

Active Circle



Active Media Connector
REST API Reference

TABLE OF CONTENTS

1. INTRODUCTION.....	4
ACTIVE CIRCLE API DOCUMENTATION	4
STRUCTURE OF THE REST URIS	4
FORMAT OF BODY	4
FORMAT OF DATE	4
2. AUTHENTICATION	4
3. WEB SERVICES OVERVIEW	5
ACTIVE CIRCLE WEB SERVICES.....	5
4. KEYS & ENUMERATIONS	6
ACTIVE CIRCLE KEYS	6
The TransferJob service.....	6
The FileAudit service.....	6
The File service.....	7
The TapeAudit service	8
The Tape service.....	8
5. ERROR CODES	9
ERROR CODES 9	
Common errors.....	9
TransferJob service.....	10
6. CONFIGURATION	13
7. RESOURCES REFERENCE	16
THE VERSION WEB SERVICE	16
1. Get the Active Media Connector version	16
THE NODE WEB SERVICE	16
1. Get the list of the Active Circle nodes.....	16
2. Getting information about a specific node.....	17
THE DATASOURCE WEB SERVICE.....	18
1. General	18
2. Getting list of available data sources.....	19
3. Getting information about a known data source	20
4. Creating a new data source	21
5. Updating a known data source.....	23

6. Deleting a known data source	23
THE TRANSFERPOLICY WEB SERVICE	24
1. General	24
2. Creating a new transferPolicy	27
3. Getting information about a known transfer policy	29
4. Getting a list of transfer policies	31
5. Updating a known transfer policy	33
6. Deleting a known transfer policy	33
THE TRANSFERPOLICYSTATS WEB SERVICE	34
1. General	34
2. Getting statistics	36
THE TRANSFERJOB WEB SERVICE	37
1. Creating a new transferJob	37
2. Getting information about a known transfer job	45
3. Getting a list of transfer jobs	53
4. Updating a known transfer job	54
5. Deleting a known transfer job	55
THE FILE WEB SERVICE (FORMERLY FILEINFO)	55
1. Getting information about a single file or directory	56
2. Modify information about a single file or directory	63
3. Deleting a single file or directory	64
THE FILEAUDIT WEB SERVICE	64
1. What were the Active Circle file operations during a given time window?	64
THE TAPE WEB SERVICE (FORMERLY TAPEINFO)	71
1. Getting information about a tape	71
2. Getting the list of tapes known by the Active Circle system	73
THE TAPEAUDIT WEB SERVICE	74
1. What were the Active Circle tape events during a given time window?	74
THE PARTIALTRANSCODE WEB SERVICE	76
1. Get the list of partial transcode features	76
2. About the transcoding features	77

1. Introduction

Active Circle API documentation

This document describes the REST API and resources provided by Active Circle.

Active Circle's REST APIs provide access to resources (data entities) via URI paths. To use a REST API, your application will make an HTTP request and parse the response. Currently, the only supported response format is JSON. Your methods will be the standard HTTP methods like GET, PUT, POST and DELETE (see API descriptions below for which methods are available for each resource).

Because the REST API is based on open standards, you can use any web development language to access the API.

Structure of the REST URIs

URIs for Active Circle's REST API resource have the following structure:

```
http[s]://host:port/acapi/api-version/resource-name
```

The current API version is 1.0.

```
// Examples:  
  
// HTTPS:  
https://apiserver:8443/acapi/1.0/transferJob  
  
// HTTP:  
http://apiserver:8081/acapi/1.0/transferJob
```

Note

If you append the request parameter **?pretty=true** to uri, you will receive a well-indented JSON.

Format of body

The request and the response bodies are written in JSON format.

Format of date

The date fields are represented by a long value, that is the specified number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.

2. Authentication

Active Circle provides an API based on REST web services. Access to these services requires authentication.

Currently only basic HTTP authentication is implemented. It consists of supplying the HTTP Authorization header followed by the login and password using base64 encoding.

Example:

```
GET /acapi/1.0/transferJob/1 HTTP/1.1
Accept: */*
Authorization: Basic bG9naW46cGFzc3dvcmQNCg==
Host: localhost:8181
Connection: Keep-Alive
Accept-Encoding: gzip,deflate
```

In this example, "bG9naW46cGFzc3dvcmQNCg==" corresponds to "login:password" encoded in base64.

This authentication mode is documented by the following RFCs:

- RFC 1945[2] (Hypertext Transfer Protocol – HTTP/1.0)
- RFC 2616 (Hypertext Transfer Protocol – HTTP/1.1)
- RFC 2617 (HTTP Authentication: Basic and Digest Access Authentication).

Note

The default configuration of the API allows you to use HTTPS with a self signed certificate. HTTP is still enabled to ensure backward compatibility with other applications but will be disabled in a future version.

3. Web Services Overview

Active Circle web services

Active Circle provides an API based on REST web services.

Here is a description of each one.

TransferJob

This service enables you to perform archiving and restore task on multiple files.

DataSource

The DataSource service enables you to declare and use your data sources with transfer policies.

TransferPolicy

This service enables you to schedule transfer jobs.

Supports advanced functionality such as file filtering, incremental transfer, data synchronization and data moving.

File

Active Circle location information on a file, or the list of the files in a directory.

If the files are versioned, this service returns only the information of the current version.

FileAudit

Traceability on the activity of the files of the Active Circle system.

It can be requested by specifying a time window or a file path.

Tape

Information on the tapes of the Active Circle system (barcode, pool, capacity and others)

TapeAudit

Traceability on the activity of the tapes of the Active Circle system (library input/output and others).

Node

Information on the nodes of the Active Circle system, and their IP addresses.

4. Keys & Enumerations

Active Circle keys

The Active Circle web services may return the following keys.

The TransferJob service

A job status may be one of the following:

- NEW: The job is not started
- SCHEDULED: The job is processing
- PAUSED : The job is paused (can be resumed by setting the status to NEW)
- STOPPED : The job is stopped (cannot be restarted)
- DONE: The job is done

The FileAudit service

A modification on a file in the Active Circle system may be one of the following:

- CREATE_FILE
- DELETE_FILE
- RENAME_FILE
- CREATE_DIRECTORY
- DELETE_DIRECTORY
- RENAME_DIRECTORY
- MOVE_FILE
- MOVE_DIRECTORY
- ARCHIVEFILEON_TAPE
- DELETEFILEON_TAPE
- MODIFY_FILE
- SIZE_FILE
- CHECKSUM_SUM
- CHECKSUM_VALID
- CHECKSUM_INVALID
- CHECKSUM_RESET
- ARCHIVE_FILE
- ARCHIVE_DIR
- UNARCHIVE_FILE
- UNARCHIVE_DIR
- PURGE_FILE
- PURGE_DIR

The File service

Accessibility

The file accessibility state can be:

- ONLINE when the file is on disk or only in cache
- NEARLINE when the file is located on an online tape
- ONCLOUD when the file is located on the cloud
- OFFLINE when the file is located on an external tape
- UNREACHABLE when its pool is not accessible (out-of-service)

Location

A file can be stored in multiple type of locations:

- CACHE
- DISK
- TAPE
- CLOUD

Each type of location has different characteristics.

CACHE location

Cache location have a 'nodes' property representing a node list.

```
{
  nodes: [
    "node1",
    "node3"
  ],
  type: "CACHE"
}
```

DISK location

Disk location have the following properties:

- format: the pool format
- pool: the disk pool name
- partitions: the of partition with name and accessibility

```
{
  format: "PACK",
  pool: "disk_node1",
  type: "DISK",
  partitions: [
    {
      name: "Partition data3-p1",
      accessibility: "ONLINE"
    }
  ]
}
```

TAPE location

Tape location have the following properties:

- format: the tape format (TAR or LTFS)
- pool: the disk pool name
- secondary: indicates whether the pool is the primary or secondary pool

- tapes: a list of tapes with accessibility, barcode, tarNumber and offsetInTar

```
{
  format: "TAR",
  pool: "tar_2",
  secondary: true,
  tapes: [
    {
      accessibility: "NEARLINE",
      barcode: "AB7371",
      offsetInTar: "365857792",
      tarNumber: "75"
    }
  ],
  type: "TAPE"
}
```

CLOUD location have the following properties:

- pool: the pool name

```
{
  pool: "cloud2",
  type: "CLOUD"
}
```

The TapeAudit service

The tape event can be one of the following:

- CREATED: a new tape is added to the Active Circle system
- REMOVED: a tape is removed from the Active Circle system
- DELETED: the tape content will be no more accessible
- MOVEDOUTFROM_LIBRARY: a tape is moved out of the library. Its accessibility is "OFFLINE"
- MOVEDINTOLIBRARY: a tape is moved into the library. Its accessibility is "NEARLINE"

The Tape service

A tape accessibility can be:

- NEARLINE (in the library: in a slot, a mailbox or a drive)
- OFFLINE (out of the library)

When the accessibility is NEARLINE, the tape location may be:

- DRIVE
- MAILBOX
- SLOT
- PICKER

The pool type can be:

- UNKNOWN: the tape is not used by the Active Circle system
- CLEAN: the tape is a cleaner tape
- NOT_AFFECTED: the tape must not be used by the Active Circle system
- USER: the tape is used by the Active Circle system and it is in a user tape pool

When the tape pool type is USER, its type can be:

- AIT-1

- AIT-2
- DLT-III
- LTO-1
- LTO-2
- LTO-3
- LTO-4
- LTO-5
- LTO-6
- SAIT-1
- SDLT I
- SDLT II
- VXA-1
- VXA-2

When the tape pool type is CLEAN, its type can be:

- AIT - cleaning
- DLT - cleaning
- LTO - cleaning
- SAIT - cleaning
- SDLT - cleaning
- VXA - cleaning

When the tape pool type is UNKNOWN, its type can be:

- Generic AIT
- Generic DLT
- Generic LTO
- Generic SAIT
- Generic SDLT
- Generic VXA

When the tape pool type is NOT_AFFECTED, its type can be any type of the previous lists.

5. Error Codes

Error codes

The Active Circle web services may return the following error codes.

Common errors

Listing

code	message (The value in between bracket are replaced by the proper arguments)
acapi.errorCode.resourceNotFound	{0}{1} not found
acapi.errorCode.serverNoSpaceLeft	No space left on device
acapi.errorCode.serverNoDbConnexion	Unavailable database connection
acapi.errorCode.serverDbError	Database error ({0})
acapi.errorCode.noNodeAvailable	No Active Node found !

Example 1 - When no space left on the AMC serverHTTP status: **503 Service Unavailable**

Content:

```
{
  "self": "https://server:port/acapi/1.0/transferJob",
  "error": {
    "code": "acapi.errorCode.serverNoDbConnexion",
    "message": "Unavailable database connection"
  }
}
```

Example 2 - Other database errorHTTP status: **500 Internal Server Error**

Content:

```
{
  "self": "https://server:port/acapi/1.0/transferJob",
  "error": {
    "code": "acapi.errorCode.serverDbError",
    "message": "Database error ({0})"
    "args": [ "0A000" ]
  }
}
```

TransferJob service**Listing**

code	message (The value in between bracket are replaced by the proper arguments)	Notes
acapi.errorCode.unSupportedProtocol	Unsupported protocol	
acapi.errorCode.unableToConnect	Unable to connect to '{0}'	
acapi.errorCode.unAvailableNode	Node '{0}' is not available	
acapi.errorCode.unknownNode	Unknown Node	
acapi.errorCode.noSpaceLeft	No space left on {1}: {0}	
acapi.errorCode.unableToSetModificationTime	Unable to set modification time '{2}' on '{1}', cause: '{3}'	
acapi.errorCode.filesDiffer	Source file size ({1}) differs from destination file size ({3})	

code	message (The value in between bracket are replaced by the proper arguments)	Notes
acapi.errorCode.modificationDatesDiffer	Source file mtime ({1}) differs from destination file mtime ({3})	
acapi.errorCode.unImplementedOption	Option '{0}' is not available	
acapi.errorCode.insufficientDataSourceCapabilities	Insufficient capabilities at '{1}' ('{0}' required)	
acapi.errorCode.unImplementedFilter	Filter {0} is not available	
acapi.errorCode.unReachablePath	Path '{0}' is unreachable ! {1}	
acapi.errorCode.unexpectedFile	File '{0}' already exists	Since 5.3.0
acapi.errorCode.unexpectedFolder	Folder '{0}' already exists	Since 5.3.0
acapi.errorCode.transferAbortedByShutdown	Transfer aborted after shutdown	
acapi.errorCode.transferAbortedByUser	Transfer aborted by User	
acapi.errorCode.policyNotSimulated	Policy can not be simulated	
acapi.errorCode.policyNotExecuted	Policy can not be executed	
acapi.errorCode.filesNotArchived	Files not archived: {0}	
acapi.errorCode.unableToArchiveFiles	Unable to archive files: {0}	
acapi.errorCode.fileTapesRequired	Required tapes: {0}	
acapi.errorCode.transferError	Error on {1}: {0}	
acapi.constraint.transferPolicy.options.unImplementedOption	Property [{0}] of class [{1}] contains an unknown option: {2}	
acapi.constraint.transferPolicy.options.optionsIncompatibility	Property [{0}] of class [{1}] contains incompatible options: {2}	
acapi.constraint.transferPolicy.options.optionNotAvailableForDataSource	Option '{2}' is not available for destination '{3}'	Since 5.5.0

code	message (The value in between bracket are replaced by the proper arguments)	Notes
acapi.constraint.transferPolicy.filters.unImplementedFilter	Property [{0}] of class [{1}] contains unimplemented filter: {2}	
acapi.constraint.transferPolicy.filters.invalidFilter	Property [{0}] of class [{1}] contains an invalid filter: {2} ({3})	
acapi.constraint.transferPolicy.filters.malFormedFilters	Property [{0}] of class [{1}] must be a list of filters	
acapi.constraint.fileRecentFilter.atLeastOneFieldMandatory	At least one of these fields must be filled ({2})	
acapi.constraint.transferJob.files.badprotocol	The source path of {0} provided to the {1} must include the protocol part of the url, this path does not: {2}	

Example 1 - Transfer error

HTTP status: **200 OK**

Content:

```
{
  self: "https://server:8443/acapi/1.0/transferJob/11332",
  ...
  error: {
    code: "acapi.errorCode.transferAbortedByShutdown",
    message: "Transfer aborted after shutdown"
  }
}
```

Examples 2 - TransferPolicy fields validation (on POST or PUT)

Incompatible option

```
{
  "self": "https://server:8443/acapi/1.0/transferPolicy",
  "error": {
    "code": "acapi.constraint.transferPolicy.options.optionNotAvailableForDataSource",
    "message": "Option 'pp_CHECKSUM' is not available for destination 'AMC FS'",
    "args": [
      "options",
      "TransferPolicy",
      "pp_CHECKSUM",
      "AMC FS"
    ]
  }
}
```

```
}
}
```

Creating a policy with the same name 'test 2' as an existing one

```
{
  "self": "https://server:8443/acapi/1.0/transferPolicy",
  "error": {
    "code": "acapi.constraint.transferPolicy.name.unique",
    "message": "Property [name] of class [TransferPolicy] with value [test 2] must be
unique",
    "args": [
      "name",
      "TransferPolicy",
      "test 2"
    ]
  }
}
```

Note

For all errors of type 'acapi.constraint.*', the first 2 args are, respectively :

- the name of the field
- the name of resource

6. Configuration

The AMC setup is defined in the configuration file **/etc/ac/acapi-config.groovy**.

