

# XML API Specification

XML API Specification .....	1
2 Foreword .....	4
3 Introduction .....	4
4 Changes since last document .....	7
5 Version Numbering .....	8
6 FlashNet XML Header .....	8
7 API Operations .....	12
7.1 OPERATION "ARCHIVE" .....	12
7.1.1 Example of an archive operation request .....	15
7.1.2 Example of a reply to an archive operation request .....	15
7.2 OPERATION "RESTORE" .....	16
7.2.1 Partial Restoration .....	17
7.2.2 Example of a restore operation request .....	19
7.2.3 Example of a reply to a restore operation request .....	19
7.3 OPERATION "STATUS" .....	20
7.3.1 Example of a status operation request of an entire job .....	23
7.3.2 Example of a reply to a status request, this job is running but has had no data transferred at this moment .....	24
7.4 OPERATION "DELETE" .....	25
7.4.1 Example of a delete operation request .....	26
7.4.2 Example of a reply to a delete request .....	27
7.5 OPERATION "LISTGROUP" .....	28
7.5.1 Example of a ListGroup operation request .....	29
7.5.2 Example of a reply to a ListGroup request .....	30
7.6 OPERATION "READLOG" .....	30
7.6.1 Example of a ReadLog operation request .....	31
7.6.2 Example of a reply to a ReadLog request .....	32
7.7 OPERATION "LISTSERVER" .....	33
7.7.1 Example of a ListServer operation request .....	35
7.7.2 Example of a reply to a ListServer request .....	36
7.8 OPERATION "STOPJOB" .....	37
7.8.1 Example of a StopJob operation request .....	38
7.8.2 Example of a reply to a StopJob request .....	38
7.9 OPERATION "PAUSEJOB" .....	39
7.9.1 Example of a PauseJob operation request .....	40
7.9.2 Example of a reply to a PauseJob request .....	40
7.10 OPERATION "RESUMEJOB" .....	41
7.10.1 Example of a ResumeJob operation request .....	42
7.10.2 Example of a reply to a ResumeJob request .....	42
7.11 OPERATION "MIGRATEASSETS" .....	43
7.11.1 Example of a MigrateAssets request .....	44
7.11.2 Example of a reply to a MigrateAssets request .....	45
7.12 OPERATION "MIGRATIONSTATUS" .....	46
7.12.1 Example of a MigrationStatus request .....	47
7.12.2 Example of a reply to a MigrationStatus request .....	47
7.13 OPERATION "SEARCHARCHIVE" .....	48
7.13.1 Example of a SearchArchive operation request .....	54

7.13.2	<i>Example of a reply to SearchArchive request</i> .....	54
7.14	OPERATION "LISTVIDEOSEVER" .....	55
7.14.1	<i>Example of a ListVideoServer operation request</i> .....	56
7.14.2	<i>Example of a reply to a ListVideoServer request</i> .....	56
8	Specialised API calls .....	57
8.1	OPERATION "GETUIDBYPATH" .....	57
8.1.1	<i>Example of a GetUidByPath operation request</i> .....	58
8.1.2	<i>Example of a reply to a GetUidByPath request</i> .....	58
8.2	OPERATION "FULLASSETRESTORE" .....	59
8.2.1	<i>Example of a restore operation request</i> .....	60
8.2.2	<i>Example of a reply to a FullAssetRestore operation request</i> .....	60
8.3	OPERATION "LISTWATCHFOLDER" .....	61
8.3.1	<i>Example of a ListWatchFolder operation request</i> .....	62
8.3.2	<i>Example of a reply to a ListWatchFolder request</i> .....	63
8.4	OPERATION "LISTRESTOREFOLDER" .....	64
8.4.1	<i>Example of a ListRestoreFolder operation request</i> .....	65
8.4.2	<i>Example of a reply to a ListRestoreFolder request</i> .....	65
9	Legacy Functions .....	66
9.1	OPERATION "LISTGUID" .....	66
9.1.1	<i>Example of a ListGuid operation request</i> .....	70
9.1.2	<i>Example of a reply to a ListGuid request</i> .....	71
9.1.3	<i>Example of a paged ListGuid operation request</i> .....	72
9.1.4	<i>Example of a reply to a paged ListGuid request</i> .....	72
10	Appendices.....	73
10.1	FLASHNET CONFIGURATION OPTIONS AFFECTING THE OPERATION OF THE XML API .....	73
10.1.1	<i>Best Practices</i> .....	75
10.1.2	<i>Function Overview</i> .....	76
10.1.3	<i>Implementing Best Practices</i> .....	78
10.1.4	<i>Archiving to Tape Groups from Automation</i> .....	79
10.1.5	<i>Archiving to Disk Volumes from Automation</i> .....	79
10.1.6	<i>Paths to media files</i> .....	80
11	Document Type Definitions (DTD) .....	81
11.1	ARCHIVE.DTD .....	81
11.2	RESTORE.DTD .....	82
11.3	STATUS.DTD .....	82
11.4	DELETE.DTD .....	82
11.5	LISTGROUP.DTD .....	83
11.6	READLOG.DTD .....	83
11.7	LISTSERVER.DTD.....	83
11.8	STOPJOB.DTD .....	84
11.9	PAUSEJOB.DTD .....	84
11.10	RESUMEJOB.DTD.....	84
11.11	MIGRATEASSETS.DTD.....	85
11.12	MIGRATIONSTATUS.DTD.....	85
11.13	SEARCHARCHIVE.DTD .....	86
11.14	LISTVIDEOSEVER.DTD .....	86
11.15	GETUIDBYPATH.DTD .....	86
11.16	FULLASSETRESTORE.DTD .....	87
11.17	LISTWATCHFOLDER.DTD.....	87
11.18	LISTRESTOREFOLDER.DTD .....	87
12	Glossary.....	88
12.1	DTOOL .....	88

12.2	THE QUEUE.....	88
12.3	ROUND ROBIN.....	88
12.4	TAPE GROUP.....	88
12.5	VOLUME.....	88
12.5.1	Current volume.....	88
12.6	NON-SPANNING GROUPS.....	89
12.7	TAPE EXCHANGE DELAYS IN WORKFLOW.....	89
13	Frequently asked Questions.....	90
13.1	WHAT ARE THE DIFFERENCES BETWEEN A DISK ARCHIVE AND TAPE ARCHIVE WITH RESPECT TO THE API?.....	90
13.2	CAN I ARCHIVE COPIES OF THE SAME MATERIAL TO DIFFERENT GROUPS?.....	91
13.3	WHAT SETTINGS SHOULD I USE FOR PRIORITY.....	91
13.4	WHAT IS VERIFICATION & WHEN IS IT USED?.....	91
13.5	GROUP PRIORITY ADDITIONAL INFORMATION.....	91
13.6	WHAT IS THE PRIORITY ALGORITHM OF ASSETS FOR RESTORE?.....	93
13.7	HOW ARE MULTIPLE TAPE LIBRARIES HANDLED BY FLASHNET?.....	93
13.8	HOW SHOULD MATERIAL BE IDENTIFIED IN THE ARCHIVE TO USERS?.....	93
13.9	WHAT VALUE SHOULD I PASS FOR APIVERSION IN THE FLASHNETXML HEADER?.....	94
13.10	IS IT POSSIBLE TO LIMIT THE USE OF DELETE (PURGE) OPERATIONS BY USER?.....	94
13.11	WHAT IS THE DIFFERENCE BETWEEN "LISTGUID" AND "SEARCHARCHIVE"?.....	94
13.12	WHEN IS SEARCHARCHIVE OPERATION NORMALLY NEEDED/USED?.....	94
13.13	WHEN IS READLOG NORMALLY NEEDED/USED?.....	94
13.14	WHAT ARE THE DIFFERENCES BETWEEN BYTE OFFSET AND FRAME BASED PARTIAL FILE RESTORE?.....	95
13.15	FOR FRAME BASED PARTIAL FILE RESTORE HOW DO I DETERMINE START AND END FRAMES?.....	95
13.16	DELETING ASSETS: DO I NEED TO SPECIFY THE VOLUME AND ARCHIVE?.....	95

## 2 Foreword

The information in this document is confidential and proprietary to Software Generation Limited (SGL) and may be used only under the terms of the product license or nondisclosure agreement (NDA). The information in this document, including any associated software program, may not be disclosed, disseminated, or distributed in any manner without the written consent of Software Generation Limited (SGL) or an approved representative thereof.

### Limitation on Warranties and Liability

This document neither extends nor creates warranties of any nature, expressed or implied. Software Generation Limited (SGL) cannot accept any responsibility for your use of the information in this document or for your use of any associated software program. Software Generation Limited (SGL) assumes no responsibility for any data corruption or erasure as a result of the use of the information in this document, or the use of software programs. You should be careful to ensure that your use of the information complies with all applicable laws, rules, and regulations of the jurisdictions with respect to which the information is used. Information in this document is subject to change without notice or obligation.

### Warning

No part or portion of this document may be reproduced in any manner or in any form without the written permission of Software Generation Limited (SGL).

## 3 Introduction

This document describes the Application programming interface (API) to the FlashNet suite of archive software products. Connection to the FlashNet cluster is achieved by opening a tcp socket connection to the FlashNet socket listener service on one of the cluster nodes. The required port number is defined in the services file on each cluster node and can therefore be set to any value. It is expected that the calling application will have a pre-defined knowledge of the FlashNet cluster node names so that if connection to one of the cluster nodes fails it can attempt to connect to an alternative node. Consequently the automatic failover of any cluster node will not affect any application attempting to communicate with the cluster via this API. The socket listener service will launch an API handler process on-demand to process your request. Once a socket connection is opened communication is achieved by writing XML formatted text to the socket then reading XML formatted text as a reply. The reply will contain an integer handle to the requested operation which can be used in subsequent calls which require reference to the previous operation. The on-demand process will then exit and the socket closed.

Prior to version 6.4, all XML requests were required to follow the example format:

```
FlashNet XML 489 <?xml version="1.0" encoding="UTF-8"?>

<FlashNetXML APIVersion="2013.001" SourceServer="cluster-svr1" UserName="Paul"
CallingApplication="test_xml_api" Operation="Restore">

  <Restore Priority.DWD="8">
    <FileDetails FileCount.DWD="1">
      <File EntryNumber="0" FullFileName="paul-sgl1:/c/Demo - MV - Ernie1.mxf" Guid="XML-
ERNIE-4" PartialStart.QWD="128000" PartialEnd.QWD="560000"
PartialType="BYTE_OFFSET" />
    </FileDetails>
  </Restore>
</FlashNetXML>
```

Where 489 specified the exact number of bytes (following and including the first <) contained in the text to be sent. The calling application was required to calculate this number.

From version 2013.001 of the XML API, the service will accept pure XML requests, with no need for a byte count. This is the preferred format for requests and the relevant DTDs are included in this document. Passing of the byte count is still supported for legacy clients. If a client passes the byte count in the request, the response will also contain the byte count. If the client passes pure XML in the request, the response will be pure XML.

Examples of each supported command and the expected replies are contained within the rest of this document.

### **3.1 *Socket Layer Communications***

Communications to and from the XML API server is handled as a set of fixed-size 'pages' of data. The page size and the maximum timeout between page reads can be controlled at the server via registry settings. The server's root registry key is HKLM\SOFTWARE\Wow6432Node\Software Generation\FlashNet6\XMLD. Under this key, the page size, in bytes, is determined by the server using the registry 32-bit DWORD value ReadBufferSize. If the value is absent, the server defaults to 8KB pages. The server waits for each page for a maximum time determined by the registry 32-bit DWORD value RequestTimeout. The value is in seconds. If a request cookie runs into multiple pages, rather than wait for another page, if at any point the server detects a well-formed XML document in the data received so far, it accepts the request and starts processing it immediately.

A successful response from the server may also run into multiple pages. If this happens, the response page size is determined by the registry 32-bit DWORD value MaxSendBufferSize. The value is in bytes. If the value is absent, the server defaults to 1 MB pages. The last page of any response will include a stream null character, which may need to be removed before the response can be sent to an XML parser. After sending the successful response, the server re-enters a wait loop until the caller closes the socket. An error response from the server is unlikely to run into multiple pages, but if it does, it should be handled in the same way as a successful response with the caveat that the server does not re-enter a wait loop, but instead closes the socket immediately.

## Changes since last document

This document has been re-organised so that the guidelines for integration are part of the API document. Some internal functions have been removed as they are not for use by external integrators.

6/8/2012      Some spelling corrections and clarification of setting for APIVersion in the XML header.

11/4/2013    Revised request format requirements section. Updated version numbering scheme. Added DTDs. Removed deprecated internal APIs.

## 4 Version Numbering

This document introduces the new version numbering strategy for all FlashNet APIs. All versions of the API and documentation prior to this one are superseded by this one.

All FlashNet APIs have a version number in the format yyyy.bbb where yyyy is the four-digit year of release and bbb is the unique build number for that year. The first API to follow this convention is therefore 2013.001. The build number increases sequentially with each released build of the API throughout the year and is reset to 001 with the first build of each year.

Although we strive to ensure backwards compatibility between versions of the API and the server, we recommend that all the FlashNet API binaries in your integration have the same version number to avoid any issues.

## 5 FlashNet XML Header

### Description

All XML requests must start with the FlashNetXML element.

