*, and then archived. This setting must be a positive integer value between 1 and 10,000,000.

<dirDeleteThreshold>
This parameter identifies the Directory Delete Threshold in seconds. This interval specifies how often the File Monitor module reloads the state of the DFM folders in Delete Mode, and commences with deletion. This setting must be a positive integer value between 1 and 10,000,000.

<fileDeleteThreshold>
This parameter identifies the File Delete Threshold in seconds. This interval specifies when a file will be deleted after copying is complete for DFM folders in Delete Mode. This setting must be a positive integer value between 1 and 10,000,000.

<maxArchiveOperations>
This parameter identifies the maximum number of simultaneous archive operations that DFM can perform. If the number of requests exceeds this threshold, pending requests must wait

until all active archive operations are complete. This setting must be a positive integer value between 1 and 10,000,000.

<maxRejectCountInitializing>
This parameter identifies the maximum number of archive retries that DFM can perform for requests rejected by DIVA Core during request initialization. Objects exceeding this limit are marked as incomplete. This setting must be a positive integer value between 1 and 10,000,000.

<maxRejectCountProcessing>
This parameter identifies the maximum number of archive retries that DFM can perform for requests rejected by DIVA Core during request processing. Objects exceeding this limit are marked as incomplete. This setting must be a positive integer value between 1 and 10,000,000.

<pollingIntervalOfRequestState>
This parameter specifies the frequency of object status polling in seconds. This setting must be a positive integer value between 1 and 10,000,000.

<repeatRequestDIVAThreshold>
This parameter identifies the interval in seconds, before DIVA Core's Archive module will resubmit rejected requests. This setting must be a positive integer value between 1 and 10,000,000.

Folders Default Configuration

The Folders Default configuration parameter values are used for folders in a folder specific configuration when a parameter is not specified. All default values can be overridden within the folder specific configuration. The Folders Default configuration accepts the following parameters:

<type>
There are two types of folders; *Single File* and *File Set*. To configure the folder for Single Files, specify **single**. To configure the folder for File Sets, specify **set**.

<mode>
This parameter identifies the DFM folder work mode. This can be **Archive** or **Delete**.

<priority>
This parameter identifies the default priority level of the requests for the folder. The folder priority range is 0 - 100. The value 0 is the lowest priority, and 100 is the highest.

<categoryName>
This parameter identifies the default value for Category Name. This value will be provided to DIVA Core if it is not overridden in the folder specific configuration, or in the File Set metadata file.

<incompleteThreshold>
This parameter identifies the Incomplete Threshold in seconds. This parameter specifies the maximum amount of time that a Single File, or File Set, can reside in the Drop Folder and not be archived before it is marked as incomplete. This must be a positive integer value between 1 and 10,000,000.

<sourceDestinationDIVAName>
This parameter identifies the Server Name to use in the Archive request.

<sourceDestinationDIVAPath>
This parameter identifies the Server Path to use in the Archive request.

<mdfExtension>
This parameter identifies the extension of the metadata file for each folder.

<originalServer>
This parameter identifies the Server to store in the object catalog if different from the actual Server. This is useful when you use an intermediate storage area for archiving, but want to store the original source for the data in the object catalog information.

<originalPath>
This parameter identifies the File Root Path to store in the object catalog, if different from the actual File Root Path. This is useful when you use an intermediate storage area for archiving, but want to store the original source for the data in the object catalog information.

<incompleteProcessingStrategy>
This parameter identifies the incomplete folder and file processing strategy. The value can be **None**, **Delete**, or **Rename**.

<numberFilesToArchiveWildcard>
This parameter identifies the threshold for the number of files in the same folder. If the number of files in the folder becomes larger than the numberFilesToArchiveWildcard value, DFM sends the Archive request to the Manager using the asterisk wildcard for the entire folder. This value must be a positive integer.

<divaMediaNamePattern>
The Media Name Pattern specifies the archive medium by extracting a part of the archive Root Path. *This option is mutually exclusive with the divaMediaName option*.

<deleteBeforeArchive>
This parameter specifies whether DFM should delete the object from DIVA Core before the archive operation, if the object already exists. You can set this parameter to either **TRUE** or **FALSE**. This is an optional parameter and the default value is **FALSE**.

<objectNameConvertRule method="simple">Name.Ext</objectNameConvertRule>
The value of <objectNameConvertRule> specifies the rule for the algorithm specified. In the current version of DFM, only the *Simple* conversion method is implemented. *See the Object Name Conversion Rules: <objectNameConvertRules> section for the* rule syntax for this method.

Folder Specific Configuration

All folders monitored by DFM must be described in the configuration file. Folder Specific configuration accepts the following parameters:

<url>
This parameter identifies the DFM folder URL. This is the monitored folder as viewed by the DFM process.

<type>
There are two types of folders; *Single File* and *File Set*. To configure the folder for Single Files, specify **single**. To configure the folder for File Sets, specify **set**.

<mode>
This parameter identifies the DFM folder work mode. This can be **Archive** or **Delete**.

<priority>
This parameter identifies the default priority level of the requests for the folder. The folder priority range is 0 - 100. The value 0 is the lowest priority, and 100 is the highest.

<categoryName>
This parameter identifies the default value for Category Name. This value will be provided to DIVA Core if it is not overridden in the File Set metadata file.

<incompleteThreshold>
This parameter identifies the Incomplete Threshold in seconds. This parameter specifies the maximum amount of time that a Single File, or File Set, can reside in the Drop Folder and not be archived before it is marked as incomplete. This must be a positive integer value between 1 and 10,000,000.

<sourceDestinationDIVAName>
This parameter identifies the Server Name to use in the Archive request.

<sourceDestinationDIVAPath>
This parameter identifies the Server Path to use in the Archive request.

<divaMediaName>
This parameter identifies the Media Name to use in the Archive request. *This option is mutually exclusive with the divaMediaNamePattern option*.

<mdfExtension>
This parameter identifies the extension of the metadata file for each folder.

<originalServer>
This parameter identifies the Server to store in the object catalog if different from the actual Server. This is useful when you use an intermediate storage area for archiving, but want to store the original source for the data in the object catalog information.

<originalPath>
This parameter identifies the File Root Path to store in the object catalog, if different from the actual File Root Path. This is useful when you use an intermediate storage area for archiving, but want to store the original source for the data in the object catalog information.

<incompleteProcessingStrategy>
This parameter identifies the incomplete folder and file processing strategy. The value can be **None**, **Delete**, or **Rename**.

<numberFilesToArchiveWildcard>
This parameter identifies the threshold for the number of files in the same folder. If the number of files in the folder becomes larger than the numberFilesToArchiveWildcard value, DFM sends the Archive request to the Manager using the asterisk wildcard for the entire folder. This value must be a positive integer.

<divaMediaNamePattern>
The Media Name Pattern specifies the archive medium by extracting a part of the archive Root Path. *This option is mutually exclusive with the divaMediaName option*.

<deleteBeforeArchive>
This parameter specifies whether DFM should delete the object from DIVA Core before the archive operation, if the object already exists. You can set this parameter to either **TRUE** or **FALSE**. This is an optional parameter and the default value is **FALSE**.

<objectNameConvertRule method="simple">Name.Ext</objectNameConvertRule>
The value of <objectNameConvertRule> specifies the rule for the algorithm specified. In the current version of DFM, only the *Simple* conversion method is implemented. *See the Object Name Conversion Rules: <objectNameConvertRules> section for the* rule syntax for this method.