Note

In the previous version (< 4.0.0), the AMC setup was defined in the file `.grails/acapi-config.groovy` located in the home directory of the Tomcat user (usually `/usr/share/tomcat` or `/usr/share/tomcat6`). When upgrading AMC to a higher version than 4.0.0, this file is moved to **/etc/ac/acapi-config.groovy**.

A sample of the default configuration file :

```
acapi {

  // Node Http client configuration
  node {
    // SSL: Trust all certificates
    trustAllCert = true
    // SSL: Define a truststore to allow specific certificates
    truststore {
      // The absolute path to the JKS
      //path = '/path/to/truststore.jks'

      // The JKS password
      //password = '*****'
    }

    // Connection pool
    connectionPool {
```

```

        // Max simultaneous connections
        maxTotal = 20

        // Max simultaneous connections per route (node)
        maxPerRoute = 1

        // Max time to get a connection from the pool (ms)
        requestTimeout = 5000
    }

    // Max time to establish the connection with the remote host (ms)
    connectTimeout = 5000

    // Max time waiting for data after establishing the connection (ms)
    socketTimeout = 30000
}

// Allows the number of threads in the quartz threadpool and thus the number of concurrent
// jobs
// quartzThreadCount = 3

// FTP socket connection timeout in millisecond, 30 seconds by default
// ftpConnectTimeout = 30000

// Data Connection socket timeout in millisecond, 5 min by default
// ftpDataTimeout = 300000

// Buffer size used for IO to/from the filesystem in putFile and StoreFile
// Set by default to 1Mb
// ftpBufferSize = 1024 * 1024

// Ftp send socket buffer size (SO_SNDBUF), 128k by default
// ftpSendBufferSize = 128 * 1024

// Ftp receive socket buffer size (SO_RCVBUF), 128k by default
// ftpReceiveBufferSize = 128 * 1024

// Ftp protect data during ftps transfers, if true, ftp will be disabled and replaced by a
// ftp to ftp copy
ftpProtectData = true

// Enable FXP for FTP transfers
ftpEnableFXP = false

transferjob {
    // use a round robin strategy for node selection
    roundRobin = true
    // Number of parallel transfer per job
    //parallelism.transfer = 2
    // Number of parallel checksum per job
    //parallelism.checksum = 2
    // Max number of errors (if exceeded, job will stop). Set to -1 to disable this limit
    //maxErrors = 100
    // Wait for checksum jobs before coping next files
    //waitChecksum = true
}

users {

    // Defines the users. The syntax is
    // userLogin {
    //     password = "passwordValue"

```

```
// }
// admin { password = "****" }
}
```

Note

This section will be **deprecated** soon since the authentication is delegated to Active Circle local directory. Setting up a user **is only necessary** if:

- you want to use the [TransferPolicy](#) service.
- you want to use the [TransferJob](#) service with an account from an external directory.

In this case you will need to declare a user with access to 'Active Circle' shares. The password must be the same as defined in Active Circle directory. Then add this account in the section below

```
// The Active Circle user used for ftp transfer
// must be a user defined in "users" and in Active Circle
// with the same login and password
// roles { activeCircleFtpUser = 'admin' }
```

Note

The 'activeCircleFtpUser' parameter is used to specify the Active Circle account to use for FTP transfers initiated by transfer policies. **This user must have access to Active Circle shares !** The account must be declare above.

```
fxpServer {
    ...
}
```

Note

The section "fxpServer" is **deprecated**. It is the old way to define FTP servers you can use in transfer jobs. You must use the [DataSource](#) web service to declare your FTP servers.

```
}
...
```

Note

The above is a sample of the acapi-config.groovy configuration file. You can find the whole file in the web-app directory of the ACPAPI package.

7. Resources Reference

This section describes each resource (web service) exposed by the Active Media Connector API, with request URLs, response body examples and the available HTTP response statuses.

The Version web service

Since version 4.1.x

This service gives the version of Active Media Connector.

1. Get the Active Media Connector version

1.1 Request URL

```
GET /acapi/1.0/version
```

1.2 Response body example

```
{
  "self": "https://server:port/acapi/1.0/version",
  "version": "5.0.0"
}
```

1.3 Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

The Node web service

This service gives information about the Active Circle configuration. The Active Circle system is composed of at least two nodes.

You can choose the node to send requests to. This is optional and can be done for the File service.

This optional field will also be available for the [TransferJob](#), [File](#) web services.

1. Get the list of the Active Circle nodes

1.1 Request URL

```
GET /acapi/1.0/node
```

1.2 Response body example

```
{
  "self": "https://server:port/acapi/1.0/node",
  "max": 100,
  "offset": 0,
  "sort": "id",
  "order": "asc",
  "date_min": null,
  "date_max": null,
  "search_path": null,
  "count": 2,
  "size": 2,
  "nodes": [
```

```

{
  "name": "nfo",
  "description": "nfo",
  "lastStartingTime": 12345678910,
  "nodeVersion": "4.0.1_9",
  "hasTapeDrive": true,
  "path": "",
  "ipAddresses": [
    "192.168.1.25"
  ],
  "httpPort": 80,
  "https": false,
  "ftpPort": 21,
  "ftps": false,
  "url": "https://server:port/acapi/1.0/node/nfo"
},
{
  "name": "node2",
  "description": "",
  "lastStartingTime": 1354801901628,
  "nodeVersion": "4.0.1_9",
  "hasTapeDrive": true,
  "path": "node2/Vannes/ActiveCircle",
  "ipAddresses": [
    "10.0.0.198",
    "10.0.0.98"
  ],
  "httpPort": 80,
  "https": true,
  "httpsPort": 443,
  "ftpPort": 22,
  "ftps": true,
  "ftpsPort": 990,
  "url": "https://server:port/acapi/1.0/node/node2"
}
]
}

```

The "url" field is given for backward compatibility reasons. If you follow the link of this field, you will not have access to additional fields.

1.3 Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

2. Getting information about a specific node

2.1 Request URL

```
GET /acapi/1.0/node/{name}
```

Where {name} is the name of the node to get information from.

2.2 Response body example

```

{
  "self": "https://server:port/acapi/1.0/node/node2",
  "name": "node2",
  "description": "",
  "lastStartingTime": 1354801901628,
  "nodeVersion": "4.0.1_9",
  "hasTapeDrive": true,
  "path": "node2/Vannes/ActiveCircle",
  "ipAddresses": [
    "10.0.0.198",
    "10.0.0.98"
  ]
}

```

```

    ],
    "httpPort": 80,
    "https": true,
    "httpsPort": 443,
    "ftpPort": 22,
    "ftps": true,
    "ftpsPort": 990
  }

```

2.3 Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

The DataSource web service

Since version 4.1.x

This service manages servers you can use as **source** or **destination** for your transfer jobs.

The dataSource request resource provides functionality to create, update, list and delete.

A description of each web service method follows.

1. General

1.1. Types

Before you create your own DataSource, be aware that there are **built-in** data sources.

List of built-in DataSource:

Type	Description
Circle	This DataSource stands for "Active Circle"
FS	This DataSource corresponds to the AMC local file system

Note

To access the local file system, AMC must be setup to run with enough privilege. Refer to installation manual.

Other DataSource types you can create:

Type	Description
Node	This DataSource type corresponds to an "Active Circle" node
FTP	This DataSource type corresponds to an FTP Server

1.2. Connection and features

When checking a DataSource connection (see following chapters), the following information is returned :

```

"connection": {
  "features": {
    "listDir": true, // Tells if AMC can browse files and folders (mandatory)
    "size": true, // Tells if AMC can read file size (mandatory)
    "lastModifiedTime": true, // Tells if AMC can update file timestamp
  }
}

```

```

    "ownerGroup": false, // Tells if AMC can set owner and group on files
    "permissions": true // Tells if AMC can set permission on files
    "secure": false // Tell if this data source use a secured protocol
  },
  "status": "SUCCESS" // The main connection status, possible values are "SUCCESS" or
  "FAILED"
}

```

Note

The 'secure' feature has been added since 5.5 and is set to 'true' for data sources supporting the FTPS protocol.

2. Getting list of available data sources

2.1. Request URL

Parameter	Value	Description
max	integer	Maximum result count. Default value: 100
offset	integer	Position in list of the first returned result
sort	string	Field to sort by. Default value: id
order	"desc" or "asc"	Ascending or descending order. Default value: asc
showConnection	"true" or "false"	If true, information about connectivity is returned

2.2. Request URL example

```

GET /acapi/1.0/dataSource
Gives the first page of data sources. This page contains 100 results ordered by ascending id.

GET /acapi/1.0/dataSource?max=10&offset=10
Gives the second page of data sources. This page contains 10 results ordered by ascending id.

```

2.3. Response body example

```

{
  "self": "https://server:port/acapi/1.0/dataSource",
  "dataSources":
  [
    {
      "id": 12121,
      "name": "AMC FS",
      "description": "AMC Local File System",
      "type": "FS",
      "dateCreated": "2013-07-11T10:07:33Z",
      "lastUpdate": "2013-09-02T08:17:30Z",
      "url": "https://server:port/acapi/1.0/dataSource/12121"
    },
    {
      "id": 12122,
      "name": "Circle",
      "description": "Circle",
      "type": "Circle",
      "dateCreated": "2013-07-11T10:07:33Z",
      "lastUpdate": "2013-07-11T10:07:33Z",
      "url": "https://server:port/acapi/1.0/dataSource/12122"
    }
  ]
}

```

```

    },
    {
      "id": 12123,
      "name": "fxp",
      "type": "FTP",
      "dateCreated": "2013-07-11T10:07:33Z",
      "lastUpdate": "2013-09-04T14:17:43Z",
      "params": {
        "port": "21",
        "pwd": "npa!",
        "host": "fxp",
        "user": "npa"
      },
      "url": "https://server:port/acapi/1.0/dataSource/12123"
    }
  ]
}

```

Parameters max, offset, sort and order can be redefined.

2.4. Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

3. Getting information about a known data source

3.1. Request URL

Parameter	Value	Description
showConnection	"true" or "false"	If set to "true", connectivity information is returned

```
GET /acapi/1.0/dataSource/{id}
```

The last field {id} is the number of the data source to get information from.

3.2. Request URL example

```
GET /acapi/1.0/dataSource/418
```

Response body :

```

{
  "self": "https://server:port/acapi/1.0/dataSource/418",
  "id": "12123",
  "name": "ftpserver1",
  "type": "FTP",
  "dateCreated": "2013-07-11T10:07:33Z",
  "lastUpdate": "2013-09-04T14:17:43Z",
  "params": {
    "port": "21",
    "pwd": "pwd!",
    "host": "ftpserver1",
    "user": "root"
  }
}

```

3.3. Request URL example with connectivity information

```
GET /acapi/1.0/dataSource/418?showConnection=true
```

Response body :

```
{
  "self": "https://server:port/acapi/1.0/dataSource/418",
  "id": "12123",
  "name": "ftpserver1",
  "type": "FTP",
  "dateCreated": "2013-07-11T10:07:33Z",
  "lastUpdate": "2013-09-04T14:17:43Z",
  "params": {
    "port": "21",
    "pwd": "pwd!",
    "host": "ftpserver1",
    "user": "root"
  },
  "connection": {
    "features": {
      "listDir": "true"
      "size": "true"
      "lastModifiedTime": "true"
      "ownerGroup": "true"
      "permissions": "true"
    },
    "status": "SUCCESS"
  }
}
```

3.4. Available HTTP response status header

- 200 (OK) application/json
- 404 (Not Found)
- 401 (Unauthorized)

4. Creating a new data source

4.1. Request URL

POST /acapi/1.0/dataSource

4.2. Request body

Field	Description
name	A unique name for this data source
description (optional)	Some text describing the data source
groupId (optional)	Common root folder for a group of data sources
type	the type of data source
params (optional)	Parameters of this data source

The "type" field may be:

Value	Description	Mandatory parameters
FTP	For FTP server	"host", "port", "user", "pwd"
FS	For AMC server file system	

List and description of **FTP** parameters:

Parameter	Description	Sample value
host	The IP or host name (string)	"ftp.myftp.com" or "192.168.1.40"

Parameter	Description	Sample value
port	The port number (positive integer)	"21000"
user	The user to login with (string)	"my_user"
pwd	The user password (string)	"myPassword"
active	The mode to use. true for active mode, false for passive mode (true or false)	"true" or "false"
secure	Enable FTPS for this datasource	"true" or "false"
secureData	Enable encryption of the data channel	"true" or "false"
allowFxp	Allow FXP when the condition are met	"true" or "false"

Note

If you don't provide the options 'secureData' and 'allowFxp', the transfer engine will use the options 'ftpProtectData' and 'ftpEnableFXP' defined in Configuration.

4.3. Request body examples**4.3.1 Simple FTP server**

```
{
  "name": "FTP Server 1",
  "description": "My first FTP server",
  "type": "FTP",
  "params": { "host": "myFtpServer1", "port": "21", "user": "user1", "pwd": "password1" }
}
```

4.3.2 A FTPS server

```
{
  "name": "FTPS Server 2",
  "description": "My first FTPS server",
  "type": "FTP",
  "params": { "host": "myFtpServer2", "port": "21000", "user": "user2", "pwd": "password2",
  "secure": "true" }
}
```

4.3.3 A Mount point on the AMC server

```
{
  "name": "My mount point datasource",
  "type": "FS",
  "rootPath": "/mnt/amc/my_mounted_nas"
}
```

4.4. Available HTTP response status header

- 201 (Created)

The "location" header indicating the location of the newly created resource. In other words, the "location" header contains the id of the newly created job.

- 400 (Bad request)

The request is malformed.

- 401 (Unauthorized)

The request needs authentication.

5. Updating a known data source

5.1 Request URL

```
PUT /acapi/1.0/dataSource/{id}
```

where "id" is the id of the data source to update.

5.2. Request body

Field	Description
name	The unique name of this data source
description	Some text describing the data source
type	the type of data source
params	Parameters of this data source

Unmodified properties are optional. If you provide other properties, they will be ignored. If you want to update some parameters from the "params" field, you must provide all parameters

5.3. Request body example

```
{
  "description": "Updated description"
}
```

5.4 Available response status header

- 204 (OK, response body is empty)
- 400 (Invalid request)

You can not modify a data source when it is in use by a transfer.

