infotel

Arcsys - ArcHP Option

ArcHP Option Guide

26.1.STS
March 12, 2026

Version: 26.1.STS

Publication date of the document: 2026-03-12

This document is intended for anyone who requires more information on the Arcsys product.

This document is written for all owners of a valid license.

The reader agrees to respect the confidential nature of this document.

Contact details:

France and Rest of World	***Germany***	***USA***
INFOTEL SA *Immeuble Le Valmy* *6/8/18 Avenue Léon Gaumont* *75020 Paris* *France*	*Insoft Software GmbH* *Sternstr. 9-11* *D-40479 Düsseldorf* *Deutschland*	*INFOTEL Corporation* *PO Box 47517 Florida* *33743 St Petersburg* *United States*
Tel: +33 (0)1 48 97 38 38	*Tel: +49 (0) 211 44 03 16-6*	*800 543 1982 – Toll-free telephone (US only)* *Tel: +1 727 343 5958*
https://techsupport.infotel.com https://infotel-software.com *<software@infotel.com>*	https://techsupport.infotel.com https://infotel-software.com *<software@insoft-infotel.com>*	https://techsupport.infotel.com https://infotel-software.com *<software@infotel.com>*

Table of Contents

Preface

1. Introduction

This document supplies detailed information for the use of ArcHP Option of the Arcsys product.

This option provides:

- scalability, failover and random load distribution with resource sharing in a single Arcsys site;
- “stateless” methods in Arcsys RMI, TCP/IP and SOAP API, to search records and perform synchronous retrievals. Parameters optimizing these methods performance are also included.

Note

The search and synchronous retrieval operations available in Arcsys REST API are by nature stateless. It is not necessary to have ArcHP Option to use stateless operations in Arcsys REST API, but the ArcHP Option is needed on the other provided features (scalability, failover...).

2. Restrictions

This option operates with the following restrictions:

- For ArcMover Tape Option: a tape library cannot be shared between several Arcsys Transfer Servers modules (cluster mode is not supported: no sharing of information or tape reservation).
- For ArcMover Disk: a file system can only be shared between several Arcsys Transfer Servers on Linux. Windows support is planned for a later date.
- For synchronizations: there cannot be several engines in cluster allowed to perform synchronizations; one single engine must have the synchronizations enabled.

3. Reference Documents

3.1. Concepts

Arcsys Presentation Manual: **Arcsys-presentation-26.1.0.STS-en.pdf**

Arcsys Functional Description Manual: **Arcsys-functional-description-26.1.0.STS-en.pdf**

3.2. Installing and Updating

Arcsys Prerequisites Manual: **Arcsys-requirements-26.1.0.STS-en.pdf**

Arcsys Installation Manual: **Arcsys-installation-26.1.0.STS-en.pdf**

3.3. Operations

Arcsys Administration Manual: **Arcsys-administration-26.1.0.STS-en.pdf**

Arcsys Errors Manual: **Arcsys-error-26.1.0.STS-en.pdf**

3.4. GUI

Arcsys Web Interface User Manual: **Arcsys-web-26.1.0.STS-en.pdf**

Interface Guide: **Arcsys-web-end-user-26.1.0.STS-en.pdf**

3.5. Development

Arcsys API Manual: **Arcsys-api-26.1.0.STS-en.pdf**

3.6. Option guides

ArcHP Option Guide: **Arcsys-option-archp-26.1.0.STS-en.pdf**

ArcREF Option Guide: **Arcsys-option-arcref-26.1.0.STS-en.pdf**

3.7. Optional modules

BatchReporting: **BatchReporting-UserGuide-26.1.0.STS-en.pdf**

ClassAssigner: **ClassAssigner-UserGuide-26.1.0.STS-en.pdf**

MetadataReplacement: **MetadataReplacement-UserGuide-26.1.0.STS-en.pdf**

StartRetentionDateAssigner: **StartRetentionDateAssigner-UserGuide-26.1.0.STS-en.pdf**

4. Symbols and Meanings

Note

Identifies information of particular interest

Important

Identifies important information

5. Definitions and Abbreviations

See the Glossary

1. When to useArcHP Option ?

1.1. Parallel Writing by Several Arcsys Transfer Servers on the Same Media

1.1.1. Without ArcHP Option

- Several Arcsys Transfer Servers can access the same file system in **read only** (for retrieval or synchronous retrieval, for the source zone of a migration or copy). Only one Arcsys Transfer Server can access a given file system in write mode (for archiving, for the destination zone of a migration or copy).
- Several Arcsys Transfer Servers cannot be associated to the same tape library.

1.1.2. With ArcHP Option

By activating the cluster mode:

- Several Arcsys Transfer Servers can access the same file system in read/write mode.
- Even with the option, several Arcsys Transfer Servers cannot be associated to the same tape library.

For more information, see page 4, « Implementing a Cluster in the Arcsys Transfer Server ».

1.2. Scalability / Failover on Arcsys Engines

1.2.1. Without ArcHP Option

There can only be one Arcsys Engine per site, which processes migration, recovery, and copy requests; there can only be one Arcsys Engine in total that processes synchronization requests.

1.2.2. With ArcHP Option

With the cluster mode on Arcsys Engines, there can be several archiving or retrieval Arcsys Engines on the same site that operate simultaneously, associated to the same Arcsys Application Agent.

If one Arcsys Engine fails to operate, another Arcsys Engine can take over the processing of subsequent archiving or retrieval requests, without requiring an operator intervention.

In cluster mode, several Arcsys Engines can create and process automatic requests (migration, recovery, copy, etc.) of the same type on a single site.

Note

Manifest synchronization processing is excluded from cluster mode; there cannot be several Arcsys Engines processing these requests simultaneously.

For more information, see page 8, « Implementing a Cluster in the Arcsys Engine ».

1.3. Scalability / Failover / Load Balancing on Arcsys Application Agents

1.3.1. Without ArcHP Option

The optional ArcIP module defines only one usable Arcsys Application Agent for archiving and retrieval requests.

1.3.2. With ArcHP Option

The optional ArcIP module can have a list of Arcsys Application Agents which it can use according to a defined strategy.

For more information, see the ArcIP module guide.

1.4. Scalability / Failover / Load Balancing on Arcsys Transfer Servers

1.4.1. Without ArcHP Option

- The Arcsys API (RMI, SOAP or REST), Arcsys Engine and Arcsys Web Agent can only use one Arcsys Transfer Server for their site for different requests. If the Arcsys Transfer Server does not respond, the requests for this site will fail.

1.4.2. With ArcHP Option

- The Arcsys API (RMI, SOAP or REST), Arcsys Engine and Arcsys Web Agent can be linked to a set of Arcsys Transfer Servers on the same site. If one of them does not respond, there is automatic load balancing to another Arcsys Transfer Server when processing the next request.

- The Arcsys Engine and Arcsys API (RMI, SOAP or REST) have the *TRANSFER_SERVER_STRATEGY* parameter, which enables you to determine the selection mechanism of the Arcsys Transfer Server for processing each request: random or first available.

For more information, see page 18, « Communication with the Arcsys Transfer Servers ».

1.5. Clusterization of the Arcsys REST APIs

1.5.1. Without ArcHP Option

- After having obtained a token on a given Arcsys REST API, all subsequent calls using this token (including the refresh token operation) must be performed on this same API.