### Attributes of <FlashNetXML>

Attribute	Requirement	Data Type	Description
APIVersion	Mandatory	String	Identifies the revision of the FlashNet XML API which the calling application has been written to. This will mainly be used for support purposes. <b>This should be set to the version number of your client package unless otherwise stated.</b>
SourceServer	Mandatory	String	The server network name of the calling application.
UserName	(see description)	String	The name of the operator making the call if multiple users are communicating with the archive. <b>This is mandatory if the APIVersion is greater than 1.0.</b>
CallingApplication	Mandatory	String[50]	The name of the product making the call to the archive.



Operation	Mandatory	String	The type of request being made to the archive
-----------	-----------	--------	---

### Sub-Elements of <FlashNetXML>

Sub-elements depend on the value of the Operation attribute.


### Values of the “Operation” attribute

Archive	Specifies a transfer of data into the archive.
Restore	Specifies a transfer of data out of the archive.
Delete	Specifies a particular asset within the archive which is to be removed from the archive.
ListGuid	Retrieve full storage details for a list of assets contained within the archive by passing a search string.
ListGroup	Retrieve the list of available media storage pools and related details.
Status	Retrieve current details pertaining to a specific operation, either Archive or Restore.
ReadLog	A request to retrieve the full log transcript for either an archive or restoration request.
ListServer	A search operation which will return the list of server names in the FlashNet cluster and the overall status of each server including the status of each of the installed services.
ListWatchFolder	Retrieve the list of available FlashNet watch folders and related details.
ListRestoreFolder	Retrieve the list of available FlashNet restore locations.
SimpleRestoreSearch	Retrieve basic storage details for a list of assets contained within the archive by passing a search string.
SimpleArchiveSearch	Retrieve the name and size of all files on the specified video server.
SimpleRestoreRestore	Instigates a restore of all files within a specified archive on a specified volume.
SimpleRestoreDelete	Instigates a deletion of a specific Guid from FlashNet
ListVideoServer	Retrieves a list of available video servers connected to FlashNet.
GetUidByPath	Returns the FlashNet unique identifier of a file by its original archive path.
ListVideoServer	A search operation which will return the list of video servers known to FlashNet.
ListWatchFolder	A search operation which will return the list of watch folders in the FlashNet cluster.
ListRestoreFolder	A search operation which will return the list of restore folders in the FlashNet cluster.
SimpleRestoreSearch	Lists information about assets stored in the archive matching

	the supplied wildcard pattern.
SimpleRestoreRestore	Command used to instigate a restore process on a FlashNet cluster.
SimpleArchiveSearch	Lists files on all video servers matching a wildcard pattern.
StopJob	Command used to stop a job on the FlashNet server.
MigrateAssets	Command used to move or copy assets from one volume to another.
MigrationStatus	Command used to obtain progress status on a migration job.
FullAssetRestore	Command used to instigate a restore process of an entire asset on a FlashNet cluster.
SearchArchive	A search operation which will return the storage details for each matching Guid.

### Return values

Returned is an XML string containing a <Reply> root element with indication of whether the requested operation was successfully accepted or not. Failures could be caused by incorrect formatting of the XML text.

-  A successful return status does not indicate the success of an archive or restore job, only that the job was successfully sent to FlashNet. To determine the success of the job itself you must take the returned RequestId and perform a Status operation on it.

### Parameters

FileDetails	A collection of details about files either contained within the archive or to be added to the archive.
GroupDetails	Details relating to the available media storage pools within the archive.

### Sub-Elements of <Reply>

The sub elements will be described in the sections to which they apply.

### Attributes of <Reply>

Attribute	Data Type	Description
Status	String	Either "Passed" or "Failed"
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.
RequestId	Int32	The numeric identification of this request which can be subsequently used in API calls enquiring about this request.



## 6 API Operations

### 6.1 Operation "Archive"

#### Description

Command used to instigate an archive process on a FlashNet cluster. The definition of an archive process is the transfer of data from a network attached source server into a storage medium within the control of the FlashNet cluster.

#### DTD

[Archive.dtd](#)

#### Sub-Elements of <FlashNetXML> for Operation="Archive"

<Archive>	Specifies parameters for the job
-----------	----------------------------------

#### Attributes of <Archive>

Attribute	Requirement	Data Type	Description
VolumeGroup	Mandatory	String	The set of tapes or disk to be used for the archive operation
VerifyFiles.DWD	Optional	Int32	Specifies if the archive should perform a bitwise verification of the archived data with the original data. This process is performed after the archive has finished writing.
DeleteFiles.DWD	Optional	Int32	Specifies if the files being archived should be deleted after the archive has finished. This is not supported by some video servers.
Priority.DWD	Optional	Int32	Prior to FlashNet Server version 6.4.01: a value from 1 to 10, 1 being highest. The default is 5. From FlashNet Server version 6.4.01 this range is extended from 0 (zero) to 100. A value of zero allows FlashNet to prioritise your job itself. The default is 5.
DisplayName	Mandatory	String	A text based description of the data being archived.

#### Sub-Elements of <Archive>

<FileDetails>	Specifies the number of files and all required information about each file in order to successfully perform the archive operation.
---------------	--


### Attributes of <FileDetails>


Attribute	Requirement	Data Type	Description
FileCount.DWD	Mandatory	Int32	The number of file to be specified in this archive operation

### Sub-Elements of <FileDetails>

<File>	All the required information about a file in order to successfully perform the archive of that file.
--------	--

### Attributes of <File>

Attribute	Requirement	Data Type	Description
FullFileName	Mandatory	String	<p>The full path of the file including the server it resides on and the file name.</p> <p>e.g. for a UNC source:  <a href="#">\\host\share\optional_path\filename.ext</a></p> <p>for a video server or ftp source:            host:/optional_path/filename.ext</p> <p> <b>In the case of video server or ftp, the host must be registered as a comms method on your FlashNet server.</b></p>
Guid	Mandatory	String	A unique identifier which must be unique throughout the entire archive.
MetaData	Optional	String	Additional, caller-defined data which the caller wishes to associate with this file while it is within the archive. This data can be retrieved in searches for this file. The data is not returned during a restore operation on the file. This attribute may not be used where MetaDataFile is specified.
MetaDataFile	Optional	String	The fully-qualified path to a file containing additional, caller-defined data which the caller wishes to associate with this file while it is within

			the archive. This data can be retrieved in searches for this file. The data is not returned during a restore operation on the file. This attribute may not be used where MetaData is specified.
Size.QWD	Optional	Int64	<p>The size of the file, in bytes. Although it is optional, we strongly recommend that you specify this parameter to help FlashNet optimise the transfer and storage of your data.</p> <p> <b>Omitting this value will have a measurable performance impact on the throughput of the archive.</b></p>

### Sub-Elements of <File>

None

### Return Values

Returned as XMLString with a <Reply> root element.

### Attributes of reply to an archive operation

Attribute	Data Type	Description
Status	String	Either "Passed" or "Failed"
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.
RequestId	Int32	The numeric identification of this request which can be subsequently used in API calls enquiring about this request.

### Sub-Elements of <Reply>

None

### 6.1.1 Example of an archive operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "archive.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="Archive">
  <Archive VolumeGroup="raid%" VerifyFiles.DWD="1" DeleteFiles.DWD="0" Priority.DWD="4"
DisplayName="Archive of two clips">
    <FileDetails FileCount.DWD="2">
      <File FullFileName="//host/share/clip1.mxf" Size.QWD="3783591163" Guid="clip1" />
      <File FullFileName="//host/share/clip2.mxf" Size.QWD="734550016" Guid="clip2" />
    </FileDetails>
  </Archive>
</FlashNetXML>
```

### 6.1.2 Example of a reply to an archive operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" RequestId="1067" />
```

## 6.2 Operation "Restore"

### Description

Command used to instigate a restore process on a FlashNet cluster. The definition of a restore process is the transfer of data from a storage medium within the control of the FlashNet cluster to a destination network attached server.

### DTD

[Restore.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="Restore"

<Restore>	Specifies parameters for the job
-----------	----------------------------------

### Attributes of <Restore>

Attribute	Requirement	Data Type	Description
Priority.DWD	Optional	Int32	Prior to FlashNet Server version 6.4.01: a value from 1 to 10, 1 being highest. The default is 5. From FlashNet Server version 6.4.01 this range is extended from 0 (zero) to 100. A value of zero allows FlashNet to prioritise your job itself. The default is 4.
DisplayName	Optional	String	A text based description of the data being archived. If this is omitted, the first Guid will be used as the display name.

### Sub-Elements of <Restore>

<FileDetails>	Specifies the number of files and all required information about each file in order to successfully perform the archive operation.
---------------	--

### Attributes of <FileDetails>

Attribute	Requirement	Data Type	Description
FileCount.DWD	Mandatory	Int32	The number of files to be specified in this restore operation



**Sub-Elements of <FileDetails>**

<File>	All the required information about a file in order to successfully perform the restoration of that file.
--------	--

**Sub-Elements of <File>**

None

**Attributes of <File>**

Attribute	Requirement	Data Type	Description
FullFileName	Mandatory	String	The fully-qualified path to which to restore the file.
Guid	Mandatory	String	The unique identifier of the file to be restored
PartialStart.QWD	Optional	Int64	If you require only part of the file, specify the start offset here.
PartialEnd.QWD	Optional	Int64	If you require only part of the file, specify the end offset here.
PartialType	Optional	String	If you require only part of the file, specify the offset type here as either "FRAME_OFFSET" or "BYTE_OFFSET". See "Partial Restoration", below.

**6.2.1 Partial Restoration**

If the file you wish to restore is very large, and you only want part of it, you should specify a Partial File Restore (PFR) operation.

To make the restore a PFR, fill in the Guid as normal, and the FullFileName field with the full path and file name of the section you want. The PartialType attribute specifies whether you require a specific range of bytes back or a specific number of frames. You may specify as many partial sections as you require for each Guid but they must not overlap.

If you specify multiple PFR's for the same GUID in a restore request job. This will create "sub-jobs" for each partial file restore, each sub-job will be a single partial file restore segment. The segments must not overlap in the same job.

Partial file restore is typically implemented as FRAME\_OFFSET where the start and end value are defined by the start and end frame required. For MXF in order

for this to function the content being archived must contain an index table segment and be compliant with SMPTE 378.

#### BYTE\_OFFSET

Specify BYTE\_OFFSET ONLY if you intend to re-wrap the media yourself. This option returns an exact range of bytes from the source clip from the PartialStart.QWD offset to the PartialEnd.QWD offset. Until the bytes are re-wrapped, the restored file will not be playable.

#### FRAME\_OFFSET

Specify FRAME\_OFFSET if you are able to calculate the frame count for the PartialStart.QWD and PartialEnd.QWD. This will result in a re-wrapped, playable clip of the specified frames.

E.g.

Supposing you wanted to restore a clip from the beginning to time code 00:34:23:22. Assuming a frame rate of 25 f/s, this would give you:

$$22 + (23 \times 25) + (34 \times 60 \times 25) = 51597 \text{ frames}$$

You would then specify PartialStart.QWD as zero and PartialEnd.QWD as 51596 and PartialType as "FRAME\_OFFSET".

### Return Values

Returned is as XMLString with a <Reply> root element.

#### Attributes of reply to a restore operation

Attribute	Data Type	Description
Status	String	Either "Passed" or "Failed"
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.
RequestId	Int32	The numeric identification of this request which can be subsequently used in API calls enquiring about this request.

#### Sub-Elements of <Reply>

None

### 6.2.2 Example of a restore operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "Restore.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="Restore">
  <Restore Priority.DWD="8">
    <FileDetails FileCount.DWD="1">
      <File FullFileName="host:/c/Demo Ernie1.mxf" Guid="IMX30-5MINS-1" />
    </FileDetails>
  </Restore>
</FlashNetXML>
```

### 6.2.3 Example of a reply to a restore operation request.

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" RequestId="1074" />
```

### 6.3 Operation “Status”

#### Description

Command used to investigate the progress of either an archive or restore operation. The RequestId supplied as an attribute of the <Reply> to the archive or restore operation is used as the only attribute.

#### DTD

[Status.dtd](#)

#### Sub-Elements of <FlashNetXML> for Operation="Status"

<Status>	Specifies the request ID for which to retrieve the status report.
----------	---

#### Attributes of <Status>

Attribute	Requirement	Data Type	Description
RequestId.DWD	Mandatory	Int32	The handle to a previous archive or restore operation.
Guid	Optional	String	If file information, such as volume and group location, is required in the Reply then the Guid of the file of interest must be supplied.

#### Return Values

Returned is as XMLString with a <Reply> root element.

#### Attributes of reply to a status operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.


- ⚠ A return status of “Passed” or “Failed” refers to the success of sending the status request, not the status of the FlashNet job itself. Only a “Passed” request with a JobStatus attribute of “Failed, “Stopped” or “Killed” indicates that the specified job has been unsuccessful. This list is not exhaustive and may change in future APIs.

