infotel

Arcsys ClassAssigner

User Guide

25.3.STS
October 30, 2025

infotel.com

Version: 25.3.STS

Publication date of the document: 2025-10-30

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.

This document is part of the Arcsys software package, a product designed and developed by Infotel. All rights are reserved for Infotel.

Contact details:

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

Table of Contents

List of Tables

Preface

1. Introduction

This document provides detailed information on how to use Arcsys ClassAssigner.

2. Reference Documents

2.1. Installing and Updating

Arcsys Prerequisites Manual: **Arcsys-requirements-25.3.STS-en.pdf**

Arcsys Installation Manual: **Arcsys-installation-25.3.STS-en.pdf**

3. Symbols and Meanings

Note

Identifies information of particular interest

Important

Identifies important information

4. Definitions and Abbreviations

See the page 13, « Glossary ».

1. Getting started with ClassAssigner

Arcsys offers the ability to define a logical organization of lots through classes. This allows to define policies, such as format, attestations or retention schedule policies related to classes rather than to indexing and storage profiles.

ClassAssigner is a batch program, using Arcsys REST API, that helps exploit and take benefit from the classes related features by providing an easy way to assign a given class to all the lots of a collection, or to a list of lots contained in an input file.

2. Installing ClassAssigner

1. Prerequisites

The prerequisites for the module align with the prerequisites of Arcsys as detailed in Arcsys Prerequisites Manual.

2. Installing the module

To install the module, please refer to the Arcsys Installation Manual.

3. Finalizing installation

3.1. Configuring ClassAssigner

After extracting ClassAssigner, the configuration files must be set with the values that depend on your environment.

Some parameters are used to contact the Arcsys REST API. To set those parameters, please refer to page 9, « etc/http-client.properties file » chapter.

3.2. How to encrypt passwords in ClassAssigner configuration files

Note

The encrypt module is installed in the root folder of Arcsys Engine or present in the Arcsys installer.

The module can be used to encrypt any value of ClassAssigner (and not only passwords). When a parameter value beginning with `ENC(` is encountered, ClassAssigner automatically decrypts it.

3.2.1. Generate key file

In order to encrypt the data, it is necessary to generate first a key file with the `generate-key` script present in the Arcsys installer. This script uses a cryptographically secured pseudo-random number generator to create a random 256 bit symmetric key. The key file is used by Arcsys to decrypt the encrypted passwords in the configuration files of different modules. This key file may be shared across modules (as it must be different each time, as required by your security policy).

The following GNU/Linux command generates a file (filename: `secret-key`) in the folder `myFolder`:

```
$ ./generate-key.sh -s -d myFolder
Secret key generated in file myFolder/secret-key
```

3.2.2. Encrypt a value

The passwords must be encrypted with the generated symmetric key file. All properties can be encrypted too.

The command for using the *encrypt* utility depends on the platform chosen:

- Windows: `bin\encrypt.cmd`
- GNU/Linux: `bin/encrypt.sh`

The following GNU/Linux command encrypts the password `myPassword` with the `secret-key` file:

```
$ ./encrypt.sh -k myFolder/secret-key -e
Please enter the value to be encrypted.

Please enter the value again.

ENC(fRO8S6BlhZY5Ci3Rn8ZJgOXWkwD5jYgaYa3QWBsV52Q=)
```

The blank line corresponds to the entry of the string to encrypt.

Note

If the option `-k myFolder/secret-key` is not specified, the default key stored in `etc/secret-key` is used.

The encrypted values (including "`ENC(`" and "`)`") of the password must then be copied in the configuration file.

The encrypted properties are *salted*, in the cryptographic sense, and associated with the key present in the `secret-key` file. If this key is renewed, all the properties encrypted using a previous key must be re-encrypted manually.

3.2.3. After installation

The key file must be saved in the `etc/` directory of ClassAssigner installation directory. Its name must be `secret-key`.

The secret key file may be installed in another directory and/or under another name. In this case, you have to manually update the `bin/overenv.sh` (or .cmd) script. The

SECRET_KEY_FILE parameter must be set to specify the path and name of the secret key file. For example:

```
SECRET_KEY_FILE=/opt/Infotel/class-assigner/secret-key
```

4. Display a quick help

After installing ClassAssigner, to display a quick help on a Linux console, the command line is:

```
./class-assigner.sh -h
```

and on a Windows console:

```
class-assigne.cmd -h
```

3. Updating ClassAssigner

To update the module, please refer to the Arcsys Installation Manual.