- 401 (Unauthorized)
- 403 (Forbidden)
- 404 (Not Found)

6. Deleting a known data source

6.1. Request URL

```
DELETE /acapi/1.0/dataSource/{id}
```

where "id" is the id of the data source to delete.

6.2. Available response status header

- 204 (OK, response body is empty)
- 404 (Not Found)
- 401 (Unauthorized)

Note

If you delete a built-in DataSource (you should not), it will be automatically recreated during the next AMC restart or upgrade.

The TransferPolicy web service

Since version 4.1.x

This service allows you to manage transfer policies.

The transferPolicy request resource provides functionality to create, update and list policies.

Note

To use this service, you need to configure a Active Circle account with access to the shares you want to use. See Configuration.

A description of each web service method follows.

1. General

1.1. DataSource relationship

A transfer policy needs two **DataSource** definitions, one for the source and another for the destination.

Note

Before creating a policy, you must ensure that two **DataSource** definitions exist.

1.2. Scheduling

To schedule a policy, just provide a valid **cron expression** in the "cron" field and set the policy status to "ENABLED".

Useful links:

[Cron tutorial](#)

[Cron generator](#)

1.3. Options

1.3.1. Standard options

You can use options to compose your policy. List of standard options:

Option	Description	Typical usage
syncDeletions	Propagate source deletions on destination	Synchronize data
deleteFilesAfter	Delete files on source after copying to destination	Move data
deleteDirsAfter	Delete directories on source if all its children have been copied to destination	Move data
forceSync	Force synchronization ignoring incremental filter	Not recommended

Warning

Be careful when using the "deleteFilesAfter" and "deleteDirsAfter" options. Test your policy with a **test folder**.

Note

To use the "deleteFilesAfter" or "deleteDirsAfter" options with Active Circle as source, ensure that the user configured for transfers have sufficient rights.

1.3.2. Advanced options**Note**

Following options are available **only for archiving jobs**, in another terms, when the target of the policy is Active Circle.

Checksum options

Option	Description	Typical usage
pp_CHECKSUM	Activate checksum verification	Add integrity check to your archiving workflow by providing MD5 files, generally used to ensure data integrity before deleting the original files at source
ppCHECKSUMONLY	Process only files with provided checksum	In conjunction with PP_CHECKSUM, you want to process only the files with a MD5 signature

Example of MD5 file 'checksum.md5' you can provide into source folder:

```
701015981c65c59e304d43ce3b9afe18 my_video.mp4
ae4a9709eed98d7b4329e859f058c675 images/background.mp4
d41d8cd98f00b204e9800998ecf8427e test.mp4
```

Note that you can have many md5 files.

Tape archiving options

Option	Description	Typical usage
pp_ARCHIVE	Trigger an 'on demand archiving process' just after the transfer with files selected in the job	You want to secure your data on tape just as soon as possible, because you don't have disk storage
ppFULLARCHIVE	Same as above, but the archive is triggered on the whole content of the target folder	Same as above, but you want to ensure that all data of the target folder are incrementally archived on tape
pp_EXPORT	Trigger an 'on demand export process' after the transfer with files selected in the job	You want to export transferred files directly on TAPE. An export pool must be configured to use this option.
ppCLOSETAPE	Close the tape at the end of the job	You want to use a different tape per policy execution
ppNOFAILURE_BEFORE	Trigger the archiving or export process only if all files selected by the policy are successfully transferred into ActiveCircle	To have consistent archive

Checksum options can be combined with archiving options.

Options *ppARCHIVE*, *ppFULLARCHIVE* and *ppEXPORT* can't be combined, you must choose once.

On the other hand, you can combine *ppCLOSETAPE* and/or *ppNOFAILUREBEFORE* with *ppARCHIVE*, *ppFULLARCHIVE* or *pp_EXPORT* option.

1.4. Filters

Filters enables your policy to work with a subset of files. List of available filters:

filter	Criteria	Mode
fileName	File name matching a pattern list	include, exclude
fileSize	File size between min and max value	include, exclude
fileRecent	File recently modified	exclude

1.4.1. File name pattern filter

```
{
  "pattern": "*.log,*.txt", // comma separated pattern
  "type": "fileName", // filter type
  "mode": "exclude" // mode: "include" or "exclude"
}
```

Note

You can use the symbols * (multiple characters) and ? (one character) in the pattern.

1.4.1. File size filter

```
{
  "min": "0", // minimum size, default value is 0
  "minUnit": "MB", // unit for minimum size: "B","KB","MB","GB","TB","PB"
  "max": "1", // maximum size, default value is infinite
  "maxUnit": "MB", // unit for maximum size: "B","KB","MB","GB","TB","PB"
  "type": "fileSize", // filter type
  "mode": "include" // mode: "include" or "exclude"
}
```

1.4.1. Exclude recently modified files filter

Version 1 (still working but deprecated)

```
{
  "val": "15", // interval in days (skip files modified during last 15 days)
  "type": "fileRecent", // filter type
  "mode": "exclude" // mode: "exclude" only
}
```

Version 2

```
// Skip files modified during last 15 days, 1 hour and 5 minutes
{
  "version": "2" // use version 2 of this filter
  "days": "15", // interval in days
  "hours": "1", // interval in hours
}
```

```

"minutes": "5", // interval in minutes
"type": "fileRecent", // filter type
"mode": "exclude" // mode: "exclude" only
}

```

2. Creating a new transferPolicy

2.1. Request URL

```
POST /acapi/1.0/transferPolicy
```

2.2. Request body

Field	Value	Description
name	String	A unique name for this policy
description (optional)	String	Some text describing this policy
groupId (optional)	String	Common root folder for a group of policies
cron (optional)	Cron expression	Cron expression used to schedule the policy
priority (optional)	Integer 0...100	The priority of jobs created by this policy, default value is 50
src	DataSource	Defines the DataSource to use as source
srcPath	String	The path at the source
dst	DataSource	Defines the DataSource to use as destination
dstDir	String	The path at the destination
options (optional)	Array of Option	Policy options
filters (optional)	Array of Filter	File filters

2.2.1. DataSource format for src and dst fields

```

"src":
{
  "id": {ds_id}
}

```

Where {ds_id} is the DataSource id

2.2.2. Options field

```

"options":
[
  "syncDeletions", "deleteFilesAfter"
]

```

2.2.3. Filters field

```

"filters":
[
  {
    "min": "0",
    "minUnit": "MB",
    "max": "1",

```

```

    "maxUnit": "MB",
    "type": "fileSize",
    "mode": "include"
  }
]

```

2.3. Request body example of a synchronization policy

```

{
  "name": "Sync AC with my FTP",
  "description": "Keep AC synchronized with my FTP server",
  "cron": "0 0 3 1/1 * ? *" // Every day at 03h00 PM
  "src":
    {
      "id": "12121" // My FTP server
    },
  "srcPath": "/myRepo/*",
  "dst":
    {
      "id": "12122" // Active Circle
    },
  "dstDir": "/share/myRepo/"
  "options":
    [
      "syncDeletions" // files are removed from Active Circle when deleted at source
    ]
}

```

2.4. Request body example of a move policy

```

{
  "name": "Move files to AC",
  "description": "Move files from FTP server to AC",
  "cron": "0 0 3 1/1 * ? *" // Every day at 03h00 PM
  "src":
    {
      "id": "12121" // My FTP server
    },
  "srcPath": "/myRepo/*",
  "dst":
    {
      "id": "12122" // Active Circle
    },
  "dstDir": "/share/myRepo/"
  "options":
    [
      "deleteFilesAfter", "deleteDirsAfter" // deletes files and folders from the source after
      having copied
    ]
}

```

2.5. Available HTTP response status header

- 201 (Created)

The "location" header indicating the location of the newly created resource. In other words, the "location" header contains the id of the newly created policy.

- 400 (Bad request)

The request is malformed.

- 401 (Unauthorized)

The request needs authentication.

3. Getting information about a known transfer policy

3.1. Request URL

```
GET /acapi/1.0/transferPolicy/{id}
The last field {id} is the number of the transfer policy to get information from.
```

This request can contain pagination criteria, to display related jobs.

Parameter	Value	Description
showJobs	boolean	Set to true to display related jobs
maxJobs	integer	Maximum job count. Default value: 1

Note

Jobs are listed from newest to oldest.

3.2. Request URL example

```
GET /acapi/1.0/transferPolicy/418
```

3.3. Response body example

```
{
  "self": "https://server:port/acapi/1.0/transferPolicy/795619",
  "id": 795619,
  "name": "Move FTP - Circle",
  "description": "Test Archive",
  "groupId": "test",
  "cron": "0 20 12 ? * TUE *",
  "priority": 50,
  "status": "DISABLED",
  "src": {
    "id": 12123,
    "name": "fxp",
    "type": "FTP"
  },
  "srcPath": "/data/folder_test/",
  "dst": {
    "id": 12122,
    "name": "Circle",
    "type": "Circle"
  },
  "dstDir": "/share/archive/",
  "options": [
    "deleteFilesAfter",
    "deleteDirsAfter"
  ],
  "dateCreated": "2013-09-30T16:17:15Z",
  "lastUpdate": "2013-10-21T14:16:32Z",
  "schedulerDate": "2014-01-23T13:52:38Z"
}
```

3.4. Response body example with last 2 jobs

```
"self": "https://server:port/acapi/1.0/transferPolicy/795619?showJobs=true&maxJobs=2",
"id": 795619,
"name": "Move FTP - Circle",
"description": "Test Archive",
```

```

"groupId": "test",
"cron": "0 20 12 ? * TUE *",
"priority": 50,
"status": "DISABLED",
"src": {
  "id": 12123,
  "name": "fxp",
  "type": "FTP"
},
"srcPath": "/data/folder_test/",
"dst": {
  "id": 12122,
  "name": "Circle",
  "type": "Circle"
},
"dstDir": "/share/archive/",
"options":
[
  "deleteFilesAfter",
  "deleteDirsAfter"
],
"dateCreated": "2013-09-30T16:17:15Z",
"lastUpdate": "2013-10-21T14:16:32Z",
"schedulerDate": "2014-01-23T13:59:52Z",
"jobs":
[
  {
    "id": 2757767,
    "type": "ArchiveJob",
    "groupId": "test",
    "priority": 50,
    "node": null,
    "userName": null,
    "dateCreated": 1382372294586,
    "lastUpdated": 1382372300885,
    "dateEnd": 1382372300234,
    "workflow": "TRANSFER",
    "stage": "TRANSFER",
    "status": "DONE",
    "running": false,
    "count": 234,
    "error": {
      "message": "FTP destination: 550 Quota limit exceeded"
    }
  },
  {
    "id": 2757526,
    "type": "ArchiveJob",
    "groupId": "test",
    "priority": 50,
    "node": null,
    "userName": null,
    "dateCreated": 1382371838204,
    "lastUpdated": 1382371845136,
    "dateEnd": 1382371844424,
    "workflow": "TRANSFER",
    "stage": "TRANSFER",
    "status": "DONE",
    "running": false,
    "count": 234,
    "error": {
      "message": "FTP source: 451 Transfer aborted. Broken pipe - FTP destination: 550 Quota limit exceeded"
    }
  }
],
"count": 34

```

3.5. Available HTTP response status header

- 200 (OK) application/json
- 404 (Not Found)
- 401 (Unauthorized)

4. Getting a list of transfer policies

4.1. Request URL

Pagination and sort parameters:

Parameter	Value	Description
max	integer	Maximum result count. Default value: 100
offset	integer	Position in list of the first returned result
sort	string	Field to sort by. Default value: id
order	"desc" or "asc"	Ascending or descending order. Default value: asc

Other available request parameters:

Parameter	Value	Description
showJobs	boolean	Set to true to display related jobs
maxJobs	integer	Maximum job count. Default value: 1

4.2. Request URL example

```
GET /acapi/1.0/transferPolicy
Gives the first page of transfer policies. This page contains 100 results ordered by ascending id.

GET /acapi/1.0/transferPolicy?max=10&offset=10
Gives the second page of transfer policies. This page contains 10 results ordered by ascending id.
```

4.3. Response body example

```
{
  "self": "https://server:port/acapi/1.0/transferPolicy?max=2",
  "transferPolicies": [
    {
      "id": 12267,
      "name": "Sync FTP - FTP",
      "description": "Monday Sync",
      "groupId": "test",
      "cron": "0 0/60 * 1/1 * ? *",
      "priority": 50,
      "status": "DISABLED",
      "src": {
        "id": 12123,
        "name": "myFTP",
        "type": "FTP"
      },
      "srcPath": "/data/test/folder_test/*",
      "dst": {
        "id": 12124,
```

```

        "name": "myOtherFtp",
        "type": "FTP"
    },
    "dstDir": "/data/test/folder_test_sync/",
    "options":
    [
        "syncDeletions"
    ],
    "dateCreated": "2013-07-16T14:47:30Z",
    "lastUpdate": "2013-11-19T12:09:47Z",
    "schedulerDate": "2014-01-23T14:08:52Z",
    "url": "https://server:port/acapi/1.0/transferPolicy/12267"
},
{
    "id": 12269,
    "name": "Copy FTP - Circle",
    "description": "Daily copy",
    "groupId": "datamover",
    "cron": "0 8 18 ? * FRI *",
    "priority": 50,
    "status": "DISABLED",
    "src": {
        "id": 12123,
        "name": "myFtp",
        "type": "FTP"
    },
    "srcPath": "/data/folder_test/*",
    "dst": {
        "id": 12122,
        "name": "Circle",
        "type": "Circle"
    },
    "dstDir": "/share/copy/folder_test/",
    "options":
    [
        "keepPermissions",
        "keepOwnerGroup",
        "pp_EXPORT"
    ],
    "filters":
    [
        {
            "min": "0",
            "minUnit": "MB",
            "max": "1",
            "maxUnit": "MB",
            "type": "fileSize",
            "mode": "include"
        }
    ],
    "dateCreated": "2013-07-16T15:39:15Z",
    "lastUpdate": "2013-12-13T09:15:10Z",
    "schedulerDate": "2014-01-23T14:08:52Z",
    "url": "https://server:port/acapi/1.0/transferPolicy/12269"
}
],
"count": 23,
"size": 2
}

```

Parameters max, offset, sort and order can be redefined.

4.4. Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

5. Updating a known transfer policy

5.1. Request URL

```
PUT /acapi/1.0/transferPolicy/{policy_id}
```

where {policy_id} is the id of the transfer policy to update.

Note

Transfer policy must be **disabled** before you can update its fields, except for status field. To disable a policy, update its status to "DISABLED".

Available request parameters:

Parameter	Value	Description
runNow	boolean	Set this parameter to "true" to execute policy now
simulate	boolean	Set this parameter to "true" to simulate policy

Note

When using request parameters **runNow** or **simulate**, request body will be ignored.