1.5.2. With ArcHP Option

- Thanks to the persistence of the token in the database, a client might call a cluster of Arcsys REST APIs without having to "stick" the calls with a given token to a particular API. The load balancing of the client calls to the different Arcsys REST APIs is not the responsability of Arcsys yet, but of the client development and infrastructure. For example, you may set up a hardware load balancer that transparently dispatches all API calls to the Arcsys REST APIs.

2. Implementing Cluster Operation

2.1. Introduction

The ArcHP Option allows cluster operation of the three following modules:

- The Arcsys Transfer Server
- The Arcsys Engine
- The Arcsys REST API

2.2. Implementing a Cluster in the Arcsys Transfer Server

2.2.1. Cluster management parameters

The following parameters enable the Transfer Server to operate in cluster mode.

Important

Stage areas of the Transfer Servers must be separated. They must not be shared, as they are not managed by the clustering mode. Sharing the same folders would cause anomalies during the requests processing.

`ArcHP`: You must have this option to use these parameters.

Section of parameters taken into account only if: `OS` = `linux`

Parameter name in `transferServer.conf`	**clustering_enabled**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Enables the cluster behavior of Transfer Server Specifies whether you authorize operation of the Transfer Server in cluster mode. This mode authorizes in particular media sharing.  **Important** **All Transfer Servers of a given site that have `clustering_enabled` set to true are in the same cluster. The Transfer Servers of another site will be in another cluster: a cluster cannot be distributed over several sites.**
Shipped value	`false`

Parameter change assessment	After module reboot.
Required	False, this parameter is optional.

Parameter name in `transferServer.conf`	**cluster_port_number**
Parameter type	Positive integer
Description	Port number of the Transfer Server inside the cluster Port number of the Transfer Server on the cluster (TCP), for incoming and outcoming cluster messages. This port number is not mandatorily the same for all nodes of the cluster.
Shipped value	`25011`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.
Minimum value	1000
Maximum value	65535

Parameter name in `transferServer.conf`	**cluster_nodes_connect_timeout**
Parameter type	Number of seconds
Description	Timeout for trying to contact each node (started or not) of the cluster. When this timeout is reached, the node will not be able to start.
Shipped value	`30`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.
Minimum value	1

Parameter name in `transferServer.conf`	**cluster_nodes_ready_timeout**
Parameter type	Number of seconds
Description	Timeout for waiting validation for each started node of the cluster.

	When this timeout is reached, the node will not be able to start.
Shipped value	`15`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.
Minimum value	1

Parameter name in `transferServer.conf`	**cluster_synchronization_retry**
Parameter type	Positive integer
Description	Number of "retry" for cluster synchronization attempts Number of attempts to synchronize cluster resources when starting up the Transfer Server.
Shipped value	`5`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.

Parameter name in `transferServer.conf`	**cluster_synchronization_timewait_ms**
Parameter type	Number of milliseconds
Description	Time to wait between two "retry" for synchronization attempts Use this parameter to specify the waiting time in milliseconds between two synchronization attempts on a cluster node after a failure.
Shipped value	`2500`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.
Minimum value	1

Parameter name in `transferServer.conf`	**cluster_synchronization_time_processing**

Parameter type	Number of seconds
Description	Max time to wait when processing the cluster synchronization before getting an error Maximum working time between the connection to a node and the end of the synchronization step with this node.
Shipped value	`30`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.
Minimum value	1

Parameter name in `transferServer.conf`	**cluster_resource_get_timeout**
Parameter type	Number of seconds
Description	Timeout for getting a resource inside the cluster When this timeout is reached, the operation requesting the resource (for example, the archiving request) will fail
Shipped value	`60`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.
Minimum value	1

Parameter name in `transferServer.conf`	**cluster_submit_timeout**
Parameter type	Number of seconds
Description	Max time to wait for a wakeup message Maximal time the Arcsys Transfer Server waits for an acknowledgment message after submiting a message on the cluster.
Shipped value	`20`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.

Minimum value	1

Parameter name in `transferServer.conf`	**cluster_submit_retry**
Parameter type	Positive integer
Description	Number of "retry" to submit a wakeup message before getting an error Number of times the Arcsys Transfer Server tries again to submit a message on the cluster before getting an error.
Shipped value	`5`
Parameter taken into account only if:	`clustering_enabled` = `true`
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.

2.3. Implementing a Cluster in the Arcsys Engine

2.3.1. Cluster management parameters

The following parameters enable the Engine to operate in cluster mode.

`ArcHP`: You must have this option to use these parameters.

Parameter name in `ENGINE.properties`	**CLUSTER_ENABLED**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Enables the clustering of engines (multicast is used when no clustering configuration file is set) Specifies whether you authorize operation of the engine in cluster mode. This mode means you can have several engines on the same site that share a particular specialization (archiving, retrieval, migration, etc.) in the same time slots. The clustering mechanism is based on the JGroups library (see http://jgroups.org) which is, for instance, used in the JBoss application server. **Important** **All engines of a given site that have `CLUSTER_ENABLED` set to true are in the same cluster. The engines of another site will be in another cluster: a cluster cannot be distributed over several sites. Synchronization is not part of the cluster specializations. Synchronization is a process involving several sites.**

Shipped value	`false`
Option(s)	`ArcHP`: You must have this option to use this parameter.
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.

Parameter name in `ENGINE.properties`	**CLUSTER_MCAST_PORT**
Parameter type	Positive integer
Description	Port used for multicast when no clustering configuration file is set This parameter specifies a port that will be used for multicast communication among Engines of the same site. This value must therefore be the same for all Engines of a cluster. This parameter allows you to isolate cluster messages between different clusters, avoiding "discarded message from different cluster" messages in the log files.
Shipped value	`7600`
Parameter taken into account only if:	`CLUSTER_ENABLED` = `true`
Option(s)	`ArcHP`: You must have this option to use this parameter.
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.

Parameter name in `ENGINE.properties`	**CLUSTER_CONFIG_FILE**
Parameter type	File path
Description	Optional clustering configuration file (cf. JGroups documentation) Path of the cluster configuration file in JGroups format. The use of a configuration file allows you to use for instance TCP instead of UDP when IP multicast poses a problem (e.g. during WAN use): see an example on http://www.jgroups.org/manual/index.html#_tcp1.
Shipped value	**No shipped value.**
Parameter taken into account only if:	`CLUSTER_ENABLED` = `true`
Option(s)	`ArcHP`: You must have this option to use this parameter.
Parameter change assessment	After module reboot.
Required	False, this parameter is optional.

2.4. Implementing a Cluster in the Arcsys REST API

2.4.1. Cluster management parameters

The following parameters enable the Arcsys REST API to operate in cluster mode:

Parameter name	**JWT_JWS_KEY_FILE**
Parameter type	Absolute path of a file. Use Unix like separators ("/"), even on Windows.
Parameter change assessment	After module reboot.
Description	Location of the file containing the signature key for tokens; the key is randomly generated on startup if no file is specified. The signature key must correspond to a file containing 256 random bits and must be generated by `generate-key` script. **Important** **This parameter is mandatory for cluster mode: the file must be the same for all APIs involved in cluster mode.**
Shipped value	No shipped value.