File Root Path and Single File Mode Advanced Configuration

The DFM configuration receives file information from the remote or local media (*FTP server, shared directory, mounted drive, and so on*). Based on the file name and folder, DFM generates the File Root Path and file name.

Long path names are supported on both Windows and Linux. Absolute path names are supported on both Windows and Linux to a maximum of 4000 characters. Relative path names are limited to 256 characters on Windows systems (*only*).

Because requirements for the file name vary, the system can either include or exclude, the parent folder in the file. The requirements for the path also vary because there are several ways the Datahub can connect to the media.

You must specify the File Root Path in Single File Mode in the following format:

<archiveFilePathTemplate platform="DETECT" options="">URL_TO_FILE</archiveFilePathTemplate>

You must specify the file name in Single File Mode in the following format:

<archiveFileNameTemplate platform="DETECT" options="">filename.ext</archiveFileNameTemplate>

DFM monitors a DFM URL (*FTP, CIFS, Windows disk, or Solaris disk*). DFM obtains a File Full Path (*for FTP, CIFS, Windows disk, or Solaris disk*) for each incoming file.

File Full Path = DFM_URL\PARENTn ...\PARENT3\PARENT2\PARENT1\Filename.Ext

or

File Full Path = URL_TO_PARENTm\PARENTm ...\PARENT3\PARENT2\PARENT1\Filename.Ext

---

Note: Your operating system determines whether you use / or \ for these statements.

<archiveFilePathTemplate platform="DETECT" options="">x:\ccc\ddd</archiveFilePathTemplate>

The format of the value is a combination of folder names, separators, and keywords for **PARENTn**, **URL_TO_FILE**, and **URL_TO_PARENTm**.

<archiveFileNameTemplate platform="DETECT" options="">filename.ext</archiveFileNameTemplate>

The format of the value is a combination of folder names, separators, and keywords for **PARENTn**, **URL_TO_PARENTm**, **URL_TO_FILE**, **Filename.Ext**, and **Filename**, where the platform is **WIN**, **SOL**, **CIFS**, or **DETECT**.

Available keywords are as follows:

PARENTn

This keyword identifies the folder name where the definition is located. For example:

DFM_FILE_FULL_PATH = DFM_URL\PARENTn ...\PARENT_3\PARENT_2\PARENT_1\Filename.ext

URL_TO_PARENTm

This keyword identifies the path from DFM_URL to **PARENTm**.

URL_TO_FILE
This keyword identifies the relative path from DFM_URL to the file.

Filename.ext
This keyword identifies the original file name.

Filename
This keyword identifies the file name without the extension.

It is important to use the correct case in all of the following examples because the file name and extension parameters are case sensitive. For example, filename.ext is different than Filename.Ext.

Workflow A

The files are located in the root of DFM_URL.

- The Drop Folder is on CIFS
- The Datahub is on Windows

  DFM_URL = file:///\\aaa\bbb\ccc\ddd

  File Full Path = \\aaa\bbb\ccc\ddd\fffffff.txt
- The Datahub sees the DFM URL as x:\ccc\ddd
- The configuration will be as follows:

  ```
  ARCHIVE_FILE_PATH_TEMPLATE = "x:\ccc\ddd"
  ARCHIVE_FILE_NAME_TEMPLATE = "Filename.Ext"
  ```

- **Filename.Ext** must support both files without the extension, and specifies keeping the extension if there is one.

If some files are located in the root of the DFM_URL, and some are located in subfolders, the path length is not constant.

This can be addressed by using ARCHIVE_FILE_NAME_TEMPLATE = "URL_TO_FILE\Filename.Ext". When the **URL_TO_FILE** parameter is empty, DFM automatically removes the backslash character. An empty ARCHIVE_FILE_PATH_TEMPLATE value is supported.

Workflow B

The files are located one folder deeper than the root of the DFM URL, and the file names in the File List for the Archive request *must not* contain the folder names.

- The Drop Folder is on FTP

  When the DFM Drop Folder is configured on an FTP server, you must copy all content expected to be archived to the DFM Drop Folder using FTP. If you copy the content to the DFM Drop Folder by other means (*local copy through the operating system, remote desktop copy, and so on*) there is no guarantee of proper processing by DFM.
- The Datahub is on UNIX

  DFM_URL = ftp://user:password@aaa/bbb/ccc/ddd

  File Full Path = /aaa/bbb/ccc/ddd/eee/fffffff.txt
- The Datahub sees the DFM URL as /ccc/ddd
- The configuration will be as follows:

  ```
  ARCHIVE_FILE_PATH_TEMPLATE = "/ccc/ddd/PARENT1"
  ARCHIVE_FILE_NAME_TEMPLATE = "Filename.Ext"
  ```

- You must address this using the ARCHIVE_FILE_PATH_TEMPLATE = "/ccc/ddd/URL_TO_FILE" parameter value.

Workflow C

The files are located one folder deeper than the root folder of the DFM URL, and the file names in the File List for the Archive request *must* contain the folder names.

- The Drop Folder is on FTP

  When the DFM Drop Folder is configured on an FTP server, you must copy all content expected to be archived to the DFM Drop Folder using FTP. If you copy the content to the DFM Drop Folder by other means (*local copy through the operating system, remote desktop copy, and so on*) there is no guarantee of proper processing by DFM.

- The Datahub is on UNIX

  DFM_URL = ftp://user:password@aaa/bbb/ccc/ddd

  File Full Path = /aaa/bbb/ccc/ddd/eee/fffffff.txt

- The Datahub sees the DFM URL as /ccc/ddd
- The configuration will be as follows:

  ```
  ARCHIVE_FILE_PATH_TEMPLATE = "/ccc/ddd"
  ARCHIVE_FILE_NAME_TEMPLATE = "PARENT1/Filename.Ext"
  ```

- You must address this using the ARCHIVE_FILE_NAME_TEMPLATE = "URL_TO_FILE/Filename.Ext" parameter value.

Workflow D

The files are located in any number of directories deeper than the root of the DFM URL, and the file names in the File List for the Archive request *must not* contain the folder names.

- The Drop Folder is on Windows
- The Datahub is on UNIX

  DFM URL = file:///c:\aaa\bbb\ccc\ddd\

  File Full Path = c:\aaa\bbb\ccc\ddd\p1\p2\p3\p4\p5\fffffff.txt

- The Datahub sees the DFM URL as root /
- The configuration will be as follows:

  ```
  ARCHIVE_FILE_PATH_TEMPLATE = "/URL_TO_FILE/"
  ARCHIVE_FILE_NAME_TEMPLATE = "Filename.Ext"
  ```

Workflow E

The files are located any number of directories deeper than the root of the DFM URL, and the file names in the File List for the Archive request *must* contain two folder names.

- The Drop Folder is on Windows
- The Datahub is on UNIX

  DFM URL = file:///c:\aaa\bbb\ccc\ddd\

  File Full Path = c:\aaa\bbb\ccc\ddd\p1\p2\p3\p4\p5\fffffff.txt

- The Datahub sees the DFM URL as root /

- The configuration will be as follows:

  ARCHIVE_FILE_PATH_TEMPLATE = "/URL_TO_PARENT2/"
  ARCHIVE_FILE_NAME_TEMPLATE = "PARENT2/PARENT1/Filename.Ext"

Using the previous example with the assumption that the Datahub points to the folder named ddd, the following is true:

- DFM URL = file:///c:\aaa\bbb\ccc\ddd\
- File Full Path = c:\aaa\bbb\ccc\ddd\p1\p2\p3\p4\p5\fffffff.txt
- The Datahub sees the DFM URL as root /ddd/
- The configuration will be as follows:

  ARCHIVE_FILE_PATH_TEMPLATE = "/ddd/URL_TO_PARENT2/"
  ARCHIVE_FILE_NAME_TEMPLATE = "PARENT2/PARENT1/Filename.Ext"

Workflow F

If DFM monitors the video server using FTP, the file must have a file extension, and the Datahub will archive through the server interface without a file extension.