**Sub-Elements of <Reply>**

<StatusInfo>	A collection of information describing the status of an archive or restore job. Some attributes may be absent, depending on the status of the job.
--------------	--

**Attributes of <StatusInfo>**

Attribute	Data Type	Description
Priority.DWD	Int32	The current priority of this RequestId
SourceServer	String	The network server name of the machine which executed the job resulting in this RequestId.
QueueKey.QWD	Int64	A unique identifier useful for tracing this request in the FlashNet Admin interface.
QueueEntryTime	String	The date and time the request was submitted to the FlashNet queue in the format "yyyy/MM/dd hh:mm:ss".
JobEndTime	String	The date and time the job finished in the format "yyyy/MM/dd hh:mm:ss".
ProcessType	String	Either "Backup" or "Restore"
ProcessId.DWD	Int32	Process identifier of the dtool process for this job,
ChangerName	String	If a tape library has been used to perform this request, the name shown will match one of the libraries defined in the FlashNet Admin changer interface.
JobStatus	String	Either "Passed", "Failed", "Stopped", "Running", "Warning", "Killed", "Media Busy", "Not processed" and possibly others in the future.
JobStatus.DWD	Int32	A numerical job status code that will be in the following ranges: <ul style="list-style-type: none"> <li>&lt; 0: unknown/error condition</li> <li>0: finished</li> <li>1: running</li> <li>&gt; 1 queued/pending</li> </ul>
JobQueuedCode.DWD	Int32	Where the job has not yet completed, this number indicates which stage FlashNet has reached in the processing of it. <ul style="list-style-type: none"> <li>-6 Cache Fill Failed</li> <li>-5 Candidate Only</li> <li>-4 Changer Error</li> <li>-3 Dtool Error</li> <li>-2 Queue Error</li> <li>-1 Stopped</li> </ul>

		<ul style="list-style-type: none"> <li>0 Finished</li> <li>1 Running</li> <li>2 Sent to Launcher</li> <li>3 Ready to send to Launcher</li> <li>4 Master Job</li> <li>5 Not Processed</li> <li>10 Not Checked</li> <li>11 Unknown Volume/Group</li> <li>12 Drives Busy</li> <li>13 Media Busy</li> <li>14 Client Busy</li> <li>15 Group Empty</li> <li>16 No Device Drivers</li> <li>17 Changer Offline</li> <li>18 Rule Blocked</li> <li>19 Overwrite Disallowed</li> <li>20 No I/O Slaves</li> <li>21 No Changer Drives Defined</li> <li>22 Changer Drives Offline</li> <li>23 Specified Drive Busy</li> <li>24 Out Of Changer</li> <li>25 No Available Cache</li> <li>26 Filling Cache</li> <li>27 Awaiting Cache Fill</li> </ul> <p>Where a restore job has been split by FlashNet into multiple, linked jobs, this code takes into consideration the state of each of the linked jobs. For example, if there are two linked jobs and one has finished, but the other is still processing, the return code will not show Finished until the second job completes.</p>
JobErrorCode.DWD	Int32	<p>A numerical code indicating the status of the job or linked jobs. Where restore jobs are linked together, this code reflects the status of the batch as a whole entity, e.g. if one of the jobs has failed, the batch has failed, and so on. The current list of codes is:</p> <ul style="list-style-type: none"> <li>1 Running</li> <li>11 Passed</li> <li>16 Stopped (usually by user intervention)</li> <li>17 Failed</li> <li>18 Passed with warning(s) (benign error)</li> <li>24 Killed (a serious error occurred on the server)</li> </ul> <p> <b>Only codes 11 and 18 may be</b></p>

		<b>considered 'success' codes for your job.</b>
DataTransferred.QWD	Int64	The amount of data currently transferred from the data source to the data destination.
DataVerified.QWD	Int64	Only relevant if the ProcessType is "Backup" and indicates the current amount of data which has been restored from the destination media and compared to the same data on the source server. Verification occurs once the entire source data has been written to the destination media.
FileSize.QWD	Int64	If known, the size of the file on the source server.
ClientName	String	The network name of the machine which represents either the source or destination for data i/o.
FileFullName	String	All path information required to open the file needed for data i/o.
VolumeGroup	String	The tape media pool to be used for this RequestId. Not filled in if the media is disk based.
ProductName	String	The name of the application which executed this RequestId.
VolumeName	String	The tape or disk media name to be used for this RequestId.
ArchiveNumber.DWD	Int32	Storage media have many archives stored on them and this number identifies the archive for this RequestId on the VolumeName in use.
LogFileKey.DWD	Int32	For future use to gain access to the full text log file of this RequestId.

### 6.3.1 Example of a status operation request of an entire job

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "Status.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="Status">
  <Status RequestId.DWD="8" />
</FlashNetXML>
```

### 6.3.2 Example of a reply to a status request, this job is running but has had no data transferred at this moment

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" >
  <StatusInfo Priority.DWD="4"
    SourceServer="paul-sgl1"
    QueueKey.QWD="3442"
    QueueEntryTime="2008/09/26 13:30:44"
    JobEndTime="1900/01/01 00:00:00"
    ProcessType="Backup"
    ProcessId.DWD="11234"
    ChangerName="HP LTO"
    JobStatus="Running"
    JobStatus.DWD="1"
    JobErrorCode.DWD="0"
    JobQueuedCode.DWD="1"
    DataTransferred.QWD="0"
    DataVerified.QWD="0"
    FileSize.QWD="0"
    ClientName="192.168.10.190"
    FileFullName="/IMX30-5MINS"
    VolumeGroup="CCLUB"
    ProductName="test_xml_api"
    VolumeName="000020L3"
    ArchiveNumber.DWD="0">
    LogFileKey.DWD="13546"
  </StatusInfo>
</Reply>
```



## 6.4 Operation “Delete”

### Description

Command used to instigate the deletion of files within the archive. Depending on the storage medium the physical removal of the files from the archive may happen immediately if the storage is disk based, or may have to wait for the defrag storage management process to be applied to tape storage; either way, the file is marked for deletion within the FlashNet database and restore of the file cannot be requested. Usually only the Guid will be specified which will result in all copies of the Guid being set for removal, which is why volume name and archive number are optional.

### DTD

[Delete.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="Delete"

<Delete>	Specifies parameters for the job
----------	----------------------------------

### Attributes of <Delete>

None

### Sub-Elements of <Delete>

<FileDetails>	Specifies the number of files and all required information about each file in order to successfully perform the delete operation.
---------------	---

### Attributes of <FileDetails>

Attribute	Requirement	Data Type	Description
FileCount.DWD	Mandatory	Int32	The number of files to be specified in this delete operation

### Sub-Elements of <FileDetails>

<File>	All the required information about a file in order to successfully perform the deletion of that file.
--------	---

### Attributes of <File>

Attribute	Requirement	Data Type	Description
VolumeName	Optional	String	The name of the media that this file resides on.
ArchiveNumber.DWD	Optional	Int32	The archive number on the media that this file resides on.
Guid	Mandatory	String	The unique identifier that specifies the file which must be removed from the archive.

### Sub-Elements of <File>

None

### Return Values

Returned is as XMLString with a <Reply> root element.

### Attributes of reply to a delete operation.

Attribute	Data Type	Description
Status	String	Either "Passed" or "Failed"
Error	String	If the XML text could not be correctly translated the error string will specify what was wrong.

### Sub-Elements of <Reply>

None

### 6.4.1 Example of a delete operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "delete.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="Delete">
  <Delete>
    <FileDetails FileCount.DWD="2">
      <File Guid="IMX30-5MINS-1" />
      <File Guid="IMX30-5MINS-2" />
    </FileDetails>
  </Delete>
</FlashNetXML>
```

### 6.4.2 Example of a reply to a delete request

```
<?xml version="1.0" encoding="UTF-8"?>  
<Reply Version="2013.001" Status="Passed" />
```

## 6.5 Operation “ListGroup”

### Description

A search operation which will return all the available storage media names and details.

### DTD

[ListGroup.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="ListGroup"

None

### Attributes of <ListGroup>

None

### Return Values

Returned is as XMLString with a <Reply> root element.

### Attributes of reply to a ListGroup operation.

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.

### Sub-Elements of <Reply>

<GroupDetails>	Contains all the available information about the defined media pools.
----------------	---

### Attributes of <GroupDetails>

Attribute	Data Type	Description
GroupCount.DWD	Int32	The number of defined media pools.

### Sub-Elements of <GroupDetails>

<Group>	All the required information about a media pool
---------	---

### Sub-Elements of <Group>

None

### Attributes of <Group>

Attribute	Data Type	Description
GroupName	String	The name of the media pool. This name is specified as the destination for data during an archive operation.
GroupAge	Int32	The period in days that the data contained in this media pool is protected for. Automatic re-cycling of media can be performed using this value.
VolumeCount	Int32	The number of tapes contained in the media pool.
ChangerName	String	The number and name of the tape library containing this media pool.
CurrentVolume	String	If the media pool supports spanning then the tape name which is going to be used for the next archive is shown here.
Enabled.DWD	Int32	Non-zero if the group is currently available on the FlashNet server.
ExcessGroup.DWD	Int32	Non-zero if this group is a tape library's excess group. Volumes in this group are part of a pool used by a real group and so must not be used as an archive target.
AllowSpanning.DWD	Int32	Non-zero if this group allows large archives to span across multiple tapes.

### 6.5.1 Example of a ListGroup operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "ListGroup.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="ListGroup" />
```

### 6.5.2 Example of a reply to a ListGroup request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" >
  <GroupDetails GroupCount.DWD="2">
    <Group      GroupName="2008/Sport"
                GroupAge.DWD="9999"
                VolumeCount.DWD="15"
                ChangerName="1 HP LTO"
                CurrentVolume="000005L3" />
    <Group      GroupName="2008/Films/Comedy"
                GroupAge.DWD="9999"
                VolumeCount.DWD="200"
                ChangerName="1 HP LTO"
                CurrentVolume="000020L3" />
  </GroupDetails>
</Reply>
```

## 6.6 Operation "ReadLog"

### Description

A request to retrieve the full log transcript for an archive or restoration request. This is typically used to give more low level detail of the archive operation, especially in the case of a failure.

### DTD

[ReadLog.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="ReadLog"

<ReadLog>	Specifies parameters for the job
-----------	----------------------------------

### Attributes of <ReadLog>

Attribute	Requirement	Data Type	Description
LogFileKey.DWD	Mandatory	Int32	The value returned in the LogFileKey attribute of the StatusInfo attribute of the reply to a status operation.
Encoding	Optional	String	The resulting log text may contain characters that are illegal in XML. By default, the text is Base64 encoded, if you require a different encoding, you may specify "URL" for URL encoding or "Plain" for un-encoded text. Be

			aware that requesting un-encoded text is prone to XML parsing errors and is therefore not recommended.
--	--	--	--

### Sub-Elements of <ReadLog>

None

### Return Values

Returned is as XMLString with a <Reply> root element.

- ⚠ From version 6.1.48 of the XML Service, the value of the LogText attribute is Base64 encoded by default. For legacy integrations that require a plain-text response, please add the attribute Encoding="Plain" to the element <ReadLog> on your request.

### Attributes of reply to a ReadLog operation.

Attribute	Data Type	Description
Status	String	Either "Passed" or "Failed"
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.
LogText	String	A base64-encoded text string of undefined length.

### 6.6.1 Example of a ReadLog operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "ReadLog.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="ReadLog">
  <ReadLog LogFileKey.DWD="2" Encoding="URL"/>
</FlashNetXML>
```