Parameter name	**CLEAN_EXPIRED_TOKENS**
Parameter type	Boolean which can be either `true` or `false` .
Parameter change assessment	After module reboot.
Description	Flag to indicate if the REST API must delete expired tokens in database. In a cluster context, it is recommended to keep it to `true` on all APIs.
Shipped value	`true`

Parameter name	**TOKEN_CLEANER_PERIOD**
Parameter taken into account only if:	*`CLEAN_EXPIRED_TOKENS`*= `true`
Parameter type	Number of seconds.
Parameter change assessment	After module reboot.
Description	The time to wait between each deletion of expired tokens in database.
Shipped value	`1800` (30 min)

3. Cluster management: practical guide

3.1. Setting up a cluster

When setting up a cluster of Arcsys Transfer Servers, Arcsys Engine, Arcsys REST API, you may choose to include in the clusters all components of a site, or only some of them. The clusterization of all components of a site is recommended because it is easier to operate and monitor.

However, it might be appropriate to isolate some nodes from the cluster for various reasons, for example:

- Performance reason: a node out of the cluster will decrease the number of messages on the network
- Functional reason: a node has a functional role isolated from the other nodes (it performs only retrieval requests for example).

To set up cluster mode:

- Configure Arcsys Transfer Servers, Arcsys Engines and Arcsys REST APIs so they can operate in cluster mode, as described in the previous chapter.
- Before launching the Arcsys Transfer Servers in cluster mode, all the Arcsys Transfer Servers configured in cluster mode must be registered in the database.

Note

Arcsys Transfer Server is registered by launching `register_transferServer.sh` script. See Arcsys Administration Manual for more information about the use of the register script of the Arcsys Transfer Server.

- For the Arcsys Transfer Server cluster part, the configuration of media in ArcMover Disk must take into account the clusterization:
  - Each filesystem should be associated to at least two Arcsys Transfer Servers. This would allow failover and load balancing on the Arcsys Transfer Servers.
  - This association is done in the Arcsys Web Agent, in the screen of filesystem modification, in the “associated transfer servers” tab.
  - Those filesystems must be shared across the physical hosts where the Arcsys Transfer Servers are run.
  - All the Arcsys Transfer Servers that access a filesystem for archiving purposes must be configured in cluster mode, so the space left on the media remains

consistent. If a filesystem is only accessible for reading purposes (retrieval requests...) by the Arcsys Transfer Servers, it is not necessary that the attached Arcsys Transfer Servers run in cluster mode, as explained in the beginning of this chapter.

3.2. Clustering and segregation

3.2.1. How to choose an Arcsys Engine for a request

When a cluster of Arcsys Engines is set up, each Arcsys Engine of the cluster may load a request attached to an Arcsys Application Agent which is associated to the site of the engine through its parameter *PRIORITY_SITE_LIST*: there is no way to control which of these Arcsys Engines loads the request.

In other words, creating a request and attaching it to an Arcsys Application Agent will indirectly target all Arcsys Engines that are on the chosen site(s).

If the need is that only one precise Arcsys Engine must process a given request, there must be a site with this specific Arcsys Engine only, and the *PRIORITY_SITE_LIST* parameter must reference only this site. In this case, if the Arcsys Engine of the site cannot load the request, the request will stay `INITIALIZED` as long as the Arcsys Engine is not able to load it.

3.2.2. How to specialize nodes to functional tasks (archiving, retrieval...)

On Arcsys Engine level, the nodes specialization is possible by using *xxx_ENGINE* (`xxx` being a specialization such as ARCHIVE, RESTITUTE...) parameters. For example, you may have one or several Arcsys Engines that process only retrieval requests by setting *RESTITUTE_ENGINE* to `true` and the other specializations to `false`.

On the Arcsys Transfer Server level, the nodes specialization is possible through the specialization of the Arcsys Engines that communicate with them, by setting the *TRANSFER_SERVERS* and *DISTANT_TRANSFER_SERVERS* parameters of these specialized Arcsys Engines to the required Arcsys Transfer Servers.

On Arcsys REST API level, it is the responsibility of client calls to target specific nodes of Arcsys REST API if needed.

3.3. Monitoring the cluster

3.3.1. Web part

- The "Sites" screens list the sites and a column "Site state" shows the state of the Arcsys Transfer Servers on the site. A green tick means that all nodes of Arcsys

Transfer Servers are currently up; a red tick means that one node at least is currently down. This tick is updated in real-time by the Arcsys Web Agent when the screen is displayed.

- The "Agents" screens list the agents and a column "Active" shows the state of the Arcsys Application Agent: this state is regularly updated by the Arcsys Engines in the database.
- The "Transfer servers" screens list the Arcsys Transfer Servers, and from 5.2 version their respective cluster port number and cluster mode (enabled or not).
- The "Engines" screens list the Arcsys Engines. The specialization of the engines can also be checked.

3.3.2. Scripts part

- For the Arcsys Transfer Servers, the monitor script of the Arcsys Transfer Server can be used to determine which node is up. It also provides information about the resources managed by the cluster and their state (whether they are locked or not, and which node holds the lock where applicable) and other technical information on the cluster.

Note

This information can be provided to the support if there is a technical issue.

See Arcsys Administration Manual for more information about the use of the monitor script of the Arcsys Transfer Server.

- For the Arcsys Engines, the status script of the Arcsys Engines can be used to determine which node is up.

3.4. Start and stop issues

A single node of Arcsys Transfer Server, Arcsys Engine or Arcsys REST API can be stopped and restarted with no effect on the cluster. The cluster will continue to perform operations as long as a node is up. There is no restriction regarding the order in which the nodes are stopped.

However, the operations being run on a node while a script is triggering its shutdown will fail, as there is no system that will allow another node to handle those operations.

More specifically for Arcsys REST API, it is possible to obtain a token on a first API, to stop this API, and continue to use this token on another API which shares the same key through the `JWT_JWS_KEY_FILE`.

For the stop and start of Arcsys Transfer Server, Arcsys Engine or Arcsys REST API following a change on the configuration, see page 15, « Updating the configuration of a component » or page 17, « Updating the version of all nodes ».

3.5. Error handling

3.5.1. Errors occurring while accessing a media

3.5.1.1. When starting Arcsys Transfer Server

When an error occurs while accessing a media during the inventory phase that occurs when starting Arcsys Transfer Server, the node sets the media offline. An anomaly "`inaccessible material`" is reported in the database, as well as an action "`set offline`" appears. Both can be seen in the filesystem detail screen of the Arcsys Web Agent, in the "`Anomaly`" and "`Attached actions`" tabs.

All Arcsys Transfer Servers that were already started and were accessing the media will still be able to access the media.

The Arcsys Engine will continue sending requests for the zone to one of the Arcsys Transfer Servers managing the zone. If those requests are handled by the Arcsys Transfer Server that cannot access to the media, they may fail. For example if there is no other media associated to the zone, or if the request was a retrieving operation. However, if the requests are handled by any of the other Arcsys Transfer Servers managing the zone and that can access to the media, the operation is successful.