- The file must be located in a single folder level
- The Drop Folder is on FTP

  When the DFM Drop Folder is configured on an FTP server, you must copy all content expected to be archived to the DFM Drop Folder using FTP. If you copy the content to the DFM Drop Folder by other means (*local copy through the operating system, remote desktop copy, and so on*) there is no guarantee of proper processing by DFM.

- The Datahub is using a special interface

  DFM URL = ftp://user:password@aaa/bbb/ccc/ddd

  File Full Path = /aaa/bbb/ccc/ddd/eee/fffffff.txt

- The Datahub sees the DFM URL as "special_#*:/ddd"
- The configuration will be as follows:

  ARCHIVE_FILE_PATH_TEMPLATE = "special_#*:/ddd/URL_TO_PARENT1"
  ARCHIVE_FILE_NAME_TEMPLATE = "PARENT1/Filename.Ext"

- This is how the file extension is removed

4

Administering, Operating, and Monitoring DFM

This chapter describes DFM administration, operations, and monitoring, and includes the following information:

Starting, Stopping, and Restarting DFM Overview

You can start, stop, and restart the DFM service using the operating system services facilities, or from the DFM command line utility as follows:

```
dfm {start|stop|restart}
```

The command line options are as follows:

- start: tells the DFM service to start
- stop: tells the DFM service to stop
- restart: tells the DFM service to stop, and then start again

Starting and Restarting the DFM Service

When the DFM service starts, or restarts, DFM loads and validates the configuration file. If DFM detects any configuration issues, the process terminates and runs a diagnostics routine.

For successful starting of DFM, you must remove the .ini extension from the dfm.trace.ini file.

If the configuration validation is successful, DFM scans all of the configured Drop Folders, then checks the status of all objects initialized before DFM was last shutdown, and then updates the internal database with the current status of the objects.

After all of these steps have completed, the DFM status will be **Running**.

Running the DFM Service

Caution: The last file to be created in the File Set Drop Folder must be the metadata file (*.mdf*), which then triggers the archive operation. *If the metadata file is transferred before the actual data files, content loss can occur*!

When DFM locates files in a configured Drop Folder, it updates the internal database and requests DIVA Core to archive all files found as new objects. In order to avoid repeated archive requests, DFM continuously updates the archive operations status in the internal database.

DFM logs information about the incomplete files and calls the DFM File Manager module to move them to the *Trash* folder.

If DIVA Core requests fail, the DIVA Core Status module informs the internal database of the failure. If the number of unsuccessful request attempts reaches a preconfigured number, the object status is changed to **Could not be archived** and the object is marked as incomplete.

If the request completes successfully, the internal database is updated by the DIVA Core Status module. DFM removes the MDF file, and the File Set folder, for a File Set object.

Stopping the DFM Service

Executing the shutdown script will terminate the DFM service. All internal process are stopped before all archive operations are completed. After all of the modules are stopped, all internal statuses are saved to disk in the internal database before the DFM completes shut down.

Summary of DFM Administration

The following is an overview of the administrative functions available in DFM. There is no special licensing required to use DFM.

Configuring the Service

The DFM configuration file is %DIVA_HOME%\Program\conf\dfm\dfm.conf.

Upgrading the Service

You use the standard DIVA Core upgrade procedure to upgrade DFM.

Installing the Service

DFM module is installed with DIVA Core in the same installation package.

The command line syntax to install the service is %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat install.

Uninstalling the Service

The command line syntax to uninstall the service is %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat uninstall.

Starting the Service

You can execute %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat start to start the service from the command line, or you can start it from the Windows Services panel.

Stopping the Service

You can execute %DIVA_HOME%\Program\InterLink\dfm\bin\dfm.bat stop to stop the service from the command line, or you can stop it from the Windows Services panel.

Service Logging

The log folder is %DIVA_HOME%\Program\log\dfm\.

The logging configuration is %DIVA_HOME%\Program\conf\dfm\dfm.trace.

Monitoring the Service

You can use the DFM logs, located in %DIVA_HOME%\Program\InterLink\log\dfm\, to obtain the current status.

DFM Command-Line Interface

You can initiate commands using the operating system command-line. The syntax and options are as follows:

dfm {install|uninstall|start|stop|restart|status|version|help} [-conf|-f]

The commands are as follows:

install
This command installs the module as a system service.

uninstall
This command removes the executable as a system service.

start
This command starts the service.

stop
This command stops the service if it is currently running.

restart
This command stops, and then subsequently starts the service.

status
This command displays the current status of the service.

version
This command displays the module release information, and then exits

help
This command displays the help information, and then exits.

The options are as follows:

-conf
This option identifies the configuration file to load the DFM settings from.

-f
This option is an alternative to using the -conf option and performs the same function.

DFM Operations and Workflows

DFM constantly monitors the configured Drop Folders. When a file, or file set, is found in one of the configured Drop Folders, it is entered into the internal database (*the internal database is*

an XML file) with the file size, date, and time when the file was originally found. After waiting for a configured period, DFM will again check the size, time, and date of the file (*or file set*) to see if any changes have occurred.

If the file in a Drop Folder configured in Single File Mode satisfies the applicable criteria in the DFM folder configuration (*that is, it was not modified within the previous 60 seconds*), DFM marks the file as complete and sends an Archive request to the DIVA Core system. If the file has not fulfilled the criteria for a complete file after a configured allowable period, then it is marked as incomplete and set for deletion.

Caution: The last file to be created in the File Set Drop Folder must be the metadata file (*.mdf*), which then triggers the archive operation. *If the metadata file is transferred before the actual data files, content loss can occur*!

In File Set Folders, after the metadata file (*.mdf*) is located in the Drop Folder, DFM marks all of the files in the File Set as complete, and initiates the Archive request to DIVA Core. If the metadata file does not exist after a configured period, then all files are marked incomplete and set for deletion.

By default, DFM will try to archive any files located in the configured and monitored Drop Folders. If the files cannot be archived according to the DFM Folder configuration, they are marked as incomplete files. The different reasons for files and file sets being either archived, marked incomplete, or deleted are identified in the following sections.

The time interval for how often the Drop Folder is scanned is specified in the fileReloadingThreshold parameter in the configuration file for that specific folder. Any files or file sets that are not complete, archived, or both, within the specified time period are flagged as an unsuccessful archive, an incomplete object, or deleted.

The DFM internal database is stored as an XML file and updated after these different tasks take place. Each time DFM checks the configured Drop Folders, it compares the file date, time, and size to what was previously written to the internal database.

Source Server files cannot be deleted from a Drop Folder after an archive completes unless the appropriate Server is configured using the -allow_delete_on_source option.

See Chapter 3 for details on configuration options, features, and examples.

Algorithm and Workflow for Single File Mode Drop Folders

For Single File Mode folders, DFM uses a series of checks to identify if a file is complete and ready for processing, already processed, or incomplete.

To begin the process, DFM checks if the file size is equal to zero, or if the file has been marked as incomplete. If the file size does equal zero, or the file was already marked incomplete, then the file is skipped.

Next, DFM checks the date of the last modification to the file, and the size of the file. If the date of the last modification to the file, or the file size, has changed then the file is skipped. The lastModificationTime value used in this check is the local DFM time when the last modification of size, or date, was captured by DFM.