5.2. Request body example

Update "src" (datasource) and "srcPath" (source path):

```
{
  "src":
    {
      "id": "149467"
    },
  "srcPath": "/data/new_src_folder/"
}
```

5.3. Available response status header

- 204 (OK, response body is empty)
- 400 (Invalid request)
- 401 (Unauthorized)
- 403 (Forbidden)

The specified transfer policy priority cannot be updated because it has already been scheduled, or the specified archive transfer policy status can only be updated to NEW.

- 404 (Not Found)

6. Deleting a known transfer policy

6.1. Request URL

```
DELETE /acapi/1.0/transferPolicy/{policy_id}
```

where {policy_id} is the id of the transfer policy to delete.

6.2. Available response status header

- 204 (OK, response body is empty)
- 404 (Not Found)
- 401 (Unauthorized)

The TransferPolicyStats web service

Since version 4.1.2

This service allows you to get transfer policies statistics.

The transferPolicyStats request resource provides functionality to get list of statistics.

A description of each web service method follows.

1. General

1.1. Data format

Statistics are return into a JSON array (see field "list").

```
{
  "self": "https://server:port/acapi/1.0/TransferPolicyStats?type=TIME&max=10&offset=10",
  "list":
  [
    ...
    {
      "policyId": 21652,
      "jobId": 12021079,
      "type": "TIME",
      "stage": "PREPARE",
      "groupBy": "TOTAL",
      "value": 18,
      "dateStart": 1394114212905,
      "dateEnd": 1394114214017
    }
    ...
  ]
}
```

Field descriptions:

Field	Description
policyId	The Policy id
jobId	The transfer job id
type	Can be 'COUNT' for file count, 'SIZE' for volume or 'TIME' for time duration
stage	The transfer stage (ALL, PREPARE, TRANSFER, TAPE...)
groupBy	The group criteria (TOTAL, SUCCESS, FAILED...)
value	The value
dateStart	The job date start
dateEnd	The job date end

1.2. Available Statistics types

type	Description
TIME	Duration in milliseconds

type	Description
COUNT	files count
SIZE	files volume

1.2.1. Stage *ALL*

groupBy	TIME	COUNT	SIZE	Description
TOTAL	x			Total job

1.2.2. Stage *PREPARE*

groupBy	TIME	COUNT	SIZE	Description
TOTAL	x	x	x	Total for this stage
IGNORED		x		Ignored files
SYNCED		x	x	Already synced files
TO_SYNC		x	x	To process files
ERROR		x		Unable to process files
BROWSE	x			Incremental browsing
LIST	x			File list construction
SORT	x			File sorting (only when restoring data from Active Circle)

1.2.3. Stage *TRANSFER*

groupBy	TIME	COUNT	SIZE	Description
TOTAL	x	x	x	Total for this stage
SUCCESS		x	x	Files in success
FAILED		x	x	Files with error
SYNC_FILE	x			File and directory synchronization
FINAL_PASS	x			Final pass (setting metadata, files and directories removing)

1.2.4. Stage *TAPE and EXPORT*

groupBy	TIME	COUNT	SIZE	Description
TOTAL	x	x	x	Total for this stage
SUCCESS		x	x	Files in success
FAILED		x		Files with error

groupBy	TIME	COUNT	SIZE	Description
CREATE_ARCHIVE	x			Archive creation (on AC system)
WAIT_ARCHIVE	x			Waiting for archive completion

2. Getting statistics

2.1. Request URL

Pagination and sort parameters:

Parameter	Value	Description
max	integer	Maximum result count. Default value: unlimited
offset	integer	Position in list of the first returned result
sort	string	Field to sort by. Default value: id
order	"desc" or "asc"	Ascending or descending order. Default value: asc

Other available request parameters:

Parameter	Value	Description
jobId	integer	Add a criteria on transfer job id
policyId	integer	Add a criteria on transfer policy id
type	'TIME', 'COUNT' or 'SIZE'	Add a criteria on statistic type
stage	'ALL', 'PREPARE', 'TRANSFER', 'TAPE', 'EXPORT'	Add a criteria on stage
groupBy	ex: 'TOTAL'	Add a criteria on groupBy

2.2. Request URL example

```
GET /acapi/1.0/transferPolicyStats
Gives ALL statistics
```

```
GET /acapi/1.0/transferPolicyStats?type=TIME&max=10&offset=10
Gives the second page of 'TIME' statistics. This page contains 10 results.
```

2.3. Response body example

```
{
  "self": "https://server:port/acapi/1.0/TransferPolicyStats?type=TIME&max=10&offset=10",
  "list":
  [
    {
      "policyId": 21652,
      "jobId": 12021075,
      "type": "TIME",
      "stage": "PREPARE",
      "groupBy": "TOTAL",
      "value": 160,
      "dateStart": 1394113846920,
      "dateEnd": 1394113848168
    },
  ],
}
```

```

{
  "policyId": 21652,
  "jobId": 12021077,
  "type": "TIME",
  "stage": "PREPARE",
  "groupBy": "TOTAL",
  "value": 3070,
  "dateStart": 1394114171194,
  "dateEnd": 1394114175363
},
{
  "policyId": 21652,
  "jobId": 12021079,
  "type": "TIME",
  "stage": "PREPARE",
  "groupBy": "TOTAL",
  "value": 18,
  "dateStart": 1394114212905,
  "dateEnd": 1394114214017
}
...
]
}

```

Parameters max, offset, sort and order can be redefined.

2.4. Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

The TransferJob web service

The transferJob request resource provides functionality to create and list transfer jobs.

A transfer job allows you to perform various operations such as:

- restore a file list to an FTP server or a local directory on the server
- restore the files to the cache of the Active Circle system to allow faster subsequent file access
- archive files from an FTP server or a local mount point
- trigger a tape or cloud archiving process on files already on Active Circle

Note

The job will operate on Active Circle with the login used to connect to the API. If you are authenticated with the account 'user', you must ensure that 'user' have access to the shares the job intend to use.

A description of each web service method follows.

1. Creating a new transferJob

1.1 Request URL

```
POST /acapi/1.0/transferJob
```

1.2 Request body

Field	Value	Description
type	"ArchiveJob"/"RestoreJob"	Type of job depending on the nature of the operation. We'll use an 'ArchiveJob' to upload or archive files to the Activecircle system and a 'RestoreJob' to restore or destage files already on the archiving system.
workflow (optional)	String	Indicates the tasks to perform. Default value is 'TRANSFER' which allows the transfer of files to or from the Activecircle system depending on the job type. All available options are detailed bellow.
files	Array	The list of files to process.
share (optional)	String	Active Circle share to archive the selected files on (for 'ArchiveJob' only).
groupId (optional)	String	Common root folder for a group of files to archive (see the detailed description of "share" field bellow).
priority (optional)	Integer 0...100	The priority level of the request in the range of 0 to 100. The default value is 50.
node (optional)	String	The Active Circle node to use in the transfer. If no node is provided, use a round-robin strategy with available nodes.
bestEffort (optional)	Boolean	Restore the maximum number of the requested files, even if some files are OFFLINE. The default value is true. If false, the job fails at the first OFFLINE file and retrieves a list of needed tapes. This field is not taken into account for a DESTAGE job. (for 'RestoreJob' only).

The "type" field

The field **type** is mandatory, you must use one of following values:

- "ArchiveJob" to transfer files **to** the Active Circle system
- "RestoreJob" to transfer files **from** the Active Circle system

The "workflow" field

Choose the right workflow according to your needs. The default workflow if you do not specify it is "TRANSFER".

The "workflow" field may be:

Job type	Value	Description
ArchiveJob / RestoreJob	TRANSFER	Transfer files to or from the Active Circle shares (default value).
ArchiveJob	TRANSFER_TAPE	Transfer files to the Active Circle shares and trigger the ActiveCircle archiving policy on the transferred files.
ArchiveJob	TRANSFER_EXPORT	Transfer files to the Active Circle shares and trigger the ActiveCircle export policy on the transferred files.
ArchiveJob	TAPE	Trigger the ActiveCircle archiving policy on the specified files.
ArchiveJob	EXPORT	Trigger the ActiveCircle export for the specified files.
RestoreJob	DESTAGE	Restore the files to the cache of the Active Circle system to allow faster subsequent file access.

The "files" field

The "files" field must contains the list of files to process. Each file can contain the following fields :

For an *ArchiveJob*:

Field of the requested files	Description
src (optional)	The URL of the file to transfer. This field is ignored if the job has a "TAPE" or "EXPORT" workflow.
dst (optional)	The path of the file in the Active Circle system.
externalId (optional)	An external ID is a custom field, that contains unique identifiers from a system outside of Active Circle API. When the "src" field is a directory, the specified externalId is the same for all the files of this directory.
depth (optional)	May be use to limit the descent in levels of directories (a non-negative integer). "0" means only apply to the content of the "src" directory. The default value is infinity.

For a *RestoreJob*:

Field of the requested files	Description
src	The path of the file in the Active Circle system
dst	The URL of the destination where the archive will be restored. This field is ignored for a RestoreJob with the "DESTAGE" workflow.
externalId (optional)	An external ID is a custom field, that contains unique identifiers from a system outside of Active Circle API. When the "src" field is a directory, the specified externalId is the same for all the files of this directory.
depth (optional)	May be use to limit the descent in levels of directories (a non-negative integer). "0" means only

Field of the requested files	Description
	apply to the content of the "src" directory. The default value is infinity.
startTc (optional)	The reference time code representing the beginning of the file (time code partial restore only).
markin (optional)	The time code of the beginning of the segment to restore (time code partial restore only).
duration (optional)	The duration of the sequence to restore (time code partial restore only).

h6. About src and dst These fields accept a path or an URL. A path can be used to indicate a file or a folder localized in the Active Circle system. In this case, the path must starts with the share.

- Example: /my_share/path/to/file.txt

For any other location like FTP server or the local file system, you must use the URL form. Actually, the following type of URL are supported:

- FTP URL, example: ftp://user:password@server:21/path/to/file.txt
- FTPS URL, example: ftps://user:password@secure-ftp.com:21000/path/to/file.txt
- Local file system URL, example: file:///home/user/folder/file.txt

For a folder, the URL or path must end with a slash ('/').

Note

If you don't want to provide your FTP password through URL, you can create an FTP **DataSource**. Then you can reference you server without password.

Example:

h6. About time code partial restore options: startTc, markin and duration

Note

Time code partial restore is an optional feature, it is not provided in the standard version.

Please contact the Activecircle Team if you are interested.

The startTc, markin and duration fields are provided to restore a segment of a file using time codes. These fields must match the following format: dd-dd-dd-dd, where d represents an integer value. If markin and duration are provided and startTc is not, the value of startTc defaults to 00-00-00-00.

The "share" field

Note

This field is optional and must be used to indicate the Active Circle share when you want to archive files without specifying the target path (in "dst" field) for each files. In this case, the system will automatically compute the destination path for each files with the following rules :

If the field "groupId" is not provided :

```
/${share}/${YEAR}/${MONTH}/${DAY}/{JOB_ID}/${FILE_src}
```

If the field "groupId" is provided :

```
/${share}/${groupId}/${YEAR}/${MONTH}/${DAY}/{JOB_ID}/${FILE_src}
```

where:

- \${share} is replaced by the provided "share".
- \${YEAR} is replaced by the current year.
- \${MONTH} is replaced by the current month with 2 digits (01-12).
- \${DAY} is replaced by the current day in month with 2 digits (01-31).
- \${JOB_ID} is replaced by the "id" of the job (integer).
- \${FILE_src} is replaced by the path of the file provided in the "src" field.

The "node" field and the round-robin strategy

You can specify the Active Circle node name to use in the transfer. If no node is provided, the transfer engine will use a round-robin strategy with available nodes. In this case, the node will automatically change between 2 job submissions. Note that the node remains the same throughout the transfer.

If you have upgraded your AMC from a previous version, the round-robin strategy may be disabled by default. To check and enable/disable this feature, you must edit the AMC Configuration.

Note

1.3 ArchiveJob request body examples

1.3.1 Archive files by explicitly specifying destination file paths (with field "dst")

```
{
  "type": "ArchiveJob",
  "share": "share",
  "groupId": "user/specific/logic",
  "priority": 14,
  "files": [
    {
      "src": "ftp://ftp.example.com/path/to/file.txt",
      "dst": "/share/and/path/to/file",
      "externalId": "060a2b3401010101010f0013-000000-00000003354a6047-060e2b347f7f-2a80"
    },
    {
      "src": "ftp://ftp.example.com/path/to/file2.txt"
      "dst": null
    }
  ]
}
```

The "dst" path specifies where to save files in the Active Circle system. When the "share" field is not provided, all the "dst" paths must be provided.

If the "share" and the "dst" path are provided, the dst path is chosen.

When the "share" field is provided, and not the dst path, it is calculated as follows:

```
/share/groupId/YEAR/MONTH/DAY/JOB_ID/src_path/to/file
```

In the example above, the unspecified "dst" path would be:

```
/share/user/specific/logic/2012/3/16/42/path/to/file2.txt
```

where 42 is the id of the corresponding archive job. If necessary, the directories are created.

The "dst" field may contain the name of the file or not. If the name of the file is omitted, the "dst" field must end with a slash. For example, the following "file.txt" will be archived to the directory "/share/and/path/to/", keeping its name.

```
{
  "src": "ftp://ftp.example.com/path/to/file.txt",
  "dst": "/share/and/path/to/"
}
```

1.3.2 Archive files by explicitly specifying destination folders (with field "dst")

```
{
  "type": "ArchiveJob",
  "priority": 14,
  "files": [
    {
      "src": "ftp://ftp.example.com/path1/",
      "dst": "/share1/and/path/"
    },
    {
      "src": "ftp://ftp.example.com/path2/",
      "dst": "/share2/newdir/"
    }
  ]
}
```

The "src" and "dst" fields contain directory paths. The paths must end with a slash.

1.3.3 Transfer files to AC and trigger an archiving process

```
{
  "type": "ArchiveJob",
  "workflow": "TRANSFER_TAPE",
  "priority": 14,
  "files": [
    {
      "src": "ftp://ftp.example.com/path1/",
      "dst": "/share1/and/path/"
    },
    {
      "src": "ftp://ftp.example.com/path2/",
      "dst": "/share2/newdir/"
    }
  ]
}
```

The "src" and "dst" fields contain directory paths. The paths must end with a slash.

1.3.4 Trigger an archiving process on Activecircle path

```
{
  "type": "ArchiveJob",
  "workflow": "TAPE",
  "priority": 50,
  "files": [
    {
      "dst": "/share1/and/path/"
    }
  ]
}
```

```

    },
    {
      "dst": "/share2/newdir/"
    }
  ]
}

```

Only the "dst" fields are required.

This job will trigger two incremental archiving on 'share1' and 'share2' on the given folders.