When another Arcsys Transfer Server that has access to the media starts, the media is set as online in the database (which is visible in Arcsys Web Agent). The status "`online` / `offline`" in the database only reflects the status for the most recently started Arcsys Transfer Server.

3.5.1.2. During the processing of a request

When an error occurs in the access to a media while processing a request, an anomaly is reported in the database. It can be seen in the filesystem detail screen of the Arcsys Web Agent, in the anomaly tab.

The request ends in error. The filesystem remains online and unlocked. There is no impact on the other nodes that can access to the media.

Subsequent requests may still be sent for the zone and node hosting the media that is in error status. The media is still used when processing any request.

For ArcMover Disk medias, if the media access is reset, the upcoming requests will succeed, without restarting the Arcsys Transfer Server.

3.5.2. Errors showing that a request is loaded by several Arcsys Engines

When a request ends in error and shows in the detail of the request that a same status appears several times (`VALIDATED`, `WORKING`, `PROCEEDED`...), it is likely that the request has been loaded by several Arcsys Engines, which is not possible if they work in cluster mode. Check that all these engines have the parameter *`cluster_enabled`* set to true.

3.5.3. Restarting a node after a problem (hang, crash)

If an Arcsys Transfer Server, an Arcsys Engine or an Arcsys REST API node crashes or hangs, you may choose between two possibilities:

- You need to restart the node as soon as possible (but with the risk the node crashes or hangs again if an anomaly is encountered). Collect the following information to send to the Arcsys technical support:
  - Date/time of the crash or hang, suspected action involved in the crash or hang (archiving, retrieval) if known, architecture diagram of the infrastructure
  - Core file of the binary or JVM that encountered a crash (if available)
  - Log files of all the nodes
  - If the Arcsys Transfer Server is involved, result of the `monitor_transferServer` command on all the running nodes
  - If the Arcsys REST API is involved, log files of the clients of the Arcsys REST API (if available)
  - Configuration files of all the nodes
  - Optional: screenshots in Arcsys Web Agent of the nodes details (Sites, Arcsys Transfer Servers, Arcsys Engines)
- Your architecture tolerates the fact to not restart immediately the node. In this case, do not restart the node (or keep it blocked) and wait for further instruction from the Arcsys technical support.

3.6. Updating the configuration of a component

Depending on the parameters updated, none, one or all nodes may need to be restarted. Here follow the possible cases.

3.6.1. Modifications that can be done without restarting

For all components, the modifications that can be done without restarting do not still need a restart in cluster mode.

3.6.2. Modifications that require restarting all the nodes of the clusters

- For Arcsys Transfer Server: any modification of the configuration of the cluster, port number, site code, component code, component host or external component host requires to restart all the nodes.

  For example, after a modification of the port number of a Arcsys Transfer Server, all of the Arcsys Transfer Servers need to be restarted. Arcsys Transfer Server updates its port number when starting. At start-up, it gets a list of all the other Arcsys Transfer Servers in the cluster, with their host name and port number. Those information are not updated later; this is why all of the Arcsys Transfer Servers need to be restarted when the port number of a node is changed.

- For the Arcsys Engine: there is no modification which, performed on a single node, would require restarting all the nodes.

- For the Arcsys REST API: modifying the key file shared between the nodes (parameter *JWT_JWS_KEY_FILE*) requires restarting all the nodes.

3.6.3. Modifications that require restarting a single node

These modifications require to restart only the associated node, and not the other nodes:

- For the Arcsys Transfer Server:
  - Any modification of the configuration of ArcMover Tape Option
  - Any modification in the configuration of an external media manager
  - Any modification in Arcsys Transfer Server parameters that are not in the categories described in « Modifications that can be done without restarting » or « Modifications that require restarting all the nodes of the clusters »
- For the Arcsys Engine: any modification (requiring to restart Arcsys Engine). For example:
  - Changing the cluster status of a node (parameter *CLUSTER_ENABLED*) can be performed without restarting the other nodes.

- Changing the specialization of a node (archiving, retrieval) can be performed without restarting the other nodes.

- For the Arcsys REST API: any modification, except the key file *JWT_JWS_KEY_FILE*: if this key file is changed on a node, the token is not shared with the other nodes.

3.7. Updating the version of all nodes

When updating Arcsys Transfer Servers, Arcsys Engines or Arcsys REST APIs, all nodes need to be stopped, as there is no guarantee that the nodes can communicate with each other from one version to another.

4. Communication with the Arcsys Transfer Servers

4.1. Introduction

The Arcsys API (RMI, SOAP or REST), Arcsys Engine and Arcsys Web Agent modules communicate with the Arcsys Transfer Servers. There are parameters that allow you to specify the communication modes and possible load balancing mechanisms.

4.2. Arcsys API (RMI, SOAP or REST) Communication with the Arcsys Transfer Server

The main configuration file of the module (`ARCSYS_RMI_API.properties`) contains parameters defining the communication with the Arcsys Transfer Server.

4.2.1. General parameters for communication with the Arcsys Transfer Server

Parameter name	**TRANSFER_SERVERS**
Parameter type	Characters string list, separated by a comma (",").
Parameter change assessment	After module reboot.
Description	List of the transfer server codes with which the API module must communicate for synchronous retrieval and creating/deleting file systems. For synchronous retrieval, the request is tried successively on the transfer servers of the list, until it succeeds, in the order given by *TRANSFER_SERVER_STRATEGY*. For creating/deleting file systems, one of the reachable transfer servers is contacted. This parameter is not mandatory. If left blank, all transfer servers of the site will be taken into consideration, in the order given by *TRANSFER_SERVER_STRATEGY*.
Shipped value	No shipped value.

Parameter name	**HEALTH_CHECK_TRANSFER_SERVERS_PROC_LOOPSLEEPTIME**
Parameter type	Number of milliseconds.
Parameter change assessment	After module reboot.
Description	Control interval for checking the presence of transfer servers, for failover or load balancing management.
Shipped value	No shipped value.

Parameter name	**TRANSFER_SERVER_STRATEGY**
Parameter type	Characters string.
Parameter change assessment	After module reboot.
Description	Definition of the access mechanism to the transfer servers. Possible values are `FIRST_AVAILABLE` and `RANDOM`. `FIRST_AVAILABLE` means that the request is always sent to the first transfer servers, then to the second if the request fails, and so on. `RANDOM` operates in the same way, but on each request, a transfer server is chosen randomly from the list of transfer servers available. If ArcHP Option is used, you should set this parameter to `RANDOM`.
Shipped value	`RANDOM`

4.3. Arcsys Engine Communication with the Arcsys Transfer Server

The main configuration file of the module (`ENGINE.properties`) contains parameters defining the communication with the Arcsys Transfer Server.