### 6.6.2 Example of a reply to a ReadLog request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed"
LogText="VGh1IERlYyAyMiAwMTowNzoxNCAYMDExICAgICAgRmxhc2h0ZXQgTG9nIEZpbGUgl
CAglFZlcnNpb24gNi40LjAxLjANCjAxOjA3OkkwNDQ6IFVzaW5nIGVycm9yIGxvZyBmaWxllGM6XG
ZsYXNobmV0Ni9lcnJvc9sb2dzL2Vycm9yMS5sb2cNDQowMTowNzpjMTIwOiBFeGVjdXRpbmcbg
24gc2VydmlvYlCdTR0w0MycNDQowMTowNzpjMDA2OiBvc2luZyB0aGUgZGF0YWJhc2UNDQow
MTowNzpjJMDIwOiBDcmVhdGluZyBpbmRleCAxIG9uIFZvbHVtZSAncmFpZDAnDQ0KMDE6MDc6
STExNjogUXVldWUgRGlzcGxheSBOYW1lIFxccc2dsNDNcZGF0YVxHRVLDn8OEw6TDnMO8w5bD
tI9hbGxjaGFyYWN0ZXJzLmpjZg0NCjAxOjA3OkY5OTkgTWWvc2FnZSBjMTI4OiBOb3QgRm91bm
QNCg0KW05vIENvbW1lbnQgU3VvcGxpZWRdDQpWb2x1bWUgJ3JhaWQwJw0KQXJjaGl2ZSAxI
GlzIDEuNjAwMDAwZSswMDUgYnI0ZXMNClIdyaXR0ZW4gYnkgYSBHRU5FUkIDIE5BUw0KTFWF4I
GJsb2NrlHNpemUgNjRldQpJbmRleCB0YXMGNCBlbnRyaWVzDQpmbGFncyBjdkENCmZpbGVzI
HdyaXR0ZW4gb24gVGh1IERlYyAyMiAwMTowNzoxNCAYMDExDQotLS0tLS0tLS0tLS0tLS0tLS
0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0t
LS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS
SBhcmNoaXZlIGZpbGUgZm9yIHdyaXRpbmcsIGVycm9vIDANDQowMTowNzpjMDA4OiBPcGVyY
XRpb24gY29tcGxldGUgYXQgVGh1IERlYyAyMiAwMTowNzoxNCAYMDExDQoNDQowMTowNzpj
MDYwOiBFeGl0aW5nIGR0b29sWzlwMDRdIEF0IFRodSBEZWMgMjlgMDE6MDc6MTQgMjAxMQ0
KDQ0K"/>
```



## 6.7 Operation “ListServer”

### Description

A search operation which will return the list of server names in the FlashNet cluster and the overall status of each server including the status of each of the installed services.

This can be used to determine the available servers to process XMLD requests in a FlashNet cluster and their status

### DTD

[ListServer.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="ListServer"

None

### Attributes of <ListServer>

None

### Return Values

Returned is as XMLString with a <Reply> root element.

### Attributes of reply to a ListServer operation.

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.

### Sub-Elements of <Reply>

<ServerDetails>	Contains all the available information about the defined cluster server nodes.
-----------------	--

### Attributes of <ServerDetails>

Attribute	Data Type	Description
Count.DWD	Int32	The total number of servers in the cluster.

**Sub-Elements of <ServerDetails>**

<Server>	All the required information about a FlashNet cluster node.
----------	---

**Sub-Elements of <Server>**

None

**Attributes of <Server>**

Attribute	Data Type	Description
Name	String	The network name of the FlashNet cluster node.
Status	String	Indicates that the node is currently executing as part of the cluster, Online, Offline or SCSI Update.
CurrentIO.DWD	Int32	The data I/O in b/s currently transferring through this node.
QueueService	String	Status of the queue service on this node. This can be one of the following values: Active, Running, Stopped or Unknown.
LaunchService	String	Status of the launch service on this node. This can be one of the following values: Active, Running, Stopped or Unknown.
StorageService	String	Status of the storage service on this node. This can be one of the following values: Active, Running, Stopped or Unknown.
ConfigService	String	Status of the configuration service on this node. This can be one of the following values: Active, Running, Stopped or Unknown.
ClientService	String	Status of the client service on this node. This can be one of the following values: Active, Running, Stopped or Unknown.
BrowseService	String	Status of the browse service on this node. This can be one of the following values: Active, Running, Stopped or Unknown.
WatchService	String	Status of the watch service on this node. This can be one of the following values: Active, Running, Stopped or Unknown.

### 6.7.1 Example of a ListServer operation request

```
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE FlashNetXML SYSTEM "ListServer.dtd">  
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"  
CallingApplication="test_xml_api" Operation="ListServer" />
```

### 6.7.2 Example of a reply to a ListServer request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" >
  <ServerDetails Count.DWD="3">
    <Server Name="Node-one" Status="Offline" CurrentIO.DWD="0" QueueService="stopped"
LaunchService="stopped" StorageService="stopped" ConfigService="stopped"
ClientService="stopped" BrowseService="stopped" WatchService="stopped" />
    <Server Name="Node-two" Status="Online" CurrentIO.DWD="39345678" QueueService="active"
LaunchService="active" StorageService="active" ConfigService="active" ClientService="active"
BrowseService="running" WatchService="running" />
  </ServerDetails>
</Reply>
```

## 6.8 Operation “StopJob”

This function requires XML Service version 6.3 or better

### Description

Command used to stop a job on the FlashNet server. Jobs can usually be stopped if they are in a queued state but cannot always be stopped if they are already running and an I/O operation is tying up the dtool process (rare case).

### DTD

[StopJob.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="StopJob"

<Stop>	Specifies parameters for the job
--------	----------------------------------

### Attributes of <Stop>

Attribute	Requirement	Data Type	Description
Job.DWD	Mandatory	Int32	The request id of the job to be stopped.

### Sub-Elements of <Stop>

None

### Return Values

Returned as XMLString with a <Reply> root element.

### Attributes of reply to a StopJob operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”

### Sub-Elements of <Reply>

None

### 6.8.1 Example of a StopJob operation request

```
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE FlashNetXML SYSTEM "StopJob.dtd">  
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"  
  CallingApplication="test_xml_api" Operation="StopJob">  
  <Stop Job.DWD="8" />  
</FlashNetXML>
```

### 6.8.2 Example of a reply to a StopJob request

```
<?xml version="1.0" encoding="UTF-8"?>  
<Reply Status="Passed" />
```

## Operation “PauseJob”

This function requires XML Service version 6.3 or better

### Description

Command used to pause a job on the FlashNet server. Jobs can usually be paused if they are in a queued state but cannot always be paused if they are already running and an I/O operation is tying up the dtool process. Use [ResumeJob](#) resume processing of the job.

### DTD

[PauseJob.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="PauseJob"

<Pause>	Specifies parameters for the job
---------	----------------------------------

### Attributes of <Pause>

Attribute	Requirement	Data Type	Description
Job.DWD	Mandatory	Int32	The request id of the job to be paused.

### Sub-Elements of <Pause>

None

### Return Values

Returned as XMLString with a <Reply> root element.

### Attributes of reply to a PauseJob operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”

### Sub-Elements of <Reply>

None

### 6.9.1 Example of a PauseJob operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "PauseJob.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="PauseJob">
  <Pause Job.DWD="8" />
</FlashNetXML>
```

### 6.9.2 Example of a reply to a PauseJob request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Status="Passed" />
```



## 6.10 Operation “ResumeJob”

This function requires XML Service version 6.3 or better

### Description

Command used to resume a paused a job on the FlashNet server.

### DTD

[ResumeJob.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="ResumeJob"

<Resume>	Specifies parameters for the job
----------	----------------------------------

### Attributes of <Resume>

Attribute	Requirement	Data Type	Description
Job.DWD	Mandatory	Int32	The request id of the job to be resumed.

### Sub-Elements of <Resume>

None

### Return Values

Returned as XMLString with a <Reply> root element.

### Attributes of reply to a ResumeJob operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”

### Sub-Elements of <Reply>

None

### 6.10.1 Example of a ResumeJob operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "ResumeJob.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="ResumeJob">
  <Pause Job.DWD="8" />
</FlashNetXML>
```

### 6.10.2 Example of a reply to a ResumeJob request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Status="Passed" />
```

## 6.11 Operation “MigrateAssets”

This function requires XML Service version 6.3 or better

### Description

Command used to move or copy assets from one volume to another. This is typically useful when content needs to be staged from deep archive to near-line archive in readiness for a restore operation.

### DTD

[MigrateAssets.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="MigrateAssets"

<Migrate>	Specifies parameters for the job
-----------	----------------------------------

### Attributes of <Migrate>

Attribute	Requirement	Data Type	Description
SourceVolume	Mandatory	String	The name of the medium with the archive containing the assets to migrate.
SourceArchive.DWD	Mandatory	Int32	The archive number containing the assets to migrate.
DestVolume	Mandatory	String	The name of the target medium.
MoveAssets.DWD	Optional	Int32	A flag to govern whether the assets are moved or copied to the target medium. 1 = Move 0 = Copy The default is to copy assets.

### Sub-Elements of <Migrate>

None

### Return Values

Returned as XMLString with a <Reply> root element.

### Attributes of reply to a MigrateAssets operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”

**Sub-Elements of <Reply>**

<JobDetail>	Specifies FlashNet jobs that have been launched to process the request.
-------------	---

**Attributes of <JobDetail>**

Attribute	Data Type	Description
JobCount.DWD	Int32	The total number of jobs launched for the migration.

**Sub-Elements of <JobDetail>**

<Job>	Information about a single migration job.
-------	---

**Sub-Elements of <Job>**

None

**Attributes of <Job>**

Attribute	Data Type	Description
Key.DWD	Int32	The unique job id of this migration job.
RequestId.DWD	Int32	The unique request id of the whole migration task.
Guid	String(132)	The unique identifier of the asset being migrated.

**6.11.1 Example of a MigrateAssets request**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "MigrateAssets.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="MigrateAssets">
  <Migrate SourceVolume="LTFS" SourceArchive.DWD="4" DestVolume="Disk2"
MoveAssets.DWD="1" Priority.DWD="6" />
</FlashNetXML>
```

### 6.11.2 Example of a reply to a MigrateAssets request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="6.3.00.0" Status="Passed" RequestId="83">
  <JobDetail JobCount.DWD="1">
    <Job Key.DWD="1003" RequestId.DWD="83" Guid="02.jpg"/>
  </JobDetail>
</Reply>
```

## 6.12 Operation “MigrationStatus”

This function requires XML Service version 6.3 or better

### Description

Command used to obtain progress status on a migration job or a specific migration Guid launched by a call to [MigrateAssets](#).

### DTD

[MigrationStatus.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="MigrationStatus"

<MigrateStatus>	Specifies parameters for the job
-----------------	----------------------------------

### Attributes of <MigrateStatus>

Attribute	Requirement	Data Type	Description
RequestId.DWD	Mandatory	Int32	The request id of the migration task.
Key.DWD	Mandatory	Int32	The job id of the job on which to query status.
Guid	Optional	String(132)	The unique identifier of the asset to query.

### Sub-Elements of <MigrateStatus>

None

### Return Values

Returned as XMLString with a <Reply> root element.

### Attributes of reply to a MigrationStatus operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”

### Sub-Elements of <Reply>

<StatusInfo>	Information about a single migration job or Guid.
--------------	---

### Sub-Elements of <StatusInfo>

None

**Attributes of <StatusInfo>**

Attribute	Data Type	Description
JobStatus.DWD	Int32	The current status of this job.
Percent	Float	If a Guid was passed in the request, this is the percentage transferred for that Guid. If no Guid was passed, this is the percentage complete of this migration job.
Percent.DWD	Int32	The integer part of the Percent attribute.

**6.12.1 Example of a MigrationStatus request**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "MigrationStatus.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="MigrationStatus">
  <MigrateStatus RequestId.DWD="6" Key.DWD="121" Guid="IMX30" />
</FlashNetXML>
```

**6.12.2 Example of a reply to a MigrationStatus request**

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="6.3.00.0" Status="Passed">
  <StatusInfo JobStatus.DWD="1" Percent="55.68" Percent.DWD="56"/>
</Reply>
```

### 6.13 Operation "SearchArchive"

#### Description

A search operation which will return the storage details for each matching Guid. This function is an extended version of [ListGuid](#) that allows greater control over the result set.

#### DTD

[SearchArchive.dtd](#)

#### Sub-Elements of <FlashNetXML> for Operation="SearchArchive"

<SearchArchive>	Specifies parameters for the job.
-----------------	-----------------------------------

#### Attributes of <SearchArchive>

Attribute	Requirement	Data Type	Description
Guid	Mandatory	String	A wildcard specification of the Guid string. The % character can be used to form a search string which will return a list of pattern matches containing the storage details of each Guid.
From.QWD	Mandatory	Int64	FlashNet keeps a record of every instance of every Guid successfully archived as well as copies created by its own Storage Manager. Over time, the archive database can grow to many millions of Guids. To ease the time and memory cost of returning so many Guids, the caller can opt to receive matches in a series of pages. To retrieve the first page, pass 0 (zero) as From.QWD. The server will return a page of results whose size is pre-determined by your system administrator. To retrieve subsequent pages, use the value returned as NextFrom.QWD as the new value for From.QWD <b>without</b>



			<b>changing any other parameter.</b> Keep doing this until ExitCode.DWD returns 4, defined below under 'Attributes of FileDetails'
Group	Optional	String	A wildcard specification of the media pool to which the Guids were originally archived. The % character can be used to specify more than one media pool e.g. 'Group%' will return all Guids archived (or subsequently copied) to volumes in Group1, Group2, Group3 etc.
Volume	Optional	String	A wildcard specification of the volume on which the Guids reside. The % character can be used to specify more than one volume e.g. 'Tape%' will return all Guids currently on Tape1, Tape2, Tape3 etc.
ArchivedFromDate	Optional	String	Specify a date and time to search only for Guids archived on or after this date and time. Acceptable formats are: "dd/MM/yyyy" or "dd/MM/yyyy hh:mm:ss". If you omit this attribute, Guids will be matched from the earliest instance. This tag may be combined with <ArchivedToDate> to specify a range of dates.
ArchivedToDate	Optional	String	Specify a date and time to search only for Guids archived on or before this date and time. Acceptable formats are: "dd/MM/yyyy" or "dd/MM/yyyy hh:mm:ss". If you omit this attribute, Guids will be matched up to the current server time. This tag may be combined with <ArchivedFromDate> to specify a range of dates.
DeletedFromDate	Optional	String	Specify a date and time to search only for Guids deleted on or after this date and time. Acceptable formats are: "dd/MM/yyyy" or "dd/MM/yyyy hh:mm:ss". If you omit this attribute, Guids will be matched

			from the earliest instance. This tag may be combined with <code>&lt;DeletedToDate&gt;</code> to specify a range of dates.
DeletedToDate	Optional	String	Specify a date and time to search only for Guids deleted on or before this date and time. Acceptable formats are: "dd/MM/yyyy" or "dd/MM/yyyy hh:mm:ss". If you omit this attribute, Guids will be matched up to the current server time. This tag may be combined with <code>&lt;DeletedFromDate&gt;</code> to specify a range of dates.
PageSize.DWD	Optional	Int32	The matching results from this function are returned in 'pages' of up to 100 results. You may specify your preferred page size by passing in a value of 1 to 100. If this parameter is omitted, or an illegal value is passed, FlashNet will return the results in pages of UIDLIST_PAGE_SIZE results. This value is defined in your GLOBALS table.
Flags.DWD	Optional	Int32	This tag is reserved for future use, please omit it or pass zero.
IncludeMetadata.DWD	Optional	Int32	A value of 1 will make the reply include any user-defined, binary metadata that was stored with the clip at the time of archive. This can slow down the search and put a heavy load on the FlashNet server. Only use this flag if you intend to use the user-defined metadata blocks. The metadata will be base64 encoded.