DFM checks if the difference between the current time (*local DFM time*) and the lastModificationTime is greater than the fileReadyThreshold parameter setting in the configuration file, whether the file is locked for processing, and whether the disk has free space available.

- If the difference is greater than the fileReadyThreshold parameter setting, the file is unknown to DFM, not locked for processing, and disk space is available, then the file will become *known* to DFM, and DFM tells DIVA Core to archive the file.

- If the difference is greater than the fileReadyThreshold parameter setting, the file is already known to DFM, not locked for processing, and should be deleted as incomplete, then DFM will delete the file.
- If the difference is greater than the fileReadyThreshold parameter setting, DFM knows about the file for longer than the incompleteThreshold parameter setting, and it is not locked for processing, then DFM will mark the file as incomplete.

Note: An *unknown* status indicates that all required file information was not added to DFM. DFM stores the date and time information for each file, but if it does not have all of the required information, the file will remain *unknown* to DFM. It will only become *known* after all file information has been added to DFM for processing.

Example 1

The following is an example workflow:

1. The file test.mpg was found by DFM, and the date and size was stored.
2. The file test.mpg was found by DFM again (*during the next cycle of folders investigation*), the date and size was changed from the previous check. Therefore, the file will not be added as *known* to DFM.
3. The file test.mpg was found by DFM again (*during the next cycle of folders investigation*), the date and size was not changed from the previous check, the difference is greater than the fileReadyThreshold parameter setting, the file is *unknown* to DFM, not locked for processing, and disk space is available. Therefore, the file was added as *known* to DFM.
4. The Archive request was sent to DIVA Core.
5. The archive was complete.

Example 2

The following is an example workflow:

1. The file test.mpg was found by DFM, and the date and size was stored.
2. The file test.mpg was found by DFM again (*during the next cycle of folders investigation*), the date and size was changed from last check. Therefore, the file will not be added as *known* to DFM.
3. The file test.mpg was found by DFM again (*during the next cycle of folders investigation*), the date and size was not changed from the previous check, the difference is greater than the fileReadyThreshold parameter setting, the file is *unknown* to DFM, not locked for processing, and disk space is available. Therefore, the file was added as *known* to DFM.
4. The Archive request was sent to DIVA Core.
5. The archive was *not* completed. Therefore, it was terminated by DIVA Core for some reason.
6. DIVA Core attempts to archive again after <repeatRequestDIVAThreshold> seconds, but the request was rejected again.
7. If the difference is greater than the fileReadyThreshold parameter setting, DFM knows about the file for longer than the incompleteThreshold parameter setting, and it is not locked for processing, then DFM will mark the file as incomplete. Otherwise, the system will continue to retry starting with Step 4.

Algorithm and Workflow for File Set Mode Drop Folders

For File Set Mode folders, DFM uses a series of checks to identify if a file is complete and ready for processing, already processed, or if it is incomplete.

To begin the process, DFM first checks to find out if the file size is equal to zero (*0*) or if the file has been marked as incomplete. If the file size does equal zero, or the file was already marked incomplete, then the file is skipped.

DFM next checks the date of the last modification to the file and the size of the file. If the date of the last modification to the file or the file size has changed, then the file is skipped. The lastModificationTime used in this check is the local DFM time when the last modification of size or date was captured by DFM.

DFM checks if the difference between the current time (*local DFM time*) and the lastModificationTime is greater than the fileReadyThreshold parameter setting in the configuration file, if the file is locked for processing or not, and if the disk has free space available.

If the difference is greater than the fileReadyThreshold parameter setting, the file is *unknown* to DFM and not locked for processing, then DFM tells DIVA Core to archive the file.

If the difference is greater than the fileReadyThreshold parameter setting, the file is *known* to DFM, not locked for processing, the file archived successfully, and the difference between the current time (*local DFM time*) and the archive time is greater than 5 seconds, or the file should be deleted as incomplete, then DFM will delete the file.

If the difference is greater than the fileReadyThreshold parameter setting, DFM knows about the file for longer than the incompleteThreshold parameter setting, and it is not locked for processing, then DFM will mark the file as incomplete.

Note: An *unknown* status indicates that all required file information was not added to DFM. DFM stores the date and time information for each file, but if it does not have all of the required information, the file will remain *unknown* to DFM. It will only become *known* after all file information has been added to DFM for processing.

Example 1

The following is an example workflow:

1. The file test.mdf was found by DFM, and the date and size was stored.
2. The file test.mdf was found by DFM again (*during the next cycle of folders investigation*), and the date and size was changed from the previous check. Therefore, the file *will not* be added as *known* to DFM.
3. The file test.mdf was found by DFM again (*during the next cycle of folders investigation*), the date and size was not changed from the previous check, the difference is greater than the fileReadyThreshold parameter setting, the file was *unknown* to DFM. Therefore, the file was added as *known* to DFM.
4. The Archive request is sent to DIVA Core.
5. The archive was completed.
6. The file test.mdf was found by DFM again (*during the next cycle of folders investigation*), the date and size was not changed from the previous check, the difference is greater than the fileReadyThreshold parameter setting, the file is *known* to DFM, not locked for processing, the file archived successfully, and the difference between the current time (*local DFM time*) and the archive time is greater than 5000. Therefore, the file was deleted by DFM.

Example 2

The following is an example workflow:

1. The file test.mdf was found by DFM, and the date and size was stored.
2. The file test.mdf was found by DFM again (*during the next cycle of folders investigation*), the date and size was changed from the previous check. Therefore, the file *will not* be added as *known* to DFM.
3. The file test.mdf was found by DFM again (*during the next cycle of folders investigation*), the date and size was not changed from the previous check, the difference is greater than the fileReadyThreshold parameter setting, the file is *unknown* to DFM. Therefore, the file was added as *known* to DFM.
4. The Archive request was sent to DIVA Core.
5. The archive was not completed, and terminated by DIVA Core for some reason.
6. DIVA Core attempts to archive again after <repeatRequestDIVAThreshold> seconds, but the request was rejected again.
7. If the difference is greater than the fileReadyThreshold parameter setting, DFM *knows* about the file for longer than the incompleteThreshold parameter setting, and it is not locked for processing, then DFM will mark the file as incomplete. Otherwise the system will continue to retry starting with Step 4.

Monitoring DFM

The following sections describe different methods for monitoring the DFM service.

Monitoring DIVA Connect from the Control GUI

The Control GUI is used for monitoring the requests initiated by DFM.

You execute %DIVA_HOME%\Program\GUI\bin\gui.bat to start the Control GUI.

The Control GUI enables viewing the status of the running requests, Datahubs, and disks. The *Objects* and *Requests* panels also offer search functionality. The Control GUI *Manager* screen displays the information in the following columns:

- Object ID
- Errors
- Type (*of request*)
- Status (*of request*)
- Step (*of DFM processing request*)
- Object (*name*)
- Category
- Comments
- Tape
- Progress (*of request*)
- Throughput (*Mb/s*)
- Data Scanned (*KB*)
- Third Party ID

- Instance

Use the Control GUI Requests panel to query completed DFM archive requests. You can search based on the Server. To make querying DFM archive requests easier, assign a dedicated Server exclusively for DFM (*possibly with the same parameters as another Server in the system*). However, note this technique will not work for DIVA delete requests issued by DFM.