4.3.1. General parameters for communication with the Arcsys Transfer Server

Parameter name	**TRANSFER_SERVERS**
Parameter type	Characters string list, separated by a comma (",").
Parameter change assessment	After module reboot.
Description	List of the transfer server codes with which the engine can communicate for all requests (archiving, retrieval, migration, etc.). If the engine is specialized for an operation, the associated transfer servers can also be specialized for these operations. If a code is not specified, in this case, all transfer servers of the site will be taken into consideration.
Shipped value	Empty (all transfer servers of the engine site)

Parameter name	**DISTANT_TRANSFER_SERVERS**
Parameter type	Characters string list, separated by a comma (",").
Parameter change assessment	After module reboot.
Description	This parameter is used optionally to "filter" the remote transfer servers targeted (for example, when archiving with a remote copy, to choose which remote transfer servers will perform the archiving). It contains a list of remote transfer server codes. By default, any transfer server can be used.
Shipped value	Empty (all transfer servers of the distant site)
Only for Arcsys option	This parameter can only be used with the option: ArcHP Option

Parameter name	**HEALTH_CHECK_TRANSFER_SERVERS_PROC_LOOPSLEEPTIME**
Parameter type	Number of milliseconds.
Parameter change assessment	After module reboot.
Description	Control interval for checking the presence of transfer servers, for failover or load balancing management.
Shipped value	No shipped value.

Parameter name	**TRANSFER_SERVER_STRATEGY**
Parameter type	Characters string.
Parameter change assessment	After module reboot.
Description	Definition of the access mechanism to the transfer servers. Possible values are `FIRST_AVAILABLE` and `RANDOM`. `FIRST_AVAILABLE` means that the request is always sent to the first transfer servers, then to the second if the request fails, and so on. `RANDOM` operates in the same way, but on each request, a transfer server is chosen randomly from the list of transfer servers available. If ArcHP Option is used, you should set this parameter to `RANDOM`.
Shipped value	`RANDOM`

4.4. Arcsys Web Agent Communication with the Arcsys Transfer Server

The main configuration file of the module (`WEBAGENT_configuration.properties`) contains parameters defining the communication with the Arcsys Transfer Server.

4.4.1. General parameters for communication with the Arcsys Transfer Server

Parameter name	**TRANSFER_SERVERS**
Parameter type	Characters string list, separated by a comma (",").
Parameter change assessment	After module reboot.
Description	List of the transfer server codes with which the module must communicate for synchronous retrieval and creating/deleting file systems. For synchronous retrieval, the request is tried successively on the transfer servers of the list (in the order they are listed), until it succeeds. For creating/deleting file systems, one of the reachable transfer servers is contacted. This parameter is not mandatory. If left blank, all transfer servers of the site will be taken into consideration (without specific order).
Shipped value	No shipped value.

Parameter name	**MILLISEC_TRANSFER_SERVER_COMMUNICATION_TIMEOUT**
Parameter type	Number of milliseconds.
Parameter change assessment	After module reboot.
Description	Timeout after which communication without a reply to the transfer server triggers an error.
Shipped value	`1000`

5. ArcHP Option APIs

5.1. Purpose of ArcHP APIs

ArcHP Option allows calling methods or operations in the context of a **cluster of APIs**, but not on all available APIs:

- The use of ArcHP Option is **not compatible** with RMI APIs.
- The use of ArcHP Option is compatible with a **specific set of dedicated stateless** SOAP APIs.
- The use of ArcHP Option is compatible with **all operations** of the Arcsys REST API, providing some conditions in specific cases.

5.2. ArcHP Option SOAP Methods

5.2.1. Introduction

The deprecated Arcsys RMI, TCP/IP and SOAP API provided methods based on a "stateful" model, as authentication must be performed initially by the `connect` method and the authentication token is sent to the methods called subsequently.

This operating mode presents issues regarding load distribution when implementing a HTTP load balancer between the client and various Arcsys RMI, TCP/IP and SOAP API instances.

The ArcHP API provides a set of "stateless" methods compatible with the use of a load balancer. Each call of those methods requires authentication.

The ArcHP API consists of the three following methods:

- `statelessSearchArchives` - to perform a search
- `statelessSearchArchivesCount` - to get the number of results returned by a search
- `statelessOnlineRestore` - to perform a synchronous document retrieval

5.2.2. Access to the Methods

The *`WS_API_ARCHP_SERVICE_NAME`* parameter defines the service name of the ARCHP API. `ArcHPWS` is the default service name.

The *`WS_API_PORT_NUMBER`* parameter in the `ARCSYS_WS_API.properties` file sets the port number. Note that this is the same port as the internal SOAP API.

5.2.3. `statelessSearchArchives` and `statelessSearchArchivesCount` methods

The `statelessSearchArchives` method performs record searches by sending the lots or objects corresponding to the search criteria.

The `statelessSearchArchivesCount` method does the same but returns the number of results instead of the actual results.

5.2.3.1. Parameters

- `LotSearchParameters` search parameter: This object contains the criteria (keywords, record state, etc.) corresponding to the search. For more information, see the javadoc of the `LotSearchParameters` class.
- In the case of `statelessSearchArchives`, optional parameter (`long` type) `startIndexLine`: The index of the first element to send; this allows you to iterate by range on a large number of results, for example. By default (set to `null` not specified), the list sent starts from the beginning.
- In the case of `statelessSearchArchives`, optional parameter (`integer` type) `nbLinesToLoad`: The number of elements to send; this allows you to iterate by range on a large number of results, for example. By default (set to `null` or not specified), the list sends all the elements.

Important

If the search criteria are not restrictive enough and if a maximum number is not specified, an `OutOfMemory` error can be generated in the API.

- `String` type user name: application account user name to make the call.
- `String` type password: unencrypted password of the application account to make the call.

5.2.3.2. Restrictions

To search in all the repositories, the lot-level criteria list must be empty.

To search in all the collections of a single repository, the lot-level keyword levels must be populated.

The boolean operators only work for the criteria on the same keyword.

5.2.4. `statelessOnlineRestore` Method

This method is used to retrieve an Arcsys object synchronously (called synchronous retrieval).

5.2.4.1. Parameters

- `long` type lot identifier: identifier of the lot in which the object to retrieve is found.
- `long` type object identifier: identifier of the object to retrieve.
- `boolean` Validation of the hash of the envelopes: to specify whether you want to validate the hash of envelopes (set to `false` to improve performance).
- `String` type user name: user name of the application account to be used to make the call.
- `String` type password: unencrypted password of the application account to be used to make the call.

5.2.4.2. Samples of Requests and SOAP Responses

Here is an example of the SOAP exchanges used to perform a stateless synchronous retrieva, with the following SOAP message:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ns1="101" xmlns:ns2="http://archp.wsapi.infotel.com/">
<SOAP-ENV:Body>
  <ns2:statelessOnlineRestore xsi:type="ns1:stdClass">
    <lotId>106</lotId>
    <objectId>10704</objectId>
    <checkTarIntegrity>false</checkTarIntegrity>
    <user>user</user>
    <password>password0</password>
  </ns2:statelessOnlineRestore>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Using the lot identifier and the object identifier, you can modify this XML request and save this XML in a file; here it is called `request.xml`. In this example, the server name `example.com` and the default port and service name are used. The following command can then be run:

```
curl --header "Content-Type: text/xml;charset=UTF-8" --header "SOAPAction:" --data
 @request.xml http://example.com:50060/ArcHPWS?wsdl
```