4. Using ClassAssigner

1. Launching ClassAssigner

The script is used as follows:

```
./class-assigner.sh <mandatory parameters> [<optional parameters>]
```

or, on Windows platform:

```
class-assigner.cmd <mandatory parameters> [<optional parameters>]
```

1.1. Mandatory parameters

The following parameters are mandatory:

- *`--repositoryCode, -rc`*: Repository code where the collection *`[--collectionCode, -cc]`* is located or where the lots of the file *`[--lotListFile, -llf]`* are located. This parameter is an alphanumeric string.

- *`--collectionCode, -cc`*: The collection code where the lots are located. The two parameters *`--collectionCode`* and *`--lotListFile`* are mutually exclusive : if you specify *`--collectionCode`* you cannot specify *`--lotListFile`* and reciprocally. It is an alphanumeric string.

- *`--lotListFile, -llf`*: Use this parameter when you want to assign a class to a given list of lots. This list must be contained in a file. The file must have the following format :

  - Each line is a lot identifier on a separate line.

  - Each lot can belong to any collection in the repository.

  - The identifiers must be in ascending order.

  It is recommended to save this file in the `etc/` directory of ClassAssigner installation directory.

- *`--class, -cs`*: The full path of the class to assign to the lots. If a class is already assigned to any of the lots, ClassAssigner adds a warning in the log file.

Note

Their absence in the command line will raise an exception with immediate abort of the batch execution.

1.2. Optional parameters

The following parameters are optional parameters that can be set to modify the default behaviour of ClassAssigner:

- *`--lotIdMin, -lm`*: It is the lowest identifier of lot that will be processed. This identifier does not necessarily have to match a lot identifier in the collection or in the lots input file.

- *`--maxLots=, -ml`*: It is the maximum number of lots to process. The default value is `1000`. When the value is `0` the number of lots to be processed is unlimited. It includes also the lots which already have a class.

- *`--maxRuntime, -mr`*: It is the total time, in minutes, allowed to process the lots. When the execution time is over, the process stops and stores the ID of the last lot processed in a text file (`last-handled-lot`). If this parameter is missing, the execution time is unlimited.

1.3. Other parameters

The following parameters can be specified. In this case, they replace any other mandatory and optional parameters:

- *`--help,-h`*: Displays the list of accepted parameters and their meaning.

- *`--version,-v`*: Displays the version of the ClassAssigner.

The following parameter can be specified in addition to the other mandatory and optional parameters:

`--quiet, -q`: Displays only the version, then end when finished.

2. Command-line examples

2.1. With collection code

In this following example, we pass to the batch the 3 mandatory parameters (in bold) and an optional one (in italic).

```
./class-assigner.sh --repositoryCode="Telecom" --collectionCode="2023 Invoices" --class=
FamilyName --maxLots=500
```

With the mandatory parameters, we operate on the "2023 Invoices" collection of the "Telecom" repository, in such a way that the "FamilyName" class is assigned to all the lots of the "2023 Invoices" collection.

With the first optional parameter *`--maxLots=500`*, we specify that the class "FamilyName" is assigned to only the first 500 lots. The ID of the last lot processed is stored in the text file `last-handled-lot`.

2.2. With an input file

In this following example, we pass to the batch the 3 mandatory parameters (in bold) and an optional one (in italic).

```
./class-assigner.sh --repositoryCode="Telecom" --lotListFile="etc\inputFileWithLots.txt" --
class=FamilyName --maxRuntime=1
```

We operate on the "Telecom" repository. The "FamilyName" class is assigned to all lots that have their ID contained in the `inputFileWithLots.txt` file.

With the first optional parameter *`--maxRuntime=1`*, 1 minute is allowed to process the lots. When the execution time is over, the process stops and the ID of the last lot processed is stored in the text file `last-handled-lot`.

3. Automatic resumption of an interrupted process

When proceeding class assigner, ClassAssigner commits the changes progressively. Therefore, if the process is interrupted for any reason (power outage, network failure, or other reason), re-executing ClassAssigner will continue to proceed the following values.

Important

The modifications being carried out immediately, it is necessary to pay attention to the use of this batch.

4. Exit codes

4.1. Process completed successfully (Exit code: 0)

If the process ends successfully, the exit code is 0 and the batch displays the following message: ***Process finished with exit code 0***.