Monitoring DFM using Logs

You use the DFM log files for status monitoring of DFM. The files are located in the %DIVA_HOME%\Program\InterLink\log\dfm\ folder. The log files contain detailed information about files and folders encountered by DFM, information about requests submitted to DIVA Core, files/folders marked as incomplete, and errors encountered in processing.

See Sample of the Full Trace Log File for detailed information about this file.

A

Configuration Files and Examples

This appendix includes default configuration files and configuration examples, and includes the following information:

Configuring Objects

All File Set objects are described by a companion MDF (*Metadata File*). You must specify object configuration in the MDF file using the following parameters:

priority
This parameter identifies the priority of the DIVA Core request for this object. Object priority can be in the range of 0 - 100. The value 0 is the lowest priority and 100 the highest.

categoryName
This parameter identifies the object Category Name. DFM provides this value to DIVA Core.

objectName
This parameter identifies the Object Name. DFM provides this value to DIVA Core.

comments
This parameter identifies optional information describing the object. DFM provides this value to DIVA Core.

sourceDestinationDIVAName
This parameter identifies the Server Name used in the archived request.

sourceDestinationDIVAPath
This parameter identifies the Server Path used in the archived request.

originalServer
This parameter identifies the Server to store in the object catalog, if different from the actual Server. This is useful when you use an intermediate storage area for archiving, but you want to store the original source for the data in the object catalog information.

originalPath
This parameter identifies the File Root Path to store in the object catalog, if different from the actual File Root Path. This is useful when you use an intermediate storage area for archiving, but you want to store the original source for the data in the object catalog information.

fileList
This parameter identifies the list of File Path Names that should be archived relative to the folder specified in sourceDestinationDIVAPath parameter.

Metadata File

Caution: The last file to be created in the File Set Drop Folder must be the metadata file (*.mdf*), which then triggers the archive operation. *If the metadata file is transferred before the actual data files, content loss can occur*!

You can create the MDF (*Metadata File*) in the original MDF format, or in XML format. Initially, DFM will attempt to parse the MDF as an XML file. If it fails, then it will parse it in the original MDF format.

Sample of the sample.mdf File in XML Format

The following is an example of the MDF file in the XML format:

```
#
#Object configuration.
#

<DIVAObjectDefinition>
 <priority>50</priority>
 <categoryName>category1</categoryName>
 <objectName>object name</objectName>
 <comments>here are our comments</comments>
 <sourceDestinationDIVAName>SD1</sourceDestinationDIVAName>
 <sourceDestinationDIVAPath>/video1/data/</sourceDestinationDIVAPath>
 <originalServer>original_server</originalServer>
 <originalPath>original_path</originalPath>
 <divaMediaName>array1</divaMediaName>
  <fileList>
  <file>object1/1.mid</file>
  <file>object1/2.mid</file>
  <file>object1/3.mid</file>
  </fileList>
</DIVAObjectDefinition>
```

Sample of the sample.mdf File in Original MDF Format

The following is an example of the MDF file in the original format:

```
#
#Object configuration.
#

priority=50
categoryName=category1
objectName=object name
<comments>
```

```
here
are
comments
</comments>

sourceDestinationDIVAName=SD1
sourceDestinationDIVAPath=/video1/data/

originalServer=original server
originalPath=original path

<fileList>
object1/1.mid
object1/2.mid
object1/3.mid
</fileList>
```

Sample of the Full Trace Log File

The following is an example of the full Trace log file:

```
08/02 15:53:12.263 INFO [main,] TraceManager: Logging (re)configured with:
file:/E:/app/DIVA/7.7.22.0/Program/InterLink/dfm/bin/../../../conf/dfm/dfm.trace, found as: file specified in
setConfigurationPath()
======================================================================
08/02 15:53:12.283 INFO [main,] TraceManager: Logging (re)configured with:
file:/E:/app/DIVA/7.7.22.0/Program/InterLink/dfm/bin/../../../conf/dfm/dfm.trace, found as: file specified in
setConfigurationPath()
======================================================================
08/02 15:53:12.465 INFO [Wrapper-Connection,] TraceManager: Logging (re)configured with:
file:/E:/app/DIVA/7.7.22.0/Program/InterLink/dfm/bin/../../../conf/dfm/dfm.trace, found as: file specified in
setConfigurationPath()
======================================================================
08/02 15:53:12.482 INFO [Wrapper-Connection,] TraceManager: Logging (re)configured with:
file:/E:/app/DIVA/7.7.22.0/Program/InterLink/dfm/bin/../../../conf/dfm/dfm.trace, found as: file specified in
setConfigurationPath()
======================================================================
08/02 15:53:12.483 INFO [Thread-1,] Start: Starting...
08/02 15:53:12.521 INFO [Thread-1,] TraceManager: Logging (re)configured with:
file:/E:/app/DIVA/7.7.22.0/Program/InterLink/dfm/bin/../../../conf/dfm/dfm.trace, found as: file specified in
setConfigurationPath()
======================================================================
08/02 15:53:12.869 WARN [Thread-1,] FOInformationBase: The dump file is absent.
08/02 15:53:12.870 WARN [Thread-1,] FOInformationBase: The dump file is absent.
08/02 15:53:12.974 INFO [Thread-1,] Main: Drop Folder Monitor started.
08/02 15:53:12.974 INFO [Thread-1,] Start: Started
08/02 15:54:12.970 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 15:55:12.924 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 15:55:12.967 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 15:56:12.872 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 15:56:12.964 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 15:57:12.869 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 15:57:12.964 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
```

```
archive operations is 5; full object list for archive size is 0
08/02 15:58:12.875 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 15:58:12.967 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 15:59:12.879 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 15:59:12.970 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 16:00:12.881 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 16:00:12.973 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 16:01:12.883 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 16:01:12.976 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 16:02:12.886 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 16:02:12.979 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 16:03:12.889 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 16:03:12.982 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 16:04:12.891 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 16:04:12.985 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 16:05:12.897 DEBUG [Thread-2,] FileMonitorThread: File prépa meeting.docx from folder
ftp://anonymous:anonymous@127.0.0.1/ is accpted!
08/02 16:05:12.988 DEBUG [Thread-4,] FOInformationBase: getObjectsForArchiving: archiving objects count is 0; max
archive operations is 5; full object list for archive size is 0
08/02 16:05:34.369 INFO [Wrapper-Connection,] Launcher: Stopping DFM...
08/02 16:05:34.408 INFO [Wrapper-Connection,] TraceManager: Logging (re)configured with:
file:/E:/app/DIVA/7.7.22.0/Program/InterLink/dfm/bin/../../../conf/dfm/dfm.trace, found as: file specified in
setConfigurationPath()
========================================================================
08/02 16:05:34.410 INFO [Wrapper-Connection,] Stop: Stopping Drop Folder Monitor...
08/02 16:05:34.410 DEBUG [Wrapper-Connection,] DFMModuleThread: Stopping module class
com.storagetek.diva.proxy.dfm.FileMonitorThread...
08/02 16:05:34.410 DEBUG [Wrapper-Connection,] DFMModuleThread: Module class
com.storagetek.diva.proxy.dfm.FileMonitorThread stopped.
08/02 16:05:34.410 DEBUG [Wrapper-Connection,] DFMModuleThread: Stopping module class
com.storagetek.diva.proxy.dfm.DMFileMonitorThread...
08/02 16:05:34.410 DEBUG [Wrapper-Connection,] DFMModuleThread: Module class
com.storagetek.diva.proxy.dfm.DMFileMonitorThread stopped.
08/02 16:05:34.410 DEBUG [Wrapper-Connection,] DFMModuleThread: Stopping module class
com.storagetek.diva.proxy.dfm.DIVAArchiveThread...
08/02 16:05:34.410 DEBUG [Wrapper-Connection,] DFMModuleThread: Module class
com.storagetek.diva.proxy.dfm.DIVAArchiveThread stopped.
08/02 16:05:34.410 DEBUG [Wrapper-Connection,] DFMModuleThread: Stopping module class
com.storagetek.diva.proxy.dfm.DIVAStatusThread...
08/02 16:05:34.411 DEBUG [Wrapper-Connection,] DFMModuleThread: Module class
com.storagetek.diva.proxy.dfm.DIVAStatusThread stopped.
08/02 16:05:34.411 DEBUG [Wrapper-Connection,] DFMModuleThread: Stopping module class
com.storagetek.diva.proxy.dfm.IncompleteLogThread...
08/02 16:05:34.411 DEBUG [Wrapper-Connection,] DFMModuleThread: Module class
com.storagetek.diva.proxy.dfm.IncompleteLogThread stopped.
08/02 16:05:34.411 DEBUG [Wrapper-Connection,] DFMModuleThread: Stopping module class
```

```
com.storagetek.diva.proxy.dfm.common.config.DFMConfigUpdater...
08/02 16:05:34.411 DEBUG [Wrapper-Connection,] DFMModuleThread: Module class
com.storagetek.diva.proxy.dfm.common.config.DFMConfigUpdater stopped.
08/02 16:05:34.411 DEBUG [Wrapper-Connection,] DFMModuleThread: Stopping module class
com.storagetek.diva.proxy.dfm.DFMStateSaverThread...
08/02 16:05:34.411 DEBUG [Wrapper-Connection,] DFMModuleThread: Module class
com.storagetek.diva.proxy.dfm.DFMStateSaverThread stopped.
08/02 16:05:34.411 INFO [Wrapper-Connection,] Stop: All Drop Folder Monitor modules stopped. DFM state will be saved.
08/02 16:05:34.414 INFO [Wrapper-Connection,] Stop: Drop Folder Monitor stopped. DFM state was saved.
08/02 16:05:34.414 INFO [Wrapper-Connection,] Launcher: DFM stopped.
08/02 16:05:34.415 INFO [Thread-9,] Stop: Stopping Drop Folder Monitor...
08/02 16:05:34.415 DEBUG [Thread-9,] DFMModuleThread: Stopping module class
com.storagetek.diva.proxy.dfm.FileMonitorThread...
```