The following response is then received:

```
--uuid:0af02e25-4efe-41c5-ae28-17e69b78b7a6

Content-Type: application/xop+xml; charset=UTF-8; type="text/xml"
```

```
Content-Transfer-Encoding: binary

Content-ID: <root.message@cxf.apache.org>

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<ns5:statelessOnlineRestoreResponse xmlns:internal="http://internal.components.infotel.com/
" xmlns:search="http://search.components.infotel.com/" xmlns:tools="http://tools.common.
aro.infotel.com/" xmlns:ns5="http://archp.wsapi.infotel.com/">
<file><xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include"
 href="cid:c0cf8730-366b-4026-8ae1-e08015845b08-1985@cxf.apache.org"/></file>
</ns5:statelessOnlineRestoreResponse>
</soap:Body>
</soap:Envelope>

--uuid:0af02e25-4efe-41c5-ae28-17e69b78b7a6

Content-Type: application/pdf

Content-Transfer-Encoding: binary

Content-ID: <c0cf8730-366b-4026-8ae1-e08015845b08-1985@cxf.apache.org>

Content-Disposition: attachment;name="/opt/infotel/ArchivingProduct/ArcsysBuffer/depotIP/
servicetest/VERS/droptest/lot1463754759650168720313876 97/DEPOT/fichier95.pdf"
[Binary content of the file]

--uuid:0af02e25-4efe-41c5-ae28-17e69b78b7a6–
```

5.3. ArcHP Option in the Arcsys REST API

5.3.1. Providing clustering thanks to the persistence of the refresh token

With ArcHP Option, you can use a cluster of Arcsys REST APIs for your client developments.

This is possible thanks to the persistence of the refresh token mechanism.

The persistence of the refresh token is a mechanism done by storing it in the Arcsys database. This allows to eventually set up a REST APIs cluster. The REST APIs will then be able to share the same refresh token.

To set up a cluster of REST APIs, it is necessary to fill the parameter *JWT_JWS_KEY_FILE* in the `arcsys-rest.properties` configuration files of all the Arcsys REST APIs that need to be used in cluster mode. This parameter must indicate the path towards a secret key that can be generated with the script `generate_key.sh` delivered in Arcsys Engine (please refer to Arcsys Installation Manual to see how to generate secret keys).

Important

All the REST API that are used in cluster mode must share the same secret key.

In order to avoid unnecessary storage in the database, a cleaning mechanism for expired refresh tokens can be activated via the parameter *`CLEAN_EXPIRED_TOKENS`* that should be set to `true` for each REST API you use.

The settings about the persistence and the cleaning of expired tokens are detailed in Arcsys Administration Manual.

6. Arcsys API (RMI, SOAP or REST) Performance

6.1. Performance-related API parameters

Parameter name	**ONLINE_RESTORE_REQUEST**
Parameter type	Boolean which can be either `true` or `false` .
Parameter change assessment	After module reboot.
Description	`true` if an synchronous retrieval request must be created in the relational database during synchronous retrieval by the RMI API or web service, otherwise `false`. If ArcHP Option is used, you should set this parameter to `false`.
Shipped value	`true`

Parameter name	**TEST_FILE_AVAILABILITY_FOR_ONLINE_RESTORE**
Parameter type	Boolean which can be either `true` or `false` .
Parameter change assessment	After module reboot.
Description	Specifies whether the API must test the status of the presence of files to synchronously retrieve on the transfer servers before sending the request. When you know with certainty, via the configuration, that the files to display are still present, set this parameter to `false`. For example, if the files are on file systems and the "Authorize direct access for synchronous retrieval" option is activated on the file systems.
Shipped value	`true`
Only for Arcsys option	This parameter can only be used with the option: ArcHP Option

Parameter name	**USE_CONSTANT_TYPE_FOR_ALL_ONLINE_RESTORE**
Parameter type	Boolean which can be either `true` or `false` .
Parameter change assessment	After module reboot.
Description	Specifies whether the Arcsys type (native, AFP, ZIP, etc.) of files to synchronously retrieve with this API is still the same. This prevents loading in the database on each display.
Shipped value	`false`
Only for Arcsys option	This parameter can only be used with the option: ArcHP Option

Parameter name	**NATIVE_TYPE_FOR_ALL_ONLINE_RESTORE**
Parameter taken into account only if:	*USE_CONSTANT_TYPE_FOR_ALL_ONLINE_RESTORE*= `true`
Parameter type	Characters string.
Parameter change assessment	After module reboot.
Description	Specifies the Arcsys constant type to be used by this API for the synchronous retrievals. Possible values: `AFP_NOP`, `AFP_TLE`, `ZIP_CSV`, `NOT_NATIVE`.
Shipped value	No shipped value.
Only for Arcsys option	This parameter can only be used with the option: ArcHP Option

Parameter name	**SIZE_FOR_CACHE_ONLINE_RESTORE_MOVER_REFERENCE**
Parameter type	Positive integer.
Parameter change assessment	After module reboot.
Description	Number of envelope cache entries for the synchronous retrieval (0, by default). If 0, there is no cache. If > 0, a cache limited by this value is implemented (everything depends on the size of the memory given to the API). In this way, the characteristics of envelopes containing the object to retrieve are not reloaded systematically if an object of this envelope has already been viewed. The oldest entries are not contained in the cache.
Shipped value	`0`
Only for Arcsys option	This parameter can only be used with the option: ArcHP Option

Parameter name	**LOAD_ONLINE_STATUS_FOR_SEARCH_ARCHIVES**
Parameter type	Boolean which can be either `true` or `false` .
Parameter change assessment	After module reboot.
Description	Specifies during a call to the `searchArchives` method whether you should load the "online" status of each record from the Arcsys Database. Setting it to `false` will improve search performance.
Shipped value	`true`
Only for Arcsys option	This parameter can only be used with the option: ArcHP Option

Parameter name	**LOAD_ARCHIVED_STATUS_FOR_SEARCH_ARCHIVES**
Parameter type	Boolean which can be either `true` or `false` .
Parameter change assessment	After module reboot.
Description	Specifies during a call to the `searchArchives` method whether you should load the "archived" status of each record from the Arcsys Database. Setting it to `false` will improve search performance.
Shipped value	`true`
Only for Arcsys option	This parameter can only be used with the option: ArcHP Option

Parameter name	**ALWAYS_CACHE_LDAP_USER**
Parameter type	Boolean which can be either `true` or `false` .
Parameter change assessment	After module reboot.
Description	Specifies whether an LDAP user cache should be implemented. If this is the case, in the next connection with the same user in the API, it does not reconnect to the LDAP, but simply checks the hash of the password transmitted. If ArcHP Option is used, you should set this parameter to `true`.
Shipped value	`false`

Glossary

Access Zone

An access zone is an independent entity within Arcsys that defines a controlled network area from which resources can be accessed. These entities can then be attached to permissions (at the repository, collection, lot, or class level) to restrict or grant access based on the client's IP address when authenticating to the Arcsys REST API, the Arcsys Web Agent or ArcWeb Module.

API *(Application Programming Interface)*

The APIs provided by Arcsys enable the product holder to fully customize a new application or user interface according to the specific ergonomic needs of their use case. Arcsys exposes several types of APIs:

- REST APIs are the recommended interface. They offer broad coverage of Arcsys's functionalities, including administration, operations, archiving, search, and archive retrieval.
- Legacy APIs based on RMI and SOAP protocols are still available for compatibility purposes but are deprecated and should no longer be used in new developments.