### Sub-Elements of <SearchArchive>

None

### Return Values

Returned is as XMLString with a <Reply> root element.

**Attributes of reply to a ListGuid operation.**

Attribute	Data Type	Description
Status	String	Either "Passed" or "Failed"
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.

**Sub-Elements of <Reply>**

<FileDetails>	Contains all the available information about the matching Guids.
---------------	--

**Attributes of <FileDetails>**

Attribute	Data Type	Description
FileCount.DWD	Int32	If a paged reply was requested, this is the number of matching Guids in this page; otherwise this is the total number of matching Guids.

The following attributes require XML Service version 6.3.01 or better

Attribute	Data Type	Description
NextFrom.QWD	Int32	If a valid From.QWD was specified in the request, to retrieve matching results in pages, this attribute contains the value that should be passed in as From.QWD in the call for the next page of results. This attribute is absent if paged results were not required. Do not attempt to alter or interpret the values in From.QWD or NextFrom.QWD as they are indexes internal to FlashNet. Any modification of these values will lead to unpredictable results.
ExitCode.DWD	Int32	If paged results were requested, the caller should continue requesting pages, updating From.QWD each call with the value from NextFrom.DWD until ExitCode.DWD returns a value of 4 or an error code (c.f. enumerated type __tagPagedListAssetError, below defined in fnapi.h of the FlashNet SDK)

**Possible values for ExitCode.DWD**

- 0 - The operation was successful
- 1 - The FlashNet server is not at the correct version
- 2 - The supplied parameters were invalid
- 3 - No matching UIDs were found
- 4 - All matches for the supplied pattern have been retrieved
- 5 - A problem occurred in the linkage between the client and the server, possibly due to a version mismatch
- 6 - Out of memory
- 7 - An internal error occurred
- 8 - Some other error occurred
- 9 - The native API was not initialised correctly (internal error)

### Sub-Elements of <FileDetails>

<File>	Contains all the available information about a single Guid.
--------	---

### Attributes of <File>

Attribute	Data Type	Description
Guid	String	The unique identifier of this file within FlashNet. Please note that there may be multiple instances of the same Guid within FlashNet.
VolumeName	String	The name of the media that this instance of the Guid resides on.
ArchiveNumber.DWD	Int32	Storage media may have many archives stored on them. This number identifies the archive which this instance resides on.
VolumeGroup	String	The tape media pool or disc volume name used to archive this instance of the Guid.
FileSize.QWD	Int64	The size of the Guid in bytes.
ArchiveDate	String	The date this Guid was added to the archive in the format "yyyy/MM/dd hh:mm:ss".
RestoreCount.DWD	Int32	The number of times this Guid has been restored.
LastRestoreDate	String	The date and time this Guid was last restored in the format "yyyy/MM/dd hh:mm:ss".
DeletedDate	String	If DeletedDate is filled in then this Guid is awaiting deletion and will eventually disappear from the list completely once the defrag operation has taken place. If this attribute is absent then the Guid has not been deleted. The format is "yyyy/MM/dd hh:mm:ss".
ChangerDetail	String	The number and name of the tape library with

		the tape containing this instance of the Guid. This attribute is absent for disc-based Guids.
ChangerPosition	String	The current position of this media in the named tape library. This references the drive or slot position of the media. This attribute is absent for disc-based Guids.
Metadata	String	Unlimited length of base64 encoded text. This attribute is absent if there is no user-defined metadata for the Guid.
Status	String	The current status of this instance of the Guid; one of: 'DELETED', 'ONLINE' or 'OFFLINE'

The following examples require XML Service version 6.3.01 or better

### 6.13.1 Example of a SearchArchive operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "SearchArchive.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="SearchArchive">
  <SearchArchive Guid="%" Group="News%" Volume="%" ArchivedFromDate="1/4/1980
6:30:00" ArchivedToDate="" DeletedFromDate="" DeletedToDate="" PageSize.DWD="50"
From.QWD="0" Flags.DWD="0" IncludeMetadata.DWD="1"/>
</FlashNetXML>
```

### 6.13.2 Example of a reply to SearchArchive request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" >
<FileDetails FileCount.DWD="100" NextFrom.QWD="101" ExitCode="4">

<File Guid="IMX30-5MINS-1" VolumeName="News1" ArchiveNumber.DWD="4"
VolumeGroup="News%" FileSize.QWD="1335000317" ArchiveDate="2011/05/12 16:15:30"/>
<File Guid="IMX30-5MINS-2" VolumeName="News1" ArchiveNumber.DWD="44"
VolumeGroup="News%" FileSize.QWD="1382000134" ArchiveDate="2011/05/12 18:00:00"/>
<File Guid="IMX30-10MINS-1" VolumeName="News2" ArchiveNumber.DWD="5"
VolumeGroup="News%" FileSize.QWD="2465001006" ArchiveDate="2011/05/13 09:30:12"/>
<File Guid="IMX30-15MINS-1" VolumeName="News2" ArchiveNumber.DWD="11"
VolumeGroup="News%" FileSize.QWD="3771010034" ArchiveDate="2011/05/13 10:22:04"/>
<File Guid="IMX30-50MINS-6" VolumeName="News4" ArchiveNumber.DWD="8"
VolumeGroup="News%" FileSize.QWD="12110000254" ArchiveDate="2011/05/13 12:03:19"/>
<File ...
...
</FileDetails>
```

## 6.14 Operation “ListVideoServer”

This function requires XML Service version 6.3 or better

### Description

A search operation which will return the list of video servers known to FlashNet.

### DTD

[ListVideoServer.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="ListVideoServer"

None

### Attributes of <FlashNetXML>

Attribute	Requirement	Data Type	Description
Pattern	Optional	String	Simple wildcard string for filtering the server name, e.g. MediaDeck*

### Return Values

Returned as XMLString with a <Reply> root element.

### Attributes of reply to a ListVideoServer operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.

### Sub-Elements of <Reply>

<ServerDetails>	Contains all the available information about the defined video servers.
-----------------	---

### Attributes of <ServerDetails>

Attribute	Data Type	Description
Count.DWD	Int32	The total number of matching video servers.

### Sub-Elements of <ServerDetails>

<Server>	All the required information about a video server.
----------	--

**Sub-Elements of <Server>**

None

**Attributes of <Server>**

Attribute	Data Type	Description
Name	String	The network name of the video server.
Type	String	The communications type of the server.
MaxConnect.DWD	Int32	The maximum number of supported concurrent connections.
CurrentConnect.DWD	Int32	The current number of concurrent connections.
TopLevelPath	String	The root of the file system on the server.

**6.14.1 Example of a ListVideoServer operation request**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "ListVideoServer.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="ListVideoServer" Pattern="*" />
```

**6.14.2 Example of a reply to a ListVideoServer request**

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed">
  <ServerDetails Count.DWD="1">
    <Server Name="SGL43" Type="SGL SDSS" MaxConnect.DWD="0"
CurrentConnect.DWD="0"/>
  </ServerDetails>
</Reply>
```



## 7 Specialised API calls

The API calls in this section were developed for specific functions and may not be required in a typical integration.

### 7.1 Operation “GetUidByPath”

This function requires XML Service version 6.3 or better

#### Description

A search operation which will return the FlashNet unique identifier of an asset, based on its original path at the time of archive.

#### DTD

[GetUidByPath.dtd](#)

#### Attributes of <FlashNetXML>

Attribute	Requirement	Data Type	Description
FullFileName	Mandatory	String	Fully-qualified path that was sent to FlashNet for archive.
Style	Optional	String	Either ‘LastUid’ or nothing. If this is blank, FlashNet will return the first matching UID for this path.

#### Sub-Elements of <FlashNetXML> for Operation="GetUidByPath"

None

#### Return Values

Returned as XMLString with a <Reply> root element.

#### Attributes of reply to a GetUidByPath operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.

#### Sub-Elements of <Reply>

None

### 7.1.1 Example of a GetUidByPath operation request

```
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE FlashNetXML SYSTEM "GetUidByPath.dtd">  
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"  
CallingApplication="test_xml_api" Operation="GetUidByPath" FullFileName="/c/flashnet6/licence.dat" />
```

### 7.1.2 Example of a reply to a GetUidByPath request

```
<?xml version="1.0" encoding="UTF-8"?>  
<Reply Version="2013.001" Status="Passed" Guid="TEST-1"/>
```

## 7.2 Operation "FullAssetRestore"

This function requires XML Service version 6.4 or better

### Description

Command used to instigate a restore process of an entire asset on a FlashNet cluster. This is typically used instead of a [Restore operation](#) *when you do not know the Guids of the individual files in the asset, but you do have the RequestId of the original archive request*. The files will be restored to their original location at the time of the backup, unless you specify a new directory. The original directory hierarchy at the time of archive will be re-created at the destination directory specified here. It is the caller's responsibility to ensure that the files do not already exist at this location at time of calling this API. If the files already exist at the target location, the restore operation will fail.

### DTD

[FullAssetRestore.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="FullAssetRestore"

<Restore>	Specifies parameters for the job
-----------	----------------------------------

### Attributes of <Restore>

Attribute	Requirement	Data Type	Description
Priority.DWD	Optional	Int32	Prior to FlashNet Server version 6.4.01: a value from 1 to 10, 1 being highest. The default is 5. From FlashNet Server version 6.4.01 this range is extended from 0 (zero) to 100. A value of zero allows FlashNet to prioritise your job itself. The default is 4.
AssetId.DWD	Mandatory	Int32	The RequestId as returned from the original call to Archive. The Guids specified in that job will be restored.
Destination	Optional	String	The desired destination directory for all the files in the asset. If this field is not supplied, each file will be restored to its original location at the time of the backup.
DisplayName	Optional	String	A text based description of the data

			being restored. If this is omitted, FlashNet will use the name of the original archive job.
--	--	--	---

### Sub-Elements of <Restore>

None

### Return Values

Returned is as XMLString with a <Reply> root element.

### Attributes of reply to a restore operation

Attribute	Data Type	Description
Status	String	Either "Passed" or "Failed"
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.
RequestId	Int32	The numeric identification of this request which can be subsequently used in API calls enquiring about this request.

### Sub-Elements of <Reply>

None

## 7.2.1 Example of a restore operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "FullAssetRestore.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="FullAssetRestore">
  <Restore Priority.DWD="8" AssetId.DWD="8" Destination="/c/temp/FAR/" />
</FlashNetXML>
```

## 7.2.2 Example of a reply to a FullAssetRestore operation request.

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" RequestId="1074" />
```

### 7.3 Operation “ListWatchFolder”

This function requires XML Service version 6.3 or better

#### Description

A search operation which will return the list of watch folders in the FlashNet cluster.

#### DTD

[ListWatchFolder.dtd](#)

#### Sub-Elements of <FlashNetXML> for Operation="ListWatchFolder"

None

#### Attributes of <FlashNetXML>

Attribute	Requirement	Data Type	Description
Pattern	Optional	String	Simple wildcard string for filtering the watch folders, e.g. NewsWatch*

#### Return Values

Returned as XMLString with a <Reply> root element.

#### Attributes of reply to a ListWatchFolder operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.

#### Sub-Elements of <Reply>

<WatchFolderDetails>	Contains all the available information about the defined watch folders.
----------------------	---

#### Attributes of <WatchFolderDetails>

Attribute	Data Type	Description
Count.DWD	Int32	The total number of matching watch folders.

#### Sub-Elements of <WatchFolderDetails>

<WatchFolder>	All the required information about a watch folder.
---------------	--

**Sub-Elements of <WatchFolder>**

None

**Attributes of <WatchFolder>**

Attribute	Data Type	Description
Nickname	String	The display name of the watch folder.
Folder	String	The physical path of the watch folder.
Destination	String	The target volume group for files archived from this watch folder.
Threshold.QWD	Int64	Where the watch folder wraps small files into larger archives this is the size (in bytes) of the archive.
CreateBrowse.DWD	Int32	Should a browse proxy be created for clips archived from this watch folder. 1 or zero.
DeleteOriginal.DWD	Int32	Should FlashNet delete files from the watch folder after successful archival. 1 or zero.
RecurseFolders.DWD	Int32	Should sub folders of the watch folder be scanned for files also. 1 or zero.
DhmArchive.DWD	Int32	Is this watch folder part of an Avid DHM workflow. 1 or zero.