Full DFM Configuration File (dfm.conf.ini)

The following is an example of the full DFM configuration file called dfm.conf.ini. You must configure and rename this file to dfm.conf to use DFM.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<dfmConfiguration>

 <!-- DIVA Manager address -->
 <managerConnetion>
  <address host="host" port="9000" userName="user" password="password" applicationName="DFM"/>
 </managerConnetion>

 <!--
WARNING: This parameter should be written in one line and there should not be any symbols or empty spaces at the
beginning of the line.
Otherwise it will not be correctly parsed by starting script.
-->
 <serviceName>DIVA CoreDFM</serviceName>

 <!-- File name of the File/Object Information Base dump -->
 <foibStateFile>../../program/interlink/dfm/bin/foibState.dmp</foibStateFile>

 <!-- File name of the Delete File/Object Information Base dump -->
 <doibStateFile>../../program/interlink/dfm/bin/doibState.dmp</doibStateFile>

 <!-- File reloading threshold in seconds. This interval will specify how
   often File Monitor Module will reload state of DFM folders.
   It should be positive integer value.
 -->
 <fileReloadingThreshold>60</fileReloadingThreshold>

 <!-- Time interval, in seconds. If file last modification time is not changed after this interval it will be marked as "copied"
and archived. It should be positive integer value.
 -->
 <fileReadyThreshold>10</fileReadyThreshold>

 <!-- Directory delete threshold in seconds(for folders in DELETE mode only). This interval will specify how often DFM File
Monitor Module will reconnect to storage areas to reload state of DFM folders. If the directory is empty, it will be deleted
upon the second connection. It should be positive integer value.
 -->
 <dirDeleteThreshold>600</dirDeleteThreshold>

 <!-- File delete threshold in seconds (for folders in DELETE mode only). This interval will specify when file will be deleted
from DFM folder after file copying to folder is completed. The state of completion will be identified by the file size. It
```

```
should be positive integer value.
 -->
<fileDeleteThreshold>86400</fileDeleteThreshold>

<!-- Maximum number of concurrent archive operations that can be performed by DFM simultaneously. The rest objects will wait until some archive operations will be completed.
 -->
<maxArchiveOperations>5</maxArchiveOperations>

<!-- Maximum number of re-archive operations that can be performed by DFM if DIVA rejected at request initializing. If this limit will be reached then the object will be marked as "incimplete".
 -->
<maxRejectCountInitializing>100</maxRejectCountInitializing>

<!-- Maximum number of re-archive operations that can be performed by DFM if DIVA rejected on the request processing. If this limit will be reached then the object will be marked as "incimplete".
 -->
<maxRejectCountProcessing>100</maxRejectCountProcessing>

<!-- Threshold before DIVA Archive Module re-request DIVA after rejection. This parameter should be provided in seconds.
 -->
<repeatRequestDIVAThreshold>180</repeatRequestDIVAThreshold>

<!-- This parameter specify polling interval of request state for DIVA Status Module. This value should be specified in seconds.
 -->
<pollingIntervalOfRequestState>30</pollingIntervalOfRequestState>

<!-- ========================================================================
   Default folder configurations.
   They may be overridden in folder-specific section
  ========================================================================
 -->
<defaultFolderConfig>
 <!-- There will be two types of directories: single and file set. If folder should be single - "single" value should be specified, if folder should be file set - "set" value should be specified.

   WARNING. Usage of the "Single" mode in combination with the FTP server source is not recommended. If the network connection is lost partial file may be stored in the archive system.
  -->
 <type>single</type>
 <!-- DFM folder work mode. Can be Archive or Delete. -->
 <mode>Archive</mode>

 <!-- Default priority of the requests for the folder. Folder priority can be in
 the range [0..100]. The value 0 is the lowest priority and 100 the highest.
  -->
 <priority>30</priority>
 <!-- Default value for category name. This value will be provided to DIVA if it will not be overridden in folder-specific configuration.
  -->
 <categoryName>dfmCategory</categoryName>

 <!-- If file is copied in "file set" folder and during incompleteThreshould seconds it was not archived it will be marked as "incomplete".
  -->
 <incompleteThreshold>86400</incompleteThreshold>

 <!-- Object Name Convert Rule. Only "simple" algorithm of conversion is implemented. -->
 <objectNameConvertRule method="simple">Name.Ext</objectNameConvertRule>
```

```
 <!-- Option to configure extension for the meta data file for each folder. -->
 <mdfExtension>mdf</mdfExtension>

 <!-- DIVA Original Server used in the archived request. -->
 <originalServer>Original Server</originalServer>

 <!-- DIVA Original Path used in the archived request. -->
 <originalPath>Original Path</originalPath>

 <!-- DIVA Source Destination name used in the archived request. -->
<sourceDestinationDIVAName>SourceDestinationServer</sourceDestinationDIVAName>

 <!-- This field defines location of the Drop Folder root on the DIVA Source Destination. In case of dedicated source
destination (so root of S/D will point to the root of DFM), FPR of S/D may contain a path and this field may be left empty.
  -->
 <sourceDestinationDIVAPath>SourceDestinationPath</sourceDestinationDIVAPath>