4.2. Process ended in error (Exit code: 1)

If the process cannot be executed, or if it is interrupted by an error, the exit code is 1 and the batch displays the following message: ***Process finished with exit code 1***.

5. Module description

The ClassAssigner module is installed in a directory called `class-assigner`.

1. Directory structure

Type	Name	Description
📄	`class-assigner.sh` (or .cmd)	ClassAssigner launch script.
📁	`bin`	Directory containing the executables for the installation platform.
📁	`etc`	Directory containing the module configuration files.
📄	`http-client.properties`	File containing the parameters of the logs.
📄	`log4j2.xml`	File containing the parameters of the Arcsys REST API.
📁	`lib`	Directory containing the Java libraries (JAR) for ClassAssigner.

Table 5.1. Content of the `class-assigner` directory

2. etc/http-client.properties file

2.1. HTTP connection settings

Arcsys uses the HTTP(S) protocol to connect to the Arcsys REST API. The following elements of the "`http-client.properties`" file define the connection to this API.

Property name	**Description**	**Default value**
`base.url`	Base URL, formed like: `<protocol>://<domain_name or ip_address>[:<port_number>][/<uri>]` `protocol` may have the value `http` or `https` . SSL will automatically be used when the protocol is https.	
`connect.timeout.ms`	This parameter allows to determine the timeout in milliseconds until a connection is established. For infinite timeout, set this parameter to `0`.	`1000`

Property name	Description	Default value
	A negative value is interpreted as undefined (system default).	
`request.timeout.ms`	This parameter allows to determine the timeout in milliseconds used when requesting a connection from the connection manager. For infinite timeout, set this parameter to `0`. A negative value is interpreted as undefined (system default).	`1000`
`socket.timeout.ms`	This parameter allows to define the socket timeout in milliseconds, which is the timeout for waiting for data or, put differently, a maximum period inactivity between two consecutive data packets. For infinite timeout, set this parameter to `0`. A negative value is interpreted as undefined (system default).	`300000`
`response.page.size`	This parameter corresponds to the page size of the HTTP response for requests getting list of entities. **Note** The higher the value is, the fewer HTTP calls will be, but the higher the value of `request.timeout.ms` delay must be set (as the messages sent from the server to the client are heavier).	`1000`
`oauth.token.url`	URL to get a OAuth2 token. URL, formed like: `<protocol>://<domain_name or ip_address>[:<port_number>][/<uri>]`	
`oauth.authorization.grant.type`	This parameter allows to configure the OAuth2 authorization grant type. Possible values are: `authorization-code`, `implicit`, `password`, `client-credentials`	`password`
`oauth.client.id`	This parameter allows to configure the OAuth2 client identifier.	
`oauth.client.secret`	This parameter allows to configure the OAuth2 client secret password. The value must be an encrypted characters string. To encrypt a clear	

Property name	Description	Default value
	characters string, use the `encrypt` encryption utility provided in the Arcsys installer. The encrypted characters string will be surrounded by `ENC(...)`.	
oauth.user.name	This parameter allows to configure the OAuth2 user name.	
oauth.user.password	This parameter allows to configure the OAuth2 user password. The value must be an encrypted characters string. To encrypt a clear characters string, use the `encrypt` encryption utility provided in the Arcsys installer. The encrypted characters string will be surrounded by `ENC(...)`.	
ssl.truststore.file	This parameter allows to configure the full path of the trust store file. In SSL mode this parameter is mandatory.	
ssl.truststore.type	This parameter allows to configure the type of trust store. Possible values are: `jks`, `pkcs12`.	`jks`
ssl.truststore.password	This parameter allows to configure the password of the trust store file. The value must be an encrypted characters string. To encrypt a clear characters string, use the `encrypt` encryption utility provided in the Arcsys installer. The encrypted characters string will be surrounded by `ENC(...)`.	
ssl.keystore.file	This parameter allows to configure the full path of the key store file. This parameter is only considered in SSL mode and is optional.	
ssl.keystore.type	This parameter allows to configure the type of key store. Possible values are: `jks`, `pkcs12`.	`jks`
ssl.keystore.password	This parameter allows to configure the password of the key store file. The value must be an encrypted characters string. To encrypt a clear characters string, use the `encrypt` encryption utility provided in the Arcsys installer. The encrypted characters string will be surrounded by `ENC(...)`.	