Note

1.4 RestoreJob request body examples

1.4.1 Restore a list of files to a FTP server (full restore)

```

{
  "type": "RestoreJob",
  "priority": 14,
  "files": [
    {
      "src": "/path/in/active-circle/to/file1.txt",
      "dst": "ftp://dstHost/path/to/file1"
    },
    {
      "src": "/path/in/active-circle/to/file2.txt",
      "dst": "ftp://dstHost/path/to/file2"
    }
  ]
}

```

1.4.2 Restore a list of files to a the local file system (full restore)

The following request will restore files to a local directory located on the server where the AMC is running.

```

{
  "type": "RestoreJob",
  "priority": 14,
  "files": [
    {
      "src": "/path/in/active-circle/to/file1.txt",
      "dst": "file:///home/amc/local/folder/"
    },
    {
      "src": "/path/in/active-circle/to/file2.txt",
      "dst": "file:///home/amc/local/folder/file2.renamed.txt"
    }
  ]
}

```

1.4.3 Request of a list of directories and a file, on two different shares (full restore)

```

{
  "type": "RestoreJob",
  "priority": 14,
  "node": "node1",
  "files": [
    {
      "dst": "ftp://fxp/home/nfo/testdir14/",
      "src": "/p1/tree/",
    }
  ]
}

```

```

    "depth":5
  },
  {
    "dst":"ftp://fxp/home/nfo/testdir15/ficctest.txt",
    "src":"/p2/ficctest.txt"
  }
]
}

```

For the first file in the above example, the "src" and "dst" fields contain directory paths. The paths must end with a slash.

1.4.4 Time code partial restoring of two segments of the same file

```

{
  "type": "RestoreJob",
  "node":"node1",
  "files":[
    {
      "dst":"ftp://fxp/home/nfo/testdir2/video_part1.mxf",
      "src":"/share/video.mxf",
      "startTc":"00-00-00-00",
      "markin":"00-00-05-00",
      "duration":"00-00-02-00"
    },
    {
      "dst":"ftp://fxp/home/nfo/testdir2/video_part2.mxf",
      "src":"/share/video.mxf",
      "markin":"00-00-10-00",
      "duration":"00-00-01-00"
    }
  ]
}

```

In the second entry, there is no startTc provided. It is equivalent to:

```

{
  "dst":"ftp://fxp/home/nfo/testdir2/video_part2.mxf",
  "src":"/share/video.mxf",
  "startTc":"00-00-00-00",
  "markin":"00-00-10-00",
  "duration":"00-00-01-00"
}

```

1.4.5 Destaging of two files

```

{
  "type": "RestoreJob",
  "workflow":"DESTAGE",
  "files":[
    {
      "src":"/share1/file1.txt"
    },
    {
      "src":"/share2/dir/file2.txt"
    }
  ]
}

```

You could also destage a directory, recursively. For example, the "src" field could be "/share2" or "/share2/dir". The response would then contain the detail of the content of the requested directory, file by file.

1.5 Available HTTP response status header

- 201 (Created)

The "location" header indicating the location of the newly created resource. In other words, the "location" header contains **the id of the newly created job**.

- 400 (Bad request)

The request is malformed.

- 401 (Unauthorized)

The request needs authentication.

2. Getting information about a known transfer job

2.1 Request URL

```
GET /acapi/1.0/transferJob/{id}
The last field {id} is the number of the job to get information from.
```

This request can contain pagination criteria, to display the files of a job by page.

Parameter	Value	Description
max	integer	Maximum files count. Default value: 100
offset	integer	Position in list of the first returned result
sort	string	Field to sort by. Default value: id.
order	"desc" or "asc"	Ascending or descending order. Default value: asc

2.2 Request URL example

```
GET /acapi/1.0/transferJob/418?max=10&order=desc
```

2.3 Response body example

2.3.1 A restore job, displayed before the job starts

```
{
  "self": "https://server:port/acapi/1.0/transferJob/7270731",
  "max": 100,
  "offset": 0,
  "from_id": 0,
  "sort": "idx",
  "order": "asc",
  "id": 7270731,
  "type": "RestoreJob",
  "groupId": "user/specific/logic",
  "priority": 50,
  "node": null,
  "userName": "admin",
  "dateCreated": 1393254445063,
  "lastUpdated": 1393254445063,
  "workflow": "TRANSFER",
  "stage": "PREPARE",
  "status": "NEW",
  "running": true,
  "bestEffort": true,
```

```

"counters": { },
"count": 1,
"size": 1,
"files": [
  {
    "src": "/partage1/dts/demo.avi",
    "dst": "file:///home/jfoulon/data/demo/demoRestore.avi",
    "size": -1,
    "status": "NEW"
  }
]
}

```

2.3.2 An archive job with workflow set to *TRANSFER_TAPE*, displayed when job is done

Field 'archives' is replacing the 'archiveName' ArchiveJob field. Instead of having archive names separated by a space character, 'archives' field is an array of items (with name and date for each archive).

Note

```

{
  "self": "https://server:port/acapi/1.0/transferJob/12657910?pretty=true&max=3",
  "max": 3,
  "offset": 0,
  "from_id": 0,
  "sort": "idx",
  "order": "asc",
  "id": 12657910,
  "type": "ArchiveJob",
  "priority": 50,
  "dateCreated": 1396613941109,
  "lastUpdated": 1396614012118,
  "dateEnd": 1396614011844,
  "workflow": "TRANSFER_TAPE",
  "stage": "TAPE",
  "status": "DONE",
  "running": false,
  "counters": {
    "COUNT.TRANSFER.SUCCESS": 22,
    "SIZE.PREPARE.TOTAL": 11000,
    "COUNT.TAPE.IN_PROGRESS": 0,
    "SIZE.TRANSFER.BROWSED": 0,
    "SIZE.TAPE.SUCCESS": 42,
    "TIME.TAPE.WAIT_ARCHIVE": 10143,
    "SIZE.PREPARE.SYNCED": 10958,
    "COUNT.PREPARE.TOTAL": 5512,
    "TIME.PREPARE.TOTAL": 8612,
    "SIZE.TRANSFER.TOTAL": 42,
    "COUNT.TRANSFER.BROWSED": 0,
    "COUNT.PREPARE.TO_SYNC": 22,
    "TIME.PREPARE.LIST": 111,
    "SIZE.PREPARE.TO_SYNC": 42,
    "COUNT.TAPE.SUCCESS": 22,
    "COUNT.TAPE.TOTAL": 22,
    "SIZE.TRANSFER.SUCCESS": 42,
    "TIME.TRANSFER.SYNC_FILE": 954,
    "TIME.ALL.TOTAL": 69882,
    "COUNT.PREPARE.SYNCED": 5490,
    "TIME.TRANSFER.TOTAL": 1244,
    "TIME.TAPE.CREATE_ARCHIVE": 18740,
    "TIME.TAPE.TOTAL": 58952,
    "COUNT.TAPE.FAILED": 0,
  }
}

```

```

"COUNT.TRANSFER.TOTAL": 22,
"SIZE.TAPE.TOTAL": 42,
"TIME.TRANSFER.FINAL_PASS": 146,
"TIME.TRANSFER.SYNC_DIR": 112,
"TIME.PREPARE.BROWSE": 8114,
"COUNT.PREPARE.IGNORED": 0
},
"count": 23,
"size": 3,
"archives":
[
  {
    "name": "Archive:share#2",
    "date": 1396614001191
  }
],
"files":
[
  {
    "src": "file:///home/npa/Dev/test/AMC_repo/src/100k.00005/dir.5k.00004/",
    "dst": "/share/test_archive/test4",
    "size": -1,
    "stage": "TAPE",
    "status": "SUCCESS",
    "mtime": 1380796404000,
    "uid": 500,
    "gid": 500,
    "mode": "775"
  },
  {
    "src": "file:///home/npa/Dev/test/AMC_repo/src/100k.00005/dir.5k.00004/dir.00081/",
    "dst": "/share/test_archive/test4/dir.5k.00004/dir.00081/",
    "size": -1,
    "stage": "TRANSFER",
    "status": "SUCCESS",
    "action": "MKDIR",
    "mtime": 1380796404000,
    "uid": 500,
    "gid": 500,
    "mode": "755"
  },
  {
    "src": "file:///home/npa/Dev/test/AMC_repo/src/100k.00005/dir.5k.00004/dir.00081/file.00004",
    "dst": "/share/test_archive/test4/dir.5k.00004/dir.00081/file.00004",
    "size": 2,
    "stage": "TRANSFER",
    "status": "SUCCESS",
    "action": "CPFILE",
    "mtime": 1380796404000,
    "uid": 500,
    "gid": 500,
    "mode": "644"
  }
]
}

```

2.3.3 A destage job with 2 files on different shares

```

{
  "self": "https://server:port/acapi/1.0/restoreJob/2302",
  "type": "RestoreJob",
  "max": 100,
  "offset": 0,
  "from_id": 0,
  "sort": "id",
  "order": "asc",

```

```

    "id":2302,
    "priority":16,
    "node":"",
    "userName":"admin",
    "dateCreated":1352968676471,
    "lastUpdated":1352977841414,
    "workflow":"DESTAGE",
    "stage":"DESTAGE",
    "status":"DONE",
    "running": false,
    "bestEffort":true,
    "counters": {},
    "destageProcessId":"10 9",
    "count":2,
    "size":2,
    "files":[
      {
        "src":"/share1/file1.txt",
        "status":"SUCCESS"
      },
      {
        "src":"/share2/dir/file2.txt",
        "status":"SUCCESS"
      }
    ]
  }
}

```

The "destageProcessId" field explains that there were two destage jobs, one per share. This field provides their identity in the Active Circle system.

Note

2.3.4 A Restore job with bestEffort field and OFFLINE files

The corresponding request was done with the field bestEffort set to true (its default value).

```

{
  "self":"https://server:port/acapi/1.0/restoreJob/1314",
  "type": "RestoreJob",
  "max":100,
  "offset":0,
  "from_id":0,
  "sort":"id",
  "order":"asc",
  "id":1314,
  "priority":10,
  "node":"",
  "userName":"admin",
  "dateCreated":1359985285021,
  "lastUpdated":1359985301827,
  "workflow":"TRANSFER",
  "stage":"TRANSFER",
  "status":"DONE",
  "running": false,
  "bestEffort":true,
  "counters": { ... }
  "count":4,
  "size":4,
  "files":[
    {
      "src":"/p1/copie2",
      "dst":"ftp://fxp/home/nfo/testdir22/",
      "size":-1,
      "status":"SUCCESS"
    },
    {

```

```

        "src":"/p1/copie2/fic2.txt",
        "dst":"ftp://fxp/home/nfo/testdir22//fic2.txt",
        "size":27,
        "message":"Required tapes:[B002276326]",
        "status":"OFFLINE"
    },
    {
        "src":"/p1/copie2/fic2.txt",
        "dst":"ftp://fxp/home/nfo/testdir22//fic2.txt",
        "size":151,
        "status":"SUCCESS"
    },
    {
        "src":"/p1/copie2/fic2test2.txt",
        "dst":"ftp://fxp/home/nfo/testdir22//fic2test2.txt",
        "size":18,
        "status":"SUCCESS"
    }
],
"tapes":{
    "nearline":[
        "B002276327"
    ],
    "offline":[
        "B002276326"
    ]
}
}

```

When the needed tape has been put into the library, this job can be scheduled again, by updating its field "status" to "NEW". The OFFLINE file will be transferred. The other ones have already been transferred the first time, so they are not transferred again.

Note

The corresponding request was done with the field `bestEffort` set to `false`.

```

{
    "self":"https://server:port/acapi/1.0/restoreJob/1340",
    "type": "RestoreJob",
    "max":100,
    "offset":0,
    "from_id":0,
    "sort":"id",
    "order":"asc",
    "id":1340,
    "priority":10,
    "node":"nfo",
    "userName":"admin",
    "dateCreated":1359985545452,
    "lastUpdated":1359985560742,
    "workflow":"TRANSFER",
    "stage":"TRANSFER",
    "status":"DONE",
    "progress":0,
    "bestEffort":false,
    "count":4,
    "size":4,
    "files":[
        {
            "src":"/p1/copie2",
            "dst":"ftp://fxp/home/nfo/testdir22/",
            "size":-1,
            "message":null,
            "status":"SUCCESS"
        }
    ],
}

```

```

    {
      "src":"/p1/copie2/fic2test.txt",
      "dst":"ftp://fxp/home/nfo/testdir22//fic2test.txt",
      "size":27,
      "message":"Required tapes:[B002276326]",
      "status":"OFFLINE"
    },
    {
      "src":"/p1/copie2/fic2.txt",
      "dst":"ftp://fxp/home/nfo/testdir22//fic2.txt",
      "size":151,
      "message":null,
      "status":"IN_PROGRESS"
    },
    {
      "src":"/p1/copie2/fic2test2.txt",
      "dst":"ftp://fxp/home/nfo/testdir22//fic2test2.txt",
      "size":18,
      "message":null,
      "status":"IN_PROGRESS"
    }
  ],
  "tapes":{
    "nearline":[
      "B002276327"
    ],
    "offline":[
      "B002276326"
    ]
  }
}

```

The field "bestEffort" is set to false, so the job was interrupted when the first OFFLINE file was found.

Note

2.4 Transfer monitoring

The TransferJob progress can be monitored by three items: "stage", "status" and "counters".

2.4.1 Stage

The execution of a job can comprise several stages called "stage" depending on the chosen workflow.

List of all possible values of the stage field:

Stage	Description
PREPARE	The job is building and optimizing the list of files to process.
TRANSFER	During this stage, files are transferred from the source to the destination.
TAPE	An archiving is triggered for the job files.
EXPORT	An export is triggered for the job files.
DESTAGE	Job files are destaged.

List of stages according to the different workflows :

Workflow	Stages
TRANSFER	PREPARE >> TRANSFER

TRANSFER_TAPE	PREPARE >> TRANSFER >> TAPE
TRANSFER_EXPORT	PREPARE >> TRANSFER >> EXPORT
TAPE	PREPARE >> TAPE
EXPORT	PREPARE >> EXPORT
DESTAGE	DESTAGE

2.4.2 Status

The main status of the job is provided by the "status" field.

The status may be one of the following values.

Field	Description
NEW	The job is not started.
SCHEDULED	The job is processing.
PAUSED	The job is paused (can be resumed by setting the status to NEW).
STOPPED	The job is stopped (cannot be restarted).
DONE	The job is done.

Each job file has its own status which may be one of the following values.

Field	Description
NEW	Initial state for files declared in the original submission, the file was not processed.
BROWSED	The file has been checked / discovered during the PREPARE stage.
IN_PROGRESS	The file is being processed.
FAILED	An error has occurred on this file.
SUCCESS	The file has been successfully processed.
OFFLINE	For restore jobs, the file is only available on a tape that is not located in the library. Call the File web service on this file to identify the tape it is located on. Insert this tape in the library to make the file available for restoration.