 <!-- Incomplete folder/file processing strategy. The value could be None, Delete, Rename
  -->
 <incompleteProcessingStrategy>Rename</incompleteProcessingStrategy>

 <!-- Defines threshold on number of files in the same directory. If the number of files in the directory will be large then a
numberFilesArchiveWildcard, DFM will send the archive request to manager using the wildcard "*" for the whole
directory. The value is integer.
 -->
 <numberFilesToArchiveWildcard>10</numberFilesToArchiveWildcard>

 <!-- This parameter is applicable only to "single" drop folders. Using this parameter one can specify the rule for which the
Root Path of the Arhcive Request will be generated.
    - platform is "WIN" | "SOL" | "CIFS" | "DETECT"

   The format of the value will be combination of directory names, separators, keywords PARENTn, URL_TO_FILE, URL_TO_
PARENTn. Keywords that are supported:
   1) PARENTn - the directory name, where the definition is: DFM_FILE_FULL_PATH = DFM_URL \ PARENTn ... \ PARENT 3 \
PARENT 2 \ PARENT 1 \ filename.ext
   2) URL_TO_PARENTm - the path from DFM_URL to PARENTm.
   3) URL_TO_FILE - the path from DFM_URL to file.
 -->
 <archiveFilePathTemplate platform="DETECT" options="">URL_TO_FILE</archiveFilePathTemplate>

 <!-- This parameter is applicable only to "single" drop folders. Using this parameter one can specify the rule for which the
file names of the Arhcive Request will be generated.

   - platform is "WIN" | "SOL" | "CIFS" | "DETECT"

   The format of the value will be combination of directory names, separators, keywords PARENTn, URL_TO_FILE,
filename.ext.
   Keywords that are supported:
    1) PARENTn - the directory name, where the definition is: DFM_FILE_FULL_PATH = DFM_URL \ PARENTn ... \ PARENT 3 \
PARENT 2 \ PARENT 1 \ filename.ext
    2) URL_TO_FILE - the path from DFM_URL to file.
    3) filename.ext - original file name.
    4) filename - file name without extension.
 -->
 <archiveFileNameTemplate platform="DETECT"  options="">filename.ext</archiveFileNameTemplate>

 <!-- This parameter specify if DFM should Delete Object before Archive Request or not for specific folder.
    This parameter can have value TRUE or FALSE.
    This is optional parameter. Default vale is FALSE.
```

```
 -->
 <deleteBeforeArchive>FALSE</deleteBeforeArchive>

 <!-- This parameter specify if DFM should Delete child and parent folders of MDF file.
    This parameter can have value TRUE or FALSE.
    This is optional parameter. Default vale is TRUE.
 -->
 <deleteParentDirectoryAndContentDirectories>TRUE</deleteParentDirectoryAndContentDirectories>

</defaultFolderConfig>

<!-- ========================================================================
   Folders configurations.
=========================================================================
 -->
 <folders>

  <folderConfig>
   <!-- Folder URL. -->
   <url>ftp://diva:diva@localhost:21/dropfolder1</url>

   <!-- There will be two types of directories: single and file set. If folder should be single - "single" value should be
specified, if folder should be file set - "set" value should be specified.
    -->
   <type>single</type>

   <!-- Folder priority can be in the range [0..100]. The value 0 is the lowest priority and 100 the highest.
    -->
   <priority>30</priority>

   <!-- This parameter can have Primary or Secondary values. Primary - configuration parameters specified in mdf are used
in archive request. Secondary - configuration parameters specified in folder specific configuration are used in archive
request.
Default valued, used when this tag is empty or missing - Primary.
 --->
   <mdfConfigPriority>Primary</mdfConfigPriority>

   <!-- Folder category name. This value will be provided to DIVA. -->
   <categoryName>Category1</categoryName>

   <!-- If file is copied in "file set" folder and during incompleteThreshould seconds it was not archived it will be marked as
"incomplete".
    -->
   <incompleteThreshold>86400</incompleteThreshold>
   <!-- DIVA Source Destination name used in the archived request. -->
   <sourceDestinationDIVAName>FTP</sourceDestinationDIVAName>

   <!-- This parameter is applicable only to "single" drop folders. Using this parameter one can specify the rule for which the
Root Path of the Arhcive Request will be generated.

     - platform is "WIN" | "SOL" | "CIFS" | "DETECT"

    The format of the value will be combination of directory names, separators, keywords PARENTn, URL_TO_FILE, URL_TO_
PARENTn. Keywords that are supported:
       1) PARENTn - the directory name, where the definition is: DFM_FILE_FULL_PATH = DFM_URL \ PARENTn ... \ PARENT 3
\ PARENT 2 \ PARENT 1 \ filename.ext
       2) URL_TO_PARENTm - the path from DFM_URL to PARENTm.
       3) URL_TO_FILE - the path from DFM_URL to file.
    -->
   <archiveFilePathTemplate platform="DETECT" options="">URL_TO_FILE</archiveFilePathTemplate>
```

```
  <!-- This parameter is applicable only to "single" drop folders. Using this parameter one can specify the rule for which the file names of the Arhcive Request will be generated.

     - platform is "WIN" | "SOL" | "CIFS" | "DETECT"

     The format of the value will be combination of directory names, separators, keywords PARENTn, URL_TO_FILE, filename.ext. Keywords that are supported:
       1) PARENTn - the directory name, where the definition is: DFM_FILE_FULL_PATH = DFM_URL \ PARENTn ... \ PARENT 3 \ PARENT 2 \ PARENT 1 \ filename.ext
       2) URL_TO_FILE - the path from DFM_URL to file.
       3) filename.ext - original file name.
       4) filename - file name without extension.
    -->
  <archiveFileNameTemplate platform="DETECT" options="">filename.ext</archiveFileNameTemplate>

  <!-- DIVA Media Name used in the archived request. -->
  <divaMediaName>Array1</divaMediaName>

  <!-- To specify filters <fileFilter> tag is used in folder configuration. "type" attribute can be "exclude" or "include". In inclusion mode, no files will be archived if no filters are defined. Each filter will allow archiving files by the specific rule. File will be archived only if file will correspond to one or more masks. If file does not correspond to any mask, file will be skipped by DFM (nor deleted neither archived).

     In exclusion mode, all files will be archived if no filters are defined (the same way is working now). Each filter will restrict archiving files by the specific rule. File will NOT be archived if file will correspond to one or more masks. If file corresponds to any mask, file will be skipped by DFM (nor deleted neither archived).

     Filter can be defined as a pattern (mask which can have ? which means any symbol or * which means any string) or as a Regex expression (please see details http://en.wikipedia.org/wiki/Regular_expression, http://www.regular-expressions.info).

     <mask> specifiyng pattern for filter.
     <regexp> specifiyng regular expression for filter.

     PLEASE NOTE: <fileFilter> parameter is optional for drop folder configuration. If it is not specified DFM will archive any file without any restriction except standard DFM logic.