3. etc/log4j2.xml file

This file contains the settings for the log files. The logs are generated with the Apache library log4j.

The parameters of this file depend on the information desired in the logs.
For any changes, consult the standard documentation of Log4J. See: https://logging.apache.org/log4j/2.x/

Glossary

API *(Application Programming Interface)*

The APIs supplied by Arcsys are used by the product holder to completely customize a new Application or User interface to meet the precise ergonomic needs of a given application. A distinction must be made between:

- low-level APIs available in Arcsys Core, in RMI or SOAP protocol, providing access to most operations performed by Arcsys (administration, operation, archiving, searches, archive retrieval);
- and high-level APIs, available in the ArcBWS option. Web services in REST protocol provide access to a number of the most-frequently used methods.

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.

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**.

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**.

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, that is Arcsys Database, the Arcsys RMI, TCP/IP and SOAP API, the Arcsys Transfer Server, the Arcsys Engine, the Arcsys Application Agent, the Arcsys Transfer Service, the Arcsys Web Agent, the Arcsys Auto-Archive Agent, the ArcFF format control module.
See Also **Arcsys**.

Arcsys Engine

The 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)
- ArcBWS (REST web services)
- ArcMover Tape Option (media manager managing file systems and robots in SCSI or ACSLS)
- 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)

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 represents a hierarchical classification of activities in an organization used to classify archived objects and link them in this way to their creation context. A classification scheme is made up of categories organized in a hierarchy over a number of levels.

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***(or metadata template)*.

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

This is an 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

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

The scheduled end of retention date, retention end date

This is the date at which the last retention date is reached. This is a concept used notably in MOREQ2010.

Disposal Hold

Arcsys can be used to "hold" one or more lots archived in the application. This puts a "hold" on the status of the lots by preventing any operation that could modify this status. It freezes the life cycle and prevents any modification on the archive. Furthermore, the user is guaranteed that when the disposal hold is active, the lot will remain in the same state as it was at the moment it was put on hold.

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 physical information stored in the envelope 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**.

Export

An export procedure is used to retrieve the ERM object with a view to transferring to another system and if appropriate deleting it from the ERM. The export must include the object, its information stored in the relational database, its traces and ideally its management rules.

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, this 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*(or metadata template)*

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 metadata element definitions.

Fixity

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

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.

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. This 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.

Metadata element definition (or keyword)

Component of an indexing mask. We use the term "metadata element definition" rather than the term "keyword" as it is closer to MOREQ2010. The metadata element definition in particular defines the type of metadata (date, string, digital, controlled) and its input mask, for example.
See Also **Indexing mask***(or metadata template)*.

MOREQ2010

MOREQ2010 (Modular Requirements for Records Systems) is a standard made available by the European Commission (DLM Forum Foundation) for electronic archiving. As Arcsys pre-dates MOREQ2010, it is taken into account in Arcsys Core via an add-on that is used to progressively include MOREQ2010 entities (aggregate, classification scheme, etc.), on top of the Arcsys entities (repository, collection, lot).

Nearline

Storage level managed through a media manager (XBSA interface). This storage level can be used with or without online storage. It creates copies on the storage media supported by the chosen media manager (disks, tapes, optical disks, etc.). This storage level is optional. All types of online storage, nearline disks or nearline cartridges support periodical content verification functions, continuity of the formats used and the portability of fixity when technical or planned migrations are performed via the collections.

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. Arcsys also allows use of a Nearline storage level (see definition).

Permissions

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

Proof folder

This is comprised of a record, a proof slip, and additional items where required (functional traces, for example) that are used, by demonstrating the fixity and the authenticity of a document, for admission as proof.

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 *(previously called "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

This is a logical entity in the Arcsys Database. The company can define as many repositories as it wants, either to define a test set, to isolate an application, or for any other reason. These repositories are entirely independent of each other. They all have their own pattern and all have the same structure.

Restore*(correct term: 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. This 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. This 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.

Storage pool

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

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 zone

The storage zone is a logical entity representing a physical storage space (e.g. set of file systems, robot, zone NetBackup, TSM, Cloud or ArcMover).

Synchronous retrieval

This is an archive retrieval that takes place in the form of a direct retrieval of a document (for direct viewing or downloading) in a Web browser. This term takes priority over the term "online viewing" or, misleadingly, "online restore".
See Also **Archive Retrieval**.

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

This is the 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.

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

infotel.com