Application Agent

There are two different types of agents at archiving level: application interface agents and user interface agents. An **application agent** can archive all the objects specific to an application (files, RDBMS table records, etc.), whereas a **web agent** performs both administration functions and manual archiving functions initiated by the user.

Archiving By Reference

Archiving by reference is a method in which data remains in its original storage location when added to an archive system, and the system generates references and metadata entries for the files. Eventually, the files are transferred to the archive system's defined storage using the copy and migration mechanism.

Archive Restitution

Archive restitution is the return and transfer of archived documents to their originator, or to a duly appointed person or organization. An Archive Restitution is in Arcsys an Archive Retrieval operation that ends with a Destruction. An Archive restitution operation can only be created through the appropriate operation in the REST API, or by using ArcEP module.
See Also **Archive Retrieval**, **Destruction**.

Archive Retrieval

Archive retrieval is an operation that makes a copy of a record available to a record requester. This term takes precedence over the term *restore*, which has another meaning at archiving level (restore in the sense of handing back the documents to the organization that created them or to its representatives, then destroying them). Archive retrieval can be

complete (misleadingly called a "complete retrieval") or partial (*Partial Archive Retrieval*, misleadingly called a "partial retrieval").
See Also **Archive Restitution**.

Arcsys
ERM published by Infotel. Arcsys refers to both the Arcsys Core product and all of its connectors and options.

Arcsys Connector
An Arcsys connector is an operational module generally requiring an additional license used to interface with an external software package (ECM, ERP, Mail) for archiving and/or archive retrieval to and from Arcsys.

Arcsys Core
The Arcsys Core represents all "essential" Arcsys modules, which are: Arcsys Database, the Arcsys RMI, TCP/IP and SOAP API, the Arcsys REST API, the Arcsys Transfer Server, the Arcsys Transfer Service, the Arcsys Engine, the Arcsys Web Agent, the Arcsys Application Agent, the Arcsys Auto-Archive Agent, the ArcFF format control module, the CopyRequestManager, the Arcsys standard Clients, the ArcsysFsComparator File systems comparator, the ArcProofFolder Proof Folder module and the ArcsysBatchs batch module.
See Also **Arcsys**.

Arcsys Engine
Central archiving platform on which synchronous and asynchronous archiving, indexing and retrieval processes operate. The engine can spread threads over multiple processors. This guarantees dialogue and traceability between the agents that are associated to it.

Arcsys Option
Arcsys options are added to the Arcsys Core for additional functionalities. They do not necessarily require an additional architectural module. They may be subject to a separate license. The main options are:

- ArcAFP Option (AFP format management)
- ArcMover Tape Option (media manager managing file systems and tape libraries)
- ArcIP (record ingestion)
- ArcEP (record extractor)
- ArcPAK Option (record compression on ArcMover and native ingestion of compressed files)
- ArcRFT Option (full text search)
- ArcSIGN Option (internal digital signature generation) and ArcVERIF (external digital signature verification)

- ArcCrypt Option (encryption of data at rest)
- ArcCFN (digital vault)
- ArcREF Option (record ingestion by reference)
- ArcMOVS3 Option (media manager allowing to archive and retrieve data on any Cloud media compatible with the Amazon S3 REST API)

Attestation policy

An attestation policy allows to define which type of attestation can be generated as well as a set of parameters concerning their generation.

Classification Scheme

A classification scheme in archiving and digital preservation refers to an organized framework for categorizing records and archival materials based on a hierarchical structure. It facilitates systematic retrieval, management, and preservation of information. In the context of Arcsys, the classification scheme is defined as the structural entity that contains a hierarchy of classes. These classes are used for organizing archives and records and for implementing specific archival policies such as retention schedules and format management. Within Arcsys, a classification scheme is linked to a specific repository, providing an organizational backbone for multiple collections. It also serves as a navigational tool for end users, enabling them to explore archives through the hierarchical structure of classes, alongside navigation by repository and collection.

Collection

Set of rules that a record must comply with. The collection is defined via the Web agent or Arcsys API, and comprises information contained in the relational database tables. A collection always refers to two rules: one concerning the **storage policy** and one relating to the **indexing mask**. A collection is assigned to the record on the initial record request.
See Also **Storage policy**, **Indexing mask**.

Deletion

MOREQ2010 provides the following definition for this concept: the act of deleting data from the relational database so that no trace remains. Generally speaking, an entity can only be deleted if is not used in a stored record. Otherwise, it can only be destroyed and not deleted, thus leaving a residual entity.
See Also **Destruction**.

Destruction

Irreversible action that deletes the documents by applying disposal criteria. It can be associated with the retention of residual information in the relational database.

Disposal

Outcome of archived documents when the retention period ends, i.e. generally, destruction or transfer.
See Also **Destruction**, **Transfer**.

Disposal due date (or retention end date)
Scheduled end of retention date.

Disposal Hold
Arcsys can be used to place a "disposal hold" on one or more lots archived in the application. This prevents certain sensitive operations, such as transitioning the lots to end-of-life status or migrating them to a different storage medium. All other operations remain authorized. The disposal hold guarantees that no irreversible change affecting the archival integrity or status of the lot can occur while the hold is active.

Electronic Attestation
Document produced to attest that an action or an electronic transaction has occurred.

Envelope
Arcsys groups documents stored in the system in envelopes, either created by Arcsys during the archiving process (in this case, files in TAR format), or created prior to Arcsys processing by the user or third-party processes (*native envelopes* in AFP or ZIP format, for example). The representation of an envelope in the Arcsys Database is called a **logical envelope**. Its technical implementation is also called *MoverReference*. Last but not least, the representation of information of where the envelope is physically stored in the optional ArcMover module is called *MoverMedia*.

Event
In Arcsys, a retention schedule can associate the start of record retention with an event with a known or unknown date. This event, created in an Arcsys repository, can thus be attached to a number of different retention schedules.
See Also **Retention schedule**.

Feature preview
A Preview status on a feature enables early access to non-production features, allowing users to explore and provide feedback for improvement.

Features in Preview status should not be used in production environment, as they are not fully implemented yet.

Fixity
The quality of a document that has not been subject to intentional or accidental destruction, alteration or modification.

Format policy
A format policy is used to define a set of rules concerning format checks for a given file type. These rules are used to specify the action that will be performed, the expected results of these actions, as well as the error cases, triggering archiving failure.

Hash value
Also called an "integrity certificate" in cryptography, the hash value is the digest of a message which guarantees a practically unique result that is impossible to reverse calculate.

The most commonly used algorithms are MD5 (128 bits), SHA-1 (160 bits), SHA256 (256 bits) and SHA512 (512 bits). Arcsys includes a module that is capable of dynamically calling several algorithms. The choice of an algorithm type is valid for all archived objects within the same Arcsys product version; compatibility with algorithms from the previous version is guaranteed. The associated term *hash function* is also used.

Indexing mask

As is the case with the storage policy, an indexing mask is a rule that is referenced by a collection. An indexing mask can be referenced by several collections. An indexing mask refers to the use of a set of Keyword = Value pairs. The keyword component is set to make sense in a specific business application (e.g. Accounting Day, Department, Account No., Account Holder, etc.). The value component can be either unrestricted, or restricted to a set of acceptable values (e.g. A, B or C), or in date format, or restricted by an input mask. Some pairs are defined as mandatory whereas others may be optional.