     PLEASE NOTE: Only one <fileFilter> can be specified per one folder.
    -->
  <fileFilter type="include">
   <mask>*.*</mask>
  </fileFilter>

  <!-- This parameter specify if DFM should Delete Object before Archive Request or not for specific folder.
 This parameter can have value TRUE or FALSE.
      This is optional parameter. Default vale is FALSE.
    -->
  <deleteBeforeArchive>TRUE</deleteBeforeArchive>

  <!-- This parameter specify if DFM should Delete child and parent folders of MDF file.
      This parameter can have value TRUE or FALSE.
      This is optional parameter. Default vale is TRUE.
    -->   <deleteParentDirectoryAndContentDirectories>FALSE</deleteParentDirectoryAndContentDirectories>

 </folderConfig>

 <folderConfig>
  <!-- Folder URL. -->
  <url>ftp://diva:diva@localhost:21/dropfolder2</url>
```

```
  <!-- There will be two types of directories: single and file set. If folder should be single - "single" value should be
specified, if folder should be file set - "set" value should be specified.
    -->
  <type>set</type>

  <!-- Folder priority can be in the range [0..100]. The value 0 is the lowest priority and 100 the highest.
    -->
  <priority>30</priority>

  <!-- Folder category name. This value will be provided to DIVA. -->
  <categoryName>Category2</categoryName>

  <!-- If file is copied in "file set" folder and during incompleteThreshould seconds it was not archived it will be marked as
"incomplete".
    -->
  <incompleteThreshold>86400</incompleteThreshold>

  <!-- DIVA Source Destination name used in the archived request. -->
  <sourceDestinationDIVAName>FTP</sourceDestinationDIVAName>

  <!-- This field defines location of the Drop Folder root on the DIVA Source Destination. In case of dedicated source
destination (so root of S/D will point to the root of DFM), FPR of S/D may contain a path and this field may be left empty.
    -->
  <sourceDestinationDIVAPath>DROPFOLDER</sourceDestinationDIVAPath>

  <!-- DIVA Media Name used in the archived request. -->
  <divaMediaName>Array1</divaMediaName>

  <!-- DIVA Media Name Pattern used in the archived request. Using this pattern GROUP will be obtained -->
  <divaMediaNamePattern>def/$GROUP/data</divaMediaNamePattern>

  <!-- This parameter should be used to enable recursive archive mode of DFM. In this mode DFM will check MDF file and
if it contains asterisks as part of data files path recursive archive will be used.
     This parameter can have value TRUE or FALSE.
     This is optional parameter. Default vale is FALSE.
    -->
  <recursiveArchive>TRUE</recursiveArchive>

 </folderConfig>

 </folders>

 <!--WARNING: This is not a comment. Do not modify the text below. It is used by Tanuki Wrapper.
#********************************************************************
#Wrapper Properties
#********************************************************************
# Java Application
wrapper.java.command=%JAVA_HOME%\bin\javaw.exe
# Java Main class. This class must implement the WrapperListener interface
# or guarantee that the WrapperManager class is initialized. Helper
# classes are provided to do this for you. See the Integration section
# of the documentation for details.
wrapper.java.mainclass=com.storagetek.diva.proxy.dfm.control.TanukiLauncher

# Java Classpath (include wrapper.jar) Add class path elements as
# needed starting from 1
wrapper.java.classpath.1=../../Program/InterLink/lib/Interlink.jar
wrapper.java.classpath.2=./wrapper.jar
wrapper.java.classpath.3=../../Program/Common/lib/Common.jar
```

```
wrapper.java.classpath.4=../log4j/log4j.jar
wrapper.java.classpath.5=../JavaLib/xerces.jar
wrapper.java.classpath.6=../JavaLib/dtdparser115.jar
wrapper.java.classpath.7=../Apache/commons-io-1.3.2.jar

#wrapper.java.classpath.7=../libs/jhall.jar
#wrapper.java.classpath.8=../libs/junit.jar
#wrapper.java.classpath.9=../libs/ojdbc14.jar

# Java Library Path (location of Wrapper.DLL or libwrapper.so)
wrapper.java.library.path.1=.

# Java Additional Parameters
#wrapper.java.additional.1=
# Initial Java Heap Size (in MB)
wrapper.java.initmemory=64

# Maximum Java Heap Size (in MB)
wrapper.java.maxmemory=256

# Application parameters. Add parameters as needed starting from 1


#*********************************************************************
# Wrapper Logging Properties
#*********************************************************************
# Format of output for the console. (See docs for formats)
wrapper.console.format=PM

# Log Level for console output. (See docs for log levels)
wrapper.console.loglevel=INFO

# Format of output for the log file. (See docs for formats)
wrapper.logfile.format=LPTM

# Log Level for log file output. (See docs for log levels)
wrapper.logfile.loglevel=INFO

# Maximum size that the log file will be allowed to grow to before
# the log is rolled. Size is specified in bytes. The default value
# of 0, disables log rolling. May abbreviate with the 'k' (kb) or
# 'm' (mb) suffix. For example: 10m = 10 megabytes.
wrapper.logfile.maxsize=10m

# Maximum number of rolled log files which will be allowed before old
# files are deleted. The default value of 0 implies no limit.
wrapper.logfile.maxfiles=4

# Log Level for sys/event log output. (See docs for log levels)
wrapper.syslog.loglevel=NONE

#*********************************************************************
# Wrapper Windows Properties
#*********************************************************************
# Title to use when running as a console
wrapper.console.title=DIVA Core DFM

#*********************************************************************
# Wrapper Windows NT/2000/XP Service Properties
#*********************************************************************
```

```
# WARNING - Do not modify any of these properties when an application
# using this configuration file has been installed as a service.
# Please uninstall the service before modifying this section. The
# service can then be reinstalled.

# Time without CPU before JVM will issue warning and extend timeout (in sec).
# Timeout will be extended by a few seconds at least once before Wrapper shuts down.
#wrapper.cpu.timeout=30

# Number of seconds to allow between the time that the Wrapper launches the JVM process and the time that the JVM
side of the Wrapper responds that the application has started.
wrapper.startup.timeout=60

# Number of seconds to allow between the wrapper pinging the JVM and the response
wrapper.ping.timeout=60

# Number of seconds to allow between the time that the Wrapper asks the JVM to shutdown and the time that the JVM
side of the Wrapper responds that it is stopping.
wrapper.shutdown.timeout=60

# Name of the service
wrapper.ntservice.name=dfm

# Display name of the service
wrapper.ntservice.displayname=DIVA Core DFM
# Description of the service
wrapper.ntservice.description=DIVA Core DFM allows to monitor FTP and local directories and to archive incoming file to
the DIVA system

# Service dependencies. Add dependencies as needed starting from 1
wrapper.ntservice.dependency.1=

# Mode in which the service is installed. AUTO_START or DEMAND_START
wrapper.ntservice.starttype=AUTO_START

# Allow the service to interact with the desktop.
wrapper.ntservice.interactive=false

-->
</dfmConfiguration>
```

Glossary

CIFS

The CIFS (*Common Internet File System*) is a Microsoft sharing protocol. *Linux based Datahubs do not support UNC paths for CIFS Source and Destination Servers*.

Complex Object

An object is defined as a *Complex Object* when it contains 1,000 (*configurable*) or more components. Complex object handling may differ from non-complex objects as identified throughout this document.

Component

A file that is part of a DIVA Core object.

DFM (*Drop Folder Monitor*)

The term DFM is an acronym for the Drop Folder Monitor.

Drop Folder

A Drop Folder is a folder on a local disk, FTP server, or a CIFS shared folder designated for Single File mode, File Set mode, or both, storage that will be monitored by DFM and from which files will be archived.

FOIB (*File Object Information Base*)

The term FOIB is an acronym for *File Object Information Base*. FOIB is persistent storage used by DFM to track its processing state.

MDF (*Metadata File and .mdf*)

The term MDF is an acronym for the DFM Metadata File. The MDF is a file containing information about the files included in a File Set, and has the file extension .mdf.

DFM Incomplete File

DFM Incomplete files are files that were put into the Drop Folder without the metadata file (*for File Set folders*), and files that cannot be archived after a specific number of attempts.