2.4.3 Counters

The field "counters" can be used to monitor the progression of the job.

Counter example for a simple transfer job:

```
"counters": {
  "COUNT.PREPARE.TOTAL": 4, // The total number of files processed during the PREPARE stage
  "SIZE.PREPARE.TOTAL": 1048576, // The total volume in bytes of files processed during the
  PREPARE stage
  "COUNT.PREPARE.TO_SYNC": 4, // The number of files to transfer
  "SIZE.PREPARE.TO_SYNC": 1048576, // The volume in bytes of files to transfer
  "COUNT.TRANSFER.SUCCESS": 3, // The number of files successfully transferred
  "SIZE.TRANSFER.SUCCESS": 1048576, // The volume in bytes of files successfully transferred
  "COUNT.TRANSFER.BROWSED": 0, // The number of identical files found at source and
  destination
  (and ignored by the incremental transfer engine)
```

```

"SIZE.TRANSFER.BROWSED": 0, // The volume of identical files found at source and
destination
(and ignored by the incremental transfer engine)
"COUNT.TRANSFER.TOTAL": 4 // The total number of files processed during the TRANSFER stage
"SIZE.TRANSFER.TOTAL": 1007576, // The total volume in bytes of files processed during the
TRANSFER stage
}

```

Field **counters** is an array of indicators. Each indicator is characterized by a key and a value. The key is a string composed by:

- the **"type"** of counter (ex: COUNT, SIZE, TIME, ...)
- the **"stage"** concerned (ex: PREPARE, TRANSFER, TAPE, ...)
- and the **"grouping"** criteria (ex: "TOTAL", "SUCCESS", "FAILED", "IN_PROGRESS", "BROWSED", ...)

For example, if you want to calculate a percentage of volume transferred :

- search for indicator { type: "SIZE.TRANSFER.SUCCESS", value: xxxxx }
- search for indicator { type: "SIZE.TRANSFER.TOTAL", value: yyyyy }
- and then calculate the progress: $\text{percent} = \text{xxxxx} * 100 / \text{yyyyy}$

If a job is active, the current transfers are available amongs counters. The progress field give the transfered bytes. The progress is updated service side every 5 seconds. In order to calculate the percentage of transferred volume, the size of the file being transfered is available.

```

"counters": {

  "SIZE.TRANSFER.TOTAL": 2147483648,
  "COUNT.TRANSFER.BROWSED": 2,
  ...
  "CURRENT.TRANSFER": [
    {
      "src": "ftp://fxp_server:21000/home/user1/file.bin",
      "dst": "/share1/file.bin.531.amc",
      "size": 1073741824,
      "progress": 859944104,
      "node": "node1"
    },
    {
      "src": "ftp://fxp_server:21000/home/user1/file.bin",
      "dst": "/partage1/file.bin.531.amc",
      "size": 1073741824,
      "progress": 197730032,
      "node": "node1"
    }
  ]
}

```

2.5 Available HTTP response status header

- 200 (OK) application/json
- 404 (Not Found)
- 401 (Unauthorized)

3. Getting a list of transfer jobs

3.1 Request URL

Pagination and sort parameters:

Parameter	Value	Description
max	integer	Maximum result count. Default value: 100.
offset	integer	Position in list of the first returned result.
sort	string	Field to sort by. Default value: id.
order	"desc" or "asc"	Ascending or descending order. Default value: asc.

3.2 Request URL example

GET /acapi/1.0/transferJob

Gives the first page of transfer jobs. This page contains 100 results ordered by ascending id.

GET /acapi/1.0/transferJob?max=10&offset=10

Gives the second page of transfer jobs. This page contains 10 results ordered by ascending id.

3.3 Response body example

```
{
  "self": "https://server:port/acapi/1.0/transferJob?max=2&order=desc&offset=2&pretty=true",
  "max": 2,
  "offset": 2,
  "from_id": 0,
  "sort": "id",
  "order": "desc",
  "count": 3222,
  "size": 2,
  "transferJobs":
  [
    {
      "id": 7270704,
      "type": "ArchiveJob",
      "groupId": "",
      "priority": 50,
      "node": null,
      "userName": "admin",
      "dateCreated": 1393251807753,
      "lastUpdated": 1393251814420,
      "dateEnd": 1393251814319,
      "workflow": "TRANSFER",
      "stage": "TRANSFER",
      "status": "DONE",
      "running": false,
      "counters": {
        "COUNT.TRANSFER.SUCCESS": 1,
        "SIZE.PREPARE.TOTAL": 1048576,
        "SIZE.TRANSFER.BROWSED": 0,
        "SIZE.TRANSFER.TOTAL": 1048576,
        "COUNT.PREPARE.TOTAL": 1,
        "COUNT.TRANSFER.BROWSED": 0,
        "COUNT.PREPARE.TO_SYNC": 1,
        "SIZE.PREPARE.TO_SYNC": 1048576,
        "SIZE.TRANSFER.SUCCESS": 1048576,
        "COUNT.TRANSFER.TOTAL": 1
      },
      "url": "https://server:port/acapi/1.0/transferJob/7270704"
    }
  ],
}
```

```

{
  "id": 7270696,
  "type": "ArchiveJob",
  "groupId": "",
  "priority": 50,
  "node": null,
  "userName": "admin",
  "dateCreated": 1392373031746,
  "lastUpdated": 1392373038503,
  "dateEnd": 1392373038339,
  "workflow": "TRANSFER",
  "stage": "TRANSFER",
  "status": "DONE",
  "running": false,
  "counters": {
    "SIZE.PREPARE.TOTAL": 2186,
    "SIZE.PREPARE.SYNCE": 2186,
    "COUNT.PREPARE.TOTAL": 1029,
    "COUNT.PREPARE.TO_SYNC": 0,
    "COUNT.PREPARE.SYNCE": 1029,
    "COUNT.PREPARE.IGNORED": 0
  },
  "url": "https://server:port/acapi/1.0/transferJob/7270696"
}
]
}

```

Parameters max, offset, sort and order can be redefined.

3.4 Available HTTP response status header

When a file can't be transferred, details about the error may be found with the file details.

The list of possible error codes can be found here: [Error codes](#)

Note

Since AMC version 4.6.0, the code 'acapi.errorCode.filesAlreadyOnTape' ('Files already on tape') is not used anymore.

3.5 Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

4. Updating a known transfer job

4.1 Request URL

```
PUT /acapi/1.0/transferJob/jobid
```

where jobid is the id of the job to update.

4.2 Request body

Field	Description
priority	The new priority of the job
status	The new status of the job

Unmodified properties are optional. If you provide other properties, they will be ignored.

You can only update the status to NEW. The files that have the OFFLINE or IN_PROGRESS status, will have the NEW status. The job will be rescheduled and the service will attempt to transfer its NEW files again.

4.3 Request body example

```
{
  "priority":29
}
```

4.4 Available response status header

- 204 (OK, response body is empty)
- 400 (Invalid request)
- 401 (Unauthorized)
- 403 (Forbidden)

The specified archive job priority cannot be updated because it has already been scheduled, or the specified archive job status can only be updated to NEW.

- 404 (Not Found)

5. Deleting a known transfer job

5.1 Request URL

```
DELETE /acapi/1.0/transferJob/jobid
```

where jobid is the id of the transfer job to delete.

5.2 Available response status header

- 204 (OK, response body is empty)
- 404 (Not Found)
- 401 (Unauthorized)

The File web service (formerly FileInfo)

This service gives information about files that have been archived in the Active Circle system.

Note

This service was formerly named FileInfo. URLs with FileInfo still run but are deprecated. Please update your client as soon as possible.

The basic authentication provided in the file request is used in the Active Circle system to check the access of the requested content.

For example, the Active Circle user Elena can browse the whole share "share1" except the directory "adminDir" and the file "admin.txt", where the administrator has limited the access.

```
share1
  userDir
    file.txt
  adminDir
    file.txt
  admin.txt
  user.txt
```

So her file request to "share1" will retrieve:

```
userDir
user.txt
```

She cannot see the directory "adminDir" nor the admin.txt file.

1. Getting information about a single file or directory

1.1 Request URL

```
GET /acapi/1.0/file/{path}
```

The {path} field is the path of the file in the Active Circle system. It is optional. If you do not provide the {path}, the service will return the list of the Active Circle shares.

There is another optional field: node.

The Active Circle system is composed of at least two nodes. You can choose the one to send requests to. If it is not accessible, the File service will choose another one for you.

```
GET /acapi/1.0/file/{path}?node={node}
```

See the Node web-service description to get the list of the Active Circle nodes.

1.2 Request URL example

```
GET /acapi/1.0/file/p1/Test.txt
Requests information on the file "Test.txt" that is stored at the root of the share "p1".
```

1.3 Response body examples when path corresponds to a file

```
{
  "self": "https://server:port/acapi/file/p1/Test.txt",
  "share": "p1",
  "path": "Test.txt",
  "size": 11080,
  "creationDate": 1330015139164,
  "modificationDate": 1301391600460,
  "type": "file",
  "bestLocation": "CACHE",
  "locations": [
    {
      "type": "CACHE",
      "nodes": [
        "node1"
      ]
    },
    {
      "format": "PACK",
      "pool": "DiskPoolPack2",
      "type": "DISK",
      "partitions": [
        {
          "name": "Partition PartitionFsTest",
          "accessibility": "ONLINE"
        }
      ]
    }
  ],
  {
    "format": "PACK",
    "pool": "DiskPoolPack1",
    "type": "DISK",
```

```

    "partitions":[
      {
        "accessibility":"OFFLINE"
      }
    ],
  },
  {
    "type":"TAPE",
    "format":"TAR",
    "pool":"PoolArchive1",
    "tapes":[
      {
        "barcode":"B004390216",
        "accessibility":"NEARLINE",
        "tarNumber":"5",
        "offsetInTar":"2983915"
      }
    ]
  }
]
}

```

The file "Test.txt" is stored on:

- two disk pools: DiskPoolpack2 and DiskPoolPack1
- a TAR file on tape B004390216

This file has just been accessed, so it is still in the cache of the node "node1".

When a pool is not accessible, there is less information about "partitions" or "tapes". In the example above, DiskPoolPack1 was not accessible when the request was made.

The locations are sorted: cache, then disk, then nearline tapes locations, then cloud, then offline tapes. The "bestLocation" field indicates the file accessibility. The fastest is "CACHE", then "DISK" then "TAPE" and "CLOUD".

1.4 Response body examples when path corresponds to a directory

For a directory, the result of the GET File request is equivalent to the Unix "ls" command. As there may be a lot of returned files, the results are paginated. They can also be sorted by file name.

If there is no pagination criteria, the default ones are used. Applied pagination criteria are always provided in the response body.

Results are not updated during the user read session. They are saved in the cache. If they have not been read for ten minutes, they are removed from the cache.

Parameter	Value	Description
max	integer	Maximum files count. Default value: 100
offset	integer	Position in list of the first returned result
sort	string	Field to sort by. Possible value: filename
order	"desc" or "asc"	Ascending or descending order. Default value: asc

```

{
  "self":"https://server:port/acapi/1.0/file/share/path/to/dir/?sort=filename&max=4",
  "max":4,

```

```

    "offset":0,
    "sort":"filename",
    "order":"asc",
    "date_min": null,
    "date_max": null,
    "count":"10",
    "size":"4",
    "share":"share",
    "path":"path/to/dir/",
    "creationDate":123456789,
    "modificationDate":123456789,
    "type":"directory",
    "files":[
      "dir1/",
      "dir2/",
      "file1",
      "file2"
    ]
  }
}

```

Here, there are ten results, but max criteria shortens the response.

1.5 Show the location of files in a Directory

It is possible to get a summary of file locations with the boolean option "showLocation". With this option, the files are sorted by their best location.

Location	Description
cache	the list of files which are only the in cache
disk	the list of files with at least one copy in a disk pool (short access time)
nearline	the list of files for which the best location is on a tape located in a library. The files are sorted by their positions on the tapes in order to optimise the copy operation.
oncloud	the list of files for which the best location is on the cloud.
offline	the list of files for which the best location is on an external tape.
unreachable	the only location of the file is an out-of-service pool.

The field "worstAccessibility" gives the lowest level of accessibility for files in the directory.

The list "tapes" contains two sublists :

- the optimized list of nearline tapes
- the list of offline tapes used by the current directory

These lists may be empty.

```

{
  "self": "https://server:port/acapi/1.0/file/share/path/to/dir/?showLocation=true",
  "max": 100,
  "offset": 0,
  "from_id": 0,
  "sort": "id",
  "order": "asc",
  "share": "share",
  "path": "path/to/dir",
  "modificationDate": "1422661533379",
  "creationDate": "1371133323791",
  "type": "directory",

```

```

"count": 23,
"size": 23,
"quota": {
  "files": {
    "softlimit": 2147483647,
    "value": 280125,
    "hardLimit": 2147483647
  },
  "usedSpace": {
    "softlimit": 966367641600000,
    "value": 312469506581,
    "hardLimit": 1073741824000000
  }
},
"locations": {
  "files": {
    "cache": [
      ".DS_Store"
    ],
    "disk": [
      "file2"
    ],
    "nearline": [
      "file3", "file1"
    ],
    "offline": [
      "file4"
    ]
  },
  "worstAccessibility": "OFFLINE",
  "tapes": {
    "nearline": [
      "E01012L4", "E01011L4"
    ],
    "offline": [
      "E01009L4"
    ]
  }
},
"files": [
  "Dir1/",
  "Dir2/",
  ".DS_Store",
  "file1",
  "file2",
  "file3",
  "dir3/",
  "file4",
]
}

```

1.6 Show extended attributes of a file or a directory

It is possible to get all extended attributes of a file or a directory with the boolean option "showXAttr".

```

{
  "self": "https://server:port/acapi/1.0/file/share/path/to/dir/?showXAttr=true",
  "share": "share",
  "path": "path/to/dir",
  "creationDate": 1528461968654,

```

```

"modificationDate": 1528461968654,
"type": "directory",
"quota": {
  "files": {
    "value": 0,
    "softlimit": 2147483647,
    "hardLimit": 2147483647
  },
  "usedSpace": {
    "value": 0,
    "softlimit": 1098412116148224,
    "hardLimit": 1125899906842624
  }
},
"offset": 0,
"count": 0,
"size": 0,
"files": [],
"xattributes": {
  "user.Mail": "name@acme.com",
  "user.Phone": "+33 1 23 45 67 89"
}
}

```

1.7 Available HTTP response status header

- 200 (OK) application/json
- 400 (Invalid request, exception information will be given)
- 404 (Not Found)
- 401 (Unauthorized)
- 444 (No response, the Active Circle system does not respond)
- 500 (Internal server error)