An application which uses an indexing mask through a collection must supply all Keyword=Value pairs as they are defined using this mask. Any indexing-related errors lead to the record being rejected for conformity. This record is then added to the list of records with errors.

The indexing mask is defined by an administrator via the Arcsys interface or APIs. It is comprised of a set of keywords.
See Also **Keyword**.

Journal

A journal is an XML file which contains a list of PREMIS events.

Keyword

Component of an indexing mask. The keyword in particular defines the type of metadata (date, string, numerical, controlled) and its input mask, for example.
See Also **Indexing mask**.

Lot

Arcsys can consolidate several different objects that form a functional set in a client application in the same physical record. It is comprised of different types of objects: files, databases, or any other type of object managed by Arcsys. It is possible to retrieve the entire lot or one of the objects contained in the lot. The MOREQ2010 record is translated in Arcsys implementation by a lot; the lot, as opposed to a MOREQ2010 record, can represent documents that are not yet archived.

Lot enrichment

The process of adding new objects to an existing archive.

Manifest

The manifest is an XML file that defines precisely the content of a record. The manifest contains: metadata associated with the record, structure metadata, a description of the physical files of the record(s) that follow, the object-by-object content of the record, object

formats, object names, their size, hash value, the algorithm used to calculate the hash value, etc. The manifest is a type of complete ID card for the record.

The manifest is always written on the storage media and precedes the record that it describes. This process is used to automatically describe storage media (irrespective of the medium). With this system, users can understand media content and metadata without installing the software that generated the records.

Object
The object is a basic archived unit that can be retrieved via Arcsys. Lots contain one or more objects. An object can be: a file, a directory, a table, a relational table, etc. The MOREQ2010 component is implemented by this object concept; the object, as opposed to a MOREQ2010 component, can represent a document that has not yet been archived.

Online
Storage level, which must be disk type, that makes records permanently available within an extremely reduced time period.

Permissions
Permissions refer to the user profiles or groups authorized to access documents or sets of documents archived in the system.

Program exit
Place in the standard workflow for picking up and executing specific code.
See Also **Workflow**.

Proof folder
A proof folder consists of a record, a proof slip, and, where appropriate, additional items (common signature or timestamp response, for example) that are used, by demonstrating the fixity and the authenticity of a document, for admission as proof. A proof slip can be generated using Arcsys Web Agent, ArcWeb Module, or Arcsys REST API. A proof folder can only be generated using ArcEP.

Record
A record is an evidential document that is deemed sufficiently important by the creator to be managed by an ERM that will manage its life cycle (retention, disposal, etc.). A record represents an archived lot. A record is archived via a *record request*. Archiving a document *creates a record*.

Relational database (or referential)
Essential component of the system, it contains all the data (excluding archived data) used by Arcsys for its operation. It includes logical entities called "repositories" (see definition).

Repository
Logical entity used to segment and isolate information within Arcsys. Each repository has its own storage profiles, indexing masks, keywords, and configuration, all fully independent from those of other repositories.

A repository also defines its own permission scope.
See Also **Storage policy**, **Indexing mask**, **Keyword**, **Permissions**.

Restore (or retrieval)
This term is used misleadingly in Arcsys to refer to the concept of archive retrieval. It is not accepted in archiving terminology as to mean transfer and then destruction.
See Also **Archive Retrieval**.

Retention and disposal schedule
This comprises all the rules defining the record retention period for a company or an organization, according to risks of unavailability and information system access requirements. It specifies the disposal after these time periods.
See Also **Retention schedule**.

Retention period
A duration expressed in days, months or years of object retention. The retention period is a concept used notably in MOREQ2010.

Retention schedule
A retention schedule defines the start and the end of the retention of records that are attached to it, either directly or through their class.

Retention start date
Date from which a retention period must be taken into account. The retention start date is a concept used notably in MOREQ2010.

Security
An ERMS requirement that involves including documents whose use (confidentiality, risk of exposure) and/or fixity (non modification of content, non-alteration of media) should be closely monitored.

Slug
A tenant slug is a short, URL-friendly identifier used to represent a tenant in a stable and readable way within application URLs. It uniquely identifies the tenant using only lowercase letters, digits, and hyphens.
See Also **Tenant**.

Storage policy
A storage policy is a rule that is referenced by a collection. The policy dictates the storage media which are successively implemented to hold a record, as well as the retention period for each media. The storage policy is defined through the graphical interface. Applications or business users use it indirectly through the reference to a collection. A storage policy can be changed over time to reflect new retention periods or new storage media. The policy covers storage units by logical pool.

Storage pool
Logical storage pool, characterized in particular by its time period (e.g. 10 years). The storage policy assigns a "zone" to a "policy".

Storage zone
The storage zone is a logical entity representing a physical storage space (e.g. set of file systems, tape libraries, cloud storage).

Synchronous retrieval
Archive retrieval that takes place in the form of a direct retrieval of a document (for direct viewing or downloading) in a Web browser.
See Also **Archive Retrieval**.

Tenant
A tenant is a distinct logical environment within a shared Arcsys installation, fully isolated in terms of customization while running on a common technical platform. It enables each client or organization to benefit from a fully personalized interface and functional setup without impacting others.

Tenants do not isolate or partition the data stored in Arcsys.
See Also **Slug**.

Time stamping
Time stamping is a technique used to associate a document with a certain date in reference to a given and recognized time system. The date set in this way is an essential element for document authentication. Time stamping can be performed internally or by a third-party time stamp.

Tracking
Result of continuously creating, capturing and maintaining information about the movement and use of the system and objects (ISO 15489-1:2001, 3.19).

Transfer
In an archival sense, this operation sends an archived object to another IT system. Once the transfer is performed, the object can be removed from the ERMS as needed. In OAIS terminology, a transfer represents more specifically the physical transmission of a record or a set of records by a service supplying an archive service. Not to be confused with the transfer of data in the purely technical sense, as with the Arcsys Transfer Server module.

Transit Zone
Entity logically connecting an application agent and a directory, along with additional configuration.

Workflow
A set of operations carried out from the beginning to the end of a process. In Arcsys, this refers to all actions carried out on archives and objects, either directly or indirectly, in the case of archives, from their pre-archiving or preparation to their removal from the system (after they have reached end-of-life). There are other workflows in Arcsys besides the archiving workflow, which are more administration-oriented. Customized workflow involves the use of at least one drop-off point to carry out customer processing.

Registered Trademarks

Firefox is a registered trademark of the Mozilla Foundation.

Linux is a registered trademark of Linus Torvalds.

UNIX is a registered trademark of the Open Group.

Microsoft Windows, Windows NT, Windows Server, SQL Server, Internet Explorer are registered trademarks of Microsoft Corporation in the United States and/or other countries.

SAP is a registered trademark of SAP AG in Germany and other countries.

MySQL is a registered trademark of Oracle and/or its subsidiaries. MariaDB is a registered trademark of Monty Program Ab.

Java is a registered trademark of Oracle and/or its subsidiaries in the United States and other countries.

Infotel is a registered trademark of Infotel SA.

All trademarks mentioned are the property of their respective owners.

Infotel Technical Support

https://techsupport.infotel.com