**7.3.1 Example of a ListWatchFolder operation request**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "ListWatchFolder.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="ListWatchFolder" Pattern="*" />
```

### 7.3.2 Example of a reply to a ListWatchFolder request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed">
  <WatchFolderDetails Count.DWD="2">
    <WatchFolder Nickname="wf1" Folder="c:\temp" Destination="destination"
Threshold.QWD="90" CreateBrowse.DWD="1" DeleteOriginal.DWD="1" RecurseFolders.DWD="1"
DhmArchive.DWD="0"/>
    <WatchFolder Nickname="wf2" Folder="c:\flashnet6\cache" Destination="somewhere
else" Threshold.QWD="10" CreateBrowse.DWD="0" DeleteOriginal.DWD="0"
RecurseFolders.DWD="0" DhmArchive.DWD="1"/>
  </WatchFolderDetails>
</Reply>
```

## 7.4 Operation “ListRestoreFolder”

This function requires XML Service version 6.3 or better

### Description

A search operation which will return the list of restore folders in the FlashNet cluster.

### DTD

[ListRestoreFolder.dtd](#)

### Sub-Elements of <FlashNetXML> for Operation="ListRestoreFolder"

None

### Return Values

Returned as XMLString with a <Reply> root element.

### Attributes of reply to a ListRestoreFolder operation

Attribute	Data Type	Description
Status	String	Either “Passed” or “Failed”
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.

### Sub-Elements of <Reply>

<RestoreFolderDetails>	Contains all the available information about the defined restore folders.
------------------------	---

### Attributes of <RestoreFolderDetails>

Attribute	Data Type	Description
Count.DWD	Int32	The total number of restore folders.

### Sub-Elements of <RestoreFolderDetails>

<RestoreFolder>	All the required information about a restore folder.
-----------------	--

### Attributes of <RestoreFolder>

Attribute	Data Type	Description
Folder	String	The fully-qualified path of the folder.



### 7.4.1 Example of a ListRestoreFolder operation request

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FlashNetXML SYSTEM "ListRestoreFolder.dtd">
<FlashNetXML APIVersion="2013.001" SourceServer="SGL43" UserName="editorA"
CallingApplication="test_xml_api" Operation="ListRestoreFolder" />
```

### 7.4.2 Example of a reply to a ListRestoreFolder request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed">
  <RestoreFolderDetails Count.DWD="1">
    <RestoreFolder Folder="c:\temp"/>
  </RestoreFolderDetails>
</Reply>
```

## Legacy Functions

### 8.1 Operation "ListGuid"

This Operation is provided for backwards compatibility with legacy applications. Please use the more flexible Operation SearchArchive instead for new integration.

#### Description

A search operation which will return the storage details for each matching Guid.

#### Sub-Elements of <FlashNetXML> for Operation="ListGuid"

<ListGuid>	Specifies parameters for the job.
------------	-----------------------------------

#### Attributes of <ListGuid>

Attribute	Requirement	Data Type	Description
Guid	Mandatory	String	A wildcard specification of the Guid string. The % character can be used to form a search string which will return a list of pattern matches containing the storage details of each Guid.
IncludeMetadata.DWD	Optional	Int32	A value of 1 will make the reply include any user-defined, binary metadata that was stored with the clip at the time of archive. It will be base64 encoded.

The following attributes require XML Service version 6.3.01 or better

Attribute	Requirement	Data Type	Description
VolumeGroup	Optional	String	A wildcard specification of the media pool to which the Guids were originally archived. The % character can be used to specify more than one media pool e.g. 'Tape%' will return all Guids archived (or subsequently copied) to Tape1, Tape2, Tape3 etc.
FromDate	Optional	String	Specify a date to search only for Guids

			archived on or after this date. Acceptable formats are: "dd/MM/yyyy" or "dd/MM/yyyy hh:mm:ss". If you omit this attribute, Guids will be matched from the earliest instance. This tag is only used for paged requests.
ToDate	Optional	String	Specify a date to search only for Guids archived on or before this date. Acceptable formats are: "dd/MM/yyyy" or "dd/MM/yyyy hh:mm:ss". If you omit this attribute, Guids will be matched up to the current server time. This tag is only used for paged requests.
Flags.DWD	Optional	Int32	<p>This function is able to return a subset of matching Guids restricted to those marked for deletion or those not marked for deletion. To control this behaviour, you may pass a bitmask of the following flags (defined in fnapi.h of the FlashNet SDK):</p> <p>8 (LIST_DELETED_UIDS) – include only Guids marked for deletion  96 (LIST_NEW_UIDS) – include only Guids not marked for deletion</p> <p>If this attribute is omitted, or its value is 0 (zero) the default behaviour is to return both deleted and non-deleted Guids, (functionally equivalent to passing (LIST_DELETED_UIDS   LIST_NEW_UIDS))</p> <p>This tag is only used for paged requests.</p>
From.DWD	Optional	Int32	FlashNet keeps a record of every instance of every Guid successfully archived as well as copies created by its own Storage Manager. Over time, the archive database can grow to many millions of Guids. To ease the time and memory cost of returning so many Guids, the caller can opt to receive matches in a series of pages. To retrieve the first page, pass 0 (zero) as From.DWD. The server will return a page of results whose size is pre-determined by your system administrator. To retrieve subsequent pages, use the value returned as

			NextFrom.DWD as the new value for From.DWD <b>without changing any other parameter</b> . Keep doing this until ExitCode.DWD returns 4, defined below under 'Attributes of FileDetails'
--	--	--	--

### Sub-Elements of <ListGuid>

None

### Return Values

Returned is as XMLString with a <Reply> root element.

### Attributes of reply to a ListGuid operation.

Attribute	Data Type	Description
Status	String	Either "Passed" or "Failed"
Error	String	If the XML text could not be correctly translated the error sting will specify what was wrong.

### Sub-Elements of <Reply>

<FileDetails>	Contains all the available information about the matching Guides.
---------------	---

### Attributes of <FileDetails>

Attribute	Data Type	Description
FileCount.DWD	Int32	If a paged reply was requested, this is the number of matching Guides in this page; otherwise this is the total number of matching Guides.

The following attributes require XML Service version 6.3.01 or better

Attribute	Data Type	Description
NextFrom.DWD	Int32	If a valid From.DWD was specified in the request, to retrieve matching results in pages, this attribute contains the value that should be passed in as From.DWD in the call for the next page of results. This attribute is absent if paged results were not required. Do not

		attempt to alter or interpret the values in From.DWD or NextFrom.DWD as they are indexes internal to FlashNet. Any modification of these values will lead to unpredictable results.
ExitCode.DWD	Int32	If paged results were requested, the caller should continue requesting pages, updating From.DWD each call with the value from NextFrom.DWD until ExitCode.DWD returns a value of 4 or an error code.

### Possible values for ExitCode.DWD

- 0 - The operation was successful
- 1 - The FlashNet server is not at the correct version
- 2 - The supplied parameters were invalid
- 3 - No matching UIDs were found
- 4 - All matches for the supplied pattern have been retrieved
- 5 - A problem occurred in the linkage between the client and the server, possibly due to a version mismatch
- 6 - Out of memory
- 7 - An internal error occurred
- 8 - Some other error occurred
- 9 - The native API was not initialised correctly (internal error)

### Sub-Elements of <FileDetails>

<File>	Contains all the available information about a single Guid.
--------	---

### Attributes of <File>

Attribute	Data Type	Description
Guid	String	The unique identifier of this file within FlashNet. Please note that there may be multiple instances of the same Guid within FlashNet.
VolumeName	String	The name of the media that this instance of the Guid resides on.
ArchiveNumber.DWD	Int32	Storage media may have many archives stored on them. This number identifies the archive which this instance resides on.
VolumeGroup	String	The tape media pool or disc volume name used to archive this instance of the Guid.
FileSize.QWD	Int64	The size of the Guid in bytes.
ArchiveDate	String	The date this Guid was added to the archive in

		the format "yyyy/MM/dd hh:mm:ss".
RestoreCount.DWD	Int32	The number of times this Guid has been restored.
LastRestoreDate	String	The date and time this Guid was last restored in the format "yyyy/MM/dd hh:mm:ss"
DeletedDate	String	If DeletedDate is filled in then this Guid is awaiting deletion and will eventually disappear from the list completely once the defrag operation has taken place. If this attribute is absent then the Guid has not been deleted. The format is "yyyy/MM/dd hh:mm:ss"
ChangerDetail	String	The number and name of the tape library with the tape containing this instance of the Guid. This attribute is absent for disc-based Guids.
ChangerPosition	String	The current position of this media in the named tape library. This references the drive or slot position of the media. This attribute is absent for disc-based Guids.
Metadata	String	Unlimited length of base64 encoded text. This attribute is absent if there is no user-defined metadata for the Guid.
Status	String	The current status of this instance of the Guid; one of: 'DELETED', 'ONLINE' or 'OFFLINE'

### 8.1.1 Example of a ListGuid operation request

```
FlashNet XML 128 <?xml version="1.0" encoding="UTF-8"?>
<FlashNetXML
  APIVersion="1.0"
  SourceServer="cluster-svr1"
  UserName="Paul"
  CallingApplication="test_xml_api"
  Operation="ListGuid">

  <ListGuid
    Guid="IMX30-5MINS-1"/>
</FlashNetXML>
```

### 8.1.2 Example of a reply to a ListGuid request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" >
<FileDetails FileCount.DWD="1">

<File
  Guid="IMX30-5MINS-1  "
  VolumeName="storage1"
  ArchiveNumber.DWD="4"
  VolumeGroup=""
  FileSize.QWD="1335000317"
  ArchiveDate="2008/09/25 16:15:30"
  LastRestoreDate="2008/09/25 17:47:51"
  RestoreCount.DWD="1" />

</FileDetails>
</Reply>
```

The following examples require XML Service version 6.3.01 or better

### 8.1.3 Example of a paged ListGuid operation request

```
FlashNet XML 128 <?xml version="1.0" encoding="UTF-8"?>
<FlashNetXML
  APIVersion="1.0"
  SourceServer="cluster-svr1"
  UserName="Paul"
  CallingApplication="test_xml_api"
  Operation="ListGuid">

  <ListGuid VolumeGroup="News%" FromDate="12/05/2011" ToDate="13/05/2011"
  From.DWD="0"
    Guid="IMX30%"/>
</FlashNetXML>
```

### 8.1.4 Example of a reply to a paged ListGuid request

```
<?xml version="1.0" encoding="UTF-8"?>
<Reply Version="2013.001" Status="Passed" >
<FileDetails FileCount.DWD="100" NextFrom.DWD="101" ExitCode="4">