1.8 Examples

Example 1: File GET request with no path provided

Request:

```

curl -i -H "Accept: application/json" \
-H "Content-type: application/json" \
-H "Authorization: Basic YWRtaW46MTIzNA==" \
https://server:port/acapi/1.0/file

```

Answer:

```

{
  "self": "https://server:port/acapi/1.0/file/",
  "max": 100,
  "offset": 0,
  "from_id": 0,
  "sort": "id",
  "order": "asc",
  "share": "",
  "path": "",
  "modificationDate": "",
  "creationDate": "",
  "type": "directory",
  "count": 2,
  "size": 2,
  "files": [
    "on_tapes/",
    "on_disks/"
  ]
}

```

```
}
```

Example 2: File GET request on a directory

Request:

```
curl -i -H "Accept: application/json" \
-H "Content-type: application/json" \
-H "Authorization: Basic YWRtaW46MTIzNA==" \
https://server:port/acapi/1.0/file/on_tapes
```

Answer:

```
{
  "self": "https://server:port/acapi/1.0/file/on_tapes",
  "max": 100,
  "offset": 0,
  "sort": "id",
  "order": "asc",
  "date_min": null,
  "date_max": null,
  "share": "on_tapes",
  "path": "",
  "modificationDate": "1339661453877",
  "creationDate": "1321281830248",
  "type": "directory",
  "count": 4,
  "size": 4,
  "files": [
    "Butterfly.wmv",
    "TEST_recette9.mxf",
    "Bear.wmv",
    "Lake.wmv"
  ]
}
```

Example 3: File GET request on file located in cache and on tape (share with a tape policy)

Request:

```
curl -i -H "Accept: application/json" \
-H "Content-type: application/json" \
-H "Authorization: Basic YWRtaW46MTIzNA==" \
https://server:port/acapi/1.0/file/on_tapes/Butterfly.wmv
```

Answer:

```
{
  "self": "localhost:80/acapi/file/on_tapes/Butterfly.wmv",
  "share": "on_tapes",
  "path": "Butterfly.wmv",
  "creationDate": 1339661432841,
  "modificationDate": 1339661434596,
  "type": "file",
  "size": 2797732,
  "bestLocation": "CACHE",
  "locations": [
    {
      "type": "CACHE",
      "nodes": [
        "node1"
      ]
    }
  ],
}
```

```
{
  "type": "TAPE",
  "format": "TAR",
  "pool": "tape_pool",
  "tapes": [
    {
      "barcode": "E01002L5",
      "accessibility": "NEARLINE",
      "tarNumber": "4",
      "offsetInTar": "2982912"
    }
  ]
}
```

Example 4: File GET request on a directory (same as Example 1)

Request:

```
curl -i -H "Accept: application/json" \
-H "Content-type: application/json" \
-H "Authorization: Basic YWRtaW46MTIzNA==" \
https://server:port/acapi/1.0/file/on_disks
```

Answer:

```
{
  "self": "https://server:port/acapi/1.0/file/on_disks",
  "max": 100,
  "offset": 0,
  "sort": "id",
  "order": "asc",
  "date_min": null,
  "date_max": null,
  "share": "on_disks",
  "path": "",
  "modificationDate": "1339603951282",
  "creationDate": "1321281777985",
  "type": "directory",
  "count": 7,
  "size": 7,
  "files": [
    "server.gif",
    "Autumn Leaves.jpg",
    "Desert Landscape.jpg",
    "Winter Leaves.jpg",
    "Waterfall.jpg",
    "Humpback Whale.jpg",
    "Tree.jpg"
  ]
}
```

Example 5: File GET request on file located only on disk (share with a disk policy)

Request:

```
curl -i -H "Accept: application/json" \
-H "Content-type: application/json" \
-H "Authorization: Basic YWRtaW46MTIzNA==" \
"https://server:port/acapi/1.0/file/on_disks/Desert%20Landscape.jpg"
```

Answer:

```
{
  "self": "localhost:80/acapi/file/on_disks/Desert%20Landscape.jpg",
  "share": "on_disks",
  "path": "Desert Landscape.jpg",
  "creationDate": 1321632433438,
  "modificationDate": 1321632433458,
  "type": "file",
  "size": 228863,
  "bestLocation": "DISK",
  "locations": [
    {
      "format": "PACK",
      "pool": "local_disk_pool",
      "type": "DISK",
      "partitions": [
        {
          "name": "Partition local_disk_poll_part1",
          "accessibility": "ONLINE"
        }
      ]
    }
  ]
}
```

2. Modify information about a single file or directory

2.1 Request URL

```
PUT /acapi/1.0/file/{path}
```

The {path} field is the path of the file in the Active Circle system.

There is another optional field: node.

The Active Circle system is composed of at least two nodes. You can choose the one to send requests to. If it is not accessible, the File service will choose another one for you.

```
PUT /acapi/1.0/file/path?node={node}
```

See the Node web-service description to get the list of the Active Circle nodes.

2.2 Modify extended attributes of a file or directory

It is possible to modify extended attributes of a file or a directory.

Here are some examples of request bodies to initialize, delete and/or modify extended attributes:

2.2.1 Initialize extended attributes (clear and set new ones)

```
{
  "xattribute": {
    "init": {
      "user.Mail": "name@acme.com",
      "user.Phone": "+33 1 23 45 67 89"
    }
  }
}
```

2.2.2 Clear all extended attributes

```
{
  "xattribute": {
    "init": {}
  }
}
```

```
}
```

2.2.3 Delete some extended attributes

```
{
  "xattribute": {
    "delete": [
      "user.Name",
      "user.Mail"
    ]
  }
}
```

2.2.4 Add or modify some extended attributes

```
{
  "xattribute": {
    "modify": {
      "user.Name": "MyName",
      "user.Mail": "name@acme.com"
    }
  }
}
```

3. Deleting a single file or directory

3.1 Request URL

```
DELETE /acapi/1.0/file/path
```

where path is the absolute path of the file or directory to delete. A directory must be empty before it can be deleted. Example: share/rep1/rep2/file.txt Example: share/rep1/rep2

3.2 Available response status header

- 204 (OK, response body is empty)
- 404 (Not Found)
- 401 (Unauthorized)
- 500 (Internal Server Error, may be caused by non-empty directory, access rights)

The FileAudit web service

This service lists files that have been created, modified or deleted during a given time window. Results are paginated. By default there are 100 results per page.

1. What were the Active Circle file operations during a given time window?

1.1 Request URL

Parameter	Value	Description
max	integer	Maximum modified files count. Default value: 100
offset	integer	Position in list of the first returned result
sort	string	Field describing the field to sort by. Default value: id

Parameter	Value	Description
order	"desc" or "asc"	Ascending or descending order. Default value: asc
date_min	long	Start of the time window (included)
date_max	long	End of the time window (excluded)
search_path	string	Filtering on repository or file name, recursively. The metacharacters * (any string) and ? (any character) are supported. The filtered fields are path and newPath.
from_id	long	index from which current results are returned
next_id	long	index from which following results can be obtained
event	string	allows you to filter certain events.

The date fields are represented by a long value, that is the specified number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.

1.2 Request URL example

```
GET /acapi/1.0/fileAudit?max=1&order=desc
Gives the last file events in the Active Circle system.

GET /acapi/1.0/fileAudit?max=1&order=desc&event=CREATE_FILE&event=MODIFY_FILE
Gives the last modified file in the Active Circle system.

GET
/acapi/1.0/fileAudit?max=10&offset=10&sort=date&order=desc&event=CREATE_FILE&event=MODIFY_FILE
Gives the second page of the most recently modified files. This page contains 10 results.

GET /acapi/1.0/fileAudit?max=2&sort=date&order=desc&date_min=1330947295698
Gives the first two results that have a date greater or equal than 1330947295698 (which is
March
5, 2012, 12:34:55).

GET /acapi/1.0/fileAudit?max=10&search_path=share1/*&search_path=share2/*
Search all file events with path matching 'share1/*' or 'share2/*'
```

1.3 Response body example

```
{
  "self": "https://server:port/acapi/1.0/fileAudit?max=1&order=desc",
  "max": 1,
  "offset": 0,
  "from_id": 0,
  "sort": "id",
  "order": "desc",
  "date_min": null,
  "date_max": null,
  "search_path": null,
  "count": 3,
  "hasNewPath": true,
  "size": 1,
  "next_id": 20,
  "files": [
    {
```

```

      "id":1,
      "date":1234567891011,
      "cell":"ac-cell-1",
      "protocol":"FTP",
      "event":"CREATE_FILE",
      "userLogin":"testUser",
      "host":"192.168.1.1",
      "path":"/path/to/file.txt"
      "newPath":null
    }
  ]
}

```

Here there are three results, but max criteria shortens the response.

The "event" field corresponds to a modification on a file. In the Active Circle system it may be one of the following:

- CREATE_FILE
- DELETE_FILE
- RENAME_FILE
- CREATE_DIRECTORY
- DELETE_DIRECTORY
- RENAME_DIRECTORY
- MOVE_FILE
- MOVE_DIRECTORY
- ARCHIVEFILEON_TAPE
- DELETEFILEON_TAPE
- ARCHIVEFILEON_CLOUD
- DELETEFILEON_CLOUD
- MODIFY_FILE
- SIZE_FILE
- CHECKSUM_SUM
- CHECKSUM_VALID
- CHECKSUM_INVALID
- CHECKSUM_RESET
- ARCHIVE_FILE
- ARCHIVE_DIR
- UNARCHIVE_FILE
- UNARCHIVE_DIR
- PURGE_FILE
- PURGE_DIR

When the "event" field is *ARCHIVEFILEONTAPE* or *DELETEFILEONTAPE*, the "protocol" and "newPath" fields are null and the "userLogin" and "host" fields are "NA", ie "Not Available".

The *MODIFY_FILE* event is thrown each time a file is written and closed. The "size" field gives the written size.

The *SIZE_FILE* event is thrown each time the size of a file is modified. The "size" field gives the new size, and the "oldSize" field the previous size.

The *CHECKSUM_SUM* event is thrown each time the checksum computation is started. The "referenceChecksum" field gives the requested reference checksum.

The CHECKSUM_VALID event is thrown each time the computed checksum is valid. The "referenceChecksum" field gives the requested reference checksum and the "newChecksum" fields the computed one.

The CHECKSUM_INVALID event is thrown each time the computed checksum is invalid. The "referenceChecksum" field gives the requested reference checksum and the "newChecksum" fields the computed one.

The CHECKSUM_RESET event is thrown each time checksum information is cleared.

The ARCHIVEFILE/ARCHIVEDIR events are thrown each time a file/directory is archived. The "archive" and "archiveFileKey" fields give the details about the archive and the "archiveSpace", "archiveMediumId" and "archiveFileNb" fields the pool ones.

The UNARCHIVEFILE/UNARCHIVEDIR events are thrown each time a file/directory is no longer archived. The "archive" and "archiveFileKey" fields give the details about the removed archive.

1.4 Available HTTP response status header

- 200 (OK) application/json
- 404 (Not Found)
- 401 (Unauthorized)
- 403 (Forbidden: the authentication is good, but the access is restricted to super users)

1.5 Example

Example 1: basic example

```
# curl -i -H "Accept: application/json" -H "Content-type: application/json" -H "Authorization:
Basic YWRtaW46MTIzNA==" https://server:port/acapi/1.0/fileAudit
{
  "self": "https://server:port/acapi/1.0/fileAudit",
  "max": 100,
  "offset": 0,
  "from_id": 0,
  "sort": "id",
  "order": "asc",
  "date_min": null,
  "date_max": null,
  "search_path": null,
  "count": 3,
  "hasNewPath": true,
  "size": 3,
  "next_id": 5,
  "files": [
    {
      "id": 2,
      "date": 1339661411843,
      "cell": "node1",
      "protocol": "FTP",
      "event": "CREATE_FILE",
      "userLogin": "admin",
      "host": "192.168.168.101",
      "path": "/on_tapes/Lake.wmv",
      "newPath": null
    },
    {
      "id": 3,
      "date": 1339661432846,
      "cell": "node1",
      "protocol": "FTP",
      "event": "CREATE_FILE",
      "userLogin": "admin",
      "host": "192.168.168.101",
      "path": "/on_tapes/Butterfly.wmv",
      "newPath": null
    }
  ]
}
```

```

    },
    {
      "id": 4,
      "date": 1339661453878,
      "cell": "node1",
      "protocol": "FTP",
      "event": "CREATE_FILE",
      "userLogin": "admin",
      "host": "192.168.168.101",
      "path": "/on_tapes/Bear.wmv",
      "newPath": null
    }
  ]
}

```

Example 2: get audit events about the files of the share on_tapes (step = 10), using from_id, next_id and search_path

```

# curl -H Accept: application/json" -H "Content-type: application/json" -H "Authorization:
Basic
YWRtaW46MTIzNA==" "https://server:port/acapi/1.0/fileAudit?search_path=/on_tapes/*.wmv&max=10&
pretty=true"
{
  "self": "https://server:port/acapi/1.0/fileAudit?search_path=/on_tapes/*.wmv&max=10&
pretty=true",
  "max": 10,
  "offset": 0,
  "from_id": 0,
  "sort": "id",
  "order": "asc",
  "date_min": null,
  "date_max": null,
  "search_path": "/on_tapes/*.wmv",
  "count": 11530,
  "hasNewPath": true,
  "size": 10,
  "next_id": 2502,
  "files":
  [
    {
      "id": 2,
      "date": 1339661411843,
      "cell": "node1",
      "protocol": "FTP",
      "event": "CREATE_FILE",
      "userLogin": "admin",
      "host": "192.168.168.101",
      "path": "/on_tapes/Lake.wmv",
      "newPath": null
    },
    {
      "id": 3,
      "date": 1339661432846,
      "cell": "node1",
      "protocol": "FTP",
      "event": "CREATE_FILE",
      "userLogin": "admin",
      "host": "192.168.168.101",
      "path": "/on_tapes/Butterfly.wmv",
      "newPath": null
    },
    { ... },
    {
      "id": 2501,
      "date": 1342535817493,
      "cell": "node1",
      "protocol": "FTP",
      "event": "CREATE_FILE",