<File Guid="IMX30-5MINS-1" VolumeName="News1" ArchiveNumber.DWD="4"
VolumeGroup="News%" FileSize.QWD="1335000317" ArchiveDate="2011/05/12 16:15:30"/>
<File Guid="IMX30-5MINS-2" VolumeName="News1" ArchiveNumber.DWD="44"
VolumeGroup="News%" FileSize.QWD="1382000134" ArchiveDate="2011/05/12 18:00:00"/>
<File Guid="IMX30-10MINS-1" VolumeName="News2" ArchiveNumber.DWD="5"
VolumeGroup="News%" FileSize.QWD="2465001006" ArchiveDate="2011/05/13 09:30:12"/>
<File Guid="IMX30-15MINS-1" VolumeName="News2" ArchiveNumber.DWD="11"
VolumeGroup="News%" FileSize.QWD="3771010034" ArchiveDate="2011/05/13 10:22:04"/>
<File Guid="IMX30-50MINS-6" VolumeName="News4" ArchiveNumber.DWD="8"
VolumeGroup="News%" FileSize.QWD="12110000254" ArchiveDate="2011/05/13 12:03:19"/>
<File ...
...
</FileDetails>
```



## 9 Appendices

### 9.1 *FlashNet configuration options affecting the operation of the XML API*

These details apply to XML API version 6.3 or better

There are 2 FlashNet Configuration options that affect the behaviour of the XML API and how job submissions are processed. They are:

"ALLOW\_DUPLICATES"  
"DUPLICATE\_JOB\_DETECTION"

These settings alter the behaviour of job submission at a global level within a FlashNet archive installation in the following way:

DUPLICATE\_JOB\_DETECTION = false  
ALLOW\_DUPLICATES = false

Do not attempt to detect duplicate job submission that is a secondary submission of exactly the same job. Check for the existence of any individual Guids submitted in the request in the FlashNet archive database. If any duplication is found, the job submission will fail and an error would be returned. Otherwise accept the job and return a new RequestId for this job.

\* This is the default behaviour for a new FlashNet installation.

DUPLICATE\_JOB\_DETECTION = false  
ALLOW\_DUPLICATES = true

Do not attempt to detect duplicate job submission that is a secondary submission of exactly the same job. Do not attempt to detect duplication of Guids within the FlashNet database and accept the job. Return a new RequestId for this job.

DUPLICATE\_JOB\_DETECTION = true  
ALLOW\_DUPLICATES = false

Attempt to detect duplicate job submission that is a secondary submission of exactly the same job. If this is detected as submission of a duplicate job, pass the submission and return the RequestId of the first submission of this job, no new job will be created. If the job is not a duplicated submission, check for the existence of any individual Guids submitted in the request in the FlashNet archive database. If any duplication is found, the job submission will fail and an error would be returned. Otherwise accept the job and return a new RequestId for this job.

DUPLICATE\_JOB\_DETECTION = true  
ALLOW\_DUPLICATES = true

Attempt to detect duplicate job submission that is a secondary submission of exactly the same job. If this is detected as submission of a duplicate job, pass the submission and return the RequestId of the first submission of this job, no new job will be created. If the job is not a duplicated submission do not attempt to detect duplication of Guids within the FlashNet database and accept the job. Return a new RequestId for this job.

## Integration Best Practices

### 9.1.1 Best Practices

The purpose of this section is to describe the “Best Practices” in integrating any third party system with SGL’s FlashNet Media Archive Content Management solution. This will provide an overview of which core functions are available in the FlashNet API’s and how they may be used to extend the typical media workflow to include an Archive Content Management solution.

FlashNet has the ability to present many different forms of physical storage to outside systems via its XML or library based API’s. As well as acting as a buffer to changes in storage technology these API’s simplify the process of managing content through different ‘layers’ of storage. For example content can be moved or copied to the archive and held on disk storage for a certain period of time. This enables more valuable space on on-line storage to be used more effectively. After a certain period of time the archive disk storage may start to fill up and it may be desirable to migrate content into more efficient and longer term forms of storage like data tape. In this case, the FlashNet Storage Manager Service will copy and/or move content deeper into the archive. Even though the original content has moved it is still available to any third party system via the same means that it was placed in the archive in the first place.

As the usage needs of the archive increase it may be necessary to add more tape drives, disk storage or even additional libraries. This can all be done without affecting the API mechanism used to gain access to content. In parallel to this it is possible to add and change the controlling systems for the archive, production and play-out for example. In this case the FlashNet API provides the facility for the controlling system to identify itself via the use of a product ID.

There are many more features and functions available to developers via the FlashNet API’s not all are listed here (they are fully described in the FlashNet SDK).

## 9.1.2 Function Overview

[Connection](#) – The FlashNet XML API is by far the most popular and offers a light weight easy to deploy mechanism to gain access to the full range of API calls. Connection to the FlashNet cluster is achieved by opening a TCP socket connection to one of the cluster nodes. The required port number is defined in the services file on each cluster node and can therefore be set to any value (typically 8199).

[Archive](#) – The calling application MUST pass the UID, source path and group at a minimum in order to archive media. It is recommended to supply a user-friendly display name for the job. It is important to note that the group that the UID belongs to may change as a result of content management rules in the Archive system itself. This must be accounted for in the design of the “grouping” structure of the client facing tools. For example, the clips may be archived to a group called PROMO%. PROMO% is a disk volume and rules in the FlashNet system move content from PROMO disk to PROMO\_LTO5 after 30 days. The clips are then deleted from the group PROMO. In the context of archiving, ‘Group’ defines the entry point to the archive and not necessarily its final destination. It is also recommended that the archive include the size and priority of the request. Optionally, a single block of user-defined metadata may be passed to the archive system to be stored along with the clip. An archive request aka asset may contain multiple related or even unrelated files.

[Product ID](#) – In order for FlashNet to have an informed perspective of which external systems ‘own’ which content, this parameter should be set. Many archive users require the ability to filter content stored by which system put the content there. The typical example of this is the increasing use of a single archive shared between a production system and transmission play-out automation system. In this case the production system would have a different Product ID to that of the play-out system. These ID’s do not have to be globally unique, just unique between controlling systems per site.

[Restore](#) – The calling application is only required to pass the desired UID to the restore request. It is recommended to supply a user-friendly display name for the job. This will effect a restore back to the original path of the archived file. Typically, however, a new destination path is supplied. Optionally, a partial restore request may be made by specifying the start and end frame offsets.

[Status](#) – Each Archive or Restore operation gets a Request ID assigned to it when submitted via the API. The status of this request ID, i.e. the bytes transferred and the exit status may be obtained and presented to the user by the calling application. If the job creates multiple sub-jobs then these status code have to be queried and aggregated to get an overall value for bytes transferred etc.

[SearchArchive](#) – The calling application may issue this command to check that an asset exists in the archive. This is often used to check if a UID exists in the archive before allowing it to be re-archived. This may also be called as a maintenance task to update the client facing database. This list can grow to many millions of UIDs over time so it is not recommended that this method be used to build the list of available archives. This method may also be called before a restore request is issued to make sure that the media for the request is actually online and is not on media that has been externalized from the changer.

[Delete](#) – The calling application may issue this command to delete a UID in the archive. This marks the UID for deletion in the FlashNet database. The file is not immediately deleted and may exist on tape for many months depending on the settings for “Tape Defrag” in the FlashNet software. Deletes from disk usually happen within a few hours.

### 9.1.3 Implementing Best Practices


#### 9.1.3.1 Limit jobs submitted to < 100

For performance reasons it is recommended to keep the number of jobs submitted to the FlashNet queue to be less than 100 at any one time. When more than 100 jobs are submitted it is recommended that the additional jobs are maintained in a queue by the Integrator's automation controller.

The most efficient way to manage archiving to tape is to submit the clips in group order wherever practical.

#### 9.1.3.2 'Round Robin'

Round robin groups can be employed to increase the number of concurrent transfers for a specific Group or Disk Volume. This also has the effect of scattering the clips across multiple volumes, which is good for restore efficiency.

 **Round Robin cannot be used for direct archiving to Non-Spanning groups such as LTFS groups.**

#### 9.1.3.3 Number of tape groups

Ideally, the total number of tape groups should match, or be less than, the number of tape drives

#### 9.1.3.4 Job Priority

Job priority should be defined as a value from 1 (highest) to 100 (lowest) so that restores have a higher priority than backups. A setting of 4 for restores and 5 for backups is a good starting point.

Assets added to the queue should have the Display Name set with a user-friendly name for the job, such as the appropriate clip name, to enable correlation between the automation controller and FlashNet process log.

Jobs submitted should have the Product ID set to identify the Automation Vendor

#### 9.1.3.5 File Size attribute

Jobs submitted should where possible have the file size attribute set where it is known. This improves the workflow by removing the need for a Cache-fill operation. The file size can be approximate.

### **9.1.3.6Tape Groups**

Tape Groups, are often, and probably should be, different than the groups in the client facing tools. For example in the MAM it may be desirable to group media (LTO cartridges) by categories such as News, Sports and Weather. If there is no difference in the archive media policy, they may all be archived to one group TAPE\_L5% which resolves to, say, TAPE\_L5\_1 and TAPE\_L5\_2.

The number of groups should also be the minimum that satisfy the differing requirements of the workflow. If consecutive jobs being archived are to different groups then this will initiate frequent tape exchanges which are unproductive and slow down the throughput into the archive.

Example: Archived media is retained indefinitely vs. Archived media that will be deleted after 30 days. If it is possible to identify and separate the media, than it can be more efficient to keep the 30 day media in a separate group as this aids the tape Lifecycle Defrag efficiency.

If media has the same retention (expiry) criteria then there is little to be gained by organizing it into different groups. An exception to this may be if specific groups (say a Series) of media are required to be removed from the archive for off-shelf storage. The value of that should be carefully considered (due to its possible effects on archiving efficiency) and if utilized it is certainly worth considering archiving the series sequentially, especially if archiving directly to tape.

### **9.1.4 Archiving to Tape Groups from Automation**

The principle here is to put enough jobs into the queue such that we do not needlessly unload tapes if the next material is to be archived to the same Group (current volume)

#### **9.1.4.1Order of material submitted for archiving to Tape Groups**

In order to minimize the number of tape exchanges that are required when archiving to different tape groups (and if operational requirements allow) it is recommended to sort the archive list according to its destination tape group.

### **9.1.5 Archiving to Disk Volumes from Automation**

Here the requirements for submitting clips grouped by media are not so stringent in terms of grouping the media. FlashNet Lifecycle rules will later group the material to be written to tape according to its tape group.

### 9.1.6 Paths to media files

There are two different types of paths to media that are commonly used by FlashNet.

- UNC paths [\\server\share\clip.mxf](#) these are resolved by Windows on the FlashNet server.
- NFS style paths `server:/path/clip.mxf` these are resolved internally by FlashNet by looking at our COMMS\_METHOD table. If the host name matches the matching comms method is used to connect to the destination device i.e. Omneon, Nexio, K2, or even Generic FTP.



## 10 Document Type Definitions (DTD)

### 10.1 *Archive.dtd*

<!ELEMENT FlashNetXML (Archive)>

<!ELEMENT Archive (FileDetails)>

<!ELEMENT FileDetails (File+)>

<!ELEMENT File (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>

<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>

<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>

<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>

<!ATTLIST FlashNetXML Operation CDATA #REQUIRED>

<!ATTLIST Archive VolumeGroup CDATA #REQUIRED>

<!ATTLIST Archive DisplayName CDATA #REQUIRED>

<!ATTLIST Archive VerifyFiles.DWD (0|1) "0">

<!ATTLIST Archive DeleteFiles.DWD (0|1) "0">

<!ATTLIST Archive Priority.DWD CDATA #IMPLIED>

<!ATTLIST FileDetails FileCount.DWD CDATA #REQUIRED>

<!ATTLIST File FullFileName CDATA #REQUIRED>

<!ATTLIST File Guid CDATA #REQUIRED>

<!ATTLIST File Size.QWD CDATA #IMPLIED>

<!ATTLIST File MetaData CDATA #IMPLIED>

<!ATTLIST File MetaDataFile CDATA #IMPLIED>

## **10.2 *Restore.dtd***

```
<!ELEMENT FlashNetXML (Restore)>
<!ELEMENT Restore (FileDetails)>
<!ELEMENT FileDetails (File+)>
<!ELEMENT File (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA #REQUIRED>

<!ATTLIST Restore DisplayName CDATA #IMPLIED>
<!ATTLIST Restore Priority.DWD CDATA #IMPLIED>

<!ATTLIST FileDetails FileCount.DWD CDATA #REQUIRED>

<!ATTLIST File FullFileName CDATA #REQUIRED>
<!ATTLIST File Guid CDATA #REQUIRED>
<!ATTLIST File PartialStart.QWD CDATA #IMPLIED>
<!ATTLIST File PartialEnd.QWD CDATA #IMPLIED>
<!ATTLIST File PartialType CDATA #IMPLIED>
```

## **10.3 *Status.dtd***

```
<!ELEMENT FlashNetXML (Status)>
<!ELEMENT Status (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA "Status">

<!ATTLIST Status RequestId.DWD CDATA #REQUIRED>
<!ATTLIST Status Guid CDATA #IMPLIED>
```

## **10.4 *Delete.dtd***

```
<!ELEMENT FlashNetXML (Delete)>
<!ELEMENT Delete (FileDetails)>
<!ELEMENT FileDetails (File+)>
```

<!ELEMENT File (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>  
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>  
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>  
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>  
<!ATTLIST FlashNetXML Operation CDATA "Delete">

<!ATTLIST FileDetails FileCount.DWD CDATA #REQUIRED>

<!ATTLIST File Guid CDATA #REQUIRED>  
<!ATTLIST File VolumeName CDATA #IMPLIED>  
<!ATTLIST File ArchiveNumber.DWD CDATA #IMPLIED>

### ***10.5 ListGroup.dtd***

<!ELEMENT FlashNetXML EMPTY>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>  
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>  
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>  
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>  
<!ATTLIST FlashNetXML Operation CDATA "ListGroup">

### ***10.6 ReadLog.dtd***

<!ELEMENT FlashNetXML (ReadLog)>  
<!ELEMENT ReadLog (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>  
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>  
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>  
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>  
<!ATTLIST FlashNetXML Operation CDATA "ReadLog">

<!ATTLIST ReadLog LogFileKey.DWD CDATA #REQUIRED>  
<!ATTLIST ReadLog Encoding (URL|Plain) #IMPLIED>

### ***10.7 ListServer.dtd***

<!ELEMENT FlashNetXML EMPTY>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>  
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>  
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>  
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>  
<!ATTLIST FlashNetXML Operation CDATA "ListServer">

### **10.8 StopJob.dtd**

<!ELEMENT FlashNetXML (Stop)>  
<!ELEMENT Stop (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>  
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>  
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>  
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>  
<!ATTLIST FlashNetXML Operation CDATA "StopJob">

<!ATTLIST Stop Job.DWD CDATA #REQUIRED>

### **10.9 PauseJob.dtd**

<!ELEMENT FlashNetXML (Pause)>  
<!ELEMENT Pause (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>  
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>  
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>  
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>  
<!ATTLIST FlashNetXML Operation CDATA "PauseJob">

<!ATTLIST Pause Job.DWD CDATA #REQUIRED>

### **10.10 ResumeJob.dtd**

<!ELEMENT FlashNetXML (Resume)>  
<!ELEMENT Resume (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>

```
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA "ResumeJob">

<!ATTLIST Resume Job.DWD CDATA #REQUIRED>
```

### **10.11 *MigrateAssets.dtd***

```
<!ELEMENT FlashNetXML (Migrate)>
<!ELEMENT Migrate (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA "MigrateAssets">

<!ATTLIST Migrate SourceVolume CDATA #REQUIRED>
<!ATTLIST Migrate SourceArchive.DWD CDATA #REQUIRED>
<!ATTLIST Migrate DestVolume CDATA #REQUIRED>
<!ATTLIST Migrate MoveAssets.DWD (0|1) "0">
<!ATTLIST Migrate Priority.DWD CDATA #IMPLIED>
```

### **10.12 *MigrationStatus.dtd***

```
<!ELEMENT FlashNetXML (MigrationStatus)>
<!ELEMENT MigrationStatus (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA "MigrationStatus">

<!ATTLIST MigrationStatus RequestId.DWD CDATA #REQUIRED>
<!ATTLIST MigrationStatus Key.DWD CDATA #REQUIRED>
<!ATTLIST MigrationStatus Guid CDATA #IMPLIED>
```

### 10.13 SearchArchive.dtd

```
<!ELEMENT FlashNetXML (SearchArchive)>
<!ELEMENT SearchArchive (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA "SearchArchive">

<!ATTLIST SearchArchive Guid CDATA #REQUIRED>
<!ATTLIST SearchArchive From.DWD CDATA #REQUIRED>
<!ATTLIST SearchArchive Group CDATA #IMPLIED>
<!ATTLIST SearchArchive Volume CDATA #IMPLIED>
<!ATTLIST SearchArchive ArchivedFromDate CDATA #IMPLIED>
<!ATTLIST SearchArchive ArchivedToDate CDATA #IMPLIED>
<!ATTLIST SearchArchive DeletedFromDate CDATA #IMPLIED>
<!ATTLIST SearchArchive DeletedToDate CDATA #IMPLIED>
<!ATTLIST SearchArchive PageSize.DWD CDATA #IMPLIED>
<!ATTLIST SearchArchive Flags.DWD CDATA #IMPLIED>
<!ATTLIST SearchArchive IncludeMetadata.DWD (0|1) "0">
```

### 10.14 ListVideoServer.dtd

```
<!ELEMENT FlashNetXML (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA "ListVideoServer">
<!ATTLIST FlashNetXML Pattern CDATA #IMPLIED>
```

### 10.15 GetUidByPath.dtd

```
<!ELEMENT FlashNetXML EMPTY>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
```

```
<!ATTLIST FlashNetXML Operation CDATA "GetUidByPath">
<!ATTLIST FlashNetXML FullFileName CDATA #REQUIRED>
<!ATTLIST FlashNetXML Style (LastUid) #IMPLIED>
```

### **10.16 FullAssetRestore.dtd**

```
<!ELEMENT FlashNetXML (Restore)>
<!ELEMENT Restore (#PCDATA)>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA "FullAssetRestore">

<!ATTLIST Restore AssetId.DWD CDATA #REQUIRED>
<!ATTLIST Restore Destination CDATA #IMPLIED>
<!ATTLIST Restore Priority.DWD CDATA #IMPLIED>
<!ATTLIST Restore DisplayName CDATA #IMPLIED>
```

### **10.17 ListWatchFolder.dtd**

```
<!ELEMENT FlashNetXML EMPTY>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Pattern CDATA #IMPLIED>
<!ATTLIST FlashNetXML Operation CDATA "ListWatchFolder">
```

### **10.18 ListRestoreFolder.dtd**

```
<!ELEMENT FlashNetXML EMPTY>

<!ATTLIST FlashNetXML APIVersion CDATA #REQUIRED>
<!ATTLIST FlashNetXML SourceServer CDATA #REQUIRED>
<!ATTLIST FlashNetXML UserName CDATA #REQUIRED>
<!ATTLIST FlashNetXML CallingApplication CDATA #REQUIRED>
<!ATTLIST FlashNetXML Operation CDATA "ListRestoreFolder">
```

## 11 Glossary

### 11.1 Dtool

Dtool is the backup/restore process (dtool.exe) which is used for data transfer.

### 11.2 The Queue

The FlashNet process and database table that keeps track of all jobs submitted to the archive it is processed by the queue service.

### 11.3 Round Robin

This can be thought of as a logical “Group of Tape Groups” or “Group of disk volumes”. It is used to enable API clients to perform more than one transfer concurrently to a disk volume or tape group. All members of the Round Robin group must be able to be matched by specifying a wildcard at the API. Example CACHE% (= Disk volumes CACHE1, CACHE2, CACHE3 etc.) or ARCHIVE% (= Tape groups ARCHIVE1, ARCHIVE2 etc.)

 **Round Robin cannot be used with Non-spanning groups.**

### 11.4 Tape Group

A logical organization of tapes.

### 11.5 Volume

Either a tape or a disk volume used for writing data, a volume can only be written to by one archive job at a time. (At the API the use of ‘group’, ‘tape’ or ‘disk volume’ is interchangeable; FlashNet will resolve the argument to the most suitable, physical volume.)

#### 11.5.1 Current volume

In a Spanning tape group, the tape that has been assigned for data written to that group. There can only be one current volume in a Spanning group at any given time.



### ***11.6 Non-spanning groups***

A non-spanning group is one where media files will not span the tape. This is less efficient in terms of storage but more convenient especially when externalising media. Also the current LTFS format is non-spanning.

Non-spanning groups can be configured to have many concurrent transfers.

For the most efficient workflow, files archived to non-spanning groups should have their [file size](#) set.

### ***11.7 Tape Exchange delays in workflow***

A function of the tape library to unload/load a tape as part of an archive/restore process. It can easily take up to 5 minutes for some libraries to complete this operation.

Consider this overhead time in a workflow where for files <6GB this can exceed the data transfer time. Fewer tape drives or more groups will compound this issue.

## 12 Frequently asked Questions

### ***12.1 What are the differences between a disk archive and tape archive with respect to the API?***

Tape archives are organised in groups.

With a disk archive there is no removable media, only disk volumes therefore there are no groups as such. Also Disk volumes are always accessible. Tape volumes can be removed from the library and therefore be offline in the API.

Physical overheads – positioning/tape exchange

Tape archives can have significant delay before data transfer starts. Also spanning of tapes will require the data transfer to be paused whilst tape exchange taking place.

Tapes or removable media can be externalised from the Library – shelf storage

If a volume is offline, then by default it cannot be written to or restored from. In the case of removable media the calling application can use SearchArchive in order to determine the assets status and location before attempting restore. If multiple copies exist the details of their location can be given to the operator UI.

This is more elegant than merely trying the restore, which –if the volume is offline by default will lead to a failed job and a message passed back like “not all of the required media are online please load volume XXX”.

Tape archives will persist in the database after deletion until defrag

When content is deleted from a disk volume, the data itself is erased quickly by the disk defrag process. However in a tape archive deleted content can be in that state permanently, or until a tape volume defrag process is executed.

For more information about Volumes and groups in general see the FlashNet help documentation.

### ***12.2 Can I archive copies of the same material to different groups?***

Not by default – neither is it a good idea. FlashNet by default does not permit duplicate material to be archived with the same GUID. If identical material is archived then either the original GUID should be deleted, or a new GUID assigned to the identical material and the “versions” tracked as separate assets externally.

FlashNet internally does duplicate content for workflow and D.R. reasons using storage manager. For instance material may be archived to disk and then automatically copied to tape, then later purged from disk automatically by storage manager delete rule.

It is considered good practice to check that a GUID is unique (does not exist) by calling SearchArchive before archiving (or by ensuring it on the client side).

### ***12.3 What settings should I use for Priority***

In a simple system the suggested starting point is 4 for restore and 5 for archive jobs, but it depends upon the workflow and how many archive controllers share the archive and their operational priorities. It would be advantageous for these settings to be configurable within your application.

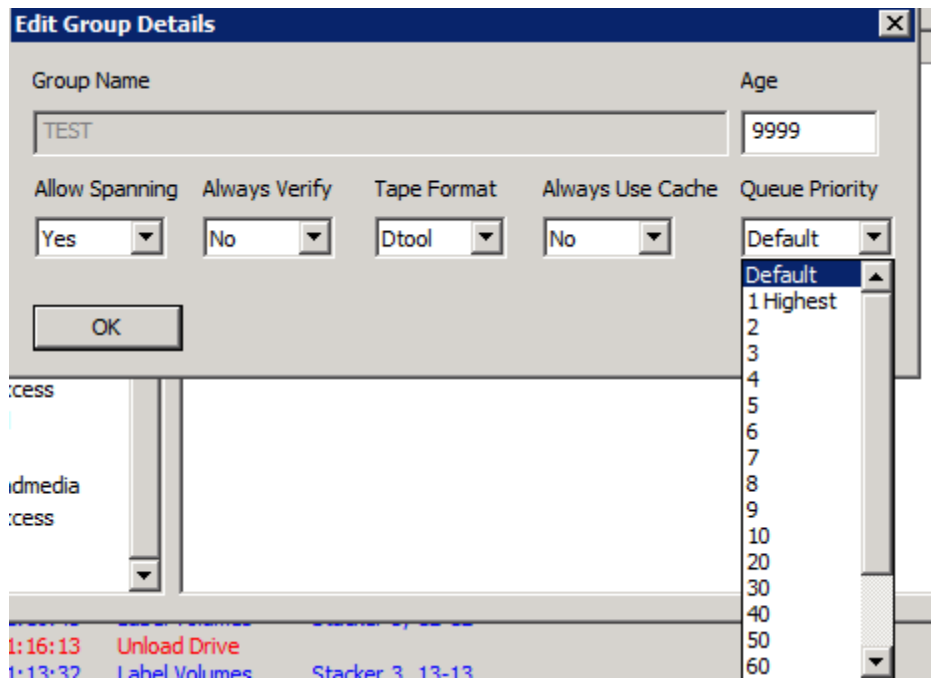
### ***12.4 What is verification & when is it used?***

Verification is an option when archiving, it enabled the data archived to be read back and compared against the target file. This does mean that data is read twice and introduces an overhead. Further overhead exists for verifying tape archives due to positioning time.

### ***12.5 Group priority additional information***

The priority determines the order for jobs on the FlashNet queue. What matters is not the number but the relative values between jobs. 1 is reserved for SGL use. Tape copying/tape defrag (re-packing) in FlashNet can be assigned priorities on a per rule basis.

In Interplay archive priorities are hard-coded to currently 4 is for restore and 5 is for archive. From FlashNet Server version 6.4 the priority submitted in the API can be overridden when they are on the queue by a systems administrator. Also it is possible to set group attributes in FlashNet to override the submitted (default) priority. The standard setting would be default.



### ***12.6 What is the priority algorithm of assets for restore?***

FlashNet will always try to restore content from disk in preference to tape. A Global setting RESTORE\_TAPE\_PRIORITY can be set to force restores to come from tape – even if they exist on tape.

If multiple copies are made then by default the newest copy will be restored. The older copies will not be used unless the newer copy volume is offline or unless the content is in different libraries which have a different “Restore Priority”.

In some workflows it may be preferable to reverse this logic. Contact [partners@sqlbroadcast.com](mailto:partners@sqlbroadcast.com) if you wish to do this.

If duplicate copies exist in different changers and the newest copy is in use in a drive in one library, the second copy may be used in the other library (changer load balancing).

### ***12.7 How are multiple tape libraries handled by FlashNet?***

A FlashNet cluster supports up to 4 tape libraries (increased to 99 from FlashNet Server version 6.4), each one will be allocated its own groups for writing data. For restoring data each library has a “Restore Priority” which determines the order for restore priority if content is duplicated among them.

### ***12.8 How should material be identified in the archive to users?***

The Display name should be used to identify the content clearly, particularly if the GUID or filename does not clearly identify it.

Example = Programme Title: Material ID

**12.9 What value should I pass for APIVersion in the [FlashNetXML Header](#)?**

We recommend you use the highest APIVersion supported by your API. Doing this targets the latest features and allows the API Server to bypass any legacy routines that have been deprecated and left in for compatibility reasons. If you target API version 1.1 or better, the UserName attribute becomes mandatory.

**12.10 Is it possible to limit the use of delete (purge) operations by user?**

This is not currently supported in the API but will be added in a later release. By passing the UserName and Setting the APIVersion at a later date to a specific version, you ensure that assets in the FlashNet database can be protected from deletion by other users when this option becomes available.

**12.11 What is the difference between “ListGuid” and “SearchArchive”?**

“[ListGuid](#)” is a legacy, supported function but [SearchArchive](#) is more efficient, has richer functionality and puts less load on the FlashNet server.

- “SearchArchive” is the recommended function for all new integrations.

**12.12 When is SearchArchive operation normally needed/used?**

SearchArchive is used to checking if assets exist prior to restore. It can be used to check if a UID exists in the archive before allowing it to be re-archived.

This function can also be used for synchronising the application database with the archive.

SearchArchive may also be called before a restore request is issued to make sure that the media for the request is actually online and is not on media that has been externalized from the changer.

**12.13 When is [ReadLog](#) normally needed/used?**

This is used to give additional detailed information back to the user so that from your application our process log can be read. This is used in some integration to provide more detail. The results by default are in Base64 but can optionally be set to plain-text.

**12.14 What are the differences between Byte offset and Frame based partial file restore?**

Byte offset partial file restore is used where you are re-wrapping the file. Frame based partial file restore is used for GXF, MXF and some QuickTime wrapped content.

**12.15 For Frame based partial file restore how do I determine start and end frames?**

The start Frame is calculated relative to the first frame of video of the source clip in the archive. Frame 0 is the first frame.

The Frame count must be calculated based upon the video frame rate as FlashNet does not make any adjustment for video standard etc.

For a 30 Second duration partial file restore including the first frame it would be:

25FPS          Start Frame 0          End Frame 749  
For an I-Frame MXF OP-1A clip.

The ability and accuracy of partial file restore is dependent upon the essence and file wrapper.

**12.16 Deleting assets: do I need to specify the volume and archive?**

Generally speaking a delete operation should be viewed as removing all instances of an asset. An option exists to specify the volume and archive for a “delete specific” operation. This is for specialised use and should normally not be used.