```

```

        "userLogin": "admin",
        "host": "172.25.168.217",
        "path": "/on_tapes/T&V_flame_medialab/NomduclientSmoke7/2012/
NomduclientSmoke7_AnnonceurSmoke7_MarqueSmoke7_1103_19110101_FL/Butterfly.wmv",
        "newPath": null
    }
}
]
}

# curl -H Accept: application/json" -H "Content-type: application/json" -H "Authorization:
Basic
YWRtaW46MTIzNA==" "https://server:port/acapi/1.0/fileAudit?search_path=/on_taes/*.wmv&max=10&
from_id=2502&pretty=true"
{
    "self": "https://server:port/acapi/1.0/fileAudit?search_path=/on_tapes/*.wmv&max=10&
from_id=2502&pretty=true",
    "max": 10,
    "offset": 0,
    "from_id": 2502,
    "sort": "id",
    "order": "asc",
    "date_min": null,
    "date_max": null,
    "search_path": "/on_tapes/*.wmv",
    "count": 11530,
    "hasNewPath": true,
    "size": 3,
    "next_id": 5636,
    "files":
    [
        {
            "id": 5622,
            "date": 1342536201824,
            "cell": "node1",
            "protocol": "FTP",
            "event": "CREATE_FILE",
            "userLogin": "admin",
            "host": "172.25.168.217",
            "path": "/on_tapes/T&V_pp_medialab/NomduclientSmoke7/2012/
NomduclientSmoke7_AnnonceurSmoke7_MarqueSmoke7_1103_20120703_PP/ELEMENTS/Demo
3D/Lake.wmv",
            "newPath": null
        },
        {
            "id": 5624,
            "date": 1342536202337,
            "cell": "node1",
            "protocol": "FTP",
            "event": "CREATE_FILE",
            "userLogin": "admin",
            "host": "172.25.168.217",
            "path": "/on_tapes/T&V_pp_medialab/NomduclientSmoke7/2012/
NomduclientSmoke7_AnnonceurSmoke7_MarqueSmoke7_1103_20120703_PP/ELEMENTS/Facing/Lake.wmv",
            "newPath": null
        },
        {
            "id": 5635,
            "date": 1342536205326,
            "cell": "node1",
            "protocol": "FTP",
            "event": "CREATE_FILE",
            "userLogin": "admin",
            "host": "172.25.168.217",
            "path": "/on_tapes/T&V_pp_medialab/NomduclientSmoke7/2012/
NomduclientSmoke7_AnnonceurSmoke7_MarqueSmoke7_1103_20120703_PP/MASTER/bear/Bear.wmv",
            "newPath": null
        }
    ]
}
]

```

```

}

# curl -H Accept: application/json" -H "Content-type: application/json" -H "Authorization:
Basic
YWRtaW46MTIzNA==" "https://server:port/acapi/1.0/fileAudit?search_path=/on_taes/*.wmv&max=10&
from_id=5636&pretty=true"
{
  "self": "https://server:port/acapi/1.0/fileAudit?search_path=/on_tapes/*.wmv&max=10&
from_id=5636&pretty=true",
  "max": 10,
  "offset": 0,
  "from_id": 5636,
  "sort": "id",
  "order": "asc",
  "date_min": null,
  "date_max": null,
  "search_path": "/on_tapes/*.wmv",
  "count": 11530,
  "hasNewPath": true,
  "size": 0,
  "next_id": 5636,
  "files":
  [
  ]
}

```

Example 3: events at file creation and modification

```

# curl -H "Content-Type: application/json"
'https://server:port/acapi/1.0/fileAudit?order=desc&
max=2&pretty=true'
{
  "self": "https://server:port/acapi/1.0/fileAudit?order=desc&max=2&pretty=true",
  "max": 2,
  "offset": 0,
  "from_id": 0,
  "sort": "id",
  "order": "desc",
  "date_min": null,
  "date_max": null,
  "search_path": null,
  "hasNewPath": true,
  "count": 11585,
  "size": 2,
  "next_id": 12206,
  "files":
  [
    {
      "id": 12205,
      "date": 1344001602521,
      "cell": "node1",
      "protocol": null,
      "event": "MODIFY_FILE",
      "userLogin": "admin",
      "host": "192.168.1.19",
      "path": "/partage1/groupId/GROUP/2012/8/3/12192/home/jfoulon/testAutoFxp",
      "newPath": null,
      "size": 198353
    },
    {
      "id": 12204,
      "date": 1344001602500,
      "cell": "node1",
      "protocol": null,
      "event": "CREATE_FILE",
      "userLogin": "admin",
      "host": "192.168.1.19",
      "path": "/partage1/groupId/GROUP/2012/8/3/12192/home/jfoulon/testAutoFxp",

```

```

    "newPath": null
  }
]
}

```

Example 4: search_path example

```

# curl -H "Content-Type: application/json"
'https://server:port/acapi/1.0/fileAudit?from_id=11746&
pretty=true&max=1&search_path=/partage1/toto/*'
{
  "self":
'https://server:port/acapi/1.0/fileAudit?from_id=11746&pretty=true&max=1&search_path=/
partage1/toto/*',
  "max": 1,
  "offset": 0,
  "from_id": 11746,
  "sort": "id",
  "order": "asc",
  "date_min": null,
  "date_max": null,
  "search_path": "/partage1/toto/*",
  "hasNewPath": true,
  "count": 11585,
  "size": 1,
  "next_id": 11747,
  "files":
[
  {
    "id": 11746,
    "date": 1343818379335,
    "cell": "node1",
    "protocol": null,
    "event": "MOVE_DIRECTORY",
    "userLogin": "admin",
    "host": "127.0.0.1",
    "path": "/partage1/tata",
    "newPath": "/partage1/toto/tata"
  }
]
}

```

The Tape web service (formerly TapeInfo)

This service gives the location of tapes known by the Active Circle system.

Results are paginated. By default there are 100 results per page.

Note

This service was formerly named TapeInfo. URLs with TapeInfo still run but are deprecated. Please update your client as soon as possible.

1. Getting information about a tape

1.1 Request URL

```
GET /acapi/1.0/tape/{barcode}
Gives the location and other information about a tape.
```

1.2 Request URL example

```
GET /acapi/1.0/tape/B004616621
```

1.2 Response body example

```
{
  "self": "https://server:port/acapi/1.0/tape/B004616621",
  "barcode": "B004616621",
  "name": "PoolArchive1-0",
  "type": "LTO-3",
  "poolName": "PoolArchive1",
  "poolType": "USER",
  "fragmented": false,
  "fragmentation": 0,
  "closed": false,
  "full": false,
  "capacity": 381178347520,
  "used": 82432,
  "externalId": null,
  "externalLocationCode": "RED_CODE",
  "lastMount": 1340707818531,
  "accessibility": "NEARLINE",
  "location": "DRIVE",
  "address": "null",
  "lastUpdated": 1340708053201
}
```

The tape name is its name in the Active Circle system.

The main tape types are: LTO and SAIT (see chapter Keys to get the full list of the types).

The pool type can be:

- UNKNOWN: the tape is not used by the Active Circle system
- CLEAN: the tape is a cleaner tape
- NOT_AFFECTED: the tape must not be used by the Active Circle system
- USER: the tape is used by the Active Circle system and it is into a user tape pool

The "capacity" field represents the maximum tape capacity. The "used" field represents the used space of the tape.

The "externalLocationCode" field is an Active Circle label that can precise the external location of a tape. It can be "on the shelves", "in the room 1", etc. It may be null.

The "lastMount" field is the most recent tape mounting date, represented by a long value.

The tape accessibility can be:

- NEARLINE (in the library: in a slot, a mailbox or a drive)
- OFFLINE (out of the library)

When the accessibility is NEARLINE, the tape location may be:

- DRIVE,
- MAILBOX,
- SLOT,
- PICKER

When the accessibility is OFFLINE, the tape location and the tape address are "null".

The "address" field is the slot number.

When the "lastMount", "capacity" or "used" fields are not yet available, their value is -1.

1.3 Available HTTP response status header

- 200 (OK) application/json
- 404 (Not Found)
- 401 (Unauthorized)

2. Getting the list of tapes known by the Active Circle system

2.1 Request URL

Parameter	Value	Description
max	integer	Maximum result count. Default value: 100
offset	integer	Position in list of the first returned result
sort	string	Field to sort by. Default value: id
order	"desc" or "asc"	Ascending or descending order. Default value: asc

2.2 Request URL example

```
GET /acapi/1.0/tape
Gives the first page of tapes. This page contains 100 results ordered by ascending barcode.
```

```
GET /acapi/1.0/tape?max=10&offset=10
Gives the second page of tapes. This page contains 10 results ordered by ascending barcode.
```

2.3 Response body example

```
{
  "self": "https://server:port/acapi/1.0/tape?max=2",
  "max": 2,
  "offset": 0,
  "sort": "id",
  "order": "asc",
  "date_min": null,
  "date_max": null,
  "count": 380,
  "size": 2,
  "tapes": [
    {
      "barcode": "B00354430",
      "name": "Unregistered Tapes-370",
      "type": "Generic LTO",
      "poolName": "MediaPoolTypeUnknown",
      "poolType": "UNKNOWN",
      "fragmented": false,
      "fragmentation": 0,
      "closed": false,
      "full": false,
      "capacity": 0,
      "used": 0,
      "externalId": null,
      "externalLocationCode": null,
      "lastMount": -1,
      "accessibility": "NEARLINE",
      "location": "SLOT",
      "address": "1",
      "lastUpdated": 1340631468073
    },
    {
```

```

        "barcode": "B00354431",
        "name": "Unregistered Tapes-371",
        "type": "Generic LTO",
        "poolName": "MediaPoolTypeUnknown",
        "poolType": "UNKNOWN",
        "fragmented": false,
        "fragmentation": 0,
        "closed": false,
        "full": false,
        "capacity": 0,
        "used": 0,
        "externalId": null,
        "externalLocationCode": null,
        "lastMount": -1,
        "accessibility": "NEARLINE",
        "location": "SLOT",
        "address": "2",
        "lastUpdated": 1340631461858
    }
]
}

```

Parameters max, offset, sort and order can be redefined.

The list of tapes is refreshed every five minutes.

2.4 Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

The TapeAudit web service

This service lists events on tapes during a given time window. Results are paginated. By default there are 100 results per page.

Tape events:

- a tape is removed from the library or inserted into the library,
- the content of a tape is deleted,
- the tape is removed from the Active Circle system,
- the tape is newly added to the library.

1. What were the Active Circle tape events during a given time window?

1.1 Request URL

Parameter	Value	Description
max	integer	Maximum tape events count. Default value: 100
offset	integer	Position in list of the first returned result
sort	string	Field describing the field to sort by. Default value: id
order	"desc" or "asc"	Ascending or descending order. Default value: asc
date_min	long	Start of the time window (included)
date_max	long	End of the time window (excluded)

Parameter	Value	Description
from_id	long	index from which current results are returned
next_id	long	index from which following results can be obtained
event	string	allows you to filter certain events.

The date fields are represented by a long value, that is the specified number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.

1.2 Request URL example

```
GET /acapi/1.0/tapeAudit?max=1&order=desc
```

Gives the most recent tape event in the Active Circle system.

```
GET /acapi/1.0/tapeAudit?max=10&offset=10&sort=date&order=desc
```

Gives the second page of the most recent tape events. This page contains 10 results.

```
GET /acapi/1.0/tapeAudit?max=2&sort=date&order=desc&date_min=1340631474946
```

Gives the first two results that have a date greater than or equal to 1340631474946 (which is June 25, 2012, 15:37:54).

GET /acapi/1.0/tapeAudit?event=DELETED List all deleted tapes

1.3 Response body example

```
{
  "self": "https://server:port/acapi/1.0/tapeAudit?max=2&sort=date&order=desc&
  date_min=1340631474946",
  "max": 2,
  "offset": 0,
  "sort": "date",
  "order": "desc",
  "date_min": 1340631474946,
  "date_max": null,
  "count": 6,
  "size": 2,
  "tapes": [
    {
      "id": 1875,
      "barcode": "B00461666",
      "date": 1340632453080,
      "event": "REMOVED"
    },
    {
      "id": 1869,
      "barcode": "B0058640123",
      "date": 1340631474946,
      "event": "DELETED"
    }
  ]
}
```

Here there are six results, but max criteria shortens the response.

The "event" field may be one of the following:

- CREATED: a new tape is added to the Active Circle system
- REMOVED: a tape is removed from the Active Circle system
- DELETED: the tape content will no longer be accessible

- `MOVEDOUTFROM_LIBRARY`: a tape is moved out of the library. Its accessibility is "OFFLINE"
- `MOVEDINTOLIBRARY`: a tape is inserted into the library. Its accessibility is "NEARLINE"

1.4 Available HTTP response status header

- 200 (OK) application/json
- 404 (Not Found)
- 401 (Unauthorized)
- 403 (Forbidden: the authentication is good, but the access is restricted to super users)

The PartialTranscode web service

Since version 4.3.3

Note

The transcoding feature is only available if you have the Active Media Explorer option installed. If you are interested by this feature, please contact the Activecircle Team.

This service allows you to list the transcoding features provided by Active Circle.

The partialTranscode request resource provides functionality to list available transcoding profiles.

A description of each web service method follows.

1. Get the list of partial transcode features

1.1. Request URL

```
GET /acapi/1.0/partialTranscode
```

1.2. Response body example

```
{
  "self": "https://server:port/acapi/1.0/partialTranscode?pretty=true",
  "shares":
  [
    "pr_on_tapes",
    "ame"
  ],
  "profiles":
  [
    {
      "avbr": "4194304",
      "codecs": "-c:v libx264 -b:v 4000k -preset fast -strict -2",
      "extension": "mp4",
      "id": "mp4-4m",
      "name": "MP4 4Mb/s"
    },
    {
      "avbr": "1048576",
      "codecs": "-c:v libx264 -b:v 1000k -preset fast -strict -2",
      "extension": "mp4",
      "id": "mp4-1m",
      "name": "MP4 1Mb/s"
    }
  ],
  "node": "node1"
}
```

Fields description:

Field	Description
shares	List of Active Circle shares on which the partial transcode feature is enabled.
node	The Active Circle node to be used to perform transcoding.
profiles	List of available transcoding profiles.
profiles id	List of available transcoding profiles.
profiles name	The profile name
profiles extension	The extension of the transcoded file.
profiles codecs	FFMPEG command line options including codecs options.
profiles avbr	Average bit-rate of the transcoding profile. Can be use to estimate the target file size.

1.3. Available HTTP response status header

- 200 (OK) application/json
- 401 (Unauthorized)

2. About the transcoding features

The transcoding feature can be used by restoring files with the **TransferJob** service.

2.1. Example: full and partial transcoding

```
POST //https://server:port/acapi/1.0/transferJob
```

POST data:

```
{
  "type": "RestoreJob",
  "node": "node1",
  "files": [
    {
      "dst": "ftp://fxp/home/nfo/testdir2/full_video_enc1m.mp4",
      "src": "/share/video.mxf",
      "encProfile": "mp4-1m"
    },
    {
      "dst": "ftp://fxp/home/nfo/testdir2/partial_video_enc4m.mp4",
      "src": "/share/video.mxf",
      "markin": "00-00-10-00",
      "duration": "00-00-01-00",
      "encProfile": "mp4-4m"
    }
  ]
}
```

Transcoding feature is activated by the addition of the key/value `"encProfile": "<profile.id>"`.

You can mix partial restore and transcoding features by adding "markin", "duration" fields.